Result Disambiguation in Web People Search

Richard Berendsen, Bogomil Kovachev, Evangelia-Paraskevi Nastou,
Maarten de Rijke, and Wouter Weerkamp

ISLA, University of Amsterdam, Science Park 904, 1098 XH Amsterdam, The Netherlands
{r.w.berendsen,b.kovachev,e.nastou,derijke,w.weerkamp}@uva.nl

Abstract. We study the problem of disambiguating the results of a web people search engine: given a query consisting of a person name plus the result pages for this query, find correct referents for all mentions by clustering the pages according to the different people sharing the name. While the problem has been studied extensively, we discover that the increasing availability of results retrieved from social media platforms causes state-of-the-art methods to break down. We analyze the problem and propose a dual strategy where we distinguish between results obtained from social media platforms and those obtained from other sources. In our dual strategy, the two types of documents are disambiguated separately, using different strategies, and their results are then merged. We study several instantiations for the different stages in our proposed strategy and manage to achieve state-of-the-art performance.

1 Introduction

Web people search is a vertical search task: given a query consisting of a person name, find web documents that are relevant to this name. In recent years, various services offering web people search facilities have emerged, e.g., `intelius.com`, `123people.com`, `spokeo.com`, `peoplesmart.com`. *Result disambiguation* in the context of web people search is the problem of finding correct referents for all occurrences of the query person name in the search results. This will allow for a result representation where documents are grouped by person, allowing the user to efficiently zoom in on the documents referring to the individual of interest. Result disambiguation has been studied in the Web People Search (WePS) campaigns [3–5], using search results obtained from a major web search engine. One of the lessons learned is that standard hierarchical agglomerative clustering (HAC) approaches using textual features achieve high performance on the task.

We revisit the result disambiguation problem for web people search and we do so in the setting of a people search engine, a vertical meta-search engine that aggregates people search results from a broad range of sources, both generic web search engines and social media platforms. The inclusion of increasing numbers of results originating from social media platforms in the aggregated result list poses new challenges for result disambiguation methods that have previously been shown to be very effective. Specifically, social media profiles are textually sparse and contain relatively large amounts of boilerplate material, making it non-trivial to extract good textual features from them.

We propose a dual strategy: different ways of treating social results and other, "non-social" results. We examine the effectiveness of various disambiguation techniques on

R. Baeza-Yates et al. (Eds.): ECIR 2012, LNCS 7224, pp. 146–157, 2012.

social results and we contrast our findings with known results for non-social results. The two types of document require different strategies for effective result disambiguation. We then propose and examine techniques for combining the outcomes of result disambiguation on social results with those on non-social results.

Our main contributions are: (i) signaling the problem that people search results contain increasing numbers of social media profiles plus its negative impact on existing disambiguation strategies for "traditional" web search results; (ii) a new strategy of treating social and non-social results separately and combining the resulting clusterings; (iii) a detailed error analysis. We also make available the data set (queries, ground truth, search results) used in this paper.

In Section 2 we discuss related work, in Section 3 we detail our methods, and in Section 4 our experimental setup. We show the results in Section 5 and perform an error analysis in Section 6. Finally we offer some conclusions and point out directions for future work in Section 7.

2 Related Work

Disambiguating search results can be considered as clustering the result documents. This clustering is usually aimed at organizing and categorizing results to facilitate, for example, search refinement or exploratory search. Various approaches to (web) search result clustering have been proposed. Examples include key phrase extraction [22], latent semantic indexing on result documents [14], and clustering based on document snippets [10, 11, 23]. A lot of work in the area focuses on selecting clusters to present to users (e.g., [9]) or assigning labels to the clusters (e.g., [13]); see [7] for an overview on web search results clustering.

As a special case of search result clustering, result disambiguation for person name queries deals with clustering search results per person. The WePS campaigns [3–6] provide an evaluation framework for this particular task. Most of the best performing approaches use hierarchical agglomerative clustering (HAC). Differences between participants mainly exist in the features that are being used. Chen et al. [8] use tokens from sentences in which the query name occurs, tokens from the entire web result page, and bigrams as features in their clustering approach; each feature is weighed using tf.idf. Monz and Weerkamp [15] compare a large range of features for HAC, including windows around the query name, different tf.idf weighting schemes, and various schemes for computing cluster similarity. Ikeda et al. [12] use a two-stage clustering algorithm, in which HAC (based on named entities, compound keywords, and links) is followed by keyword extraction from the resulting clusters to add documents to these clusters. Yoshida et al. [21] build on the system created by Ikeda et al. [12], but replace the second stage with a bootstrapping algorithm, Espresso [17]; this bootstrapping algorithm uses single words as patterns and documents as instances. When the number of clusters is known beforehand, the person name disambiguation task can be formulated as a classification problem. For recent work, see [16, 18].

The work in this paper differs from previous work in that it focuses on a people search engine setting, in which social media profiles play an important role. We show that the task of disambiguation is harder when these profiles are common.

3 Dual Strategies for Result Disambiguation

In this section we describe our methods for disambiguating search engine results in the setting of web people search. We start with a high level overview and then zoom in on individual steps that make up our method.

As explained in the introduction, we work in the setting of a meta-search engine, one that aggregates search results from generic web search engines and from a range of social media platforms. Details about the social media platforms considered are given in Section 4 below. Algorithm 1 specifies our dual strategy, with separate disambiguation steps for social and non-social documents, followed by a merge step in which we combine the results of the two clustering steps.

Splitting D into Social and Non-social Documents. Let D be the set of result documents retrieved for a given person name query. If the URL of a result document d contains the top level domain of one of the social media platforms we consider, we add d to the set of *social* results D_s, and otherwise we add it to the *non-social* results D_{ns}. We dub this method "By URL."

Algorithm 1. Dual strategy for result disambiguation

1: **Input**: query q, search engine results D
2: **Uses**: clustering methods M_{ns} and M_s
3: Split document set D into two sets, social documents D_s and non-social documents D_{ns}
4: **for** non-social documents D_{ns} **do**
5: disambiguate D_{ns} by creating clusters using method M_{ns}
6: **return** clustering $\mathcal{C}_{ns}$
7: **for** social documents D_s **do**
8: disambiguate D_s by creating clusters using method M_s
9: **return** clustering $\mathcal{C}_s$
10: Merge cluster results $\mathcal{C}_{ns}$ and $\mathcal{C}_s$ into $\mathcal{C}_{final}$
11: **return** merged clustering $\mathcal{C}_{final}$

Clustering Methods Considered for M_{ns} and M_s. We now discuss the clustering methods that we consider. Due to the fact that the algorithm has to run online, at query time, we have a strong preference for relatively light-weight methods.

- One-in-one: This simple baseline method creates a singleton cluster for each document.
- All-in-one: This simple baseline method creates one cluster which contains all documents.
- HAC: Hierarchical agglomerative clustering (HAC) approaches have been successful in WePS-2 [15]. Our implementation uses both single link and centroid clustering, a minimal similarity threshold as a stopping criterion, and cosine similarity between document vectors with tf.idf term weights. Inverse document frequencies (idf) are calculated with respect to search results over all queries and term frequencies are normalized. We use Porter stemming. Section 4 has details on the parameter settings.

Additional Clustering Methods Considered for M_s. We apply four methods that are specifically targeted at results obtained from social media platforms: (i) Cross links, (ii) Coclicks, (iii) Clicked in the same burst, and (iv) Picasa. In the first three methods, we perform single link HAC with a binary similarity matrix:

– Cross links: If we find a hyperlink between two result documents, we set the similarity of these two documents to one, and otherwise the similarity is zero.
– Coclicks: For each query and people search engine user, we check if this user clicked two results from different social media platforms for the same query. If this is the case for at least two users, we set the similarity between these results to one, otherwise to zero.
– Clicked in same burst: In the query logs of the people search engine, we count the unique daily visitors who issue a given query, resulting in a time series. We then define a burst in this time series to be a number of consecutive days in which the unique daily search count exceeds the mean daily search count plus two standard deviations. In addition, a burst-day needs to have at least ten unique daily searches. We record for each search the last clicked result document. If two search results are clicked on last during the same burst, we set their similarity to one, otherwise to zero.
– Picasa: Finally, we extract all user profile pictures from the social media profiles and load them in Google Picasa, which has a built-in face recognition component. We let Picasa find groups of faces using its default parameters and without any user feedback. We then cluster the corresponding profiles.

Merging Methods. We consider two methods for merging clusterings $\mathcal{C}_{ns}$ and $\mathcal{C}_s$ of the non-social and social result documents contained in D:

– Baseline merge: We merge clusters by taking the union of the respective clusterings: $\mathcal{C}_{final} := \mathcal{C}_{ns} \cup \mathcal{C}_s$.
– Advanced merge: We apply Algorithm 2 to merge results. The algorithm uses a similarity threshold τ and penalizes clusters that contain a social media result using parameter w.

Dual Result Disambiguation Methods Considered. The options listed above give rise to a large number of combinations for dual strategy runs. We limit ourselves to the methods for result disambiguation specified in Table 1.

Algorithm 2. Merging social media cluster results $\mathcal{C}_s$ with non-social cluster results $\mathcal{C}_{ns}$

Input: social cluster results $\mathcal{C}_s$, non-social cluster results $\mathcal{C}_{ns}$
Parameter: τ, similarity threshold between clusters.
Parameter: w, social penalty: decreases similarity with clusters in $\mathcal{C}_{ns}$ once they contain social results.
Parameter: $HACsim$, cluster similarity function.
while $\mathcal{C}_s \neq \emptyset$ **do**
  $\mathcal{C}_{temp} := \emptyset$
  for cluster i in C_s **do**
    find cluster $j \in \mathcal{C}_{ns}$ with highest $sim(i,j) = \frac{HACsim(i,j)}{1+nw}$,
    # where n is the number of social results already in j
    if $sim(i,j) < \tau$ **then**
      $\mathcal{C}_{temp} := \mathcal{C}_{temp} \cup \{i\}$
      $\mathcal{C}_s := \mathcal{C}_s \setminus \{i\}$
  for cluster k in C_{ns} **do**
    find cluster $l \in \mathcal{C}_s$ with highest $sim(k,l) = \frac{HACsim(k,l)}{1+nw}$,
    $k := k \cup l$
    $\mathcal{C}_s := \mathcal{C}_s \setminus \{l\}$
  $\mathcal{C}_{ns} := \mathcal{C}_{ns} \cup \mathcal{C}_{temp}$
return merged clustering $\mathcal{C}_{ns}$

4 Experimental Setup

To test previously established methods and our own methods, we use the experimental setup detailed in this section. We list the research questions we seek to answer, introduce

Table 1. Result disambiguation methods

Name	Splitting M_{ns}	M_s	Merge method
Dual baseline	By URL	HAC single link	One in one	$\mathcal{C}_{final} = \mathcal{C}_{ns} \cup \mathcal{C}_s$
Dual merge	By URL	HAC single link	One in one	Algorithm 2

the query and document sets used, report on our ground truth creation, explain our parameters, and finally, introduce the metrics on which we report.

Research Questions. In the next section we answer the following research questions.

RQ 1 How does the lightweight and state-of-the-art HAC single link method described in Section 3 perform on our search results with a large number of social media documents?

RQ 2 Can we find evidence in the textually sparse social media platform results for clustering them? We consider cross links between profiles, evidence from query log files, and a visual clue: the user profile picture.

RQ 3 Can we successfully cluster the complete search result set, with social and non-social documents, by treating social documents differently from other web documents, clustering them separately and merging the clusterings back together?

Query and Document Set. To help answer our research questions we select queries from query logs of a people search engine. The logs have been collected between September 2010 and February 2011 and contain queries, associated clicks, and browser cookies (for user identification); Weerkamp et al. [20] present a detailed study of the logs. To select ambiguous queries, we required queries to have clicks to at least three profiles within one social media platform. In addition, we required that at least seven searches were performed, with clicks to at least two search engines or social media platforms, to make sure we have some evidence in our click data for clustering. The document set is constructed by retrieving 20 documents (profiles) from each of five large social media platforms (e.g., Facebook, LinkedIn, Twitter) and 50 documents from three major web search engines (Google, Yahoo! and Bing). The resulting document set is de-duplicated based on URL. Documents that do not contain the person name are ignored.[1]

Ground Truth. Annotations were created in the same way as for the WePS campaigns, allowing for a comparison between the two datasets. A document can be assigned to multiple clusters, if the document contains references to two or more persons with the same name. If the annotator could not say for sure whether a document belonged to existing clusters, a new cluster was created. Annotations were done on the full dataset and not on separated datasets as a result of splitting. In case no evidence whatsoever could be found in a document (e.g., private profiles without pictures), the document was discarded. We also discarded documents in the web results that were returned by other people search engines. We distributed queries to multiple annotators, with insufficient means to study inter annotator agreement.

[1] The dataset and annotations are available at
`http://ilps.science.uva.nl/resources/ecir2012rdwps`

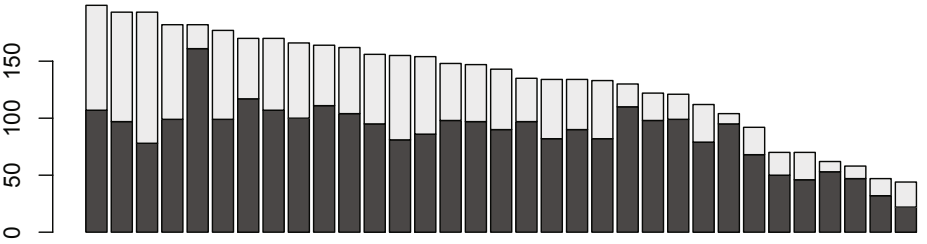

Fig. 1. Number of documents per topic in our dataset, split as in our dual strategy method in D_s and D_{ns}. Dark boxes contain non-social results D_{ns}; light boxes contain social results D_s.

Figure 1 shows the number of annotated results per query in our dataset. We have split out the results in non-social (D_{ns}) and social results (D_s) as in our dual strategy method. The topics are ordered by the number of annotated documents.

Parameter Settings. When it comes to applying HAC, we follow Monz and Weerkamp [15]. They did not report on a minimal threshold stopping criterion value. We perform a partial parameter sweep on the WePS-2 dataset, resulting in the value 0.225 which we use for both single link and centroid HAC on all datasets. For our dual strategy with merge (viz. Algorithm 2) we use the following parameter values: single link clustering as $HACsim$, $\tau = 0.5$, and w = 1. We did not use a separate training set, but explored a few combinations on our test set. We report on parameter sensitivity in Section 6.

Metrics. We use the B-cubed metrics [1] for evaluation of cluster quality, as was done in WePS-2. An extended version of the metrics is used to accommodate for overlapping clusters. For each topic, B-cubed precision (B^3P) and B-cubed recall (B^3R) are computed, and a macro-averaged F-measure with $\beta = 0.5$ is computed.

Significance Tests. We use two-tailed paired t-tests. We look for significant differences at an optimistic level $\alpha = 0.05$, denoted $\star$ and a more conservative level, $\alpha = 0.001$, denoted $\star\star$. We also use a paired randomization test [19], and report significant differences at $\alpha = 0.05$ and $\alpha = 0.001$ with $\bullet$ and $\bullet\bullet$, respectively.

Unanimous Improvement Ratio. The F-measure weighs precision and recall with the β parameter. We set β to 0.5 as in the WePS campaigns, to favour neither precision nor recall. Choosing a different β may affect system ranking. To estimate which pairwise performance differences in $F_{\beta=0.5}$ are robust against different β values, we employ the Unanimous Improvement Ratio (UIR) [2]. For two systems A and B, Let T_A be the set of queries for which system A achieves precision and recall scores that are greater than or equal to the scores of system B. For these queries, the F score for system A will not be smaller for any value of β. Let T be the set of all queries. Then $\mathrm{UIR}(A, B) = (|T_A| - |T_B|)/|T|$. For the people search clustering task, Amigó et al. [2] give a rule of thumb that we employ in our result section: if $\mathrm{UIR}(A, B) \geq 0.25$ then an observed improvement of system A over B in average $F_{\beta=0.5}$ is robust.

5 Results and Analysis

We report on three sets of experiments. First, we examine the performance of methods from the literature on our people search engine data set, for which we consider the

performance on the full data set, and its restrictions to search engine results and social media results (RQ 1). Second, we examine the performance of methods designed for social media profiles (RQ 2). Finally, we present results of our dual strategies on the full data set (RQ 3). The results are followed by an analysis.

Table 2. B-cubed precision, B-cubed recall, and macro averaged F-measure of standard methods. For $F_{\beta=0.5}$, significant differences are with regard to HAC single link.

	all sources			Search results from search engines			social media platforms		
	B^3P	B^3R	$F_{\beta=0.5}$	B^3P	B^3R	$F_{\beta=0.5}$	B^3P	B^3R	$F_{\beta=0.5}$
All in one	0.17	1.00	0.25 ⋆⋆ ●●	0.22	1.00	0.31 ⋆⋆ ●●	0.13	1.00	0.20
One in one	1.00	0.48	0.62	1.00	0.43	0.58 ⋆⋆ ●●	1.00	0.86	0.92 ⋆⋆ ●●
HAC single link	0.56	0.87	0.67	0.76	0.86	0.79	0.14	1.00	0.21
HAC centroid	0.72	0.71	0.69	0.89	0.70	0.75	0.17	0.96	0.27 ⋆⋆ ●●

Results. We present the results of applying default clustering methods for web people search results on our people search engine dataset in Table 2. The performance of single link HAC on our full dataset is substantially lower than its performance on the WePS-2 data (F measure: 0.81 [15]). On our full dataset the improvement of single link HAC over the one in one baseline is not significant or robust (UIR < 0.25). When we restrict ourselves to documents returned by web search engines, we find that the performance of HAC improves dramatically, approaching the levels reported for WePS-2. Here, single link HAC improves significantly over the one in on baseline. The improvement is not robust. On social media documents, the performance of HAC is about as bad as the all in one baseline, which simply adds all documents to a single cluster.

Experiments using single link and centroid HAC reveal that the difference between the two is limited. On our full dataset and on the social media profiles centroid HAC works best. If we limit ourselves to the web search results, we find that single link HAC performs better. The difference is only significant on the social media profiles, and it is nowhere robust. Looking at the social media results, we find that the one in one baseline is the best system. The observed difference in best performing approaches between the two document types clearly shows that the two behave very differently, which motivates our dual strategies. From the experimental results reported on social media documents, it is almost safe to assume that each document corresponds to a unique individual.

Next, we turn to result disambiguation methods that focus on social results. Table 3 lists the results of these experiments. We observe that none of the "social" methods is able to beat the one in one baseline. The only method that differs significantly from the one in one baseline is "Cross links," but it is worse (⋆⋆ ●●). It is also the only method over which the improvement of the one in one baseline is robust.

Table 3. B-cubed precision and recall, macro averaged F-measure of social clustering methods

Search results from social media platforms			
	B^3P	B^3R	$F_{\beta=0.5}$
One in one	1.00	0.86	0.92
Cross links	0.83	0.88	0.84
Coclicks	0.99	0.87	0.91
Clicked in Same Burst	0.98	0.86	0.91
Picasa	1.00	0.86	0.92

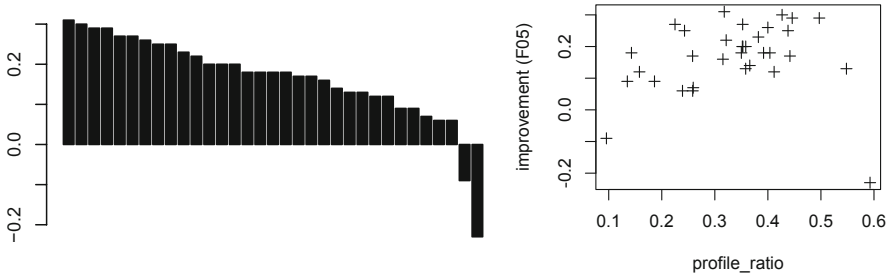

Fig. 2. (Left) Difference in $F_{\beta=0.5}$ score per query between dual merge and HAC single link; a positive difference indicates a query where dual merge outperforms HAC single (and vice versa). (Right) Improvement in $F_{\beta=0.5}$ versus the ratio of social documents in the total result set for a given query.

Our final set of experimental results concerns the dual strategies, for which results are listed in Table 4. We observe that even with a naive merging strategy (dual baseline), we manage to achieve scores comparable to those achieved with HAC on WePS-2. Apparently, we are able to suppress the negative impact resulting from social media results. The dual baseline has large and significant ($\star\star$ $\bullet\bullet$) improvements over all other methods on our full dataset. It improves robustly only over centroid HAC, however. With regard to single link HAC, the dual baseline improves precision on all topics, but it also looses a bit of recall on all topics, indicating that there is room for improvement. The more sophisticated merge method (Algorithm 2) improves slightly but significantly and robustly on the dual baseline ($\star$ $\bullet\bullet$). It has higher recall on about six out of ten queries and never a lower precision: the intended effect of this method. Dual merge has robust improvements also over centroid HAC and the one in one baseline.

Table 4. B-cubed precision and recall, macro averaged F-measure of the dual strategies

	Search results from all sources		
	B^3P	B^3R	$F_{\beta=0.5}$
Dual baseline	0.90	0.78	0.82
Dual merge	0.90	0.80	0.83

Analysis. In our analysis, we compare our dual strategy with the single link HAC baseline and we investigate why our "social" methods fail to improve over the one in one baseline.

Dual merge vs. single link HAC. As shown in the previous paragraph, single link HAC is the best performing method on the results from search engines, which is why we use it as M_{ns} in our dual strategy approaches. Here, we compare this method to our dual merge strategy. Figure 2 (Left) compares the difference between the dual merge strategy and our baseline (single link HAC) on a per-query basis. For almost all queries, there is a clear improvement when using the dual strategy.

Our strategy of treating social media documents in a separate manner leads to large improvements and we expect to see a stronger improvement in cases where more of social media documents are present. Figure 2 (Right) shows, however, that there is no clear correlation between the ratio of social media results returned for a query and the improvement after distinguishing between social and non-social search results.

Returning to Figure 2 (Left), the query that shows the largest drop in performance, going from the HAC baseline to our dual strategy method, is the query with the highest ratio of social documents. For this query, all search engines return noise: after automatically filtering out results that did not contain the person name, we are left with only 27 results for this query. During annotation, another 17 of these results were discarded, leaving only ten documents for clustering. Some profiles among these documents do refer to the same person, leading to the degraded performance for this query.

The second query for which our dual merge method performs worse than the baseline concerns a not very common name, but it is the name of a celebrity (*Joey Spaan*). Consequently, this person dominates the search results completely. He has profiles on various social media platforms, and since our dual merge strategy is designed to cluster only few social media profiles, the small loss in recall for this query is unsurprising.

Analysis of "social" methods. The performance of baseline methods on social media results shows that such results should be treated differently from other web documents. The good performance of the one in one baseline is caused by people generally (i) having only one profile per platform, and (ii) using different platforms for very different reasons. While web documents returned by general web search engines can be completely dominated by a single person, many people will typically be represented in the social media results. Besides, it is likely that people make an effort to keep their Facebook profile (for friends) separated from their LinkedIn profile (for work related contacts), leading to limited overlap in content.

The one in one baseline on social media profiles is not perfect. We investigate why our ("social") methods fail to discover the few clusters that are there. First, in eight out of our 33 queries, the one in one baseline is actually perfect. We examined fifteen random queries of the remaining 25 and found that the main reasons for our annotators to cluster social documents together are: (i) a user profile picture (40 pairs of profiles), (ii) a company name or affiliation (12 pairs) and (iii) an occupation (11 pairs).

The method that leads to the highest recall is the cross links method, although it looses precision, too. A simple cause for this is that, for example, LinkedIn profiles contain links to other profiles with the same person name ("Find a different John Smith"). Adding a rule that ignores within platform cross links leads to too few links to make a noticeable difference. The Coclicks method and the Clicked in same burst method have almost no effect on performance. They share one problem: clicks are very sparse in the query logs of our people search engine, making them hard to use. Finally, the Picasa method fails to recognize and match enough faces in the user profile pictures. All in all, we fail to improve over the one in one baseline with our specific "social" methods.

It proves challenging to identify textual evidence in the social documents. In follow-up experiments, we considered dedicated content extractors for each of the social media platforms, so that only relevant text is extracted and not the boilerplate material. Using these extractors results in an increase in precision for the single link HAC baseline, but it also leads to a drop in recall, resulting in little change to the F measure.

Table 5. Impact of parameters τ (Left) and w (Right) on the performance of the dual merge method

τ	w	B^3P	B^3R	$F_{\beta=0.5}$
$\tau = 0.225$	$w = 1.0$	0.77	0.82	0.78
$\tau = 0.500$	$w = 1.0$	0.90	0.80	0.83
$\tau = 0.775$	$w = 1.0$	0.90	0.78	0.82

w	τ	B^3P	B^3R	$F_{\beta=0.5}$
$w = 0.0$	$\tau = 0.5$	0.74	0.82	0.76
$w = 0.5$	$\tau = 0.5$	0.86	0.81	0.82
$w = 1.0$	$\tau = 0.5$	0.90	0.80	0.83
$w = 1.5$	$\tau = 0.5$	0.90	0.78	0.82
$w = 2.0$	$\tau = 0.5$	0.90	0.78	0.82

6 Discussion

In this section we explore the impact of various parameters on our results. First, we look at the similarity threshold in HAC and second, we explore the parameters of our merging algorithm.

The Minimal Similarity Threshold in HAC. Artiles et al. [4] observe that performance of HAC is strongly dependent on the minimal similarity threshold used as a stopping criterion. Different topics have different optimal thresholds and the authors provide an upper and lower bound for HAC by doing a parameter sweep and taking for each query the optimal value. The variety in optimal threshold is such that learning an average optimal value on one dataset is no guarantee for success on another dataset.

We try a different, query-dependent approach to estimating the similarity threshold, based on the observation that if a name is very ambiguous, we would require more evidence to cluster two documents with this name together. For example, it would not be unlikely to have two different John Smiths's playing basketball in New York, but it would be unlikely to have two Jack Rumplestilskin's doing so. A parameter sweep on the WePS-1 data shows that if a name is very ambiguous, we require a high similarity threshold, just as we expect. After visual inspection of the results, we perform a test run on the WePS-2 data with the following simple rule: if there are more than 500 LinkedIn profiles for a given query, use a similarity threshold of 0.360, otherwise use the default threshold of 0.225. Results of this experiment show a slight increase in precision, but an equal drop in recall, leading to the same F measure as a run without query-dependent thresholding. Finding a more sophisticated way to use query characteristics to predict thresholding parameters is an interesting direction for future research.

Parameter Sensitivity of Our Merging Algorithm. Algorithm 2 has a number of parameters. For $HACsim$ we use single link clustering, as it performs best overall. Here, we explore the impact of the minimal similarity threshold (τ) and the parameter w, which regulates how strong the similarity between a social and a non-social cluster decreases for each social cluster already present in the non-social cluster. Table 5 lists the performance for different values of τ (Left), while keeping w stable, and different values of w (Right), while keeping τ stable. We find that increasing τ leads to better precision, but decreasing recall. For w, we find a similar patterns of improving precision with higher w-values, at the cost of recall.

7 Conclusion

In this paper we studied the problem of disambiguating the search results of a people search engine. Our results show that the increasing availability of results retrieved from social media platforms causes state-of-the-art methods to break down.

We proposed a dual strategy where we treat social search results differently from non-social results. For non-social results, we used single link HAC as a strategy, the best performer when we evaluate on the subset of results obtained via generic web search engines. For social results, we investigated several methods but were unable to beat the one in one baseline. Therefore, we selected the one in one baseline as a strategy here. We tested two methods for merging the two clusterings. The first is a baseline method that simply takes the union of both clusterings. This method already achieved a large boost in precision. With the second merging method we were able to gain recall without losing precision, obtaining results comparable to state of the art results obtained on WePS datasets.

For future work we want to explore possibilities to estimate query-dependent stopping thresholds for HAC and apply state-of-the-art image processing tools to cluster social media profiles based on pictures.

Acknowledgements. This research was supported by the European Union's ICT Policy Support Programme as part of the Competitiveness and Innovation Framework Programme, CIP ICT-PSP under grant agreement nr 250430, the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreements nr 258191 (PROMISE Network of Excellence) and 288024 (LiMoSINe project), the Netherlands Organisation for Scientific Research (NWO) under project nrs 612.061.814, 612.061.-815, 640.004.802, 380-70-011, 727.011.005, the Center for Creation, Content and Technology (CCCT), the Hyperlocal Service Platform project funded by the Service Innovation & ICT program, the WAHSP project funded by the CLARIN-nl program, under COMMIT project Infiniti and by the ESF Research Network Program ELIAS. We thank Enrique Amigó and Julio Gonzalo for their assistance with UIR.

References

[1] Amigó, E., Gonzalo, J., Artiles, J., Verdejo, F.: A comparison of extrinsic clustering evaluation metrics based on formal constraints. Information Retrieval 12, 461–486 (2009)

[2] Amigó, E., Gonzalo, J., Artiles, J., Verdejo, M.F.: Combining evaluation metrics via the unanimous improvement ratio and its application to clustering tasks. Journal of Articial Intelligence Research 42, 689–718 (2011)

[3] Artiles, J., Gonzalo, J., Sekine, S.: The semeval-2007 weps evaluation: Establishing a benchmark for the web people search task. In: Proceedings of the 4th International Workshop on Semantic Evaluations, pp. 64–69 (2007)

[4] Artiles, J., Gonzalo, J., Sekine, S.: Weps 2 evaluation campaign: overview of the web people search clustering task. In: 2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference (2009)

[5] Artiles, J., Borthwick, A., Gonzalo, J., Sekine, S., Amigó, E.: WePS-3 Evaluation Campaign: Overview of the Web People Search Clustering and Attribute Extraction Tasks. In: CLEF 2010 Working Notes (2010)

[6] Balog, K., Azzopardi, L., de Rijke, M.: Resolving person names in web people search. In: Weaving Services, Location, and People on the WWW (2009)
[7] Carpineto, C., Osiński, S., Romano, G., Weiss, D.: A survey of web clustering engines. ACM Computing Surveys 41(3), 17:1–17:38 (2009)
[8] Chen, Y., Lee, S., Huang, C.: Polyuhk: A robust information extraction system for web personal names. In: 2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference (2009)
[9] Crabtree, D., Gao, X., Andreae, P.: Improving web clustering by cluster selection. In: WI 2005, pp. 172–178 (2005)
[10] Ferragina, P., Gulli, A.: A personalized search engine based on web-snippet hierarchical clustering. Software: Practice and Experience 38(2), 189–225 (2008)
[11] Geraci, F., Pellegrini, M., Pisati, P., Sebastiani, F.: A scalable algorithm for high-quality clustering of web snippets. In: SAC 2006, pp. 1058–1062 (2006)
[12] Ikeda, M., Ono, S., Sato, I., Yoshida, M., Nakagawa, H.: Person Name Disambiguation on the Web by Two-Stage Clustering. In: 2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference (2009)
[13] Janruang, J., Kreesuradej, W.: A new web search result clustering based on true common phrase label discovery. In: Int. Conf. Comp. Intell. for Modelling, Control and Automation (2006)
[14] Mecca, G., Raunich, S., Pappalardo, A.: A new algorithm for clustering search results. Data & Knowledge Engineering 62(3), 504–522 (2007)
[15] Monz, C., Weerkamp, W.: A comparison of retrieval-based hierarchical clustering approaches to person name disambiguation. In: SIGIR 2009 (2009)
[16] On, B., Lee, I., Lee, D.: Scalable clustering methods for the name disambiguation problem. Knowledge and Information Systems 0219-1377, 1–23 (2011)
[17] Pantel, P., Pennacchiotti, M.: Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In: ACL 2006, pp. 113–120 (2006)
[18] Pilz, A., Paaß, G.: From names to entities using thematic context distance. In: CIKM 2011 (2011)
[19] Smucker, M., Allan, J., Carterette, B.: A comparison of statistical significance tests for information retrieval evaluation. In: CIKM 2007 (2007)
[20] Weerkamp, W., Balog, K., de Rijke, M., Berendsen, R., Kovachev, B., Meij, E.: People searching for people: Analysis of a people search engine log. In: SIGIR 2011 (2011)
[21] Yoshida, M., Ikeda, M., Ono, S., Sato, I., Nakagawa, H.: Person name disambiguation by bootstrapping. In: SIGIR 2010, pp. 10–17. ACM (2010)
[22] Zhang, D., Dong, Y.: Semantic, Hierarchical, Online Clustering of Web Search Results. In: Yu, J.X., Lin, X., Lu, H., Zhang, Y. (eds.) APWeb 2004. LNCS, vol. 3007, pp. 69–78. Springer, Heidelberg (2004)
[23] Zhu, D., Dreher, H.: Improving Web Search by Categorization, Clustering, and Personalization. In: Tang, C., Ling, C.X., Zhou, X., Cercone, N.J., Li, X. (eds.) ADMA 2008. LNCS (LNAI), vol. 5139, pp. 659–666. Springer, Heidelberg (2008)