An Architecture for Emergent Semantics

Sven Herschel, Ralf Heese, and Jens Bleiholder

Humboldt-Universität zu Berlin
Unter den Linden 6, 10099 Berlin, Germany
{herschel, rheese, bleiho}@informatik.hu-berlin.de

Abstract. Emergent Semantics is a new paradigm for inferring semantic meaning from implicit feedback by a sufficiently large number of users of an object retrieval system. In this paper, we introduce a universal architecture for emergent semantics using a central repository within a multi-user environment, based on solid linguistic theories.

Based on this architecture, we have implemented an information retrieval system supporting keyword queries on standard information retrieval corpora. Contrary to existing query refinement strategies, feedback on the retrieval results is incorporated directly into the actual document representations improving future retrievals.

An evaluation yields higher precision values at the standard recall levels and thus demonstrates the effectiveness of the emergent semantics approach for typical information retrieval problems.

1 Introduction

The elementary challenge in all retrieval tasks is to find an object representation that can later effectively and efficiently be matched against a user query in order to find and rank the objects according to the user's needs.

Researchers in information retrieval (IR), to select a prominent example, have been very successful in finding document representations for later retrieval. Albeit being today's state of the art, the vector space model [1] used in conjunction with Latent Semantic Indexing [2], constitute only a syntactical approach in finding a so-called semantic representation.

Emergent semantics aims to emerge object representations by aggregating many user's opinions about the object content, therefore providing object representations that a majority of actual users of a system agree upon. We believe that finding such a representation considerably improves precision, since it was created by the users themselves. A basic example illustrates the idea behind emergent semantics: In a park near our campus, the landscape architects decided on not paving walkways initially. Instead, they covered the entire area with lawn. After a year they came back and knew exactly where to pave walkways, since the walkers had obviously decided which pathways they will use by actually walking them: the paths were all torn and muddy. We claim that this approach can be transferred into the area of computer science and, termed emergent semantics, represents a major advancement in the way object representations are created and maintained.

J.F. Roddick et al. (Eds.): ER Workshops 2006, LNCS 4231, pp. 425–434, 2006.
© Springer-Verlag Berlin Heidelberg 2006

Our contribution with this paper is a formal approach to the emergent semantics paradigm, an architecture for utilizing its full potential and an implementation within the area of IR that demonstrates the possibilities associated with this paradigm. Our preliminary results show higher precision values at the standard recall levels, thus demonstrating the effectiveness of the emergent semantics approach for typical IR problems.

Structure of this paper. This paper is structured as follows: in Section 2, "Groundwork", we introduce the linguistic background of syntax, semantics, and pragmatics. We then adopt these cognitions for computer science by introducing a universal architecture for emergent semantics in "Model House" (Section 3). In Section 4, "Construction", we introduce our implementation of this architecture for a classic IR scenario in order to be able to present the promising results of our evaluation in Section 5. Related work is outlined in "Neighborhood" (Section 6) and "Roof and windows" (Section 7) concludes the paper with an outlook on open research issues and future work.

2 Groundwork

In this section, we introduce the linguistic background of syntax, semantics, and pragmatics in order to motivate our universal architecture for emergent semantics. This is essential for understanding the process leading from the user need, expressed by a query, to the retrieval and ranking performed by the system.

Semiotics is the study of signs, their meaning, and their interpretation by humans. The three subfields of semiotics are, in accordance with Morris [3]:

Syntax: the study of signs and their interrelation
Semantics: the study of signs and their relation to the objects they represent
Pragmatics: the study of signs and their relation to the user interpreting them

According to this theory, every sign is assigned a meaning within the context of a person's understanding. So the letters *t, r, e*, and *e* form the word *tree* which, in English, refers to the biological wooden structure. Different people, however, may have something different in mind when being confronted with the word *tree*. Concepts may reach from a beautiful day in the park to the latest forest fire in Portugal. Similarly, the section headlines of this paper were deliberately chosen to potentially carry different meanings within this paper and outside its scope.

Algorithms for generating object representations from objects are usually syntactical with the implied hope of inferring something similar to semantics by applying clever extraction algorithms. An example for such an advanced algorithm is Latent Semantic Indexing (LSI) in the context of textual IR [2] which claims to extract document representations that are – by human judgment – considered good representations for the respective documents.

The semiotic triangle (see Figure 1) puts the three semiotic concepts (syntax, semantics, and pragmatics) into relation. The directed edges between the concepts stand for semiotic transitions that can potentially be explored by computer science algorithms.

Latent semantic indexing tries to walk the line from syntax to semantics: they analyze the document content and create a term-document matrix identifying the most important terms for each document and the most discriminating terms within a document collection. These algorithms therefore extract the meaning of a document by representing a document with terms found in the document collection. The obvious drawback of this purely syntactical approach is that only terms already found in the document collection itself can be used within document representations.

Emergent semantics in turn walks the line from pragmatics to semantics. It aggregates different users' opinions about the meaning of an artifact and creates an artifact representation that the majority of users agress upon.

Pragmatics includes the specific user background, i.e., culture, education, and social background, as well as very time-dependent influences like the user's mood, for example. It is our assumption that this diverse background will lead to a lot of noise, if applied unconditionally to the document representations. It is therefore necessary to somehow eliminate this noise. This elimination of unwanted noise can essentially be accomplished in two ways: (1) by applying user specific filters ("user profiles") and possibly aggregating these into opinion networks ("collaborative filtering") or (2) by assuming that the majority of users knows best and incorporating users' opinions directly into the document representation – through the noise semantics will emerge.

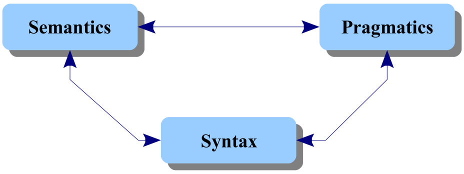

Fig. 1. Semiotic triangle

We believe that an architecture for emergent semantics needs to provide facilities for both approaches (see Section 3), however, we already achieved promising results with the later approach alone (see Section 4).

3 Model House

In this section, the basic building blocks of an emergent semantics (EmSem) architecture are introduced and their functionality is explained. The architecture is best explained describing the query processing workflow. Please refer to Figure 2 for an overview of both the architecture and the fundamental query processing steps. For now we assume a simple environment consisting of a central repository storing all data and a large number of clients querying the server.

3.1 Ingredients

In the context of emergent semantics, we understand object retrieval as a four step process. At first, a user develops an idea of her information need and formulates a *query* to the best of her knowledge (arrow 1). This query implicitly includes her individual context, e.g., her academic or social background.

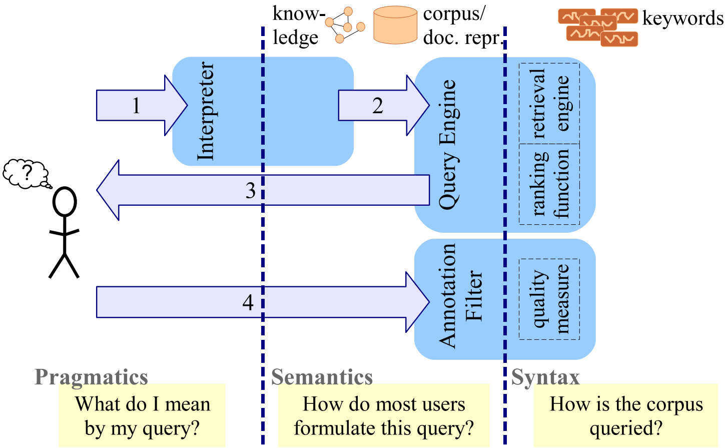

Fig. 2. An architecture for emergent semantics

The query is then analyzed by a component we term the *interpreter*. The interpreter is responsible for reformulating the query in such a way that the user's pragmatic background is resolved and explicitly stated as part of the query. This is accomplished in two ways: (a) utilizing query expansion or query reduction strategies in order to include a specified user context and thus reduce the possibilities of (mis)interpreting the query, (b) calibrating the query to be in accordance with the retrieval system, i.e., replacing terms with terms from a controlled vocabulary. An example for (a) is detailed in [4].

The result of this interpretation step is what we term a *canonical query*, i.e., a query that does not contain any pragmatic context any more (arrow 2). The interpreter thus bridges pragmatics and semantics on the query processing side.

The canonical query is then fed to the retrieval system. Three ingredients are used by the query engine to retrieve and rank suitable objects: *keywords* (a syntactical representation of stored objects), *corpus knowledge* (knowledge that is derived from the entirety of the object collection) and *external knowledge* (knowledge that is independent from the object collection). The query engine therefore bridges semantics and syntax within our architecture.

The result of the query engine is a list of ranked results (arrow 3) which is returned to the user. These results include an object surrogate which the user evaluates to determine which documents fulfill her information need.

By actually retrieving the relevant documents the user feeds her *original query*, including all its implicit pragmatic context, back to the system (arrow 4).

If this last step is performed a sufficiently large number of times by a sufficiently large number of users, new document semantics will be created. This is the reason for us calling it *emergent semantics*: the pragmatic view of many users of an information system is gradually converted into document semantics.

In complex scenarios, an annotation filter with a specified quality measure, could reject the addition of new keywords, i.e., if it is not in accordance with regulations or to suit a specific retrieval model.

3.2 Distinctions

Three aspects of the emergent semantics paradigm should be emphasized, since they differentiate the approach from current state of the art:

Entirely new keywords. EmSem allows entirely new terms to be introduced into the system. Even advanced approaches like LSI or query expansion can only work with terms already contained in the collection. In addition, we tackle the synonym problem, since a sufficient number of users will annotate the document with relevant synonyms.

Changing document representations. Keeping things simple, EmSem directly alters document representations. No new layers in query processing, no user specific state to be held, instant gratification to all users.

Living document representations. The entire process is based on the assumptions that most users "know best". If users change their mind about the supposed meaning of an object, the meaning of the respective object will change over time (e.g., historic events that are reevaluated after some time, cars that become oldtimers, or changes in the use of language).

4 Construction

In this section, we instantiate the previously motivated architecture within the area of information retrieval. IR aims at satisfying a user information need usually expressed in natural language [5]. To accomplish this task and to effectively rank documents according to a user's needs, an IR system models the relationship between a query and the relevant documents to this query. Approaches include the Boolean model, the vector space model [6], or the probabilistic model [7]. In the following, we introduce basic terminology and give a global view on the problem of IR within our architecture.

4.1 Terminology

We consider a collection of objects, e.g., (text) documents or images, forming the *corpus* $\mathcal{C}$ on which retrieval is performed. A common way to summarize content and meaning of these objects is to represent them using keywords (interchangeably called terms in this paper). Therefore, each document contained in the corpus is represented by a finite set of terms taken from a *universe of terms* $\mathcal{T}$. This annotation is called the document representation.

Definition 1 (Document Representation). *The* document representation *of a document is a set of terms:* $r(d) = \{t_1, \ldots, t_n\}, d \in \mathcal{C}, t_i \in \mathcal{T}$.

In full-text IR, the prevalent form of IR today, these keywords correspond to the words of a (text) document, usually preprocessed and filtered, i.e., by a stopword filter or a stemmer, eliminating the most frequent words of a language and converting the remaining words into their canonical form.

The user expresses her information need by issuing a query to the system in natural language. As a query itself is interpreted as a document, we define it in the same way as the document representation:

Definition 2 (Query). *A user* query *is a set of terms:* $q = \{t_1, \ldots, t_n\}$, $t_i \in \mathcal{T}$.

As a result of query evaluation, the system returns a ranked list of document surrogates to the user. Based on this list, the user selects the documents which fulfill her information need. We define the answer to a query as follows:

Definition 3 (Answer). *Let* q *be a query. The answer of* q *is defined as* $D_q = \{d_1, \ldots, d_n\}, d_i \in \mathcal{C}$. *We denote with* $D_r \subseteq D_q$ *the set of documents classified as relevant by the user.*

Please note, that set D_r is specific for each user and each query of the system. Particularly, this implies that a specific D_r does not necessarily contain *all* relevant documents but only the ones *classified* as relevant by the user.

4.2 Running the System

Before effective IR can be performed, the entire collection must be indexed. This phase is called the bootstrapping phase and usually needs to be performed only once for each collection. As a result of bootstrapping, the components *keywords* and *corpus knowledge* of our architecture (see Figure 2) are initialized: while *keywords* represent the syntactical document representations, stemmed and reduced by stopwords, the *corpus knowledge* in our case is the term-document matrix of our collection filled with TF/IDF weights.

The *retrieval engine* is based on an inverted index of all terms $\mathcal{T}$ and the *ranking function* is based on the vector space model: both documents and queries are represented as term vectors carrying term weights from the term-document matrix above. The similarity between a query and each document is calculated as the angle between these vectors: the smaller the angle, the more relevant is the document for the respective query.

After the query has been processed, the user selects the relevant documents D_r by actually retrieving them. We assume that the result list contains enough information that a user can decide on the relevance of a document. This actual retrieval of the document leads to the document annotation of the relevant documents being completed with the query: $\forall d \in D_r : r(d) = r(d) \cup r(q)$. The intuition behind this is that if a document is found by the terms within the query and this document is additionally marked as relevant then all terms of the query are also related to the content of the document.

Since the TF/IDF-matrix depends on the document representation, some parts of the matrix have to be recalculated. The following list enumerates the possible changes to the document representation and recalculated elements of the TF/IDF-matrix. Let be $t \in r(q)$ and be $d \in D_r$:

1. Existing term ($t \in r(d)$): The weight of the term t increases for the document d, all other values for other documents remain unchanged.

2. New term/document combination ($t \notin r(d) \wedge t \in \mathcal{T_C}$): Weights for all documents are calculated, because the document frequency changes for t.
3. New term to corpus ($t \notin r(d) \wedge t \notin \mathcal{T_C}$): Since the document frequency of t is known in advance, it is sufficient to calculate the weight of the term t with regard to d.

In the context of the architecture presented in the previous section, adding all query terms to the document representation is only one way of a quality measure for the annotation filter, where all terms are considered to be of some and equal good quality. In the following, we outline some more sophisticated strategies to modify the document representation: A variation of the method described above is to ignore terms contained in the query having a high document frequency. These terms are not discriminative and thus, do not improve the document representation. As a side effect of omitting such terms, a smaller region of the TF/IDF-matrix has to be recalculated. Although they have yet not been validated we present two further strategies: (a) inclusion of additional external knowledge and (b) measuring the quality of a document representation with regard to its semantic precision. The first approach utilizes external knowledge to select the terms of a query being added to the document representation. For example, the decision of adding a query term may be based on a domain ontology, e.g., a term is only added if it is contained in the ontology. Considering the second approach, we will develop a quality function which measures the quality of the document representation and only add a new term if it increases the quality of the document representation.

5 Assessment

To evaluate our approach we chose the standard “Communications of the ACM” information retrieval collection (CACM). We chose this collection because it features an overlap between query terms as well as between corresponding result sets (see “Evaluation remarks” below).

5.1 Evaluation Setup

We used the document title and, if available, the document abstract, for indexing and retrieval. They were tokenized and indexed by the Apache Lucene inverted index [8] using the Vector Space Model with TF/IDF weighting. Then, the following steps were repeated for each query in the corpus:

Query. From the CACM corpus, a query is chosen and presented to the system as a disjunctive query (“or” semantics).

Retrieval and ranking. Documents are retrieved from the index and ranked using the vector space model with TF/IDF weighting of terms.

Feedback. According to the given gold standard, precision and recall measures are calculated. The query is then tokenized and attached to all relevant documents as content.

5.2 Evaluation Results

As performance indicators for our emergent semantics approach, we determined precision-recall-measures along the eleven standard recall levels. See [5] for a discussion of individual IR performance indicators.

The results in Figure 3 demonstrate a major improvement in retrieval performance after feeding back all query terms unconditionally. These results mean that retrieval performance increases with each query if multiple users pose the same query to the system. In addition, we experimented with a *quality measure* (see Figure 2), which only allows feeding back terms with a document frequency smaller than 300 (10% of the documents). This prevents feedback of frequent terms. However, we were surprised to see that the results were identical to the results before; we expected these frequent terms to introduce a lot of noise and therefore reduce precision. We believe that his lack of noise results from the very distinctive query terms within the CACM corpus: since the queries mostly contain infrequent terms, the amount of noise introduced did not do much harm.

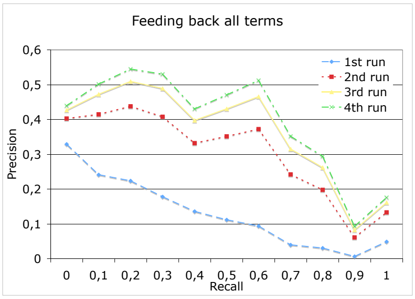

Fig. 3. Augmenting document representation with all query terms.

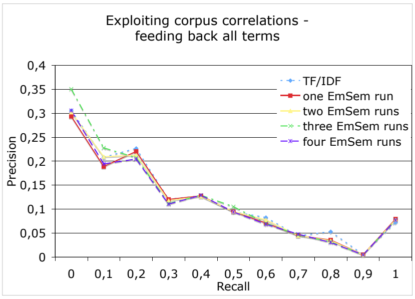

Fig. 4. Augmenting corpus part 1 with all query terms. Querying corpus part 2.

These results might have been anticipated: Few people would deny that feeding back the exact queries into the system will lead to improved retrieval performance for the same queries. Therefore, we split the corpus into halves and processed queries of the first half as described above. After different numbers of these runs (which we call "EmSem runs" in the figure), the second half was queried without feeding back these queries into the system. We were therefore able to measure the impact of query processing from the first half of the corpus onto the second (see Figure 4). While these results were not as stunning as the results in Figure 3, we were content to see that emergent semantics made an impact even in this scenario where neither the overlap between queries nor the overlap between result sets is big enough for a real-world scenario.

5.3 Evaluation Remarks

We chose the CACM collection after evaluation of the (also standard) Medline collection. It turned out, however, that there is no overlap in retrieval results between the queries defined in the Medline collection and therefore the emergent semantics approach will not work. Ideally, we look for a collection of highly correlated queries with a highly overlapping result set. In such a scenario, emergent semantics unfolds its full potential.

It should be noted that we do compare our results to today's best precision/recall values. It is our ambition to introduce the new paradigm of actually modifying document representations instead of using (possibly user-specific) query refinement strategies. Therefore, the baseline of our comparison is standard document retrieval using the vector space model and TF/IDF weighting. We expect emergent semantics techniques to have a similar impact on more sophisticated IR algorithms.

6 Neighborhood

Emergent semantics claims to integrate many users' opinions on objects in order to find better document representations. While the user context plays an important role both in collaborative filtering approaches [9,10] as well as in contextual service adaptation [4], they both introduce a separate layer into the object retrieval process. Emergent semantics as a paradigm leaves the underlying retrieval system untouched and achieves its goal through direct modification of document representations. This is also contrary to the standard approach in using relevance feedback where the query is expanded [5].

Annotation or tagging systems [11] bear some similarity to the emergent semantics paradigm, however, they usually require the user to explicitly determine suitable tags or annotations for the object. During system usage, object descriptions are not altered. Emergent semantics takes advantage of the user query (implicit information) to accomplish its goal. Similarly, Grosky et al. emerge the semantics of multimedia objects (i.e., web pages) by including objects along a user's browsing path into the context of the respective object [12].

Emergent semantics through gossiping [13] aims at determining schema mappings between independent information provider nodes through measuring the information quality along feedback cycles.

Emergent semantics relies on an underlying IR infrastructure, whether this is based on the vector space model with TF/IDF weights [1] or with LSI weights [2]. It advances these approaches to the integration of users' opinions into the system, thus allowing for more representative terms or even additional terms that have not yet existed within the collection. In addition, recalculation of LSI weights in the event of changes in the document collection is quite expensive.

7 Roof and Windows

In this paper, we presented emergent semantics, a new paradigm for integrating many users' opinions directly into an object retrieval system. On the linguistic side, this paradigm represents the shift from many user's individual pragmatic views to a unifying semantic representation of the object. We introduced an architecture capturing all aspects of emergent semantics and detailed the expected functionality of its components. An implementation of the emergent semantics approach within the area of IR, including promising results within a standardized evaluation on the CACM corpus, demonstrates the feasibility of our approach.

Future work includes exploration of other applications of emergent semantics, especially its potential ability to reduce the size of object representations, while still allowing for good retrieval results. We will also evaluate the applicability of the emergent semantics paradigm within a distributed environment.

Acknowledgment. This research was supported by the German Research Society (DFG grants no. NA 432 and GRK 316), and by the German Ministry of Research (InterVal Berlin Research Center for the Internet Economy).

References

1. Salton, G., McGill, M.: Introduction to Modern Information Retrieval. McGraw-Hill Book Company (1984)
2. Deerwester, S.C., Dumais, S.T., Landauer, T.K., Furnas, G.W., Harshman, R.A.: Indexing by Latent Semantic Analysis. Journal of the American Society of Information Science **41**(6) (1990) 391–407
3. Morris, C.W.: Foundations of the Theory of Signs. Chicago University Press, Chicago (1938)
4. Jacob, C., Radusch, I., Steglich, S.: Enhancing legacy services through context-enriched sensor data. In: Proceedings of the International Conference on Internet Computing. (2006)
5. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. ACM Press (1999)
6. Salton, G., Lesk, M.E.: Computer evaluation of indexing and text processing. Journal of the ACM **15**(1) (1968) 8–36
7. Robertson, S.E., Jones, K.S.: Relevance weighting of search terms. Journal of the American Society for Information Science **27**(3) (1976) 129–146
8. The Apache Software Foundation: http://lucene.apache.org (2006)
9. Shardanand, U., Maes, P.: Social information filtering: algorithms for automating “word of mouth”. In: Proceedings of the SIGCHI conference on Human factors in computing systems, New York, NY, USA, ACM Press. (1995) 210–217
10. Oard, D.W.: The state of the art in text filtering. User Modeling and User-Adapted Interaction **7**(3) (1997) 141–178
11. Heflin, J., Hendler, J., Luke, S.: SHOE: A knowledge representation language for internet applications. Technical Report CS-TR-4078 (UMIACS TR-99-71), University of Maryland (1999)
12. Grosky, W.I., Sreenath, D.V., Fotouhi, F.: Emergent semantics and the multimedia semantic web. SIGMOD Rec. **31**(4) (2002) 54–58
13. Aberer, K., Cudré-Mauroux, P., Hauswirth, M.: The chatty web: emergent semantics through gossiping. In: Proceedings of the Twelfth International Conference on World Wide Web, New York, NY, USA, ACM Press (2003) 197–206