Toward an adaptive String Similarity Measure for Matching Product Offers

Andreas Thor

Dept. of Computer Science, University of Leipzig [*]
thor@informatik.uni-leipzig.de

Abstract: Product matching aims at identifying different product offers referring to the same real-world product. Product offers are provided by different merchants and describe products using textual attributes such as offer title and description. String similarity measures therefore play an important role for matching corresponding product offers. In this paper, we propose an adaptive string similarity measure that automatically adjusts the relevance of terms for the product matching. This adaptation is done step-by-step during the match process and does not require training data. We demonstrate that this approach improves the match quality in comparison to the generic TFIDF string similarity measure.

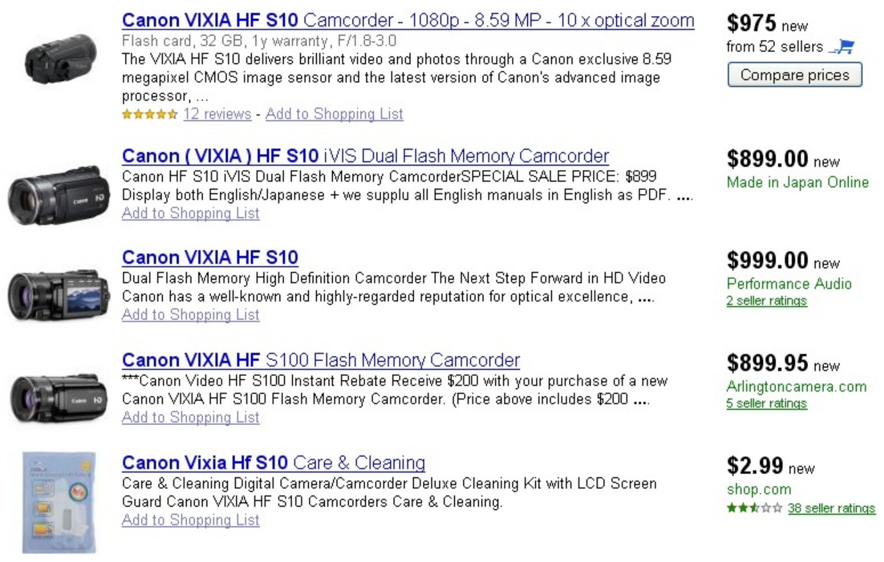
1 Introduction

Product matching deals with the identification of different product offers referring to the same real-world product. Product offers are typically represented as entities (e.g., database records) with several domain-specific attributes, e.g., title, description, price, and manufacturer. Product matching is therefore a special application of entity matching (also known as deduplication, entity resolution, or reference reconciliation; see [EIV07, KR10] for recent surveys) that is a fundamental problem for data management and data integration, in particular. Product matching is obviously a crucial aspect for e-commerce portals that combine offers from several merchants to allow users to find the best price for a certain product or to efficiently find suitable products from a variety of merchants.

We have analyzed a large real-world dataset that was provided by such an e-commerce portal.[1] It comprises several different product families (e.g., cars, electronics) with a total of more than 7 million offers that refer to an estimated 5 million products. The dataset reveals that only 10% of all offers are annotated with the standardized product identifiers UPC (universal product code) and MPN (manufacturer part number). Therefore price comparison websites have to carry out a product matching based on product attributes such as offer title, description, product category, or price. The enormous number of product offers requires an (at least semi-) automatic approach to efficiently match product offers.

[*]Currently on leave at University of Maryland Institute for Advanced Computer Studies (UMIACS)

[1]Because of a non-disclosure agreement we are not able to provide any details on the e-commerce portal.


Canon VIXIA HF S10 Camcorder - 1080p - 8.59 MP - 10 x optical zoom
Flash card, 32 GB, 1y warranty, F/1.8-3.0
The VIXIA HF S10 delivers brilliant video and photos through a Canon exclusive 8.59 megapixel CMOS image sensor and the latest version of Canon's advanced image processor, ...
12 reviews - Add to Shopping List
$975 new
from 52 sellers
Compare prices

Canon (VIXIA) HF S10 iVIS Dual Flash Memory Camcorder
Canon HF S10 iVIS Dual Flash Memory CamcorderSPECIAL SALE PRICE: $899 Display both English/Japanese + we supplu all English manuals in English as PDF. ...
Add to Shopping List
$899.00 new
Made in Japan Online

Canon VIXIA HF S10
Dual Flash Memory High Definition Camcorder The Next Step Forward in HD Video Canon has a well-known and highly-regarded reputation for optical excellence, ...
Add to Shopping List
$999.00 new
Performance Audio
2 seller ratings

Canon VIXIA HF S100 Flash Memory Camcorder
***Canon Video HF S100 Instant Rebate Receive $200 with your purchase of a new Canon VIXIA HF S100 Flash Memory Camcorder. (Price above includes $200 ...
Add to Shopping List
$899.95 new
Arlingtoncamera.com
5 seller ratings

Canon Vixia Hf S10 Care & Cleaning
Care & Cleaning Digital Camera/Camcorder Deluxe Cleaning Kit with LCD Screen Guard Canon VIXIA HF S10 Camcorders Care & Cleaning.
Add to Shopping List
$2.99 new
shop.com
38 seller ratings

Figure 1: Duplicate web entities related to the Canon VIXIA HF camcorder series in Google Product Search

A potentially high update rate on product offers (including offers for new products) may additionally limit the feasibility of a manual matching approach.

Numerous approaches for automatic entity matching have been proposed in the literature but only a few of them have been evaluated using real world product data.[2] One reason for this fact is that product offers are particularly challenging to match as they are often highly heterogeneous and of different data quality. Figure 1 illustrates some of the problems for the popular entity search engine Google Product Search and duplicate entries in its search result for a specific camcorder. The entries refer to different merchants that use heterogeneous names, descriptions and other attributes for the same product and may also contain misspellings and other errors. For example, the product names for the considered product Canon Vixia HF S10 contain additional information that may complicate entity matching, e.g., to find out that the first three entries refer to the same product. On the other hand, this information can help to recognize that the fourth entry is a similar but different product and the last entry does not represent the camcorder of interest, but only accessories.

In this paper we report on preliminary results toward an efficient and effective product matching approach. We first show that generic string similarity measures are insufficient for product matching. For example, the product titles of the second and fourth entry in Figure 1 have a very high string similarity but they refer to different products. In contrast, the first two offers for the same camcorder have more diverse titles. We then introduce the concept of an adaptive string similarity measure that is tailored to the specific charac-

[2]See http://dbs.uni-leipzig.de/fever for an annotated list of research publications on entity matching.

teristics of product offers. The key idea of our similarity measure is that it automatically adjusts term weights based on intermediate matching results. In the example of Figure 1, the term S10 should be given a high weight for matching the first three entries and at the same time separating them from the fourth entry. This automatic adaptation is done without any training data and it is motivated from the observation that term relevance varies across different subsets of product offers. We finally evaluate our approach and thereby demonstrate that it improves upon the standard TFIDF similarity measure.

2 Matching product offers using string similarity measures

We consider product matching as a partitioning problem within a single data source. Given a set of product offers $O = \{o_1, o_2, ..., o_n\}$, a match result is a partitioning of O, i.e., a set of non-overlapping partitions $P = \{p_1, p_2, ..., p_k\}$ with $p_i \cap p_j = \emptyset$ for any $1 \leq i < j \leq k$ and $p_1 \cup p_2 \cup ... \cup p_k = O$. The goal is that all offers o_i of a partition p_j refer to the same real world product ($precision = 100\%$) and all other offers refer to a different product ($recall = 100\%$).

A reasonable match/partitioning strategy would make use of several attributes. Product categories, manufacturer names, and price information can be used for blocking [BCC03], i.e., to calculate a coarse-grained partitioning of product offers to limit the search space and thus help increase matching efficiency. Afterwards, offer title and description should be analyzed to identify corresponding products within partitions. In the remainder of this paper we focus on the second step and investigate how string similarity measures can be applied to match product offers based on their title.

Similarity functions are a common tool to identify entities sharing similar attribute values (e.g., product offer title). For a similarity function sim one can distinguish between matching and non-matching entity pairs by introducing a threshold t. Two entities o_x and o_y are considered to match if $sim\,(o_x, o_y) \geq t$. The partitioning can then be determined based on the computed similarity values. Given a similarity function sim and a threshold t, $\{o_x, o_y\}$ is a subset of a partition if and only if the transitive closure of the binary relation $S = \{(o_i, o_j)\,|sim\,(o_i, o_j) \geq t\}$ contains (o_x, o_y). The quality of such a matching approach obviously depends on the similarity function and on an appropriate threshold.

Generic string similarity measures such as TFIDF have been successfully applied for entity matching in several domains [CRF03]. Unfortunately, they fall short for the scenario of product offers. Figure 2 shows the distribution of the TFIDF similarity for all correspondences of the perfect match result (determined by UPC/MPN codes). For a given similarity threshold t Figure 2 plots the percentage of correspondences that have a similarity smaller or equal to t. For example, approx. 25% of all correspondences of the product category Video Games have a similarity below or equal to 0.5. For comparison reasons, Figure 2 also shows the string similarity distribution for matching research publications based on the publication title of a dataset (comprising data from DBLP and the ACM Digital Library) we have used in previous work on entity matching [KTR10, TR07].

The main observation from Figure 2 is that the TFIDF string similarity measure does not

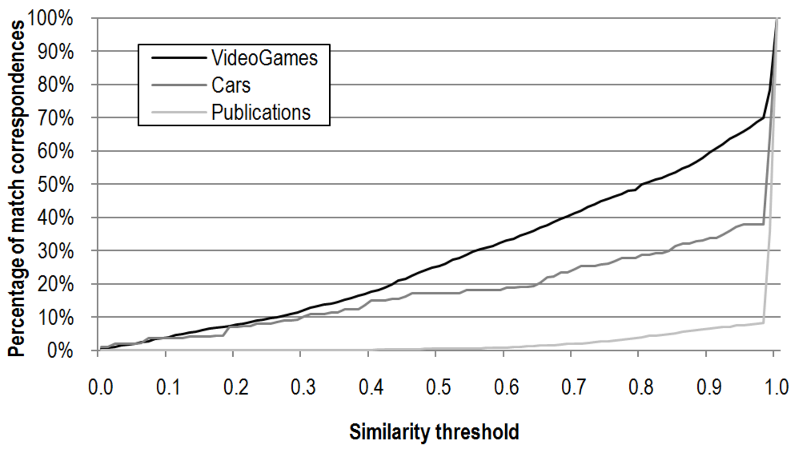


Figure 2: Cumulative distribution of TFIDF similarity of product offer titles for matching offer correspondences (determined by UPC/MPN)

discriminate well between matching and non-matching product offers. This is especially valid for the category Video Games where the similarity values are close to an uniform distribution. This makes it nearly impossible to find a reasonable threshold for the match decision. For example, given a minimal recall of 90% the TFIDF threshold must not be greater than 0.3 for products whereas the threshold can be set to 0.98 for publications. The low thresholds in turn lead to very a low precision. We have repeated the analysis of Figure 2 with other similarity measures (e.g., QGram, Levenshtein, amongst others) and have achieved similar results.

The overall match result can, of course, be improved by combining several similarity measures and using multiple attributes including the aforementioned product description, category, and price. Background knowledge such as reference lists of manufacturers, colors, or synonyms may also increase match effectiveness. Moreover, with the help of training data and machine learning techniques optimal match parameters for combining different similarity values can be calculated. In this work we focus on an orthogonal problem and try to develop a more sophisticated similarity measure for product offers because similarity measures are the foundation for any reasonable match strategy.

3 Adaptive string similarity measures for product offers

The string similarity measure applies the vector space model that is commonly used for representing textual information such as the title or description of product offers. Two strings s_1 and s_2 are represented as term vectors, i.e., $s_1 = [w_{1,1}, w_{1,2}, ..., w_{1,n}]$ and $s_2 = [w_{2,1}, w_{2,2}, ..., w_{2,n}]$. Each dimension corresponds to a term t, i.e., the non-negative weight $w_{i,k}$ reflects the weight of term t_k. If t_k doesn't occur in the string s_i, the corresponding weight $w_{i,k}$ is set to zero. The similarity of s_1 and s_2 is then defined as the cosine similarity of the two vectors.

The most prominent weight computation is TFIDF, i.e., $w_{i,k} = tf_{i,k} \times log\left(1/df_k\right)$. Here $tf_{i,k}$ denotes the term frequency (i.e., number of occurrences of term t_k in s_i divided by the sum of all term occurrences in s_i) and df_k denotes the document frequency (i.e., the number of strings where t_k occurs divided by the total number of strings).

The development of a product-specific string measure is driven from two observations. First, the product offer title usually contains several information such as the product name, product type, manufacturer, a manufacturer-specific product code (e.g., HF S10 for Canon camcorder; see Figure 1), and information about size, color, weight, or other product specific attributes. Second, products are very heterogeneous and different terms should be used as "key terms" for matching. For example, for camcorders a manufacturer-specific product code should be used whereas for car accessories the combination of product type and size might be the most relevant information. Even products of the same product family (e.g., electronics) may have different key aspects (e.g., size of TV sets vs. storage capacity for MP3 players).

We are therefore looking for terms that are relevant in a subset of product offers but not in the complementary set. The idea is that such terms receive a higher weight than other terms for the similarity computation. Since TFIDF calculates term weights mainly based on their global frequency it cannot capture this type of information. We propose a measure that makes use of an existing partitioning, i.e., an intermediate match result. In other words, the same term t_k may have therefore different weights in different partitions. Such an adaptive string similarity measure can be applied consecutively because the computation of a new partitioning may lead to different term weights and, thus, may change the partitioning again. In particular, the application of the adaptive string measure may split an existing partition into two or more partitions. The intention is that this iterative segmentation increases the precision while at the same time preserving the recall.

To this end we introduce the Term Partition Relevance (TPR) as follows: $TPR_k\left(p_m\right) = N\left(t_k, p_m\right)/N\left(t_k\right)$. Here $N\left(t_k\right)$ denotes the number of strings that contain the term t_k; $N\left(t_k, p_m\right)$ is the number of strings in the partition p_m that contain t_k. The term partition relevance falls in the $[0, 1]$ range and a high value indicates that a term t_k occurs in the partition p_m more frequently than in other partitions.

Often similar products are described with terms of the same type (product codes, storage capacity, ...). Obviously such terms are very important for distinguishing between similar but different products. Hence we also take into account the partition relevance of terms of the same type. For simplicity we assume that two terms t_1 and t_2 are of the same type if they have the same length n and the i's character (for all $1 \leq i \leq n$) of both t_1 and t_2 are either both letters or both numbers or both special characters (i.e., neither letters nor numbers). For example, the terms XSU002B and XSU321A are of the same type but XSU002B and TV-IP7 are not. We leave the development of a more sophisticated definition of term types for future work.

Based on the term type and the TPR we introduce the Term Type Partition Relevance (TTPR) as $TTPR_k\left(p_m\right) = N\left(\hat{t}_k, p_m\right)/N\left(\hat{t}_k\right)$. This definition is accordant to TPR but uses the term set $\hat{t}_k$ instead of t_k. The set $\hat{t}_k$ is the set of all terms that share the same type with t_k.

Our adaptive similarity measure then combines TFIDF, TPR, and TTPR as follows:

$$w_{i,k}\left(p_m\right) = tf_{i,k} \times log\left(\alpha \times 1/df_k + \beta \times TPR_{i,k}\left(p_m\right) + \gamma \times TTPR_{i,k}\left(p_m\right)\right)$$

with $\alpha, \beta, \gamma \in [0, 1]$ and $\alpha + \beta + \gamma = 1$. The weighted sum allows for a flexible control of the impact of both TPR and TTPR. We will investigate parametrization strategies for α, β, γ in future work. In the following evaluation we use a fixed parametrization scheme.

4 Evaluation

We briefly evaluate our approach for two selected product families of the real-world dataset mentioned in Section 1. The product family Cars contains 2,645 product offers that refer to 2,444 real-world products; there are 4,477 Video Games offers referring to 2,353 distinct products.

In a pre-processing step the product offer titles are tokenized using a standard tokenizer as provided by the Lucene search engine framework [HG04]. Thus, multiple string variants such as VIXIA and (VIXIA) (see Figure 1) are mapped to the same term. Unfortunately, white spaces may lead to heterogeneous term representations for certain pieces of information, e.g., product codes (S10 vs. S 10). The development of a tailored tokenizer for product titles is therefore subject to future work.

For the evaluation we use the common metrics recall and precision which are defined as follows: $Recall = |Corr_P \cap Corr_M| \,/\, |Corr_P|$ and $Precision = |Corr_P \cap Corr_M| \,/\, |Corr_M|$. Here $Corr_P$ denotes the perfect match result determined by the UPC/MPN code, i.e., the set of all pair-wise correspondences between offers referring to the same product. $Corr_M$ is the set of all correspondences determined by the match approach, i.e., the set of all correspondences (o_1, o_2) where o_1 and o_2 are in the same partition.

The evaluation results have been achieved using the following procedure: We start with the computation of TFIDF $(\alpha = 1, \beta = 0, \gamma = 0)$ on the offer title with a fixed threshold t. We then repeatedly apply our similarity measure (using the same threshold) to the achieved partitioning and gradually decrease the TFIDF impact and increase the TPR impact in steps of 0.05 until we reach the configuration $(\alpha = 0.05, \beta = 0.95, \gamma = 0)$. Afterwards we decrease the influence of TPR and increase the impact of TTPR simultaneously in steps of 0.05. Finally we end up with $(\alpha = 0.05, \beta = 0, \gamma = 0.95)$ and evaluate the resulting partitioning (named "Adaptive" in Figure 3).

The repeated application of the similarity measure increases the number of partitions step-by-step. The conducted parametrization scheme reflects the intuition that TFIDF should be used for a first match result because TPR and TTPR are only beneficial if there is a significant number of partitions. This is especially true for TTPR that considers sets of terms of the same type. The TTPR impact therefore increases at the end of the procedure

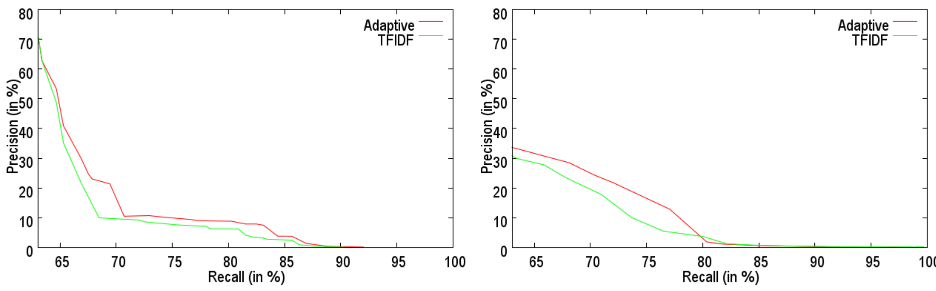


Figure 3: Evaluation result for product family Cars (left) and Video Games (right)

at the expense of the TCR impact. We conduct the described procedure several times while varying the similarity threshold t from 0.2 to 1 in steps of 0.05.

In this evaluation we are interested how the adaptive string similarity measure can improve an existing match result determined by the TFIDF similarity. In particular, Figure 3 plots the precision against the recall for TFIDF (lower curve) and the adaptive similarity measure (upper curve). The main observation is that the adaptive similarity measure increases the precision while at the same time preserving the recall. This proofs our initial assumption that an adaptive term weight approach based on an intermediate match result may increase the match quality. It is therefore reasonable to further investigate on adaptive term weighting approaches.

The comparison of the two product families Cars and Video Games also shows the influence of the string similarity measure on the match quality. As illustrated in Figure 1, the TFIDF similarity measure is more suitable for Cars than for Video Games. As a result, both TFIDF and the TFIDF-based adaptive measure achieve a better match quality for the product family Cars than for Video Games. This observation also justifies the assumption that a string similarity is a key element of any effective product match approach.

Figure 3 also demonstrates that the sole use of a string similarity for product offer titles does not solve the product matching problem satisfactorily. The maximal achieved F-Measures are 67% (Cars) and 50% (Video Games), respectively. The adaptive similarity measure could not substantially increase these maxima. Nevertheless, when combined with other match approaches (e.g., blocking techniques, similarity of other attributes) an improved similarity measure may help improve the overall match result.

5 Related work

Entity matching has received a lot of attention in the research community (see [EIV07, KR10] for recent surveys) and numerous approaches have been proposed but only a few have been evaluated using real-world product data. [BBS05] proposes an adaptive approach that learns a composite similarity function for product offers based on a training data. The presented results with data from Froogle show a F-Measure below 60% and are in line with our experience that product matching is exceptionally challenging. Our work focuses on a basic similarity function and can therefore complement [BBS05]. The work presented in [BGMG$^+$07] also deals with matching product offers (from Yahoo) but focuses on scalability aspects. Efficiency is very important due to the large number of product offers and complex similarity functions. To this end, the authors present a family of algorithms (D-Swoosh) that allows for distributed entity matching across multiple processors. The presented evaluation therefore only focuses on performance measures but not on the match quality.

Adjusting parameters such as thresholds or weights for improving entity matching has been intensively studied for learning-based techniques using training data [KTR10]. Based on generic similarity functions (see [CRF03] for a comparison) there is also work on learnable string similarity measures (e.g., [BMC$^+$03] introduces an adaptive version of edit distance). In contrast to our work, all these approaches require training data. To the best of our knowledge, our approach is the first that automatically adapts term weights for string similarity computation based on intermediate match results in the absence of any training data.

Product matching is also related to document classification and document clustering because relevant information is stored in textual attributes, e.g., product title and product description. Beside many natural language processing techniques (e.g., part-of-speech tagging [Sch94]) weighting schemes have also been applied for classification and clustering of documents. For example, [FYL02] introduces a document classification approach based on discriminative category matching. Similar to our approach, extracted document features are assigned higher weights if they appear in fewer categories.

6 Summary and future work

We have presented our preliminary results toward an adaptive string similarity measure for matching product offers. We have proposed a flexible approach that makes use of intermediate match results and adjusts term weights accordingly. We could demonstrate that our adaptive measure can improve the TFIDF measure for product matching.

In future work we will further investigate on our similarity measure. We will use other attributes (e.g., product category) for generating a first partitioning of product offers and apply our term weighting scheme for both offer title and offer description. We will also apply the measure in product matching strategies and combine it with other matching techniques. Our implementations will be incorporated into our object matching framework FEVER [KTR09] which allows for comparative evaluations and automatic parameter tuning of different match approaches.

References

[BBS05] Mikhail Bilenko, Sugato Basu, and Mehran Sahami. Adaptive Product Normalization: Using Online Learning for Record Linkage in Comparison Shopping. In *Proc. of International Conference on Data Mining (ICDM)*, 2005.

[BCC03] Rohan Baxter, Peter Christen, and Tim Churches. A comparison of fast blocking methods for record linkage. In *Proc. of Workshop Data Cleaning, Record Linkage, and Object Consolidation*, 2003.

[BGMG$^+$07] Omar Benjelloun, Hector Garcia-Molina, Heng Gong, Hideki Kawai, Tait Eliott Larson, David Menestrina, and Sutthipong Thavisomboon. D-Swoosh: A Family of Algorithms for Generic, Distributed Entity Resolution. In *Proc. of International Conference on Distributed Computing Systems (ICDCS)*, 2007.

[BMC$^+$03] Mikhail Bilenko, Raymond J. Mooney, William W. Cohen, Pradeep D. Ravikumar, and Stephen E. Fienberg. Adaptive Name Matching in Information Integration. *IEEE Intelligent Systems*, 18(5), 2003.

[CRF03] William W. Cohen, Pradeep D. Ravikumar, and Stephen E. Fienberg. A Comparison of String Distance Metrics for Name-Matching Tasks. In *Proc. Workshop on Information Integration on the Web (IIWeb)*, 2003.

[EIV07] Ahmed K. Elmagarmid, Panagiotis G. Ipeirotis, and Vassilios S. Verykios. Duplicate Record Detection: A Survey. *IEEE Trans. Knowl. Data Eng.*, 19(1), 2007.

[FYL02] Gabriel Pui Cheong Fung, Jeffrey Xu Yu, and Hongjun Lu. Discriminative Category Matching: Efficient Text Classification for Huge Document Collections. In *Proc. of the International Conference on Data Mining (ICDM)*, pages 187–194, 2002.

[HG04] Erik Hatcher and Otis Gospodnetic. *Lucene in Action (In Action series)*. Manning Publications Co., Greenwich, CT, USA, 2004.

[KR10] Hanna Köpcke and Erhard Rahm. Frameworks for entity matching: A comparison. *Data Knowl. Eng.*, 69(2), 2010.

[KTR09] Hanna Köpcke, Andreas Thor, and Erhard Rahm. Comparative evaluation of entity resolution approaches with FEVER. *PVLDB*, 2(2), 2009.

[KTR10] Hanna Köpcke, Andreas Thor, and Erhard Rahm. Evaluation of learning-based approaches for matching web data entities. *IEEE Internet Computing*, 99, 2010.

[Sch94] Helmut Schmid. Probabilistic part-of-speech tagging using decision trees. In *Proc. of International Conference on New Methods in Language Processing*, volume 12. Citeseer, 1994.

[TR07] Andreas Thor and Erhard Rahm. MOMA - A Mapping-based Object Matching System. In *Proc. of Conference on Innovative Data Systems Research (CIDR)*, 2007.