Semi-Supervised Multiple Disambiguation

Kambiz Ghoorchian
School of Electrical Engineering
Royal Institute of Technology (KTH)
Stockholm, Sweden
Email: ghoorian@kth.se

Fatemeh Rahimian
School of Electrical Engineering
Royal Institute of Technology (KTH)
Stockholm, Sweden
Email: rahimian@kth.se

Sarunas Girdzijauskas
School of Electrical Engineering
Royal Institute of Technology (KTH)
Stockholm, Sweden
Email: sarunasg@kth.se

***Abstract*—Determining the true entity behind an ambiguous word is an NP-Hard problem known as *Disambiguation*. Previous solutions often disambiguate a single ambiguous mention across multiple documents. They assume each document contains only a single ambiguous word and a rich set of unambiguous context words. However, nowadays we require fast disambiguation of short texts (like news feeds, reviews or Tweets) with few context words and multiple ambiguous words. In this research we focus on *Multiple Disambiguation (MD)* in contrast to *Single Disambiguation (SD)*. Our solution is inspired by a recent algorithm developed for SD. The algorithm categorizes documents by first, transferring them into a graph and then, clustering the graph based on its topological structure. We changed the graph-based document-modeling of the algorithm, to account for MD. Also, we added a new parameter that controls the resolution of the clustering. Then, we used a supervised sampling approach for merging the clusters when appropriate. Our algorithm, compared with the original model, achieved 10% higher quality in terms of F1-Score using only 4% sampling from the dataset.**
***Keywords*-Graph Algorithms; Parallel Processing; Coreference resolution; Diffusion Clustering**

I. Introduction

Given a set of input documents containing an ambiguous word which refers to multiple different entities, *disambiguation* is the task of clustering these documents into groups that refer to a distinct entity each. Disambiguation serves as a preprocessing step to other text processing problems including *Multi-Document Summarisation* and *Question Answering*. There has been a large body of work addressing this problem, including both supervised and unsupervised methods. While still a complicated task, disambiguation generally works based on a simplifying assumption that there is a single ambiguous word which is known in advance. This assumption is not generally valid in real applications. To find out which words are ambiguous the input data often needs to undergo some preprocessing. The result of such preprocessing is a set of ambiguous words which are mentioned across different documents in the input set. Next, existing algorithms need to be executed once per ambiguous word. The drawback of such an approach is not only that we need to spend more resources and time for completing the task, but also for each run associated with one ambiguous word, we have to drop all the other ambiguous words, thus weaken the context of the target ambiguous word. On the contrary, if we could disambiguate all the ambiguous words together, then the shared context between them would have the potentials to improve the accuracy of the final clustering. Moreover, a single run could give us the clustering for all the ambiguous words at the same time.

In this paper, we address the problem of *Multiple disambiguation (MD)* in contrast to *Single Disambiguation (SD)*, by building a co-occurrence graph that incorporates all the ambiguous words as well as non-ambiguous words in one single graph. Our approach is a semi-supervised learning method that exploits a strong property of human language, that is, “a word is characterized by the company it keeps” [1]. This property is also studied and approved in many other works [2]. Therefore, we build a co-occurrence graph that captures the similarities across documents and provides us with a graph in which a group of documents that refer to the same real-world entity belong to the same topological cluster. Then, we run a community detection algorithm to find the natural clusters in the graph. Our work in this paper is based on a recent algorithm [3] developed for single disambiguation across documents.

We show how [3] can be used for multiple disambiguation. Our main contributions are, (i) manipulation of the graph representation so that the model adjusts for MD, and (ii) development of a new parameter that controls the resolution of the clustering algorithm. Our approach resulted in 10% higher F1-Score compared with the previous model using only 4% sampling from the dataset.

II. Related Work

Numerous solutions have been proposed to the problem of disambiguation [4], [5], [6], [7], [8]. Traditional solutions [4], [5], [6], are based on pairwise similarity clustering of mention vectors extracted from linguistic features. Rapid growth of publications over the internet turned disambiguation into a web-scale document analysis problem, confronting traditional solutions with major challenges regarding the scalability and efficiency.

Recent advancement in parallel processing of massive data offered new opportunities to overcome those challenges. Proposals in this category generally apply a distributed pairwise similarity clustering using *Hadoop MapReduce* [7], [9], [10], inference based *Monte Carlo Markov Chain* [8] or multi-pass clustering [11]. However, these solutions still suffer from their underlying vector based document representation, due to two reasons. First, the high dimension of the vectors (equal to the total number of unique words in the whole dataset), significantly increases the complexity of the processing for every single comparison. Second, the inherent pairwise similarity comparison, imposes a large number of unnecessary comparisons between documents with non overlapping contexts.

Addressing these issues, requires more efficient techniques rather than the traditional pairwise similarity approaches. More complex solutions based on Conditional Random Field (CRF) [12], Discriminative hierarchical entity partitioning [13] and diffusion based graph clustering [3], are the most recent proposals of this type.

Current solutions, to our knowledge, are designed for single disambiguation. However, in this work we want to disambiguate multiple ambiguous mentions synchronously. Therefore, we decided to extend the diffusion based graph clustering [3] that has been shown the most successful solution in terms of quality and scalability. In addition, this solution considers the graph representation as its document modeling approach which makes it even easier to be implemented for MD. There exist various distributed community detection algorithms [14] over graphs, however since the diffusion algorithm has been tailored for the specific document-click graph representation in this model, we decided to use the same community detection algorithm and adjust it for MD.

III. Motivation

This section underpins the essentials of SD and explains our motivations behind MD. The following terms will be frequently used in this section and the rest of the paper:

– *Document:* is a sentence, a paragraph, or any coherent and meaningful context constructed from a set of words.
– *Entity:* is a distinctive and independent existence in the world like Paris, the capital city of France.
– *Ambiguous-Mention:* is a word or a phrase, like “Paris”, that can refer to multiple entities, e.g. Paris Hilton and the city of Paris.
– *Context-Words (Context):* are a set of words or phrases in a sentence, which are assumed to be unambiguous and are used for the disambiguation.

Single Disambiguation (SD) is traditionally defined as *“the task of categorizing a set of documents containing a single mention into a set of groups such that all the documents in each group refer to the same entity”*. For example, having a set of documents containing the ambiguous phrase *“John Smith”*, SD clusters the documents into different groups such that all the “John Smiths” in each group refer to the same entity in the real world. Disambiguation is one of the most important preprocessing tasks for many language processing problems. For example, it can be used for *Entity Linking* [15] for information retrieval in question answering. It can also be used in automatic text summarization or author identification [16]. Trend detection and hot topic extraction in news feeds and linguistic content based social media, like Twitter, are other use-cases where disambiguation is required.

SD relies on a strong assumption, requiring each document

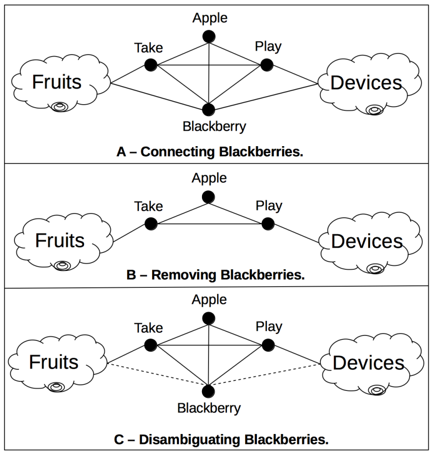


Fig. 1. Single vs Multiple disambiguation.

to contain a single ambiguous mention that is representative of the main topic of the document. Such disposition may apply to specific cases like the categorization of the abstract of scientific papers where each document contains minimum amount of ambiguity and follows an explicit argumentation. Nevertheless, it is not applicable in Web-based documents, like *Twitter Posts* or *Product Reviews*, where sentences often contain more than a single mention and a small set of context words. To apply SD over such datasets, we must either remove all the other ambiguous words or assume that they are unambiguous. The first approach, reduces the overall knowledge by eliminating some existing information, and the second, increases the number of false positives by wrongly combining unrelated documents. Moreover, both approaches will impose an extra overhead by enforcing separate disambiguation of other ambiguous mentions.

Let’s illustrate these drawbacks further, with an example. Suppose we have the following sentence with two ambiguous words “Apple” and “Blackberry”, that can either be a ”Fruit” or a ”Device”, and two unambiguous context words “play” and “take”, that are related to “Device” and “Fruit” communities respectively (Figure 1).

– “Can you take a *Blackberry* while playing with an *Apple*?”

Disambiguation of the “Apple” in this sentence requires a comparison with similar sentences that are less ambiguous. Such comparison is not possible, since the “Blackberry” is also ambiguous. SD, currently ignores this fact, and assumes that all the “Blackberries” are unambiguous and refer to the same entity. This approach, as we can see (Figure 1A), increases the uncertainty regarding the choice of the “Apple” by adding noise between the two communities. Another solution is to remove all the “Blackberries” (Figure 1B), that also decreases the overall entropy by eliminating the existing information. Therefore, the only way to prevent such side effects is to consider “Blackberry” as an ambiguous mention (Figure 1C) and

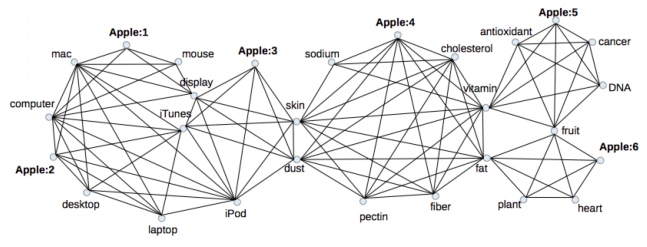


Fig. 2. Disambiguation graph constructed from sentences in Table I.

try to find its real affinity to either of the communities. This way, the assignment of “Apple” will equally depend on the size of each community and the affinity of the “Blackberry” to either of them. This is a new type of disambiguation we define as *Multiple Disambiguation (MD)*. The next section will explain the details of our algorithm developed for MD as an extension to [3].

IV. Solution

Similar to [3], we represent documents as graphs and extract similarities as topological communities inside the graph. The main idea is to first, create a graph in which similar documents construct dense topological areas called communities and then, extract those communities using a diffusion based clustering algorithm. This model was initially designed for SD. However, we changed the graph representation and the clustering algorithm to account for MD. This section covers both steps by first, presenting the basic idea from [3] and then, justifying our motivations behind different modifications and their implementation.

A. Graph-based Document Modeling

A graph $G = (V, E)$ is a set of vertices V that represents objects and a set of Edges E that represents relations between the objects (like users and their friendship relations in Facebook). An edge can be *weighted/unweighted* and *directed/un-directed*. The weight indicates the significance of the relation and the direction specifies the orientation. An edge from a vertex to itself is called a *loop* and two or more edges connecting the same vertices are called *multiple edges*. In our algorithm, we use weighted and un-directed graphs that do not contain loops and multiple edges.

There are different ways for representing documents as graphs [17]. Authors in [3] use a bag-of-words approach, where they consider words as objects and co-occurrence of two words in the same document as their relation. They construct the graph as follows,

- Extract the mention and a set of context words (bag of words) from each document.
- Assign a vertex to each word such that:
    - Each mention gets assigned to a unique vertex.
    - Same context words get assigned to the same vertex.
- Connect all the vertices based on their co-occurrence in the same document.

TABLE I
Sample sentences for single disambiguation.

Row	Document
1	**Apple** sells a variety of computer accessories for Macs, including Thunderbolt display and Magic mouse.
2	**Apple** designs and creates iPod, iTunes, Mac laptop and Desktop computers.
3	**Apple** skin protects your iPod. It also fits your iTunes and is enhanced with anti dust treatment. It gives sharper look to the display.
4	**Apple** contains no fat, sodium or cholesterol and is a good source of fiber. Its skin has the highest concentration of vitamins and is mainly covered with pectin and dust.
5	Like many fruits, **Apple** contains vitamins as well as other antioxidants, which may reduce the risk of cancer by preventing DNA damage.
6	The **Apple** has the most diverse fruit plants in the world. It was found to increases the burn of fat and reduces the risk of heart attack

TABLE II
Sample sentences for multiple disambiguation.

Row	Document
1	I have been given an **Apple** pancake when I visited **Orange** offices to fix their phone system
2	An **Apple** iPhone with an **Orange** subscription is the best combination for my office.
3	When I told my parents that I want an **Apple**, my mom said, ”there is an **Orange** at home eat that first”.
4	**Apple** and **Orange** should team up and make a phone called the Pie.
5	We have **Apple** pie and **Orange** pancake are usual sweets in every home.

Figure 2 shows a sample graph generated from sentences in Table I. As we can see, a unique vertex is assigned to each mention “Apple:1”, “Apple:2”, etc., and a single vertex to same context words like “skin” and “dust”.

We use the same bag-of-words approach for document representation. However, since documents in MD may contain more than a single ambiguous mention, a separate unique representation scheme for each mention is required. For example for the set of documents in Table II with “Apple” and “Orange”, the graph, shown in Figure 3, contains different vertices for “Apples” and “Oranges” as “Apple:1”, “Apple:2”, etc. and “Orange:1”, “Orange:2”, etc. respectively. This representation simplifies our evaluation by maintaining the clustering information for different mention groups. We will refer to that in the evaluation part.

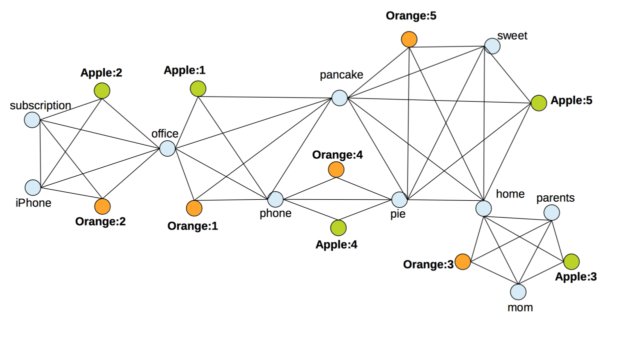


Fig. 3. Disambiguation graph constructed from sentences in Table II.

B. Diffusion-based Clustering

Clustering aims to extract similarities as dense areas in the graph. Authors in [3], use different colors to represent different clusters and employ an agglomerative clustering approach to first, assign a unique cluster (color) to each document and then, gradually merge similar clusters. The algorithm consists of two steps, an initialization round followed by a number of diffusion iterations until a termination condition is satisfied.

During the initialization every vertex is assigned one or more colors depending on the number of documents that its corresponding word has appeared in. For example mentions (like "Apple:1") and unique context words (like "heart"), in Figure 2, are assigned a single color, whereas shared context words, like "dust", that appeared in different documents, are assigned multiple colors.

After the initialization, every vertex i must communicate with its immediate neighbors and perform the following tasks in each iteration:

1) Determine its main cluster (*Dominant color*), which is the strongest (has maximum quantity) in i's neighborhood.
2) Strengthen its position in that cluster by preserving a portion of the dominant color.
3) Expand the boundary of its cluster by diffusing a portion of the dominant color.
4) Send away all the other colors that are non-dominant.

To find the dominant color, vertex i first, calculates the total amount of each color k as $S_i^{(k)}$ in its vicinity

$$S_i^{(k)} = C_i^{(k)} + \sum\nolimits_{j \in N_i} C_j^{(k)}, \tag{1}$$

that is the current amount of k in i, $C_i^{(k)}$, plus the sum of color k in i's immediate neighbors $\sum_{j \in N_i} C_j^{(k)}$. Then, vertex i finds its dominant color D_i as the color with maximum quantity in its vicinity,

$$D_i = \underset{k}{\operatorname{argmax}}\ S_i^{(k)}. \tag{2}$$

Authors, developed a new parameter called γ to preform the second and the third steps. γ indicates the amount of the dominant color that each vertex must keep.

Non-dominant colors should also be conducted towards their corresponding clusters to first, reinforce the sustainability of those clusters and second, prevent their interference in stability of the other clusters. This is the main goal of the fourth step. Authors assigned *Roles* to vertices for that purpose. The role R_i, of the vertex i will be *Interior*, if all the neighbors of i have the same dominant color as i, or *Boundary*, Otherwise,

$$R_i = \begin{cases} Interior & \forall j \in N_i, \quad C_j^{(D)} = C_i^{(D)} \\ Boundary & Otherwise \end{cases}$$

Boundary vertices diffuse non-dominant colors to their neighborhood since they are directly connected to the corresponding clusters. However, interior vertices that are disconnected from those clusters need another transformation mechanism. Authors developed a *Repository* for that purpose. Repository is a shared memory data structure that contains a bucket for each color. Every interior vertex, in each iteration, puts all its non-dominant colors in the repository and takes δ percent of the color that is the same as its dominant color from the repository. δ is a parameter that ensures fare distribution of each color among all the interior nodes of the corresponding cluster in each iteration. Appropriate values of α and δ must be empirically determined to achieve the best clustering. The diffusion process terminates in an iteration where less than a certain fraction of vertices changed their clusters.

These strategies have been designed such that the vertices act as progressively as possible to achieve the maximum possible clustering resolution in SD. However, as we can see in Figure 3 graphs in MD are more complex than SD. Our experiments show that this complexity causes higher number of false positives for mentions that refer to multiple entities from the same topic. For example, the ambiguous mention "Clinton", referring to either "the Hilary Clinton" or "the Bill Clinton", often categorizes into same cluster, since both entities belong to the same topic, namely politics. Based on that, we developed a new parameter, called *Attachment* α, to prevent these false positives by controlling the resolution of the clustering,

$$\alpha_i = \frac{\left| N_i^{(D)} \right|}{|N_i|}. \tag{3}$$

Attachment is interpreted as the ratio between the number of neighbors of i with the same dominant color to the total number of $i's$ neighbors. The value of α is an indication of the affinity of vertices to clusters and therefore, is used to assign roles to vertices as follows,

$$R_i = \begin{cases} Interior & \alpha_i > Threshold \\ Boundary & Otherwise. \end{cases}$$

Thus, a vertex will be interior if its attachment to the cluster of its dominant color is greater than a specific threshold.

The effectiveness of α as a resolution factor in our model is due to the following empirical observations:

1) Resistance of a cluster depends on the number of vertices in that cluster.
2) Changes in the attachment α significantly affects the number of members in a cluster.

Figure 4 depicts these observations. The figure shows a summary (75 iterations) of execution of the algorithm for an MD dataset over different thresholds of α. The number of interior vertices (upper) are plotted against the number of clusters (lower) for different values of α. Numbers on the y-axis are in log scale and line styles represent different thresholds. There is a significant change around iteration 3 in the upper plot that shows the beginning of the clustering. The variation of the number of interior vertices in this iteration together with the fixed number of clusters on the same iteration in lower plot, shows a negative correlation between α and the number of

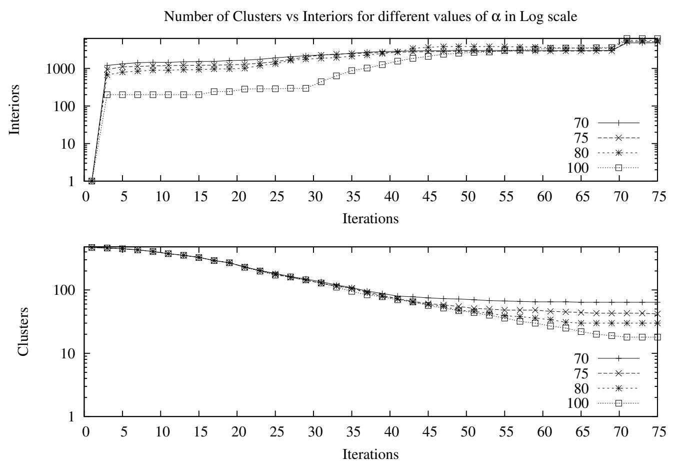


Fig. 4. Analysis of the effect of the *Attachment* α parameter on the behavior of the algorithm with respect to the number and the size of the clusters.

vertices in a clusters (second observation). In other words, the figure shows that lower values of α results in higher number of initial interior vertices that consequently creates stronger clusters. Moreover, the number of remaining clusters, after iteration 70 in the lower plot, indicates that higher initial number of interior vertices prevents the monopolistic behavior of the bigger clusters by increasing the resistance of the smaller ones (the first observation). In fact the parameter α is acting as the *Resolution Factor* of the algorithm, allowing us to control the depth of the clustering. We use α to extract more confident results by having a larger number of high quality clusters (high precision and low recall) rather than fewer number of low quality clusters (low precision and high recall). The number of clusters is tuned in a way that the clusters still have high precision and are not too many so that we can manually detect the topics for each cluster in a scalable manner. Then, we merge those clusters that fall into the same topic by relying on preexisting knowledge of the ground truth or by manual investigation. We developed an appropriate sampling mechanism that provides us with the highest increase in recall while keeping the precision high. After applying the algorithm using our new parameters and reaching the stable state, we first, normalize the number of mentions in each cluster with the total number of mentions in the whole data set, then, randomly take r samples from clusters with normalized size greater than a specific threshold s. Afterwards, we manually tag the sampled documents based on the ground truth and merge their corresponding clusters. In our experiments, we empirically achieved best clustering result by taking $r = 4\%$ samples from $s = 3\%$ higher populated clusters, that will be explained in the experiments section.

V. Experiments

A. Dataset Construction

An appropriate dataset for MD requires documents with more than two mentions. Current baseline datasets like *John Smith* [4] and *Person-X* [6] do not satisfy this property. To evaluate our algorithm, we constructed a synthetic dataset with following properties:

1) Contains overlapping categories, and
2) Reflects natural distribution of ambiguous mentions.

We selected fruit names, used by companies, like *Apple*, *Blackberry* and *Orange* as our ambiguous mentions to satisfy the first property. To maintain the second property, we used *Wikipedia* inter-links, since they resemble main features of real world datasets like term frequency and entity distribution. Wikipedia inter-links are URLs directing from one Wikipedia page to another. We used a DBpedia dump of the whole English Wikipedia from 2014 to construct our dataset in the following five steps,

1) We chose 6 different entities from Wikipedia *Apple-Fruit, Apple-inc., Blackberry-fruit, Blackberry-inc., Orange-fruit* and *Orange-inc.*
2) Crawled all the other Wikipedia pages containing a link to the corresponding Wikipedia page of each entity.
3) Extracted the URL as the mention and its surrounding paragraph as the document.
4) Applied the following refinements to each document to extract the corresponding context words:
   a) Cleaning: by removing all the wild cards and URLs.
   b) POS-Tagging: using Stanford POS-tagger [18] .
   c) Filtering: extracted ”Nouns (Singular and Plural)” and “Named Entities”.
5) Enforced the constraint to keep all unique documents with more than 20 extracted context words.

TABLE III
Entity distribution on the datasets.

	Number of Documents	
Entity	Before refinement	After refinement
Apple fruit	260	53
Apple inc.	1362	316
Blackberry fruit	87	24
Blackberry inc.	226	41
Orange fruit	77	13
Orange inc.	194	24

Table III presents the details of the dataset. It shows the number of documents for different entities before and after enforcing the constraint on the number of context words (step 5). In general, the dataset contains 471 documents. As mentioned in the solution, we use a bag-of-words approach, by removing the function words, to create a graph from those documents. The graph contains 6431 vertices, including mentions and context words, and 351780 edges.

B. Metrics

1) B-Cubed [4]: is an evaluation score that considers precision and recall for each document in isolation and computes a weighted sum over the entire set of documents as the total *Precision* and *Recall*. Precision and recall are calculated as the ratio between the number of documents correctly classified (TP), with respect to a specific class c, to the total number of documents classified in c, $(TP + FP)$, and the total number of documents exist in c $(TP + FN)$ respectively,

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

Total precision and recall are then computed as the weighted sum of precision and recall respectively, over the total set of documents:

$$TotalPrecision = \sum_{i=1}^{N} w_i \times P_i, \quad TotalRecall = \sum_{i=1}^{N} w_i \times R_i$$

Applying B-Cubed to the whole dataset in MD creates incorrect results. The reason is that documents in MD contain more than a single mention and two different mentions referring to two different entities can be categorized into the same group. Suppose we have the following set of documents from two ambiguity groups (Apple and Blackberry):

$$D = \{AF_1, AF_2, AC_1, AC_2, AC_3, BF_1, BF_2, BC_1\},$$

where A stands for Apple, B for Blackberry, F for Fruit and C for Company. A categorization of these documents into two groups including:

$$\{AC_1, AC_2, AC_3, BC_1\} \ and \ \{AF_1, AF_2, BF_1, BF_2\},$$

will be evaluated as 66.7% B-Cubed, which is not intuitively correct. This result, in fact, shows a perfect classification of documents in each ambiguity group. All the documents referring to "Apple Company" are totally separated from those referring to "Apple Fruit", similarly for Blackberries. To prevent this effect, we compute a separate B-Cubed value for each mention group. Therefore, the evaluation result of the disambiguation in previous example, will be 100% B-Cubed for both Apple and Blackberry. Mention groups, as explained in part A of the solution, are inherently preserved in our graph representation, that simplifies the distinctive application of B-Cubed in our model.

2) V-Measure [19]: is an entropy based evaluation metric for measuring the quality of the clustering with respect to *Homogeneity* and *Completeness* of the results. Assume the set of true classes $C = \{c_1, c_2, ...\}$ and the clustering result $K = \{k_1, k_2, ...\}$. Homogeneity measures the distribution of the classes given a specific clustering whereas completeness shows the integration of the members of each class. A clustering with a single cluster for each class results in the highest homogeneity and the lowest completeness. In contrast, a single clustered result features the highest completeness and the lowest homogeneity. V-Measure is computed as the weighed (β) harmonic mean of the two values as the following:

$$V_\beta = \frac{(1 + \beta) * h * c}{(\beta * h) + c}$$

According to [19], V-Measure performs better than F-Score because, it contemplates the conditional distribution of all the classes thus, captures irregularities and matching of the classes across the result. In other words, V-Measure is sensitive to the diversity of classes in all the clusters and prefers the result with lower diversity of classes in each cluster. We use this measure to compare our approach with the results from the original diffusion algorithm in [3].

TABLE IV
EVALUATION COMPARISONS FOR $\alpha = 100\%$ VS $\alpha = 75\%$ OUR MODEL TOGETHER WITH SAMPLING.

α	B-Cubed	V-Measure
100%	62.7%	29.0%
75%	52.7%	28.8%
75% + sampling	**71.5%**	**40.0%**

C. Experiments

All the experiments are run on a single machine with 8 cores and 16 Gb RAM and the algorithm was developed on GraphChi, C++ [20], version 3.1.3, the parallel processing framework. We developed two different experiments using the "Apple-Blackberry-Orange" dataset explained in Section 5.1. Values of all the parameters, except α, are identical between the two experiments. We assigned $\gamma = 66\%$ of the dominant color and $\delta = 0.001\%$ of the color in repository. In our first experiment we used $\alpha = 100\%$ to mimic the situation of the original setting over MD and in the second experiment we used $\alpha = 75\%$ with which our best result acquired using a qualitative approach. Finally, we examined different sampling ratios to find the minimum number of samples required to achieve the best merging result. $s = 3\%$ provided this outcome by sampling only $r = 4\%$ of the whole dataset.

VI. RESULTS

A. Model Comparison

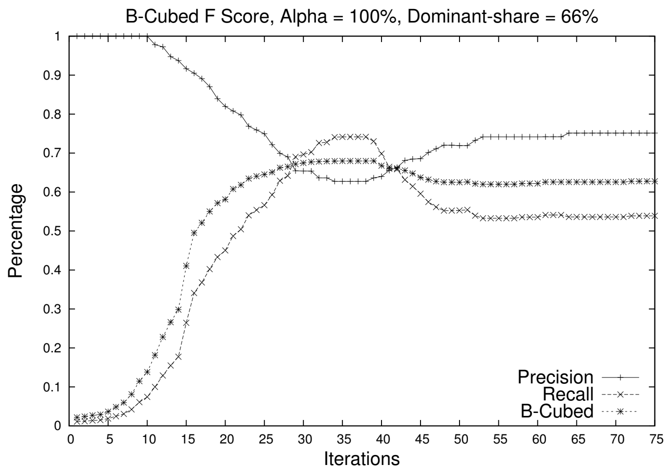
The first two rows in Table IV represent the evaluation results of our algorithm with $\alpha = 75\%$ compared to the baseline approach where $\alpha = 100\%$. It shows an expected shift in the behavior of the algorithm as a consequence of the changes in the new parameter α. Figure 5 depicts this effect in terms of precision and recall. The difference between the precision and recall in the lower graph is an indication of a higher number of clusters in our approach compared with the baseline model (the upper graph).

B. Sampling

Sampling was added as a final merging step to improve the quality of the clustering. Our new diffusion strategies together with sampling, have significantly improved the original model (upper graph in Figure 5) with around 10% over B-Cubed and 11% V-measure (Figure 6).

C. Single vs Multiple

To study the effect of MD we computed the B-Cubed score for SD of each entity in isolation to be compared with the corresponding result for MD with sampling. Figure 7 shows this effect for the three entities in our synthetic data set. For each entity, the upper diagram shows the result of isolated



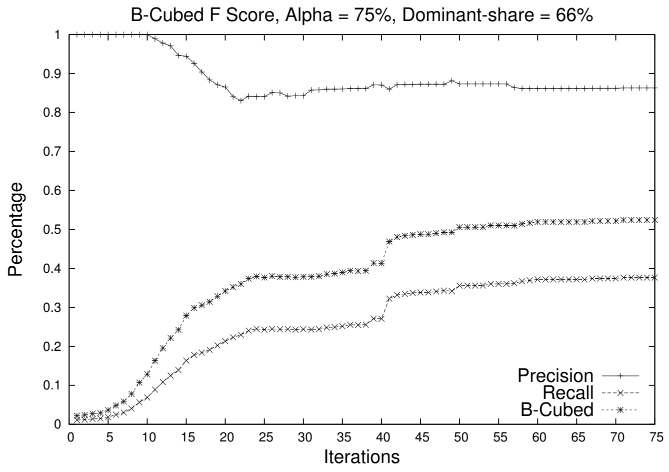


Fig. 5. The effect of changing the Attachment α value on the quality of the result. The differences show the shift in the behavior of the algorithm from highly progressive to rather conservative that results in a higher number of clusters.

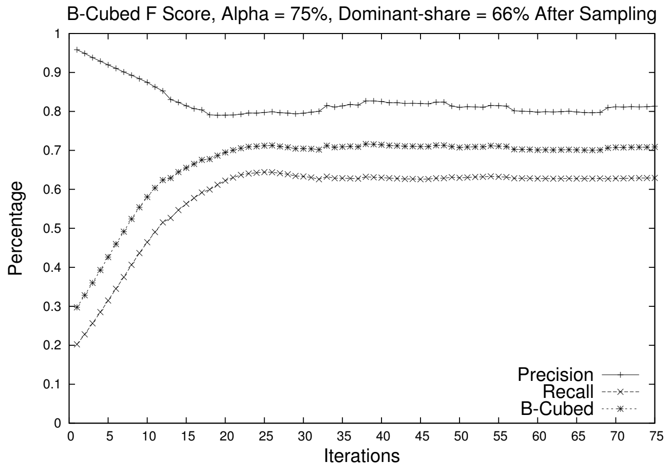


Fig. 6. The improvement of the enhanced results with $\alpha = 75\%$ using 4% supervised sampling of 3% higher populated clusters.

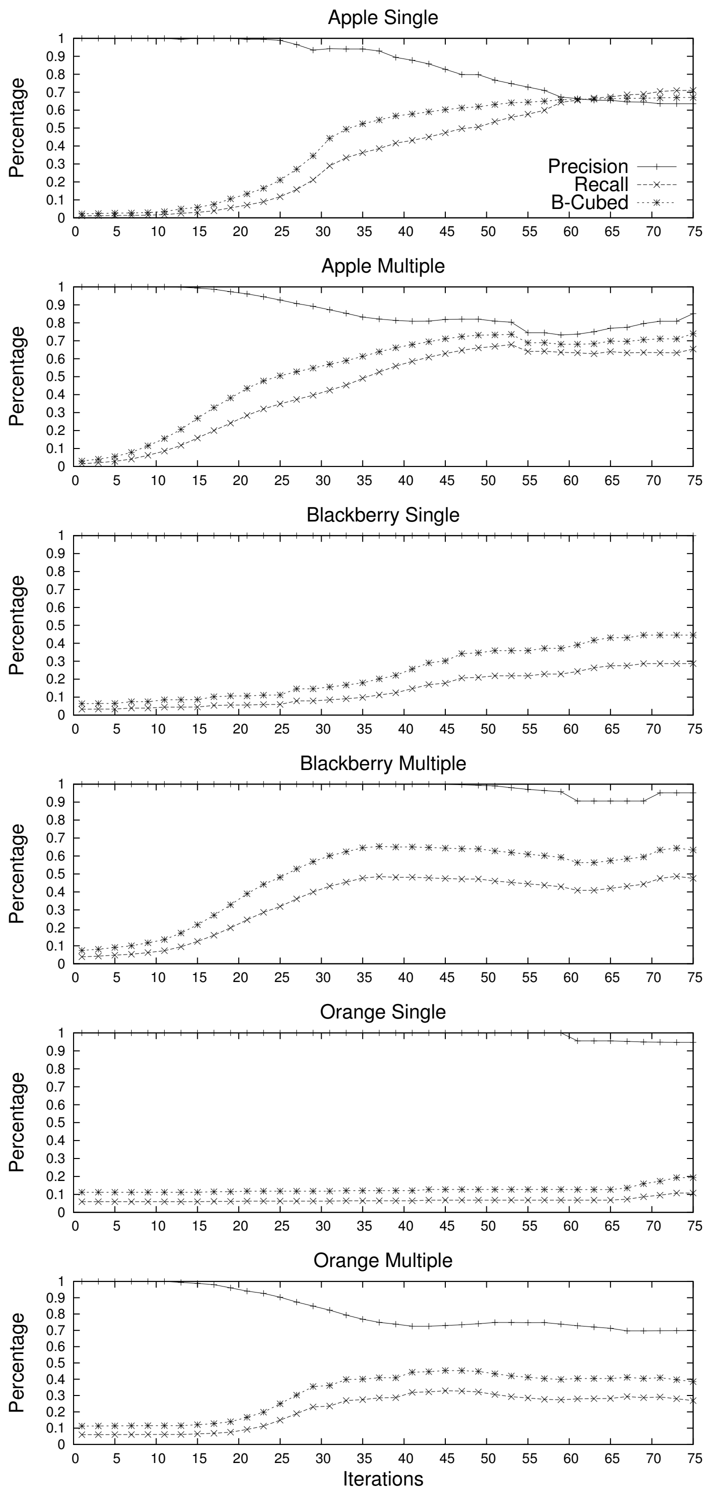


Fig. 7. SD in isolation (left column) compared with MD with sampling (right column). Results are computed with $\alpha = 100\%$ for SD and $\alpha = 75\%$ using 4% supervised sampling of 3% higher populated clusters for MD.

SD and the lower diagram shows the distinctive B-Cubed score after applying MD and sampling. The overall result has improved in all cases as expected. However, the magnitude of the changes differs for different entities that is a reasonable consequence, considering the number of documents for each entity.

VII. Conclusion

In this paper, we developed a new model of multiple disambiguation (MD) as an extended version of single disambiguation (SD) [3]. We believe disambiguation in its modern applications is more about the relationship between different contextual units (sentences or paragraphs) rather than the entire documents. Therefore, the assumption of having a single ambiguous mention per document does not hold any more.

Our solution uses a new graph based document representation model to account for MD. In addition, a new parameter is added to the original algorithm to control the resolution of the clustering. Finally, the quality of the clustering is improved by extracting a higher number of smaller and smoother clusters and merging them using supervised sampling. Experiments show that MD, not only is solvable over small datasets containing entities with overlapping semantic boundaries, but also improves the quality of the results compared to multiple isolated SDs.

Since scalability was not the main focus in this research, we didn't consider it as an issue. Nevertheless, we believe that our algorithm can be significantly improved in this respect. In addition, the sampling approach incorporated in our model can still be improved to provide higher or similar B-cubed values using lower number of samples.

References

[1] W. Haas, "J.r. firth: Papers in linguistics, 19341951. xii, 233 pp., 11 plates. london, etc.: Oxford university press, 1957. 35s." *Bulletin of the School of Oriental and African Studies*, vol. 21, pp. 668–671, 10 1958.

[2] D. Yarowsky, "Unsupervised word sense disambiguation rivaling supervised methods," in *33rd Annual Meeting of the Association for Computational Linguistics, 26-30 June 1995, MIT, Cambridge, Massachusetts, USA, Proceedings.*, 1995, pp. 189–196.

[3] F. Rahimian, S. Girdzijauskas, and S. Haridi, "Parallel community detection for cross-document coreference," in *Proceedings of the 2014 IEEE/WIC/ACM International Conference on Web Intelligence*, ser. WI'14, August 2014.

[4] A. Bagga and B. Baldwin, "Entity-based cross-document coreferencing using the vector space model," in *Proceedings of the 17th International Conference on Computational Linguistics - Volume 1*, ser. COLING '98. Stroudsburg, PA, USA: Association for Computational Linguistics, 1998, pp. 79–85.

[5] J. Mayfield, D. Alexander, B. J. Dorr, J. Eisner, T. Elsayed, T. Finin, C. Fink, M. Freedman, N. Garera, P. McNamee, S. Mohammad, D. W. Oard, C. D. Piatko, A. B. Sayeed, Z. Syed, R. M. Weischedel, T. Xu, and D. Yarowsky, "Cross-document coreference resolution: A key technology for learning by reading," in *AAAI Spring Symposium: Learning by Reading and Learning to Read'09*, 2009, pp. 65–70.

[6] C. H. Gooi and J. Allan, "Cross-document coreference on a large scale corpus," in *HLT-NAACL'04*, 2004, pp. 9–16.

[7] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008.

[8] S. Singh, A. Subramanya, F. Pereira, and A. McCallum, "Large-scale cross-document coreference using distributed inference and hierarchical models," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, ser. HLT '11. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011, pp. 793–803.

[9] P. Pantel, E. Crestan, A. Borkovsky, A.-M. Popescu, and V. Vyas, "Web-scale distributional similarity and entity set expansion," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, ser. EMNLP '09. Stroudsburg, PA, USA: Association for Computational Linguistics, 2009, pp. 938–947.

[10] L. Kolb, A. Thor, and E. Rahm, "Dedoop: Efficient deduplication with hadoop," *Proc. VLDB Endow.*, vol. 5, no. 12, pp. 1878–1881, Aug. 2012.

[11] L. Sarmento, A. Kehlenbeck, E. Oliveira, and L. Ungar, "An approach to web-scale named-entity disambiguation," in *Proceedings of the 6th International Conference on Machine Learning and Data Mining in Pattern Recognition*, ser. MLDM '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 689–703.

[12] B. Wellner, A. McCallum, F. Peng, and M. Hay, "An integrated, conditional model of information extraction and coreference with application to citation matching," in *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, ser. UAI '04. Arlington, Virginia, United States: AUAI Press, 2004, pp. 593–601.

[13] M. Wick, S. Singh, and A. McCallum, "A discriminative hierarchical model for fast coreference at large scale," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ser. ACL '12. Stroudsburg, PA, USA: Association for Computational Linguistics, 2012, pp. 379–388.

[14] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3-5, pp. 75 – 174, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0370157309002841

[15] M. A. Khalid, V. Jijkoun, and M. De Rijke, "The impact of named entity normalization on information retrieval for question answering," in *Proceedings of the IR Research, 30th European Conference on Advances in Information Retrieval*, ser. ECIR'08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 705–710. [Online]. Available: http://dl.acm.org/citation.cfm?id=1793274.1793371

[16] E. Stamatatos, "A survey of modern authorship attribution methods," *J. Am. Soc. Inf. Sci. Technol.*, vol. 60, no. 3, pp. 538–556, Mar. 2009. [Online]. Available: http://dx.doi.org/10.1002/asi.v60:3

[17] S. S. Sonawane and P. A. Kulkarni, "Article: Graph based representation and analysis of text document: A survey of techniques," *International Journal of Computer Applications*, vol. 96, no. 19, pp. 1–8, June 2014, full text available.

[18] K. Toutanova and C. D. Manning, "Enriching the knowledge sources used in a maximum entropy part-of-speech tagger," in *Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora: Held in Conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13*, ser. EMNLP '00. Stroudsburg, PA, USA: Association for Computational Linguistics, 2000, pp. 63–70. [Online]. Available: http://dx.doi.org/10.3115/1117794.1117802

[19] A. Rosenberg and J. Hirschberg, "V-measure: A conditional entropy-based external cluster evaluation measure." in *EMNLP-CoNLL*. ACL, Jun 2010, pp. 410–420.

[20] A. Kyrola, G. Blelloch, and C. Guestrin, "Graphchi: Large-scale graph computation on just a pc," in *Presented as part of the 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI 12)*. Hollywood, CA: USENIX, 2012, pp. 31–46. [Online]. Available: https://www.usenix.org/conference/osdi12/technical-sessions/presentation/kyrola