LANGUAGE PRAGMATICS, CONTEXTS AND A SEARCH ENGINE

Ville H. Tuulos and Tomi Silander

Helsinki Institute for Information Technology (HIIT)
Univ. of Helsinki & Helsinki Univ. of Technology
P.O. Box 9800, FIN-02015 TKK, Finland

ABSTRACT

We introduce and motivate an approach for content-based information retrieval. We consider the corpus formed by real-world web pages as a dynamic and descriptive sample of natural language upon which measures of relevance can be built on. This setting is quite typical in information retrieval, although seldom in relation with the Web. However, we divert also from the information retrieval tradition by not trying to model syntax or semantics. Instead, we rely on the pragmatical dimension of language. The central characteristic of our approach is that it is lossless. Instead of building elaborate and often brittle abstractions based on the data, we let the user reflect her conception of semantics to the corpus in an efficient and flexible manner. We conclude with a few examples from our full-blown search engine which implements the ideas presented in this paper in practice.

1. INTRODUCTION

Abundance of unstructured written natural language in digital form, in particular in the World Wide Web, causes unprecedented challenges for information retrieval. The sheer amount bits per se is a considerable technical challenge. Yet the challenge posed by the content is unparalleled in its deepness. Suddenly the age-old philosophical dilemma of understanding the essence of natural language has become a part of casual marketing speech of corporations which are fiercely competing on the mastery of information. There is a danger that the challenge becomes so muddied by partly incompatible philosophical, linguistical, technical and economical interests that the underlying problem in itself does not get the attention it deserves.

A grand unifying theory on language would naturally serve all our needs. As with physics, it is not clear whether a goal like this is even worth of attaining. Even though we would regard it as an interesting challenge of its own, we argue that each field should keep its own focus and approach. Mixing different goals is likely to produce suboptimal results for every case. Naturally, knowledge and information should keep flowing freely between the fields.

In this paper we will first elaborate the above argument and then motivate our pragmatical focus on treating natural language. We will introduce a novel ranking scheme for content-based information retrieval which relies heavily on contextual information. We show that regardless of our practical focus and the seeming simplicity of our approach, the results often feel meaningful and understandable to the human observer. We will conclude with a few real-world examples from our content-based web search engine which well exemplify richness and unpredictability of pragmatics of natural language.

2. FOCI

In this section we will characterize three different foci for language modelling. By *language modelling* we refer generally to scientific inquiry which aims at forming *compact general statements*[1] on its subject of study, namely natural language. We argue that even though in practice the foci are not always recognized explicitly, they have different ontological commitments and distinctive goals. This diversity should be considered beneficial.

Zellig Harris' *Methods in Structural Linguistics* [1][2], first published in 1951, can be seen as the manifesto for an approach having its *focus on language*. The approach is strongly data-driven. It aims at providing a rigorous procedure so that one can start with raw data and end with statements of grammatical structure (H§2.2). One starts by tabulating linguistical elements with their contexts (environments) in which they appear in the data. Using the table of contexts, we may measure "freedom of occurrence" of each element and gain insight on the role of the element (H§7.22). We may measure similarity of two elements, A and B, by analyzing their mutual substitutability. If it is possible to substitute A with B in various contexts without altering the meaning radically, we may consider A and B similar (H§2.6).

This procedure does not allow us to make any statements about semantics in the first place (H§2.5). Note that the above measure of similarity relies heavily on the human observer's intuition about semantics. The goal is *observational adequacy* i.e. one aims at *describing* the data at hand compactly. Various connectionistic methods resemble closely this approach. Consider e.g Simple Recurrent Network for prediction of tokens [2] or a Self Organizing Map which automatically groups tokens appearing in similar contexts near each other, revealing some grammatical structures [3]. Naturally the connectionistic

[1] Phrase *compact general statements* was originally used by Zellig Harris in [1] (H§13.4) to describe the goals of distributional linguistics.

[2] Sections in the Harris' book are referred as (H§N.N)

methods replace the manual measure of similarity with a computable one, such as the Euclidean metric.

Noam Chomsky's *Syntactic Structures* [4][3] exemplify the second focus, *focus on mind*. Focus on mind refers to *explanatory adequacy* – the primary goal is to model *the cause* for language skill (vs. behavior) so that we can *explain and understand* the phenomenon. In contrast to focus on language, Chomsky states that "it is absurd to attempt to construct a grammar that describes observed linguistic behavior directly" [5], stressing the fact that the approach is not data-driven but relying on *a priori* knowledge.

Both Harris and Chomsky recognize infeasibility of a purely syntactical model for language viz. a grammar lacking the dimension of semantics (H§2.5, C§10). It is impossible to decouple natural language to some independent components or to a linguistic hierarchy so that the levels would not interact in subtle ways, including semantics (H§18.4, C§8.1).

Language phenomena which frequently have a strong semantic component include ambiguity, productivity and, at a more grammatical level, parts of speech. In computational modelling point of view, semantics lurk in model ambiguity (many models seem equally good), imperfect model assumptions (we do not know the phenomenon well enough), lack of scalability (abundance of seemingly different tokens) and especially in the choice of similarity measure.

We do not make any claims what may be inherently beyond the reach of our models. Yet we recognize the immense deepness of questions which are related to semantics. As noted in the introduction, language modelling is of enormous practical importance nowadays. Many practical applications which deal with natural language are not actually interested in observational nor explanatory adequacy, content-based search engines being a prominent example. They do not have to describe language behavior thoroughly nor they have to provide a satisfactory explanation for causes of the language skill. It suffices that they are *practically adequate* and work in a designated, constrained domain. We call this approach *focus on tools*.

Chomsky distinguishes between *problems* and *mysteries* [6] in language modelling. The latter refers especially to questions which insidiously lead us to model the whole human mind as a subproblem of solving the semantics[4]. While focusing on tools, we try to avoid mysteries and focus on problems. This tenet follows closely the *End-to-end argument* which states that "Functions placed at low levels of a system may be redundant or of little value when compared with the of cost of providing them at that low level." [8]. In other words, functions of the system should be moved closer to the application that uses the function, instead of providing them as a generic service. In the context of language modelling, we interpret semantics being such a function. Focus on tools may be seen as an unholy alliance between Harris' and Chomsky's approaches: We resort to data-driven models but we do not expect finding there semantics.

Yet the question remains whether a third way like this is feasible at all. In the following we will show that using simple methods that do not involve any semantics, we may attain a working compromise. The semantics stay in the user's end who on the other hand may outsource the burden of bulk data processing to the machine.

3. CONTENT-BASED SEARCH

There is no obvious method to fetch documents from a large document collection. The most obvious one is to filter the corpus with keywords that have to occur in the document. In its simplicity and comprehensibility, the keyword search is a powerful tool that can be implemented effectively. However, the basic keyword filtering has many shortcomings:

1. It is not necessarily the keyword we would like to use as a filter but a concept related to the keyword. A word may have many different meanings (homonymy) and one meaning can be conveyed with many different words (synonymy).

2. Filtering is not enough to provide a useful retrieval result. The document set may be far too large and we need to rank the documents so that the most desirable ones get top rankings.

3. Inflection, in languages like Finnish this is a big problem.

Clearly it is communication of meaning that is the major problem. Despite of our hopes, in it detachedness of the world as experienced by humans, a database engine holding semantic content is not a semantic agent, thus communicating semantics to it appears to be impossible, or at least so with our current practices. Therefore, we have to rely on the corpus itself that has been produced by and for humans. This simple reasoning emphasizes the transparency of the search engine. The user should be able to understand the operations performed by the search engine, so that the search engine could be effectively used as a tool. Transparency should also facilitate query refinement so that user could modify the query if the search results for the first retrieval attempt were not satisfactory.

Stating desiderata for search engines is one thing, fulfilling them another. It is not immediately clear what are the easily extractable and comprehensible features of the documents that could be used as cues for retrieval. Depending on the corpus, there may be some structured information attached to the documents, such as dates, topic categorizations, language, location. Clearly these should be fully utilized. However, often unstructured natural language content contains the information we are mostly interested in, posing a difficult problem for search engines.

Driven by our desiderata, but realizing that there are not many simple features of the document content available and bound by limited computational resources, we

[3] Sections in the Chomsky's book are referred as (C§N.N)

[4] For discussion about subtle complexity of human-related phenomena, see [7]

have studied a retrieval system that is based on a simple notion of co-occurrence of the words in a document. Conceptually the idea of two words occurring in a same document is not much more complicated than the idea of one word occurring in a document (even keyword search usually allows specifying many keywords), but it turns out that even this humble reach induces structures that are not only computationally challenging but also possibly useful as a basis for document retrieval.

The idea of using lexical co-occurrences in information retrieval dates back decades. One can see the connection between distributional linguistics (cf. Harris above) and our approach. However, we try to keep the *a priori* modelling assumptions at minimum. We use the co-occurrence matrix of words as the whole, without reducing its dimensionality or imposing any other restrictions to it. We replace the notion of word with context in which it appears. Correspondingly a document is a set of contexts.

This approach takes the pragmatical dimension of language to the extreme. We observe and record the language usage *in vivo* and, without any further assumptions, reflect given queries to the context in which it has been used. One might see a peculiar connection between the Wittgensteinian language games and the pragmatical context matching game of ours. However no such connection should be taken too seriously.

In the following we will describe the details of our method.

3.1. Document

We are interested in documents. A document is a N_d-length finite sequence of tokens

$$d = (t_1, ..., t_{N_d}).$$

We do not need to explicitly define tokens. For simplicity, one may interpret tokens as words. Likewise there is no need to define order in the sequence rigorously. Intuitive idea about order of words suffices.

If we lose the order in the document we get a multiset

$$D' = \{t_1, ..., t_{N_d}\}.$$

Like with ordinary sets, order is ignored but multiplicity of tokens is explicitly significant. This gives us a so called *bag of words* representation for document which is dominant in the information retrieval tradition. In this paper we will only consider binary bags of words, thus we may cast multiset D' to an ordinary set D.

Set of documents is called a corpus $\mathcal{C}$. Set of all words occurring in a corpus is called a lexicon $\mathcal{L}$. In the following, term document will refer to its bag of words representation D unless otherwise noted.

3.2. Lexical co-occurrences

For each word $l \in \mathcal{L}$ we may define an *inverted set* i.e. the set of all documents in which l occurs. Let T_l denote the inverted set for the word l, formally

$$T_l = \{D \in \mathcal{C} | l \in D\}.$$

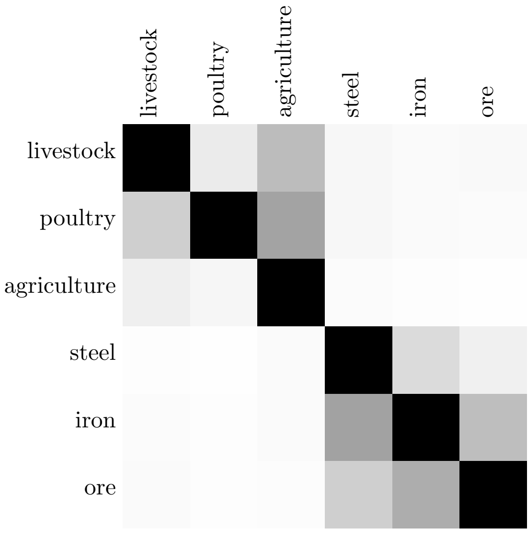

Figure 1. Conditional distributions of a few words

By definition, T_l is non-empty for each $l \in \mathcal{L}$. Now let us pick two words $m, n \in \mathcal{L}$. If the intersection

$$A = T_m \cap T_n$$

is non-empty we say that words m and n co-occur i.e. they appear at least once in the same document. We may enumerate all possible word pairs in a $L \times L$ matrix Λ and define

$$\Lambda_{ij} = |T_i \cap T_j|.$$

Due to symmetry of the intersection, we may consider only the upper triangular matrix. We call Λ word-word or lexical co-occurrence matrix. For brevity, we will refer Λ simply as co-occurrence matrix. A co-occurrence matrix tells how many times a word appears together with another word. Intuitively, we are interested in how often a word appears in the same context with another word, context being here a document.

We may illustrate parts of the matrix. Since items of Λ are unbounded sizes of intersections, they are not convenient for visualization as such. Therefore we consider conditional distributions

$$P(m|n) = \frac{P(m, n)}{P(n)} = \frac{\Lambda_{mn}}{\sum_{l \in \mathcal{L}} \Lambda_{ml}}$$

which define the probability of seeing word m given that we have seen n. In general $P(m|n) \neq P(n|m)$, so we show the whole matrix.

Figure 1 shows the conditional distributions for six chosen words from a public corpus of 806,791 Reuters news articles in English [9]. The corresponding full co-occurrence matrix consists of 389,988 words. As the image clearly shows, the freedom of occurrence for a word is strongly constrained. The image shows that the lexical co-occurrences are governed by some statistical invariances, thus the matrix might indeed contain some useful information.

Concerning the chosen words in the example, it is important to distinguish between interpretation about their semantical relatedness as seen by the human observer and their relation as it appears in the data. Since we are not building any further abstractions based on the matrix, such as topics or clusters, the labels are unnecessary and irrelevant for us. However, we hope that when reflecting a query of the user to the matrix, the resulting reflection makes sense to her. In many cases, the reflection reviles more about language pragmatics than about semantics, which is desirable in our point of view. This illustrates a crucial difference between our lossless approach and, for instance, clustering: A clustering method makes inevitably judgements about relevance *a priori*. Therefore, the user can not tell the difference between clustering artefacts or improper model assumptions and natural, yet unexpected, phenomena in the actual language pragmatics.

4. QUERY INTERFACE

Many recent attempts to formalize the query process have been based on the idea of a predictive query model where documents are ranked by the probability they give to the query [10]. The basic assumption in these models seems to be that the query is generated by the same kind of process as the documents and that the documents in their abundance provide a rich source of information that can be used to evaluate the similarity of the document and the query.

However, the assumption of the similarity of the document and the query is not necessarily a natural one. While both can be expressed using natural language, the generative mechanisms, i.e. the reasons producing the document and the query, are clearly different. This suggests using non-symmetric ”similarity” measures (such as Tversky similarity [11]) or abandoning the idea of the similarity as a basis of the ranking altogether.

In our system the query is a kind of superword that is represented in the same space as the words of the documents i.e. as a set of documents. We call these sets of documents quesets. The idea somewhat resembles standard query expansion procedure in information retrieval [12, 13, 14]. Give a set of query words S the corresponding queset is

$$Q = \bigcup_{s \in S} T_s$$

Note that Q resides in the same topological space as the words represented by T, thus queset Q may indeed be seen as a “possible word” or superword. As the queset is the “looking glass” through which we observe the corpus, we would like it to be easily malleable by the user. Even though desirably straightforward, the above union is not the only way to form the queset. For instance, we often include stemmed word forms for S in Q in our search engine.

The role of the enquirer is then to build a queset that is used to rank the documents of the corpus. In a representation space the query formation itself selects a set of documents, thus the relation of the query and a document to be retrieved becomes a relation between a document set (the queset) and the collection of the document sets (contexts of the words of the document). In this scheme, the query resembles single word rather than a document. For transparency, it is desirable that the user, while operating with words, could perceive the query formation as a way to select a set of documents that represent the query.

One may conceptualize the idea by considering documents as points in some space. In our contextual representation, a word l is a union of possibly disconnected regions, as defined by T_l and a queset is a union of these regions. Relevance or score of a document with respect to a query is a measure of overlap between the queset and regions spanned by words of the document. Formally, we define the score for a document D given queset Q as follows

$$Sco(D|Q) = \frac{1}{|D|} \sum_{l \in D} \frac{|Q \cap T_l|}{|Q \cup T_l|}.$$

The above similarity between the queset and a word context is the standard Tanimoto or Jaccard coefficient[12]. The document score is simply a sum of scores of its words, normalized by the document length. Naturally one is free to choose an alternative similarity measure, such as the Tversky similarity [11]. Especially, the measure does not have to be symmetric with respect to Q and D.

4.1. Keys & cues

Quesets are operands in ranking. However a search engine must also decide what to rank, not just how.

For example, consider that the user is interested in George Bush’s foreign politics and provides us the words “George Bush foreign politics”. There is no way to see whether the user is explicitly interested in *foreign politics* in the context of *G.B* or in *G.B* but mainly in contexts dealing with *foreign politics*. We see the above being an example of *asymmetric query* where the query words have actually different roles. Query word weighting [15] is traditionally seen as a way to tune the query but it does not suit well to the cases in which the words have actually equivalent importance but different roles. It seems overwhelmingly difficult to infer the roles automatically. In many cases, such as above, it is not even possible.

We propose that the user should be able to make the distinction clear in asymmetric queries. The user sees a normal query interface and she may perform queries by typing in a few words. In this case we find all the documents containing all the query words. The matching documents are ranked using all the given words. This is the *key* -part.

In addition, user may optionally give *cues* for ranking. Recall the previous example: In this case the user is given an opportunity to type “George Bush /foreign /politics”. The cue words are prefixed here with a slash but the actual way to make the distinction clear is not crucial. We could even have two separate query boxes for keys & cues. In this case we find all the documents containing the words *George Bush* (the key) but rank the matching documents only with respect to the words *foreign politics*. Thus all the documents necessarily mention *G.B* and the documents probably dealing with *foreign politics* are ranked to the top.

Keys & cues also solves ambiguity of some queries nicely. Consider e.g. query “apple”. There is no way to know whether the user is interested in fruits or the company Apple. However the queries “apple /computer” or “apple /steve /jobs” or “apple /banana” are all practically unambiguous in the ranking point of view. Note that a cue may be extremely vague thanks to the ranking scheme.

It is perfectly feasible to make a search using only cue words. This is actually a form of query by example. However, in technical point of view, even a few keywords usually reduce the number of documents to be ranked considerably. By requiring at least one keyword and by excluding the most frequent of them, we may reduce the computational load caused by ranking.

We combine the best from the both worlds: The exactness and versatility of keyword queries with the ability to return relevant documents given only some vague words. One might see the extra syntax as an additional burden to the user. However, considering our approach it is crucial that the user has the full control on search results. Following the end-to-end argument, the user must be allowed to utilize all her semantical capabilities since the system in itself lacks them.

5. RANKING ALGORITHM

In this section we will present the actual algorithm for ranking documents, following the ideas presented above. It is clear that the full co-occurrence matrix can not be represented as such, due to its quadratical growth with respect to the size of lexicon. The matrix would take about 283 gigabytes for the Reuters corpus and 465 terabytes for our full index on the Finnish web[5]. Instead, we rely on the fact that inverted sets T are readily efficiently represented in the inverted index of a typical search engine, like ours.

Consider the following naive brute-force algorithm to compute the score for each document according to the above ranking model:

```
1. Form queset $Q$ based on cue-words in $S$:
    $Q = \emptyset$
    For each $s \in S$:
        $Q = Q \cup \texttt{inverted\_index}(s)$
```

Function $\texttt{inverted\_index}(s)$ returns a list of documents containing word s from the inverted index. Thus Q is easily formed just with straightforward requests to inverted index.

```
2. Compute word's score $score_l$ for each word in the index:
    For each $l \in \mathcal{L}$:
        $S = \texttt{inverted\_index}(l))$
        $score_l = \texttt{isect}(Q,S)/(|S| + |Q| - \texttt{isect}(Q,S))$
```

This step involves going through the whole inverted index and calculating the intersection between the set of documents containing a word, l, and the query set. The score for each word is saved to an array. Note that the operation may be implemented as a continuous sweep over the inverted index which optimizes the cache usage.

```
3. Compute score for each document:
    For each $D \in \mathcal{C}$:
        $Sco(D|Q) = 0$
        For each $l \in D$:
            $Sco(D|Q) = Sco(D|Q) + score_l$
        $Sco(D|Q) = Sco(D|Q)/|D|$
```

As can be seen, implementation of the ranking method is trivial given a properly structured index. Computational load is caused by the amount of data accessed per each query, not by some particularly expensive computations. However an efficient implementation of set intersection, `isect`, above is crucial. The algorithm is embarrassingly parallel, both with respect to the lexicon and the ranked documents.

6. EXAMPLES

The ideas presented in this paper have been implemented in a full-blown content-based search engine called Aino. The implementation currently scales to millions of documents. The chosen lossless approach shows its full power in large-scale realistic settings which include every easily conceivable topic and large spectrum of various forms of language usage. We have built a publically available search engine[6] for the Finnish web, currently covering some 4.2 million web pages (documents) and 11 million tokens.

Table 2 shows a few words from the Reuters corpus and their closest neighbors according to its co-occurrence matrix. Table 2 shows similar neighborhoods for the Finnish web. Inflected word forms are prominent in the Finnish results. Actually Aino performs stemming but keeps them separate from the inflected forms which are retained as well. Language productivity and pragmatical richness shows up nicely with word “äpy”, a humorous student magazine, which seems to cause superfluous use of Scandinavian characters ä and ö in the nearby words. Aino makes possible to search documents with some stylistical cues. Word “yxin”, a teenage slang form of word “yksin” brings up similar slang words.

Next we show some realistic queries to Aino and the corresponding three top-ranking snippets, as returned by Aino. First consider ambiguous keyword “jukola” with two different cue-words which help to resolve the ambiguity even though the cues are not quite exact.

- **jukola /simeoni**

    1. *Simeoni, liuhuparta, valittaa se ”ihmisparka, syntinen, saatana, kurja”.*

    2. *Juhani, Tuomas, Aapo, Simeoni...*

[5] The figures are for dense matrices, yet the co-occurrence matrix is inherently sparse due to Zipf’s law. Even though only 0.01% of matrix entries would be occupied, the matrix would still be impractically large

[6] See http://aino.hiit.fi

Table 1. Reuters: Words with closest neighbors

Word	Neighbors
the	of, to, on, in
although	still, there, some, could
space	nasa, earth, mir, shuttle
nuclear	megawatt, mw, weapons, reactor
finland	helsinki, finnish, markka, sweden
apocalypse	horsemen, rougee, irsee, activision
boom	bust, exchequer, economist, recession

Table 2. Aino: Words with closest neighbors (in Finnish)

Word	Neighbors
akrr	akrr05, amklc, krbio, openconf
semanttinen	semanttisen, semanttisten, semanttista, semanttisesti
pragmatiikka	pragmatiikan, semantiikka, fonologia, kontrastiivinen
parturi	kampaamo, kampaamot, maahantuojat, kauneudenhoito
nenä	korva, kurkkutaudit, sisätaudit, naistentaudit
matala	korkea, pensasmainen, kasvista, lämpöisestä
äpy	äpyvän, rähästö, äpyn, wappulehti
yxin	voisittexte, iltajutust, listäkää, burggaballonkin
halonen	tarja, presidentti, tasavallan, halosen

3. *Heikki Kinnunen (Aapo), Heikki Alho (Simeoni), Arno Virtanen (Timo), Ilari Paatso (Lauri) ja Juha Muje ...*

- **jukola /juoksu**

1. *Nuorten Jukola 2002*
2. *on tullut tutkittua suunnistuskarttoja (tio-mila, jukola, tanska jne.*
3. *Jukola-katsastus...*

With mainstream search engines it's often difficult to find honest opinions about products since the results are biased by commercial product pages. Aino lets you to make searches concerning the tone of language by specifying cues with the desired tone.

- **mcdonalds /kamalaa /yäk /kuvottaa**

1. *Face it, you smell like McDonalds and Wallmart/ By killing you I'm akting globally, doing a small part.*
2. *liha tulee ulkomailta (siis jos mun mcdonalds tietämys pitää paikkansa).*
3. *McDonalds on vähän toisenlainen ongelma, terveydellinen ongelma.*
4. *'ylikanallista mcdonalds'-kulttuuria, joka yksinkertaisesti hävittää molemmat kulttuurit?*

Sometimes you just don't know what would be the correct keywords to solve your problem, for instance if your operating system crashes.

- **windows /kaatuu**

1. *Specified DLL funktion not found" ja ohjelma kaatuu....DLL tiedosto on viallinen tai se puuttuu*
2. *Win media player kaatuu ...Kappas kehveliä, Windows media player kaatuu heti(Windows Media Player on havainnut virheen, ja tuote on suljettava.*
3. *VMwarelta tärkeä korjaus Windows 2003 server virtuaalikoneisiin*

7. CONCLUSION

We argued that a third way, focus on tools, might prove useful in contrast to the foci dealing with semantics. The rationale was that semantics would force us to make complex and often brittle model assumptions *a priori*, such as the choice of similarity measure. On the other hand, each model assumption would probably affect semantical meaningfulness of the model behavior and results. Especially, a generic search engine has remarkably little knowledge on relevance before seeing the query. Furthermore, the query contains scarcely information about the user's conception of relevance. We aim at reflecting it as faithfully as possible to the language as it is used and let the user judge the results with respect to her current "inner semantic state".

We represent words by contexts in which they occur. The query is represented as a union of contexts viz. as a possible word. This setting allows the user to retrieve documents by explicating some vague cues to aid ranking. Explicit cues allow the user to evade the system's fallacies and experiment with different viewpoints on data. As the examples show, in many occasions the user is able to formulate queries matching her semantics.

8. ACKNOWLEDGMENTS

Aino the search engine was supported in part by the National Technology Agency, under the project Search-Ina-Box and by the Academy of Finland, under the project Prose. Further support was received by the IST Program of the European Community under the Alvis project. Tomi Silander was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views.

9. REFERENCES

[1] Z. Harris, *Methods in Structural Linguistics, 4th edition,* University of Chicago Press, 1960.

[2] J. Elman, "Language as a dynamical system," in *Mind as Motion: Explorations in the Dynamics of Cognition*, pp. 195–223. MIT Press, 1995.

[3] T. Honkela, V. Pulkki, and T. Kohonen, "Contextual relations of words in grimm tales analyzed by self-organizing map," in *Proc of ICANN-95, International Conference on Artificial Neural Networks*, Paris, 1995, pp. 3–7.

[4] N. Chomsky, *Syntactic Structures*, Mouton, eighth edition, 1969.

[5] N. Chomsky, "Acquisition of language," in *Chomsky: Selected Readings*. Oxford University Press, third edition, 1972.

[6] L. M. Antony and N. Hornstein (Eds.), *Chomsky and his critics*, Blackwell, first edition, 2003.

[7] V. H. Tuulos, J. Perkiö, and T. Honkela, "Modelling multimodal concepts," Tech. Rep. A75, Helsinki University of Technology, 2005.

[8] J. H. Saltzer, D. P. Reed, and D. D. Clark, "End-to-end arguments in systems design," *ACM Transactions in Computer Systems*, vol. 2, no. 4, pp. 277–288, 1984.

[9] D. D. Lewis, Y. Yang, T. Rose, and F. Li, "Rcv1: A new benchmark collection for text categorization research," in *Journal of Machine Learning Research*, vol. 5, pp. 361–397. 2004.

[10] W.B. Croft and J. Lafferty, Eds., *Language Modeling for Information Retrieval,* Kluwer Academic, 2003.

[11] A. Tversky, "Features of similarity," *Psychological Review*, pp. 327–352, 1977.

[12] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*, Addison Wesley, 1999.

[13] Yonggang Qiu and Hans-Peter Frei, "Concept-based query expansion," in *16th Annual Intl. ACM SIGIR Conference*, Pittsburgh, US, 1993, pp. 160–169.

[14] Jinxi Xu and W. Bruce Croft, "Query expansion using local and global document analysis," in *19th Annual Intl. ACM SIGIR Conference*. 1996, pp. 4–11, ACM Press.

[15] Gerard Salton and Christopher Buckley, "Term-weighting approaches in automatic text retrieval," *Inf. Process. Manage.*, vol. 24, no. 5, pp. 513–523, 1988.