Automatic Knowledge Representation using a Graph-based Algorithm for Language-Independent Lexical Chaining

Gaël Dias
HULTIG
University of Beira Interior
Covilhã, Portugal
ddg@di.ubi.pt

Cláudia Santos
HULTIG
University of Beira Interior
Covilhã, Portugal
claudia@dmnet.ubi.pt

Guillaume Cleuziou
LIFO
University of Orléans
Orléans, France
cleuziou@univ-orleans.fr

Abstract

Lexical Chains are powerful representations of documents. In particular, they have successfully been used in the field of Automatic Text Summarization. However, until now, Lexical Chaining algorithms have only been proposed for English. In this paper, we propose a greedy Language-Independent algorithm that automatically extracts Lexical Chains from texts. For that purpose, we build a hierarchical lexico-semantic knowledge base from a collection of texts by using the Pole-Based Overlapping Clustering Algorithm. As a consequence, our methodology can be applied to any language and proposes a solution to language-dependent Lexical Chainers.

1 Introduction

Lexical Chains are powerful representations of documents compared to broadly used bag-of-words representations. In particular, they have successfully been used in the field of Automatic Text Summarization (Barzilay and Elhadad, 1997). However, until now, Lexical Chaining algorithms have only been proposed for English as they rely on linguistic resources such as Thesauri (Morris and Hirst, 1991) or Ontologies (Barzilay and Elhadad, 1997; Hirst and St-Onge, 1997; Silber and McCoy, 2002; Galley and McKeown, 2003).

Morris and Hirst (1991) were the first to propose the concept of Lexical Chains to explore the discourse structure of a text. However, at the time of writing their paper, no machine-readable thesaurus was available so they manually generated Lexical Chains using Roget's Thesaurus (Roget, 1852).

A first computational model of Lexical Chains is introduced by Hirst and St-Onge (1997). Their biggest contribution to the study of Lexical Chains is the mapping of WordNet (Miller, 1995) relations and paths (transitive relationships) to (Morris and Hirst, 1991) word relationship types. However, their greedy algorithm does not use a part-of-speech tagger. Instead, the algorithm only selects those words that contain noun entries in WordNet to compute Lexical Chains. But, as Barzilay and Elhadad (1997) point at, the use of a part-of-speech tagger could eliminate wrong inclusions of words such as *read*, which has both noun and verb entries in WordNet.

So, Barzilay and Elhadad (1997) propose the first dynamic method to compute Lexical Chains. They argue that the most appropriate sense of a word can only be chosen after examining all possible Lexical Chain combinations that can be generated from a text. Because all possible senses of the word are not taken into account, except at the time of insertion, potentially pertinent context information that is likely to appear after the word is lost. However, this method of retaining all possible interpretations until the end of the process, causes the exponential growth of the time and space complexity.

As a consequence, Silber and McCoy (2002) propose a linear time version of (Barzilay and Elhadad, 1997) lexical chaining algorithm. In particular, (Silber and McCoy, 2002)'s implementation creates a structure, called meta-chains, that implicitly stores all chain interpretations without actually creating them, thus keeping both the space and time usage of the program linear.

Finally, Galley and McKeown (2003) propose a chaining method that disambiguates nouns prior to the processing of Lexical Chains. Their evaluation shows that their algorithm is more accurate than (Barzilay and Elhadad, 1997) and (Silber and McCoy, 2002) ones.

One common point of all these works is that Lexical Chains are built using WordNet as the standard linguistic resource. Unfortunately, systems based on static linguistic knowledge bases are limited. First, such resources are difficult to find. Second, they are largely obsolete by the time they are available. Third, linguistic resources capture a particular form of lexical knowledge which is often very different from the sort needed to specifically relate words or sentences. In particular, WordNet is missing a lot of explicit links between intuitively related words. Fellbaum (1998) refers to such obvious omissions in WordNet as the “tennis problem” where nouns such as *nets*, *rackets* and *umpires* are all present, but WordNet provides no links between these related tennis concepts.

In order to solve these problems, we propose to automatically construct from a collection of documents a lexico-semantic knowledge base with the purpose to identify cohesive lexical relationships between words based on corpus evidence. This hierarchical lexico-semantic knowledge base is built by using the Pole-Based Overlapping Clustering Algorithm (Cleuziou et al., 2004) that clusters words with similar meanings and allows words with multiple meanings to belong to different clusters. The second step of the process aims at automatically extracting Lexical Chains from texts based on our knowledge base. For that purpose, we propose a new greedy algorithm which can be seen as an extension of (Hirst and St-Onge, 1997) and (Barzilay and Elhadad, 1997) algorithms which allows polysemous words to belong to different chains thus breaking the “one-word/one-concept per document” paradigm (Gale et al., 1992)[1]. In particular, it implements (Lin, 1998) information-theoretic definition of similarity as the relatedness criterion for the attribution of words to Lexical Chains[2].

2 Building a Similarity Matrix

In order to build the lexico-semantic knowledge base, the Pole-Based Overlapping Clustering Algorithm needs as input a similarity matrix that gathers the similarities between all the words in the corpus. For that purpose, we propose a contextual analysis of each nominal unit (nouns and compound nouns) in the corpus. In particular, each nominal unit is associated to a word context vector and the similarity between nominal units is calculated by the informative similarity measure proposed by (Dias and Alves, 2005).

2.1 Data Preparation

The context corpus is first pre-processed in order to extract nominal units from it. The TnT tagger (Brants, 2000) is first applied to our context corpus to morpho-syntactically mark all the words in it. Once all words have been morpho-syntactically tagged, we apply the statistically-based multiword unit extractor SENTA (Dias et al., 1999) that extracts multiword units based on any input text[3]. For example, multiword units are compound nouns (*free kick*), compound determinants (*an amount of*), verbal locutions (*to put forward*), adjectival locutions (*dark blue*) or institutionalized phrases (*con carne*). Finally, we use a set of well-known heuristics (Daille, 1995) to retrieve compound nouns using the idea that groups of words that correspond to a priori defined syntactical patterns such as *Adj+Noun, Noun+Noun, Noun+Prep+Noun* can be identified as compound nouns. Indeed, nouns usually convey most of the information in a written text. They are the main contributors to the “aboutness” of a text. For example, *free kick, city hall, operating system* are compound nouns which sense is not compositional i.e. the sense of the multiword unit can

[1] This characteristic can be interesting for multi-topic documents like web news stories. Indeed, in this case, there may be different topics in the same document as different news stories may appear. In some way, it follows the idea of (Krovetz, 1998).

[2] Of course, other similarity measures (Resnik, 1995; Jiang and Conrath, 1997; Leacock and Chodorow, 1998) could be implemented and should be evaluated in further work. However, we used (Lin, 1998) similarity measure as it has shown improved results for Lexical Chains construction.

[3] By choosing both the TnT tagger and the multiword unit extractor SENTA, we guarantee that our architecture remains as language-independent as possible.

not be expressed by the sum of its constituents senses. So, identifying lexico-semantic connections between nouns is an adequate means of determining cohesive ties between textual units[4].

2.2 Word Context Vectors

The similarity matrix is a matrix where each cell corresponds to a similarity value between two nominal units[5]. In this paper, we propose a contextual analysis of nominal units based on similarity between word context vectors.

Word context vectors are an automated method for representing information based on the local context of words in texts. So, for each nominal unit in the corpus, we associate an N-dimension vector consisting of its N most related words[6].

In order to find the most relevant co-occurrent nominal units, we implement the Symmetric Conditional Probability (Silva et al., 1999) which is defined in Equation 1 where $p(w_1, w_2)$, $p(w_1)$ and $p(w_2)$ are respectively the probability of co-occurrence of the nominal units w_1 and w_2 and the marginal probabilities of w_1 and w_2.

$$SCP(w_1, w_2) = \frac{p(w_1, w_2)^2}{p(w_1) \times p(w_2)} \qquad (1)$$

In particular, the window context for the calculation of co-occurrence probabilities is settled to F=20 words. In fact, we count, in all the texts of the corpus, the number of occurrences of w_1 and w_2 appearing together in a window context of $F - 2$ words. So, $p(w_1, w_2)$ represents the density function computed as follows: the number of times w_1 and w_2 co-occur divided by the number of words in the corpus[7]. In the present work, the values of the $SCP(.,.)$ are not used as a factor of importance between words in the word context vector i.e. no differentiation is made in terms of relevance between the words within the word context vector. This issue will be tackled in future work[8].

2.3 Similarity between Context Vectors

The closeness of vectors in the space is equivalent to the closeness of the subject content. Thus, nominal units that are used in a similar local context will have vectors that are relatively close to each other. However, in order to define similarities between vectors, we must transform each word context vector into a high dimensional vector consisting of real-valued components. As a consequence, each co-occurring word of the word context vector is associated to a weight which evaluates its importance in the corpus.

2.3.1 Weighting score

The weighting score of any word in a document can be directly derived from an adaptation of the score proposed in (Dias and Alves, 2005). In particular, we consider the combination of two main heuristics: the well-known *tf.idf* measure proposed by (Salton et al., 1975) and a new density measure (Dias and Alves, 2005).

tf.idf: Given a word w and a document d, the $tf.idf(w, d)$ is defined in Equation 2 where $tf(w, d)$ is the number of occurrences of w in d, $|d|$ corresponds to the number of words in d, N is the number of documents in the corpus and $df(w)$ stands for the number of documents in the corpus in which the word w occurs.

$$tf.idf(w, d) = \frac{tf(w, d)}{|d|} \times \log_2 \left(\frac{N}{df(w)} \right) \qquad (2)$$

density: The basic idea of the word density measure is to evaluate the dispersion of a word within a document. So, very disperse words will not be as relevant as dense words. This density measure $dens(.,.)$ is defined in Equation 3.

$$dens(w, d) = \sum_{k=1}^{tf(w,d)-1} \frac{1}{\ln(dist(o_{(w,k)}, o_{(w,k+1)}) + e)} \qquad (3)$$

For any given word w, its density $dens(w, d)$ is calculated from all the distances between all its occurrences in document d, $tf(w, d)$. So, $dist(o_{(w,k)}, o_{(w,k+1)})$ calculates the distance that separates two consecutive occurrences of w in terms of words within the document. In particular, e is the

[4] However, we acknowledge that verbs and adjectives should also be tackled in future work.

[5] Many works have been proposed on word similarity (Lin, 1998).

[6] In our experiments, N=10.

[7] We note that multiword units are counted as single words as when they are identified (e.g. President of the United States), they are re-written in the corpus by linking all single words with an underscore (e.g. President_of_the_United_States)

[8] We may point at the fact that satisfying results were obtained by the Symmetric Conditional Probability measure compared to the Pointwise Mutual Information for instance (Cleuziou et al., 2003)

base of the natural logarithm so that $ln(e) = 1$. This argument is included into Equation 3 as it will give a density value of 1 for any word that only occurs once in the document. In fact, we give this word a high density value.

final weight: The weighting score $weight(w)$ of any word w in the corpus can be directly derived from the previous two heuristics. This score is defined in Equation 4 where $\overline{tf}$ and $\overline{dens}$ are respectively the average of $tf(.,.)$ and $dens(.,.)$ over all the documents in which the word w occurs i.e. N_w.

$$weight(w) = \overline{tf}.idf(w) \times \overline{dens}(w) \tag{4}$$

where $\overline{tf} = \frac{\sum_d tf(w,d)}{N_w}$ and $\overline{dens}(w) = \frac{\sum_d dens(w,d)}{N_w}$

2.3.2 Informative Similarity Measure

The next step aims at determining the similarity between all nominal units. Theoretically, a similarity measure can be defined as follows. Suppose that $X_i = (X_{i1}, X_{i2}, X_{i3}, , X_{ip})$ is a row vector of observations on p variables associated with a label i. The similarity between two words i and j is defined as $S_{ij} = f(X_i, X_j)$ where f is some function of the observed values. In the context of our work, X_i and X_j are 10-dimension word context vectors.

In order to avoid the lexical repetition problem of similarity measures, (Dias and Alves, 2005) have proposed an informative similarity measure called $infoSimBA$, which basic idea is to integrate into the Cosine measure, the word co-occurrence factor inferred from a collection of documents with the Symmetric Conditional Probability (Silva et al., 1999). See Equation 5.

$$InfoSimBA(X_i, X_j) = \frac{A_{ij}}{B_i \times B_j + A_{ij}} \tag{5}$$

where

$$\begin{aligned} A_{ij} &= \sum_{k=1}^{p}\sum_{l=1}^{p} X_{ik} \times X_{jl} \times SCP(w_{ik}, w_{jl}) \\ \forall i, B_i &= \sqrt{\sum_{k=1}^{p}\sum_{l=1}^{p} X_{ik} \times X_{il} \times \mathrm{SCP}\left(w_{ik}, w_{il}\right)} \end{aligned}$$

and any X_{zv} corresponds to the word weighting factor $weight(w_{zv})$, $SCP(w_{ik}, w_{jl})$ is the Symmetric Conditional Probability value between w_{ik}, the word that indexes the word context vector i at position k and w_{jl}, the word that indexes the word context vector j at position l.

In particular, this similarity measure has proved to lead to better results compared to the classical similarity measure (Cosine) and shares the same idea as the Latent Semantic Analysis (LSA) but in a different manner. Let's consider the following two sentences.

(1) Ronaldo defeated the goalkeeper once more.

(2) Real_Madrid_striker scored again.

It is clear that both sentences (1) and (2) are similar although they do not share any word in common. Such a situation would result in a null Cosine value so evidencing no relationship between (1) and (2). To solve this problem, the $InfoSimBA(.,.)$ function would calculate for each word in sentence (2), the product of its weight with each weight of all the words in sentence (1), and would then multiply this product by the degree of cohesiveness existing between those two words calculated by the Symmetric Conditional Probability measure. For example, *Real_Madrid_striker* would give rise to the sum of 6 products i.e. *Real_Madrid_striker* with *Ronaldo*, *Real_Madrid_striker* with *defeated* and so on and so forth. As a consequence, sentence (1) and (2) would show a high similarity as *Real_Madrid_striker* is highly related to *Ronaldo*.

Once the similarity matrix is built based on the $infoSimBA$ between all word context vectors of all nominal units in the corpus, we give it as input to the Pole-Based Overlapping Clustering Algorithm (Cleuziou et al., 2004) to build a hierarchy of concepts i.e. our lexico-semantic knowledge base.

3 Hierarchy of Concepts

Clustering is the task that structures units in such a way it reflects the semantic relations existing between them. In our framework nominal units are first grouped into overlapping clusters (or soft-clusters) such that final clusters correspond to conceptual classes (called "concepts" in the following). Then, concepts are hierarchically structured in order to capture semantic links between them.

Many clustering methods have been proposed in the data analysis research fields. Few of them propose overlapping clusters as output, in spite of the interest it represents for domains of application such as Natural Language Processing or Bioinformatics. PoBOC (*Pole-Based Overlapping Clustering*) (Cleuziou et al., 2004) and CBC (*Clustering By Committees*) (Pantel and Lin, 2002) are two clustering algorithms suitable for the word clustering task. They both proceed by first constructing tight clusters[9] and then assigning residual objects to their most similar tight clusters.

A recent comparative study (Cicurel et al., 2006) shows that CBC and PoBOC both lead to relevant results for the task of word clustering. Nevertheless CBC requires parameters hard to tune whereas PoBOC is free of any parametrization. The last argument encouraged us to use the PoBOC algorithm.

Unlike most of commonly used clustering algorithms, the Pole-Based Overlapping Clustering Algorithm shows the following advantages among others : (1) it requires no parameters i.e. input is restricted to a single similarity matrix, (2) the number of final clusters is automatically found and (3) it provides overlapping clusters allowing to take into account the different possible meanings of lexical units.

3.1 A Graph-based Approach

The Pole-Based Overlapping Clustering Algorithm is based on a graph-theoretical framework. Graph formalism is often used in the context of clustering (graph-clustering). It first consists in defining a graph structure which illustrates the data (vertices) with links (edges) between them and then in proposing a graph-partitioning process.

Numerous graph structures have been proposed (Estivill-Castro et al., 2001). They all consider the data set as set of vertices but differ on the way to decide that two vertices are connected. Some methodologies are listed below where V is the set of vertices, E the set of edges, $G(V, E)$ a graph and d a distance measure:

- Nearest Neighbor Graph (NNG) : each vertex is connected to its nearest neighbor,
- Minimum Spanning Tree (MST) : $\forall (x_i, x_j) \in V \times V$ a path exists between x_i and x_j in G with $\sum_{(x_i,x_j)\in E} d(x_i, x_j)$ minimized,
- Relative Neighborhood Graph (RNG) : x_i and x_j are connected iff $\forall x_k \in V \setminus \{x_i, x_j\}$, $d(x_i, x_j) \leq \max\{d(x_i, x_k), d(x_j, x_k)\}$
- Gabriel Graph (GG) : x_i and x_j are connected iff the circle with diameter $\overline{x_i x_j}$ is empty,
- Delaunay Triangulation (DT) : x_i and x_j are connected iff the associated Voronoi cells are adjacent.

In particular, an inclusion order exists on these graphs. One can show that $NNG \subseteq MST \subseteq RNG \subseteq GG \subseteq DT$.

The choice of the suitable graph structure depends on the expressiveness we want an edge to capture and the partitioning process we plan to perform. The Pole-Based Overlapping Clustering Algorithm aims at retrieving dense subsets in *a graph where two similar data are connected and two dissimilar ones are disconnected.* Noticing that previous structures do not match with this definition of a proximity-graph[10], a new variant is proposed with the Pole-Based Overlapping Clustering Algorithm in definition 3.1.

Definition 3.1 *Given a similarity measure s on a data set X, the graph (denoted $G_s(V, E)$) is defined by the set of vertices $V = X$ and the set of edges E such that* $(x_i, x_j) \in E \Leftrightarrow x_i \in \mathcal{N}(x_j) \wedge x_j \in \mathcal{N}(x_i)$.

In particular, $\mathcal{N}(x_i)$ corresponds to the local neighborhood of x_i built as in equation 6.

$$\mathcal{N}(x_i) = \{x_j \in X | s(x_i, x_j) > s(x_i, X)\} \qquad (6)$$

where the notation $s(x_i, I)$ denotes the average similarity of x_i with the set of objects I i.e.

$$\sum_{x_k \in I} \frac{s(x_i, x_k)}{|I|} \qquad (7)$$

This definition of neighborhood is a way to avoid requiring to a parameter that would be too dependent of the similarity used. Furthermore, the use of local neighborhoods avoids the use of arbitrary thresholds which mask the variations of densities. Indeed, clusters are extracted from a similarity graph which differs from traditional proximity graphs (Jaromczyk and Toussaint, 1992) in the definition of local

[9] The tight clusters are called “committees” in CBC and “poles” in PoBOC.

[10] Indeed, for instance, all of these graphs connect an outlier with at least one other vertex. This is not the case with PoBOC.

neighborhoods which condition edges in the graph. Neighborhood is different for each object and is computed on the basis of similarities with all other objects. Finally, an edge connects two vertices if they are both contained in the neighborhood of the other one. Figure 1 illustrates the neighborhood constraint above. In this case, as x_i and x_j are not both in the intersection, they would not be connected.

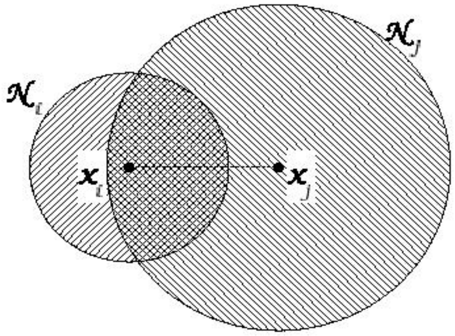

Figure 1: To be connected, both x_i and x_j must be in the intersection.

3.2 Discovery of Poles

The graph representation helps to discover a set of fully-connected subgraphs (cliques) highly separated, denoted as Poles. Because $G_s(V,E)$ is built such that two vertices x_i and x_j are connected if and only if they are similar[11], a clique has the required properties to be a good cluster. Indeed, such a cluster guarantees that all its constituents are similar.

The search of maximal cliques in a graph is an NP-complete problem. As a consequence, heuristics are used in order to (1) build a great clique around a starting vertex (Bomze et al., 1999) and (2) choose the starting vertices in such a way cliques are as distant as possible.

Given a starting vertex x, the first heuristic consists in adding iteratively the vertex x_i which satisfies the following conditions:

- x_i is connected to each vertex in P (with P the clique/Pole in construction),
- among the connected vertices, x_i is the nearest one in average ($s(x_i, P)$).

As a consequence, initialized with $P = \{x\}$, the clique then grows until no vertex can be added.

The second heuristic guides the selection of the starting vertices in a simple manner. Given a set of Poles $P_1, \ldots, P_m$ already extracted, we select the vertex x as in Equation 8.

$$s(x, P_1 \cup \cdots \cup P_m) = \min_{x_i} s(x_i, P_1 \cup \cdots \cup P_m) \quad (8)$$

A new Pole is then built from x if and only if x satisfies the following conditions:

- $\forall k \in \{1, \ldots, m\}\ ,\ x \notin P_k$,
- $s(x, P_1 \cup \cdots \cup P_m) < s(X, X) = \frac{1}{|X|^2} \sum_{x_i} \sum_{x_j} s(x_i, x_j)$

Poles are thus extracted while $P_1 \cup \cdots \cup P_m \neq X$ and the next starting vertex x is far enough from the previous Poles. In particular, as Poles represent the seeds of the further final clusters, this heuristic gives no restriction on the number of clusters. The first Pole is obtained from the starting point x^* that checks Equation 9.

$$x^* = \arg\min_{x_k \in X} s(x_k, X) \quad (9)$$

3.3 Multi-Assignment

Once the Poles are built, the Pole-Based Overlapping Clustering algorithm uses them as clusters representatives. Membership functions $m(.,.)$ are defined in order to assign each object to its nearest Poles as shown in Equation 10.

$$\forall x_i \in X,\ P_j \in \{P_1, \ldots, P_m\}\ :\ m(x_i, P_j) = s(x_i, P_j) \quad (10)$$

For each object x_i to assign, the set of poles is ordered $(P_1(x_i), \ldots, P_m(x_i))$ such that $P_1(x_i)$ denotes the nearest pole[12] for x_i, $P_2(x_i)$ the second nearest pole for x_i and so on. We first assign x_i to its closest Pole ($P_1(x_i)$). Then, for each pole $P_k(x_i)$(in the order previously defined) we decide to assign x_i to $P_k(x_i)$ if it satisfies to the following two conditions :

- $\forall k' < k,\ x_i$ is assigned to $P_{k'}(x_i)$,
- if $k < m$,

$$s(x_i, P_k(x_i)) \geq \frac{s(x_i, P_{k-1}(x_i)) + s(x_i, P_{k+1}(x_i))}{2}$$

This methodology results into a coverage of the starting data set with overlapping clusters (extended Poles).

[11] In the sense that x_i (resp. x_j) is more similar to x_j (resp. x_i) than to other data on average.

[12] $P_1(x_i) = \arg\max_{P_j} s(x_i, P_j)$

3.4 Hierarchical Organization

A final step consists in organizing the obtained clusters into a hierarchical tree. This structure is useful to catch the topology of a set of a priori disconnected groups. The Pole-Based Overlapping Clustering algorithm integrates this stage and proceeds by successive merging of the two nearest clusters like for usual agglomerative approaches (Sneath and Sokal, 1973). In this process, the similarity between two clusters is obtained by the average-link (or complete-link) method:

$$s(I_p, I_q) = \frac{1}{|I_p|.|I_q|} \sum_{x_i \in I_p} \sum_{x_j \in I_q} s(x_i, x_j) \tag{11}$$

To deal with overlapping clusters we considere in Equation 11 the similarity between an object and itself to be equal to 1 : $s(x_i, x_i) = 1$.

4 Lexical Chaining Algorithm

Once the lexico-semantic knowledge base has been built, it is possible to use it for Lexical Chaining. In this section, we propose a new greedy algorithm which can be seen as an extension of (Hirst and St-Onge, 1997) and (Barzilay and Elhadad, 1997) algorithms as it allows polysemous words to belong to different chains thus breaking the "one-word/one-concept per document" paradigm (Gale et al., 1992). Indeed, multi-topic documents like web news stories may introduce different topics in the same document/url and do not respect the "one sense per discourse" paradigm. As we want to deal with real-world applications, this characteristic may show interesting results for the specific task of Text Summarization for Web documents. Indeed, comparatively to the experiments made by (Gale et al., 1992) that deal with "well written discourse", web documents show unusual discourse structures. In some way, our algorithm follows the idea of (Krovetz, 1998). Finally, it implements (Lin, 1998)'s information-theoretic definition of similarity as the relatedness criterion for the attribution of words to Lexical Chains.

4.1 Algorithm

Our chaining algorithm is based on both approaches of (Barzilay and Elhadad, 1997) and (Hirst and St-Onge, 1997). So, our chaining model is developed according to all possible alternatives of word senses. In fact, all senses of a word are defined by the clusters the word appears in[13]. We present our algorithm below.

```
Begin with no chain.
 For all distinct nominal units in text order do
  For all its senses do
    a) - among present chains find the sense
         which satisfies the relatedness
         criterion and link the new word to
         this chain.
       - Remove unappropriate senses of the
         new word and the chain members.
    b)if no sense is close enough, start a new chain.
  End For
 End For
End
```

4.2 Assignment of a word to a Lexical Chain

In order to assign a word to a given Lexical Chain, we need to evaluate the degree of relatedness of the given word to the words in the chain. This is done by evaluating the relatedness between all the clusters present in the Lexical Chain and all the clusters in which the word appears.

4.2.1 Scoring Function

In order to determine if two clusters are semantically related, we use our lexico-semantic knowledge base and apply (Lin, 1998)'s measure of semantic similarity defined in Equation 12.

$$simLin(C_1, C_2) = \frac{2 \times \log P(C_0)}{\log P(C_1) + \log P(C_2)} \tag{12}$$

The computation of Equation 12 is illustrated below using the fragment of WordNet in Figure 2.

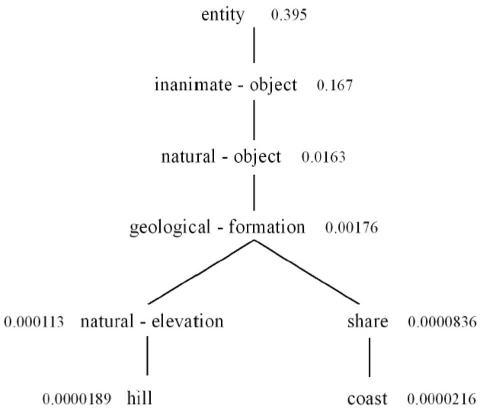

Figure 2: Fragment of WordNet (Lin, 1998).

[13] From now on, for presentation purposes, we will take as synonymous the words *clusters* and *senses*

In this case, it would be easy to compute the similarity between the concepts of *hill* and *coast* where the number attached to each node C is $P(C)$. It is shown in Equation 13.

$$simLin(hill, coast) = \frac{2 \log P(geological - formation)}{\log P(hill) + \log P(coast)} = 0.59 \quad (13)$$

However, in our taxonomy, as in any knowledge base computed by hierarchical clustering algorithms, only leaves contain words. So, upper clusters (i.e. nodes) in the taxonomy gather all distinct words that appear in the clusters they subsume. We present this situation in Figure 3.

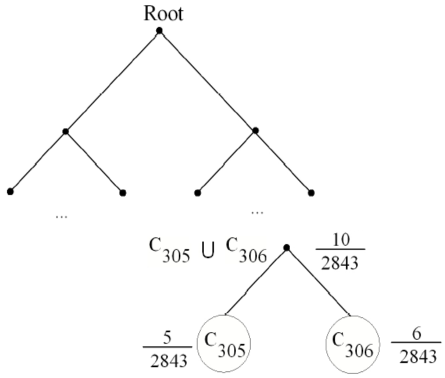

Figure 3: Fragment of our taxonomy.

In particular, clusters C_{305} and C_{306} of our hierarchical tree, for the domain of Economy, are represented by the following sets of words C_{305} ={life, effort, stability, steps, negotiations} and C_{306} ={steps, restructure, corporations, abuse, interests, ministers} and the number attached to each node C is $P(C)$ calculated as in Equation 14[14].

$$P(C_i) = \frac{\text{\# of words in the cluster}}{\text{\# of distinct words in all clusters}} \quad (14)$$

4.2.2 Relatedness criterion

The relatedness criterion is the threshold that needs to be respected in order to assign a word to a Lexical Chain. In fact, it works like a threshold. In this case, it is based on the average semantic similarity between all the clusters present in the taxonomy. So, if all semantic similarities between a candidate word cluster C_k and all the clusters in the chain $\forall l, C_l$ respect the relatedness criterion, the word is assigned to the Lexical Chain. This situation is defined in Equation 15 where c is a constant to be tuned and n is the number of words in the taxonomy. So, if Equation 15 is satisfied, the word w with cluster C_k is agglomerated to the Lexical Chain.

$$\forall l, simLin(C_k, C_l) > c \times \frac{\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} simLin(C_i, C_j)}{\frac{n^2 - n}{2}} \quad (15)$$

In the following section, we present an example of our algorithm.

4.2.3 Example of the Lexical Chain algorithm

The example below illustrates our Lexical Chain algorithm. Let's consider that a node is created for the first nominal unit encountered in the text i.e. *crisis* with its sense (C_{31}). The next appearing candidate word is *recession* which has two senses (C_{29} and C_{34}). Considering a relatedness criterion equal to 0.81 and the following similarities, $simLin(C_{31}, C_{29}) = 0.87$, $simLin(C_{31}, C_{34}) = 0.82$, the choice of the sense for *recession* splits the Lexical Chain into two different interpretations as shown in Figure 4, as both similarities overtake the given threshold 0.81.

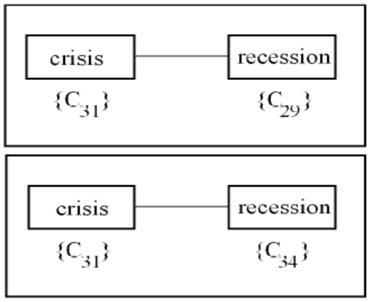

Figure 4: Interpretations 1 and 2.

The next candidate word *trouble* has also two senses (C_{29} and C_{32}). As all the words in a Lexical Chain influence each other in the selection of the respective senses of the new word considered, we have the following situation in Figure 5.

So, three cases can happen: (1) all similarities overtake the threshold and we must consider both representations, (2) only the similarities related to one representation overtake the threshold and we

[14] The value 2843 in Figure 3 is the total number of distinct words in our concept hierarchy.

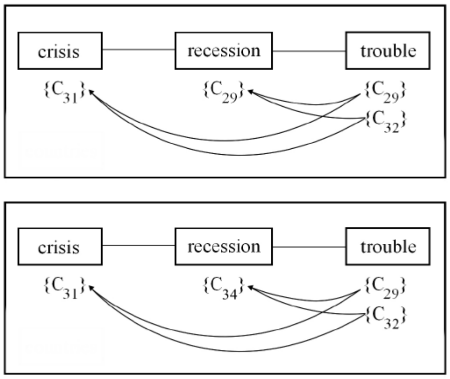

Figure 5: Selection of senses.

only consider this representation or (3) none of the similarities overtake the threshold and we create a new Lexical Chain. So, we proceed with our algorithm for both interpretations.

Interpretation 1 shows the following similarities $simLin(C_{31}, C_{29}) = 0.87$, $simLin(C_{31}, C_{32}) = 0.75$, $simLin(C_{29}, C_{29}) = 1.0$, $simLin(C_{29}, C_{32}) = 0.78$ and interpretation 2 the following ones, $simLin(C_{31}, C_{29}) = 0.87$, $simLin(C_{31}, C_{32}) = 0.75$, $simLin(C_{34}, C_{29}) = 0.54$, $simLin(C_{34}, C_{32}) = 0.55$.

By computing the average similarities for interpretations 1 and 2, we reach the following results: $average(Interpretation1) = 0.85 > 0.81$ and $average(Interpretation2) = 0.68 \not> 0.81$.

As a consequence, the word *trouble* is inserted in the Lexical Chain with the appropriate sense (C_{29}) as it maximizes the overall similarity of the chain and the chain members senses are updated. In this example, the interpretation with (C_{32}) is discarded as is the cluster (C_{34}) for *recession*. This processing is described in Figure 6.

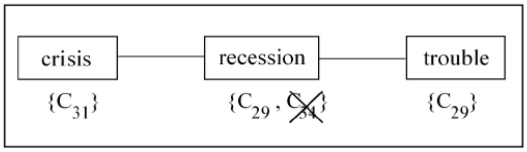

Figure 6: Selection of appropriate senses.

4.2.4 Score of a chain

Once all chains have been computed, only the high-scoring ones must be picked up as representing the important concepts of the original document. Therefore, one must first identify the strongest chains. Like in (Barzilay and Elhadad, 1997), we define a chain score which is defined in Equation 16 where $|chain|$ is the number of words in the chain.

$$score(chain) = \frac{\sum_{i=1}^{|chain|-1} \sum_{1}^{card(C_i)} \sum_{j=i+1}^{|chain|} \sum_{1}^{card(C_j)} simLin(C_i, C_j)}{\sum_{i=1}^{|chain|-1} \sum_{j=i+1}^{|chain|} card(C_i) \times card(C_j)} \quad (16)$$

As all chains will be scored, the ones with higher scores will be extracted. Of course, a threshold will have to be defined by the user. In the next section, we will show some qualitative and quantitative results of our architecture.

5 Evaluation

The evaluation of Lexical Chains is generally difficult. Even if they can be effectively used in many practical applications, Lexical Chains are seldom desirable outputs in a real-world application, and it is unclear how to assess their quality independently of the underlying application in which they are used (Budanitsky and Hirst, 2006). For example, in Summarization, it is hard to determine whether a good or bad performance comes from the efficiency of the lexical chaining algorithm or from the appropriateness of using Lexical Chains in that kind of application. It is also true that some work has been done in this direction (Budanitsky and Hirst, 2006) by collecting Human Lexical Chains to compare against automatically built Lexical Chains. However, this type of evaluation is logistically impossible to perform as we aim at developing a system that does not depend on any language or topic. So, in this section, we will only present some results generated by our architecture (like (Barzilay and Elhadad, 1997; Teich and Fankhauser, 2004) do), although we acknowledge that other comparative evaluations (with WordNet, with Human Lexical Chains or within independent applications like Text Summarization) must be done in order to draw definitive conclusions.

We have generated four taxonomies from four different domains (Sport, Economy, Politics and War) from a set of documents of the DUC 2004[15]. Moreover, we have extracted Lexical Chains for all four

[15] http://duc.nist.gov/duc2004/

domains to show the ability of our system to switch from domain to domain without any problem.

5.1 Quantitative Function

Four texts from each domain of the DUC 2004 corpus have been used to extract Lexical Chains based on the four knowledge bases built from all texts of DUC 2004 for each one of the four following domains: Sport, Economy, Politics and War. However, in this section, we will only present the results from the Sport Domain as results show similar behaviors for the other domains. In particular, we present in Table 1 the characteristics of each document.

	# Words	#Distinct Words	#Distinct Nouns
Doc 1	8133	1956	672
Doc 2	3823	1630	708
Doc 3	4594	953	324
Doc 4	4530	1265	431

Table 1: Characteristics of Documents for Sport

The first interesting conclusion shown in Table 2 is that the number of Lexical Chains does not depend on the document size but rather on the nominal units distribution. Indeed, for example, the number of words in Document 1 is twice as big as in Document 2. Although, we have more Lexical Chains in Document 2 than in Document 1, as Document 2 has more distinct nominal units.

	c=5	c=6	c=7	c=8
Doc 1	27	43	73	73
Doc 2	31	52	81	83
Doc 3	28	40	51	51
Doc 4	29	53	83	87

Table 2: # Lexical Chains per Document

The second interesting conclusion is that our algorithm does not gather words that belong to only one cluster and take advantage of the automatically built lexico-semantic knowledge base. This is illustrated in Table 3. However, it is obvious that by increasing the constant c the words in a chain tend to belong to only one cluster as it is the case for most of the best Lexical Chains with $c = 8$.

5.2 Qualitative Evaluation

In this section, as it is done in (Barzilay and Elhadad, 1997; Teich and Fankhauser, 2004), we present the five highest-scoring chains for the best threshold that we experimentally evaluated to be $c = 7$ for each domain (See Tables 4, 5, 6, 7). It is clear that the obtained Lexical Chains show a desirable degree of representativeness of the text in analysis.

	c=5	c=6	c=7	c=8
Doc 1	19	13	7	7
Doc 2	13	6	3	3
Doc 3	3	4	4	4
Doc 4	6	4	3	3

Table 3: # Clusters per Lexical Chain

Domain=Sport, Document=3, c=7
- #0, 1 cluster and score=1.0: {United States, couple, competition}
- #6, 3 clusters and score=1.0: {boats, Sunday night, sailor, Sword, Orion, veteran, cutter, Winston Churchill, Solo Globe, Challenger, navy, Race, supposition, instructions, responsibility, skipper, east, Melbourne, deck, kilometer, masts, bodies, races, GMT, Admiral's, Cups, Britain, Star, Class, Atlanta, Seattle, arms, fatality, sea, waves, dark, yacht's, Dad, Guy's, son, Mark, beer, talk, life, Richard, Winning, affair, canopy, death}
- #9, 1 cluster and score=1.0: {record, days, hours, minutes, rescue}
- #16, 3 clusters and score=1.0: {Snow, shape, north, easters, thunder, storm, change, knots, west, level, maxi's, search, Authority, seas, helicopter, night vision, equipment, feet, rescues, Campbell, suffering, hypothermia, safety, foot, sailors, colleagues, Hospital, deaths, bodies, fatality}
- #19, 2 clusters and score=1.0: {challenge, crew, Monday, VC, Offshore, Stand, Newcastle, mid morning, Eden, Rescuers, aircraft, unsure, whereabouts, killing, contact}

Table 4: 5 best Lexical Chains for Sport

Domain=Economy, Document=5, c=7
- #88, 4 clusters and score=1.0: {sign, chance, Rio, Janeiro, Grande, Sul, uphill, promise, hospitals, powerhouse, success, inhabitants, victory, pad, presidency, contingent, exit, legislature}
- #50, 1 cluster and score=1.0: {transactions, taxes, Stabilization, spate, fuel, income, fortunes, means}
- #77, 1 cluster and score=1.0: {proposal, factory, owners, Fund, Rubin's}
- #126, 1 cluster and score=1.0: {disaster, control, investment, review}
- #12, 2 clusters and score=0.99: {issue, order, University, population, question, timing, currencies}

Table 5: 5 best Lexical Chains for Economy

For instance, the Lexical Chain #16 in the domain of Sport clearly exemplifies the tragedy of climbers that were killed in a sudden change of weather in the mountains and who could not be rescued by the authorities.

However, some Lexical Chains are less expressive. For instance, it is not clear what the Lexical Chain #40 expresses in the domain of Politics. Indeed, none of the words present in the chain seem

Domain=Politics, Document=3, c=7
- #5, 1 cluster and score=1.0: {report, leaders, lives, information}
- #33, 1 cluster and score=1.0: {past, attention, defenders, investigations}
- #28, 2 clusters and score=0.95: {investigators, hospital, ward, wounds, neck, description, fashion, suspects, raids, assault, rifles, door, further details, surgery, service, detective, Igor, Kozhevnikov, Ministry}
- #40, 2 clusters and score=0.92: {security, times, weeks, fire}
- #24, 3 clusters and score=0.85: {enemies, Choice, stairwell, assailants, woman, attackers, entrance, car, guns, Friends, relatives, Mrs. Staravoitova, founder, movement, well thought, Sergei, Kozyrev, Association, Societies, supporter, Stalin's, council, criminals, Yegor, Gaidar, minister, ally, suggestions, measures, smile, commitment}

Table 6: 5 best Lexical Chains for Politics

Domain=War, Document=1, c=7
- #25, 2 clusters and score=1.0: {lightning, advance, Africa's, nation, outskirts, capital Kinshasa, troops, Angola, Zimbabwe, Namibia, chunk, routes, Katanga, Eastern, Kasai, provinces, copper}
- #53, 1 cluster and score=1.0: {Back, years, Ngeyo, farm, farmers, organization, breadbasket, quarter, century, businessman, hotels, tourist, memory, rivalry, rebellions}
- #56, 1 cluster and score=1.0: {political, freedoms, Hutus, Mai-Mai, warriors, Hunde, Nande, militiamen, Rwanda, ideology, weapons, persecution, landowners, ranchers, anarchy, Safari, Ngezayo, farmer, hotel, owner, camps}
- #24, 2 clusters and score=0.87: {fighting, people, leaders, diplomats, cause, president, Washington, U.S, units, weeks}
- #51, 2 clusters and score=0.82: {West, buildings, sight, point, tourists, mountain, gorillas, shops, guest, disputes}

Table 7: 5 best Lexical Chains for War

to express any idea about Politics. Moreover, due to the small number of inter-related nominal units within the Lexical Chain, this one can not be understood as it is without context. In fact, it was related to problems of car firing that have been occurring in the past few weeks and provoked security problems in the town.

Although some Lexical Chains are understandable as they are, most of them must be replaced in their context to fully understand their representativeness of the topics or subtopics of the text being analyzed. As a consequence, we deeply believe that Lexical Chains must be evaluated in the context of Natural Language Processing applications (such as Text Summarization (Doran et al., 2004)), as comparing Lexical Chains as they are is a very difficult task to tackle which may even lead to inconclusive results.

6 Conclusions and Future Work

In this paper, we implemented a greedy Language-Independent algorithm for building Lexical Chains. For that purpose, we first constructed a lexico-semantic knowledge base by applying the Pole-Based Overlapping Clustering algorithm (Cleuziou et al., 2004) to word-context vectors obtained by the application of the $SCP(.,.)$ measure (Silva et al., 1999) and the $InfoSimBA(.,.)$ (Dias and Alves, 2005) similarity measure. In a second step, we implemented (Lin, 1998)'s similarity measure and used it to define the relatedness criterion in order to assign a given word to a given chain in the lexical chaining process. Finally, our experimental evaluation shows that relevant Lexical Chains can be constructed with our lexical chaining algorithm, although we acknowledge that more comparative evaluations must be done in order to draw definitive conclusions. In particular, in future work, we want to compare our methodology using WordNet as the basic knowledge base, implement different similarity measures (Resnik, 1995; Jiang and Conrath, 1997; Leacock and Chodorow, 1998), experiment different Lexical Chains algorithms (Hirst and St-Onge, 1997; Barzilay and Elhadad, 1997; Galley and McKeown, 2003), scale our greedy algorithm for real-world applications following (Silber and McCoy, 2002) ideas and finally evaluate our system in independent Natural Language Processing applications such as Text Summarization (Doran et al., 2004).

References

R. Barzilay and M. Elhadad. 1997. *Using Lexical Chains for Text Summarization*. Proceedings of the Intelligent Scalable Text Summarization Workshop (ISTS-97), ACL, Madrid, Spain, pages 10-18.

I. Bomze, M. Budinich, P. Pardalos, and M. Pelillo. 1999. *The Maximum Clique Problem*. Handbook of Combinatorial Optimization, volume 4. Kluwer Academic publishers, Boston, MA.

T. Brants. 2000. *TnT - a Statistical Part-of-Speech Tagger*. In Proceedings of the 6th Applied NLP Conference, ANLP-2000. Seattle, WA.

A. Budanitsky and G. Hirst. 2006. *Evaluating WordNet-based Measures of Lexical Semantic Relatedness*. In Computational Linguistics, 32(1). pages: 13-47.

L. Cicurel, S. Bloehdorn and P. Cimiano. 2006. *Clustering of Polysemic Words*. In Advances in Data Analysis - 30th Annual Conference of the German Classification Society (GfKl). Berlin, Germany, March 8-10.

G. Cleuziou, L. Martin, and C. Vrain. 2004. *PoBOC: an Overlapping Clustering Algorithm. Application to Rule-Based Classication and Textual Data*. In Proceedings of the 16th European Conference on Artificial Intelligence, pages 440-444, Spain, August 22-27.

G. Cleuziou, V. Clavier, L. Martin. 2003. *Une Méthode de Regroupement de Mots Fondée sur la Recherche de Cliques dans un Graphe de Cooccurrences*. In Proceedings of Rencontres Terminologie et Intelligence Artificielle, France. pages 179-182.

B. Daille. 1995. *Study and Implementation of Combined Techniques for Automatic Extraction of Terminology*. In The balancing act combining symbolic and statistical approaches to language. MIT Press.

G. Dias and E. Alves. 2005. *Unsupervised Topic Segmentation Based on Word Co-occurrence and Multi-Word Units for Text Summarization*. In Proceedings of the ELECTRA Workshop associated to 28th ACM SIGIR Conference, Salvador, Brazil, pages 41-48.

G. Dias, S. Guilloré and J.G.P. Lopes. 1999. *Language Independent Automatic Acquisition of Rigid Multiword Units from Unrestricted Text Corpora*. In Proceedings of 6th Annual Conference on Natural Language Processing, Cargèse, France, pages 333-339.

W. Doran, N. Stokes, J. Carthy and J. Dunnion. 2004. *Assessing the Impact of Lexical Chain Scoring Methods and Sentence Extraction Schemes on Summarization*. In Proc. of the 5th Conference on Intelligent Text Processing and Computational Linguistics.

V. Estivill-Castro, I. Lee, and A. T. Murray. 2001. *Criteria on Proximity Graphs for Boundary Extraction and Spatial Clustering*. In Proceedings of the 5th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer-Verlag. pages 348-357.

C.D. Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, New York.

W. Gale, K. Church, and D. Yarowsky. 1992. *One Sense per Discourse*. In Proceedings of the DARPA Speech and Natural Language Workshop.

M. Galley and K. McKeown. 2003. *Improving Word Sense Disambiguation in Lexical Chaining*. In Proceedings of 18th International Joint Conference on Artificial Intelligence (IJCAI'03), Acapulco, Mexico.

G. Hirst and D. St-Onge. 1997. *Lexical Chains as Representation of Context for the Detection and Correction of Malapropisms*. In WordNet: An electronic lexical database and some of its applications. MIT Press.

J.W. Jaromczyk and G.T. Toussaint. 1992. *Relative Neighborhood Graphs and Their Relatives*. P-IEEE, 80, pages 1502-1517.

J.J. Jiang and D.W. Conrath. 1997. *Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy*. In Proceedings of International Conference on Research in Computational Linguistics, Taiwan.

R. Krovetz. 1998. *More than One Sense per Discourse*. NEC Princeton NJ Labs., Research Memorandum.

C. Leacock and M. Chodorow. 1998. *Combining Local Context and WordNet Similarity for Word Sense Identification*. In C. Fellbaum, editor, WordNet: An electronic lexical database. MIT Press. pages 265-283.

D. Lin. 1998. *An Information-theoretic Definition of Similarity*. In 15th International Conference on Machine Learning. Morgan Kaufmann, San Francisco.

G. Miller. 1995. *WordNet: An Lexical Database for English*. Communications of the Association for Computing Machinery (CACM), 38(11), pages 39-41.

J. Morris and G. Hirst. 1991. *Lexical Cohesion Computed by Thesaural Relations as an Indicator of the Structure of Text*. Computational Linguistics, 17(1).

P. Pantel and D. Lin. 2002. *Discovering Word Senses from Text*. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pages 613-619.

P. Resnik. 1995. *Using Information Content to Evaluate Semantic Similarity*. In Proceedings of the 14th International Joint Conference on Artificial Intelligence, Montreal. pages 448-453.

P.M. Roget. 1852. *Roget's Thesaurus of English Words and Phrases*. Harlow, Essex, England: Longman.

G. Salton, C.S. Yang and C.T. Yu. 1975. *A Theory of Term Importance in Automatic Text Analysis*. In American Society of Information Science, 26(1).

G. Silber and K. McCoy. 2002. *Efficiently Computed Lexical Chains as an Intermediate Representation for Automatic Text Summarization*. Computational Linguistics, 28(4), pages 487-496.

J. Silva, G. Dias, S. Guilloré and J.G.P. Lopes. 1999. *Using LocalMaxs Algorithm for the Extraction of Contiguous and Non-contiguous Multiword Lexical Units*. In Proceedings of 9th Portuguese Conference in Artificial Intelligence. Springer-Verlag.

P. H. A. Sneath and R. R. Sokal. 1973. *Numerical Taxonomy - The Principles and Practice of Numerical Classification*. San Francisco, Freeman and Co.

E. Teich and P. Fankhauser. 2004. *Exploring Lexical Patterns in Text: Lexical Cohesion Analysis with WordNet*. In Proceedings of the 2nd International Wordnet Conference, Brno, Czech Republic. pages 326-331.