GrASP: Rich Patterns for Argumentation Mining

Eyal Shnarch **Ran Levy*** **Vikas Raykar‡** **Noam Slonim**
IBM Research - Haifa, Israel
‡IBM Research - Bangalore, India
{eyals, ranl, noams}@il.ibm.com viraykar@in.ibm.com

Abstract

We present the GrASP algorithm for automatically extracting patterns that characterize subtle linguistic phenomena. To that end, GrASP augments each term of input text with multiple layers of linguistic information. These different facets of the text terms are systematically combined to reveal rich patterns. We report highly promising experimental results in several challenging text analysis tasks within the field of Argumentation Mining. We believe that GrASP is general enough to be useful for other domains too.

1 Introduction

Many standard text analysis tasks can be addressed relatively well while exploiting simple textual features, e.g., Bag-Of-Words representation and Naive Bayes for document classification (McCallum and Nigam, 1998). However, the identification of more subtle linguistic phenomena, that are further reflected via relatively short texts – as opposed to whole documents – may require a wider spectrum of linguistic features.

The main contribution of this work is in outlining a simple method to automatically extract rich linguistic features in the form of patterns, and demonstrate their utility on tasks related to *Argumentation Mining* (Mochales Palau and Moens, 2009), although we believe that the proposed approach is not limited to this domain.

Argumentation mining involves automatically identifying argumentative structures within a corpus – e.g., claims or conclusions, and evidence instances or premises – as well as their interrelations. For instance, each of the following sentences includes a *claim* for a **[topic]**.

1. Opponents often argue that *the open primary is unconstitutional.* **[Open Primaries]**
2. Prof. Smith suggested that *affirmative action devalues the accomplishments of the chosen*. **[Affirmative Action]**
3. The majority stated that *the First Amendment does not guarantee the right to offend others*. **[Freedom of Speech]**

These sentences share almost no words in common, however, they are similar at a more abstract level. A human observer may notice the following underlying common structure, or *pattern*: `[someone][argue/suggest/state][that][topic term][sentiment term]`

We present GrASP, standing for GReedy Augmented Sequential Patterns, an algorithm that aims to automatically capture such underlying structures of the given data. Table 1 shows the pattern that GrASP may find for the above examples, along with its matches in those texts. Such patterns can then be used to detect the existence of the phenomenon in new texts.

The algorithm starts with augmenting the terms of the input with various layers of *attributes*, such as hypernyms from WordNet (Fellbaum, 1998), named entity types, and domain knowledge (Section 3.1). This multi-layered representation enables GrASP to consider many facets of each term. Next, it finds the most indicative attributes (Section 3.2) and iteratively grows patterns by blending information from different attributes (Section 3.3). A greedy step is performed at the end of each iteration, when the algorithm only keeps the top k patterns, ranked by their predictive power. This results with a set of cross-layered patterns whose match in a given text instance suggests the appearance, or the non-appearance, of the target phenomenon.

Researchers can add layers of attributes of different kinds without being worried about which of

*First two authors contributed equally.

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 1345–1350
Copenhagen, Denmark, September 7–11, 2017.

[noun]		[express]	[that]		[noun,topic]		[sentiment]	
Opponents	often	argue	that	the	open primary	is	unconstitutional	.
Prof. Smith		suggested	that		affirmative action		devalues	the ...
The majority		stated	that	the	First Amendment	does not guarantee ... to	offend	others.

Table 1: Claim sentences aligned by their common underlying pattern. [express] stands for all its (in)direct hyponyms, and [noun,topic] means a noun which is also related to the topic.

them are useful for the detection of the target phenomenon, and how to combine them. GrASP performs feature selection while generating complex patterns out of these attributes that best capture aspects of the target phenomenon.[1]

In experiments over different argumentation mining tasks, we show that GrASP outperforms classical techniques, and boosts full argumentation mining systems when added to them.

2 Background

While some aspects of GrASP were considered in the past, to the best of our knowledge, no prior work has presented a framework that allows users to: (i) easily add any type of attribute to the pattern alphabet, and (ii) consider **all** attributes when searching for patterns. GrASP provides a framework to integrate information from different layers, choosing the best combination to produce highly expressive patterns.

The alphabet of Hearst (1992) patterns is mainly stop words and noun-phrase tags, while Snow et al. (2004) add syntactic relations. Yangarber et al. (2000) consider a larger set of attributes (e.g., named entities, numeric expressions), however they commit to one generalization of each term. In contrast, we do not limit our alphabet and systematically consider all attributes of each term.

Riloff and Wiebe (2003) start with a small set of syntactic templates, composed of a single syntactic relation and a single POS tag, to learn a variety of lexicalized patterns that match these templates. RAPIER (Califf and Mooney, 2003) constraints are similar to our attributes, but are basic (surface form, POS tag, and hypernyms only), and expanding them will exponentially increase its complexity. In contrast, adding attributes to GrASP only increment runtime linearly (see Section 3.2).

To summarize, prior works usually have a basic alphabet and commit to one rule to generalize each term. Commonly, they do not allow gaps between their elements, nor assigning several attributes to a single element of the pattern.

Such characteristics are presented in *sequential patterns* (Agrawal and Srikant, 1995) which are mainly used for data mining and rarely for unstructured text (Jindal and Liu, 2006). GrASP also has these characteristics, and in addition it can learn *negative* patterns, indicating that the examined text *does not* contain the target phenomenon.

The phenomena we target are from the area of Argumentation Mining (see Lippi and Torroni (2016) for a survey). We focus on open-domain argument extraction. In this context, Levy et al. (2014) detect claims relevant to a debatable topic, Lippi and Torroni (2015) defined the context-*independent* claim detection task, and Rinott et al. (2015) introduced the context dependent evidence detection task (which is further split into different types of evidence, e.g., a *study* that supports a claim or a relevant *expert* testimony). These tasks aim to capture a subtle and rare linguistic phenomenon within large corpora, hence are suitable for demonstrating the potential of GrASP.

3 The GrASP Algorithm

The algorithm depicted in Algorithm 1. Its input is a set of positive and negative examples for the target phenomenon. The output is a ranked list of patterns, aiming to indicate the presence – or absence – of this phenomenon. In the following, a pattern is considered to be *matched* in a text iff all its elements are found in it, in the specified order, possibly with gaps between them, within a window of size w.

3.1 Multi-Layered Term Representations

Consider the verbs (argue/suggest/state) in the examples in Section 1. Using the POS tag *verb* to generalize them will end up with an overly general representation, while their hypernym, *express*, offers a better level of generalization.

Aiming to formalize this intuition, we start by augmenting each term in the input with a variety of linguistic *attributes* such as its POS tag, its syntac-

[1] Get GrASP cloud service at http://ibm.biz/graspULP

Algorithm 1: The GrASP algorithm.

Input: positive/negative text examples, k_1, k_2, *maxLen*
Output: a ranked list of patterns

```
1  (pos, neg) ← augment(positives, negatives)
2  attributes ← extractAttributes(pos, neg)
3  alphabet ← chooseTopK(attributes, k1)
4  patterns ← alphabet
5  last ← patterns
6  for length ← 2 to maxLen do
7      curr ← θ
8      for p ∈ last do
9          for a ∈ alphabet do
10             curr ← curr ∪ {growRight(p, a)}
11             curr ← curr ∪ {growInside(p, a)}
12     last ← curr
13     patterns ←
         chooseTopK(patterns ∪ current, k2)
14 return patterns
```

tic relation in a parse tree, and semantic attributes such as its hypernyms, WordNet superclasses, indications whether it is a named entity, and whether it bears a sentiment.[2] This attributes set can serve as a starting point for many text analysis tasks.

In addition, GrASP allows to add task-specific attributes. Thus, for context-dependent arguments detection we add boolean attributes indicating whether the term is related to the topic, whether it appears in a lexicon characterizing argumentative texts, or in a lexicon characterizing the topic.[3]

After augmentation, the representation of *argue*, from the first example, is: [`argue, VB, hypernyms={present, state, express}, in claim lexicon, root node, supeclass = communication`].

3.2 Defining the Patterns Alphabet

The augmented representation, described above, is the first step (line 1 in Algorithm 1). Next, we define the alphabet of attributes that will be used to compose longer patterns (lines 2–3). To that end, we first discard non-frequent attributes that are matched in less than t_1 of all input examples.

Then, we sort all remaining attributes by their information gain (Mitchell, 1997) with the label, and select the top k_1 attributes. We discard redundant attributes whose correlation to some previously selected attribute is above t_2, measured by the normalized mutual information (Cover and Thomas, 2006). The selected k_1 attributes constitute the *alphabet* of the algorithm, or "patterns" of length 1. Note that considering additional attributes only affects this first iteration, and only increases it linearly.

3.3 Growing Patterns

Learning longer patterns is done by iteratively growing patterns selected in previous iterations, keeping only the most indicative ones (lines 6–13). We apply two methods for growing a pattern, p (e.g., [`noun`]) w.r.t. an attribute a (e.g., *obj*): (i) grow right – add a as another term in the pattern (i.e., [`noun`][`obj`]); (ii) grow inside – add a as another attribute to the last term of p, making it more specific (i.e., [`noun,obj`]). After each iteration (line 13), the top k_2 patterns are kept (after sorting by information gain and discarding redundant ones). Iterations continue till reaching *maxLen*.

Since GrASP relies on information gain for sorting, it can identify indicative *negative* patterns, implying that the target phenomenon is *less likely* to be presented in the examined example if such patterns were matched in it.

GrASP can be seen as a simple formal interface, allowing the user to examine a wide range of information sources letting the algorithm to select and combine them all and come up with the most useful patterns.

4 Evaluation and Results

In the following experiments we used a logistic regression classifier on top of the extracted patterns. Each pattern is used as a binary feature, which receives value of 1 iff it is matched in the candidate.

To demonstrate the robustness of the this approach, in all experiments we report the results of a single configuration of GrASP parameters, selected based on a quick analysis over a small portion of the claim-sentence detection data (task (a) below).[4] Specifically, we used minimal frequency threshold $t_1 = 0.005$, correlation threshold $t_2 = 0.5$, size of the alphabet $k_1 = 100$, number of patterns in the output $k_2 = 100$, maximal pattern length *maxLen* $= 5$, and window size $w = 10$.

This configuration is by no means the optimal one, and we saw that by carefully tuning the parameters per task, results were improved.

[2] We used OpneNLP POS tagger, Stanford NER, McCord and Bernth (2010) parser, WordNet superclasses, and the lexicon in Hu and Liu (2004) for sentiment words.

[3] We utilize existing lexicons, learning them is out of the scope of this work.

[4] We randomly chose 10 topics. The performance over them was somewhat inferior to that over all 58 topics.

system	(a) Claim sentence P@5	P@10	P@20	(b) Expert evidence P@5	P@10	P@20	(c) Study evidence P@5	P@10	P@20
Naive Bayes	13.8	10.2	8.1	8.4	9.6	8.0	13.1	11.6	9.3
Basic patterns	21.4	15.5	12.7	16.5	15.2	12.2	18.3	16.1	12.8
CNN	25.5	21.2	16.9	18.2	16.3	14.6	26.5	22.2	18.4
GrASP alphabet	30.0	25.7	22.8	25.8	22.5	18.7	30.5	25.6	**21.1**
GrASP	**41.7****	**34.5****	**27.0****	**29.0***	**25.2***	**21.9***	**35.4***	**25.7**	20.0

Table 2: Macro-averaged precision results for GrASP over three argumentation mining tasks. Significant results in comparison to GrASP alphabet/CNN are marked with **/* respectively (paired t-test with $p<0.01/0.02$ respectively).

4.1 Direct Evaluation

We consider three context-dependent argumentation mining tasks: (a) Claim sentence detection (Levy et al., 2014), (b) Expert evidence detection, and (c) Study evidence detection. The latter two tasks are described in Rinott et al. (2015), where the goal is to detect sentences that can be used as an evidence to support/contest the topic.[5]

The benchmark data for these tasks was extracted from the data released by Rinott et al. (2015), consisting of 547 Wikipedia articles in which claims and evidence instances were manually annotated, in the context of 58 debatable topics. In all tasks the data is highly skewed towards negative examples (only 2.5% of 80.5K instances are positives in task (a), 4% of 55.6K in task (b), and 3.7% of 31.8K in task (c)), making these tasks especially challenging.

As (Levy et al., 2014; Rinott et al., 2015) we use a leave-one-topic-out schema; training over 57 topics, testing over the left out topic.

Our group develops debate supportive technologies which can assist humans to reason, make decisions, or persuade others.[6] Since in this scenario humans mainly consider top results (similar to information retrieval), precision is more relevant than recall. Thus, we report the macro-averaged Precision@K, where $K \in \{5, 10, 20\}$.

We considered the following baselines:

Naive Bayes: over BOW representation, discarding unigrams which appear less than 10 times.

Basic Patterns: a baseline that reflects common practices in the literature where a pattern is a consecutive ordered list of stop words or POS tags. We add a symbol for topic match (for the context-dependent tasks). A brute force process generates all possible patterns up to size *maxLen* and selects top k by the same procedure as GrASP. For each task we report the best results obtained with $k \in \{50, 100, .., 400\}$.

Convolutional Neural Network (CNN): following (Kim, 2014; Vinyals and Le, 2015) we used CNN whose input is a concatenation of the topic and the candidate.[7] The final state vector is fed to a LR soft-max layer. Cross-entropy loss function was used for training. The embedding layer was initialized using word2vec vectors (Mikolov et al., 2013). Hyper-parameters were tuned on the same portion of the dataset as used by GrASP for tuning.

For these baselines, we are not aware of available methods to incorporate GrASP multi-layered representation.

GrASP alphabet: a simplified version of GrASP which uses the chosen alphabet, or "patterns" of length 1. This baseline does utilize all the information available for GrASP.

Table 2 shows that *Naive Bayes* performance is the lowest, demonstrating that a simple representation is not sufficient for such complex tasks. Using *Basic patterns* yields better performance, and *CNN* performs even better. *GrASP alphabet* outperforms CNN, indicating the potential of explicitly incorporating linguistic information. Finally, using the patterns extracted by *GrASP* outperforms all other methods, emphasizing the added value of constructing patterns over the initial contribution of the multi-layered representation.

GrASP provides an easy way to analyze the importance of each attribute by inspecting its score at the end of the first iteration, the one which determines the alphabet. For example, *PERCENT* score was very high in the alphabet for Study evidence patterns (task b), and *Person* and *Organization* were ranked high in the alphabet of the Expert evidence (task c). Still, these three named enti-

[5] We did not examine the Anecdotal type due to the small size of the available benchmark data.

[6] for more details see IBM Debating Technologies.

[7] RNN, LSTM (Hochreiter and Schmidhuber, 1997) and GRU (Cho et al., 2014) were also considered but resulted with inferior performance.

system	P@5	P@10	P@20	P@50	R@50
Levy14	–	–	–	18	40
Levy14-rep	30.9	27.3	23.5	17.6	38.4
GrASP	32.7	30	23.9	17.5	36.2
Combined	**40***	**32.4***	**28***	**20.2***	**43.2***

Table 3: Adding GrASP to a full claim detection system. Significant results in comparison to Levy14-rep are marked by * (paired t-test with $p < 0.02$).

ties were not selected as part of the alphabet for Claim sentences (task a) – reflecting the importance of *PERCENT* in sentences describing studies and their numeric results, and the importance of authoritative source (either *Person* or *Organization*) in evidence based on expert testimonies.

For task (a) we performed two ablation tests, each of them yielded a decrease in performance: (i) not limiting the match of a pattern in a window (a decrease of 10.3 for P@5 and 2.8 for P@20), and (ii) not enforcing the order defined by the pattern (a decrease of 7.6 for P@5 and 2.8 for P@20).

4.2 Indirect Evaluation

In this evaluation we add GrASP patterns as additional features to the full claim detection system of Levy et al. (2014) to inspect their contribution. This evaluation is performed on a second claim detection benchmark (on which they have reported results), released by Aharoni et al. (2014) (1,387 annotated claims associated with 33 topics).

The system of Levy et al. (2014) is comprised of a cascade of three components; (i) detecting sentences which contain claims, (ii) identifying the exact boundaries of the claim part within the sentence, and (iii) ranking the claim candidates. Each of these components applies a classifier over dedicated features. Results were reported for the full cascade and for the first component, which is our task (a). For an idea on how to adapt GrASP for the claim boundaries detection task, see Section 5.

Table 3 presents measures reported in Levy et al. (2014) (right hand side) as well as additional measures which reflect the focus of this work on the precision of the top ranked candidates (performance of all systems on P@200 and R@200 were comparable and were omitted due to space limitations). The system of Levy et al. (2014), denoted *Levy14*, and our reproduction of it, denoted *Levy14-rep*, obtained comparable results.[8]

[8]We reproduced their work to perform significant test and report the additional measures.

Evidently, utilizing GrASP patterns alone achieve similar performance as Levy14-rep. Considering the fact that Levy14-rep is a full system, tailored for claim detection via a lengthy feature engineering process, these results, obtained using only GrASP patterns, are promising. Adding GrASP features to Levy14-rep, denoted *Combined*, we observe a significant improvement, demonstrating their complementary value.

5 Discussion

GrASP extracts rich patterns that characterize subtle linguistic phenomena. It exploits a wide variety of information layers in a unified manner, identifying the most discriminative attributes for the given task, and greedily composes them into patterns. We demonstrated GrASP significant impact on several argumentation mining tasks.

As this was not the focus of this work, we chose standard statistical criteria to sort the candidate patterns and to filter redundant ones. Considering other criteria, and also more sophisticated search strategies to explore the huge space of possible patterns, is left for future work.

In addition to their value in classification tasks, the patterns revealed by GrASP are easy to interpret, in contrast to alternative techniques, like Deep Learning. Thus, these patterns can provide researchers with additional insights regarding the target phenomenon. These insights can be integrated back to by considering additional attributes to be explored in subsequent runs. Thus, GrASP can significantly expedite the research process, especially when addressing novel tasks.

Finally, we would like to hint on a sequel work that demonstrates how GrASP can be easily modified to address another important task – detecting the claim boundaries within its surrounding sentence (see italic text in the examples in Section 1). To cope with this unique task, we enhance the term representation (Section 3.1), by tripling each attribute a to distinguish between its appearance before (`PRE-a`), within (`IN-a`), or after (`POST-a`) the candidate claim boundaries. With this change only, GrASP was able to identify patterns for this new task, that were used to indicate the boundaries of a claim with promising preliminary results.

Acknowledgments

The authors thank Matan Ninio for giving the algorithm its catchy name.

References

Rakesh Agrawal and Ramakrishnan Srikant. 1995. Mining sequential patterns. In *International Conference on Data Engineering*, pages 3–14.

Ehud Aharoni, Anatoly Polnarov, Tamar Lavee, Daniel Hershcovich, Ran Levy, Ruty Rinott, Dan Gutfreund, and Noam Slonim. 2014. A benchmark dataset for automatic detection of claims and evidence in the context of controversial topics. In *First Workshop on Argumentation Mining*, pages 64–68.

Mary Elaine Califf and Raymond J. Mooney. 2003. Bottom-up relational learning of pattern matching rules for information extraction. *Journal of Machine Learning Research*, pages 177–210.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *EMNLP*, pages 1724–1734.

Thomas M. Cover and Joy A. Thomas. 2006. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, Massachusetts.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING*, pages 539–545.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 168–177.

Nitin Jindal and Bing Liu. 2006. Mining comparative sentences and relations. In *AAAI*, pages 1331–1336.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*, pages 1746–1751.

Ran Levy, Yonatan Bilu, Daniel Hershcovich, Ehud Aharoni, and Noam Slonim. 2014. Context dependent claim detection. In *COLING*, pages 1489–1500, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Marco Lippi and Paolo Torroni. 2015. Context-independent claim detection for argument mining. In *IJCAI*, pages 185–191.

Marco Lippi and Paolo Torroni. 2016. Argumentation mining: State of the art and emerging trends. *ACM Transactions on Internet Technology (TOIT)*, 16(2):10.

Andrew McCallum and Kamal Nigam. 1998. A comparison of event models for naive bayes text classification. In *AAAI workshop on learning for text categorization*, pages 41–48. AAAI Press.

Michael C McCord and Arendse Bernth. 2010. Using slot grammar. *IBM TJ Watson Res. Center, Yorktown Heights, NY, IBM Res. Rep. RC23978*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119.

Thomas M. Mitchell. 1997. *Machine Learning*. McGraw-Hill, Inc.

Raquel Mochales Palau and Marie-Francine Moens. 2009. Argumentation mining: the detection, classification and structure of arguments in text. In *Artificial Intelligence and Law*, pages 98–109.

Ellen Riloff and Janyce Wiebe. 2003. Learning extraction patterns for subjective expressions. In *EMNLP*, pages 105–112.

Ruty Rinott, Lena Dankin, Carlos Alzate Perez, Mitesh M. Khapra, Ehud Aharoni, and Noam Slonim. 2015. Show me your evidence - an automatic method for context dependent evidence detection. In *EMNLP*, pages 440–450, Lisbon, Portugal. Association for Computational Linguistics.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2004. Learning syntactic patterns for automatic hypernym discovery. In *NIPS*.

Oriol Vinyals and Quoc V. Le. 2015. A neural conversational model. *CoRR*, abs/1506.05869.

Roman Yangarber, Ralph Grishman, and Pasi Tapanainen. 2000. Unsupervised discovery of scenario-level patterns for information extraction. In *ANLP-NAACL*.