Preliminary report / Prethodno priopćenje
Manuscript received: 2016-04-25
Revised: 2016-05-29
Accepted: 2016-05-31
Pages: 43 - 51

An Application of Fuzzy Inductive Logic Programming for Textual Entailment and Value Mining

Sandro Skansi
IN2data
Data Science Company Ltd.,
Zagreb, Croatia
sandro.skansi@in2data.eu

Branimir Dropuljić
IN2data
Data Science Company Ltd.,
Zagreb, Croatia
branimir.dropuljic@in2data.eu

Robert Kopal
IN2data
Data Science Company Ltd.,
Zagreb, Croatia
robert.kopal@in2data.eu

Abstract: The aim of this preliminary report is to give an overview of textual entailment in natural language processing (NLP), to present our approach to research and to explain the possible applications for such a system. Our system presupposes several modules, namely the sentiment analysis module, the anaphora resolution module, the named entity recognition module and the relationship extraction module. State-of-the-art modules will be used but no amount of research will go into this. The research focuses on the main module that extracts background knowledge from the extracted relationships via resolution and inverse resolution (inductive logic programming). The last part focuses on possible economic applications of our research.

Keywords: natural language processing, value mining, textual entailment, inductive logic programming

INTRODUCTION

Natural language processing is an old part of artificial intelligence (AI), which is the oldest and most prolific field of computer science. Natural language processing, as a goal for AI has been stressed at the Dartmouth conference in 1956 [18], and received a substantial boost in the Cold War in an effort to make feasible machine translation systems. AI during this period was catering to NLP. These were mainly crisp rule based systems with obvious shortcomings. By the early 1980's, machine learning (ML) received a new boost and applications to NLP were abundant [9]. During the 1990's, a related field, information retrieval (IR) came of age with the advent of search engines.

NLP today relies heavily on both ML and IR. The advent of user-friendly computing (where there is no requirement on precision placed on the user), advocated mainly by the tech giants brought new funding to NLP, and various new areas surfaced.

Big Data offered a major paradigm shift. As Halevy, Norvig and Pereira underlined [3], today much more focus is placed on data than on algorithms. Although the author begs to differ, the fundamental idea holds: today's algorithms need to be first and foremost scalable, and then, only at a distant second place, they need to be precise. This is best illustrated by the stochastic gradient descent example, where it is better (in terms of overall performance) to have a quick and suboptimal method for finding local minima, and reiterate this process with random initial parameters for thousands of times, than to have a slow but excellent method for finding the global minimum. This change in paradigm only boosted NLP.

Simpler and quicker algorithms and architectures were developed, which fed on data rather than being made by a skilled programmer. An excellent example is the now widely used sentiment analysis (SA) algorithm[1].

The algorithm first takes the text, parses it to a bag or even set, discards all the stopwords (sometimes this is done by length), uses an optional stemmer and runs a Naïve Bayes classifier on it, or a linear regression if there is a need for a fine grained classification. The most common dataset used is the Movies Reviews and Amazon reviews, where the stars are mapped to votes (-2,-1,0,1,2) or in case of a binary classification, 1 and 2 stars are mapped to -1, and 4 and 5 stars to 1, and 3 star reviews are discarded.

Figure 1: *Amazon comment example.*

[1] We use a singular voice, although there are many algorithms for SA, but this approach seems to be the most commonly used with excellent results.

As an example take the review shown on Figure 1: this review can be trimmed to a list by simply using the following python code:

```
output=[x for x in set(initialText.lower().split()) if len(x)>2]
```

Which produces the following output (combined with the 4 star translation in integers, which depending on the information source might require quite sophisticated computer vision algorithms):

```
[[‘and’, ‘old’, ‘amateurish.’, ‘deathstar’, ‘character’, ‘some’, ‘second’, ‘mask.’, ‘have’, ‘chewbaca’, ‘fun.’, ‘seem’, ‘seemed’, ‘humor’, ‘needed’, ‘telegraphed’, ‘movie.’, ‘should’, ‘better’, ‘acting’, ‘production’, ‘too’, ‘way’, ‘new’, ‘was’, ‘good.’, ‘nice’, ‘never’, ‘1/2’, ‘than’, »didn’t«, ‘solo’, ‘welcome’, ‘retread’, ‘ford’, ‘but’, ‘characters’, ‘half’, ‘fisher’, ‘removed’, ‘with’, ‘great.’, ‘scene’, ‘like’, ‘return.’, ‘back.’, ‘stars.’, ‘ren’, ‘1-3.’, ‘were’, ‘fresh’, ‘the’, ‘having’, ‘first’],1]
```

The idea behind it is that there will be enough information to learn the relevant information about sentiment in general even though it is genre specific (movies, books). E. g. by offering a »bad movie« and a »good movie« review, the classifier learns to ignore »movie« and just use the »good« and »bad«, so the classifier trained on these datasets generalizes well despite it being trained on a particular topic. In this way it is possible to efficiently process a large number of reviews. The same is true for most fragmentary NLP tasks.

The topic of textual entailment is different. The problem is the explosive growth of the searchspace when considering background premises. The first formulation of this task was using rule-based systems of the 1970’s, and a unified account was given by Norvig in 1986 [8]. These early systems did not work well due to the limited technology of the time, and in no small part, due to the fact that probabilistic approaches where only rudimentary for NLP (Norvig was one of the pioneers, as well as Manning, who afterwards systematized the field in [5]). The revival came in 2004 with the paper [2].

Norvig in his debate with Chomsky [15] points out that NLP is inherently probabilistic. But we beg to differ. A probabilistic model gives a probability, but then it collapses to right/wrong when the event happens. E. g. the proposition »It will rain tomorrow« with a probability of 90% is probabilistic. But the proposition »By »sometimes« people mean X« which is true in 90% of the cases is not probabilistic but fuzzy, since there is no future state that will collapse the 90% to a »True« or »False«, as there is in the probabilistic case. This might seem like a minor distinction, but it is an important one, since probability tends to wait and revise, while fuzzy sets try to work with the information at hand, and methodology diverges considerably.

Despite the differences, they both improve drastically on computational speed, since it can be precisely formulated in which order to search, in terms of trimming the search tree.

This brings us back to the problem of the exponential increase in size of the search space. The list for SA shown above has the text and a number indicating the sentiment. If instead of the sentiment we search for all hidden premises by ML, we would need to have a set of all possible premises at hand, and append all their possible combinations

to the text (as features), and tag the result with 1 or 0. Even for moderate dataset of 10000 Facebook comments and 100 premises, this would result in more than a googolplex (billions of times more than the number of subatomic particles in the universe).
This means that a different approach must be undertaken. The revival of textual entailment was slow, and the first tries were very modest. I came in the form of *recognizing* textual entailment, where the algorithm took two texts as a tuple, and the outputted a simple yes/no according to whether the first test entailed the second. A quick addition was the degree of confidence, so the Boolean output was recast in probability. These algorithms used primarily the similarity of the two texts. E.g.: »The guy sitting in the White House is great« and »Obama rules FTW« would get a 0% of entailment, even though they are almost equivalent statements. Even stemming and string distance does not offer much in the way of optimization.
This sets the stage for our research. We want to find the hypotheses needed for one text to entail the other, but we focus on a subproblem: what does a text need to imply a proposition. A proposition is the meaning of a sentence, so e.g. »The snow is white« and »La neve e' bianca« share the same proposition. In our case, for each topic subject we have two propositions. The debate starts with a question, and the answers to the question form the two propositions. E. g. if the question of the discussion is »Is Obama a good president?« the possible answers »Obama is a good president« and »Obama is not a good president«.

FORMATTING THE DATA

In this section we explain how our module will receive data. Since this is report on the research in progress for our module, this section will constitute the core part of this paper and explain how to format data for our textual entailment module.
In the context of our textual entailment algorithm all of this counts as preprocessing, although this is not text preprocessing in the usual sense, since it encompasses many advanced algorithms. The overall process can be found in the picture below.
We will explain the modules in more detail. The raw text in the chart refers to text that has been collected and segmented in individual comments with an unused delimiter (e.g. | might be a good choice since it seldom appears in natural language texts). But asides from this we assume no preprocessing in the raw text part. Since the design of the parsing module is one of the most challenging tasks, and depends on the needs of other modules, we will address this module last.
For named entity recognition one of the standard approaches today is via conditional random field sequence models (cf. [11]) and we will use the Stanford NER software[2].
For sentiment analysis a simple approach as the one described above would serve our purpose well. Top performing algorithms for anaphora resolution can be found in [6], and the Lappin and Leass algorithm for pronominal anaphora resolution is still widely used [4]. Pronominal anaphora resolution is the most common type of anaphora resolution, resolving pronouns, and it is the one we need.

[2] available via GNU license from http://nlp.stanford.edu/software/CRF-NER.shtml

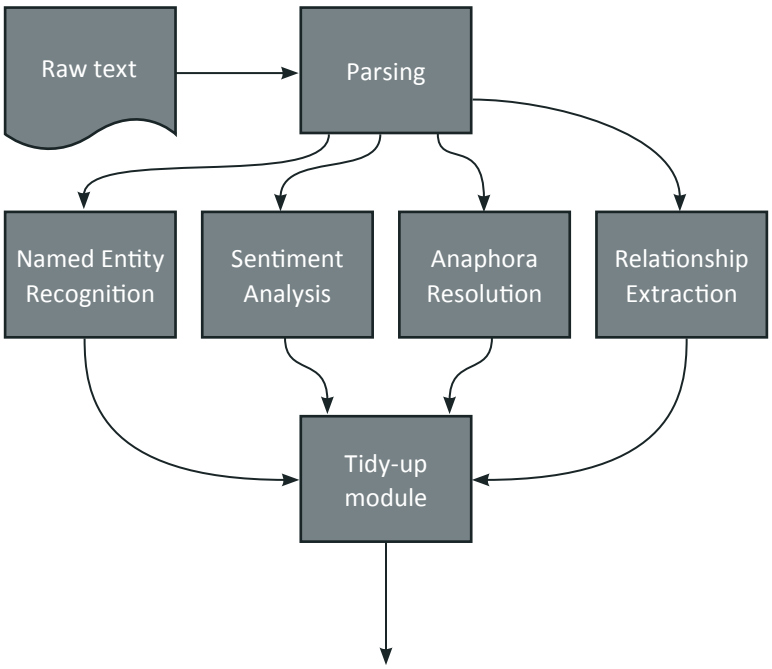

Figure 2: *System architecture.*

Relationship extraction is technically a part of IR, and not of NLP, and this means it received quite a bit more attention, and the most influential software is IBM Watson [17]. The main results in relationship extraction today are based on kernel methods for relationship extraction first proposed in [13]. Modern approaches build on this and emphasize hybrid methods [10]. Most of the system used are used for one-off relationship extraction, but there has been a number of interesting papers dealing with automatic evaluation (and reevaluation) of industrial relation extraction systems, and we single out the excellent paper from Bronzi, Guo, Mesquita, Barbosa and Merialdo [1].

The aim of our research is not to produce state of the art production software but to make a European Commission Technology Readiness Level 5 (TRL 5) [16], since the current level for the integrated system is TRL2. As such, we do not optimize individual modules, and leave this open for further research.

One last thing that needs to be explicated. We will not train the different modules with our raw text data, but on separate datasets designed to enable optimal training of the separate modules, since they will offer superior performance with the modules for our purpose.

PROTOTYPE TIDY-UP MODULE

Our main manifold module is nicknamed the Tidy-up module. The Tidy-up module is a helper module that connects the other modules. The purpose of this module is to take the results from the previous modules and the raw text and reformat the information

in extended relations. E.g. from: »The character of Ren was needed but he should never have removed the mask«, the modules extract a positive sentiment, »Ren« and »mask« as named entities, »Ren« is substituted in place of »he« and a simple calculus (based on logical equivalence) yields **For(Ren)** and **For(RenWithMask)**. Here we go in deep granularity with the second statement, but this kind of granularity is not needed.
The Tidy-up module is the main focus of current development and the biggest challenge in our research. The worst case scenario is that we need to employ a brute forcing ML approach, but we hope to get a more precise module. The finished Tidy-up module will take as input the information from the other modules and return the local logical representation of the key values and propositions expressed in the text.

TEXTUAL ENTAILMENT MODULE

Our main module receives as input relations. We will need a cleaner example. Suppose the starting sentence is that speaker **S** utters is *»I don't like Obama because he doesn't like guns«* is to be reduced to **For$_s$(Guns)** by the Tidy-up module.
We will describe the textual entailment module in detail. The basic module contains a set of axioms, the extracted statement and the topic subject PRO/CON as a goal (Image 3) to be interpreted as a literal in the engine (we use resolution and inductive logic programming [7]):

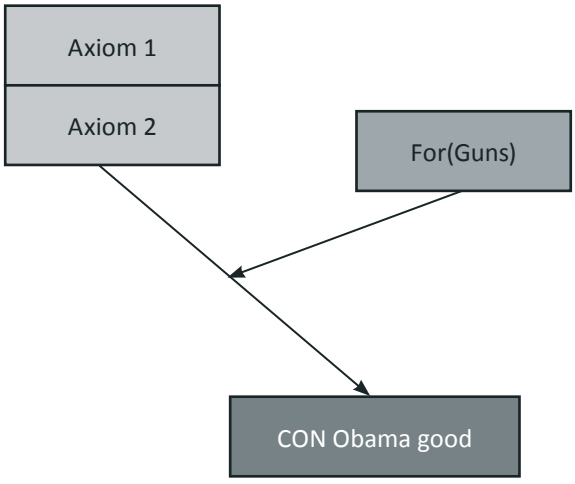

Figure 3: *Inductive Logic Programming, general setting.*

This is the ideal case. The problem is that the set of axioms might not be complete. We have two possible strategies here: (i) we could augment the axioms with a new axiom (of the form **P=>Q**), or (ii) we could add a reason (of the form **P**), which will be used as an auxiliary statement.
The main problem with (i) is that a new axiom must connect **P** and **Q**, and this connection is either trivial (in the sense that it connects a needed P with a random Q found in other axioms), or highly complex. To see this complexity take the axiom set to be {**A=>G**,

C=>D} and that the goal is **G**, and the statement is {**C**,**H**}. We have the trivial option of just adding **D=>G**, or we could just add **H=>G** and we do not have to use any axiom at all. Adding an axiom is the natural way to go, since it captures our intuitions, but formally it would have to add relevance, fuzziness and probably dispensability. Such an approach would make extensive use of substructural fuzzy-relevance logic which is a system less than a year old [12], but this is the way to go for a stable system, and this is the long term research goal.

There are however a couple of remarks to be made. First, the goal **G** must appear in the consequent of at least one axiom. Second, there is an option to prefer either the shortest or the longest chain. By preferring the shortest chain of inference, we obtain the necessary hidden premises, but to get any possible premise we could prefer the longest (with all the possible detours). Then simply we take all the chain elements including the statement and the goal, and we have what the speaker believes in. We could also place confidence metrics. This is superior provided we can count this confidence/importance along the chains in a meaningful way. As we will see, the final suggested approach will be a hybrid.

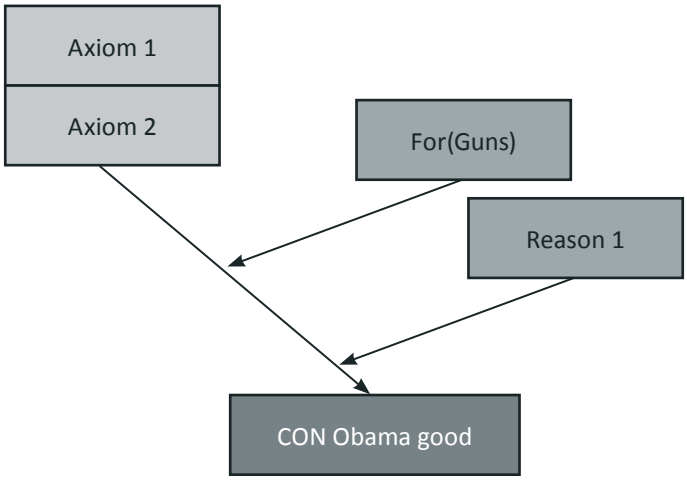

Figure 4: *ILP, adding a reason.*

The idea behind (ii) is to add reasons to the derivation, since they can be controlled in terms of axioms (Image 4). This has the advantage of a natural confidence scoring. The smallest amount of reasons to derive a goal is the most likely set of background information used. They also convey an epistemological aspect: what is that the speaker knows but he did not made explicit in his statement? The reasons can be also given a confidence level to measure this. A case could be made that the axioms constitute the societal background knowledge, whereas the reasons constitute the speaker's background knowledge, and the interaction between the two could provide additional insight.

APPLICATIONS

The completed system could have a great advantage over the state-of-the-art, namely that it is capable of filtering user values from their online posts. This may seem of limited practical use, but subjective information is harder to filter out automatically, and as such in more demand. There are two areas which could benefit greatly from such a system. The first is political campaigns of all kinds. Being able to identify the values of the people on social media makes it possible to individually target actions to gain their support.

The second, and much larger area of application is marketing. Customer values are a core component in building a good relation, and it is even more vital for targeted product placement. Since »privacy« is also a value that can be identified, this enables to classify customers according to their privacy preference, and include value-targeted advertisement for the customer with a low privacy preference. E.g. a green person who does not emphasize privacy, would get an ecological product basket, whereas if she were have privacy as her value, she could be getting the same product offers but scrambled with non-eco products.

CONCLUDING REMARKS

The aim of our research is to construct a finished module, whose architecture is presented in the present paper. Possible improvements are due to the possible hybridization of our approaches provided it will gain more accuracy and not be too expensive (in the computational sense) at the same time. New approaches such as substructural fuzzy relevance logic will be considered and implemented, and the finished module will be tested against a benchmark dataset.

REFERENCES

[1] Bronzi, M., Guo, Z., Mesquita, F., Barbosa, D. and Merialdo, P. (2012). Automatic Evaluation of Relation Extraction systems on Large-scale. In *AKBC-WEKEX '12 Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pp. 19-24.

[2] Dagan, I. and Glickman, O. (2004). Probabilistic Textual Entailment: Generic Applied Modelling of Language Variability. In *PASCAL Workshop on Learning Methods for Text Understanding and Mining*.

[3] Halevy, A., Norvig, P. and Pereira, F. (2009). The Unusual Effectivness of Data. *IEEE Intelligent Systems*, vol.24, no. 2, pp. 8-12.

[4] Lappin, S. and Leass, H. J. (1994). An Algorithm for Pronomial Anaphora Resolution. *Computational Linguistics*, vol. 20, no. 4, pp. 535-561.

[5] Manning, C. (1999). *Statistical Natural Language Processing*. Cambridge: MIT Press.

[6] Mitkov, R. (2002). *Anaphora Resolution (Studies in Language and Linguistics)*. London: Routledge.

[7] Nienhuys-Cheng, S.-H. and de Wolf, R. (1997). *Foundations of Inductive Logic Programming*. Berlin: Springer.

[8] Norvig, P. (1986). *A Unified Theory of Inference for Text Understanding*. Ph.D. thesis, printed as Berkeley EECS Dept. Report No. UCB/CSD 87/339.
[9] Russell, S. and Norvig, P. (2009). *Artificial Intelligence: a Modern Approach*. Harlow: Pearsons Education Ltd.
[10] Shen, W., Wang, J., Luo, P. and Wang, M. (2015). A Hybrid Framework for Semantic Relation Extraction over Enterprise Data. *International Journal on Semantic Web & Information Systems*, vol. 11, no. 3, pp 1-24.
[11] Sutton, C. and McCallum, A. (2012). An Introduction to Conditional Random Fields. *Foundations and Trends in Machine Learning*, vol. 4, no. 4, pp. 267-373.
[12] Yang, E. (2015). Substructural Fuzzy-Relevance Logic. *Notre Dame Journal of Formal Logic*, vol. 56, no. 3, pp. 471-491.
[13] Zelenko, D., Aone, C. and Richardella, A. (2003). Kernel Methods for Relation Extraction. *Journal of Machine Learning Research*, vol. 3, no. 1, pp. 1083-1106.
[14] (2016-05-20) ec.europa.eu/research/participants/data/ref/h2020/wp/2014_2015/annexes/h2020-wp1415-annex-g-trl_en.pdf
[15] (2016-05-25) norvig.com/chomsky.html
[16] (2016-05-20) www.cs.cmu.edu/~nbach/papers/A-survey-on-Relation-Extraction.pdf
[17] (2016-05-20) www.ibm.com/smarterplanet/us/en/ibmwatson/developercloud/relationship-extraction.html
[18] (2016-05-25) www-formal.stanford.edu/jmc/history/dartmouth/dartmouth.html