High-Performance Information Extraction with AliBaba

Peter Palaga, Long Nguyen, Ulf Leser
Dept. of Computer Science
Humboldt-Universität zu Berlin, Germany
palaga,nqlong,leser@informatik.hu-berlin.de

Jörg Hakenberg
Arizona State University
Tempe, Arizona, US
hakenberg@asu.edu

ABSTRACT

A wealth of information is available only in web pages, patents, publications etc. Extracting information from such sources is challenging, both due to the typically complex language processing steps required and to the potentially large number of texts that need to be analyzed. Furthermore, integrating extracted data with other sources of knowledge often is mandatory for subsequent analysis. In this demo, we present the AliBaba system for scalable information extraction from biomedical documents. Unlike many other systems, AliBaba performs both entity extraction and relationship extraction and graphically visualizes the resulting network of inter-connected objects. It leverages the PubMed search engine for selection of relevant documents. The technical novelty of AliBaba is twofold: (a) its ability to automatically learn language patterns for relationship extraction without an annotated corpus, and (b) its high performance pattern matching algorithm. We show that a simple yet effective pattern filtering technique improves the runtime of the system drastically without harming its extraction effectiveness. Although AliBaba has been implemented for biomedical texts, its underlying principles should also be applicable in any other domain.

1. INTRODUCTION

Only a small fraction of the available knowledge in biomedical research is stored in databases. Much valuable information, and especially confirmed knowledge rather than raw experimental results, can only be found in the scientific literature. The most important collection of scientific publications is PubMed, which currently contains more than 16 million publications and grows by more than 500K documents per year.

PubMed basically supports only IR-style keyword searches. These are sufficient when specific pieces of information about a particular biological entity are sought, such as the interplay of the NFkappaB gene and the Cox-2 gene in colorectal tumors. This query results in only three abstracts that can easily be processed manually. The task gets cumbersome when queries are less specific or address classes of objects rather than single objects. For instance, a researcher who wants to find all transcription factors (TF) regulating expression of Cox-2 first needs to pose a series of queries to cope with linguistic variability (“Cox-2 regulation” (>3500 hits), “Cox-2 activation” (>2500 hits) etc.) and then has to study thousands of abstracts to find the relevant TFs. AliBaba is a tool that eases this task by automatically extracting and highlighting the most valuable information from PubMed search results (http://alibaba.informatik.hu-berlin.de). It takes a PubMed query as input and forwards it to the original PubMed service, thus leveraging the power of its search engine and the familiarity of users with its syntax. The resulting abstracts are piped through a text mining workflow that tags different classes of biological entities (proteins, diseases, drugs etc.) and semantic relationships between those entities (protein-protein interactions (PPI), association of a gene to a disease, etc.). The tagged abstracts are parsed by the client and the extracted information is displayed graphically (Fig. 4). Thus, AliBaba serves several purposes:

- It automatically extracts the most relevant information in large search results, providing a quick summary for the user.
- It displays extracted information in a graphical form with various options for navigation. Specific information can be found much quicker than from a list of titles or abstracts.
- Extracted data from multiple searches can be combined into a single graph, supporting the analysis of many query results.
- Extracted information is integrated in various ways with data from external databases. Additional information on all objects in the text is available immediately.
- Extracted information can be stored in an RDBMS or as an XML file.

In summary, AliBaba can be used to turn unstructured text into structured data records.

Parts of AliBaba have been described elsewhere. A user-centric presentation can be found in [11], and the pattern learning approach was described in [7]. Here, we focus on recent improvements in data integration and in the matching phase of the relationship extraction. In this stage, AliBaba uses a pattern-based approach offering higher precision at the cost of increased complexity compared to regular expression matching or simple co-occurrence. We describe a simple yet effective technique to filter patterns based on their potential to result in high scoring matches, which considerably improves the speed of the matching phase while incurring only a negligible penalty in effectiveness.

2. BACKGROUND

The extraction of facts from text has a long tradition in the IR and Machine Learning community. Recently, it has also drawn considerable attention in the database community [4]. However, most projects in the DB community currently only consider entity extraction [2, 3], while we also target relationships between objects. Relationship extraction is addressed using co-occurrence (CO, e.g. [9]), pattern matching (like AliBaba), and machine-learning (ML, e.g. [13]). The first (CO) suffers from low precision, while ML-based approaches still are too slow to be used in an online-setting.

EDBT'09, March 24-26, 2009, Saint Petersburg, Russia.
Copyright 2009 ACM 978-1-60558-422-5/09/0003 ...$5.00.

Pattern matching depends on the availability of a comprehensive set of patterns. Creating these patterns usually is performed manually (the system described in [14] required several hundred manually defined patterns). To alleviate this problem, we developed a method for learning language patterns from a database of pairs of related objects. This method will be summarized shortly in Section 3.2. Another difference between AliBaba and most other IE-systems is that AliBaba performs IE on the result of a (PubMed) query, while other systems work on entire corpora. Both approaches are important: Corpus-wide analysis is more suitable for an analysis of all extracted information, while query-based IE provides better support for the every-day work of scientists.

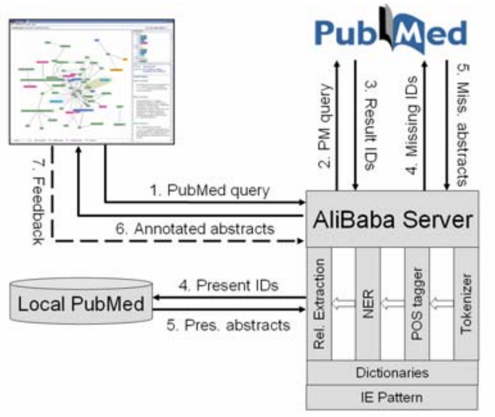


Fig. 1. AliBaba workflow.

3. SYSTEM OVERVIEW

AliBaba's architecture is depicted in Fig. 1. The client is a Java Web Start application where users formulate queries using the PubMed query syntax. Queries are forwarded to PubMed (step 2), which returns the IDs of all matching abstracts. Abstracts for these IDs are searched in a local PubMed copy (steps 4+5). IDs not present locally are sent back to PubMed to obtain the abstracts. All abstracts are processed by the AliBaba IE-pipeline (next section). Annotated abstracts are sent back to the client (step 6), which displays and enriches the information in various ways (Sec. 4).

3.1 IE Pipeline

The core of AliBaba is its ability to extract information from text. This task is performed by a pipeline consisting of several modules (in the style of [6]), including a sentence splitter, a tokenizer, a part-of-speech tagger (POS), a stemmer, and a dictionary-based named entity recognizer. Dictionaries are compiled from several important biological databases (UniProt, MesH, KEGG, DrugBank etc.), and the matching allows for slight linguistic variations (plural etc.). We prefer a dictionary-based NER approach over statistical methods because it not only recognizes the name, but also provides links to external sources important for data integration (Sec. 4.1). In the last step of the pipeline, sentences are analyzed to extract information about semantic relationships between previously recognized objects.

3.2 Extracting Relationships

AliBaba uses a pattern-based approach to the relationship extraction. Therefore, it learns typical linguistic representations of relationships and encodes them in *language patterns*. Patterns are matched against sentences using sentence alignment (Sec. 3.3).

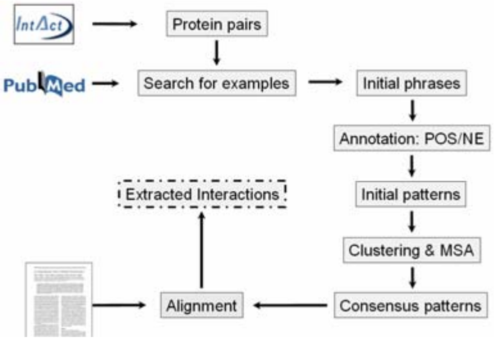


Fig. 2. Pattern learning and matching for relationship extraction using sentence alignment.

We first sketch how we learn patterns, using the extraction of protein-protein interactions (PPI) as an example (see [7] for details). An overview of our approach is shown in Fig. 2. We first retrieve pairs of interacting proteins from a database of PPIs (here: IntAct) and search PubMed for sentences containing both proteins from any pair and a word from a list of ~150 words indicating PPI ("binding", "phosphorylation" etc.). Each of these sentences is considered a positive PPI example. From those sentences, we extract the tokens between the three recognized words and a certain neighborhood around them (*core phrase*). From each core phrase we generate an *initial pattern* which consists of three layers: the phrase itself, a stemmed version of the phrase, and the sequence of POS tags of the phrase (see Tab. 1 for an example).

Orig. word	FADD	immediately	activates	procaspase-8
Ent. type / POS	PTN	ADV	VBZ	PTN
Word stem	PTN	immediat	activat	PTN

Tab. 1. A multi-layered language pattern; PTN: protein name, ADV: adverb, VBZ: verb, present tense.

If initial patterns are matched directly against new text, results show very good precision but low recall, because initial patterns essentially overfit to the training data. To generalize, we cluster patterns based on their similarity. We define the similarity of two patterns as the weighted mean edit-distance between the respective layers. For each cluster, we derive a *consensus pattern*, which may be thought of as the "average" of all patterns in the cluster. It is obtained by first aligning all patterns in the cluster into a multiple sentence alignment (MSA) and then deriving a consensus value for each layer and each column in the MSA. The consensus pattern thus is a generalization of all cluster members. By adapting the threshold for cluster quality, we may tune the systems towards precision (small and coherent cluster, specific consensus patterns) or recall (larger and more heterogeneous clusters, less specific consensus patterns). This is powerful possibility to adapt to different application scenarios.

In the search phase, PPIs are detected by matching text against all consensus patterns (see next section). For a given sentence, all patterns are considered as matches that achieve a score that deviates from the maximal achievable match score for this pattern only by a factor of *dec_factor*. *dec_factor* is another tunable parameter of AliBaba: increasing it shifts the performance towards higher precision at the expense of recall.

We evaluated our algorithm on the SPIES corpus [8]. Note that this manually annotated corpus is only used for estimating the expected performance of the learned patterns, and not for learning the patterns. We used ~120K initial patterns and clustered them into ~10K consensus patterns, where the cluster threshold was optimized for the F1-measure. We reach an F1-score of ~63%, which is among the best existing systems for PPI extraction [12].

3.3 Pattern Matching

In the matching phase of the relationship extraction step, we essentially need to match all patterns against all sentences. The matching itself uses alignment in all three layers (also known as edit distance or Levenshtein distance), i.e., it finds a order-preserving mapping between tokens/tags/stems in a consensus pattern and tokens/tags/stems in a sentence such that the cost of transforming one into the other is minimized. We thus have two sources of complexity: (1) Alignment is O(n*m), when m is the length of a pattern and n is the length of a sentence. (2) We need to compute a matching for every pair of (pattern, sentence). Both sources together result in high runtime requirements. The previous version of AliBaba required ~0.5sec to match a set of ~1000 patterns against a given sentence, leading to unacceptable waiting times when more than a handful of abstracts had to be analyzed.

Sequence alignment has been studied a lot in the bioinformatics community [1]. The bad news is that the alignment itself cannot be sped up considerably. Since patterns need not match an entire sentence, but only a portion of it, we need a certain form of *local alignment*, which, in contrast to the *global alignment* problem, since long withstands all attempts to find tricks for considerable runtime improvements [10]. The good news is that several techniques have been developed for the second source of complexity, i.e., the need to perform k alignments per sentence, when k is the number of patterns. We observed that usually only very few patterns eventually give rise to an extraction event, while the overwhelming number of patterns results in very low scores. Exploiting this observation, we devised an algorithm to filter patterns given a sentence. The idea is similar to q-gram methods [1, 5], but we apply it to patterns, not to words or sequences. Instead of performing k alignments, we, for each sentence separately, first chose n (typically 10-50) out of the k patterns and perform the alignment only for those. Of course, the choice of n should be such that recall does not suffer substantially.

Filtering is performed only at the POS layer, which proved most effective. Upon server start-up, we count the number of POS bi-grams in every pattern. Let B_i be the set of POS bi-grams in pattern p_i. Given a new sentence s, we annotate it with POS tags and then compute the set B_s of its POS bi-grams. For matching, we compute a score *score(s,p$_i$)* for every pattern:

$$score(s, p_i) = \frac{B_s \cap B_i}{B_s \cup B_i}$$

We rank the patterns according to this score and perform alignments only for the top-n ones. We experimented with various variations of this approach, such as using tri-grams or including the number of occurrences of a given bi-gram in a sentence/pattern instead of only looking at the set. However, we so far found our simple method to perform best.

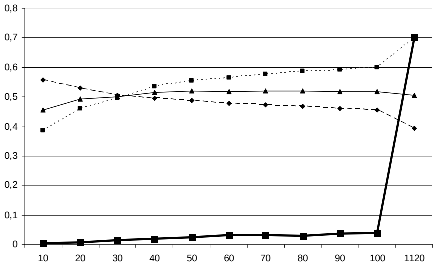
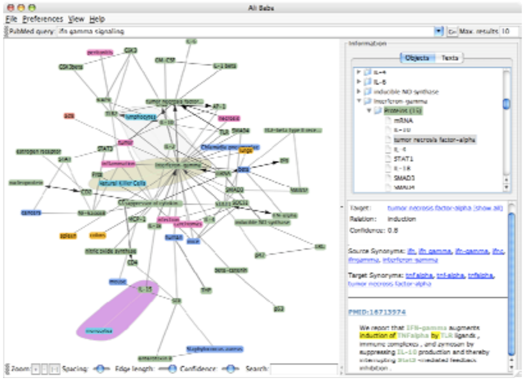


Fig. 3. Evaluation result. Dotted line: precision, dashed: recall, solid: F-Measure, bold: runtime (scaled). Note the jump from 100 to 1120 patterns in the rightmost measure.

Fig. 3 shows the results of experiments using a test collection of ~1000 sentences and 1120 patterns with *dec_factor*=0.95. Using all patterns, the system needed 421sec to analyze all sentences, achieving a recall of 0.7 at a precision of 0.4. Restricting the number of patterns drastically improves the runtime, increases precision and reduces recall. The best F-measure is reached with 50 patterns, requiring only 15sec. An almost equal F-measure, at different precision/recall levels, is achieved using only 30 patterns requiring only 9sec. Using only 10 patterns, which requires only 3sec, still achieves an F-measure of 0.45. Note that the increase in runtime is not linear in n as the length of the chosen patterns varies greatly.

One further idea we have not tested yet is to take the overall frequency of bi-grams into account to discriminate between common and less common (i.e., more pattern-specific) bi-grams. This could improve pattern selection by leading to a lower drop in recall.



PubMed query: ifn gamma signaling

Max. results: 10

- IL-4
- IL-6
- inducible NO synthase
- interferon-gamma
  - Proteins (15)
    - mRNA
    - IL-10
    - tumor necrosis factor-alpha
    - IL-6
    - STAT1
    - IL-18
    - SMAD3
    - SMAD4

Target: tumor necrosis factor-alpha [show all]

Relation: induction

Confidence: 0.8

Source Synonyms: ifn, ifn gamma, ifn-gamma, ifng, ifngamma, interferon gamma

Target Synonyms: tnf alpha, tnf-alpha, tnfalpha, tumor necrosis factor-alpha

PMID:16713974

We report that IFN-gamma augments induction of TNFalpha by TLR ligands, immune complexes, and zymosan by suppressing IL-10 production and thereby interrupting Stat3 -mediated feedback inhibition.

Fig. 4. AliBaba's user interface showing the results of a query for "ifn gamma signaling".

4. DEMO

In the demo, we shall show the workflow and algorithms of AliBaba and demonstrate all features of the user interface (see Fig. 4). In the graph panel, entities are displayed as nodes colored according to their type, and relationships are represented as arcs. The graph can be filtered by the type of objects, the in/out-degree of nodes, and the confidence of arcs. Complex graphs may be explored iteratively by "stepping" through objects and relationships. Since the confidence of the arc is given by the match score of the pattern extraction, users may thereby focus on only the most confident information.

In the text panel, all extracted information is additionally displayed using tree views with the different object types as root classes. The original abstract is also displayed (lower right panel). Highlighting of recognized objects and relationships is used to quickly provide the textual evidence to all extracted information.

4.1 Data Integration

AliBaba supports integration of all extracted results with additional knowledge sources on multiple levels. First, all extracted information may be stored either in an XML file or in a RDBMS for further processing. Secondly, recognized objects are linked to a couple of important type-specific external databases, such as disease or protein databases (middle right panel in Fig. 4).

Thirdly, we developed a tight integration of AliBaba with the KEGG database of biological pathways. A biological pathway can be thought of as a graph where nodes represent molecules and arcs represent chemical reactions. Pathway databases such as KEGG are manually curated and therefore prone to be incomplete and outdated; furthermore, they only include selected types of data into their pathways. Fig. 5 shows how information extracted with AliBaba and a pathway from KEGG can complement each other. AliBaba is able to open KEGG pathways and to display them in a layout that is similar to the original one (for easier orientation). Users may then add information extracted from PubMed queries. In the figure, first the WNT pathway was loaded from KEGG (Fig. 5, right side). Then, a query for “dickkopf” (a gene of Fruitflies) was performed and the results were analyzed by AliBaba and connected to the appropriate objects in the pathway. For instance, one can now see immediately that the dickkopf gene, and thus the entire pathway, is also related to Alzheimer’s disease (a red node). We believe that such an approach implements a very powerful and intuitive method for improving and enriching structured data with IE-based information.

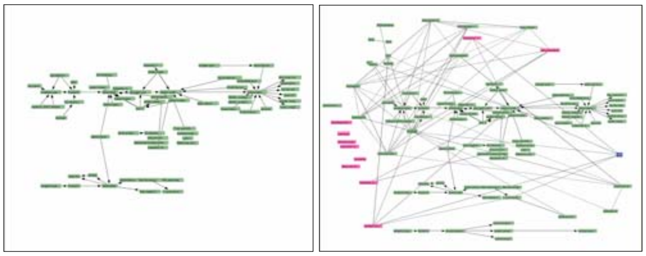


Fig. 5. Pathway overlay. Left: WNT pathway from KEGG in AliBaba. Right: Additional information extracted from a query (green: proteins; red: diseases).

Finally, AliBaba also makes use of external information to improve the search itself. A very common complication we have not mentioned yet are synonyms. Cox-2, for instance, is also known as PTGS2, GRIPGHS, PHS-2, “cyclooxygenase 2b”, “prostaglandin-endoperoxide synthase 2” etc. Finding all relevant abstracts usually requires searching for all names. AliBaba uses its integrated name dictionaries to alleviate this problem: Users may simply search for a UniProt ID, which will simultaneously search for all known synonyms of the protein.

5. DISCUSSION

We presented the AliBaba system for online information extraction from PubMed search results. AliBaba’s IE engine concentrates on the complex task of extracting relationships between objects. It is based on language patterns, which offers several advantages over regular expression patterns (RE), co-occurrence analysis (CO), or ML-approaches. (1) It has much higher precision than CO and RE. (2) It is much faster than ML-methods. (3) We can assign a meaningful confidence score to each extracted relationship derived from the match score. (4) The method can extract the actual type of relationships by analyzing the token mapping in the optimal alignment. (5) The ability of the system to adapt to changing requirements in terms of precision, recall, and runtime. Furthermore, our approach is domain-independent and may quickly be adapted to scenarios entirely different from biomedical IE. Drawbacks are a slightly lower recall and a higher complexity than RE/CO. However, we showed that an appropriately designed pattern filtering step improves the performance considerably, which in turn allows to use much larger patterns sets leading to an increased recall.

References

1. Altschul, S.F., et al., Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res, 1997. 25(17): p. 3389-402.
2. Chen, F., et al. Efficient Information Extraction over Evolving Text Data. in 24th International Conference on Data Engineering. 2008. Cancun, Mexico.
3. Cheng, T., X. Yan, and K.C.C. Chang. EntityRank: searching entities directly and holistically. in 33rd International Conference on Very Large Data Bases. 2007. Vienna, Austria.
4. Doan, A.H., R. Ramakrishnan, and S. Vaithyanathan. Managing information extraction: state of the art and research directions. in SIGMOD (Tutorial). 2006.
5. Gravano, L., et al. Approximate String Joins in a Database (Almost) for Free. in 7th Conference on Very Large Database Systems. 2001. Roma, Italy.
6. Gruhl, D., et al., How to build a WebFountain: An architecture for very large-scale text analytics. IBM Systems Journal, 2004. 43(1).
7. Hakenberg, J., et al., Gene mention normalization and interaction extraction with context models and sentence motifs. Genome Biol, 2008. 9 Suppl 2: p. S14.
8. Hao, Y., et al., Discovering patterns to extract protein-protein interactions from the literature: Part II. Bioinformatics, 2005. 21(15): p. 3294-300.
9. Jenssen, T.K., et al., A literature network of human genes for high-throughput analysis of gene expression. Nat Genet, 2001. 28(1): p. 21-8.
10. Myers, E. and R. Durbin, A Table-Driven, Full-Sensitivity Similarity Search Algorithm. Journal of Computational Biology, 2003. 10(2): p. 103-117.
11. Plake, C., et al., AliBaba: PubMed as a graph. Bioinformatics, 2006. 22(19): p. 2444-5.
12. Pyysalo, S., et al., Comparative analysis of five protein-protein interaction corpora. BMC Bioinformatics, 2008. 9 Suppl 3: p. S6.
13. Ramakrishnan, C., K.J. Kochut, and A.P. Sheth. A Framework for Schema-Driven Relationship Discovery from Unstructured Text. in Int. Semantic Web Conference. 2006.
14. Saric, J., et al., Extraction of regulatory gene/protein networks from Medline. Bioinformatics, 2006. 22(6): p. 645-50.