Using Verb-Noun Patterns to Detect Process Inputs

Munshi Asadullah, Damien Nouvel, and Patrick Paroubek

Laboratoire d'Informatique pour la Mécanique et les Sciences de l'Ingénieur (LIMSI)
Rue John von Neumann, Campus Universitaire d'Orsay, Bât 508,
91405 Orsay cedex, France
{munshi.asadullah, damien.nouvel, pap}@limsi.fr

Abstract. We present the preliminary results of an ongoing work aimed at using morpho-syntactic patterns to extract information from process descriptions in a semi-supervised manner. The experiments have been designed for generic information extraction tasks and evaluated on detecting ingredients from cooking recipes in French using a large gold standard corpus. The proposed method uses bi-lexical dependency oriented syntactic analysis of the text and extracts relevant morpho-syntactic patterns. Those patterns are then used as features for different machine learning methods to acquire the final ingredient list. Furthermore, this approach may easily be adapted to similar tasks since it relies on mining generic morpho-syntactic patterns from the documents automatically. The method itself is language independent, considering language specific parsers being used. The performance of our method on the DEFT 2013 data set is nevertheless satisfactory since it significantly outperforms the best system from the original challenge (0.75 vs 0.66 MAP).

1 Process Inputs in Specification Document

We are looking for a generic method to extract clearly defined target information from specification documents written in natural language. Specification in this context is the generic class of documents written to explain how a set of input elements are to be used or intended to be used in the processes described in the document. For the purpose of clarity we shall exemplify all the formal discussion within the cooking recipe domain.

1.1 Specification

The dictionary definition of "Specification" can be; "a detailed description of work to be done or materials to be used in a project; an instruction that says exactly how to do or make something"[1]. A large number of domains can crawl under this umbrella definition (e.g. cooking recipes, software specification, description of experiments, instruction manuals etc.). We are not suggesting that specification is a category of sentences, a taxonomical class (specification sentences), coined by Higgins [1]. We are in fact in total agreement with Heycock [2] in this aspect that a "specification" class for sentences is rather unnecessary.

[1] Source: Merriam-Webster Online Dictionary

P. Sojka et al. (Eds.): TSD 2014, LNAI 8655, pp. 181–188, 2014.

However, it is reasonable to assume that the type of text to be found in a specification document uses a finite number of sentence patterns to express the action descriptions. We thus hypothesize that a finite variation of morpho-syntactic pattern must exist to express process-object interaction. We then verified the hypothesis on the cooking recipe corpus described in Section 2.1. First, we need to establish the theoretical basis of events, processes and their interaction with objects.

1.2 Objects, Events and Processes

We are not trying to provide the philosophical or ontological basis for these concepts, rather present the extent of these concepts that have been used in our research. A detail study on objects and events and process can be found in [3], and we shall present most of the definition according to this work. However, the granularity of details for these concepts required for our research is much coarse and it shall be reflected in the following discussion.

Objects are spatial elements i.e. something that occupies physical space, that can change over time (e.g. from egg to boiled egg), but do not have any temporal part (e.g. an egg after one hour is still an egg). According to [3] objects can have spatial parts, but those parts are not the same object, rather matters or different objects (e.g. half of an egg is not an egg). However, This relation has little impact in our work (e.g. one may need only egg white for a recipe but the ingredient may still be an egg). Therefore, we restrict our focus to normalized concrete objects and their interaction in a process or an event.

Events and processes are difficult concepts to put one's fingers on precisely. Both events and processes has been described in [3] as an action over time, where processes describe an action without defined temporal boundary and events, with defined temporal boundaries i.e. start and end. From the cooking recipe a definitive example can be "boiling an egg for 15 minutes". By definition, this is an event i.e. start boiling an egg and finished after 15 minutes. Any intervals in between, by definition are not events, thus are processes.

Another property of the events and processes is that they can have spatial components i.e. involvement of objects in the action. In the cooking recipes all the processes and events are thus expected be informative. For our research these stringent definitions have little impacts, thus events and processes have been used as interchangeable through out the whole article. However, these definitions can be useful in the adaptation of the method for some other tasks. For the ingredient extraction task we are only interested in the inputs of a process or an event.

1.3 Why Focusing on Process Inputs?

Following Faure and Nédellec [4], we have assumed that verbs play a fundamental role in process detection and extract objects that are connected to them as direct objects, where subject dependency is optional, in particular for cooking recipes where verbs are often in imperative mood. From a syntactic and semantic point of view, we are looking for common structures involving a verb and their arguments, among which we are less interested in the syntactic subject and more in the semantic patients and

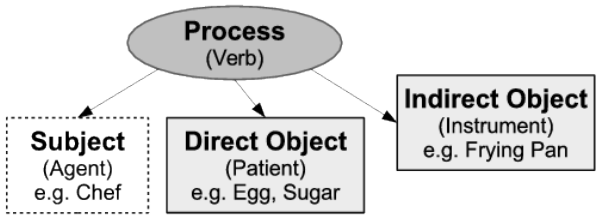

Fig. 1. Generic Verb-Argument Semantic

recipients arguments of a verb. For instance, in the following sentences, the presence of interesting verbs and their arguments are strong clues for recognizing processes and process inputs:

> **Put** the *flour*, *yeast*, *sugar* and *butter* in a food processor with a pinch of *salt*. […] **Add** the *butter* and *egg white*, pulse to a paste, then **fold in** the *chocolate* and set aside. […] **Sprinkle** with the remaining *almonds* and a little *caster sugar* and bake for 40–50 mins until puffed up and golden."[2]

Detection of the process input is a task specific requirement, thus, the adaptation of the method would require necessary changes at this level. We look for specific relations to identify ingredients, but one can look for the actors or instruments of a process for a different set of tasks (e.g. actors might be more interesting for named entity recognition).

2 Extracting Ingredient

From the original DEFT Challenge [5], we chose to evaluate on the task of extracting the ingredients from a recipe using a list of ingredients. This particular task was chosen because of the large amount of gold standard data. We shall give an overview of the task, performance of the participants of the original challenge and our approach in the following sub-sections.

2.1 The DEFT Challenge

The DEFT challenge is an annual French text mining evaluation workshop. Inspired by the Computer Cooking Contest[3], the 9th edition of this challenge was focused on the analysis of recipes written in French. There were 4 tasks from 2 main category,

1. Document Classification (Task 1–3)
2. Information Retrieval (Task 4)

[2] From `http://www.bbcgoodfood.com/recipes/3466/double-chocolate-easter-danish`
[3] `http://computercookingcontest.net`

For the first category, participants had to discover the level of difficulty (4 levels) for a recipe, the type of dish (starter, main dish or desert) a recipe most likely to be and identifying the best title for a recipe from a list of possible titles. For the information retrieval task, participants had to identify the ingredients for each recipe from a normalized list of possible ingredients. The details of the DEFT Challenge corpus, has been presented in Table 1. It is important to note that the normalized list

Table 1. DEFT Corpus

Corpus	Recipes	Sentences	Words	Ingredients
Training	13,866	141,613	2,013,934	101,563
Test	9,230	93,338	1,311,802	74,796

of possible ingredients, contains at least all the ingredients to be found in the recipes of the corpus. There is also the possibility that the actual ingredient name is not in the recipe text directly (e.g. "Fry egg" implies the presence of oil) and ingredients that are not present in the training corpus at all [6].

2.2 Evaluation and Results of The Challenge

There were 6 teams participated in the challenge, of which 2 industrial participants and the rest from the academic arena. All the results were processed using an evaluation system designed specifically for the challenge. We also used this evaluation platform to evaluate our system's performance. The evaluation metric used for the task was Mean Average Precision (MAP). Let us consider that there are N recipes and for any recipe R_i there are n_i ingredients (i.e. $\{I_i^1 \ldots I_i^j \ldots I_i^{n_i}\}$) and P be the precision, then the MAP metric is,

$$MAP = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{n_i}\sum_{j=1}^{n_i} P\left(I_i^j\right)$$

The final MAP score of the top 5 teams are listed in Table 2. "*Celi France*" performed

Table 2. DEFT Challenge Result (MAP Score)

Team	LIM&Bio	GREYC	LIA	Celi Fr.	Wikimeta
Run #1	0.4115	0.4881	*0.6287*	**0.6662**	0.5675
Run #2	0.4170	0.5074	0.6218	—	*0.6428*
Run #3	0.4649	0.5556	0.6191	—	—
Rank	5	4	3	1	2

the best with a MAP of **0.6622**. They presented a system that uses a hybrid approach

to solve the given problem [6]. They used a rule-based system to identify the potential ingredient and then filter them using a classifier on the basis of the type of the recipe (from task 2). Their system relied strongly on the lexical features (mostly lemma). We shall try to present the contrast between this method and our approach in the following section.

2.3 Dependency Sub-Tree Patterns

After careful revision of the approaches used in the original challenge [5], we were convinced that lexical features (regardless of the type of the approach i.e. rules, regular expressions or machine learning) are not effective for this particular problem. Thus, we decided to use document level features. However, we attempted several methods using lexical features before using pattern features. Searching the ingredient tokens from the global list in a recipe has been used as baseline. Several improvements has been attempted for the lexical approaches (e.g. using token, lemma, token+lemma etc.). We then decided to use morpho-syntactic patterns at document level. Our patterns are dependency sub-trees obtained by generating sub-trees of all possible depth using either or both lemma and POS features. The use of sub-trees for text representation can be found in [7]. the significant features of our sub-tree patterns are,

1. All possible depth are explored.
2. Each element of a pattern can represent lemma or POS or both.
3. Each element of a given pattern is concrete i.e. no variable element exists.
4. For a verb we explore the direct object and the prepositional phrases.

These patterns were extracted (as in [8]) automatically and since we focused on the dependency structure of a sentence, less significant tokens (e.g. prepositions) were discarded. It in turn gave use more information rich patterns. Example of some patterns can be found in Table 3, Among these patterns we found the longer and the verb centred patterns to be information rich. Different document level heuristics of these patterns have been used in our system as features for the ingredient extraction. The following sections will present an overview of our experiments.

Table 3. Top Patterns for Some Ingredients

Sel(Salt)	**Œuf(Egg)**	**Courgette**
N/poivre	N/sucre	N/courgette
N/huile/N/olive	N/chocolat	**N/huile/N/olive**
N/tomate	N/oeuf	**N/dés/N/légume**
N/pomme/N/terre	N/gâteau	**N/grains/N/moitié**
N/viande	V/battre	**N/rondelle/N/courgette**
N/oignon	**N/jaune/N/oeuf**	**V/couper/N/courgette**
N/poulet	N/crème	N/poivre
N/ail	A/vanillé	N/ail
N/courgette	N/vanille	**N/blanquette/N/veau**

3 Data Preprocessing

The recipes were parsed using statistical dependency parser for French[4]. Partly developed by ANR-SEQOIA Project[5], BONSAI is a collection of resources for French parsing, namely 3 statistical parsers. This resources have been presented in [9], that also reported 86.8% accuracy for dependency output using Berkeley parser [10] for French. We used the Berkeley parser for the preprocessing that establishes the dependencies between two tokens in FTB-DEP [11] formalism.

FTB Surface Dependency Annotation Guide [12] lists the basic annotation guideline for FTB-DEP formalism. FTB-DEP is based on the Dependency Grammar (DG) [13] formalism and like any DG based formalism, FTB-DEP adapted the relation types according to the target language and domain. There are 12 relations to annotate the relations of a token with the verbal governors (e.g. suj, obj etc.) and 8 to annotate the relations with non verbal governors (e.g. mod, coord etc.). There are 8 more specific relations reserved for manual annotation (e.g. mod_loc etc.). There is a virtual "*ROOT*" element for each sentence in the FTB-DEP, which is the hierarchical nucleus of a sentence and a natural extension for many formalisms of the DG family. Among all the relations only coordination required some normalization.

The coordination dependency is represented as a chain rather than separate relations, i.e. N coordinated tokens are represented using a combination of two dependencies, "*COORD*", connects the first conjunct with the first coordinator and "*DEP–COORD*" connect the next coordinator. Consecutive conjuncts are connected in a chain with the "*COORD*" relation. We had to resolve the "*COORD*" to a flatter representation for some of the experiments. All the parsers output in an adapted **CoNLL**[6] data format. Once we have the parsed output[7] we can extract the patterns and start experimenting.

4 Experiments and Results

We experimented with two machine learning methods, logistic regression and perceptron. For the pattern based experiments, we have to map each pattern at document level for each ingredient. Thus, in the training set if we have n documents $D = \{d_1, d_2 \cdots d_n\}$ and m ingredients $I = \{i_1, i_2 \cdots i_m\}$ then we can have $D^x \subset D$, where the ingredient $i^x (1 < x < m)$ is present. All the features for the machine learning methods had calculated from this subset. The results from out experiments are listed in Table 4,

All the systems with the prefix "identify" uses string matching and represent the baseline for the task. All the system with the prefix "learn" is a machine learning system. Except for "learn-percept" and "learn-percept-rank" all the machine learning system uses logistic regression. In Table 4, the suffix of a system name state the features used (e.g. tokens, lemma etc.). the suffix element "mine" means, the patterns had been used as features. Although it is not explicitly mentioned, the perceptron based system uses the patterns as features. The suffix element "coord" refers to the fact that the data has

[4] `http://alpage.inria.fr/statgram/frdep/fr_stat_dep_parsing.html`

[5] `https://sites.google.com/site/anrsequoia/home`

[6] `http://nextens.uvt.nl/depparse-wiki/DataFormat`

[7] script available at `https://github.com/eldams/ConLL-SimpleReader`

Table 4. Experimental Results

System Features	MAP	P(5)	P(10)	P(100)	R(5)	R(10)	R(100)
identify-tokens	0.3564	0.4960	0.3556	0.0375	0.3607	0.4930	0.5114
identify-lemmas	0.4355	0.5402	0.4193	0.0456	0.4000	0.5893	0.6262
identify-tokens+lemmas	0.4430	0.5375	0.4249	0.0469	0.3990	0.5986	0.6428
learn-lemmas	0.7196	0.7409	0.5306	0.0695	0.5420	0.7446	0.9493
learn-lemmas+mine	0.7362	0.7565	0.5432	0.0695	0.5538	0.7615	0.9487
learn-lemmas+mine+coord	0.7364	0.7555	0.5431	0.0695	0.5532	0.7610	0.9490
learn-lempos	0.7182	0.7414	0.5305	0.0695	0.5423	0.7446	0.9495
learn-percept	**0.7500**	0.7588	0.5547	0.0706	0.5545	0.7779	0.9648

been normalized, following the discussion in Section 3 and then used in the extraction process.

The primary reason for the lexical features to fail is found to be the similar reasons discussed in Section 2.3. For example if "*fry*" is mentioned in a recipe that implicitly considering "*oil*" as an ingredient. If "*oil*" never appears in the recipe, there shall be no lexical map between "*oil*" and "*fry*". But by document level mapping, any pattern may be linked to any ingredient e.g. the verb "*fry*" to "*oil*". As it can be seen clearly, even the logistic regression produces higher scores when patterns are used in conjunction with lexical features. Although the performance of the "perceptron" algorithm is very good, the first significant improvement was seen when document level mapping was used ("learn-lemmas" shows about 30% improvement) where as using the patterns improves the MAP between 2% and 4%. All the results using document level features shows better performance than the top system from the original challenge.

5 Conclusion

We have presented our preliminary experimental results in search of a semi-supervised information retrieval System. The results are never the less inspiring. We can be confident of the performance of the system once tested with data from different domains. We are also looking forward to test the system for some other information retrieval task, that might require us to push the boundary of our current systems. Introducing document level semantic information is also a possible future direction of the research.

References

1. Higgins, F.R.: The pseudo-cleft construction in English. PhD thesis, Massachusetts Institute of Technology (MIT), Cambridge, MA, United States (1973)
2. Heycock, C.: Specification, equation, and agreement in copular sentences. The Canadian Journal of Linguistics / La Revue Canadienne De Linguistique **57** (2012) 209–240
3. Galton, A., Mizoguchi, R.: The water falls but the waterfall does not fall: New perspectives on objects, processes and events. Applied Ontology **4** (2009) 71–107
4. Faure, D., Nedellec, C.: Knowledge acquisition of predicate argument structures from technical texts using machine learning: The system asium. In: Proceedings of the 11th

European Workshop on Knowledge Acquisition, Modeling and Management. EKAW '99, London, UK, Springer-Verlag (1999) 329–334
5. Grouin, C., Zweigenbaum, P., Paroubek, P.: DEFT2013 se met à table : présentation du défi et résultats. In: Actes de DEFT 2013: 9e DÉfi Fouille de Textes, Les Sables d'Olonne, France (2013) 1–14
6. Dini, L., Bittar, A., Ruhlmann, M.: Approches hybrides pour l'analyse de recettes de cuisine DEFT, TALN-RECITAL 2013. In: Actes de DEFT 2013: 9e DÉfi Fouille de Textes, Les Sables d'Olonne, France (2013) 53–65
7. Pak, A., Paroubek, P.: Text representation using dependency tree subgraphs for sentiment analysis. In: Proceedings of the 16th International Conference on Database Systems for Advanced Applications. DASFAA'11, Berlin, Heidelberg, Springer-Verlag (2011) 323–332
8. Nouvel, D., Antoine, J.Y., Friburger, N.: Pattern mining for named entity recognition. LNCS/LNAI Series **8387 (post-proceedings LTC 2011)** (2014)
9. Candito, M., Nivre, J., Denis, P., Henestroza Anguiano, E.: Benchmarking of statistical dependency parsers for french. In: Proceedings of the 23rd International Conference on Computational Linguistics: Posters. COLING '10, Beijing, China (2010) 108–116
10. Petrov, S., Barrett, L., Thibaux, R., Klein, D.: Learning accurate, compact, and interpretable tree annotation. In: Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics. ACL-44, Stroudsburg, PA, USA, Association for Computational Linguistics (2006) 433–440
11. Candito, M., Crabé, B., Pascal, D.: Statistical french dependency parsing: Treebank conversion and first results. In: Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10), Valletta, Malta, European Language Resources Association (ELRA) (2010) 1840–1847
12. Candito, M., Crabbé, B., Falco, M.: Dépendances syntaxiques de surface pour le français – Schéma d'annotation pour un corpus en dépendances obtenu par conversion du FrenchTreebank. (2011)
13. Tesnière, L.: Éléments de Syntaxe Structurale. Éditions Klinksieck, Paris, France (1959)