# Automatic Gloss Finding for a Knowledge Base using Ontological Constraints

Bhavana Dalvi
Carnegie Mellon University
Pittsburgh, PA 15213
bbd@cs.cmu.edu

Einat Minkov
University of Haifa
Haifa, Israel, 31905
einatm@is.haifa.ac.il

Partha P. Talukdar
Indian Institute of Science
Bangalore, India 560012
ppt@serc.iisc.in

William W. Cohen
Carnegie Mellon University
Pittsburgh, PA 15213
wcohen@cs.cmu.edu

## ABSTRACT

While there has been much research on automatically constructing structured Knowledge Bases (KBs), most of it has focused on generating facts to populate a KB. However, a useful KB must go beyond facts. For example, glosses (short natural language definitions) have been found to be very useful in tasks such as Word Sense Disambiguation. However, the important problem of Automatic Gloss Finding, i.e., assigning glosses to entities in an initially gloss-free KB, is relatively unexplored. We address that gap in this paper. In particular, we propose GLOFIN, a hierarchical semi-supervised learning algorithm for this problem which makes effective use of limited amounts of supervision and available ontological constraints. To the best of our knowledge, GLOFIN is the first system for this task.

Through extensive experiments on real-world datasets, we demonstrate GLOFIN's effectiveness. It is encouraging to see that GLOFIN outperforms other state-of-the-art SSL algorithms, especially in low supervision settings. We also demonstrate GLOFIN's robustness to noise through experiments on a wide variety of KBs, ranging from user contributed (e.g., Freebase) to automatically constructed (e.g., NELL). To facilitate further research in this area, we have made the datasets and code used in this paper publicly available.

**Categories and Subject Descriptors:** I.2.6[Artificial Intelligence]: Learning - Knowledge acquisition

**Keywords:** Gloss Finding; Hierarchical Learning; Web Mining.

## 1. INTRODUCTION

Automatic construction of knowledge bases (KBs) has attracted much attention over the past years. Knowledge bases provide a structured representation of entities and the relationships between them, which is key in semantic processing tasks such as concept tagging, disambiguation and normalization. Freebase [6] and DBPedia [28] are well-known examples of KBs that represent broad-domain world knowledge, which are constructed and maintained collaboratively. Much work has been done on development of information extraction systems to produce such KBs: e.g., the Never-Ending-Language-Learning (NELL) [7], TextRunner [45], and PROSPERA [33] are targeted at extracting facts at Web scale.

While facts are obviously essential for a KB, a useful KB must contain more than facts. Many widely-used KBs, including Freebase and WordNet, also include *glosses*—i.e., natural language definitions of the entities in the KB. A large KB may contain glosses for each of the many entities it contains. In manually-constructed KBs like WordNet [29], glosses are typically provided when the KBs are constructed by human experts; however, in many automatically generated KBs, like NELL [30] or YAGO [39], there are no glosses, or only a few. For instance, YAGO supports glosses, but only a small fraction of entities (68.9K out of 10M) have glosses assigned to them. Since manually adding definitions for a large number of entities is infeasible, glosses must be added automatically. In this paper, we focus on the problem of augmenting an existing KB with gloss information. To the best of our knowledge, this is the first attempt of enriching automatically-constructed KBs with glosses.

To illustrate the problem, Figure 1 (top part) describes the structure of the NELL KB, as an example. As shown, the entities represented in NELL are organized in a hierarchy, in which nodes are linked with *is-a* (subset) relations. General semantic categories, like *Fruit* and *Company*, are represented by nodes that are linked to higher level categories in the hierarchy–*Food* and *Organization*, respectively. Concrete entity nodes, such as *Microsoft* and *Google*, are linked to their category–*Company*. In addition, entities are associated in NELL with their lexical aliases; e.g., 'Microsoft' and 'MS' are both aliases of *Microsoft*. Notably, lexical aliases are often ambiguous, e.g., the name 'Apple' refers to either *Fruit:Apple* or *Company:Apple*. While this representation of NELL KB provides valuable structured semantic knowledge, it does not contain glosses. Ideally, we would like to associate glosses to each entity node. E.g., *Company:Apple* may be described as "Apple, formerly Apple Inc, is an American company headquartered in Cupertino...".

In addition to helping users understand the semantics (or intended semantics) of an entity, glosses are used in many technical tasks. In information retrieval, several recent approaches to query expansion make use of glosses [10, 13] to improve the performance of ad-hoc information retrieval, based on a semantic, entity-rich representation of queries and documents. More generally, in the *entity linking* (EL) task [21, 8], named entity mentions in text are mapped onto canonicalized nodes in the KB, thus disambiguating the named entity mentions. Classical approaches to EL make heavy


*WSDM'15,* February 2–6, 2015, Shanghai, China.

http://dx.doi.org/10.1145/2684822.2685288.

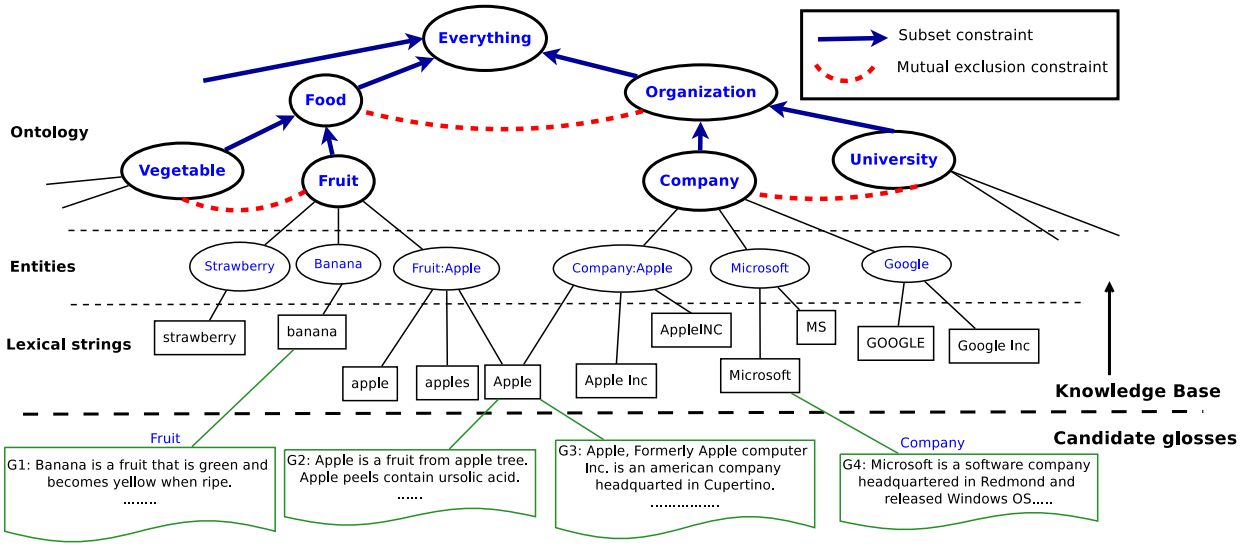


**Figure 1: Graph constructed by GLOFIN for the gloss finding problem using lexical matches and ontological constraints (e.g., mutual exclusion, subsumption etc).**

use of glosses: typically, an entity mention and surrounding context are treated as a query against a repository of glosses, with cosine similarity or word overlap [24] between the query and the node descriptions used for scoring, and the highest-scoring gloss being used to indicate the matching entity. For example, given a mention of 'Apple' in the context of 'Cupertino', a better match is to be expected against the gloss of *Company:Apple* (Fig. 1), compared with the alternative, *Fruit:Apple*. Having glosses for KB entities can also provide a basis for enriching entities with additional structured knowledge i.e. in asserting or refuting uncertain relations.

One potential way of producing glosses might be to construct sentences out of the facts already stored in the KB: e.g., if the entity node *Michael Jordan* is linked to the category node *Professor* and related to the entity *UC/Berkeley* via the relationship *employedBy*, then it is possible to create a gloss that contains the sentence "Michael Jordan is a professor working at UC/Berkeley". However, many KB entities have few known relationships, resulting in short and uninformative descriptions. Hence, rather than treating the task as a natural-language generation task, we consider an alternative method: we collect large numbers of definitional sentences, and attempt to match these existing sentences to KB entities. While the experiments described in this paper focus on definitions collected from DBPedia, our approach is general, and could be applied to definitions harvested from any dictionary or glossary.

In short, the gloss finding task addressed in this paper corresponds to matching potential glosses with the respective KB nodes. While some matches are obvious, many are ambiguous. While ambiguous matches are a problem in many other alignment tasks (e.g., the EL task described above)[1], this task is unusual in that it is an *asymmetric* resource alignment, where a collection of lexical glosses, which contain no structure information, is aligned against a structural KB, which contains no textual descriptions.

We define and address this task using semi-supervised learning (SSL) techniques. We rely on several simple intuitions. First, it has already been observed [39, 34] that entity ambiguity is often resolved given its semantic category. Hence, rather than solve a named entity linking problem, we solve an entity categorization problem and map glosses to entity categories. Second, we use the unambiguous glosses as labeled data for this entity categorization task. However, such labeled data is limited and noisy. We therefore propose **Gloss Finder (GLOFIN)**, a semi-supervised Expectation Maximization (EM) method that learns effective classifiers given little labeled data and large amount of unlabeled data. GLOFIN uses constrained optimization, taking into account the hierarchy of semantic categories present in the knowledge base. Specifically, GLOFIN incorporates known constraints between the given semantic classes, including subset relationships (e.g., "Mammal" is subset of class "Animal"), and mutual exclusion constraints (e.g., "Mammal" is mutually exclusive with "Automobile").

### Contributions of this paper

- We formulate an approach to the important problem of Automatic Gloss Finding for Knowledge Bases (KB), i.e., automatically identifying glosses (short definitional sentences) for entities in a KB. Even though glosses have been found to be very useful in other related areas (e.g., Word Sense Disambiguation), the task of adding glosses to entities in a gloss-free KB is relatively unexplored – we address that gap in this paper.
- We propose GLOFIN for the gloss finding problem. To the best of our knowledge, this is the first such system for this task. GLOFIN is a semi-supervised algorithm which also makes use of available ontological constraints. Through extensive experiments on real-world datasets, we demonstrate GLOFIN's effectiveness. In particular, it is encouraging to find that GLOFIN outperforms even other state-of-the-art semi-supervised baselines, especially in limited supervision settings.
- We demonstrate GLOFIN's robustness to KB quality through experiments on KBs, ranging from hand-curated KBs (e.g., Freebase) to automatically constructed ones (e.g., NELL).
- To facilitate further research on this topic, we have made the gloss finding datasets and GLOFIN code publicly available[2].

[1]Some methods for semantic enrichment of text map ambiguous words to multiple concepts, rather than perform alignment [18]

[2]Dataset and code available to download at `http://rtw.ml.cmu.edu/wk/WebSets/glossFinding_wsdm_2015_online/index.html`

*Outline*

The rest of the paper is organized as follows. We describe the related work in Section 2. We then present our proposed gloss finding algorithms in Section 3. Datasets and experimental methodology are presented in Section 4. Experimental results are presented in Section 5, followed by the conclusions and future work.

# 2. RELATED WORK

Glosses are an important resource for word meanings, which have been used by word sense disambiguation (WSD) [35] algorithms for decades. For example, the popular Lesk method [24] and its variants [22, 3] infer a word's sense by measuring the overlap between available context words and the glosses of candidate senses. Traditionally, WordNet [17] has been used as the main resource of word senses and respective glosses. Several recent works have investigated the problem of automatic gloss extraction, so as to improve coverage in specific knowledge domains. Duan and Yates [14] constructed sense descriptions for selected keywords given unlabeled documents in the biomedical domain, and used these descriptions to perform WSD. Others [16, 5] constructed domain-specific glossaries using definitional sentences extracted from the Web, and successfully used these glossaries for domain specific WSD. We share their motivation for obtaining glosses, however rather than generate these glosses, our goal is to associate available glosses with an existing KB.

In this work we match glosses against the respective entity nodes in the KB. This task is closely related to Entity Linking (EL), an entity-focused variant of the WSD problem [31], where named entity mentions in a given text are linked with the respective nodes in a KB. Advanced EL methods consider multiple entity mentions in a document collectively in order to enrich the contextual information that is modeled per entity [19, 42]. Such methods are ineffective for short text fragments, like search queries, or glosses. In fact, we face a 'chicken and egg' problem, as most EL methods assume that KB entities have glosses, whereas we do not [20, 25]. Here, we are instead interested in enriching a KB with glosses, so as to support entity linking and other downstream applications, such as ad-hoc information retrieval [10].

Various works have addressed the problem of resource alignment, but considered different settings. The Yago ontology [39], for example, unifies Wikipedia and WordNet using heuristics that consider the structure of both resources. Ponzetto and Navigli [38] mapped Wikipedia articles onto WordNet synsets, using the textual descriptions of entity pairs. We rather consider an asymmetric setting, where the KB includes a semantic structure but lacks textual descriptions, and the aligned glosses are not assumed to be organized in any fashion. Similarly, a recent work [37] addressed the alignment of an arbitrary pair of lexical resources, including machine-readable dictionaries with no structure. They proposed to induce a network structure for dictionary resources; however textual similarity remains an important component in their approach. Importantly, having enriched a given KB with glosses using our proposed framework will facilitate its alignment and integration with other KBs [9].

In this work, we address the gloss finding problem using a semi-supervised framework. In particular, we consider label propagation (LP), where gloss nodes are associated with candidate entity nodes based on their relatedness in a joint graph. Talukdar [41] discuss the use of weakly supervised label propagation methods in information extraction and integration. Previous work applied LP to perform WSD [46], and the graph modeled words in parallel documents in two languages, where word ambiguity in one language was resolved based on graph relatedness to their candidate translation in the other language, and vice versa. The use of graph-based methods is highly prevalent in WSD. In particular, random walks over a semantic graph, like WordNet, have been shown to yield state-of-the-art results [38, 1]. Notably, linking nodes in the WordNet graph based on lexical gloss overlaps has been shown to contribute significantly to the performance of graph-based WSD [1].

The GOLFIN algorithm that is proposed in this paper is inspired by the EM framework that has been employed by Dalvi et al. [12, 11] for the task of exploratory learning (semisupervised learning in the presence of unanticipated classes) and hierarchical semi-supervised learning. The methods proposed in this paper are different in the sense that we incorporate a class hierarchy into the model. Further, the divide and conquer method proposed by Dalvi et al. [11] was limited to tree structured class hierarchy, whereas we propose a mixed integer programming based optimization method that can deal with any class hierarchy defined by sets of subset and mutual exclusion constraints. Interestingly, another recent work has used mixed integer linear programming techniques to encode ontological constraints [34]–they assign multiple plausible KB categories to 'emerging' entities, which are not yet represented in the KB, and consider mutual exclusion constraints as a post-processing step, so as to output a consistent set of category assignments. In contrast, we apply subset as well as mutual exclusion constraints, within an iterative EM algorithm, using semi-supervised learning. Finally, while the application of earlier variants of the EM-based framework focused on the named entity categorization task, we use the proposed methods for addressing the novel task of generating glosses for an existing KB.

# 3. AUTOMATIC GLOSS FINDING

In this section, we first go through the background information, and then describe approaches to the Gloss Finding problem, including GLOFIN, our proposed method. Before getting into the details of methods, in Section 3.2, we first describe an entity-gloss candidate generation stage whose output is used by all the methods considered in the paper. After that, in Section 3.3 we propose our first attempt at the gloss finding problem, using Label Propagation (LP). However, through experiments on real datasets as we will see in Section 5, we find that such LP-based methods are not always effective on the gloss finding problem. To overcome this limitation, in Section 3.4 we present our proposed method GLOFIN.

## 3.1 Preliminaries

Here we briefly present the relevant terminologies and the problem statement for the gloss finding task.

### *3.1.1 Knowledge Base (KB)*

In this paper, we consider a KB as a directed graph consisting of the following three types of nodes and relations among them. Part of a KB is shown in the top half of Figure 1.

- **Category nodes**: Each category node corresponds to a semantic type. For example, in Figure 1, *University* and *Vegetable* are examples of category nodes. Category nodes are inter-linked through the *subCategoryOf* relation, resulting in the overall ontology (category hierarchy). Additionally, two categories may be connected through a *mutuallyExclusive* relationship (shown using red dotted lines). For example, in Figure 1, *Fruit-subCategoryOf-Food* and *Food-mutuallyExclusive-Organization*.
- **Entity nodes**: Entities are instances of categories. An entity node is connected to its most specific category node though an *isA* edge. In Figure 1, *Banana-isA-Fruit*.

- **Lexical string nodes**: Lexical string nodes correspond to text fragments (usually noun phrases) used to express entities in free text. For example, the lexical strings *Apple* or *Apple Inc.* may be used to refer to the entity *Company:Apple*. Please note an *ambiguous* lexical string may be connected to multiple entities: *Apple* in Figure 1 is connected to both *Company:Apple* and *Fruit:Apple*.

Note that even though a KB may contain additional types of relations among its nodes (e.g., *Microsoft-competesWith-Google*), we omit them here as they are not relevant for the rest of the paper.

### 3.1.2 Glosses

Glosses are natural language definitions of entities in the KB. For example, *"Platinum is a chemical element with the chemical symbol Pt and an atomic number of 78"* is a valid gloss for *Platinum*, whereas *"Platinum deposits are present in the state of Tamil Nadu, India."* is not a gloss. In this paper, we consider DBPedia abstracts as a repository of glosses. However, our methods are generic, and applicable to any set of definitional sentences as candidate glosses.

### 3.1.3 The Gloss Finding Problem

Given a KB (e.g., NELL or Freebase) and a repository of candidate glosses (e.g., DBPedia abstracts), the Gloss Finding Problem requires to identify for each entity in the KB the correct gloss(es) from the candidate gloss pool. For example in Figure 1, we would like to infer that for entity *Fruit:Apple*, *"Apple is a fruit from apple tree ..."* is the only correct gloss. Sometimes it is possible that for certain entities in the KB, there exists no correct gloss in the given candidate gloss pool (e.g., *Strawberry* in Figure 1).

## 3.2 Candidate Entity-Gloss Generation

In this step, for each gloss in the candidate pool of glosses given as input, we first identify the noun phrase that is being defined by the gloss. We refer to this noun phrase as *head-NP* in this paper. For DBPedia abstracts this step is trivial in the sense that DBPedia dataset gives us the head-NP for each short abstract. However, we can easily apply existing syntactic taggers, e.g., the Stanford tagger [26], and detect the *head-NP* in the definitional sentence(s). After this, the head NP of each gloss is lexically matched against the entities in the KB. Even though simple string matching is used for the experiments in this paper, more sophisticated string matchers may also be used.

At the end of this stage, we end up with a set of candidate entity-gloss matchings. Note that this relationship may be many-many and thereby ambiguous as one entity may be connected to multiple glosses, and a single gloss may be assigned multiple entities. Such ambiguity are resolved using the techniques described below.

## 3.3 First approach: Label Propagation

In order to remove uncertainties from the candidate matchings generated in Section 3.2, we first use Modified Adsorption (MAD) [40], a representative graph-based semi-supervised learning (SSL) algorithm. This choice was prompted by the fact that such Label Propagation (LP) techniques have achieved considerable success in various weakly-supervised NLP tasks, and that they could potentially exploit the graph structure of the current problem.

In order to apply MAD to this Gloss Finding problem, we first create an augmented version of the graph in Figure 1 by connecting each gloss to the content words which passed a TF-IDF based filtering. Mutual exclusion constraints were not used in this augmented graph as MAD is not capable of handling such relations. We then inject each entity node with its own node-specific label. This self-labeling approach was also used in [44], although for an entirely different problem. Starting with this augmented graph and self-injected seed labels, the MAD algorithm is used to classify the rest of the nodes in the graph. At the end of the algorithm, MAD assigns each candidate gloss a set of labels, where each label corresponds to an entity since the labels were entity-specific self-injections.

This distribution of entities on a candidate gloss node is intersected with the candidate entities generated in Section 3.2, and the resulting entities are sorted based on the assignment score generated by MAD. The top entity in this ranked list is then chosen as the inferred entity match for the candidate gloss. In experiments below, LP-based approach is only effective in some settings. To address this shortcoming, we propose a new method, GLOFIN.

**Algorithm 1** GLOFIN

**Input**: $X_l$ labeled glosses; $Y_l$ category labels of $X_l$; $X_u$ unlabeled glosses; $O$ class constraints (subclass-superclass or mutual-exclusion kind). $N = |X|$ and $K$ is the number of classes.
**Output:** $Y_u$ labels for $X_u$
Initialize parameters $\theta_j^0$ for each $C_j$ using seeds provided for $C_j$.
**for** $t$ = 1 ... till convergence **do**
  **for** $i$=1 to $|X|$ **do**
    {E step: Assign a bit vector of categories to each gloss}
    Find $P(C_j|x_i; \theta_j^{t-1})$ for all classes $C_j$
    $Y_i^t$ = find consistent assignment by solving MIP($P(C_j|x_i)$, $O$).
  **end for**
  {M step: Recompute model parameters.}
  Recompute $\theta_j^t$ based on current class assignments $Y^t$.
**end for**
return

## 3.4 Proposed Approach: Gloss Finder (GLOFIN)

We propose Gloss Finder (GLOFIN), a hierarchical semi-supervised Expectation Maximization (EM) algorithm for the Gloss Finding problem. GLOFIN uses automatically acquired labeled data (the unambiguous candidate glosses) and large amount of unlabeled data (the rest of the candidate glosses) in an iterative EM framework to learn a model for this task. The GLOFIN algorithm is presented in Algorithm 1. Before presenting the details of the algorithm, let us define the following notations.

Let $X = \{x_1, \ldots x_N\}$ be the candidate glosses, and $\{C_1, \ldots C_K\}$ be the KB categories. Let $y_{ji} \in \{0, 1\}$ be an indicator variable specifying whether candidate gloss $x_i$ belongs to category $C_j$. Using the model parameters $\theta_j$ for class $C_j$, we can estimate $P(C_j|x_i)$, the probability of gloss $x_i$ belonging to category $C_j$. GLOFIN aims to find optimal values of label assignments $y_{ji}$ and cluster models $\theta_j$, so as to maximize the overall data likelihood. Let *Subset* be the set of all subset or inclusion constraints, and *Mutex* be the set of all mutual exclusion constraints. In other words, *Subset* = $\{\langle i, k\rangle : C_i \subseteq C_k\}$ and *Mutex* = $\{(i, k) : C_i \cap C_k = \phi\}$.

GLOFIN takes in as input the set of glosses $X$, out of which $X_l$ is the labeled subset with category labels in $Y_l$, and the remaining $X_u$ is unlabeled. Additionally, it takes as input a set of ontological constraints, $O$. In the E step of this algorithm, each candidate gloss $X_i$ is assigned a bit vector of labels $Y_i = [y_{1i}, \ldots y_{Ki}]$. The class hierarchy is incorporated as constraints on bits in this bit vector. For example, if for a gloss with head NP *Cat*, a bit corresponding to KB category *Mammal* is set then the bit corresponding to category *Animal* should also be set due to the subset constraint: *Mammal* $\subseteq$ *Animal*, and for the same gloss the bit corresponding to category *Reptile* should not be set due to the mutual exclusion constraint: Mammal $\cap$ Reptile = $\phi$. The M step recomputes the model param-

eters for each KB category using the label assignments done in the E step.

The E step of GLOFIN (Algorithm 1) may be broken down into two stages, which we describe in greater detail below.

**Assigning Entities to Candidate Glosses (Algorithm 1: Line 6)**

This step computes probabilities $P(C_j|x_i;\theta_j)$, where $\theta_j$ is the current estimate of model parameters for category $C_j$. A variety of techniques may be used for this estimation, and we briefly describe one such choice here: the semi-supervised version of multinomial Naive Bayes [36]. In this model $P(C_j|x_i) \propto P(x_i|C_j) * P(C_j)$, for each unlabeled gloss $x_i$. The probability $P(x_i|C_j)$ is estimated by treating each feature in $x_i$ as an independent draw from a class-specific multinomial. In this task, the features are word occurrences in the candidate glosses, and the number of outcomes of the multinomial is the vocabulary size. We shall refer to this version of GLOFIN as **GLOFIN-NB**. Additional variants using other learning algorithms, along with features used by our methods are discussed in Section 4.

**Entity Assignment Refinement using Ontological Constraints (Algorithm 1: Line 7)**

Given the category memberships probabilities $\{P(C_j|x_i)\}$ estimated above, this step computes the category membership variables $\{y_{ji},\ \forall\ 1 \le i \le N,\ 1 \le j \le K\}$. GLOFIN solves a Mixed-Integer Program (MIP) to estimate these variables. One such problem is solved for each gloss. This MIP takes the scores $\{P(C_j|x_i)\}$, and class constraints $O$ as input and produces a bit vector of labels as output, each bit representing whether the gloss belongs to that particular category.

The MIP formulation for gloss $x_i$ is presented in Equation 1. For each gloss, this method tries to maximize the sum of scores of selected labels, after penalizing for violation of class constraints. Let $\zeta_{jk}$ are slack variables for *Subset* constraints, and $\delta_{jk}$ are slack variables for *Mutex* constraints.

$$\underset{\{y_{ji}\},\zeta_{jk},\delta_{jk}}{\text{maximize}} \left( \sum_{\mathrm{j}} \mathrm{y_{ji}} * \mathrm{P(C_j|x_i)} - \sum_{\langle \mathrm{i,k}\rangle \in \mathrm{Subset}} \zeta_{\mathrm{ik}} - \sum_{(\mathrm{i,k}) \in \mathrm{Mutex}} \delta_{\mathrm{ik}} \right)$$

subject to,

$$\begin{aligned} y_{ji} &\ge y_{ki} - \zeta_{jk}, \quad \forall \langle j,k\rangle \in \textit{Subset} \\ y_{ji} + y_{ki} &\le 1 + \delta_{jk}, \quad \forall (j,k) \in \textit{Mutex} \\ \zeta_{jk}, \delta_{jk} &\ge 0,\ y_i \in \{0,1\}, \quad \forall j,k \end{aligned} \tag{1}$$

## 3.5 Scaling GLOFIN

The MIP formulation presented in Equation 1 adds a constraint for each subset or mutual exclusion constraints in the KB ontology. Further, in the E step of every iteration, we solve a MIP for each gloss. We make the method more scalable and efficient in following ways:

- We discard redundant mutex constraints - i.e., the ones that can be inferred using remaining subset and mutex constraints.
- We reduce the number of classes considered per gloss, by keeping only top-Q classes relevant to the gloss, as detailed below.
- Since the MIP for each gloss is independent from other MIPs, we parallelize the E step of every iteration, and consolidate results in the M-step.

**Reducing the MIP size per gloss:** We keep only a small number of categories in the optimization problem that is being solved for each gloss. We tried following ways of limiting candidate classes:

- Diameter of the class graph: The class graph is an undirected graph of ontology classes, where nodes are classes and edges represent either subset or mutual exclusion constraints. In this approximation, we rank all classes by the scores $P(C_j|x_i)$ for a datapoint $x_i$, and choose the top Q classes, where Q is the diameter of the category graph. Since the class assignments are hierarchical, we also include all ancestors of these top-Q classes in the optimization.
- Square-root of number of classes: Here, we select $Q = \sqrt{K}$, where $K$ is the total number of classes in the ontology. Similar to diameter based method, we include all ancestors of these top-Q classes in the optimization.
- Thresholding: If the score $P(C_j|x_i)$ is greater than a predefined threshold then we consider the category. Note that the threshold is set for the entire dataset, but for each datapoint might result in different number of categories.

# 4. DATASETS AND EXPERIMENTAL METHODOLOGY

In our experiments, we enrich two existing knowledge bases with available glosses, namely NELL and Freebase. NELL is a machine generated knowledge base that does not have existing glosses, whereas most Freebase entities have glosses we can compare against. Our resource of glosses is DBPedia, a database of factual descriptions extracted from Wikipedia. DBPedia contains a large set of Wikipedia titles and short abstracts. We do not use any structure information from DBPedia to make sure that our methods are generic, and applicable to any set of definitional sentences as candidate glosses. E.g., we can replace DBPedia abstracts with definitions from online glossaries like Wiktionary or WordNet without any need to change the GLOFIN method.

A DBPedia short abstract is essentially the first paragraph (up to 500 characters) on the Wikipedia page of that entity. Some sample entity titles and their corresponding DBPedia abstracts are given in Table 1. We consider two experimental datasets that were labeled for evaluation purposes. We have made both these gloss finding datasets available[3] for research community to help the future research in this field.

| Entity Title | Definition (DBPedia short abstract) |
|---|---|
| Platinum | Platinum is a chemical element with the chemical symbol Pt and an atomic number of 78. ... |
| Britwell | Britwell is a residential housing estate and civil parish in the north west of Slough Berkshire in the south of England. ... |

**Table 1: Sample candidate glosses**

## 4.1 Freebase dataset

Freebase is a huge tuple database used to structure general human knowledge. The data in Freebase is collaboratively created, structured, and maintained. Public read/write access to Freebase is allowed through an HTTP-based graph-query API using the Metaweb Query Language (MQL) as a data query and manipulation language. Currently Freebase consists of over 44M topics and 2613M facts about entities, their relationships and attributes. We use the Freebase entity snapshot provided by ERD'14 challenge [8] as a gold standard mapping of Freebase to DBPedia abstracts. Thus

[3]The dataset can be downloaded from `http://rtw.ml.cmu.edu/wk/WebSets/glossFinding_wsdm_2015_online/index.html`

we get a good quality training and test data for our semisupervised methods on the task of finding glosses for Freebase.

Table 2 includes detailed statistics of the Freebase dataset. Overall, Freebase entities in this dataset belong to 46 Freebase classes. Having considered their parent types, the dataset includes a set of 66 classes. There are 5.5M Freebase entities in total that belong to these 66 classes. In order to obtain the underlying subset and mutual exclusion constraints, we built a co-occurrence matrix of entities belonging to each pair of classes. Based on this matrix, we inferred 46 subset and 1,455 mutual exclusion constraints.

| Statistic | Dataset | |
|---|---|---|
| | Freebase | NELL |
| #Classes | 66 | 275 |
| #Subset class constraints | 46 | 313 |
| #Mutex class constraints | 1455 | 18.1K |
| Diameter of class graph | 4 | 10 |
| $\sqrt{\#\text{classes}}$ | 9 | 17 |
| #DBPedia abstracts | 284.5K | 246.7K |
| #Words | 496.8K | 472.4K |
| #(abstract, word) edges | 5.7M | 7.1M |
| #Unambiguous DBPedia abstracts | 257.3K | 195.8K |
| #Ambiguous DBPedia abstracts | 32.8K | 50.0K |
| #Ambiguous abstracts with ground-truth KB mappings | 12.4K | 383 |

**Table 2: Statistics about the datasets. (Sections 4.1, and 4.2)**

## 4.2 NELL dataset

The NELL knowledge base is created by a machine learning system named Never Ending Language Learning [30]. NELL is composed of various information extraction components [7, 43, 23] that independently extract facts from text and semi-structured information on the Web. Our version of the NELL KB consists of 275 categories and 1.67M instances belonging to these categories. Though the facts in NELL are extracted by a computer system, it takes as input a human created ontology containing categories, plus subset and mutual exclusion constraints between them.

We constructed a dataset using the NELL KB and DBPedia abstracts. Statistics about the NELL dataset are included in Table 2. Unlike Freebase, NELL entities do not have glosses associated with them, so we do not have ground truth gloss data for NELL. Further, since this KB is automatically generated by a computer system it contains many noisy facts, hence the training data used by our methods is noisy.

Next we present results of two manual evaluation experiments for the NELL dataset. We do our first evaluation to understand the quality of automatically acquired seeds and predicted entity labels for ambiguous DBPedia abstracts. This will later help us to make claims about robustness of our proposed methods towards noisy training data. Further, we do not have a gold standard set of mappings from DBPedia abstracts to NELL entities, which are needed to evaluate our systems. Hence, we manually constructed a gold standard set of mappings by labeling 383 ambiguous DBPedia abstracts to the correct NELL entities. Also note that these gold standard mappings are used only as a test set, and our methods use only automatically generated training data for learning.

### *Quality of automatically acquired seeds*

Our methods use unambiguous mappings of DBPedia abstract onto NELL entity as labeled data. Since these mappings are determined automatically for the NELL dataset, we performed manual evaluation in order to assess the quality of the unambiguous alignments, inspecting 100 randomly sampled unambiguous DBPedia-NELL matches. For each abstract in this sample, we evaluate whether the assigned class is *precise*, *correct*, and whether a higher level category is correct. We illustrate these measures using examples. If a DBPedia abstract about "Shively Field (public airport)" was mapped to the category "airport", then the mapping is considered to be precise, correct, and the higher level category is correct as well. Mapping between another DBPedia abstract about "Christopher McFarland (baseball player)" to the category "person-north-america", is judged as correct, but not precise, as there exists more concrete "athlete" category in NELL. Finally, a mapping between "Jonathan Andersson (hockey player)" and "director" is incorrect and not precise. The higher level category in this case is correct however, since "director" is a descendant of "person", which is correct.

| Statistic | Value |
|---|---|
| #unambiguous abstracts evaluated | 100 |
| #abstracts s.t. assigned category was correct | 81 |
| #abstracts s.t. assigned category was most precise | 72 |
| #abstracts s.t. assigned higher level category was correct | 94 |

**Table 3: Evaluating quality of unambiguous mappings of DBPedia abstracts to NELL entities/categories. (Section 4.2)**

Table 3 shows the results of this evaluation. We found that out of 100 unambiguous mappings between abstracts and NELL categories, 72% were precise and 81% were correct. The higher level category was correct in 94% of the considered examples. While these alignments are imperfect, the experimental results described below will show that our techniques make effective use of this automatically labeled data.

### *Manually creating gold-standard mappings from DBPedia abstracts to NELL entities*

As we have already stated, NELL does not have glosses for its entities and unlike Freebase there is no gold-standard mapping available from DBPedia abstracts to NELL entities. Hence we randomly sampled 383 DBPedia abstracts for which multiple candidate NELL entities and categories were found. For each abstract, we manually checked whether one of the candidate NELL entities and categories was a correct match. We also checked whether the most precise entity is present in the NELL KB. For some DBPedia abstracts, the most precise (entity, category) combination was not present in NELL, but the most precise category was present. The statistics of our evaluation are listed in Table 4.

| Statistic | Value |
|---|---|
| #abstracts evaluated | 383 |
| %abstracts $\exists$ at least 1 NELL entity, category match | 79% |
| %abstracts $\exists$ most precise entity candidate in KB | 68% |
| %abstracts $\exists$ most precise category candidate in KB | 98% |

**Table 4: NELL Dataset: Manual evaluation of ambiguous glosses. Please refer Section 4.2 for more details.**

From Table 4 we can say that 79% DBPedia abstracts have at least one correct entity match in the candidate entity set. E.g., a DBPedia abstract about "Michael Jordan" as Basketball player, can be matched to the NELL entity "Michael Jordan:Person", however most precise entity will be "Michael Jordan:Athlete". We consider "Michael Jordan:Person" as a correct candidate entity, however "Michael Jordan:Athlete' is the most precise entity, and "Athlete" is the most precise category.

## 4.3 Methods

We experimented with the following eight methods for the gloss finding task. First two methods are existing baseline methods.

- **SVM:** This is a traditional supervised linear SVM algorithm applied for the task of classifying head-NPs from candidate glosses into KB categories. We learnt a separate binary classifier for each KB category. For this purpose, we used the publicly available LibLinear package [15] with varying values of parameter 'C' that controls penalty on slack variables. A similar method was used by Martinez et al. [27] which trains a decision list classifier on monosemous (unambiguous) words for the word sense disambiguation task; we choose a more widely used supervised learning method, SVM.
- **Label propagation:** This is a representative graph-based semi-supervised learning (SSL) algorithm, called Modified Adsorption (MAD) [40], as described in Section 3.3.

The next three methods are variants of our proposed GLOFIN algorithm (Section 3.4) using different document representations.

- **GLOFIN-NB:** As already described in Section 3.4, this is a semi-supervised version of multinomial Naive Bayes [36].
- **GLOFIN-KM:** Here we use the seeded spherical K-Means algorithm proposed by Basu and Mooney [4] as another variant of semi-supervised EM.
- **GLOFIN-VMF:** Banerjee et al. [2] proposed a generative mixture model approach to clustering datapoints based on von Mises-Fisher distribution, defined for data distributed on the unit hypersphere ($L_2$ norm equals 1). We use a seeded version of this algorithm, similar to Dalvi et al. [12] who used it for the task of exploratory learning.

To evaluate the benefits of incorporating hierarchy, we also consider flat versions of semisupervised Naive Bayes, seeded spherical K-Means and seeded von-Mises Fisher. For each flat method, the E step (Algorithm 1: Line 6) computes $P(C_j|x_i)$ for all leaf classes $C_j$ in the ontology and picks the one with maximum probability, skipping the mixed integer programming step (Algorithm 1: Line 7). M step does not change. These methods are referred to as GLOFIN-flat-NB, GLOFIN-flat-KM and GLOFIN-flat-VMF.

## 4.4 Features

Each gloss can be represented by the term frequency (TF) of words that occurred in it. We can also collect inverse document frequency (IDF) of each word in the vocabulary from the entire gloss corpus. This gives us a choice of using either TF (mere term frequencies) or TFIDF (multiplication of TF and IDF values) feature representations. We use the TFIDF values to threshold the words used for each gloss. For all the experiments presented here, we keep only those words per gloss that exceed a TFIDF threshold of 1E-3, thereby discarding stop-words.

For each method, we performed experiments with both TF and TFIDF based feature representations (considering only those words that pass TFIDF threshold). In our experiments we found that SVM and Label-propagation methods give better performance for TFIDF based feature representation, whereas all semisupervised EM methods (GLOFIN variants) work best with a TF based feature representation. Henceforth, we present all results using these feature representations.

Since feature engineering is not the focus of this paper, we use a standard bag of words features for all our experiments. Any additional features can be incorporated easily in our algorithm and are complementary to the techniques introduced in this paper.

# 5. EXPERIMENTAL RESULTS

## 5.1 Gloss Finding Results

In this section we present experimental results on the gloss finding task. For the first set of results we use all unambiguous glosses (glosses with only one candidate entity mapping to the KB) as training data. The next set of experiments make use of only 10% of unambiguous matches as training data, to test the robustness of the methods for more ambiguous data. We compare all eight methods that we presented in Section 3: SVM, label propagation, the hierarchical and flat variants of GLOFIN.

For the SVM method, we tried different values of 'C' for the LibLinear package. We observed the change in performance varying the values of 'C' in the range [1E-24, 100] for both datasets (plots omitted due to space constraints). We found that best performance is achieved for 'C' in the range [1E-6, 0.01]. Results presented here are with C = 1E-6 (results for C = 0.01 are very similar).

From the experiments presented here, we try to answer the following research questions:

- How does our proposed GLOFIN-NB method compare with SVM and label propagation on the task of assigning glosses for NELL and Freebase datasets?
- How do the Naive Bayes, K-Means and von Mises-Fisher variants of GLOFIN compare?
- Does the use of ontological constraints always improve results for GLOFIN?
- What is the effect of the different strategies of limiting the set of considered classes per gloss on scalability (and performance) of GLOFIN?

### *5.1.1 Comparing GLOFIN-NB against SVM and label propagation*

| Method | Performance on Ambiguous Glosses | | | | | |
|---|---|---|---|---|---|---|
| | NELL | | | Freebase | | |
| | P | R | F1 | P | R | F1 |
| SVM | 59.3 | 21.3 | 31.3 | 87.8 | 13.0 | 22.7 |
| Label Propagation | 42.8 | 54.0 | 47.8 | 89.8 | **89.1** | **89.4** |
| GLOFIN-NB | **70.4** | **65.4** | **67.8** | **94.6** | 74.2 | 83.2 |

**Table 5: Comparison of gloss finding methods using all unambiguous glosses as training data and ambiguous glosses as test data. Best values in each column are bold-faced. GLOFIN-NB method is robust to noisy training data for the NELL dataset. Please refer Section 5.1.1 for more details.**

Here we compare our proposed GLOFIN-NB method against supervised SVM and semi-supervised label propagation methods. We compare them in two settings: first using all unambiguous glosses as training data and second, by simulating a harder problem i.e. using only 10% of the unambiguous glosses for training.

**Using all unambiguous glosses for training:** Table 5 shows the summary of results while using all unambiguous glosses as training data and ambiguous glosses for evaluation.

We can see that, supervised SVM classification performs worst on both datasets. Label propagation gives the best F1 score for Freebase, however it performs poorly on the NELL dataset, whereas GLOFIN-NB works well on both NELL and Freebase datasets. On the Freebase dataset, GLOFIN-NB has higher precision, lower recall, and hence slightly lower, but comparable F1 than the label propagation method.

| Method | NELL Dataset | | | | | | Freebase Dataset | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Ambiguous glosses | | | All glosses | | | Ambiguous glosses | | | All glosses | | |
| | Precision | Recall | F1 | Precision | Recall | F1 | Precision | Recall | F1 | Precision | Recall | F1 |
| SVM | 64.2 | 17.3 | 27.3 | 99.9 | 27.9 | 43.7 | 87.8 | 11.3 | 20.1 | 97.9 | 71.1 | 82.5 |
| Label Propagation | 55.0 | 12.3 | 20.1 | 99.7 | 5.2 | 9.8 | 84.6 | 27.5 | 41.5 | 99.7 | 88.1 | **93.6** |
| GLOFIN-NB | 71.7 | 62.0 | **66.5** | 99.9 | 62.0 | **76.5** | 95.1 | 72.0 | **82.0** | 97.6 | 79.6 | 87.6 |

**Table 6: Comparison of gloss finding methods with 10% of the unambiguous abstracts used as training data. GLOFIN-NB always gives best or near-best F1 scores. Please refer Section 5.1.1 for more details.**

Both label propagation and GLOFIN-NB perform better on the Freebase dataset than the NELL dataset. This can be explained by the the difference in quality of the automatically acquired seeds for the two datasets. Note that the unambiguous glosses used for the Freebase dataset are very accurate, whereas for the NELL dataset our evaluation in Table 3 shows that only 81% of the seed gloss mappings were accurate. Hence the results in Table 5 indicate that our proposed GLOFIN-NB method is relatively robust to noise compared to the label propagation method.

We also investigated why the supervised SVM method performs poorly even though 80% of the total data is used as training. We found that there is a huge skew in the class frequencies. For the Freebase dataset with 45 leaf classes, the average number of training instances per class are 2.27K with a standard deviation of 10.8K; e.g., "/location/location" category has 72.6K training examples, whereas "/education/university" category has just 10 training examples. We hypothesize that the small amount of training data for many KB categories is the reason why the SVM method performs poorly.

Note that in Table 5, the number of unambiguous glosses used as training data covered a large part of the overall dataset (80% for NELL and 90% for Freebase). However, in real life scenarios, the amount of training data can be much smaller. The Freebase dataset is artificially created using ERD'14 data and a subset of Freebase KB categories. NELL populates its KB by bootstrapping against many different categories simultaneously, beginning with a small set of hand-chosen seeds. It may be the case that this learning process tends to favor entity names that can be confidently assigned to a single category (i.e., that are not highly ambiguous); to support this conjecture, we note that an alternative instance of NELL which used a larger set of automatically generated low-quality seeds required modifications to suppress the use of ambiguous entities in bootstrapping [32]. This suggests that other real-world KBs may have a lower proportion of unambiguous entities. Next, we conduct experiments which simulates this setting, by using only 10% of the available unambiguous matches as seeds.

**Using 10% unambiguous glosses for training:** Table 6 shows detailed results of GLOFIN-NB compared to SVM and label propagation methods when only 10% of the unambiguous glosses are used as training data. Since we are using only a fraction of unambiguous glosses as training data, the rest of them can be used as test data. Additionally, all gold-standard mappings of ambiguous glosses are always part of test data. We generate 10 random train/test partitions and average results on these 10 runs. Performance on ambiguous glosses is particularly interesting hence is listed separately. Note that "All glosses" include ambiguous as well as unambiguous glosses.

We can see that with less training data the performance of all methods degrades, to varying degrees. Also, all methods give higher performance on "all glosses" compared to the harder task of matching ambiguous glosses. In terms of F1 scores on ambiguous glosses, SVM results in the worst performance and GLOFIN-NB method gives the best performance. In terms of F1 score on "all glosses", Comparing GLOFIN-NB vs. label propagation we can see that GLOFIN-NB performs better in all cases, except Freebase dataset "all glosses" case where GLOFIN-NB has comparable precision and lower recall, and hence lower F1 score compared to label propagation.

To summarize, GLOFIN performs best on the harder task of matching ambiguous glosses when the amount of training data is small.

**Discussion:** Comparing the results across Tables 5 and 6, we can hypothesize that, with large fraction (90% for Freebase dataset) of the glosses being unambiguous in our artificially created dataset, it is very easy to get high values for F1 score using label propagation (Table 5), because the rest of the 10% nodes might get high quality labels from its nearest neighbors. However when the amount of training data is smaller, generative models like GLOFIN-NB work better (Table 6) as they generalize better.

One more advantage of GLOFIN lies in its generative nature. Label propagation is a transductive approach, whereas GLOFIN is a generative approach, I.e., EM models learnt by GLOFIN on a set of datapoints can be applied to an unseen datapoint having similar vocabulary. In other words, they can predict labels for an unseen datapoint. However label propagation just predicts labels for the datapoints in the set being used while learning.

| Method | Performance on Ambiguous Glosses | | | | | |
|---|---|---|---|---|---|---|
| | NELL | | | Freebase | | |
| | P | R | F1 | P | R | F1 |
| GLOFIN-flat-KM | 74.8 | 36.5 | 49.1 | 97.6 | 61.0 | 75.1 |
| GLOFIN-KM | 65.2 | 49.8+ | 56.5+ | 96.1 | 71.8+ | 82.2+ |
| GLOFIN-flat-VMF | 73.6 | 44.5 | 55.5 | 96.3 | 56.0 | 70.8 |
| GLOFIN-VMF | 77.2+ | 59.5+ | 67.2+ | 95.7 | 59.3+ | 73.2+ |
| GLOFIN-flat-NB | 70.3 | 59.1 | 64.3 | 95.9 | 71.1 | 81.7 |
| GLOFIN-NB | 70.4+ | 65.4+ | **67.8+** | 94.6 | 74.2+ | **83.2+** |

**Table 7: Comparison of GLOFIN variants using all unambiguous glosses as training data and ambiguous glosses as test data. Best values of F1 for each dataset is bold-faced. '+' in front of a hierarchical method score indicates that the score improved over its flat version. (a) All hierarchical methods perform better than their flat counterparts. (Section 5.1.2) (b) GLOFIN-NB method performs the best (Section 5.1.3)**

### *5.1.2 Effect of using ontological constraints*

From Table 7, we can also see that all hierarchical semisupervised EM methods are better than their flat counterparts. Hence we conclude that using ontological constraints improves the gloss finding performance. Note that relative improvements of the hierarchical methods are higher for the NELL dataset (upto 21% relative improvement in F1 scores). The reason traces back to evaluation of NELL seeds in Section 4.2, Table 3. We saw that 81% leaf category seed labels were correct, whereas 94% of higher level category la-

bels were correct. Thus learning separate model parameters for higher level categories in the ontology and using ontological constraints to resolve ambiguity employed by hierarchical GLOFIN proves beneficial. Since Freebase seed labels are more accurate, and hierarchy contains just 2 levels, hierarchical models do not have the much added advantage over flat models as for the NELL dataset, resulting in only 9% relative improvement in F1 scores.

### 5.1.3 Comparing variants of GLOFIN

Table 7 shows detailed results of our proposed methods (GLOFIN-NB, GLOFIN-KM and GLOFIN-VMF) w.r.t their flat counterparts (GLOFIN-flat-NB, GLOFIN-flat-KM and GLOFIN-flat-VMF). GLOFIN-NB method works the best on both NELL and Freebase datasets.

GLOFIN-NB outperforms GLOFIN-KM in the experiments. Importantly, GLOFIN-NB models class priors (as discussed in Section 5.1.1)). As mentioned before, the class frequencies in our datasets are very skewed; hence modeling class priors is highly useful. In contrast, GLOFIN-KM merely computes cosine similarity of a datapoint with respect to a cluster centroid, hence its inferior performance. The GLOFIN-VMF method provides the second-best performance to GLOFIN-NB on the ambiguous gloss. Although GLOFIN-VMF is cluster-based, it also models class priors.

### 5.1.4 Comparing different ways of scaling GLOFIN

We tried various approximations to reduce runtime of GLOFIN(Please refer to Section 3.5 for details). The summary is presented in Table 8. We found that setting $Q$, the number of classes considered in a MIP (Equation 1), as the diameter of the class graph gives huge time savings and does not harm the performance in terms of F1 score on ambiguous entities. Hence we use this approximation for all runs of GLOFIN in this paper. In addition to this, we also use 12 parallel threads in the E step to compute label assignments of all datapoints in parallel.

Note that for NELL dataset these approximations are crucial. Otherwise, run time on a PC machine (with 30GB memory) exceeds 64 hours. Due to large processing requirements, we do not have F1 scores for this case. Besides, results in Table 6 are averaged over 10 random train/test partitions for both NELL and Freebase. However, due to large processing times for some variants, results in Table 8 are averaged over 3 and 9 train/test partitions for the NELL and Freebase respectively.

| Approximations | Performance on Ambiguous Glosses | | | |
|---|---|---|---|---|
| | NELL | | Freebase | |
| | F1 | Time | F1 | Time |
| keep all classes | - | >230.4K | 80.4 | 15.2K |
| Q = diameter | 68.1 | 5.2K | 81.8 | 5.7K |
| Q = $\sqrt{K}$ | 65.5 | 14.5K | 83.1 | 3.6K |
| score threshold= 1E-5 | 64.5 | 22.9K | 71.4 | 9.3K |

**Table 8: Comparison of different approximations to scale our GLOFIN-NB method using 10% of the unambiguous glosses as training and ambiguous glosses as test data. Time measurements are in seconds. Refer to Section 5.1.4 for more details.**

## 5.2 Evaluating NELL to Freebase mappings via common glosses

Aligning two KBs to the same set of glosses gives information about how the categories in those two KBs should be aligned. To evaluate the value of this information, we took the output of GLOFIN-NB and randomly sampled 100 glosses that were assigned to entities from both NELL and FreeBase. Then we did a manual evaluation whether the entities from NELL and Freebase correspond to each other, and whether the categories they belong to in the respective KBs are semantically similar.

| Eval Type | Statistic | Value |
|---|---|---|
| #Abstracts | Evaluated | 100 |
| | Assigned entities are correct, corresponding categories are semantically related | 93 |
| | Assigned NELL category is precise | 92 |
| | Assigned Freebase category is precise | 38 |
| #Category pairs | Found in 100 abstracts evaluated | 39 |
| | NELL category = Freebase category | 6 |
| | NELL category $\subset$ Freebase category | 23 |
| | Freebase category $\subset$ NELL category | 1 |

**Table 9: Evaluating quality of NELL to Freebase mappings via common DBPedia abstracts. (Please refer to Section 5.2.)**

From Table 9, we can see that out of 110 abstracts that were evaluated, 93 of them had correct NELL and Freebase entities assigned to them, and their corresponding categories were semantically related. For 92 of those abstracts, the NELL category was precise and correct, while for 38 of them the Freebase category was precise. This is due to the fact that the Freebase categories we use to build our dataset are more general categories like "/organization/organization", "/location/location", and more precise categories like "/location/city" and "/organization/biotech-company" are missing. NELL has all these categories at finer granularity, hence it can classify DBPedia abstracts into more fine-grained categories. For instance, a DBPedia abstract about "Biotronik" is classified into "biotechcompany" category from NELL, and "/organization /organization" category from Freebase. We evaluated whether the entities from the two KBs are correct, and whether the corresponding categories are semantically related.

We also evaluated whether we can come up with a relationship between the categories corresponding to NELL and Freebase mappings. From 100 abstracts, we found 39 category pairs, which we manually evaluated. We found 6 category pairs that can be considered as equivalent. E.g., We marked the category "geopoliticallocation" from NELL to be equivalent to category "/location/location" from Freebase. In 23 category pairs, we found that the NELL category was strict subset of Freebase category. E.g., the "actor" category from NELL is strict subset of "/person/person" category from Freebase. Only one category in Freebase "/broadcast/broadcast" was found to be strict subset of a NELL category "company". For 9 category pairs we could not define a equality or subset relation between them. They include either semantically unrelated class pairs like "visualartist" and "/location/location" corresponding to incorrect entity matches, and semantically related categories like "drug" and "/business/brand". There are many drugs like Donnatal, Excedrin that are classified as "drugs" in NELL and as "/business/brand" in Freebase. Though these categories are related to each other we can not easily classify them into equivalence or subset relations. We have also provided some sample outputs of our methods at the following URL `http://rtw.ml.cmu.edu/wk/WebSets/glossFinding_wsdm_2015_online/index.html`.

*Acknowledgments:*

This work is supported in part by the Google PhD fellowship in Information Extraction, a Google research grant and BSF grant No. 2010090. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Google, or BSF.

# 6. CONCLUSION

In this paper, we proposed GLOFIN, a method for the important but relatively unexplored problem of Automatic Gloss Identification, i.e., automatically generating glosses (short definitional sentences) for an initially gloss-free knowledge base (KB) by matching candidate glosses to entities in the KB. To the best of our knowledge, this is the first such system for this task. GLOFIN employs hierarchical clustering algorithm that internally uses unambiguous DBPedia abstracts as seeds and the KB ontology as constraints to match an ambiguous candidate gloss to the right entity from KB. GLOFIN uses mixed integer programming formulation to assign the most likely set of labels for each gloss, while following the class constraints posed by the KB ontology.

We present experiments using NELL and Freebase as KBs and DBPedia abstracts as candidate glosses. Our quantitative and qualitative evaluations show that GLOFIN is effective and that it outperforms other state-of-the-art algorithms like label propagation and Support Vector Machines (SVM). We also demonstrate GLOFIN's robustness to noise through experiments on a wide variety of KBs, ranging from user contributed (e.g., Freebase) to automatically constructed (e.g., NELL). To facilitate further research in this area, we have made the datasets and code used in this paper publicly available. In future, we would like to extend GLOFIN to discover new entities and new categories to further enrich the knowledge bases. We are also working on extracting a large set of definitional sentences from a web scale text corpus, that can enable us to improve the coverage of glosses and entities in the KB beyond those that are part of existing online glossaries like Wikipedia.

# 7. REFERENCES


[1] E. Agirre, O. L. de Lacalle, and A. Soroa. Random walks for knowledge-based word sense disambiguation. *Computational Linguistics*, 2014.

[2] A. Banerjee, I. S. Dhillon, J. Ghosh, and S. Sra. Clustering on the unit hypersphere using von mises-fisher distributions. In *JMLR*, 2005.

[3] S. Banerjee and T. Pedersen. Extended gloss overlaps as a measure of semantic relatedness. In *IJCAI*, 2003.

[4] S. Basu, A. Banerjee, and R. Mooney. Semi-supervised clustering by seeding. In *ICML*, 2002.

[5] F. D. Benedictis, S. Faralli, and R. Navigli. Glossboot: Bootstrapping multilingual domain glossaries from the web. In *ACL*, 2013.

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

[7] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. H. Jr., and T. M. Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, 2010.

[8] D. Carmel, M.-W. Chang, E. Gabrilovich, B.-J. P. Hsu, and K. Wang. ERD 2014: Entity recognition and disambiguation challenge. *SIGIR Forum*, 2014.

[9] N. Choi, I.-Y. Song, and H. Han. A survey on ontology mapping. *SIGMOD*, 2006.

[10] J. Dalton, L. Dietz, and J. Allan. Entity query feature expansion using knowledge base links. In *SIGIR*, 2014.

[11] B. Dalvi, W. W. Cohen, and J. Callan. Classifying entities into an incomplete ontology. In *AKBC*, 2013.

[12] B. Dalvi, W. W. Cohen, and J. Callan. Exploratory learning. In *ECML*, 2013.

[13] B. Dalvi, C. Xiong, and J. Callan. A language modeling approach to entity recognition and disambiguation for search queries. In *ERD, Entity Recognition and Disambiguation Challenge at SIGIR*, 2014.

[14] W. Duan and A. Yates. Extracting glosses to disambiguate word senses. In *NAACL HLT*, 2010.

[15] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *JMLR*, 2008.

[16] S. Faralli and R. Navigli. A new minimally-supervised framework for domain word sense disambiguation. In *EMNLP-CONLL*, 2012.

[17] C. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[18] E. Gabrilovich and S. Markovitch. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *IJCAI*, 2007.

[19] H. Hajishirzi, L. Zilles, D. S. Weld, and L. Zettlemoyer. Joint coreference resolution and named-entity linking with multi-pass sieves. In *EMNLP*, 2010.

[20] X. Han, L. Sun, and J. Zhao. Collective entity linking in web text: a graph-based method. In *SIGIR*, 2011.

[21] H. Ji, R. Grishman, and H. T. Dang. Overview of the TAC 2011 knowledge base population track. In *TAC*, 2011.

[22] A. Kilgarriff and J. Rosenzweig. English senseval:report and results. In *LREC*, 2000.

[23] J. Krishnamurthy and T. M. Mitchell. Which noun phrases denote which concepts? In *NAACL HLT*, 2011.

[24] M. Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *International conference on Systems documentation*, 1986.

[25] T. Lin, O. Etzioni, et al. Entity linking at web scale. In *AKBC*, 2012.

[26] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky. The Stanford CoreNLP natural language processing toolkit. In *ACL demo*, 2014.

[27] D. Martinez, O. L. de Lacalle, and E. Agirre. On the use of automatically acquired examples for all-nouns word sense disambiguation. *JAIR*, 2008.

[28] P. N. Mendes, M. Jakob, and C. Bizer. Dbpedia: A multilingual cross-domain knowledge base. In *LREC*, 2012.

[29] G. A. Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 1995.

[30] T. Mitchell. Nell: Never-ending language learning. *http://rtw.ml.cmu.edu/rtw/*.

[31] A. Moro, A. Raganato, and R. Navigli. Entity linking meets word sense disambiguation: a unified approach. *TACL*, 2014.

[32] D. Movshovitz-Attias and W. W. Cohen. Alignment-hmm-based extraction of abbreviations from biomedical text. In *BioNLP workshop*. NAACL, 2012.

[33] N. Nakashole, M. Theobald, and G. Weikum. Scalable knowledge harvesting with high precision and high recall. In *WSDM*, 2011.

[34] N. Nakashole, T. Tylenda, and G. Weikum. Fine-grained semantic typing of emerging entities. *ACL*, 2013.

[35] R. Navigli. Word sense disambiguation: A survey. *ACM Computing Surveys*, 41(2), 2009.

[36] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine learning*, 2000.

[37] M. T. Pilehvar and R. Navigli. A robust approach to aligning heterogeneous lexical resources. In *ACL*, 2014.

[38] S. P. Ponzetto and R. Navigli. Knowledge-rich word sense disambiguation rivaling supervised systems. In *ACL*, 2010.

[39] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *WWW*, 2007.

[40] P. Talukdar and K. Crammer. New regularized algorithms for transductive learning. In *ECML-PKDD*. 2009.

[41] P. P. Talukdar. Graph-based weakly-supervised methods for information extraction & integration. 2010.

[42] C. Wang, K. Chakrabarti, T. Cheng, and S. Chaudhuri. Targeted disambiguation of ad-hoc, homogeneous sets of named entities. In *WWW*, 2012.

[43] R. C. Wang and W. W. Cohen. Character-level analysis of semi-structured documents for set expansion. In *EMNLP*, 2009.

[44] D. Wijaya, P. P. Talukdar, and T. Mitchell. Pidgin: ontology alignment using web text as interlingua. In *CIKM*, 2013.

[45] A. Yates, M. Cafarella, M. Banko, O. Etzioni, M. Broadhead, and S. Soderland. Textrunner: open information extraction on the web. In *NAACL HLT demo*, 2007.

[46] M. Yu, S. Wang, C. Zhu, and T. Zhao. Semi-supervised learning for word sense disambiguation using parallel corpora. In *FSKD*, 2011.