# Building a Lexically and Semantically-Rich Resource for Paraphrase Processing

Wannachai Kampeera and Sylviane Cardey-Greenfield

Centre Tesnière - Équipe d'Accueil EA 2283
Université de Franche-Comté - UFR SLHS
30 rue Mégevand, 25030 Besançon Cedex, France
{wannachai.kampeera,sylviane.cardey}@univ-fcomte.fr
http://tesniere.univ-fcomte.fr

**Abstract.** In this paper, we present a methodology for building a lexically and semantically-rich resource for paraphrase processing on French. The paraphrase extraction model is rule-based and is guided by means of predicates. The extraction process comprises 4 main processing modules: 1. derived words extraction; 2. sentences extraction; 3. chunking & head word identification, and 4. predicate-argument structure mapping. We use the corpus provided by an agro-food industry enterprise to test the 4 modules of the paraphrase structures extractor. We explain how each processing module functions.

**Keywords:** paraphrase, paraphrase structures, paraphrase structures extraction, lexical resource.

## 1 Introduction

Paraphrase processing is an important issue in Natural Language Processing. A great number of natural language applications integrate paraphrase processing modules to improve their performance [1]. [2] provides an extensive survey on paraphrase processing.

We have carried out a linguistic analysis on a collection of 899 verbatims or emails ($\approx$ 2376 sentences) in French in the agro-food industry domain to discover linguistic properties of paraphrases. The study reveals that a great number of paraphrases are constructed by means of derived adjectives and nouns with triggered syntactic transformations.

Inspired by the result of this linguistic analysis and by the fact that the lexicon largely varies from one domain to another whereas the sentence structure merely changes, our intention is to build a paraphrase structures database for various paraphrase processing tasks. To do so, automation guided by linguistic knowledge appears to be one of the most convenient means for extracting paraphrase structures.

In the next section, we summarize the paraphrase structures extraction model and interactions between its four main components. Following the processing flow, we explain how derived words can be extracted from Wiktionary in section 2.1.

H. Isahara and K. Kanzaki (Eds.): JapTAL 2012, LNAI 7614, pp. 138–143, 2012.

Section 2.2 discusses the candidate sentences extraction process. In section 2.3, we provide examples of sentences output by a chunker. We describe the mapping from chunked sentences to predicate-argument structures using ontologies in section 2.4.

## 2 Paraphrase Structures Extraction

The architecture of the extractor and its processing flow is shown in Fig. 1. We have implemented a prototype for each of the main processing modules. For a given verb, the system outputs a set of paraphrase structures. As we want to construct a reliable lexical resource which will be used by many paraphrase processing systems, human intervention is necessary in the final validation phase.

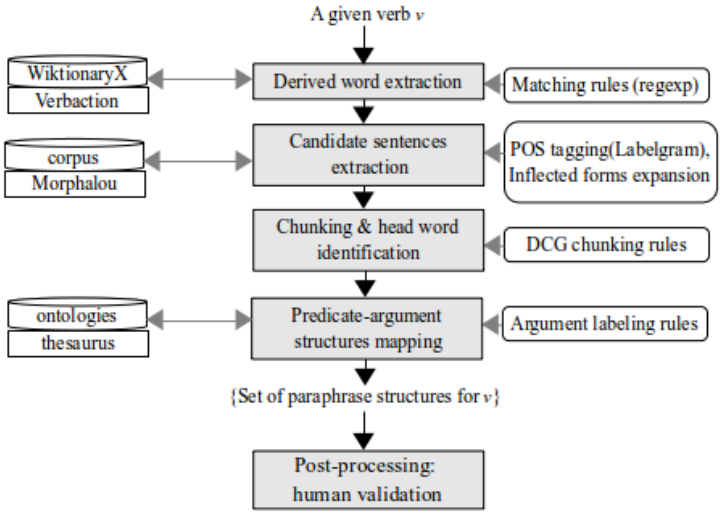


**Fig. 1.** Paraphrase structures extraction process

### 2.1 Extracting Derived Nouns and Adjectives

The first step is to identify for a given verb all its derived nouns and adjectives. There are few available French lexical resources for such a task. Verbaction [3] is an exploitable resource which provides, for a given verb $v$, a corresponding noun meaning *the action of* $v$.

As a result, our idea is to extract more derived nouns and adjectives of a given verb from WiktionaryX [4], a more compact and exploitable version of French Wiktionary. However, a lexical entry in WiktionaryX does not contain a ready-to-use derived words list, i.e. it includes antonyms, words or expressions whose meaning differs considerably from the base verb. The entry *goûter* 'to taste' has the following derived words:

– *goûteur* 'taster'
– *dégouter* 'to disgust'.

Being a verb, *dégouter* can easily be excluded from the candidates list. For *goûteur* 'taster' however, morphological checking is needed to make sure that it is a possible derived word of *goûter* 'to taste'. Thus, *goûteur* 'taster' will be validated if the following definition is satisfied: a noun or adjective $w$ is said to be derived from a verb $v$ if and only if $w$ can be decomposed into two parts: the morphological root of $v$ and a noun or adjective suffix.

The noun *goûteur* 'taster' is decomposed into: morphological root *gout* + noun suffix *eur*. Thus, *goûteur* 'taster' is accepted as a derived noun of *goûter* 'to taste' and other irrelevant lexical units are rejected.

Nevertheless, the most frequent case is that Wiktionary does not provide the derived words section, hence neither does WiktionaryX. To discover missing derived words of an entry, we make use of available information such as its definition and etymology. Let us take the example of *aimer* 'to like' for which a semantically derived noun *amateur* 'the person who likes' is found by virtue of the latter's etymological information: *[...] Du latin amator ("celui qui aime") [...]* 'From latin amator ("the person who likes")'. The definition *"celui qui aime"* 'the person who likes' is matched by the rule which states that: for a given verb $v$, find its derived agent noun(s) having the meaning/definition *"person who conjugated-v"*.

The advantage of this method is that we retrieve not only lexically derived words but also those semantically derived from the verb. In fact, trying to derive *amateur* from *aimer* will not succeed because they simply have different morphological roots in modern French. The input verb together with the extracted derived words constitute the canonical keywords set.

### 2.2 Candidate Sentences Extraction

The verb and its derived nouns and adjectives represent the canonical keywords set as explained in 2.1. We then compile all conjugated forms of the verb and all inflected forms of the derived nouns and adjectives using the inflected forms dictionary Morphalou [5]. The result is the final keywords set $K$ including lemmas and their inflected forms.

The next step is to extract candidate paraphrase structures from a corpus. To do so, Labelgram [6] first performs POS tagging. Next, we compare each word of the sentences against the keywords in $K$. The sentences which contain at least one keyword belonging to $K$ represent the set of candidate sentences to be processed in the next module.

### 2.3 Chunking and Head Word Identification

In this experimental stage, chunking is done by Definite Clause Grammars (DCG) with Prolog. We exclusively target verb groups, noun groups and adjective groups so as to discover the predicate-argument structure of the sentence.

In fact, we assume that arguments are *a priori* noun groups surrounding the predicate. Ontological information on each argument determines the predicate-argument structure configuration in the next processing module (see 2.4). Predicates in this context refer to keywords in $K$ no matter their grammatical category.

Let us give some examples of chunks, output by our chunker for the set of keywords $K$ of the verb *décevoir* 'to disappoint'. Note that N stands for noun chunk, ADJ for adjective, V for verb; the line immediately below each example is its literal English translation.

(a) [N je] [V suis] [ADJ déçue] par [N une tablette de chocolat ProductName]
'I am disappointed by a bar of chocolate ProductName'
(b) [N je] [V trouve] [N ce produit très décevant]
'I find this product really disappointing'
(c) [N ce produit] [V est] [ADJ assez décevant]
'This product is quite disappointing'
(d) [N CompanyName] [N me] [V déçoit]
'CompanyName disappoints me'

Before the predicate-argument labeling phase, it is necessary to identify the head of each chunk and remove trivial words because we aim for the predicate-argument structure of the sentence and not the local structure of each chunk. Also, inflected words are replaced by their lemmas at this stage. For (a), (c), and (d), identifying head words is straightforward giving the following results:

(a) [N je] [V être] [ADJ déçue] par [N chocolat ProductName]
'I be disappointed by chocolate ProductName'
(c) [N produit] [V être] [ADJ décevant]
'product be disappointing'
(d) [N CompanyName] [N me] [V décevoir]
'CompanyName disappoint me'

If the keyword is a part of noun groups as in (b), a decomposition rule (2) applies:

(1) removing trivial words: [N ce produit très décevant] $\longrightarrow$ [N produit décevant]
(2) decomposing a noun group into a noun chunk and an adjective chunk: [N produit décevant] $\longrightarrow$ [N produit] [ADJ décevant],

yielding:

(b) [N je] [V trouver] [N produit] [ADJ décevant]
'I find product disappointing'

Chunking is efficient enough for the current corpus which is domain-specific containing verbatims. This can be explained by the fact that clients (senders) use a mixture of familiar and standard language. The writing style is rather direct, emotional and brief.

### 2.4 Mapping into Predicate-Argument Structures

In 2.3, we obtained core structures composed of a keyword and their potential arguments. To map these structures to predicate-argument ones, in other words paraphrase structures, we have created domain-specific ontologies for this corpus. For now, we focus on two categories of entities: *Company* (company, brand, product, product names, you, your, etc.); and *Client* (I, my, our, consumer, client, etc.). The predicate-argument labeling relies on these ontologies, e.g. argument1 belongs to *Company*, argument2 belongs to *Client*.

As described in 2.3, the key assumption is that arguments are noun groups surrounding the predicate. Therefore, verbs and prepositions which do not belong to the keywords set are left as such. The result of the mapping is a set of paraphrase structures formalized as predicate-arguments. This set of paraphrase structures is 'owned' by the lexeme *décevoir* 'to disappoint':

(a) arg2(n:client) être pred(déçu) par arg1(n:product)
(b) arg2(n:client) trouver arg1(n:product) pred(décevant)
(c) arg1(n:product) être pred(décevant)
(d) arg1(n:company) arg2(n:client) pred(décevoir)

The paraphrase structures above are domain-dependent, as well as the current method for argument labeling. Nevertheless, more general paraphrase structures can be obtained by simply removing ontological information, e.g. company, product, client.

Besides, instead of using domain-specific ontologies, it is possible to generalize the labels of each argument using a thesaurus, or a semantically rich lexical database such as WordNet [7]. We are currently investigating the possibility of using Wolf [8], a WordNet for French.

### 2.5 Related Work

The Classificatim system [9] applied a rule-based paraphrase recognition module to classify verbatims into sets of sentences which convey the same meaning. The system reports 99% of success on corpora of the agro-food domain (with normalized text and 84% with raw text). However, writing paraphrase rules manually required a significant time and effort. The present methodology would contribute to reduce time and human effort in the rules-writing phase for such projects.

DIRT [10] is an unsupervised paraphrase extraction algorithm using the Distributional Hypothesis [11]. We think that the Distributional Hypothesis is certainly a convenient theory for simpler NLP tasks such as POS tagging. However, stating that words which occur in the same contexts tend to have similar meaning is not necessarily true. In fact, for a frequent path *I...Verb...Pizza* in our corpus, *Verb* can be anything (like, dislike, ordered, finished, throw). Secondly, while DIRT mainly extracts paraphrase patterns with two arguments *X...Y*, our approach is not concerned by such a limit because instead of using two arguments (contexts) *X...Y* to discover paraphrases, we use lexical relations such as

semantic derivation (and we will be using in future work, synonymy, negation on antonyms, hyperonymy and the likes) to find paraphrase candidates and their arguments.

## 3 Conclusions

We have presented, as the result of a refined linguistic analysis, a methodology for building a lexical resource for paraphrase processing. The paraphrase structures extracting method is lexical-driven and rule-based. The outlined methodology would allow rapid and innovative lexical resources' development as linguists are concerned in validating the final output by the extractor. The linguistic knowledge discovery (paraphrase structures) is mainly performed by the extractor.

## References

1. Kampeera, W., Cardey, S.: Paraphrases in Natural Language Processing. In: Proceedings of the 12th International Symposium on Social Communication - Comunicación Social en el Siglo XXI, vol. II, pp. 963–967. Santiago de Cuba, Cuba (2011)
2. Androutsopoulos, I., Malakasiotis, P.: A Survey of Paraphrasing and Textual Entailment Methods. Journal of Natural Language Processing 11, 151–198 (2009)
3. Hathout, N., Namer, F., Dal, G.: An Experimental Constructional Database: The MorTAL Project. In: Boucher, P. (ed.) Many Morphologies. Cascadilla, Somerville (2002)
4. Sajous, F., Navarro, E., Gaume, B., Prévot, L., Chudy, Y.: Semi-automatic Endogenous Enrichment of Collaboratively Constructed Lexical Resources: Piggybacking onto Wiktionary. In: Loftsson, H., Rögnvaldsson, E., Helgadóttir, S. (eds.) IceTAL 2010. LNCS, vol. 6233, pp. 332–344. Springer, Heidelberg (2010)
5. Romary, L., Salmon-Alt, S., Francopoulo, G.: Standards going concrete: from LMF to Morphalou. In: Workshop on Electronic Dictionaries, Coling 2004, Geneva, Switzerland (2004)
6. Cardey, S., Greenfield, P.: Disambiguating and Tagging Using Systemic Grammar. In: Proceedings of the 8th International Symposium on Social Communication, pp. 559–564 (2009)
7. Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press, Cambridge (1998)
8. Fišer, D., Sagot, B.: Combining Multiple Resources to Build Reliable Wordnets. In: Sojka, P., Horák, A., Kopeček, I., Pala, K. (eds.) TSD 2008. LNCS (LNAI), vol. 5246, pp. 61–68. Springer, Heidelberg (2008)
9. Cardey, S., Greenfield, P., Bioud, M., Dziadkiewicz, A., Kuroda, K., Marcelino, I., Melian, C., Morgadinho, H., Robardet, G., Vienney, S.: The Classificatim Sense-Mining System. In: Salakoski, T., Ginter, F., Pyysalo, S., Pahikkala, T. (eds.) FinTAL 2006. LNCS (LNAI), vol. 4139, pp. 674–683. Springer, Heidelberg (2006)
10. Lin, D., Pantel, P.: Discovery of Inference Rules for Question Answering. Natural Language Engineering 7(4), 343–360 (2001)
11. Harris, Z.: Distributional Structure. In: Katz, J.J. (ed.) The Philosophy of Linguistics, pp. 26–47. Oxford University Press, New York (1985)