Information Systems 37 (2012) 336–351

Contents lists available at SciVerse ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/infosys

Quality-aware similarity assessment for entity matching in Web data[☆]

Surender Reddy Yerva, Zoltán Miklós*, Karl Aberer

EPFL IC LSIR, Lausanne, Switzerland

ARTICLE INFO

Available online 24 September 2011

Keywords:
Entity matching
Web
Similarity functions
Person name disambiguation
Twitter message classification

ABSTRACT

One of the key challenges to realize automated processing of the information on the Web, which is the central goal of the Semantic Web, is related to the entity matching problem. There are a number of tools that reliably recognize named entities, such as persons, companies, geographic locations, in Web documents. The names of these extracted entities are, however, non-unique; the same name on different Web pages might or might not refer to the same entity. The entity matching problem concerns of identifying the entities, which are referring to the same real-world entity. This problem is very similar to the entity resolution problem studied in relational databases, however, there are also several differences. Most importantly Web pages often only contain partial or incomplete information about the entities.

Similarity functions try to capture the degree of belief about the equivalence of two entities, thus they play a crucial role in entity matching. The accuracy of the similarity functions highly depends on the applied assessment techniques, but also on some specific features of the entities. We propose systematic design strategies for combined similarity functions in this context. Our method relies on the combination of multiple evidences, with the help of estimated quality of the individual similarity values and with particular attention to missing information that is common in Web context. We study the effectiveness of our method in two specific instances of the general entity matching problem, namely the person name disambiguation and the Twitter message classification problem. In both cases, using our techniques in a very simple algorithmic framework we obtained better results than the state-of-the-art methods.

1. Introduction

Entity matching is a well studied problem in the context of relational databases [1–7], for a survey see [8]. Even if the papers are dated back quite early, this topic has also regained in importance recently. It is more and more common and easy to combine independent data sources, especially on the Web. There is a number of tools which recognize named entities, such as persons, companies, geographic locations, in Web documents. The names of the entities are, however, non-unique, the same name on different Web pages might or might not refer to the same entity. The entity matching problem concerns with identifying the entities, which are referring to the same real-world entity. This problem is very similar to the entity resolution problem studied in relational databases, however, there are also several differences. Most importantly Web pages often only contain partial or incomplete information about the entities. Web pages are also much less structured as database records. Many of the models, which were developed for databases are not directly applicable in the new setting, for example the model of fuzzy duplicates [3] does not fit well the new context. The information that could help here is the content of the Web pages, where the

[☆] This work was partially supported by the European Commission, through the NisB project (FP7-ICT-256955).

* Corresponding author.

E-mail addresses: surenderreddy.yerva@epfl.ch (S.R. Yerva), zoltan.miklos@epfl.ch (Z. Miklós), karl.aberer@epfl.ch (K. Aberer).

0306-4379/$ - see front matter © 2011 Elsevier Ltd All rights reserved.
doi:10.1016/j.is.2011.09.007

entity appears. They are on the one hand rich sources of information, but on the other hand this source is often not so straightforward to exploit, as it is very hard to distinguish the relevant information from noise and the relevant information might be even missing.

Entity matching is also essential for realizing the Semantic Web. In order to process information on Web pages automatically, one needs to identify the entities in Web documents and then match them to other entities in entity collections or to entities described by ontologies. Entity matching is also needed to create such large entity collections themselves. This process is described in [9]. Linking entities present in unstructured Web documents to each other can in many ways contribute to the development of the Semantic Web, independently of whether such large entity repositories will emerge.

We study two specific variants of the general entity matching problem, namely the *person name disambiguation* problem and the *Twitter message classification*. In the person name disambiguation problem we are given a set of Web documents, each containing a given name and the goal is to cluster the documents such that two documents are in the same cluster if and only if they refer to the same real-world person. In the Twitter classification problem, we are given a set of Twitter messages, each containing a particular keyword, which is a company name. The goal is to classify the messages whether they are related to the company or not. For this problem, we develop company profiles and the task is then to match these profiles to the messages. While these problems require some specific algorithmic techniques, they both can be seen as entity matching problems. We use these settings to demonstrate our quality-aware similarity assessment technique.

Similarity functions try to capture the degree of belief about whether two entities refer to the same real-world entity. There is a number of known techniques to derive similarity values. One can observe that the quality of these methods varies and highly depends on the input, and specific features of the input. The quality-aware similarity assessment technique combines similarity assessments from multiple sources. As opposed to other combination methods, we estimate the accuracy of individual sources for specific regions of the input (i.e. they are not global estimations) and we use this accuracy estimate for combining similarity values. Additionally, as we are dealing with Web data, the lack of information poses an additional difficulty. We give particular attention to this challenge that is often not addressed by techniques in the machine learning literature.

We describe a systematic design of similarity assessment particularly suited for Web data, including novel ways of partitioning the input for quality-estimations. At the same time we demonstrate that one can obtain and accuracy comparable or even better than the state-of-the-art methods with a very simple algorithmic technique, with the help of quality-aware similarity assessment. We analyzed our techniques experimentally, on real-world datasets. The results show systematic improvements compared with state-of-the-art methods. While we are studying the quality improvements within our algorithmic framework, we think that our quality-aware similarity assessment technique can lead to quality improvements in other entity matching algorithms as well.

The rest of the paper is organized as follows. Section 2 discusses the general entity matching problem and our method of constructing quality-aware similarity functions. Section 3 elaborates on the person-name disambiguation problem, while Section 4 discusses the Twitter classification problem; both sections present algorithmic frameworks which make use of quality-aware similarity functions. Section 5 contains details on the experimental evaluation, Section 6 summarizes related work and finally Section 7 concludes the paper.

2. Quality-aware similarity assessment

2.1. Problem definition

We consider the following general entity matching problem. We are given two sets of Web documents D_A and D_B (for example, Web pages, Twitter messages, semi-structured profiles), such that each document $d \in D_A$ (or $d' \in D_B$) is associated with some named entities (for example, persons, geographic locations, companies, organizations, etc.). We assume that the set of named entities is already extracted, and they are available as sets A and B which are extracted references to the same entity type. Let R_A and R_B be the set of real world entities and for an entity $a \in A$, let $r(a) \in R_A$ denote the corresponding real world entity. The entity matching problem aims to find the pairs (a,b), such that $a \in A$, $b \in B$ and a and b are representing the same real-world entity, i.e. $r(a) = r(b)$. Note that in some cases the set of real world entities or their relation is not known, or only partially known. In such cases, our goal is to find the pairs that best corresponds to our available training sets.

In particular, we study two specific variants of the general entity matching, namely the *person name disambiguation* problem and the *Twitter message classification* problem. In the case of person name disambiguation problem we are given a set of documents, containing a particular name. In this setting the set D_A and D_B coincides (this is our document collection) and the goal is to cluster the set of documents, such that each document within a cluster refers to the same real-world person. In our document collection, for a given name, each document refers to only one of the persons. In other terms, there is a one-to-one correspondence between a name and a document. This assumption simplifies the algorithmic framework. Our quality-aware similarity assessment techniques are applicable also in the more complex algorithmic framework that is needed, if we drop this assumption. The number of persons (with the same name) is not known in advance. In the case of Twitter classification the set of documents D_A is a set of Tweet messages, each containing a given company name (for example, Apple). The set D_B is a set of profiles (see Section 4) for a given company (with the same name as D_A) and the goal is to identify whether the documents in D_A (i.e. the tweets) are really referring to the company or not, for example, decide whether the word “apple” in a tweet refers to the company Apple, represented in the profiles or something else (e.g. a fruit).

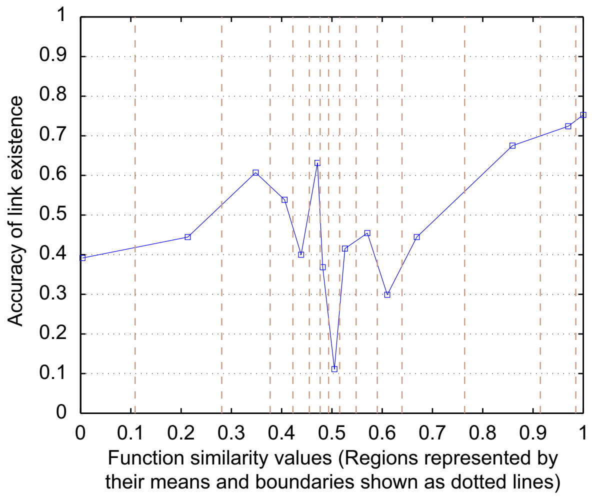

Fig. 1. Accuracy of a similarity function.

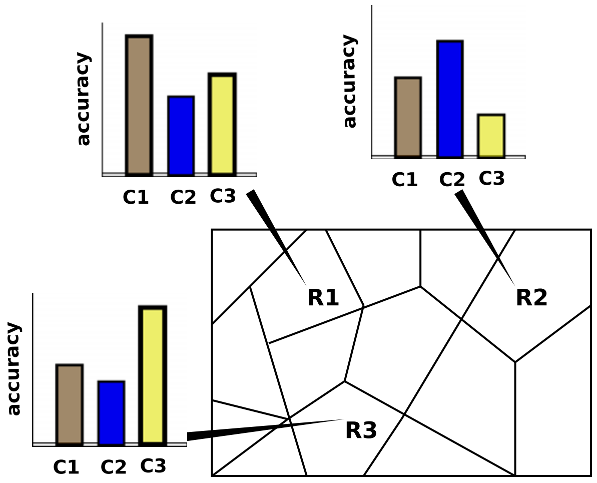

Fig. 2. The accuracy of the similarity values varies depending on the region of the input. For example, *C*1 might have overall the best accuracy, while in region *R*3 the function with the best accuracy is *C*3.

2.2. *Challenges of assessing similarities*

Assessing similarities between entities in Web documents is a challenging task. One faces (among others) the following difficulties:

- Similarity assessments focus on some specific features of the entities only. It is not clear what features one should compare and with which technique.
- Independently of which feature one chooses for assessing similarities of entities, it is likely that the Web documents contain incomplete, imprecise information about the entities or the relevant information may be completely missing. As a result, the similarity assessment techniques are often inaccurate.
- Moreover, they have varying accuracy on different inputs and even on different parts of the same input.

In the following we give an example for the above-mentioned problem of varying accuracy, from our own experiments. Fig. 1 shows accuracy of similarity values on a training set. On the *x*-axes one can see the similarity values, while on the *y*-axis is the accuracy of the measured value. For a given interval of similarity values, we computed, how many entities match (based on the ground truth). In real datasets we do not get monotonic figures, where higher similarity values indicate an entity match with higher accuracy, rather the type of graph in Fig. 1. (The values are obtained for the person "Cohen", in the WWW'05 dataset, see Section 5.2. Even if the actual values might depend on the dataset, the variation of accuracy is a common phenomenon.)

2.3. *Matching with quality-aware similarity assessment*

We propose a technique that addresses the above problems. While elements of this technique are known and also used elsewhere (see Section 6), we apply them systematically and in novel ways. As a result, with the help of a very simple algorithmic framework that we explain below (Algorithm 1) we could obtain matching results even better than the state-of-the-art methods.

1. We first *compute similarity values*, using multiple techniques, since we do not know beforehand which feature to look for. The similarity functions are specific to the particular problem, we will explain them in detail in Sections 3 and 4.
2. We *partition the input* into regions. In a smaller region we can much more reliably *estimate the accuracy* of the similarity values (Fig. 2) than for the entire function, because each function has varying accuracy in each of these regions. In our paper we used several ways to identify these regions. We explain the techniques, which are specific to the Web context, in Sections 3 and 4.
3. Using the accuracy estimations, we *combine* the similarity values using different combination techniques into one single similarity value that we finally use to *decide* whether two entities match or not.

Algorithm 1. Quality-aware entity matching.

```
compute similarity values, using multiple methods
identify regions of the input, where we can estimate the quality of
the computed similarity values
estimate the accuracy of each similarity value, for each region
combine the similarity values using the estimated accuracy
decide whether the entities match
output the decision
```

3. Person-name disambiguation

In this section we discuss the person name disambiguation problem and the use of quality-aware similarity functions for this problem. This problem is relevant for many applications, for example for person search engines who collect information from Web pages, or for news agencies (or for the online publishing industry in general).

To enrich and to interlink online information (e.g. to construct `owl:sameAs` statements) person name disambiguation is essential.

Our technique relies on the quality-aware similarity assessment, and uses a simple algorithmic framework. First, in Section 3.1, we elaborate on the basic similarity functions we used. Then, in Section 3.2 we explain how we defined the regions of the input and how we estimate the accuracy of the basic similarity functions, finally in Section 3.3 we explain the algorithm addressing the person name disambiguation problem.

3.1. *Basic similarity functions*

Similarity functions associate a value from the interval [0,1] to a pair of entities. (We use normalized values if the function does not return values in [0,1]). In our case, instead of comparing the entities themselves, we compare the related web-pages. As a preprocessing step we apply information extraction tools, so the input to the similarity functions is the extracted information and not the pages themselves. In other terms, we apply (dictionary-based) named entity recognition techniques.

Each similarity function compares two webpages based on a particular feature (like concepts, URLs, etc.) using a similarity measure (like cosine similarity, number of overlaps, etc.) [10,11]. We use common observations in coming up with the following similarity functions. Two webpages are about a same person, if the concepts or organizations or person names, etc. mentioned on the pages are similar/overlap, or if the pages URLs are on a same Web domain.

Regarding the implementation: For extracting features from the webpages we used several information extraction tools, including “alchemy API” [12] to extract named entities, “GATE” [13], “openCalais” [14] to extract other types of entities, such as organizations and locations. We also extract wikipedia-based concepts using “semhacker” [15]. Finally for representing a webpage as document vector we use the services provided by lucene [16]. The similarity functions we consider are summarized in Table 1.

3.2. *Quality-aware similarity assessment*

3.2.1. *Accuracy estimations*

We estimated the accuracy of individual similarity functions in different ways. These include *global accuracy estimates*, where we give an overall estimate for the entire similarity function and *region-based estimates*, where we partition the input into smaller regions, where we can do estimations much more reliably.

Global accuracy estimation: Given a single similarity function, we can consider two related persons equivalent if their similarity value is higher than an appropriately chosen threshold. Indeed, for each function we have chosen such a threshold, and based on the training set, we estimated the accuracy of the threshold-based decision: we computed, what is the percentage of correct matches, if we would consider that two entities with similarity values above the threshold do match.

The accuracy of such decisions clearly depends on the choice of the threshold. For each function, we have chosen a threshold, which – based on the training set – maximizes the number of correct decisions. We used these estimations as a base-line for our experiments.

As we discussed in Section 2, as an alternative to global accuracy estimations, we can partition the input to smaller regions and compute accuracy estimates for these parts.

Region-based accuracy estimation: We tried multiple ways to divide the input into regions:

1. We defined the regions based on the similarity values: we divided the similarity values to equal sized sub-intervals: [0,0.1),[0.1,0.2), ... ,[0.9,1], and one region consists the pairs having the values in a given range. This is a very simple definition, however, the similarity values do not have a uniform distribution in the [0,1] interval, thus by this definition, some regions contain significantly larger than others.
2. We clustered the similarity values corresponding to the training set using the *k-means* clustering technique. (We have chosen $k=15$.) The pairs whose similarity values fall into one cluster form a region.

In the case of the functions $F5$ and $F6$ we further divided the regions we constructed in this way. The function $F6$ computes the number of overlaps of person names in the corresponding Web documents. If we obtain the value 0, this can have multiple reasons. Either (one of the) Web documents do not contain such person names, or they both contain person names, but the two sets are different. In such cases, we defined the regions using “dimensions”: the similarity value and the existence/non-existence of information. We call this extension *improved*

Table 1
Basic similarity function descriptions.

Fn.	Feature	Similarity measure
$F1$	Weighted concept vector	Cosine similarity
$F2$	URL of the page	String similarity
$F3$	Most frequent name on the page	String similarity
$F4$	Concepts vector	Number of overlapping concepts
$F5$	Organizations entities on the page	Number of overlapping organizations
$F6$	Other person-names on the page	Number of overlapping persons
$F7$	The name closest to the search keyword	String similarity
$F8$	TF-IDF (based weights) words vector	Cosine similarity
$F9$	TF-IDF (based weights) words vector	Pearsons correlation similarity
$F10$	TF-IDF (based weights) words vector	Extended Jaccard similarity

definition of regions (and the corresponding functions *iF*5, *iF*6), as opposed to *basic* definitions, where we only rely on similarity values.

Based on the training set, for each region we computed an accuracy estimate. From the training sample set, each region would contain certain sample points corresponding to link existence and non-existence. Accuracy for a region is then defined as the percentage of the sample points representing link existence. If this value is lower than 0.5 then it suggests that the majority pairs should not be considered as a link. Note that the accuracy estimations are based on the small training set and not on the entire data, so computationally the method remains feasible.

3.2.2. *Combining multiple functions*

Given the heterogeneity of the Web, we cannot expect that we can design a single similarity function which would perform optimally in all cases. To overcome this problem we compute several similarity functions and try to make our decision based on a combination of the similarity functions. To find a suitable way of combination involves a lot of challenges.

The different functions report similarity values with very different value distributions as they capture different aspects of similarity. Thus instead of combining the similarity values themselves, we try to combine the decisions (whether or not to consider two entities as equivalent) and the estimated accuracy values. Our decision criteria for a single function was to have a similarity value above the threshold and having an estimated accuracy (for the corresponding region) at least 0.5.

In this way, for each function f_i we obtain a graph $G^{f_i}_{D_j}$, together with accuracy estimates, where D_j is the decision criteria, i.e. whether we decide upon a single threshold or also consider the accuracy estimates. Our goal is to combine the individual graphs $G^{f_i}_{D_j}$ into a single graph $G_{combined}$. First we obtain a multi-graph, where the multiple edges between two nodes correspond to the edges from the individual graphs. We weight the edges with the individual accuracy estimates, which we consider as estimations of the probability of a link. Then we compute a weighted average and obtained an optimal threshold, based on our training set. If the combined value is above this threshold we add an edge to $G_{combined}$.

We also considered other combination techniques. Instead of considering the weighted average of the values, we used other aggregation functions, namely we have selected the maximum value. Interestingly, this combination technique performed the best on our datasets, which might not always be the case. It is important to note that not always the same function performed the best.

3.3. *Entity resolution algorithm*

We say that two entity references (names) n_i and n_j are equivalent ($n_i \equiv n_j$) if they refer to the same person. Clearly this relation is transitive. The relation of the entity references can be represented as a graph, in which for each entity reference there is a vertex in the graph, and two vertices are connected by an edge whenever the two corresponding entities are equivalent. We refer to this graph as the entity graph. The goal of the entity resolution algorithms is to reconstruct this entity graph as accurately as possible. Note that the entity graph has very specific properties: it is not a connected graph, it is a union of pairwise disjunct connected components and each component is a clique, i.e. a complete graph, because of the transitivity of the equivalence relation.

Our entity resolution technique is the following. First we compute a complete weighted graph $G^{f_i}_{w}$ for each similarity function f_i. (The nodes of the graph $G^{f_i}_{w}$ correspond to the Web pages, while the weights on the edges are the similarity values reported by f_i.) To avoid computational bottlenecks, we apply a basic blocking technique, so essentially we only compute the similarity values between documents, which are about a person with the same name.[1] From the graph $G^{f_i}_{w}$ we would like to obtain a graph G_{D_j}, a (not-weighted) graph, where an edge between two nodes shall indicate whether the entities corresponding to the nodes are the same. This transformation depends on the decision criteria D_j. These decision criteria include to choose values above a threshold or also consider accuracy estimates, as it is explained in Section 3.2.1. Once we have all the graphs $G^{i}_{D_j}$, for all functions f_i and all decision criteria D_j, we obtain a combined graph $G_{combined}$, which is explained in Section 3.2.2. For this we also use accuracy estimates $acc(G^{i}_{D_j})$, based on the training set. Finally, we apply clustering techniques to obtain the final entity resolution. In our implementation we compute the transitive closure of the graph $G_{combined}$, but we also experimented with several other clustering techniques, such as correlation clustering [17]. The overall procedure is summarized in Algorithm 2.

Algorithm 2. Entity resolution.

```
compute the graph $G^{f_i}_{w}$ for each $f_i$ (per block)
obtain the decision criteria $D_j$(threshold, regions, etc.) from the training set
apply the decision $D_j$ to the data, to compute $G^{i}_{D_j}$, for each $i$ and $D_j$
compute the accuracy $acc(G^{i}_{D_j})$
combine them, for all $i$, $D_j$
apply a clustering algorithm
output the final entity resolution
```

4. Classifying Twitter messages

In this section we focus on a second problem, where we apply our quality-aware similarity assessment techniques, namely the Twitter classification problem. Twitter[2] is a popular service where users can share short messages (a.k.a. tweets) on any subject. Twitter is currently one of the most popular sites of the Web: as of February 2010, Twitter users send 50 million messages

[1] Such blocking strategy is very natural in the datasets we used, where the documents already organized around person names. In general, one needs to consider the applicable blocking schemes more carefully.

[2] http://twitter.com

per day.[3] As users are sharing information on what matters to them, analyzing twitter messages can reveal important social phenomena, indeed a number of recent studies [18]. Clearly, twitter messages are also a rich source for companies, to study the opinions about their products. To perform sentiment analysis or obtain reputation-related information, one needs first to identify the messages which are related to a given company. This is a challenging task on its own as company or product names are often homonyms. This is not accidental, companies deliberately choose such names as part of their branding and marketing strategy. For example, the company Apple Inc. shares its name with the fruit apple, which again could have a number of figurative meanings depending on the context, for example, "knowledge" (Biblical story of Adam, Eve and the serpent) or New York (the Big Apple).

In this section we focus on how to relate tweets to a company that can be seen as a special case of the entity matching problem. We assume that we are given a set of companies and for each company a set of tweets, which might or might not be related to the company (i.e. the tweets contain the company name, as a keyword). Constructing such a matcher is a challenging task, as tweet messages are very short (maximum 140 characters), thus they contain very little information, and additionally, tweet messages use a specific language and often also incorrect grammar, they are full with proprietary abbreviations, which are hard to interpret without further background knowledge. To overcome this problem, we constructed profiles for each company, which contain more rich information. For each company, in fact, we constructed several profiles, some of them automatically, some of them manually. The profiles are essentially sets of keywords, which are related to the company in some way. We also created profiles, which explicitly contains unrelated keywords. Once we have the profiles, we are facing an entity matching problem. In this context, we make use of our quality-aware similarity assessment.

Below, in Section 4.1 we give a more precise problem definition. In Section 4.2 we explain how we represent Tweet messages and company profiles. We explain in Section 4.3 the use of quality-aware similarities and our Twitter classification technique.

4.1. Problem statement

In this section we formulate the problem and our computational framework more formally. The task is concerned to classify a set of Twitter messages $\Gamma = \{T_1, \ldots, T_n\}$, whether they are related to a given company C. We assume that each message $T_i \in \Gamma$ contains the company name as a sub-string. We say that the message T_i is related to the company C, $related(T_i, C)$, if and only if the Twitter message refers to the company. It can be that a message refers both to the company and also to some other meaning of the company name (or to some other company with the same name), but whenever the message T_i refers to company C we try to classify as TRUE otherwise as FALSE. The task has some other inputs, such as the URL of the company $url(C)$, the language of the webpage, as well as the correct classification for a small number of messages (for some of the companies).

For the Twitter classification problem, we assume that we have training sets corresponding for a few companies (C^{TR}). Our goal is to classify test sets corresponding to new (unseen) companies (C^{Test}), for which we do not have training data, i.e. $C^{TR} \bigcap C^{Test} = 0$.

4.2. Information representation

The tweet messages and company names alone contain very little information to realize the classification task with good accuracy. To overcome this problem, we created profiles for the companies, several profiles for each company. These set of profiles can be seen as a model for the company. In this section, we discuss how we represent tweet messages and companies and we briefly discuss how we obtained these profiles.

4.2.1. Tweet representation

We represented a tweet as a bag of words (unigrams and bigrams). We do not access the tweet messages directly in our classification algorithm, but apply a preprocessing step first, which removes all the stop-words, emoticons, and twitter specific stop-words (such as, for example, RT, @username). We store a stemmed[4] version of keywords (unigrams and bigrams), i.e.

$$T_i = set\{wrd_j\}.$$

4.2.2. Company representation

We represent each company as a collection of profiles, formally

$$E^k = \{P_1^k, P_2^k, \ldots, P_n^k\}.$$

Each profile is a set of weighted keywords, i.e. $P_i^k = \{wrd_j : wt_j\}$, with $wt_j \geq 0$ for positive evidence (i.e. keywords, which – if contained in a message – shall indicate that the message is related to the company) and $wt_j < 0$ for negative evidence.

For the tweets classification task, we eventually compare the tweet with the entity (i.e. company) profile. For better classification results, the entity profile should have a good overlap with the tweets. Unfortunately, we do not know the tweet messages in advance, so we tried to create such profiles from alternative sources, independently of the tweet messages. The entity profile should not be too general, because it would result many false positives in the classification and also not too narrow, because then we could miss potential relevant tweets.

We generated most of our profiles automatically, i.e. if one would like to construct a classifier for a previously unseen company, one can automatically generate the

[3] http://www.telegraph.co.uk/technology/twitter/7297541/Twitter-users-send-50-million-tweets-per-day.html

[4] Porter stemmer from python based natural language toolkit available at http://www.nltk.org.

profiles. Further, small, manually constructed profiles further improve the accuracy of the classification process.

We used the following profiles: the *homepage profile* contains keywords, extracted from the Web page of the company, the *metadata profile* relies on the metadata of the Web page. The *category profile* contains keywords relevant to the domain of the company. Similarly, for *common-knowledge profile* we obtained relevant keywords from GoogleSets.[5] We also defined *user-feedback-profiles* containing positive and negative keywords from users. For more details on semi-automatic profile construction see [19]. Table 2 shows how an "Apple Inc."[6] company entity is represented using different profiles. As we constructed the profile semi-automatically, some of the keywords might be incorrect or hard to interpret.

4.2.3. Features extraction

We define a feature extraction function, which compares a tweet T_i to the company entity representation E_k and outputs a vector of features.

$$Fn(T_i,E_k) = \{\overbrace{G_1,\ldots,G_m}^{\textit{profile-features}},\underbrace{F_1,\ldots,F_n}_{\textit{tweet-specific}},\overbrace{U_1,\ldots,U_z}^{\textit{ad-hoc}}\}.$$

Profile features: Here the G_i $(i \in [1,m])$ are profile-specific, which are entirely based on the quality of the entity profiles and do not depend on Tweet message T_i. One could use different ways of quantifying the quality of the profiles.

- *Boolean*: In this work we make use of boolean metrics to represent if a profile is empty or has sufficient keywords.
- Other possibility is that a human can inspect the profiles and assign a metric of $x \in [0,1]$ based on the perceived quality. One could think of exploring an automated way of assigning this number.

Tweet-specific features: The F_i $(i \in [1,n])$ features are tweet specific features, i.e. they quantify how close a tweet overlaps with the entity profiles. We use a comparison function to compare the tweet message T_i, which is a bag of words, with jth profile P_j^k, which is also a bag of weighted keywords, to get the F_jth feature. In this work we use Boolean overlap as one of the comparison functions, which compares two bags of words looking for exact overlap of keywords, and for all such keywords the sum of their weights quantify how close the tweet message is to the entity profile. Formally with $T_i = Setw_1^t, w_2^t, \ldots, w_k^t\}$ and $P_j^k = Setw_1^p : wt_1, w_2^p : wt_2, \ldots, w_m^p : wt_m\}$, we compute the F_j feature using the Boolean overlap comparison function as

$$F_j = BooleanOverlap(T_i,P_j^k) = \sum_q wt_q, \text{ where } q$$

is the index of overlapping words, i.e.

$$w_q^p \in Set\{w_1^t,w_2^t,\ldots,w_k^t\} \bigcap Set\{w_1^p,w_2^p,\ldots,w_m^p\}. \quad (1)$$

The above comparison function is simple and easy to realize, but it may miss out some potentially similar words. We also make use of Edit-Distance and Jaro similarity based comparison functions to identify similar words.

Ad hoc features: The U_i $(i \in [1,z])$ features encapsulate some user based rules, for example, presence of the company URL domain in the tweet URL list is a big enough evidence to classify the tweet as belonging to the company.

4.2.4. Classification process

The classifier is a function which takes the feature vector as input and classifies the tweet as $\{TRUE,FALSE\}$, with *TRUE* label if the tweet is related to the company and *FALSE* otherwise. We use the Naive Bayes classifier model for designing the individual classifiers. We have chosen to use the Naive Bayes technique, as it was easy to realize and still promises acceptable accuracy. For each company in the training set (C^{TR}), based on the company tweets, we find the conditional distribution of values over features for two classes, for the class of tweets which are related to the company and the another class of tweets, which are not related. With the help of these conditional probabilities, as shown in Eqs. (2) and (3) and by applying Bayes theorem, we can classify an unseen tweet whether it is related to the company or not.

Let us denote the probability distribution of features of the tweets that are related to a given company with

$$P(f_1,f_2,\ldots,f_n|C), \quad (2)$$

and the probability distribution of features of the tweets that are not related to the company with

$$P(f_1,f_2,\ldots,f_n|\overline{C}). \quad (3)$$

Then, for an unseen tweet t, using the features extraction function we compute the features values: $(f_1,f_2,\ldots,f_n)$. The posterior probabilities of whether the tweet is related to the company or not are calculated as in the following equation:

$$P(C|t) = \frac{P(C)*P(t|C)}{P(t)} = \frac{P(C)*P(f_1,f_2,\ldots,f_n|C)}{P(f_1,f_2,\ldots,f_n)}, \quad (4)$$

$$P(\overline{C}|t) = \frac{P(\overline{C})*P(t|\overline{C})}{P(t)} = \frac{P(\overline{C})*P(f_1,f_2,\ldots,f_n|\overline{C})}{P(f_1,f_2,\ldots,f_n)}. \quad (5)$$

Depending on whether $P(C|t)$ is greater than $P(\overline{C}|t)$ or not, the Naive Bayes classifier decides whether the tweet t is related to the given company or not, respectively.

4.3. Entity matching with quality-aware similarities

Given the representation we explained in Section 4.2, the Twitter classification can be seen as an entity matching problem. We address the problem with our quality-aware entity matching strategy (Algorithm 3). First we design individual classifiers, based on our training set. Each of them uses a subset of all features or possible

[5] GoogleSets http://labs.google.com/sets is a service that generates a set of keywords, given a few examples.

[6] http://www.apple.com

Table 2
Apple Inc. company profiles.

Profile Type	Keywords
HomePage profile	iphone, ipod, mac, safari, ios, iphoto, iwork, leopard, forum, items, employees, itunes, credit, portable, secure, unix, auditing, forums, marketers, browse, dominicana, music, recommend, preview, type, tell, notif, phone, purchase, manuals, updates, fifa, 8 GB, 16 GB, 32 GB, etc.
Metadata profile	{empty}
Category profile	opera, code, brainchild, movie, telecom, cruncher, trade, cathode-ray, paper, freight, keyboard, dbm, merchandise, disk, language, microprocessor, move, web, monitor, diskett, show, figure, instrument, board, lade, digit, good, shipment, food, cpu, moving-picture, fluid, consign, contraband, electronic, volume, peripherals, crt, resolve, yield, server, micro, magazine, dreck, byproduct, spiritualist, telecommunications, manage, commodity, flick, vehicle, set, creation, procedure, consequence, second, design, result, mobile, home, processor, spin-off, wander, analog, transmission, cargo, expert, record, database, tube, payload, state, estimate, intersect, internet, print, factory, contrast, outcome, machine, deliver, effect, job, output, release, turnout, convert, river, etc.
GoogleSet profile	itunes, intel, belkin, 512 mb, sony, hp, canon, powerpc, mac, apple, iphone, ati, microsoft, ibm, etc.
UserFeedback positive profile	ipad, imac, iphone, ipod, itouch, itv, iad, itunes, keynote, safari, leopard, tiger, iwork, android, droid, phone, app, appstore, mac, macintosh
UserFeedback negative profile	fruit, tree, eat, bite, juice, pineapple, strawberry, drink

Table 3
Individual classifiers.

Set1	Set2	Profiles used	Comparison fn
BB1	B1	**X**=[Homepage Profile, Category Profile, Metadata Profile, GoogleSet Profile, UserFeedback Positive Profile, UserFeedback Negative Profile]	BooleanOverlap
BB2	B2	**X**	EditDistance
BB2	B2	**X**	JaroSimilarity
BB4	B4	[Homepage Profile]	**Y**=[BooleanOverlap, EditDistance, JaroSimilarity]
BB5	B5	[UserFeedback Negative Profile]	**Y**

comparison methods (Table 3). We then identify the regions of the unseen companies and estimate the accuracy of the individual classifiers in these regions. Once we have these accuracy estimates, we combine the decision of individual classifiers for unseen companies and we use these combined values as a basis for our final decision, whether we consider two entities as a match.

Algorithm 3. Twitter classification.

```
compute decisions using multiple individual classifiers
identify the regions in the feature space for the companies in the test set
estimate the accuracy, for each classifier
combine the decisions of the individual classifiers, using the estimates, for unseen companies
decide whether the entities match
output the decision
```

The above algorithm can be seen to proceed in two phases. In the first phase, we construct many individual classifiers based on the training set. We need many individual classifiers, as we do not know beforehand, which set of features and comparison functions one should use. In the second phase, we chose the best possible classifier for the unseen companies. In this phase, we rely on accuracy estimates of the individual classifier in the "neighborhood" of the unseen companies. We explain these phases for two different scenarios.

1. In the first scenario, the individual classifiers are in the Set1={BB1, BB2, BB3, BB4, BB5}. Each classifier in the Set1 is trained *per company and per feature group*. We have in total $|C^{TR}|*5$ individual classifiers. We consider only the classifiers of the companies that are "similar" to the unseen company. For each unseen company, we consider the $K=5$ closest companies ("neighborhood") from the training set as possible candidates, using the dot product distance metric, where each company profile is seen as a vector in the terms-dimension space.
2. For the second scenario, the individual classifiers are in the Set2={B1, B2, B3, B4, B5}. We have one classifier per feature set for the entire training set, i.e. in its design it makes use of tweets of all the companies in the training set. In this case, we have five individual classifiers in total. We divide the companies in the training set into six groups using the *k*-means clustering technique ($k=6$, the number of categories from the training set). Each cluster is considered as a region and we estimate the accuracy of each classifier for each region. Fig. 3 depicts the accuracy estimates of the individual classifiers. The figure suggests that there is no single best classifier and by combining the classifiers we have a chance to achieve better performance. For an unseen company, we first decide to which region it belongs to, by computing its distance to the means of the different regions. As the combination strategy, we choose the most accurate classifier in this region, and we use it as the classifier for the unseen company.

Regarding the computational efforts, in the first case, we are creating many classifiers, but each of them is constructed using a small subset of the training set, containing the relevant company name. In the second case, we have only a few classifiers, each constructed using the entire training set.

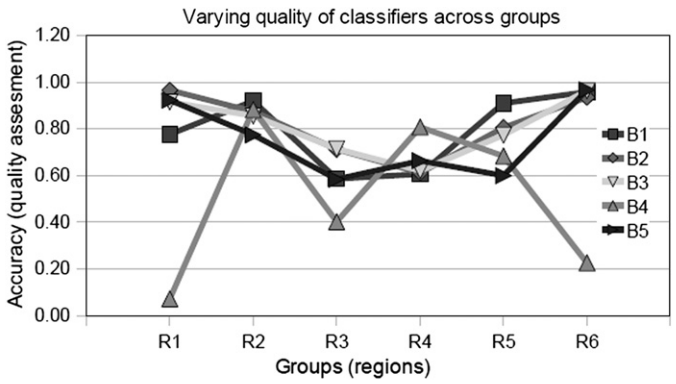

Fig. 3. Accuracy estimates.

Overall, if we have many classifiers making decisions about the entity match, the next question is how can we decide on the final result. One way is to choose the globally most accurate classifier among the many individual classifiers and use it for making the decisions on the test set. The globally most accurate classifier might not necessarily be the best classifier for unseen companies. However, we can do better if we can make use of the accuracy estimates associated with the regions. Other simple alternative combining strategies could be taking the weighted averages, maximal voting, etc. of the individual classifier decisions. For comparison, we also train an SVM classifier [20,21] as a generic classifier, which makes use of all features: profile-features, tweet-specific features and ad-hoc/heuristics-based features, in its classification task.

5. Experimental evaluation

5.1. Experimental setup

We performed our experiments on a 2 GB RAM, Genuine Intel(R) T2500 @ 2.00 GHz CPU. Linux Kernel 2.6.24, 32-bit machine. We implemented our methods using matlab, java and python.

5.2. Person name disambiguation

5.2.1. Datasets

For our experiments for evaluating the person name disambiguation we used two different datasets: the WWW'05 and the WePS-2. The WWW'05 dataset was created in [22]. This dataset was also used in a series of papers, which enabled us to compare our methods with other techniques. The dataset contains Web documents for 12 different person names. The dataset was created by querying the Web using the google search engine with the different person names. The top 100 returned Web documents for the web search were gathered and labeled manually. For each person, the correct resolution is available together with the data. We used this ground truth to measure the quality of our techniques. The number of clusters for each person name is different, it varies from 2 to 61.

WePS-2 test data is provided by the Web people search clustering task [23]. The test data consisted 30 Web page collections, each one corresponding to one ambiguous name. These 30 person names were chosen from three different sources: wikipedia, ACL'08 (Association for Computational Linguistics Program committee members) and US census data. Each person name was queried using yahoo search API and the top 150 results were included into the dataset. We have evaluated our techniques on WePS-2 dataset. We report the performance figures we observed on the 10 person names chosen from the ACL'08.

5.2.2. Measures of interest

Various measures are considered to assess the quality of entity resolution. Precision, recall and *F*-measure are widely used in information retrieval. We also measure the Rand-index [10] and the F_p-measure [11], which is the harmonic mean of purity and inverse purity. They are typically measures from information retrieval or variants of those measures. We summarize here the definitions. Some of these definitions can be found in [10].

An entity resolution algorithm tries to predict the entity graph. Given a prediction graph, one can categorize its links with respect to the ground truth, i.e. the correct entity graph, into four categories: true positives (*TP*), true negatives (*TN*), false positives (*FP*) and false negatives (*FN*). The true positives are links which are correctly predicted while the wrongly predicted links are the false positives. Similarly, the correctly predicted missing links fall into the true negatives category, while wrongly predicted missing links are false negatives. We also denote the number of links in the corresponding category with *TP*, *TN*, *FP*, *FN*.

Precision (*P*), recall (*R*) and *F*-measure (*F*) is defined as

$$P=\frac{TP}{TP+FP},\quad R=\frac{TP}{TP+FN},\quad F=\frac{2PR}{P+R}.$$

Accuracy (a.k.a. Rand index, *RI*) is the percentage of correct decisions for the predicted links:

$$RI=\frac{TP+TN}{TP+TN+FP+FN}.$$

F_p-measure (F_p) is the harmonic mean of purity and inverse purity. Purity is defined as follows [11]: let

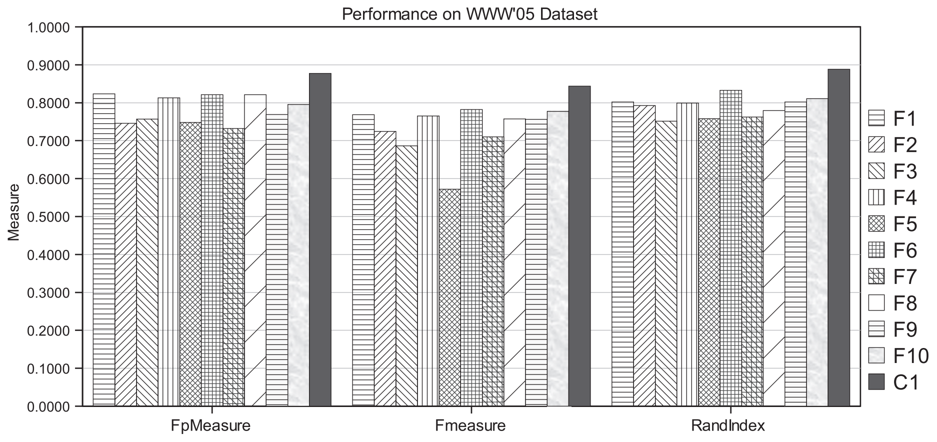

Fig. 4. WWW results graph.

$M=\{M_1,\ldots,M_n\}$ be the clusters of the ground truth and let $C=\{C_1,\ldots,C_m\}$ be the clusters predicted by the algorithm and let $Prec(C_i,M_j)$ denote the precision of C_i w.r.t. M_j. Purity is defined as

$$Pur(C,M)=\sum_{C_i\in C}\frac{|C_i|}{|C|}\max_{M_i\in M} Prec(C_i,M_j),$$

while inverse purity[7] as

$$IPur(C,M)=\sum_{M_i\in M}\frac{|M_i|}{|M|}\max_{C_i\in C} Prec(M_i,C_j).$$

We note here that the above measures rely on the fact, that we know the ground truth, which is unrealistic in the Web context. We could apply them for the document collections in our experiments, as we had this information available.

5.2.3. *Methods*

Given the dataset, we use 10% of the complete dataset as the training set. The performance of the entity resolution (entity matching) algorithm depends on how well the training set represents the features of the complete dataset. In order to avoid any bias, we repeated the experiments for five runs and the averages of the observed results are presented. On each run we randomly choose the training subset from the complete dataset. We make use of a standard 10-fold-cross validation technique to obtain the optimal parameters of a classifier. For computing the parameters we minimize the loss function, i.e. the number of incorrect decisions.

5.2.4. *Experimental results*

Fig. 4 shows the performance of the individual similarity functions on the entire WWW'05 dataset. The figure shows three metrics, namely F_p-measure, F-measure and Rand-index. The final column, depicted as black in the figure, is the combined performance of our quality-aware combination technique, which clearly shows improved performance. Similarly, Fig. 5 shows the experimental results on the WePS-2 dataset.

Table 5 contains the achieved F_p values, for each individual person, by each individual function in the WWW'05 dataset. One can observe that each function performs differently for different persons. For example, for "Voss" the function F8 has the highest F_p-value, while for "Mulford" the best function is F6.

Table 4 shows that by considering more and more functions we indeed get a better performance for both datasets. The first three columns show the maximal performance considering just the threshold-based technique, by including functions $I4=\{F4,F5,F7,F9\}$, $I7=\{F3,F4,F5,F7,F8,F9,F10\}$, $I10=\{F1,\ldots,F10\}$, respectively. The columns $C4$, $C7$ and $C10$ take the same functions as the first three columns, respectively, but there we choose the best decision criteria, based on accuracy estimation of the regions. The column W shows the performance of weighted average combination result. The table also contains a comparison with the figures reported in the literature. The best results for the WWW'05 dataset were reported in the paper [25], however, they manually improved the available ground truth (and the improved data is not public), therefore the comparison is not precise. The last column contains the result achieved by the WePS-2 competition winner. We found this result in [24], but we could not obtain the original reference.

Fig. 6 depicts the Rand index values, for two similarity functions $F5$ and $F6$ (person overlap, organization overlap, see Table 1), on the WWW'05 dataset. For both functions, the left bar (F5,F6) is the mean Rand index value across all the person names, where we were relying on regions defined by similarity values only (basic regions). For obtaining the value depicted in the right bar ($iF5$,$iF6$), we refined the regions and also consider whether a value is missing or not (improved regions). The improved performance is due to the refinements in the definition of the regions. Low similarity values can have many reasons; here we distinguish the cases where information is missing from the cases, where the information is dissimilar. Considering missing information does not increase the number of regions, thus it does not increase

[7] The name "inverse purity" is supported by the fact that $Pur(C,M)=IPur(M,C)$.

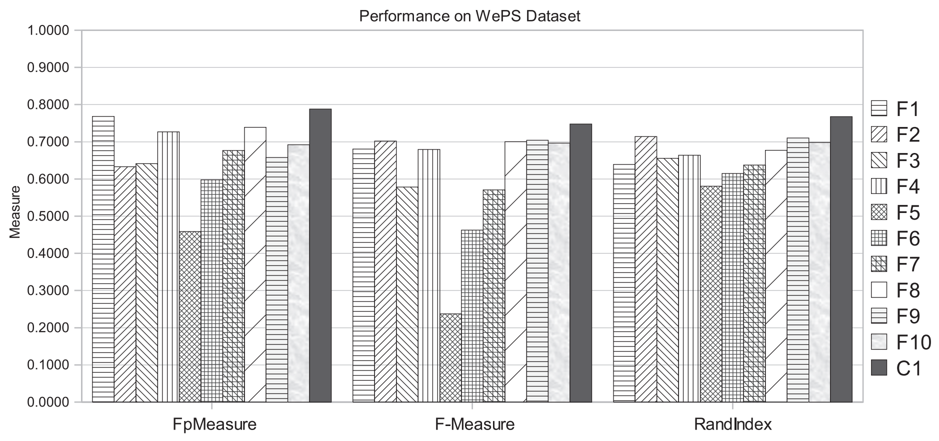

Fig. 5. WePS-2 results graph.

Table 4
Comparison of results.

Dataset	Metric	*I*4	*I*7	*I*10	*C*4	*C*7	*C*10	*W*	Related work
WWW'05	F_p-measure	0.8128	0.8211	0.8232	0.8537	0.8732	0.8774	0.8371	0.864 [24], 0.9000 [25]
	F-measure	0.7654	0.7773	0.7822	0.8338	0.8376	0.8438	0.8168	0.8000 [22], 0.8 [25]
	Rand index	0.8018	0.8109	0.8326	0.8747	0.8814	0.8886	0.8531	
WePS-2	F_p-measure	0.7270	0.7388	0.7682	0.7560	0.7659	0.7880	0.7785	0.791 [24], WePS-2: 0.7800
	F-measure	0.7042	0.7042	0.7042	0.7127	0.7231	0.7476	0.7190	
	Rand index	0.7102	0.7102	0.7139	0.7492	0.7531	0.7675	0.7290	

Table 5
F_p measure for each name in WWW'05 dataset.

Person Name	*F*1	*F*2	*F*3	*F*4	*F*5	*F*6	*F*7	*F*8	*F*9	*F*10	*C*10	*W*
Cheyer	0.9686	0.9948	1.0000	0.9686	0.7950	0.9948	1.0000	0.9948	0.9948	0.9948	1.0000	0.9948
Cohen	0.8724	0.3827	0.7368	0.8859	0.8444	0.8991	0.8839	0.8746	0.8746	0.8718	0.8991	0.8816
Hardt	0.8680	0.8828	0.8985	0.8680	0.4717	0.9074	0.8985	0.8828	0.8828	0.8779	0.9074	0.8828
Israel	0.8206	0.7568	0.7881	0.8312	0.8093	0.8476	0.7257	0.8315	0.7536	0.7568	0.8476	0.8690
Kaelbling	0.9831	0.9944	0.9711	0.9831	0.9012	0.9467	0.9711	0.9944	0.9888	0.9944	0.9944	0.9944
Mark	0.7871	0.7871	0.7228	0.7871	0.7871	0.7871	0.7668	0.7871	0.7915	0.7871	0.8104	0.7871
Mccallum	0.7921	0.7391	0.6642	0.7812	0.8066	0.8248	0.4667	0.8024	0.5851	0.8187	0.9670	0.8597
Mitchell	0.8473	0.7756	0.5796	0.7417	0.7981	0.7733	0.4448	0.5966	0.7097	0.7382	0.8575	0.6448
Mulford	0.7471	0.7467	0.7569	0.7471	0.7337	0.7582	0.7569	0.7467	0.7467	0.7467	0.8053	0.7467
Ng	0.8607	0.7111	0.7493	0.8660	0.7938	0.8163	0.7031	0.8086	0.7082	0.7082	0.8813	0.8845
Pereira	0.7215	0.5420	0.6362	0.7180	0.6389	0.6942	0.5571	0.7326	0.5554	0.5519	0.7573	0.7438
Voss	0.6094	0.6365	0.5813	0.5760	0.5993	0.6073	0.6135	0.8016	0.6391	0.6979	0.8016	0.7567

the computational efforts needed for computing the accuracy estimates.

Fig. 7 depicts the improvements in the combination. The value iC10 is obtained using the refined regions, $\{F1,\ldots,iF5,iF6,\ldots,F10\}$, i.e. as opposed to *I*10, the functions *F*5 and *F*6 are replaced by *iF*5 and *iF*6.

We also wanted to see how far are our methods from the accuracy values that we could have potentially achieved using the same similarity functions. We defined an oracle combination function as follows. Whenever one of the classifiers has a decision agreeing with the ground truth, the oracle decides on that value. We computed the accuracy of this oracle combination function (Ora10,iOra10) and we depict it together with C10 and iC10 in Fig. 8.

5.3. *Twitter classification*

5.3.1. *Dataset*

Our experimental setup was the following. For our experiments we used the WePS-3 dataset, which is available at http://nlp.uned.es/weps/weps-3/data. We are given a general training set, which consists tweets related to about 50 companies (we denote this set as C^{TR}). For each company $c \in C^{TR}$ we are provided around 400 tweets with their corresponding ground truth, i.e. if the tweet is related to the company or not. For each company, we are provided with the following meta-information: URL, Language, Category. We have trained a generic *classifier* based on this training set. The test set for this task consisted tweets of around 50 new companies. We denote this set of companies as C^{Test}. There was no overlap with the training set, $C^{TR} \bigcap C^{Test} = 0$. For each company $c \in C^{Test}$ there are about 400 Tweets, which are to be classified. We classified them with the classifiers explained in Section 4.2.4.

5.3.2. Experimental results

The task is of classifying the tweets into two classes: one class which represents the tweets related to the company (positive class) and second class represents tweets that are not related to the company (negative class). Our performance metric for the evaluation was accuracy.

We conduct two series of experiments. In the first case we design five classifiers (BB1, …, BB5) per company in the training set. For each training set company, we compute how a particular classifier performs in its neighborhood and take it as an accuracy estimate for that classifier in that region. Given these set of individual classifiers, we would like to see the performance of quality-aware combination against the case of choosing the most accurate classifier for the complete test set. The results are shown in Fig. 9.

In the second series of experiments, we design five global classifiers (B1, …, B5) for the complete training set. These five global classifiers would have different performances on each individual company in the training set. This performance is taken as an accuracy estimate of a particular global classifier for that region. For quality-aware combination, we choose the global classifier with most accuracy in the region of the company in consideration. We also show the SVM classifier, which makes the quality-aware combination decision implicitly, as a comparison. The results are depicted in Fig. 10. We compare these quality-aware combination against these five global classifiers used against all the test set.

In both figures, we see that quality-aware combination techniques outperform the other techniques which do not take regions based accuracy into consideration.

The amount of accuracy one could achieve with our technique depends on two factors. They are the quality of

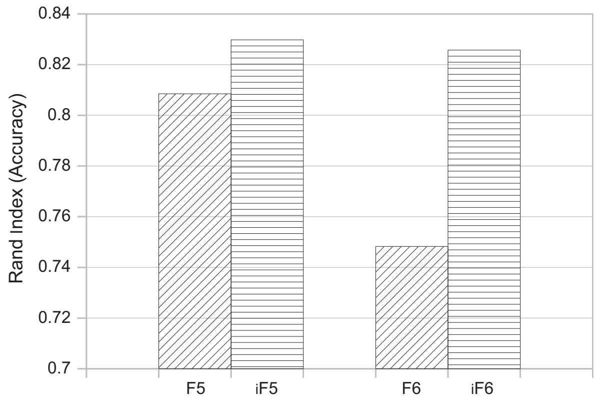

Fig. 6. Improvements in the Rand index for individual functions. (Basic vs. improved regions.)

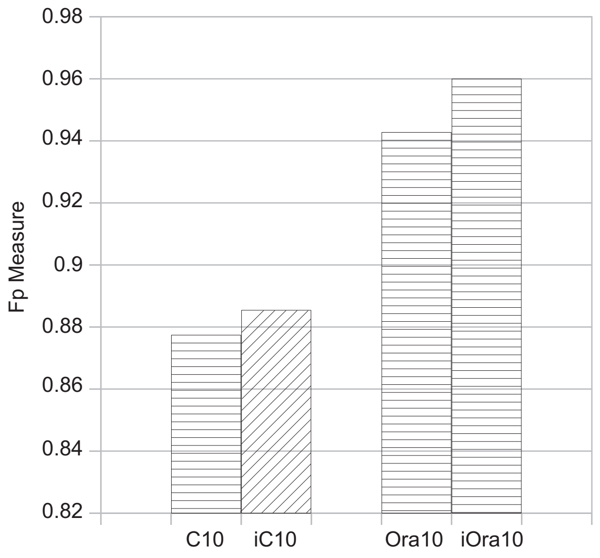

Fig. 8. F_p-measure of the combined classifier and the maximum plausible performance. (Basic vs. improved regions.)

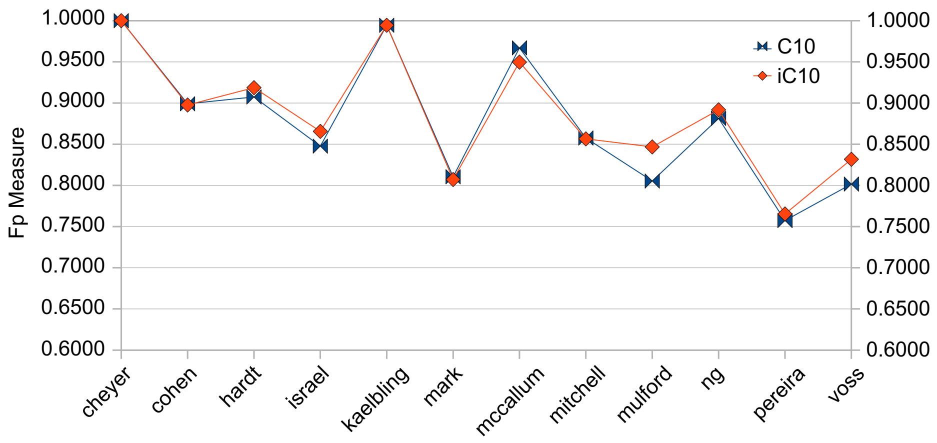

Fig. 7. Improvement of combined values. (Basic vs. improved regions.)

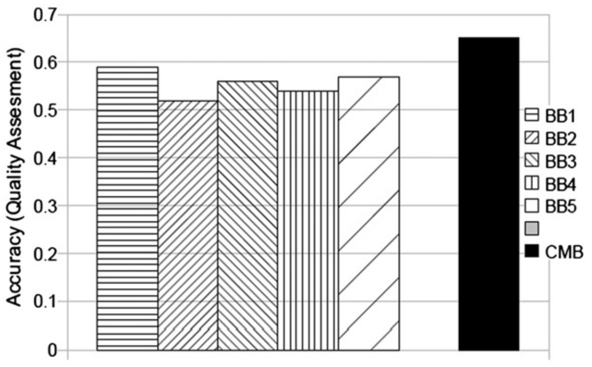

Fig. 9. Individual classifiers vs. quality-aware combination.

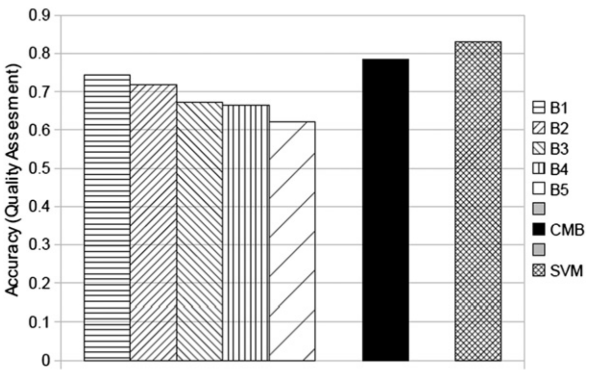

Fig. 10. Individual feature set classifiers vs. quality-aware combination.

similarity functions and the type of combination function. We show that starting with a rich set of similarity functions and even with a choice of a simple combination technique we can get results better the state-of-art-techniques. In some cases, more sophisticated combination techniques, like SVM, achieve further improvements.

6. Related work

Our work address the entity matching problem in Web context. In this section we relate our method with other entity matching and Twitter classification techniques. We also relate our work to the literature on combining multiple classifiers.

6.1. Entity matching

The entity resolution and related problems, such as for example duplicate detection has an extensive literature in the database community, a few important references include [1,2,6,26]. For a survey, see [8], for a comparison framework, see [27]. Many papers suggest (for example [2]) incremental clustering-based methods, while others propose pairwise comparison-based techniques. A recent paper [5] presents a pairwise comparison-based method, where the authors also consider confidence values during the resolution process. They propose to merge database records, which refer to the same entity, right away, as they are found to be equivalent by the algorithm. The algorithm also computes a new combined confidence value for the merged record. A more complete analysis of results can be found in [7], where the authors also study how to choose the sequence of the records to be processed, such that the running time of the algorithm remains low. In our work, we do not merge or recompute the similarity values.

Chauduri et al. [3] introduce a model for detecting fuzzy duplicates in databases. They extended their model also to a more general setting in [28]. Their paper is particularly important from methodological point of view, as they systematically derive their entity resolution algorithms from an axiomatic model. Unfortunately their model cannot be easily extended to the Web context because the properties of similarity functions for entities in Web documents do not show the same properties as in the case of fuzzy duplicates, so the basic assumptions of their model are not satisfied.

6.2. Entity matching on the web

Entity resolution in Web context was studied by Kalashnikov et al. [29]. They propose to create an entity resolution graph, using the feature-based similarities. The graph witnesses the uncertainty of the features by having multiple nodes, the so called “choice nodes” are corresponding to possible references to a given entity. The authors apply heuristic graph measures to measure the connectedness of entities. The underlying idea behind their heuristic is the “context attraction principle”: if two entities are related then it is likely that there are multiple chains in the entity resolution graph between their corresponding nodes. The authors further improved their techniques in [24]. In [24] and in many other approaches, such as for example in [6], the authors consider a more complex graph, which captures more complex relations, rather than the similarities between the entities as in our work. We limited ourselves to a simple representation and to focus on the issues in this simpler case, our framework could be later extended to a more complex setting. Their work and their use of context information in [29] is a similar technique to our quality-aware similarity assessment technique. We rely on different features, which are also easier to estimate.

Cudré-Mauroux et al. [30] take a different approach to entity resolution in the Web context. They propose a graphical model-based probabilistic framework to capture the relations among the entities. Their framework also includes trust assessments about the providers of the entity equivalence assertions. These trust assessment values are later adjusted as their probabilistic reasoning framework eliminates the detected inconsistencies. While this approach has many advantages, it is not fully applicable to our case, as the underlying factor graph model would have very large cliques, as subgraphs, which could easily lead to poor convergence of the probabilistic reasoning.

On the Semantic Web, person names might be annotated with a globally accepted ontology. This direct link between the ontology helps to disambiguate the person names. However, such globally accepted ontologies are not present in the emerging Semantic Web. Instead, ontologies are very often used as local schemas, thus one needs to relate the existing annotation to the ontology one would like to use. The Semantic Web community has developed a plethora of such techniques, see [31]. The OKKAM project suggests a different approach [32]. They propose a service, which provides globally unique identifiers on large scale for entities, for (Semantic) Web applications. Their approach relies on the existence of a large and clean (i.e. resolved) collection of entity profiles. Entity profiles collect relevant attributes of real world entities. Our techniques can contribute to create or extend such an entity profile collection.

This paper is an extended version of our own work on person name disambiguation [33]. In this paper we introduced the quality-aware similarity assessment technique and we applied these also to the Twitter message classification problem. Furthermore, we define our regions based on a richer feature set, by also considering whether the relevant information is missing. We also analyzed the limitations of the approach more carefully. In this work we also use the WePS-2 dataset for our experiments, that we did not use in [33].

6.3. *Combining classifiers*

Combining multiple classifiers is studied in the machine learning and also in data mining community [34]. The techniques can be broadly divided into two main categories:

1. Classifiers fusion, in which the final decision on a sample point is based on the fusion of decisions of individual classifiers, in some sense similar to achieving consensus. Examples include majority voting, weighted voting.
2. *Dynamic Classifier selection*: In this scenario, the decision of one of the classifier is chosen as the combined decision. Here, the classifier is chosen based on which classifier best represents the sample point.

Our methods use both combination techniques.

Different techniques have been proposed to identify regions for accuracy estimates. Woods et al. [35] discuss a method which divides the sample space into partitions either on predefined criteria or on the features. Each classifiers performance is estimated for each partition. This estimates would be used in choosing a best classifier for each partition. Liu et al. [36] propose a novel way of combining classifiers: which is by a technique called as clustering and selection. The input sample space is partitioned into several regions and clustering the correct and incorrect decisions separately. Each classifier performance is estimated for each region. On seeing a new sample, the region to which it belongs to is identified and the classifier with best performance for that region is chosen for the final decision. Strehl et al. [37] address the problem of combining multiple partitionings of a set of objects into a single consolidated clustering without accessing the features or algorithms that created the partitions. In our work we use conceptually similar techniques as the papers above, but the actual definitions are specific to the application.

Chen et al. [25] studied the combination of multiple classifiers, where the classifiers are applied for performing entity resolution. They also suggest that the performance of the classifiers depends on the context. Their method introduces techniques to exploit the context and find regions, where the classifier work better. Their method highly depends on their estimation of the total number of clusters (entities), which can be highly unreliable. Once they obtained the combination of the clustering methods, they also apply further techniques to improve their method, such as correlation clustering [17] and related heuristic approximation techniques. Their overall strategy is similar to ours, but their way of defining regions and combining similarity values is different.

Bilenko et al. [38] propose to use SVM classifiers for entity matching in databases. They also adapt the distance functions, which are string similarity functions, with the help of machine learning techniques. A multiple classifier approach was used by Zhao et al. [39] for entity identification in heterogenous database integration scenarios. They also consider various classifier combination techniques to improve the classification accuracy. Our work applies similar techniques, but in a more general context.

Bi et al. [40] propose a classifier combination technique, based on Dempster–Shafer theory of evidence and evidential reasoning. We did not consider this approach, as our classifiers are often not independent.

6.4. *Twitter classification*

The classification of tweets has already been addressed in the literature, in different contexts. Some of the relevant works include [41–44].

In [41], the authors take up the task of classifying the tweets from twitter into predefined set of generic categories such as News, Events, Opinions, Deals and Private Messages. They propose to use a small set of domain-specific features extracted from the tweets and the user's profile. The features of each category are learned from the training set. This task which can be seen as a supervised learning scenario is different from our current task which is a generic learning task.

The authors in [42] build a news processing system based on Twitter. From the twitter stream they build a system that identifies the messages corresponding to late breaking news. Some of the issues they deal with are separating the noise from valid tweets, forming tweet clusters of interest, and identifying the relevant locations associated with the tweets. All these tasks are done in an online manner. They build a Naive Bayes classifier for distinguishing relevant news tweets from irrelevant ones. They construct the classifier from a training set. They represent intermediate clusters as a feature vector, and they associate an incoming tweet with cluster if the distance metric to a cluster is less than a given threshold.

In [43,44], the authors make use of twitter for the task of sentiment analysis. They build a sentiment classifier, based on a tweet corpus. Their classifier is able to classify tweets as positive, negative, or neutral sentiments. The papers identify relevant features (presence of emoticons, n-grams), and train the classifier on an annotated training set. Their work is complementary to ours: the techniques proposed in our work could serve as an essential pre-processing step to these sentiment or opinion analysis, which identifies the relevant tweets for the sentiment analysis.

Our earlier work on Twitter classification includes [19,45]. While we used the same profiles and dataset in fact we realized different experiments and we used different classifiers. In this work we use our quality-aware similarity assessment technique and define regions to improve the quality of our accuracy estimations, which was not studied before. The focus of the paper [19] was the dynamic maintenance and improvement of company profiles that is not used in this work. According to [46], the classifier described in [45] outperformed all other competing classifiers in the WePS-3 evaluation campaign.

7. Conclusion and future work

We studied two variants of the general entity matching problem in the Web context, namely the person name disambiguation and the Twitter classification problem. Such entity matching tasks are essential for realizing the Semantic Web: in order to process the information on the Web automatically, one needs to connect the entities present in unstructured Web documents to descriptions of entities, or to entity collections. We designed a simple algorithmic framework for both problems.

We studied the design of similarity assessment techniques. Our proposed method estimates the quality of available similarity values, for particular regions of the input and not globally, as the assessment techniques themselves produce results of different quality. Also it takes specifically into account if some information is not available or missing, which is very common in the context of Web documents. We demonstrated the effectiveness of these methods in our framework: for both problems our techniques show promising results. Quality-aware similarity functions can be used in combination with other algorithmic frameworks as well. The systematic quality assessment and quality-aware combination technique results improved similarity values and improves the overall performance of these algorithms. Clearly, there is a balance between the definition of regions and computational efficiency. Our way of defining regions is simple and easy to realize, yet different from other techniques.

In our future work we would like to find other ways for defining regions for accuracy estimations. We also plan to address the effect of incomplete information available in the Web pages on the accuracy of the similarity functions even more directly, by considering entropy based metrics, similar to [47]. We also would like to extend our quality estimations to more dynamic settings, which is essential if the Twitter messages have to be classified on the fly, as they arrive in the Twitter stream.

References

[1] I. Fellegi, A. Sunter, A theory for record linkage, Journal of the American Statistical Association 64 (328) (1969) 1183–1210.
[2] M.A. Hernández, S.J. Stolfo, The merge/purge problem for large databases, ACM SIGMOD Record 24 (2) (1995) 127–138.
[3] S. Chaudhuri, V. Ganti, R. Motwani, Robust identification of fuzzy duplicates, in: Proceedings of the 21st International Conference on Data Engineering (ICDE), 2005, pp. 865–876.
[4] Z. Chen, D.V. Kalashnikov, S. Mehrotra, Adaptive graphical approach to entity resolution, in: Proceedings of the 7th ACM/IEEE-CS Joint Conference on Digital Libraries, 2007, pp. 204–213.
[5] D. Menestrina, O. Benjelloun, H. Garcia-Molina, Generic entity resolution with data confidences, in: First International VLDB Workshop on Clean Databases, 2006.
[6] X. Dong, A. Halevy, J. Madhavan, Reference reconciliation in complex information spaces, in: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, 2005, pp. 85–96.
[7] O. Benjelloun, H. Garcia-Molina, D. Menestrina, Q. Su, S.E. Whang, J. Widom, Swoosh: a generic approach to entity resolution, The VLDB Journal 18 (1) (2009) 255–276.
[8] P.G. Ipeirotis, V.S. Verykios, A.K. Elmagarmid, Duplicate record detection: a survey, IEEE Transactions on Knowledge and Data Engineering 19 (1) (2007) 1–16.
[9] Z. Miklós, N. Bonvin, P. Bouquet, M. Catasta, D. Cordioli, P. Fankhauser, J. Gaugaz, E. Ioannou, H. Koshutanski, A. Mana, C. Niederée, T. Palpanas, H. Stoermer, From Web data to entities and back, The 22nd International Conference on Advanced Information Systems Engineering CAiSE'10), Lecture Notes in Computer Science, vol. 6051, Springer, 2010, pp. 302–316.
[10] C.D. Manning, P. Raghavan, H. Schütze, Introduction to Information Retrieval, Cambridge University Press, 2008.
[11] J. Hu, L. Fang, Y. Cao, H.-J. Zeng, H. Li, Q. Yang, Z. Chen, Enhancing text clustering by leveraging Wikipedia semantics, in: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2008, pp. 179–186.
[12] alchemiAPI, 〈http://www.alchemyapi.com/〉.
[13] H. Cunningham, GATE, a general architecture for text engineering, Computers and the Humanities 36 (2002) 223–254.
[14] OpenCalais, 〈http://www.opencalais.com/documentation/calais-web-service-api〉.
[15] Semantic Hacker, 〈http://www.semantichacker.com/〉.
[16] lucene, 〈http://lucene.apache.org/〉.
[17] N. Bansal, A. Blum, S. Chawla, Correlation clustering, Machine Learning 56 (1–3) (2004) 89–113.
[18] W. Galuba, K. Aberer, D. Chakraborty, Z. Despotovic, W. Kellerer, Outtweeting the twitterers—predicting information cascades in microblogs, in: 3rd Workshop on Online Social Networks (WOSN'10), 2010.
[19] S.R. Yerva, Z. Miklós, K. Aberer, What have fruits to do with technology? The case of orange, in: International Conference on Web Intelligence, Mining and Semantics (WIMS'11), ACM, 2011, p. 48.
[20] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-based Learning Methods, Cambridge University Press, 2000.
[21] D. Metzler, S. Dumais, C. Meek, Similarity measures for short segments of text, Advances in Information Retrieval, Lecture Notes in Computer Science, vol. 4425, 2007, pp. 16–27.
[22] R. Bekkerman, A. McCallum, Disambiguating Web appearances of people in a social network, in: Proceedings of the 14th International Conference on World Wide Web, 2005, pp. 463–470.
[23] Second Web People Search Evaluation Workshop, WePS-2-dataset, 〈http://nlp.uned.es/weps/weps-2-data/〉, 2009.
[24] D.V. Kalashnikov, Z. Chen, S. Mehrotra, R. Nuray-Turan, Web people search via connection analysis, IEEE Transactions on Knowledge and Data Engineering 20 (11) (2008) 1550–1565.
[25] Z. Chen, D.V. Kalashnikov, S. Mehrotra, Exploiting context analysis for combining multiple entity resolution systems, in: Proceedings of the 35th SIGMOD International Conference on Management of Data, 2009, pp. 207–218.
[26] V.S. Verykios, G.V. Moustakides, M.G. Elfeky, A Bayesian decision model for cost optimal record matching, The VLDB Journal 12 (1) (2003) 28–40.
[27] H. Köpcke, E. Rahm, Frameworks for entity matching: a comparison, Data and Knowledge Engineering 69 (2) (2010) 197–210.
[28] S. Chaudhuri, A.D. Sarma, V. Ganti, R. Kaushik, Leveraging aggregate constraints for deduplication, in: Proceedings of the 2007 ACM

SIGMOD International Conference on Management of Data, 2007, pp. 437–448.
[29] D.V. Kalashnikov, S. Mehrotra, Domain-independent data cleaning via analysis of entity-relationship graph, ACM Transactions on Database Systems 31 (2) (2006).
[30] P. Cudré-Mauroux, P. Haghani, M. Jost, K. Aberer, H. de Meer, idMesh: graph-based disambiguation of linked data, in: Proceedings of the 18th International World Wide Web Conference (WWW'09), 2009.
[31] J. Euzenat, P. Shvaiko, Ontology Matching, Springer, 2007.
[32] P. Bouquet, T. Palpanas, H. Stoermer, M. Vignolo, A conceptual model for a Web-scale entity name system, The Semantic Web, Fourth Asian Conference, ASWC 2009, Lecture Notes in Computer Science, vol. 5926, Springer, 2009, pp. 46–60.
[33] S.R. Yerva, Z. Miklós, K. Aberer, Towards better entity resolution techniques for Web document collections, in: 1st International Workshop on Data Engineering meets the Semantic Web (DESWeb'2010), 2010.
[34] G. Seni, J. Elder, Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions, Morgan and Claypool Publishers, 2010.
[35] K. Woods, W.P. Kegelmeyer Jr., K. Bowyer, Combination of multiple classifiers using local accuracy estimates, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (4) (1997) 405–410, doi:10.1109/34.588027.
[36] R. Liu, B. Yuan, Multiple classifiers combination by clustering and selection, Information Fusion 2 (3) (2001) 163–168.
[37] A. Strehl, J. Ghosh, C. Cardie, Cluster ensembles—a knowledge reuse framework for combining multiple partitions, Journal of Machine Learning Research 3 (2002) 583–617.
[38] M. Bilenko, R.J. Mooney, Adaptive duplicate detection using learnable string similarity measures, in: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2003, pp. 39–48.
[39] H. Zhao, S. Ram, Entity identification for heterogeneous database integration: a multiple classifier system approach and empirical evaluation, Information Systems 30 (2) (2005) 119–132.
[40] Y. Bi, J. Guan, D. Bell, The combination of multiple classifiers using an evidental reasoning approach, Artificial Intelligence 172 (2008) 1731–1751.
[41] B. Sriram, D. Fuhry, E. Demir, H. Ferhatosmanoglu, M. Demirbas, Short text classification in twitter to improve information filtering, in: Proceedings of the ACM SIGIR 2010 Posters and Demos, ACM, 2010.
[42] J. Sankaranarayanan, H. Samet, B.E. Teitler, M.D. Lieberman, J. Sperling, Twitterstand: news in tweets, in: GIS'09: Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM, New York, NY, USA, 2009, pp. 42–51, doi:10.1145/1653771.1653781.
[43] A. Pak, P. Paroubek, Twitter as a corpus for sentiment analysis and opinion mining, in: Proceedings of the Seventh Conference on International Language Resources and Evaluation (LREC'10), European Language Resources Association (ELRA), Valletta, Malta, 2010.
[44] B. Jansen, M. Zhang, K. Sobel, A. Chowdury, Twitter power: tweets as electronic word of mouth, Journal of the American Society for Information Science and Technology 60 (2009) 2169–2188.
[45] S.R. Yerva, Z. Miklós, K. Aberer, It was easy, when apples and blackberries were only fruits, in: Third WePS Evaluation Workshop: Searching Information about Entities in the Web, CLEF (Notebook Papers/LABs/Workshops), 2010.
[46] E. Amigó, J. Artiles, J. Gonzalo, D. Spina, B. Liu, A. Corujo, Weps3 evaluation campaign: overview of the on-line reputation management task, in: M. Braschler, D. Harman, E. Pianta (Eds.), CLEF (Notebook Papers/LABs/Workshops), 2010.
[47] P. Cudré-Mauroux, A. Budura, M. Hauswirth, K. Aberer, PicShark: mitigating metadata scarcity through large-scale P2P collaboration, The VLDB Journal—The International Journal on Very Large Data Bases 17 (6) (2008) 1371–1384.