STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **XLVIII**, Number 1, 2003

WORDS CLUSTERING IN QUESTION ANSWERING SYSTEMS

DOINA TĂTAR AND GABRIELA ŞERBAN

Abstract. Clustering words can be useful in construction of a hierarchy of hypernyms or a set of synonyms for languages different of English for which doesn't exist such hierarchy as WordNet (as in Romanian language case). A such of hierarchy is very important in some problems of disambiguation [10], as to perform automatic query expansion in a QA system for Romanian [7]. In this paper we we describe how a list of similar words with a given word can be constructed. Some words and word-clusters similarity measures are discussed.

The experiments are made using a Romanian corpus.

1. Introduction

Semantic knowledge is increasingly important in NLP. The key of organizing semantic knowledge is to define reasonable similarity measures between words. The purpose to develop a hierarchy of words based on a untagged corpus can be realized by using hierarchical and non hierarchical clustering algorithms. In many papers the similarity between two words is obtained by the n-grams models [8], by mutual information [2] or by syntactic relations [9]. One other mode to define this similarity is the vector-space model, which we use in this paper. In our paper the vector $\vec{w_i}$ is associated with a word w_i as following: let us consider that $\{v_1, v_2, \cdots, v_m\}$ are m words of a high frequency in corpus. They can be of any POS set including prepositions and conjunctions from the closed class of words. The reason for this choice it is the known Zipf's result that a small set consisting of most frequent words can be used as framework for study of a natural language.

We define:

$$w_i^j = number\, of\, occurences\, of\, the\, word\, v_j\, in\, the\, same\, context\, with\, w_i$$

$$(for\, a\, number\, of\, contexts).$$

Received by the editors: February 7, 2003.

2000 *Mathematics Subject Classification.* 68T50, 68Q32.

1998 *CR Categories and Descriptors.* I.2.7 [**Computing Methodologies**]: Artificial Intelligence – *Natural Learning Processing*; G.3 [**Mathematics of Computing**]: Statistical Computing.

Let us remark that other vector-space models were used in the literature. In [1] is presented a hierarchy of nouns such that the vector $\vec{w_i} = (w_i^1, w_i^2, \cdots, w_i^m)$ associated with a noun w_i is constructed as follows: $w_i^j = 1$, if the noun w_j occurs after w_i separated by the conjunction *and* or an appositive, or else $w_i^j = 0$.

In [5], the vector $\vec{w_i} = (w_i^1, w_i^2, \cdots, w_i^m)$(where $m = 2 \times z$) associated with a word w_i, is constructed as follows: w_i^j = number of occurrences of a word in the position $j = 1\,to\,z$ at left or $z + 1\,to\,m$ at right.

The paper is arranged as follows. Section 2 presents known clustering algorithms [6]: agglomerative algorithm for hierarchical clustering and divisive non-hierarchical k-means algorithm, adopted for our vector-space model. Section 3 proposes our variant for an agglomerative algorithm for hierarchical clustering such that a single "best" word is clusterized at a step. Section 4 describes how a list of similar words with a given word can be constructed. In section 5 we propose an experiment for Romanian language and present comparatively the results obtained by applying the clustering algorithms described in Section 2.

2. Clustering algorithms

Let us consider that the objects to be clusterized are the vectors of n words, $\{w_1, w_2, \cdots, w_n\}$. A vector

$$\vec{w_i} = (w_i^1, w_i^2, \cdots, w_i^m)$$

is associated with a word w_i as above.

Let us observe that the corpus must not be POS tagged or parsed since we are interested only of words and not of their syntactic role. However, we used a stammer to recognize the flexional occurrences of the same word (Romanian language is a very inflexional language).

The similarity measure between two words w_a, w_b is the *normalised cosine* between the vectors $\vec{w_a}$ and $\vec{w_b}$ [4]:

$$sim(\vec{w_a}, \vec{w_b}) = cos(\vec{w_a}, \vec{w_b}) = \frac{\sum_{j=1}^{m} w_a^j \times w_b^j}{\sqrt{\sum_{j=1}^{m} w_a^{j^2}} \times \sqrt{\sum_{j=1}^{m} w_b^{j^2}}}.$$

Agglomerative algorithm for hierarchical clustering [6]

Input The set $X = \{w_1, w_2, \cdots, w_n\}$ of n words to be clusterised, the similarity function $sim : X \times X \to R$.

Output The set of hierarchical clusters
$C = \{C_1, C_2, \cdots, C_n, C_{n+1}, \cdots, C_{n+k}\}$

begin

$FOR\ i = 1\ TO\ n\ DO\ C_i = \{w_i\}$
$C = \{C_1, C_2, \cdots, C_k\}$
$j := n + 1$
$WHILE\ \mid C \mid > 1\ DO$
$(C_{u^*}, C_{v^*}) := argmax_{(C_u, C_v)} sim(C_u, C_v)$
$C_j = C_{u^*} \cup C_{v^*}$
$C = C \setminus \{C_{u^*}, C_{v^*}) \cup \{C_j\}$
$j := j + 1$

end

As similarity $sim(C_u, C_v)$ we considered:

$$sim(C_u, C_v) = \frac{\sum_{a_i \in C_u} \sum_{b_j \in C_v} sim(a_i, b_j)}{\mid C_u \mid \times \mid C_v \mid}.$$

The clustering algorithm begins by considering each word in its own cluster and ends when all the words are in the same cluster $C_{all} = C_{n+k}$. Let us consider $\{s_{n+1}, s_{n+2}, \cdots, s_{n+k}\}$ the values of similarities such that $s_i = sim(C_{u^*}, C_{v^*})$ and (C_{u^*}, C_{v^*}) has the same sense as in above algorithm. In other words, $\{s_{n+1}, s_{n+2}, \cdots, s_{n+k}\}$ are the values of similarities such that a new cluster $C_j = C_{u^*} \cup C_{v^*}$ is formed, $j = n + 1$ to $n + k$. The similarities $\{s_1, s_2, \cdots, s_n\}$ are all set to 1.

The similarities $\{s_1, s_2, \cdots, s_{n+k}\}$ are ordered decreasing from 1 (the similarities in clusters $C_i = \{w_i\}$, $i = 1, \cdots n$) to s_{n+k}, the similarity in the cluster $C_{all} = C_{n+k}$, as they occur on the dendrogram.

Non-hierarchical clustering algorithm: k-means algorithm [6]

Input The set $X = \{\vec{w_1}, \vec{w_2}, \cdots, \vec{w_n}\}$ of n vector words to be clusterised, the distance measure $d : R^m \times R^m \to R$, a function for computing the mean $\mu : P \to R$, the coefficient σ.

Output The set of clusters $C = \{C_1, C_2, \cdots, C_k\}$

begin

Select k initial centroids $\{\vec{f_1}, \vec{f_2}, \cdots, \vec{f_k}\}$
WHILE the diameter of a cluster $\geq \sigma$ *DO*
FOR all clusters C_j *DO*
$C_j = \{\vec{x_i} \mid \forall \vec{f_l}\ d(\vec{x_i}, \vec{f_j}) \leq d(\vec{x_i}, \vec{f_l})\}$
FOR all clusters C_j *DO*

$\vec{f_j} = \vec{\mu}(C_j)$

end

As distance measure we considered:

$$d(\vec{w_a}, \vec{w_b}) = \frac{1}{sim(\vec{w_a}, \vec{w_b})}$$

and as centroid:

$$\vec{\mu}(C_j) = \frac{1}{\mid C_j \mid} \sum_{\vec{x} \in C_j} \vec{x}$$

A diameter of a cluster we define as *the distance between the least similar elements in a cluster.*

3. An incremental algorithm for clustering

The following algorithm has the property that at the begin of the process it arrange at a time only one word to an appropriate cluster.

For a word w_i let $N(w_i)$ be the set of words w_j such that $sim(w_i, w_j) \neq 0$. For a set of words C, $N(C)$ will denote $\bigcup_{w_i \in C} N(w_i)$. The set $N(C)$ is similar with the set of $neighbours$ of C in [9] but there the problem is solved on a graph model.

Let C be a set of words from W and $u \in N(C) \backslash C$.

We define

$$sim(u, C) = \sum_{w \in C} sim(w, u).$$

The best node u' from a set Q of words, which can be added to the set C, denoted $Best(C, Q)$, is the node which maximizes $sim(u, C)$:

$$Best(C, Q) = argmax_{u \in Q \cap N(C) \backslash C} sim(u, C).$$

Our hierarchical algorithm differs of the above hierarchical algorithm by the fact that in a step we form a new cluster by adding to a cluster C of the only word $Best(C, Q)$.

The algorithm is:

Input The set $X = \{w_1, w_2, \cdots, w_n\}$ of n words to be clusterised, the similarity function $sim : X \times X \to R$.

Output The set of hierarchical clusters
$C = \{C_1, C_2, \cdots, C_n, C_{n+1}, \cdots, C_{n+k}\}$

begin

$FOR\ i = 1\ TO\ n\ DO\ C_i = \{w_i\}$
$C = \{C_1, C_2, \cdots, C_n\}$
$Q = X$
$j := n + 1$
$WHILE\ Q \neq \Phi DO$
$s = argmax_{k=1,\cdots,j-1} Best(C_k, Q)$
$u' = Best(C_s, Q)$
$C_j = \{u'\} \cup C_s$
$C = (C \setminus \{C_s\}) \cup \{C_j\}$
$Q = Q \setminus \{u'\}$
$j := j + 1$

end

Let us remark that after all words from X (Q initial) are clustered, the algorithm stops.

Let us mention that our algorithm consider only a sense of a word u and for it exists only a cluster C such that $u = Best(C, Q)$. Of course this is not the case for polysemous words. In [9] is established that if G is a graph of words build on the base of a symmetric syntactic relation, and $G \setminus w$ is the subgraph which results from the removal of w, then the connected components of the subgraph $G \setminus w$ correspond to the senses of the word w. The above algorithm can be adopted in this sense, the symmetric relation being sim.

Once that some measure of similarity between words are established, we can begin a new process of divisive splicing in clusters. We seek to partition the set W of words into two subsets W_1, W_2 of the same size so that the similarity between W_1, W_2 is minimal: that means that

$$(W_1, W_2) = argmin_{V_1, V_2} \sum_{w_i \in V_1} \sum_{w_j \in V_2} sim(w_1, w_2)$$

An algorithm for implement a such of partition is a variant of hill-climbing search ([3]): after guessing an initial partition (W_1, W_2) we exchange two words between W_1 and W_2 if the exchange minimize $\sum_{w_i \in V_1} \sum_{w_j \in V_2} sim(w_1, w_2)$. We stop when no further decrease is possible.

4. The list of similar words for a given word

Input The set of hierarchical clusters $C = \{C_1, C_2, \cdots, C_n,\ C_{n+1}, \cdots, C_{n+k}\}$ (as above), the set of similarities $\{s_1, s_2, \cdots, s_{n+k}\}$, a word $w \in X$

Output The lists *Elem* and *SimDecr* containing the elements in X in decreasing order of similarity with w and the sequence of these similarities.

begin
Set $j = 1$, $Elem(1) = w$ and $SimDecr(1) = 1$
FOR i=n+1 *TO* n+k *DO*
IF $w \in C_i$ *(* $C_i = \{C_{i,1}, \cdots, C_{i,p_i}\}$*) THEN*
FOR t=1 *TO* p_i *DO*
IF not$(C_{i,t} \in Elem)$ *THEN*
$j := j + 1$
$Elem(j) = C_{i,t}$; $SimDecr(j) = s_i$
end

A corresponding algorithm for calculating the list of similar words for a given word can be imagined using the k-means algorithm : for each word w, the words in the same cluster (let say C), in order of distances to w, begin the list. That list contains then the words from the others clusters, in order of distance (the inverse of similarity) from C. The similarity is: $sim(C_u, C_v) = \frac{\sum_{a_i \in C_u} \sum_{b_j \in C_v} sim(a_i, b_j)}{|C_u| \times |C_v|}$.

5. Results and evaluation

5.1. **Applications.** In this section we want to show how the clustering process (based on the algorithms described in the previous section) works.

The first application uses the non-hierarchical clustering algorithm (NHCA - section 2), the second uses the hierarchical clustering algorithm (HCA - section 2).

Both NHCA and HCA are written in JDK 1.4. The aim is to clusterize a set of words.

The NHCA algorithm starts with a set of contexts, a set of words having a maximum frequency in the given contexts and with a set of "focus" words (*terms*) used in the clustering process. As a result of the clustering, the algorithm reports a set of clusters (in a cluster will be the *similar* words - the words having similar senses).

The process starts with a set of initial clusters (based on the *focus* words), and after that, learns, based on the information obtained from the initial contexts to clusterize the set of words.

We have to notice that the set of *terms* used for the clustering is very important (this was shown experimentally).

The HCA algorithm starts with the same initial information as NHCA, except the set of *terms*. As a result of the clustering, the algorithm reports, for each word

w, a cluster that will contain the *similar* words with w, in descending order after their similarities.

Because in the HCA algorithm the process does not depend on a set of *focus* words, the clustering result is more exact than the result of NHCA algorithm.

It is obvious, for both algorithms, that if the number of contexts grow, the clustering's precision grows, too (this is shown experimentally).

The initial information, for both algorithms, is read from a text file.

5.2. **The applications design.** The basis classes used for implementing the two applications are the same; differs only the clustering algorithm. The main classes are:

- **CList**: defines the type the structure of a list of objects, having methods for:
  - adding an object in the list;
  - accessing elements from the list;
  - updating elements from the list;
  - returning the dimension of the list;
- **CLine**: defines the structure of a list having as elements real values (is defined using the *CList* class);
- **CContext**: defines the structure of a list having as elements words (is defined using the *CList* class);
- **CLine**: defines the structure of a list having as elements lists with real values (is defined using the *CList* class);

5.3. **Experimental results.** In this section we propose an experiment for the Romanian language: the aim is to clusterize a set of words (to group the words after the similarity of their meanings). We have applied both the NHCA and the HCA algorithms.

We mention that we used a set of 26 contexts. We also note that if we grow the number of contexts, the clustering's precision grow.

The initial information (the set of words to be clusterized, the contexts, the focus words) is read from a text file having the following structure:

---

the words to be clusterized

oameni oras durata timp partid persoana localitate perioada organizatie sat asociatie

a set of words that the clustering process is based on (at us, these are words having a maximum frequency in the contexts)

de in la sa care ca pe munca premier

the contexts

(1) indreptatirea la masurile reparatorii prevazute de prezentul articol este conditionata de continuarea activitatii ca persoana juridica pana la intrarea in vigoare a prezentei legi sau de imprejurarea ca activitatea lor
(2) In vederea desfasurarii anchetei disciplinare, salariatul va fi convocat in scris de persoana imputernicita de catre conducatorul unitatii sa realizeze ancheta
(3) Memoriul a ajuns la scoala din localitate, la primarie, la prefectura si la Insepectoratul Scolar al judetului Harghita
(4) Totodata, la Conel, persoanele detasate la unitati din alta localitate, precum si cele delegate in afara locului de munca au castiguri uriase
(5) ...

the focus words
persoana localitate perioada organizatie

After applying the NHCA algorithm, we obtained the following clusters:

Cluster 1	*timp partid persoana sat*
Cluster 2	*oras localitate*
Cluster 3	*durata perioada*
Cluster 4	*oameni organizatie asociatie*

As a measure for evaluation of the NHCA algorithm we propose the *precision* of the clustering, defined as follows:

$$P = \frac{\sum_{i=1}^{k} \frac{n_i}{N_i}}{k} \tag{1}$$

where k is the number of clusters, n_i is the number of words correctly placed in the i-th cluster, and N_i is the total number of words placed in the i-th cluster.

We mention that for our experiment, the precision of the NHCA algorithm is 93%.

For the same set of words to clusterize, we have applied the HCA algorithm.

The result obtained for the word *asociatie* is given in Table 1 (each word is followed by its similarity with the given word)

We mention that we ran the clustering algorithms on bigger data sets (10000 contexts), 200 words to clusterize an the results are very good.

We also mention that this clustering applications are part of a QA system that is developed for the Romanian language [7].

Word	Similarity
asociatie	1.0
oameni	0.8498365855987975
oras	0.6255587777150006
localitate	0.6255587777150006
organizatie	0.6255587777150006
timp	0.6255587777150006
persoana	0.6255587777150006
sat	0.6255587777150006
durata	0.5183688447475575
perioada	0.5183688447475575
partid	0.31611039139928965

TABLE 1. The result of applying the HCA algorithm for the word *asociatie*

6. Conclusion and future directions

The above algorithms must be connected with the word sense disambiguation algorithms [10] to work well with ambiguity. A WSD algorithm must be run to distinguish between two (or more) different senses of a polysemic word. In this case, the different occurrences of senses correspond to different words.

We intend to evaluate the QA system [7] by expanding of query terms, word by word, with most similar words in the lists.

We intend to use the similarity between two words to disambiguating a group of two words: it is well known that two polysemic words are better disambiguated when they occur together (for example *doctor* and *nurse* which are both polysemic [8]).

References

[1] S. A. Caraballo, "Automatic construction of hypernym-labeled noun hierarchy from text", Proceedings of ACL, 1999.
[2] I. Dagan, L. Lee, F. C. N. Pereira, "Similarity-based models of Word Coocurences Probabilities", MLJ 34 (1–3), 1999.
[3] Y. Even-Zohar, D. Roth, D. Zelenko, "Word prediction and Clustering", http://citesser.nj.nec.com/even-zohar99word.html
[4] D. Jurafsky, J. Martin, "Speech and language processing", Prentice Hall, 2000.
[5] J. Karlgren, M. Sahlgren, "From words to understanding", CSLI 2001, pp. 294–308.
[6] C. Manning, H. Schutze, "Foundation of statistical natural language processing", MIT, 1999.
[7] C. Orăşan, D. Tătar, G. Şerban, D. Avram, A. Oneţ, "How to build a QA system in your back-garden: application to Romanian", EACL '03, Budapest, April 2003.

[8] P. Resnik, "Semantic Similarity in a Taxonomy: An information-Based Measure and its Application to Problems of Ambiguity in Natural language", Journal of AI Research, 1998.
[9] D. Widdows, B. Dorow, "A graph model for unsupervised lexical acquisition".
[10] G. Şerban, D. Tătar, "Word Sense Disambiguation for Untagged Corpus: Application to Romanian Language", Proceedings of CICLing 2003 (Intelligent Text Processing and Computational Linguistics), Mexico City, Mexic, Lecture Notes in Computer Science N 2588, Springer-Verlag, 2003, pp.270-275.

Babeş-Bolyai University, Faculty of Mathematics and Computer Science, Cluj-Napoca, Romania
E-mail address: dtatar@cs.ubbcluj.ro, gabis@cs.ubbcluj.ro