Akshaya: A Framework for Mining General Knowledge Semantics From Unstructured Text (Demo Paper)

Sumant Kulkarni*
International Institute of Information Technology Bangalore,
26/C, Electronics City, Bangalore India
sumant.k@iiitb.org

Srinath Srinivasa
International Institute of Information Technology Bangalore,
26/C, Electronics City, Bangalore India
sri@iiitb.org

Priyanaka Shukla
International Institute of Information Technology Bangalore,
26/C, Electronics City, Bangalore India
priyanka.shukla@iiitb.org

ABSTRACT

We report a tool called *Akshaya*, which implements a framework to mine four types of "general knowledge semantics" (analytical semantics) from unstructured text. The semantics being mined are - *semantic siblings, topical anchors, topic expansion* and *topical markers*. The framework provides options to embed more such general knowledge semantic mining algorithms into it. We use a term co-occurrence graph representation of unstructured text corpora to mine these semantics relations between terms. The semantic mining algorithms use different graph algorithms like random walk, graph clustering and so on to mine semantic relations. The tool can currently read plain text documents and generate a term co-occurrence graph and perform semantic association mining on it.

Keywords: Analytical Semantics, General Knowledge Semantics, Text Semantics, Text Mining, Semantic Siblings, Topical Anchors, Topic Expansion, Topical Markers

1. INTRODUCTION

Rachakonda et.al [6] have proposed methods to mine four forms of general knowledge semantic associations from co-occurrence graphs. Akshaya is a tool developed to build a comprehensive framework to mine these analytical semantics from a given unstructured text corpus. An example of an analytical semantics is a semantic relation like "is of type" which connects a given concept to its *type* concept. For example, given a concept like "car", and asked for its *type*, the semantic mining algorithm is expected to generate "vehicle" as output.

Mining general knowledge semantics has widespread applications in health care, patent management, judiciary and various other domains that deal with textual data. For example, it can be used in print media industry to link different documents using labeled semantic relations. We can link two documents discussing similar topics by a relation called "discusses similar topics". Similarly, we can cluster semantically similar documents using these approaches. We can use semantic mining to semi-automate the workflows in all such domains where extensive text data is available.

There have been many attempts to mine such analytical semantic relationships earlier. The major ones are [2, 3, 4, 5, 6, 7]. We have derived our inspiration to develop Akshaya from [2, 5, 6]. There are four semantic associations listed in [6], which form the basis for Akshaya. They are listed below.

Semantic Siblings is an algorithm which takes a set of sibling concepts as input and identifies other similar sibling concepts.

Topical Anchors is an algorithm, which takes a set of concepts as input and produces the concept which represents the topic of the input.

Topic Expansion generates a hypothetical context, which captures the predominant sense of the usage of the given input term in the corpus. It also separates out different contexts where the term assumes different meanings.

Topical Markers algorithm takes a concept as input and generates another term which uniquely and predominately identifies the input concept.

Table 1 gives hypothetical examples for the input and outputs for each semantic relationship mining algorithm. The detailed explanation of the algorithms can be found in [6].

Akshaya intends to mine these semantic relationships from unstructured text corpora. It converts the given unstructured text corpus into a term co-occurrence graph and runs these semantic mining algorithms for the given input terms. The semantic relationships between concepts are specific to the corpus being used. We currently use Wikipedia corpora of years 2006 and 2011 to mine these semantics. However, we can also use any other unstructured human generated corpus for the same.

*We thank Paras Mittal and Dipesh Joshi for their help in implementation.

The 20th International Conference on Management of Data (COMAD), 17th-19th Dec 2014 at Hyderabad, India.

Algorithm	Input	Output
Semantic Sibling	*Apple, Orange, Banana*	*Papaya, Mango, Pear, Sweet Lime*
Semantic Sibling	*Pele, Maradona, Messi, Ronaldo*	*Beckham, Zidane, Rooney, Kaka*
Topical Anchor	*Pele, Penalty Shootout, Offside*	*Football*
Topic Expansion	*Tennis*	*Nadal, Court, Wimbledon, Grand Slam, Open, Service, Ace, Volley*
Topical Marker	*Cricket*	*Googly*

Table 1: Example Input and Results of the Four Algorithms on a Hypothetical Text Corpus.

2. THE ALGORITHMS

In this section, we briefly describe each semantics association mining algorithm. The intention here is not to get into an in-depth understanding of the algorithms. Instead, we focus on understanding just the idea behind each algorithm. The algorithms have been discussed in detail in [6].

Semantic Siblings. The semantic siblings algorithm intends to identify the concepts which play the same role as the given concepts in given contexts. But the identified concepts are not synonyms of the given concepts. Table 1 shows two examples of semantic siblings. The core idea behind semantic sibling identification is "replaceability". A semantic sibling of given terms can replace them in predominant contexts common to the input, without distorting the meaning of those contexts. For example, in the statement - "Pele scored a goal", we can replace Pele with any of Beckham, Zidane, Rooney, Kaka. Hence, they are all semantic siblings of Pele. The semantic sibling algorithm attempts to identify the replaceability for the concepts in corpus. There are two approaches used to identify the replaceability. The details of the two algorithms are given in [6].

Topical Anchor. Topical anchor of a set of concepts represents the topic of the coherent context where the concepts appear predominantly. Table 1 gives a hypothetical example of a topical anchor. The topic is the concept whose probability of generation is the most, in the coherent context of the given concepts. In other words, topic is the most central concept to the conversation involving input terms. For example, if there is a conversation which mentions "glucose, blood sugar, insulin, hypoglycemia", then the term "diabetes" becomes the most expected (central) term in the conversation. Hence, it is the topic of the conversation. Authors use cash leaking random walk on the term co-occurrence graph to identify the topic of the given input concepts. There are three different varieties of the topical anchors algorithm. The details of the algorithms are given in [6].

Topic Expansion. Topic expansion is a process of expanding a given concept into different hypothetical contexts which represent the predominant usage of the concept in corpus. In text corpus, where polysemy exists, topic expansion generates many clusters for different senses of input term. For example, the term "Java" can have two hypothetical topic expansions as given below.

1. Java, Object Oriented Programming, Class, Object, Interface, Byte code, JVM, Compiler, ...
2. Java, Jakarta, Javanese, West Java, Central Java, East Java, Banten, Indonesia, Island, Yawadvipa, Population Density, ...

Topic expansion algorithm uses clustering based techniques on the term co-occurrence graph to generate such clusters of terms. There are two types of topic expansion algorithms. The details of the algorithms are given in [6, 2].

Topical Markers. Topical markers of an input concept are the concepts which are unique to the topic of input and are very unlikely to be related to other topics. For example, *double fault* is a topical marker of *tennis*. A term t_m which is well known and is very specifically used with another term t in the corpus becomes the topical marker of t. The authors use HITS algorithm [1] on the term co-occurrence graph to identify the topical markers. The co-occurrence graph is duplicated to create a bi-partite graph on which HITS algorithm is run. The details of the algorithms are given in [6].

All the above mentioned semantic association mining algorithms require a term co-occurrence graph representation of unstructured text corpus. They take concepts as input and generate other concepts which are connected by the mentioned association to the input terms.

3. THE TOOL

Akshaya is a library which allows us to perform following tasks.

1. Load data to create term co-occurrence graph.
2. Query the primitives of co-occurrence graph mentioned in [6].
3. Query for the documents containing a given term.
4. Query for the tf-idf scores of a given term for all documents where it is present.
5. Perform semantic association mining given a set of terms.

To perform the above mentioned tasks, Akshaya maintains two data structures – a term co-occurrence graph and an inverted index. We extract noun phrases from the text

using statistical noun phrase extraction techniques. The extracted noun phrases are used to build a co-occurrence graph as explained in [6]. The graph is used for querying co-occurrence primitives and also to perform semantic association mining. Akshaya maintains an inverted index for all the documents used to create co-occurrence graph. The inverted index is used to answer queries related to document retrieval and tf-idf scores. Akshaya currently supports four semantic association mining algorithms discussed earlier. It also allows addition of similar semantic association mining algorithms, which take a set of terms as input, work on the term co-occurrence graph and generate terms as output.

Akshaya is a ruby gem shipped with a command line interface. The Agama graph store [1] is used to store the co-occurrence graph. It uses SQLite [2] to store the inverted index. The Akshaya command line tool lets us run all semantics mining algorithms. Any algorithm bundled in Akshaya can be run using a generic command of the following form.

```
$ <algorithm-name> <options> <terms>
```

An example command to generate topic expansion results for the term "corpus" is given below.

```
$topicexpansion -s fast -d similar -c
wiki2006 corpus
```

The flag `-s fast` tells that it should be a fast execution and `-d similar` tells that the clusters generated can be similar. The flag `-c wiki2006` tells the algorithm to run topic expansion on Wikipedia 2006 corpus. The term `corpus` is the query term. Results of topic expansion on 2006 Wikipedia corpus for the term "corpus" are given below.

1. corpus, habeas corpus, eighth amendment, healthy americans act, theodore marley brooks, andrew blodgett mayfair, patricia savage, william, harper littlejohn, section 1983, thomas j. roberts

2. corpus, brown corpus, part-of-speech tagging, george kingsley zipf, empirical law, derose, ambiguous, native speaker, word sense disambiguation, parts of speech

3. corpus, hippocratic corpus, hippocratic cap-shaped bandage, medicine in ancient greece, project hippocrates, risus sardonicus, pyopneumothorax, the hippocrates project, 370 bc, clinical medicine

4. CONCLUSION AND FUTURE WORK

In this work, we have explained a generic semantic association mining framework called Akshaya. Akshaya is a ruby gem, which lets us mine different semantic associations like semantic siblings, topical anchors, topic expansion and topical markers. It uses term co-occurrence graph representation of the given corpus to mine semantics. It also gives the flexibility to add more such algorithms to the library. The future work includes adding other semantic mining algorithms like [3]. We also intend to develop this into a complete web application to perform end-to-end tasks – from corpus upload till semantic association mining.

[1] https://github.com/arrac/agama
[2] http://www.sqlite.org/

5. REFERENCES

[1] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

[2] S. Kulkarni, S. Srinivasa, and R. Arora. Cognitive modeling for topic expansion. In *On the Move to Meaningful Internet Systems: OTM 2013 Conferences*, pages 703–710. Springer, 2013.

[3] S. Kulkarni, S. Srinivasa, J. N. Khasnabish, K. Nagal, and S. G. Kurdagi. Sortinghat: A framework for deep matching between classes of entities. In *Data Engineering Workshops (ICDEW), 2014 IEEE 30th International Conference on*, pages 90–93. IEEE, 2014.

[4] R. Navigli and M. Lapata. An experimental study of graph connectivity for unsupervised word sense disambiguation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(4):678–692, 2010.

[5] A. R. Rachakonda and S. Srinivasa. Finding the topical anchors of a context using lexical cooccurrence data. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 1741–1744. ACM, 2009.

[6] A. R. Rachakonda, S. Srinivasa, S. Kulkarni, and M. Srinivasan. A generic framework and methodology for extracting semantics from co-occurrences. *Data & Knowledge Engineering*, 2014.

[7] B. Wei, J. Liu, J. Ma, Q. Zheng, W. Zhang, and B. Feng. Motif-re: motif-based hypernym/hyponym relation extraction from wikipedia links. In *Neural Information Processing*, pages 610–619. Springer, 2012.