Classifying Search Engine Queries Using the Web as Background Knowledge

David Vogel[1,2], Steffen Bickel[3], Peter Haider[3], Rolf Schimpfky[3], Peter Siemen[3], Steve Bridges[2], Tobias Scheffer[3]
[1]A.I. Insight, Inc., Orlando, Florida
[2]MEDai, Inc., Orlando, Florida
[3]Humboldt-Universität zu Berlin, Department of Computer Science, Berlin, Germany
dvogel1@cfl.rr.com, flsrb@bellsouth.net,
{bickel, haider, schimpfk, siemen, scheffer}@informatik.hu-berlin.de

ABSTRACT

The performance of search engines crucially depends on their ability to capture the meaning of a query most likely intended by the user. We study the problem of mapping a search engine query to those nodes of a given subject taxonomy that characterize its most likely meanings. We describe the architecture of a classification system that uses a web directory to identify the subject context that the query terms are frequently used in. Based on its performance on the classification of 800,000 example queries recorded from MSN search, the system received the Runner-Up Award for Query Categorization Performance of the KDD Cup 2005.

1. INTRODUCTION

Most web search queries contain only two or three terms and therefore provide very limited information about the user's information need to the search engine. Utilizing this information is a key factor to constructing effective web search engines. One way of approaching this problem computationally is to approximate the intended meaning of a query by a node, or a set of nodes, in a given subject taxonomy. For instance, a query "the raven" can indicate that a user searches for information on *entertainment/movies* or on *zoology.* Thus, the intuitive problem of capturing the intended meaning of a query is reduced to the computational problem of mapping the query string to a set of nodes in a given – fixed, but arbitrary – subject taxonomy.

The KDD Cup 2005[1] casts this problem setting into a competitive benchmarking framework that allows to evaluate and compare methods and systems which solve this problem under a specified experimental setting. The KDD Cup data set comprises of a two-level taxonomy with 67 second level nodes and 800,000 MSN web search queries, 111 of which are labeled with up to five nodes of the taxonomy.

The rest of this paper is organized as follows. We discuss related work in Section 2, detail the problem setting in Section 3, and describe the classification architecture in Section 4. Section 5 discusses the evaluation, Section 6 concludes.

[1]http://kdd05.lac.uic.edu/kddcup.html

2. RELATED WORK

Capturing the intended meaning of a search query with the help of word sense disambiguation has been heavily investigated. Early studies (see [12] for a complete overview) have found no or only little improvement in precision. Word sense disambiguation is often resolved implicitly when queries are sufficiently long; in these rare cases, the additional words provide sufficient context to resolve the intended meaning. Web search queries most often contain only two or three words and the work of Schütze and Pederson [13], Gonzalo et al. [9] and Stokoe et al. [14] demonstrates that word sense disambiguation can improve information retrieval.

A different line of research focuses on interactive query expansion or query reformulation; Bruza and Dennis [4; 6] develop a hyper-index to provide the user with query reformulation recommendations. Anick and Tipirneni [3] develop an interactive query refinement system that is based on the analysis of semantically related lexical compounds. Allan and Raghavan [2] refine query semantics with the use of part-of-speech patterns and formulate clarification questions for the user. Glance [8] studies a collaborative recommendation system that records the queries of all users and supports new searches by recommending related query strings of other users.

Clustering the set of search results is another way to deal with query ambiguity: ideally, ambiguous queries result in multiple clusters that reflect the possible interpretations of the query. After a topic insensitive web search the results are clustered based on content [5; 16; 11] and link structure [15].

Our solution is based on a taxonomic mapping between a web directory and the subject taxonomy. Integrating objects from one taxonomy into another has been studied by different authors. Agrawal and Srikant [1] develop an enhanced naive Bayes algorithm that is based on the intuition that if two documents share their category in one taxonomy, they are likely also share their category in another taxonomy. Zhang and Lee [17] use a similar approach in combination with Support Vector Machines; in addition, they develop a co-bootstrapping method.

Instead of exchanging objects between taxonomies Doan et al. [7] learn a direct mapping between taxonomy nodes using the joint distribution of concepts. All those approaches require a substantial number of training examples in the source as well as in the target taxonomy. In our case, the web directory provides an abundance of filed web pages whereas there are only 111 example queries and no example documents for the subject taxonomy.

3. PROBLEM SETTING

The problem that we address is driven by the intuition that a significant aspect of the semantics behind a search query can be captured by identifying likely nodes in a subject taxonomy. An instance of the query categorization problem is described by an arbitrary but fixed subject taxonomy. The nodes are named (e.g., *living*, *entertainment*, *sports* for first level, *living/pets and animals*, *sports/olympic games* for second level nodes and so on), and their semantics are primarily characterized by these names. In addition, example queries that are labeled with nodes may be available that exemplify search queries for documents within these categories.

For the KDD Cup 2005, a two-level taxonomy of 67 nodes is provided together with a total of 111 example queries; each of these 111 queries is manually tagged with up to five nodes in the taxonomy. An example excerpt of these data is displayed in Table 1. The example queries are not guaranteed to be drawn from the same distribution that governs the evaluation data.

From this input, a classifier has to be obtained that automatically labels new queries with one or several nodes of the taxonomy. The subject taxonomy can be assumed to be fixed (over a reasonable period of time) for any given problem instance. Therefore, it is reasonable to allow for some manual involvement over the process of generating a classifier. The classifier, on the other hand, has to work automatically and be efficient in order to process a large volume of queries posted to a search engine.

For the KDD Cup, a set of 800,000 hold-out queries are available. For a subset of these queries (it has been unknown which subset), manually annotated labels are available. A natural performance criterion to be maximized is the F-measure, the harmonic mean of classification precision and recall. In addition, a second sub-task of the Cup is to maximize precision, subject to the constraint that the F-measure be within the ten highest values for F-measure obtained by all submissions. This criterion is difficult to maximize because it requires a conjecture about the submissions of competitors.

4. CLASSIFICATION ARCHITECTURE

The classification system that we devise consists of a component that searches a given query in a web directory; we use the Google directory search interface that searches the open web directory Dmoz.org. This component generates an ordered list of categories of the web directory. In general, the categories of the web directory will be distinct from the subject taxonomy that defines the query categorization problem instance at hand. Therefore, a second component of the architecture maps directory categories to nodes of the subject taxonomy. Web directories tend to be large and manually mapping each of their nodes to a node of the given subject taxonomy is impractical; we therefore construct a tool that supports a semi-automatic mapping.

Given a query, the third component processes the sorted list of categories returned by the web directory search component, consults the mapping component to translate these into nodes of the subject taxonomy, and combines all information into probability scores for the nodes of the subject taxonomy. This component conjectures the final result of up to five nodes in the subject taxonomy that maximize either the F-measure, or the precision at a desired minimum F-measure level.

4.1 Web Directory Search

This component of our classification architecture searches a web directory – we use the Domz.org web directory – for the query. The component preprocesses the query by first removing search options (e.g., “filetype:”) from the query string, and then posts the query to the Google directory search which scans the Dmoz directory for occurrences of the query within the Dmoz categories.

The web directory searching component processes the first 100 search results and transforms them into an ordered list of those directory categories that the 100 retrieved documents are classified into. Note, however, that the Dmoz taxonomy (or any other web directory) is generally distinct from the given subject taxonomy, in our case the KDD Cup taxonomy.

The web directory searching component exploits two additional features of the Google directory search service. Google suggests alternative queries that in many cases correct spelling errors in the posted queries (“did you mean: . . . ”). The web directory searching component always accepts these suggestions which in many cases improves the results on misspelled search queries. Google directory search also provides a list of “related categories” along with the search results. These related categories are stored and influence the final weighting process.

4.2 Semi-Automatic Category Mapping

The taxonomy of the web directory generally differs from the subject taxonomy that defines the instance of the query classification problem. The category mapping component of our classification architecture therefore translates the directory taxonomy into the subject taxonomy.

The subject taxonomy is characterized by the descriptive names of its nodes and very few example queries—for many nodes, only a single exemplifying query is available in the KDD Cup data. The web directory taxonomy, on the other hand, is defined by the collection of web pages filed under each node in addition to the descriptive name of each node. Given the salient role played by the descriptive names in the subject taxonomy and given the substantial size of the Dmoz web directory that contains thousands of nodes, a semi-automatic mechanism is advised. Fully automatic learning of a taxonomy mapping is an elegant alternative path that can be taken when sufficiently many example queries for both taxonomies are available—in our case, a total of 111 training examples renders this approach little promising.

We manually assign directory categories to categories of the subject taxonomy by inspecting the web directory up to the second level, in some branches to the third or forth level where a finer granularity is needed. Bounding the depth for the manual inspection is necessary to keep the task manageable. Each directory node is assigned up to three categories of the subject taxonomy, resulting in an $n : m$ mapping between the taxonomies. Some examples of the resulting mapping table are shown in Table 2.

The web directory organizes country specific pages in the

Table 1: Examples of the training queries with first two categories.

Query	**Subject taxonomy node 1**	**Subject taxonomy node 2**	...
bank loans	living/finance & investment	information/companies & industries	...
bar exam result	information/law & politics	living/career & jobs	...
basset hound dogs	living/pets & animals	information/science & technology	...
beginner guitar	entertainment/music	information/education	...
behr paint color samples	living/furnishing & houseware	shopping/buying guides & researching	...
Beijing 2008	sports/Olympic Games	information/local & regional	...
bench grinders	living/tools & hardware	living/landscaping & gardening	...
bowling clip art pictures	entertainment/pictures & photos	sports/other	...

Table 2: Examples of the taxonomy mapping after applying the semi-automatic assignment.

Web directory taxonomy node	**Subject taxonomy node 1**	**Subject taxonomy node 2**
health	living/health & fitness	
health/animal	living/pets & animals	
health/beauty	living/fashion & apparel	living/health & fitness
health/medicine	living/health & fitness	information/science & technology
health/medicine/employment	living/career & jobs	
health/mental health/humor	entertainment/humor & fun	
health/nursing/education	living/health & fitness	information/education
health/search engines	information/references & libraries	living/health & fitness
health/weight loss/low fat cooking	living/food & cooking	

same structure as the main taxonomy. They can be reached from the main taxonomy under the *regional* branch. Because the regional information is not relevant for the categorization the path nodes that correspond to regional information get removed, so that the regional directory nodes are projected onto the non-regional parts of the main taxonomy. For example, the category *regional/Europe/Germany/health* is treated like *health*.

When a node of the web taxonomy is mapped to a node of the subject taxonomy then, by default, its entire subtree is mapped to the same node. This default behavior is overruled when nodes within that subtree are explicitly mapped to a distinct node of the subject taxonomy.

The manual mapping of the first two (partially up to four) levels of the taxonomies imposes the risk of missing relevant nodes hidden deeper in the hierarchy. On the other hand, constructing a mapping for thousands of nodes is cumbersome. We develop an automatic recommendation system that provides suggestions for correspondences of deeper categories. We define a characteristic query for each category of the subject taxonomy; e.g., *information/law & politics* is assigned the query "legal OR politics". The mapping recommendation system posts these queries to the web directory search engine; the categories of retrieved web pages are added to a candidate list of recommended mappings.

After manually inspecting these recommendations for the KDD Cup task and accepting some 200 of them, we obtain a mapping that entails 763 rules following the schema visualized in Table 2.

4.3 Generating a Final Conjecture

The web directory searching component returns for each query q an ordered list of up to 100 web directory categories $C'(q)$, we denote the elements of $C'(q)$ as c'_i. The mapping component maps a web directory category c'_i to a list of subject taxonomy categories $M(c'_i)$ whose elements we write as c_j. Conversely, let $M^{-1}(q, c_j)$ be the ordered subset of $C'(q)$ whose elements c'_i have c_j in their assigned subject taxonomy nodes; i.e., $M^{-1}(q, c_j) \subseteq C'(q)$ with $c'_i \in M^{-1}(q, c_j)$ if and only if $c_j \in M(c'_i)$. The ordering of the elements of $M^{-1}(q, c_j)$ is equal to their ordering in $C'(q)$.

Equation 1 defines the weight that associates the query q to subject taxonomy category c.

$$w(q, c_j) = f_1(c_j) f_2(q) \sum_{\substack{c'_k \in M^{-1}(q,c_j) \\ k=1\ldots|M^{-1}(q,c_j)|}} f_3(q, c'_k) \alpha_6^k \alpha_7^{d_1(c'_k)} \alpha_8^{d_2(c'_k)-d_1(c'_k)} \quad (1)$$

The terms of the weighting function are defined as follows. Many technical decisions were made according to preliminary experimental studies. The weighting parameters $\alpha_1, \ldots, \alpha_8$ are determined through an iterative procedure. Starting with intuitive values, we repeatedly pick one of those parameters and manually adjust it to achieve optimal accuracy on the training set while holding the other ones fixed.

- Regularization term $f_1(c_j) = \alpha_1$ if c_j is an "other" category (e.g., *sports/other*), and 1 otherwise. This factor compensates for the disproportionally large number of these categories in the manual mapping. In our setting, α_1 is empirically adjusted to 0.5.

- Prior adjustment term $f_2(i) = \alpha_2^{\log \#Results(q)}$, where $\#Results(q)$ is the total number of results returned by the web directory search engine for query q. A higher total number of results implies better quality of the top 100 result and so our intuition is that the category associations of queries with more search results should influence the result stronger; α_2 is set to 1.05.

- The $f_3(q, c'_k)$ term is defined as $f_3(q, c'_k) = \alpha_3^{r(q,c'_k)}$ if $r(q, c'_k) < 10$, and $\alpha_3^{10}\alpha_4^{r(q,c'_k)-10}$ otherwise, where $r(q, c'_k)$ is the position of c'_k within the result list of the web directory search engine. This term reflects the relevance of the results for the query, as the search engine sees it. The relevance of a result degrades exponentially with its rank within all results, for the first 10 results with decay factor α_3 (we choose $\alpha_3 = 0.9$), and for subsequent results with decay factor $\alpha_4 > \alpha_3$ ($\alpha_4 = 0.97$). In addition, $f_3(q, c'_k)$ allows to integrate the "related categories" that Google directory search retrieves in addition to the ranked list of matching pages. Web directory categories that have been obtained in such a way are weighted with a factor of $f_3(q, c'_k) = \alpha_5 = 1.1$.

- The factor α_6^k accounts for the redundancy in multiple web directory taxonomy categories that are mapped to the same subject taxonomy category. Any result for c_j beyond the first provides less new information than the preceding ones, so the exponential decay factor of α_6 is applied, which we set to 0.9.

- Two depth parameters account for the depth $d_1(c'_k)$ of the left hand side of the mapping rule that maps c'_k to its counterparts in the subject taxonomy, and the depth $d_2(c'_k)$ of c'_k itself in the web taxonomy. The term $\alpha_7^{d_1(c'_k)}$ reflects the higher confidence in mapping rules for more specialized branches of the hierarchy. For instance, the application of the rule with left-hand side *arts/performing arts/acting/actors and actresses* yields a higher confidence in the resulting taxonomic category than the application of the rule with left-hand side *arts*. Experiments with the training set indicate $\alpha_7 = 1.3$ as a good value.

  The difference $d_2(c'_k) - d_1(c'_k)$ measures the number of taxonomic levels that separate a web taxonomy node from its matching rule. For instance, if the category *health/nursing/employment* gets matched with the rule *health* $\rightarrow$ *living/health & fitness*, two excess levels are ignored which reduces the quality of the resulting category set. The decay factor is set to $\alpha_8 = 0.8$.

The final classification conjecture for a query q is based on three factors: firstly, the weights $w(q, c_j)$ for each category c_j. Secondly, the rank $r_w(q, c_j)$ of category c_j is the position at which c_j occurs when all categories are ordered according to $w(q, c_j)$. Thirdly, an additional margin criterion that measures the distance between the weights of c_j and the next likely category, $\delta(q, c_j) = w(q, c_j) - max_{c:w(q,c)\leq w(q,c_j)\wedge c\neq c_j}\{w(q, c)\}$.

We determine the category weights $w(q, c_j)$ for all training queries q and fit a logarithmic curve to the data to model the probability $P(c|w(q, c_j))$. Figure 1 shows the empirical data and the fitted curve; queries are grouped into equally sized batches. We estimate the discrete distribution $P(c_j|r_w(c_j))$ from the training data, Figure 2 shows the result.

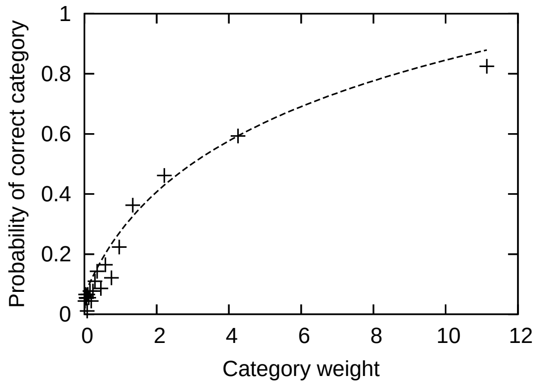

Figure 1: Relation between category weight $w(q, c_j)$ of a query-category pair and its probability of correctness $P(c|w(q, c_j))$ on the training data.

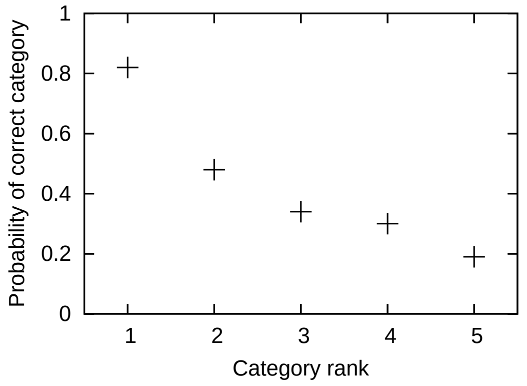

Figure 2: Relation between category rank $r_w(c_j)$ and its probability of correctness $P(c_j|r_w(c_j))$ on the training data.

A logistic regression model combines the weight, rank, and margin into a single probability $P(c_j|q)$. We fit the logistic regression model [10] as given in Equation 2 using the Mitch data analysis software package.

$$P(c_j|q) = \frac{1}{1 + e^{-(\beta_0+\beta_1 P(c|w(q,c_j))+\beta_2 P(c_j|r_w(c_j))+\beta_3\delta(q,c_j))}} \quad (2)$$

The final step is to determine the probability thresholds for the two tasks (F-measure and precision). The probability threshold θ for the best F-measure score is determined through a hill climbing procedure. The value θ is initialized to 0.5 and we determine the direction of search by computing the F-measure gradient $\nabla F(\theta) = \lim_{\epsilon\rightarrow 0} F(\theta+\epsilon) - F(\theta)$. This gradient is dependent on a binary random variable X that determines whether a query with probability $P(c_j|q) = \theta + \epsilon$ is correct. We calculate the expectation over X in Equation 3. The expected gradient of F can be split up in Equation 4 because for the case that $X = 1$ only TP increases and only the gradient with respect to TP is relevant

Table 3: Performance measures of our and winning solutions.

Task	Description	Evaluation data	F-measure	Precision
F-measure	our submission	training	0.51	0.52
F-measure	our submission	hold-out	0.41	0.45
F-measure	winner	hold-out	**0.44**	0.41
Precision	our guess 1 (submission)	training	0.29	0.89
Precision	our guess 1 (submission)	hold-out	0.21	0.75
Precision	our guess 2	hold-out	0.41	**0.45**
Precision	winner	hold-out	0.43	0.42

because $\nabla F(\theta) = \frac{\delta F}{\delta TP}(\theta)$ and for $X = 0$ only the one with respect to FP is relevant. TP and FP are the number of true positives and false positives respectively. We get Equation 5 by substituting $P(X = 1) = \theta+\epsilon$ and $P(X = 0) = 1-(\theta+\epsilon)$.

$$\nabla F(\theta) = \lim_{\epsilon\to 0} E[F(\theta+\epsilon) - F(\theta)] \quad (3)$$

$$= \lim_{\epsilon\to 0} \left[P(X=1)\frac{\delta F}{\delta TP}(\theta) + P(X=0)\frac{\delta F}{\delta FP}(\theta)\right] \quad (4)$$

$$= \lim_{\epsilon\to 0} \left[(\theta+\epsilon)\frac{\delta F}{\delta TP}(\theta) + (1-(\theta+\epsilon))\frac{\delta F}{\delta FP}(\theta)\right] \quad (5)$$

$$= \theta\frac{\delta F}{\delta TP}(\theta) + (1-\theta)\frac{\delta F}{\delta FP}(\theta) \quad (6)$$

With Equation 6 we determine the direction of search from each new threshold value θ and conduct a binary search. In each search step the search space is halved and the direction of search is determined through the F-gradient.

For the precision task we need to specify a minimum F-measure that we want to achieve. For finding the best precision we start with a threshold $\theta = 1$ and step down in sufficiently small steps until the minimum F-measure is reached.

5. EVALUATION

The trade-off between F-measure and precision, dependent on the probability threshold θ, is visualized in Figure 3. The hill climbing procedure described in Section 4.3 successfully finds the maximum F-measure at 0.51. Table 3 gives an overview of the different solutions of our team and the winning team. We can see that the F-measure performance of our submission on the held out data set that is used for the competition result is lower (0.41) than the estimation on the training data. In this case measuring the performance on the 111 training examples can only be used as a rough estimate of the actual performance. The winning solution for the F-measure task of the winning team reaches 0.44; our submission of 0.41 is rated second best and receives the Runner-Up Award.

For the precision task we have to make a guess at the F-measures of the competing teams as the goal is to achieve the maximum precision while remaining within the top ten F-measure submissions. We prepare two guesses, one with an F-measure of 0.29 and the other with an F-measure of 0.51 (same as the submission for the F-measure task) on the training data (see Table 3). Assuming that most competing teams would risk a high precision at a relatively low recall level, we decided to submit the first solution that achieves 75% precision on the hold-out data. Refuting our assumption, the ten teams whose solutions achieve the highest F-measure decide for more balanced solutions that outperform our F-measure without getting close to our precision. Our second guess of an F-measure of 0.41 and a precision of 45% would even have beaten the winning team's submission of an F-measure of 0.43 and 42% precision. So the winning team not only had an excellent solution to the query categorization problem, but also solved the "poker playing" problem of predicting their competitors' hands well.

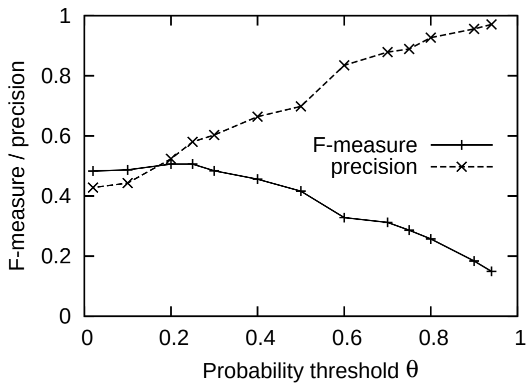

Figure 3: Relation between F-measure/precision and the probability threshold θ estimated on the training data.

6. CONCLUSIONS

Extracting semantic information from search queries is important for a search engine to understand a user's information need. Mapping a search query to a set of nodes of a subject taxonomy captures an aspect of that query's semantics.

We devised an architecture that allows to map queries to an arbitrary subject taxonomy; the instantiation of the architecture to a given taxonomy requires manual effort that is supported by a tool that suggests translation rules from the web directory to the taxonomy. The architecture comprises of a web directory search component that determines a ranked list of categories of directory categories that are relevant to the query. A taxonomy mapping component translates these categories into (possibly multiple) nodes of the target taxonomy, and a conjecturing component combines this information into the final taxonomy categories. A logistic regression model trained with the Mitch software package combines the association weights between query and

taxonomy nodes, the ranking of the nodes, and a margin term into a probability used to determine the most likely nodes.
The KDD Cup 2005 provides a benchmarking framework for the query categorization task. Our system achieved the second-highest F-measure among all participants; this argues that our system and architecture provide a good and appropriate solution to this problem.

7. ACKNOWLEDGMENTS

The work reported here was partially supported by the German Science Foundation DFG under Grants SCHE540/10-1 and SCHE540/10-2. We acknowledge A.I. Insight, Inc. and MEDai, Inc. for the use of their predictive modeling technology Mitch (Multiple Intelligent Tasking Computer Heuristics).

8. REFERENCES

[1] R. Agrawal and R. Srikant. On integrating catalogs. In *Proceedings of the Conference on the World Wide Web*, 2001.

[2] J. Allan and H. Raghavan. Using part-of-speech patterns to reduce query ambiguity. In *Proceedings of the International ACM SIG Conference on Research and Development in Information Retrieval*, 2002.

[3] P. Anick and S. Tipirneni. The paraphrase search assistant: terminological feedback for iterative information seeking. In *Proceedings of the International ACM SIG Conference on Research and Development in Information Retrieval*, 1999.

[4] P. Bruza and S. Dennis. Query reformulation on the internet: Empirical data and the hyperindex search engine. Proceedings of the Conference on Computer-Assisted Information Searching on Internet, 1997.

[5] D. R. Cutting, J. O. Pedersen, D. Karger, and J. W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings of the International ACM SIG Conference on Research and Development in Information Retrieval*, 1992.

[6] S. Dennis, P. Bruza, and R. McArthur. Web searching: a process-oriented experimental study of three interactive search paradigms. *American Society for Information Science and Technology*, 53(2):120–133, 2002.

[7] A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Ontology matching: A machine learning approach. In S. Staab and R. Studer, editors, Handbook on Ontologies in Information Systems. Springer-Verlag, 2003.

[8] N. Glance. Community search assistant. In *Proceedings of the International Conference on Intelligent User Interfaces*, 2001.

[9] J. Gonzalo, F. Verdejo, I. Chugur, and J. Cigarran. Indexing with wordnet synsets can improve text retrieval. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics and International Conference on Computational Linguistics*, 1998.

[10] D. Hosmer and S. Lemeshow. Applied logistic regression. New York, Wiley, 1989.

[11] D. W. J. Stefanowski. Carrot2 and language properties in web search results clustering. In *Proceedings of the International Atlantic Web Intelligence Conference*, 2003.

[12] M. Sanderson. Retrieving with good sense. *Information Retrieval*, 2(1):49–69, 2000.

[13] H. Schutze and J. Pederson. Information retrieval based on word senses. In *Proceedings of the Annual Symposium on Document Analysis and Information Retrieval*, 1994.

[14] C. Stokoe, M. Oakes, and J. Tait. Word sense disambiguation in information retrieval revisited. In *Proceedings of the International ACM SIG Conference on Research and Development in Information Retrieval*, 2003.

[15] M. K. Y. Wang. C4-2: Combining link and contents in clustering web search results to improve information interpretation. In *Proceedings of the International Conference on Database and Expert Systems Applications*, 2002.

[16] O. Zamir and O. Etzioni. Grouper: a dynamic clustering interface to Web search results. *Computer Networks*, 31(11–16):1361–1374, 1999.

[17] D. Zhang and W. Lee. Learning to integrate web taxonomies. *Web Semantics*, 2(2):131–151, 2004.

About the Authors

David Vogel is the Lead Scientist at A.I. Insight, where he has spent over 7 years leading the development of their modeling tool MITCH (Multiple Intelligent Tasking Computer Heuristics). David's research interests include development of innovative modeling techniques, scalable algorithms, and new applications for predictive models.

Steffen Bickel is Ph.D. student at Humboldt University, Berlin. His research interests are machine learning methods for information retrieval.

Peter Haider, Rolf Schimpfky and Peter Siemen are Master's students in computer science at Humboldt University, Berlin, with a focus on machine learning methods.

Steve Bridges is the Software Architect at MEDai. He has more than sixteen years of experience in developing software, much of it software for decision support and data mining applications.

Tobias Scheffer is Assistant Professor at Humboldt University, Berlin. He received his PhD in 1999 from Berlin University of Technology. He was awarded an Emmy Noether Fellowship of the German Research Foundation in 2002 and an Ernst von Siemens Fellowship in 1999. He has previously been working at Magdeburg University, the University of New South Wales, Sydney, and Siemens Corporate Research, Princeton. His research focuses on machine learning and information retrieval.