Biological event extraction using SVM and composite kernel function

Maha Amami*, Aymen Elkhlifi**, Rim Faiz***

*LARODEC, ISG de Tunis, 2000, Le Bardo, Tunisie
MahaAmami@isg.rnu.tn,
**LaLIC, Université Paris-Sorbonne, 28, rue Serpente, 75006 Paris, France
Aymen.Elkhlifi@paris4.sorbonne.fr
***LARODEC, IHEC de Carthage, 2016, Carthage Présidence, Tunisie
Rim.Faiz@ihec.rnu.tn

Abstract. With an overwhelming of experimental and computational results in molecular biology, there is an increasing interest to provide tools that will automatically extract structured biological information recorded in freely available text. Extraction of named entities such as protein, gene or disease names and of simple relations of these entities, such as statements of protein-protein interactions has gained certain success, and now the new focus research has been moving to higher level of information extraction such as co-reference resolution and event extraction. It is precisely the last of these tasks which will be focused in this paper. The biological event template allows detailed representations of complex natural language statements, which is specified by a trigger and arguments labeled by semantic roles.

In this paper, we have developed a biological event extraction approach which uses Support Vector Machines (SVM) and a suitable composite kernel function to identify triggers and to assign the corresponding arguments. Also, we make use of a number of features based on both syntactic and contextual information which where automatically learned from the training data.

We implemented our event extraction system using the state-of-the-art of NLP tools. We achieved competitive results compared to the BioNLP'09 Shared task benchmark.

1 Introduction

The past decade has seen an explosive growth in the amount of experimental and computational biological data. This growth is accompanied by an increase in the number of biological texts discussing the results. The MEDLINE [1] database contains in 2010 over 20 million articles, and the database is currently growing at a rate of more than 10% each year (Ananiadou and al., 2006). The availability of huge textual resources provides the scientists with the chance

1. http://www.ncbi.nlm.nih.gov

to search correlations or associations such as protein-protein interactions, and gene-disease associations. However, retrieving and processing this information is difficult due to the lack of formal structure in the natural language in these documents (Mooney and Bunescu, 2005; Elkhlifi and Faiz, 2009). Using keyword queries that retrieve a large set of relevant papers, scientists can navigate through hyperlinks between genome database and referenced papers. To extract the requisite knowledge from the retrieved papers, they must identify the relevant information. Such manual processing is time consuming and repetitive, because of the bibliography size, and the database continuous updating. From the Medline database, the focused query "Bacillus subtilis and transcription" which returned 2209 abstracts in 2002, retrieves 3727 of them today. As a result, there has been an increased interest in the application of *information extraction* techniques to support database building and to intelligently find knowledge in documents.

Previous research in biological information extraction have focused in particular on the task of recognizing named entities in texts, such as protein, gene, drug or disease names (Rindflesch and al., 2000) and on the extraction of simple relations of these entities, such as statements of protein-protein interactions (Blaschke and al., 1999). Recently, the focus of research has been moving to higher level of information extraction such as co-reference resolution (McCarthy and al., 1996) and event extraction (Kim and al., 2009). It is precisely the last of these tasks which will be focused in this paper. In fact, the biological event extraction task has a broad range of applications ranging from support and annotation of pathways to automatic population and enrichment of databases and ontologies.

With regard to the event extraction from news articles task, Elkhlifi and Faiz (2007, 2009, 2010b) propose a machine learning approach to extract events based on an ontology. Also, they propose an effective algorithm to annotate these events whith order of n^5 time complexity (Elkhlifi and Faiz, 2010a).

The event extraction from biological texts is a non-trivial task, though the maturation of basic technologies. It has to recognize diverse surface forms in text that describe the same biological process (event trigger class) and identify which biological entities are involved (event arguments).

The rest of the paper is organized as follows: In Section 2, we present techniques adopted for event extraction from biological texts. In Section 3, we propose our event extraction approach with Support Vector Machines (SVM) and composite kernel function. In Section 4, we describe the implementation of our event extraction system and the experimental results evaluating our approach.

2 Related works on biological event extraction methods

The biological event extraction refers to the task of detection of typed, text bound events and assignment of proteins as arguments, using basic tools for biological text analysis and manually annotated resources such as and BioInfer and GENIA corpora (Kim and al., 2008).

A biological event extraction template is defined by a trigger and arguments (Kim and al., 2009). The semantic roles are assigned to these arguments.

For instance, in the sentence *"Monocyte tethering by P-selectin regulates monocyte chemotactic protein-1 and tumor necrosis factor-alpha secretion."*, the event is characterized by an event trigger verb "regulates" which is a regulation event class, and the event arguments are "P-

selectin", "monocyte chemotactic protein-1" and "tumor necrosis factor-alpha". The first argument "P-selectin" is tagged by the semantic role "agent" or "cause". The arguments "monocyte chemotactic protein-1" and "tumor necrosis factor-alpha" acts as theme.

Hence, most of the event extraction approaches are pipelines of three major sub-tasks (Björne and al., 2008; Vlachos and al., 2009; Cohen and al., 2009).

1. Pre-processing: it provides tokens, POS tags and dependency parsers as an input to the event detector.
2. Event trigger detection: it requires assignment of each token to an event class.
3. Event argument detection: it consists of finding all participants in an event and assigning the functional role to each of the determined participants in an event.

Several approaches are developed for biological event extraction sub-tasks using techniques such as full parsing (Vlachos and al., 2009), pattern-based (Buyko and al., 2009), machine learning (Björne and al., 2009) and ontology driven techniques (Cohen and al., 2009).

The most commonly used techniques in event extraction approaches are the patterns matching, which implements a set of manually defined rules developed by experts or automatically learned from training data; and the machine learning, which exploits various features to extract events. The pattern-based approaches try to use context information for finding biological events. They usually look for certain words occurring near named entities or use part-of-speech (POS), syntax and semantic information. Hence, patterns can be written using dictionaries, preposition based parsing and so forth.

An example of pattern-based method is the work by Buyko and al. (2009) which employ a number of dictionaries to extract event triggers containing, discriminative trigger for an event class (e.g., trigger "Phosphorylate" for the event class "Phosphorylation") and not fully discriminative event trigger with common strings (in-context disambiguators) to help in the identification of the event class.

Another technique of pattern matching is the preposition parsing, event extraction templates are filled with parsed material surrounding prepositions such as "by" and "of" which are often cue strings of theme or cause roles. The sentence "apoptosis induced by the p53 tumor suppressor" contains preposition "by" which mention using parse trees and hand-coded patterns, "p53 tumor suppressor" as cause argument, "apoptosis" as theme argument and "induced" as an event trigger verb.

The pattern-based approaches exhibit high precision but their recall is low because many of the relations in the text are left undiscovered by coded patterns.

Machine learning approaches to biological event extraction have utilized various techniques such as Support Vector Machines (SVM), Hidden Markov Model (HMM), and k-Nearest Neighbors (k-NN).

The work by Björne and al. (2009) applies SVM to detect biological events using a wide array of features and semantic networks derived from full dependency analysis. Thus, they represent each sentence in term of graph where the nodes correspond to proteins and event triggers and edges correspond to event arguments. Event nodes are formed based on the prediction of individual tokens, and event edges are identified by predicting for each trigger-trigger or trigger-named entity pair whether it corresponds to an appropriate event argument. The features used in the SVM classifier include the morphological properties of the token to be classified, such as character bigrams and trigrams, and tokens that depend on it, the number of named entities and the bag of word of token counts in the sentence. For a given class, the

classifier calculates the confidence score of a token belonging to the class. After event trigger detection, all potential edges, which connect an event node to another or to a named entity node, are classified based on the SVM classifier as a theme, cause or negative class. The set of edge features are built by combining the attributes of tokens, the n-grams which define the variation of dependency directions of 2 to 4 consecutive tokens, the semantic node features which combine the token features of the two terminal event or entity node of the potential edge, individual component features which combine a token or an edge attribute with the token or edge position at either the interior or the end of the path.

Björne and al. (2009) treat texts as dependency trees which is the most important source of features. A deeper semantic and contextual analysis to learn semantic features can tackle the problem of disambiguation. In addition, they use the linear SVM (linear kernel) which is not able to capture the event extraction task specific similarity between the string features.

In this paper, we use the machine learning technique for trigger and argument detection sub-tasks. We will use a kernel-based method to predict events, namely the Support Vector Machine (SVM). However, one of the major challenges in kernel-based method is the choosing of a suitable kernel function for a given classification problem. Thus, we attempt to design a composite kernel function suitable for event extraction. In addition, we propose a rich set of features derived from the dependency and semantic analysis.

3 Proposed event extraction approach from biological texts

We aim at developing a new and effective method for extracting biological events from the literature. Our proposal is to generate automatically a wide number of features, and use an SVM classifier with a suitable kernel function abled to capture the event detection task specific similarity between these features. The whole process is described in the next sections.

3.1 Pre-processing

For extracting events from text, we employ many natural language processing techniques. We apply state-of-the-art systems trained on biological corpora for splitting, tokenization and Part of speech (POS) tagging. Then, we use parsers to analyze the syntactic relations among the entities in the sentence. Finally, the syntactic analysis is complemented by a semantic processing; a step which assigns semantic classes (e.g., gene, protein, cell type, etc.) using semantic resources.

3.2 Trigger detection

The event trigger detection is the task of identifying individual words in the sentence that act as an event trigger words and assigning the correct event class to each of the determined triggers. First, we filter out tokens, that are a named entity and whose POS tag is not a noun, a verb, or an adjective; and sentences that do not have any proteins. Then, we proceed with extracting a set of features for each candidate trigger based on both the context in sentence and the dependency parse. Our initial attributes include both features similar to those used in (Björne and al., 2009; Elkhlifi and Faiz, 2009) and new ones. The tuned feature set is showed in Table 1.

Type	Feature
Token features	Token text
	Stem from the Porter stemmer (Porter, 1980)
	Lemma from the Natural Language Toolkit [2]
	Token POS
	Word and POS of the nearest protein
	Presence of symbol / Capital letter
	N-grams (n = 2, 3) characters
	Indicator whether the token is a stop word
	Presence of an adjacent verb or noun
	Presence in a trigger gazetteer
	Semantic type
Frequency features	Number of named entities in the sentence
	Number of stop words in the sentence
	Bag of word counts of token words in the sentence
	TF-IDF score of token word in the training set
Dependency features	Set of dependency chains features up to depth of three
	Dependency label path of the nearest protein
Shortest path features	N-grams of dependencies (n = 2, 3, 4)
	N-grams of words (n = 2, 3, 4)
	Length of the shortest path
	Presence of some token along the shortest path in a trigger gazetteer

TAB. 1 – *Features for trigger detection.*

After that, the candidate triggers are classified into event classes (e.g., gene expression, transcription, phosphorylation), and a negative event class using a machine learning classifier.

However, traditional machine learning classification techniques perform poorly when working directly because of the high dimensionality of the data. Thus, we use the kernel-based method SVM which has been scales relatively well to high dimensional data. One of the major challenges in kernel-based method is the choosing of **a suitable kernel function** for the given classification problem (Burges, 1998).

In fact, there are standard choices such as a gaussian or polynomial kernel functions that are the default options. However, they prove ineffective to train the classifier with the large data sets (Hsu and al., 2010). Also, they are not applied to string features.

In the work presented by (Björne and al., 2009), the linear kernel function is used in trigger detection with large training sets. It computes the dot product between instances as,

$$K(X,Y) \quad = \quad X^T \cdot Y \tag{1}$$

where, $\langle x_i, y_i \rangle = 1$ if x_i and y_i are the same and 0 otherwise.

2. http://www.nltk.org/Home/

However, when we are dealing with string features, such dot product based similarity computation is not able to capture the trigger detection task specific similarity between string features.

In what follows, we elaborate a composite kernel function based on vector representation of features. We define the similarity function as kernel in SVM for each type of feature i.e., word text, n-gram and dependency path.

First, we give the definition of a similarity matrix in the input X. The similarity matrix S is a $n \times n$ matrix with two entries for every pair of vectors in X, $S(i_k, j_k) = s_{ij}$ for i, $j \in \{1, 2 \ldots, n\}$ the indices of instances in X.

$$S = \begin{pmatrix} s_{11} & s_{12} & \ldots & \ldots & s_{1n} \\ \vdots & \vdots & \ddots & s_{ij} & \vdots \\ s_{n1} & s_{n2} & \ldots & \ldots & s_{nn} \end{pmatrix}$$

Then, we define the global similarity matrix based inner product $\langle .|. \rangle_{MAT} : X^l \times X^l \mapsto \mathbb{R}$ as,

$$\langle x_i | x_j \rangle_{MAT} = \sum_{k=1}^{l} S(i_k, j_k) \tag{2}$$

where k = {**word text**, **n-gram**, **dependency path**}.

WordNet similarity We use the WordNet HSO similarity measure (Hirst and St-Onge, 1998) which measure the semantic relatedness between two lemmas defined as following,

$$rel_{HS}(w_1, w_2) = C - PathLength - k * d \tag{3}$$

where C and k are constants, $PathLength$ is the length of the shortest path and d is the number of changes of direction in the path.

N-grams kernel function To compute the similarity between the bigrams and trigrams of two strings, we use the k-spectrum (n-grams) kernel function (Leslie and al., 2002). Given a string x, an alphabet A ($|A| = l$), we define a feature map from X to $\mathbb{R}^{k^l}$ by,

$$\phi_k(x) = (\phi_a(x))_{a \in A^k} \tag{4}$$

where $\phi_a(x)$ = number of occurrences of a in x.

Thus, the k-spectrum kernel function is defined as,

$$K_k(x, y) = \langle \phi_k(x), \phi_k(y) \rangle \tag{5}$$

Dependency kernel Our dependency kernel is a modification of Kim and al. (2008) walk kernel for a dependency structure, which is tested with a SVM classifier on the LLL 05 challenge task to extract genic interactions and achieved a promising result. We define our dependency graph kernel to capture the isomorphism between two graph structures. For this, we sum up the number of common walks features between two dependency graphs $G(V, E)$

and $G'(V', E')$. Note that the graph means the directed dependency chain paths at depth of n ($n = 1, 2, 3$).

In our work, we consider the walk of length 1 called a v-walk. In addition, we present an e-walk that begins and ends with an edge e. We generate lexical walk features, which consist of lexical words L_w; and syntactic walk features S_w, which consist of POS and dependency relations. The set of lexical and syntactic walk features is noted by F_w of an edge e.

Hence, our dependency graph kernel is expressed in Equation 6.

$$K(G, G') = \sum_{e \in E} \sum_{e' \in E'} K_{walk}(e, e') \tag{6}$$

where,

$$K_{walk}(e, e') = \begin{cases} 1 & \text{if} \quad f_w = f'_w \\ 0 & \text{Otherwise.} \end{cases}$$

Linear kernel Binary features are used within a linear kernel (i.e., dot product) as defined in Equation 1.

Note that, we normalize the computed kernels using the cosine similarity modifier given by,

$$K(x, y)' = \frac{K(x, y)}{\sqrt{K(x, x)}\sqrt{K(y, y)}} \tag{7}$$

3.3 Argument detection

First, we generate features for all shortest dependency paths between predicted trigger and named entity. Then, we define a kernel based similarity computation which is able to capture the argument detection task specific similarity between shortest path features. Each shortest path example is classified as belonging to one of the argument type classes (theme or cause) or as negative.

Like the trigger detection, we define a kernel function that depends on string feature values presented in the shortest path vectors. The similarity matrix is defined as follows,

$$\langle x_i | x_j \rangle_{MAT} = \sum_{k=1}^{l} S(i_k, j_k) \tag{8}$$

where k = {**dependency n-gram**, **dependency path**, **numerical**, **semantic**}.

We select for each group of feature an appropriate kernel by summing up the following kernel functions:

- Dependency n-gram: we use the k-spectrum kernel between the n-grams of the two shortest paths.
- Dependency path: we use the dependency kernel described above in the trigger detection to calculate the similarity between the dependency edges relative to the two shortest paths.

Type	Feature
Frequency features	Length of the shortest path between two entities
	Number of named entities and event triggers per type in the sentence
N-grams features	N-grams of dependencies (n = 2, 3, 4)
	N-grams of consecutive words
Terminus token features	Trigger / Argument word
	Trigger / Argument type
	Trigger / Argument POS
	Confidence scores of terminus tokens obtained by trigger detection
Single element features	Directions of dependency edges relative to the shortest path
	Types of dependency edges relative to the shortest path
Semantic features	Annotation label of the shortest path
	Combination of the specific type of the terminus token of the shortest path and their categories

TAB. 2 – *Features for argument detection.*

– Numerical: the cosine similarity is a simple appropriate kernel to calculate the similarity between two numerical vectors defined as,

$$K(A,B) = \frac{\sum_{i=1}^{n} A_i \cdot B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \cdot \sqrt{\sum_{i=1}^{n}(B_i)^2}} \tag{9}$$

– Semantic: we use the Edit distance kernel (Tikk and al., 2010) which calculates the similarity between the annotation labels of the shortest paths.

Note that our kernels for trigger and argument detection are constructed from existing valid ones (i.e, satisfying Mercer's theorem) using this rule $k(x_1, x_2) = k_1(x_1, x_2) + k_2(x_1, x_2)$ which means that the summation of the two valid kernels k_1 and k_2 results a valid kernel k. Hence, our kernels are satisfying the Mercer's theorem.

3.4 Argument Grouping

The target output of the argument detection is in the form of a primary frame consisting of an event class, semantic role and participants (protein or event). For argument grouping, we need to find the best combinations of event frames that are detected by the argument detector to represent complex events (i.e., binding and regulation). We construct classification models for the complex event detection. First, we design features of a complex event candidate for complex event detection that constrain the event argument types and combinations defined in the event ontology. The features contain three relations, (1) relations between arguments, (2) relations between triggers and outer proteins, (3) and relations between arguments and outer triggers. Hence, we apply the feature-based argument extractor as shown in Table 2 for three types of substructures: each shortest path in the complex event, all pairs among arguments, all shortest paths including event trigger outside of events, all pairs between argument proteins and their closest proteins in binding event.

The first relations are used to remove candidates that contain non-related arguments, and the second and third relations are used to remove candidates by finding the shortest paths that should be included in the candidates and more appropriate combinations of event arguments.

4 Experimentation

We present the implementation of our proposed approach to solve the event extraction task with the BioNLP'09 Shared task [3] resources (Kim and al., 2009).

The experimental data set are prepared based on the GENIA corpus in the context of the BioNLP Shared Task. They consist of PubMed documents (title and abstract only).We count several items of data presented in Table 3 (Kim and al., 2009).

	Training	Development	Test
Abstract	800	150	260
Sentence	7499	1450	2447
Token	176146	33937	57367
Event	8597	1809	3182

TAB. 3 – *Types and statistics of experimental data sets (Kim and al., 2009).*

The event extraction pipeline consists of four major parts, a pre-processor, a trigger detector, an argument detector and a complex event detector.

The documents of the training, development and testing data set from the GENIA corpus are segmented and tokenized using the GENIA sentence splitter and the GENIA tagger provided by U-Compare [4]. Then, we run the McClosky-Charniak domain-adapted parser (McClosky and Charniak, 2008). The output of the parser is provided in the standard Penn Treebank (PTB) format. Finally, we annotate the semantic class for each term using WordNet and the UMLS Metathesaurus. Gene, protein, RNA, cell line and cell type names are identified by ABNER [5]. Then, the trigger detector proceed with filtering out candidate triggers to remove 70% of tokens, trigger-based features extraction methods and finally, training and testing trigger classification using the LIBSVM software (Chang and Lin, 2010). We implement these methods in JAVA using Eclipse. As same, we develop the argument detector and complex argument detector methods. Finally, our kernel functions are added to the LIBSVM for evaluation.

The results of the experiments carried out with the testing data in terms of recall, precision and f-score (Kim and al., 2009) are shown in Table 4.

Then, we obtain a significant value of f-score compared to the UTURKU system (Björne and al., 2009), JULIE system (Buyko and al., 2009) and CCP-BTMG system.

3. http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/SharedTask/
4. http://u-compare.org/
5. http://pages.cs.wisc.edu/bsettles/abner/

	Recall (%)	Precision (%)	F-score (%)
UTURKU system	46.73	58.48	51.95
JulieLab system	35.72	30.36	32.82
CCP-BTMG system	13.45	**71.81**	22.66
BioEv system	**50.57**	64.88	**56.83**

TAB. 4 – *Experimental results for event extraction (precision / recall / f-score).*

5 Conclusion

In the previous decade of work on automatic information extraction from biological texts, efforts have focused in particular on the basic task of recognizing entity names in text and on the extraction of relations of these entities and, more recently, on the biological event extraction.

In our work, we propose an event extraction approach using support vector machines and composite kernel function. We start processing texts by analyzing natural language documents using lexical resources to obtain sentences, tokens and POS tags. Then, tokens are organized into groups after a syntactic and semantic analysis has assigned meaning to these tokens or groups of tokens. In the trigger and argument detection phase, we extract feature vectors for training and testing using a SVM modeling. We combine multiple layers of syntactic and semantic information by applying distinct kernels on features. The combination of distinct kernels is achieved through summing the values of each kernel for each type of feature.

In order to evaluate our approach we implement our event extraction system. We obtain a recall around 50.57%, a precision around 64.88% and an f-score around **56.83%**, for a set of GENIA abstracts.

Our first future work consists of evaluating the performance of our approach on the GENIA full text articles, different corpora such as BioInfer corpus and comparing our composite kernels to benchmarks of various kernels.

Another line of research will be to exploit the event extraction output in text mining tasks such as event network analysis, hypothesis generation, pathway extraction and others.

References

Ananiadou, S., D. B. Kell, and J. ichi Tsujii (2006). Text mining and its potential applications in systems biology. *TRENDS in Biotechnology 14*.

Björne, J., F. Ginter, S. Pyysalo, J. Tsujii, , and T. Salakoski (2008). Complex event extraction at pubmed scale. *BMC Bioinformatics*, 1–25.

Björne, J., J. Heimonen, F. Ginter, A. Airola, T. Pahikkala, and T. Salakoski (2009). Extracting complex biological events with rich graph-based feature sets. In *Proceedings of the workshop on BioNLP: Shared Task*, pp. 10–18.

Blaschke, C., M. Andrade, C. Ouzounis, and A. Valencia (1999). Automatic extraction of biological information from scientific text: protein-protein interactions. In *Proceedings of*

the Seventh International Conference on Intelligent Systems for Molecular Biology, pp. 60–67.

Burges, C. (1998). A tutorial on Support Vector Machines for pattern recognition. *Data Mining and Knowledge Discovery*.

Buyko, E., E. Faessler, J. Wermter, and U. Hahn (2009). Event extraction from trimmed dependency graphs. In *Proceedings of the workshop on BioNLP: Shared Task*, pp. 19–27.

Chang, C.-C. and C.-J. Lin (2010). LIBSVM : a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*.

Cohen, K. B., K. Verspoor, H. L. Johnson, C. Roeder, P. V. Ogren, W. A. B. Jr., E. White, H. Tipney, , and L. Hunter (2009). High-precision biological event extraction with a concept recognizer. In *Proceedings of the Workshop on BioNLP: Shared Task*, pp. 50–58.

Elkhlifi, A. and R. Faiz (2007). Machine learning approach for the automatic annotation of the events. In D. Wilson and G. Sutcliffe (Eds.), *FLAIRS Conference, the Proceedings of the Twentieth International Florida Artificial Intelligence Research Society Conference* (AAAI Press ed.)., pp. 362–367.

Elkhlifi, A. and R. Faiz (2009). Automatic annotation approach of events in news articles. *International Journal of Computing and Information Sciences (IJCIS) 7*, 40–50.

Elkhlifi, A. and R. Faiz (2010a). Event extraction approach for web 2.0. *ACS/IEEE International Conference on Computer Systems and Applications - AICCSA 2010*, 1–8.

Elkhlifi, A. and R. Faiz (2010b). French-written event extraction based on contextual exploration. In H. W. Guesgen and R. C. Murray (Eds.), *Proceedings of the Twenty-First International Florida Artificial Intelligence Research Society Conference* (AAAI ed.)., pp. 180–185.

Hirst, G. and D. St-Onge (1998). Lexical chains as representation of context for the detection and correction malapropisms. *WordNet: An Electronic Lexical Database*, 305–332.

Hsu, C. W., C. C. Chang, and C. J. Lin (2010). A practical guide to support vector classification. *Bioinformatics 1*, 1–16.

Kim, J. D., T. Ohta, S. Pyysalo, Y. Kano, and J. Tsujii (2009). Overview of BioNLP'09 Shared Task on event extraction. In *Proceedings of the workshop on BioNLP: Shared Task*, pp. 1–9.

Kim, J. D., T. Ohta, and J. Tsujii (2008). Corpus annotation for mining biomedical events from literature. *BMC Bioinformatics*.

Kim, S., J. Yoon, , and J. Yang (2008). Kernel approaches for genic interaction extraction. *Bioinformatics 24*, 118–126.

Leslie, C., E. Eskin, and W. S. Noble (2002). The specrum kernel: A string kernel for SVM protein classification. In *Pacific Symposium on Biocomputing 7*, pp. 566–575.

McCarthy, J. F., C. Brodley, J. Clouse, B. Crites, D. Mammen, E. Brown, J. Callan, A. Diwan, and rian Pinnette (1996). A trainable approach to coreference resolution for information extraction.

McClosky, D. and E. Charniak (2008). Selftraining for biomedical parsing. In *Proceedings of ACL-08: HLT*, pp. 101–104.

Mooney, R. J. and R. Bunescu (2005). Mining knowledge from text using information extrac-

tion. *SIGKDD Explorations (special issue on Text Mining and Natural Language Processing)*, 3–10.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, 130–137.

Rindflesch, T., L. Tanabe, J. Weinstein, and L. Hunter (2000). Edgar: Extraction of drugs, genes and relations from the biomedical literature. In *Proceedings Pacific Symposium on Biocomputing*, pp. 517–528.

Tikk, D., P. Thomas, P. Palaga, J. Hakenberg, and U. Leser (2010). A comprehensive benchmark of kernel methods to extract proteinÚprotein interactions from literature. *PLoS Computational Biology*, 1–19.

Vlachos, A., P. Buttery, D. O. Seaghdha, and T. Briscoe (2009). Biomedical event extraction without training data. In *Proceedings of the workshop on BioNLP: Shared Task*, pp. 37–40.

Résumé

Étant donné l'importance des résultats scientifiques publiés dans le domaine de la biologie moléculaire, il est nécessaire de mettre en place des outils aidant les scientifiques à extraire des informations précises et structurées. L'extraction des entités nommées telles que les protéines, les gènes et les relations simples telles que les interactions entre les protéines, les associations entre les gènes et les maladies sont des taches quasiment résolues.

Désormais, les recherches en extraction d'information biologique se sont orientées à la résolution des taches plus complexes telles que la coréférence et l'extraction des événements. Dans cet article, nous nous intéressons au particulier à l'extraction des représentations complexes des événements biologiques. Une représentation d'un événement biologique dans un langage naturel comprend une ou plusieurs unités linguistiques désignant un déclencheur d'événement et des entités biologiques participantes à cet événement.

Dans cet article, nous avons présenté notre approche d'extraction des événements biologiques qui utilise les machines à vecteurs de support (SVM) et une fonction noyau composite afin d'identifier les déclencheurs des évènements (event trigger) et les participants à cet événement (event argument). Aussi, nous avons utilisé un nombre d'attributs basés sur les informations syntaxiques et contextuelles, générés automatiquement à partir des documents d'apprentissage.

Nous avons implémenté notre système d'extraction des événements en utilisant des outils de traitement automatique de langage naturel (TALN). Nous avons obtenu des résultats compétitifs par rapport au benchmark de BioNLP'09 Shared Task.