
Occurrence Statistics of Entities, Relations and Types on the Web

SUBMITTED IN PARTIAL FULFILLMENT OF REQUIREMENTS FOR THE DEGREE OF MASTER OF
TECHNOLOGY

BY

AMAN MADAN

UNDER THE GUIDANCE OF PROF. SUNITA SARAWAGI



*Department of Computer Science and Engineering
Indian Institute of Technology Bombay*

APRIL, 2014

Occurrence Statistics of Entities, Relations and Types on the Web

April 2014

Abstract

The problem of collecting reliable estimates of occurrence of entities on the open web forms the premise for this report. The models learned for tagging entities cannot be expected to perform well when deployed on the web. This is owing to the severe mismatch in the distributions of such entities on the web and in the relatively diminutive training data. In this report, we build up the case for maximum mean discrepancy for estimation of occurrence statistics of entities on the web, taking a review of named entity disambiguation techniques and related concepts along the way.

1 Introduction

1.1 Problem Statement

The Internet is a web of mostly unstructured knowledge woven around things. However, these things; people, places, technologies, movies, products, books etc. are mostly just mentioned by their name, with other crucial bits of information about them scattered around the point of mention. The cosmic scale of such unstructured information has stemmed the dream of a semantic web. A web which is aware of the links

that make sense, which *understands* what the user is looking for and which is gifted with the intelligence of locating the desideratum. There are several pieces in the puzzle of the semantic web, this report is an attempt to understand one important piece; entities on the web and their co occurrence statistics.

Given a knowledge base such as Yago or Freebase consisting of entities and relations, and the Web, our goal is to attach reliable estimates of the frequency of occurrences on the Web of various entities and relations as singletons, pairs (ordered and unordered) in a sentence. The aim is to collect statistics so as to be able to assign prior probabilities to the set of entities and relations that can co-exist in a sentence or a paragraph. These statistics have applications in query interpretation and language understanding tasks. We can view it as being analogous to statistics in relational catalogs.

1.2 Named Entity Recognition and Disambiguation

For collecting the statistics about entities on the web, we need a method to determine which words in the free flowing interminable text are of interest, i.e. represent entities.

Consider the following sentence :

Michael Jordan is a Professor at Berkeley

We first want to identify all the **named entities** in the text. The task is called named entity recognition and is formally defined as :

Definition 1 (Named entity recognition¹)

Named-entity recognition (NER) (also known as entity identification and entity extraction) is a subtask of information extraction that seeks to locate and classify atomic elements in text into predefined categories such as the names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages, etc.

but we do not stop at that, we want to link each of the named entities thus recognized to a knowledge base². Thus, our problem has a 2 step solution :

- Step 1 : **Identify** entities

Michael Jordan_PERSON is a professor at Berkeley_INSTITUTION

- Step 2 : **Link** entities to knowledge bases :

Michael Jordan_ENTITY (http://en.wikipedia.org/wiki/Michael_I._Jordan) is a professor at Berkeley_ENTITY (http://en.wikipedia.org/wiki/University_of_California,_Berkeley)

The stanford NER library is a popular choice for recognizing named entities. [5]

1.2.1 Applications

In simple terms, disambiguating named entities in the unstructured text imparts a structure to the document. We need two more data points to further appreciate the power that such a tool provides to us. The first is the size of the web. As of 31st March 2014, there are atleast 1.8 billion indexed web pages.[7] The second is the number of wikipedia entities. The wikipedia statistics [8] estimate the number of pages to be around 32 million.

²The knowledge base is a catalog of entities, like Wikipedia. Refer section [2]

Yago, a catalog of entities made from wikipedia has 12, 727, 222 entities. Imparting structure to documents at this magnitude has far reaching implications in the information extraction and is a bridge towards the hitherto dream of a semantic web.

It is highly recommended that the reader pays <http://www.google.co.in/insidesearch/features/search/knowledge.html>, the google knowledge graph project, a visit.

1.2.2 Terminology

The following terms are widely used in the literature on named entity disambiguation and thus in this article.

- **Mention, Spot**
A piece of text which needs to be disambiguated. For example, the sentence “**Amazon** has attracted a lot of visitors”.
- **Entity**
A named entity as defined in the definition 1.
- **Candidates**
A set of entities which might be the correct disambiguation for a given mention. For example, possible candidates for the sentence above are “Amazon river” and “Amazon.com”.
- **Prior**
Probability of a mention linking to a particular entity. For example, the mention “Amazon“ may be used to refer to the website (say) 60% of the time.
- **Knowledge base**
A catalog of Entities where an entity is as defined above. For example, Wikipedia or yago.

1.3 A Baseline : Label and Collect

The baseline which presents itself given the above problem is labeling the corpora with the named entities and then collecting the markings, keeping track of which entity was seen when along the way. As intuitive as it seems, the method is unlikely to perform

well in the present scenario, owing to the mismatch in the training and test distribution [20]. Our training data, hand labeled corpora, is paltry in comparison with the massive open web, where such systems are supposed to be deployed. This is true even for large training datasets like the Wikipedia.

1.4 Maximum mean discrepancy

The observation that we don't really want the individual labels is a first step towards a better solution. There are 3 reported methods for direct estimation of class ratios [20]. We are interested in using one of them, maximum mean discrepancy (mmd) for solving the problem in hand.

We introduce mmd and propose a formulation for determining class ratios in section 5.

1.5 Structure

Section 2 gives an overview of what are knowledge bases. This is important since the concept of such repositories of structured knowledge is central to the report.

Section 3 begins with an introduction to the problem of named entity disambiguation, the terminology and applications, and goes on to cover the techniques for named entity disambiguation in some detail. We give an overview of the two broad categories of disambiguation techniques, Local and global disambiguation.

Section 4 begins with a discussion on definition of Aggregate statistics and some of their applications. Finally, in section 5, we discuss Maximum mean discrepancy and its application for estimating the aggregate statistics over entities.

2 Structured Knowledge Repositories

2.1 What are knowledge bases?

Before the digital age, Encyclopedias, such as the Encyclopedia Britannica were hailed as the repositories containing all that is known to the mankind. As the

computer age dawned, it didn't take long for people to realize that a lot can be achieved if somehow all this information could be made available in a digital format. Wordnet [15] was perhaps the first such attempt. As the years passed, the research effort in the field of information extraction and creating structured knowledge got a huge pat on the back from the explosion of the web. Wikipedia catalyzed the community, which motivated development of structured knowledge bases like dbpedia and yago.

We discuss how knowledge bases fit in the context of named entity disambiguation, and give a list of several important knowledge bases, along with links to each for the interested reader.

2.2 Knowledge bases and Named Entity Disambiguation

Many named entity disambiguation algorithms exploit large knowledge bases. On the other hand, reliable named entity disambiguators will be conducive towards fabrication of gargantuan knowledge bases from the open web. We thus see a chicken and egg situation here. As is often the case in such standoffs, the cycle is broken with the help of extensive manual effort. In the present case, Wikipedia helps the situation.

2.3 Existing Knowledge Bases

We give a brief overview of some of the popular knowledge bases.

2.3.1 Wordnet

- Wordnet has a clean, hand crafted type hierarchy. Well documented APIs, such as the nltk toolkit (<http://www.nltk.org/howto/wordnet.html>) are available for using wordnet for a plethora of tasks, such as listing all the senses of a word, finding distances between 2 concepts and the likes.
- Introduction to Wordnet <http://wordnetcode.princeton.edu/5papers.pdf>

2.3.2 YAGO

- An attempt to create a knowledge base that combines the clean type hierarchy of wordnet with the huge information that Wikipedia provides. <http://www.mpi-inf.mpg.de/yago-naga/yago/> has link to an online interface. Refer [16] for details.

2.3.3 DBpedia

- DBpedia <http://dbpedia.org/About> extracts information from the Wikipedia into RDF and provides an interface that can be used to ask semantic questions. Users can use SPARQL to ask complicated queries with results spanning several pages. Amazon also provides a DBpedia machine image for the users of AWS.

2.3.4 Patty

- Patty <http://www.mpi-inf.mpg.de/yago-naga/patty/> is a repository of relation patterns. The aim is to create “Wordnet” for relations. The authors also create a subsumption hierarchy for the 350, 569 pattern synsets. Refer [18] for details.

2.3.5 Freebase

- Freebase [19] relies on crowd sourcing for creation of a rich but clean knowledge base. The development of Freebase follows the same chain as Wikipedia, with users flagging issues, and cleaning and augmenting information. Freebase also provides access to itself using web APIs.

3 Named Entity Disambiguation Techniques

We have already given an introduction to the problem and the applications in the introduction. The next section discusses the solutions based on local disambiguation, i.e., figuring out the correct entity based on just the local evidences. Section 3.2 discusses the

intuition behind having a global strategy for disambiguation, and the optimization problem that results from such an objective. The final section summarizes a recent work which pragmatically selects global and local evidences, to get the best of both worlds.

3.1 Local Disambiguation of named entities

3.1.1 Introduction

In local disambiguation, we collect just local evidences for each mention for its disambiguation. This was state of the art until the CSAW[1] paper came along. We start by defining the problem and discussing the general form of solutions. We then provide a short summary of approach followed in Wikify [9] and the famous Milne and Witten paper [6]. A solution based on machine learning[1] concludes the subsection.

3.1.2 Problem definition

We need to disambiguate a mention by collecting the local evidences. The evidences can be anything, POS tags, gender information, dictionary lookup etc. By local disambiguation, we mean that **we cannot use the disambiguation information for any other entities for solving the problem.**

3.1.3 Solutions

Every local disambiguation techniques fall into one of the following two categories[9]

- **Knowledge based**

Derived from the classical word sense disambiguation literature, this technique depends on the information drawn from the definitions provided by the knowledge base. (See Lesk’s algorithm [14]). This is based on the overlap of context with the definitions of each of the candidate senses as given in the knowledge base.

- **Machine Learning based**

This method is based on collecting features from the mention and its surroundings, and training

a classifier to give a verdict on a particular sense being a likely disambiguation of a mention. Machine learning based local disambiguation was almost unanimously adopted by the NED community as the solution for local disambiguation. AIDA changed the scene by introducing a knowledge based local similarity score which works well.

3.1.4 Related Work

Wikify[9] The biggest contribution of this paper is perhaps presenting Wikipedia as the catalog against which were supposed to disambiguate. The paper also identifies two broad methods of doing named entity disambiguation : Knowledge based and data based. Since the paper dates back to 2007, when the problem of NED was not as established, there are a lot of references to the problem of word disambiguation.

Learning to link with Wikipedia[6] This paper defined three different features for disambiguation :

- Commonness : This is the prior defined in Chapter 1.
- Relatedness : Perhaps the biggest contribution of this paper, the relatedness score, gives a measure for determining how similar the two entities are. This measure is based on the number of common inlinks to entities in question. The relatedness measure as defined here has been used in a lot of works. In fact, all the approaches presented in the subsequent subsections use this relatedness score, popular as the Milne-Witten score for finding out entity entity similarity. This score is defined as follows

$$r(\gamma, \gamma') = \frac{\log|g(\gamma) \cap g(\gamma')| - \log(\max\{|g(\gamma)|, |g(\gamma')|\})}{\log c - \log(\min\{|g(\gamma)|, |g(\gamma')|\})}$$

Where

- $g(\gamma)$: Set of wikipedia pages that link to γ
- c : Total number of Wikipedia pages
- $r(\gamma, \gamma')$: Relatedness of topics γ and γ'

The algorithm selects a few unambiguous links in the document, and uses the similarity of the candidates with these unambiguous links as a criteria for disambiguation. Thus, in some sense, although the technique is not totally local, it shies away from doing anything to maintain coherence among the entities that are unveiled and thus we do not call this method a “Global method”, which are discussed in the following subsection.

3.1.5 Machine learning based local disambiguation

As mentioned, there are primarily two approaches for local disambiguation. This subsection discusses a machine learning based local disambiguation method in some detail. This subsection is based on the local disambiguation approach taken in [1].

Definitions We first repeat the definitions for quick reference :

- s : Spot, an Entity to be disambiguated (Christian leader John Paul)
- γ : An entity label value (http://en.wikipedia.org/wiki/Pope_John_Paul_II)
- $f_s(\gamma)$: A feature function that creates a vector of features given a spot and a candidate entity label.

Local compatibility : Feature design The feature function takes the spot and the candidate as arguments.

- The following information about a candidate γ is used
 - Text from the first descriptive paragraph of γ
 - Text from the whole page for γ
 - Anchor text within Wikipedia for γ .
 - Anchor text and 5 tokens around γ
- We now have 4 pieces of information about γ . We take each of these, and apply the following operations with one argument as the spot

- Dot-product between word count vectors
- Cosine similarity in TFIDF vector space
- Jaccard similarity between word sets

Thus, for a candidate - mention pair, we get a total of 12 Features (3 operations, 4 argument pairs).

In addition to these, we also use a sense probability prior as defined in the introduction. A popular way of obtaining the prior is counting the number of times the spot has been linked to a particular entity. For example, the hypertext “Linux” might be linked to the page for the Linux kernel 70% of the times, and to the page for Linux based operating systems rest of the times.

Compatibility Score Once we have the features, we train the classifier by using the following optimization objective :

- Local compatibility score between a spot s and a candidate is given by $w^T f_s(\gamma)$
- w is trained using an SVM like training objective

$$w^T f_s(\gamma) - w^T f_s(\gamma) \geq 1 - \epsilon_s$$

Finding the best candidate

Note that a multi class classifier is not learned for several reasons, all of which can be mapped to the large number of classes.

3.2 Collective Disambiguation of Named Entities

3.2.1 The key intuition

We have seen several different “local” solutions, attempting to solve the problem by collecting evidence around a mention and then using it to disambiguate. Milne and Witten [6] came close to inculcating some sort of coherence, but they couldn’t totally build up the intuition. It was after a wait of 2 years that CSAW [1] took the game to a whole new level by working on the following key intuition :

- A document is usually about one topic

- Disambiguating each entity using the local clues misses out on a major piece of information : Topic of a page

- A page is usually has one topic, you can expect all the entities to be *related* to the topic *somehow*

Michael Jackson : 30 Disambiguations

John Paul : 10 disambiguations

But if they are mentioned on the **same page**, the page is most likely about Christianity, A big hint towards disambiguating **both** of them.

Since the CSAW[1] paper, every work on named entity disambiguation includes a notion of *Topical coherence* in the solution.

3.2.2 Challenges

Though the notion of topical coherence is very natural and intuitive, there are a lot of challenges involved when it comes to actually mapping these intuitions to an optimization problem. We present the challenges involved and the solution given by the CSAW team.

- Capturing local compatibility
 - Create a scoring function to rank possible candidates
- Inculcating topical coherence in the overall objective
 - Define Topical coherence

Out of these two challenges, various solutions to the problem of capturing the local compatibility are presented in Chapter 2. In this subsection, we focus on the problem of collective disambiguation.

3.2.3 The Dominant Topic Model

- Need to define a collective score based on pair-wise topical coherence of all γ_s used for labeling.

Data: A Document d

Result: Annotated document d' with every mention linked to the best candidate entity

foreach mention m in the document **do**

| calculate $\operatorname{argmax}_{c_m \in \Gamma} w^T f_m(c_m)$ where $\Gamma = c_m : c_m$ is a possible disambiguation of m

end

Algorithm 1: Local disambiguation

- The pairwise topical coherence, $r(\gamma_s, \gamma'_s)$ is as defined above.

- For a page, overall topical coherence :

$$\sum_{s \neq s' \in S_0} r(\gamma_s, \gamma'_{s'})$$

- Can be written as clique potential as in case of node potential

$$\exp(\sum_{s \neq s' \in S_0} r(\gamma_s, \gamma'_{s'}))$$

3.2.4 The Optimization objective

With different notations as above, we would like to maximize the following to get the best results.

$$\frac{1}{\binom{|S_0|}{2}} \sum_{s \neq s' \in S_0} r(\gamma_s, \gamma'_{s'}) + \frac{1}{|S_0|} \sum_{s \in S_0} w^T f_s(\gamma)$$

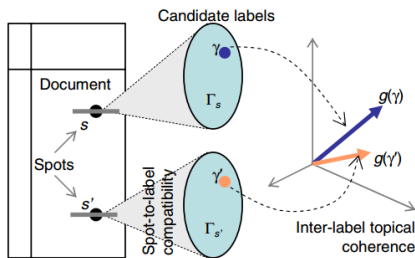


Figure 3: Labels $\gamma \in \Gamma_s, \gamma' \in \Gamma_{s'}$ have to be chosen for spots s, s' to maximize a combination of spot-to-label compatibility scores $NP_s(\gamma), NP_{s'}(\gamma')$ as well as topical similarity between γ and γ' , say, $g(\gamma)^T g(\gamma')$. 3

In verbose, we want that the entity-entity coherence be maximized, while choosing the disambiguation which is the best.

³Reproduced from [1]

3.2.5 Solving the optimization objective

The authors compare 2 different approaches for solving the optimization objective.

- LP rounding approach

$|\Gamma| + |\Gamma|^2$ binary variables were introduced. The first set of binary variables decide the candidate that each mention takes, and the second set has one binary variable for each possible candidate pair. The authors relax this integer programming to a linear programming and then used rounding with a threshold of 0.5 to obtain the best solution.

- Hill climbing

Starting from all assignments set to NA, assignments are done based on local potentials only. The following figure (from the paper) illustrates the process.

```

1: initialize some assignment  $y^{(0)}$ 
2: for  $k = 1, 2, \dots$  do
3:   select a small spot set  $S_\Delta$ 
4:   for each  $s \in S_\Delta$  do
5:     find new  $\gamma$  that improves objective
6:     change  $y_s^{(k-1)}$  to  $y_s^{(k)} = \gamma$  greedily
7:   if objective could not be improved then
8:     return latest solution  $y^{(k)}$ 

```

3.3 Pragmatic combination of Local and Global Disambiguations

3.3.1 Introduction

Recall that Chapter 2 was about local disambiguation. In subsection 3, we saw how global disambiguation can be combined with the overall objective. A

recent work, Robust disambiguation of named entities in text [10], proposes that blindly opting for global disambiguation may not be always right. Consider the sentence : “Manchester will play Madrid in Barcelona”.

All the 3 named entities in the sentence are Cities as well as football clubs. Collective disambiguation may *coerce* all the three mentions to be either football clubs or cities. The work aims to solve this problem by being selective about when to go for collective disambiguation.

3.3.2 Approach

This approach first creates a mention to candidate graph. The sample graph for the sentence “They performed Kashmir written by Page and Plant. Page played unusual chord on his Gibson.” is as shown below :

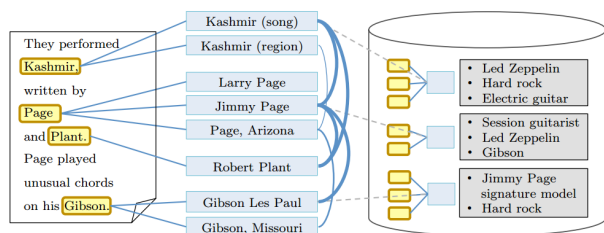


Figure 1: Mention Entity Graph

Having created the graph, we need to assign the edge weights. Clearly, there are 2 kinds of edges involved :

- **Mention - Entity edge** : The authors used a knowledge based approach to assign this weight. This is as outlined in subsection 2. The details about this score are given in [11].
- **Entity - Entity edge** : Milne witten score as defined in subsection 2 is used for this purpose.

With the graph ready, the authors pluck the in a greedy manner such that there is only one edge between each mention and entity.

3.4 Further Readings on Named Entity Disambiguation

For this report, only a small subset of the papers was selected to cover as much ground as possible. The following list may be valuable to the interested readers.

- **Mining evidences for named entity disambiguation** The authors discuss a modified LDA model for gathering more words that are important to disambiguate an entity. Li, Yang, et al. "Mining evidences for named entity disambiguation." Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2013.
- **We have emphasized on Wikipedia as the catalog. The following work presents a general approach** Sil, Avirup, et al. "Linking named entities to any database." Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. Association for Computational Linguistics, 2012.
- **Large scale named entity disambiguation.** Cucerzan, Silviu. "Large-Scale Named Entity Disambiguation Based on Wikipedia Data." EMNLP-CoNLL. Vol. 7. 2007.
- **One of the initial works on NED** Bunescu, Razvan C., and Marius Pasca. "Using Encyclopedic Knowledge for Named entity Disambiguation." EACL. Vol. 6. 2006.
- **Quick entity annotations for short text** Suchanek, Fabian M., Gjergji Kasneci, and Gerhard Weikum. "Yago: a core of semantic knowledge." Proceedings of the 16th international conference on World Wide Web. ACM, 2007.

4 Distributional Statistics of Named entities

Once you have a catalog of things, it makes sense to ask which of these “things” are more important than the others. In fact, one might extend the question and ask, “Which pairs (or triples) of these things appear together on the open web?”. We define several different statistics one might be interested in over these entity catalogs, discuss some applications, propose a baseline method and finally, prepare the ground for the next section by giving an outline of a solution which is aimed at directly providing us with the statistics we are looking for.

4.1 What Statistics?

4.1.1 Which sense dominates for an entity?

For starters, we might want to calculate the number of times a particular “sense” of an entity⁴ is used. For example, the entity Michael Jordan has several disambiguations, : The Professor, Basketballer and the botinist. We want to find out the distribution of occurrences of these senses. We call this number the sense prior.

It is important to note that Entity Prior is different from mention prior, which is the fraction of times a mention links to a particular entity. For example, the text “Gingerbread” might refer to several different concepts; from perhaps the most famous Android 2.3 to the novel. Mention prior is to find out how many *Gingerbreads* mentions on the web refer to Gingerbread the Operating system. Entity sense prior would tell us how frequent is Gingerbread the OS compared with Gingerbread the novel.

$$\text{Sense Prior}(S_i, E) = P(E \text{ appears as the } i^{\text{th}} \text{ sense}) = P(S_i | “E”) \quad (1)$$

Where S_i is the i^{th} sense⁵ of the entity E.

⁴Please note that we refer to entity in general terms. For example, any object having a YAGO id is an entity

⁵ i^{th} disambiguation in Wikipedia parlance

4.1.2 How often do the 2 entities appear together?

A second interesting statistic would be to count how many times do two given entities, taking two given senses appear together. For example, We might want to know how many times does Nokia <http://en.wikipedia.org/wiki/Nokia> appears with Gingerbread [http://en.wikipedia.org/wiki/Gingerbread_\(operating_system\)](http://en.wikipedia.org/wiki/Gingerbread_(operating_system))

We call these counts Entity bi grams. We note that in contrast to word bi-grams and relational grams [12], entity bi grams are symmetric, and there is no obvious use case where we might need to know the order dependent occurrence count of the entities. However, such a formulation will lead to a sparse distribution, since each count will have to be normalized by the total number of entity bigrams. We thus define the entity bi gram count as follows :

$$\text{Entity Bi Gram}(E2|E1) = P(E2 \text{ follows } E1) = P(E2|E1) \quad (2)$$

We propose an application of Entity bi grams for finding out important entities motivated by [12].

4.2 Applications

We list a few applications of the sense prior and outline an application of the entity bigrams.

4.2.1 Sense Prior

A prior over the sense will be helpful in many applications related to information retrieval.

- Entity Querying
- Knowledge graph based searching

4.2.2 Entity Bigrams

Given an entity, we want to find out other important entities that are related to it. For example, given an entity **Barack Obama, President of the USA**, we need to provide top 10 entities that are “close” to Barack Obama the President. Since the solution is only a slight modification of the solution presented

in [13] for finding out important relations, we only sketch an outline here.

For the entity we are interested in, Say X, create a node. Now attach to the node X all the entities E for which $P(E|X) > \epsilon$ where ϵ is some threshold. Let the weight of the edge be defined as

$$P(E|X) + P(X|E) \quad (3)$$

We then apply personalized page rank on the X sub graph, starting with X having a page rank of 1 and other nodes having a page rank of 0. We can then sort the nodes based on their page ranks upon convergence.

4.3 Baseline Approach : Label and Collect

How do we collect the aforementioned statistics? This question shouldn't be too difficult to answer now. The whole of part 3 was dedicated towards tagging entity mentions in the text. We can use any of the methods (for example, AIDA can be set up as a rest service) to tag the corpus, and then iterate over the corpus to collect these statistics in single pass.

4.4 Solution based on estimating class ratios

While estimating class ratios by doing per mention disambiguation seems pretty intuitive, we are doing more than what we need to do. We are not interested in what each mention disambiguates to, a count of how many times does a particular entity appears is the desideratum. There are 3 different methods in the open domain for directly estimating the class ratio [20], without going through the label and collect route. In particular, [20] discuss a solution based on maximum mean discrepancy and proves some upper bounds on errors.

If mmd really works, we should expect better estimation of the sense prior and the entity grams. The next section outlines the mmd based solution and how mmd may be used to estimate the sense priors for different entities.

5 MMD for estimating ratios of named entities in text

This section discusses the MMD approach for direct estimation of class ratios [20]. We first provide an intuition for the solution, follow it up with some results

5.1 Introduction

The following hypothetical example is aimed to capture the gist of class ratio estimation using mmd. Suppose that in a factory producing balls, there are 3 different ball production machines, (say) A, B and C. Since neither of the machines is perfect, they do not produce spherical balls. Rather, the balls are ellipsoids. Thus, for each ball, we have 3 different features corresponding to the three semi-axes. Since all the machines are different, they have their own unique view of how balls should look like, and thus we expect that the semi axes are a good way of telling the machine which produced a given ball.

Also assume that for all the 3 machines, we also have the most likely (expected) semi axes measures of the balls produced by them. Let us call these $\phi_a(x)$, $\phi_b(x)$, and $\phi_c(x)$. These are the *expected feature weights*.

Suppose we are given a 150 balls produced from these three machines. For 120 balls out of them, we know the machine from which the ball was produced. For the remaining 30 balls, we are asked to give an estimate of how many balls came from machine A, B and C.

How do we do this? Of course, we can learn a classifier from the 120 known instances and then learn the label each of the 30 balls and collect counts (label and collect approach). MMD takes the following route to reach the solution.

Suppose we are magically given the true class ratios, say, θ_a , θ_b and θ_c . Let ϕ be the average of the semi axes of the 30 balls. Let ϕ' be defined as

$$\phi' = \phi_a * \theta_a + \phi_b * \theta_b + \phi_c * \theta_c \quad (1)$$

Clearly, we would expect ϕ to match ϕ' .

Note that we don't really know the θ_s , but all is not lost since we know what to look for; we look for

the thetas that minimize :

$$\|\phi_a * \theta_a + \phi_b * \theta_b + \phi_c * \theta_c - \phi'\|^2 \quad (2)$$

While ensuring that :

- All the θ s sum to 1.
- All the θ s are non negative.

This is the motivation behind MMD for class ratio estimation.

5.2 MMD Formulation

With the above example by our side

5.2.1 Problem Definition

We reproduce the problem statement from [20]

- Let $X = x \in R_d$ be the set of all instances and $Y = 0, 1, \dots, c$ be the set of all labels.
- Given a labeled dataset $D(\subset X \times Y)$, design an estimator that for any given set $U(\subset X)$ can estimate the class ratios $\theta = [\theta_0, \theta_1, \dots, \theta_c]$ Where θ_y denotes the fraction of instances with class label y in U

5.2.2 Objective

- Match two distributions based on the mean of features in the hilbert space induced by a kernel K .
- Assume that distribution of features is same in both training and test data $P_U(x|y) = P_D(x|y), \forall y \in Y$
- Thus, the test distribution must equal $Q(x) = \sum_y P_D(x|y)\theta_y$
- Let $\bar{\phi}_y$ and $\bar{\phi}_u$ denote the true means of the feature vectors of the y th class and the unlabeled data

- Suppose we somehow get the true class ratios θ . The true mean of the feature vector of the unlabeled data can then be obtained by $\sum_y \theta_y \bar{\phi}_y$.

- So ideally, $\sum_y \theta_y \bar{\phi}_y = \bar{\phi}_u$

The objective thus is

$$\operatorname{argmin}_{\theta} \sum_{y \in Y} \|\sum_y \theta_y \bar{\phi}_y - \bar{\phi}_u\|^2 \quad (3)$$

Such that

- $\forall y, \theta_y \geq 0$
- $\sum_{y=0}^c \theta_y = 1$

Interesting discussion on theoretical bounds on the error in the class ratios thus predicted and methods for learning Kernel can be found in [20]

5.2.3 Estimating entity ratios using MMD

Given a corpus with mentions identified (using, say [5]), we want reliable estimates of frequency of each of the entities. In this subsection, we gloss over the solution.

• Features

Each mention has several candidate disambiguations. This gives one way of formulating the features. For each mention, we can have a (sparse) feature vector having non zero scores for the candidates.

• Training data

Can be obtained by splicing the named entity disambiguation pipeline of any of the popular named entity disambiguators. [21] discusses how to achieve this for AIDA, a popular named entity disambiguator.

6 Conclusion

The potential of open web can only be harnessed to its full extent by adding structure to it. The process involves creating structured repositories derived

from the web that can answer interesting questions pertaining to entities that exist on the web.

Many such smart applications that rely on structured web will rely on frequencies of occurrence of the former. The report has been a buildup to achieving that. We started by briefing what knowledge bases are. In the second part, we introduced the problem of disambiguating the mentions of named entities and presented solutions roughly spanning last 8 years of research in the field.

In the third part, we elaborated on what is meant by aggregate statistics and presented several applications of the same. We presented maximum mean discrepancy approach for class ratio estimation via an example and discussed the problem formulation. We briefly outlined how mmd can be applied for estimating occurrence statistics of entities.

State of the art approaches for named entity disambiguation brush the figure of 90% accuracy. It is thus expected that the focus of the community will now shift to making the process of disambiguation faster and integrating the disambiguators in the search pipeline. It remains to be seen how approaches based on direct estimation of entity occurrence ratios perform in comparison with the standard tools, both in terms of speed and accuracy.

7 Acknowledgement

This report is a summary of selected readings undertaken while working under the guidance of Prof. Sunita Sarawagi on application of mmd for collecting occurrence statistics of entities on the web. I would like to thank her for the guidance. It was immensely helpful in gaining the understanding required for writing this report.

Thanks to Mr. Arun Iyer for all the help with understanding maximum mean discrepancy and its implementation.

Lectures by Prof. Soumen Chakrabarti provided useful insights into the problem of named entity disambiguation.

References

- [1] Kulkarni, Sayali, et al. "Collective annotation of Wikipedia entities in web text." Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2009.
- [2] <http://www.cse.iitb.ac.in/~soumen/OWI/Slides/>
- [3] William Cohen's Survey available at 2
- [4] http://en.wikipedia.org/wiki/Named-entity_recognition
- [5] <http://nlp.stanford.edu/software/CRF-NER.shtml>
- [6] Milne, David, and Ian H. Witten. "Learning to link with wikipedia." Proceedings of the 17th ACM conference on Information and knowledge management. ACM, 2008.
- [7] ws <http://www.worldwidewebsite.com/>
- [8] <http://en.wikipedia.org/wiki/Wikipedia:Statistics>
- [9] Mihalcea, Rada, and Andras Csomai. "Wikify!: linking documents to encyclopedic knowledge." Proceedings of the sixteenth ACM conference on Conference on information and knowledge management. ACM, 2007.
- [10] Hoffart, Johannes, et al. "Robust disambiguation of named entities in text." Proceedings of the Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2011.
- [11] Hoffart, Johannes, et al. "Kore: keyphrase overlap relatedness for entity disambiguation." Proceedings of the 21st ACM international conference on Information and knowledge management. ACM, 2012.
- [12] Balasubramanian, Niranjan, Stephen Soderland, and Oren Etzioni. "Rel-grams: a probabilistic model of relations in text." Proceedings of the

- Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction. Association for Computational Linguistics, 2012.
- [13] Balasubramanian, Niranjan, Stephen Soderland, and Oren Etzioni Mausam. "Generating Coherent Event Schemas at Scale." Proceedings of the Empirical Methods in Natural Language Processing. ACM (2013).
- [14] Michael Lesk. 1986. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In Proceedings of the 5th annual international conference on Systems documentation (SIGDOC '86), Virginia DeBuys (Ed.). ACM, New York, NY, USA, 24-26. DOI=10.1145/318723.318728 <http://doi.acm.org/10.1145/318723.318728>
- [15] <http://wordnet.princeton.edu/wordnet/>
- [16] Suchanek, Fabian M., Gjergji Kasneci, and Gerhard Weikum. "Yago: a core of semantic knowledge." Proceedings of the 16th international conference on World Wide Web. ACM, 2007.
- [17] Auer, Sören, et al. "Dbpedia: A nucleus for a web of open data." The semantic web. Springer Berlin Heidelberg, 2007. 722-735.
- [18] Nakashole, Ndapandula, Gerhard Weikum, and Fabian Suchanek. "PATTY: a taxonomy of relational patterns with semantic types." Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. Association for Computational Linguistics, 2012.
- [19] Bollacker, Kurt, et al. "Freebase: a collaboratively created graph database for structuring human knowledge." Proceedings of the 2008 ACM SIGMOD international conference on Management of data. ACM, 2008.
- [20] Iyer, Arun, Saketha Nath, and Sunita Sarawagi. "Maximum Mean Discrepancy for Class Ratio Estimation: Convergence Bounds and Kernel Selection." Proceedings of The 31st International Conference on Machine Learning. 2014.
- [21] Using Structured learning for named entity disambiguation, <http://www.cse.iitb.ac.in/~amanmadaan/docs/rnd/structentity.pdf>