

# Automatic Concept Space Discovery

Pawel Matykiewicz, Wlodzislaw Duch, John Pestian

April - May, 2005

A model for text “text understanding” is presented. This model includes learning and recognition phase. Learning phase provides the model with the *a priori* knowledge whether recognition phase makes use of this knowledge. This model includes few intermediate steps like: collocations extraction, semantic space construction, and word sense disambiguation. Words co-occurrence matrix ( HN matrix ) is used to extract collocations as well as to build semantic vectors. Furthermore average linkage clustering algorithm is used for both to discriminate between different senses of a word or collocation and to gather the senses into similar components called here concepts.

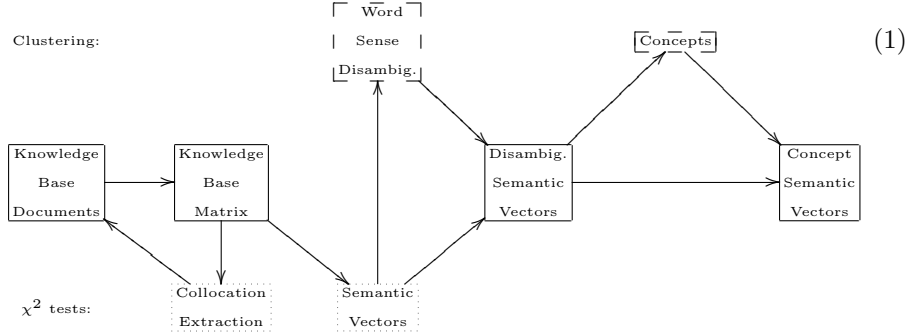
## 1 Introduction

Our objective is to construct a model that is capable to create special numerical representation for any document with a use of *a priori* knowledge. This representation should cope with three natural language processing problems:

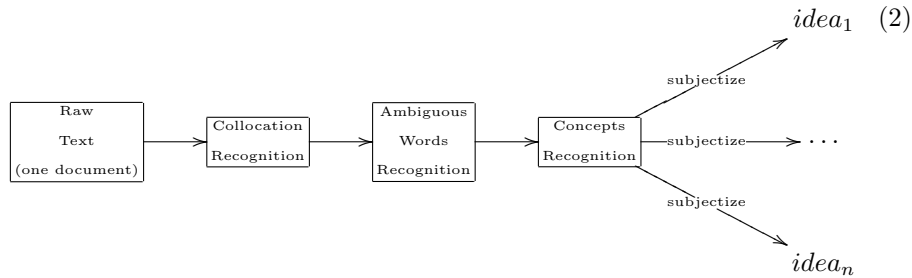
1. two documents can share the same words but with a different meaning
2. two documents can have different number of words but describe exactly the same idea
3. one document can describe more then one idea

Solution to the fist problem is word sense disambiguation, to the second gathering words that have similar meaning into concepts and to the third text segmentation via concept semantic vectors ( cohesion problem ). Presented method is based on two phases: a learning phase and recognition phase. First phase

creates a priori knowledge:



When all parts in the learning phase are generated a construction of special concept representation for a testing set can be created. The recognition phase schema is as follows:

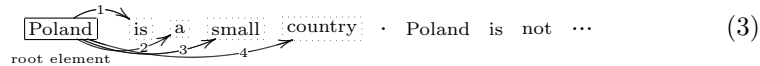


The recognition phase needs to be repeated for every document. Most significant feature of this model is that it does not impose any particular method for clustering and term weighting. Nonetheless some generic algorithms are chosen.

## 2 Knowledge Base Matrix

### 2.1 Math

First step is to construct a 3-dimensional matrix of co-occurrence of words. First dimension of this matrix is a root word, second dimension is co-occurring word and third is a relative position of the co-occurring word. Example of extracting this information is in the 3 diagram. After parsing sentence “Poland is a small country.” one should get a value 1 for a root word “poland” and co-occurring word “is” in the distance 1 and so on. Let call this matrix  $HN$  then this fact can be written as  $HN_{Poland,is,1} = 1$  or rather  $HN_{Poland,is,1} := HN_{Poland,is,1} + 1$ .



The maximal distance between two words is 5. It could be less if the co-occurring element is the period. This means that we do not check the co-occurrence of

words between sentences. Nonetheless sometimes a word creates collocation with the period then the length of a sentence is extended to other period that is not in any collocation. Moreover it is important to notice that  $HN_{i,j,-d} = HN_{j,i,d}$ . So in fact we gather information of co-occurrence of words in a window of size 10. This fact will be used for construction of semantic vectors.

This matrix has been described and used for modeling some cognitive functions by Hecht-Nielsen. It can be used also to correct sentences.

## 2.2 Example

As an example of our *knowledge base texts* 10 files with descriptions of 10 diseases: pneumonia, asthma, seizure disorder/epilepsy, anemia, urinary tract infection, juvenile rheumatoid arthritis, cystic fibrosis, cerebral palsy, otitis media, gastroenteritis. Descriptions of these diseases are based on different sources medical on-line dictionaries.

# 3 Collocations Extraction

## 3.1 Math

Extraction of collocations is conducted with a use of Dunning's hypothesis testing. It is more appropriate for sparse data than the  $\chi^2$  test. The likelihood ratio of a hypothesis that is a formalization of independence of two words and dependence of two words is evaluated in the following way:

$$\log \lambda_{i,j,d} = \frac{L(H_{independence}^{i,j,d})}{L(H_{dependence}^{i,j,d})} \quad (4)$$

We check only collocations with a distance between words  $d = 1$ . Moreover we use usual maximum likelihood estimates for  $p$ ,  $p_1$ ,  $p_2$  and write  $c_i$ ,  $c_j$ ,  $c_{ij1}$  for the number of occurrence of word  $i$ ,  $j$ ,  $c_{ij1} = HN_{i,j,1}$  whereas  $N$  is the number of all parsed words in the corpus:

$$p = \frac{c_2}{N}, \quad p_1 = \frac{c_{ij1}}{c_1}, \quad p_2 = \frac{c_2 - c_{ij1}}{N - c_1}$$

Assuming a binominal distribution:

$$b(k; n, x) = \binom{n}{k} x^k (1-x)^{(n-k)}$$

the likelihood of getting the counts for  $i$ ,  $j$  and  $i, j$  with distance 1 that we actually observed is then  $L(H_{independence}^{i,j,1}) = b(c_{ij1}; c_i, p)b(c_j - c_{ij1}; N - c_1, p)$  and  $L(H_{dependence}^{i,j,1}) = b(c_{ij1}; c_i, p_1)b(c_j - c_{ij1}; N - c_1, p_2)$ . Now we can simplify

equation 4 in the following way:

$$\begin{aligned}
\log \lambda_{i,j,1} &= \log \frac{b(c_{ij1}; c_i, p)b(c_j - c_{ij1}; N - c_1, p)}{b(c_{ij1}; c_i, p_1)b(c_j - c_{ij1}; N - c_1, p_2)} \\
&= \log p^{c_{ij1}}(1-p)^{c_i - c_{ij1}} + \log p^{c_j - c_{ij1}}(1-p)^{N - c_i} \\
&\quad - \log p_1^{c_{ij1}}(1-p_1)^{c_i - c_{ij1}} - \log p_2^{c_j - c_{ij1}}(1-p_2)^{N - c_i}
\end{aligned} \tag{5}$$

It is more convenient to use  $-2 \log \lambda_{i,j,d}$  because it is asymptotically  $\chi^2$  distributed. If  $-2 \log \lambda_{i,j,d} > 10.83$ ,  $c_i > 9$  and  $c_j > 9$  than two words are considered as a collocation. In terms of  $\chi^2$  test it means that we reject independence hypothesis with confidence level of 0.001. After extracting all collocations we can parse our corpus again but this time every two words ( $i, j$ ) that create collocation are changed to one element ( $i\_j = k$ ). This way it is possible to find three-word collocation ( $k\_l = i\_j\_l$ ) and so on. This process can be repeated until no new collocations will be found.

### 3.2 Example

## 4 Semantic Space Construction

### 4.1 Math

Semantic vectors are constructed in similar way to collocations. For every word  $i$  all words  $i_j$  that occurred in its neighborhood  $d = -5 \dots 5$  are checked with the likelihood ratio test. Here we use the confidence level of  $\approx 0.1$  that is:

$$\begin{aligned}
&\text{if} \quad -2 \log \lambda_{i,i_j,d} > 2.7075, \quad c_i > 9 \quad \text{and} \quad c_{i_j} > 9 \\
&\text{then} \quad s_{i,i_j} := s_{i,i_j} + \frac{1}{|d|} H N_{i,i_j,d}
\end{aligned} \tag{6}$$

Moreover a stop-list with all pronouns, prepositions, articles, coordinating and subordinating conjunctions is used. This procedure give us  $S = (s_{i,j})_{n \times n}$  symmetrical and square matrix which is a special projection of  $HN$  matrix. Semantics vectors are rows of the  $S$  matrix so that  $\mathbf{s}_{ij} = s_{i,j}$

### 4.2 Example

## 5 Clustering Method

### 5.1 Math

As a clustering algorithm average linkage algorithm is used from the "The C Clustering Library" developed at The University of Tokyo, Institute of Medical Science, Human Genome Center.

Function  $\mathcal{D}(S) = D$  uses rows of any  $S$  matrix to calculate distance matrix. Instead of simple cosine measure which measures similarity between two vectors

we use  $d_{i,j} = 1 - \cos(\mathbf{s}_i, \mathbf{s}_j)$  which defines a dissimilarity or distance matrix  $D = (d_{i,j})_{n \times n}$  that is:

$$\begin{aligned} \mathcal{D}((s_{i,j})_{n \times m}) &= (d_{i,j})_{n \times n} \\ \text{where} \quad d_{i,j} &= 1 - \frac{\mathbf{s}_i \cdot \mathbf{s}_j}{|\mathbf{s}_i| |\mathbf{s}_j|} \end{aligned} \quad (7)$$

In order to determine the number of cluster a “goodness of fit” or “stopping rule” function  $\mathcal{G}(c)$  is used. Global maximum of this function suggests that this is a well-separated number of clusters. To construct this type of function we use mean square error and mean square maximum defined in a following way:

$$\overline{err^2}(c) = \frac{\sum_{i=1}^c \sum_{j,k \in \mathcal{C}_i} d_{j,k}^2}{\sum_{i=1}^c \sum_{j,k \in \mathcal{C}_i} 1}, \quad (8)$$

$$\overline{max^2}(c) = \frac{\sum_{i=1}^c \max_{j,k \in \mathcal{C}_i}^2 d_{j,k}}{c}, \quad (9)$$

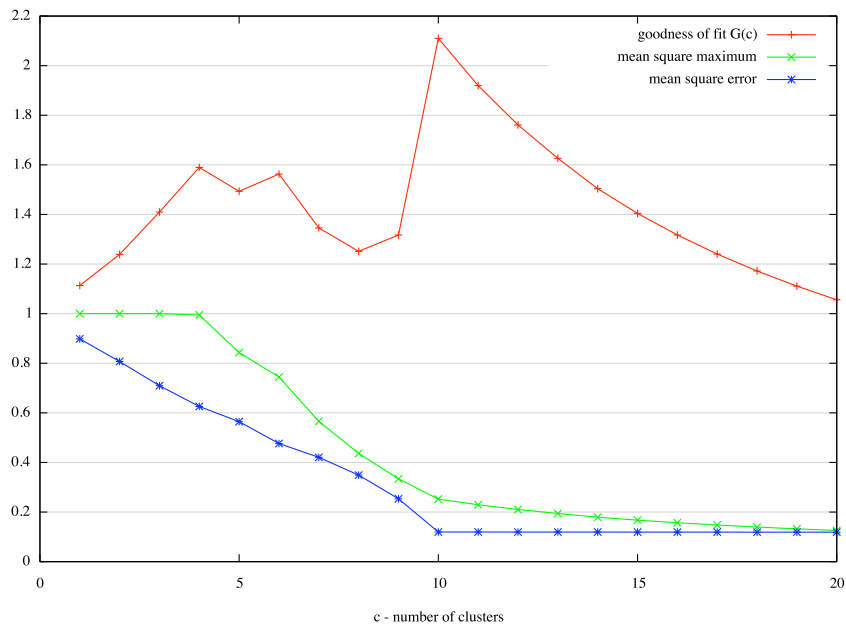
where  $c$  is a number of clusters,  $\mathcal{C}_i$  is a set of indices of rows belonging to the  $i$ th cluster. When number of clusters are increased then  $\overline{err^2}$  and  $\overline{max^2}$  is minimalized. Optimal solution to this problem is a balance between those two variables that is a global maximum of the following function:

$$\mathcal{G}(c) = \frac{\overline{max^2}}{\overline{err^2}} \quad (10)$$

If  $|\mathcal{C}_i| = 1$  and  $j \in \mathcal{C}_i$  then  $j$  is treated as an outlier and  $\mathcal{C}_i$  is deleted so that  $j \notin \bigcup_k \mathcal{C}_k$ .

## 5.2 Example

$G(c)$  function is tested on specially created vectors that simulate documents. 10 classes are created with 100 pseudo-documents per class. First 100 vectors have random values at randomly selected first 100 constituents with Gaussian distribution mean 50 and standard deviation 30. Next 100 vectors have mean in 150 and so on. Standard deviation is the same for every class.



## 6 Word Sense Disambiguation

### 6.1 Math

Unsupervised disambiguation of an  $i$ th word is a two-step process. First clustering of words  $i_j$  that are in  $i$  word neighborhood. To accomplish this task second order matrices  $S^i = (s_{i_j,k}^i)_{n \times n}$  are created for every  $i$ th word:

$$s_{i_j,k}^i = \begin{cases} s_{i_j,k} & \text{if } s_{i,i_j} \neq 0 \wedge s_{i,k} \neq 0 \\ 0 & \text{if } s_{i,i_j} = 0 \vee s_{i,k} = 0 \end{cases} \quad (11)$$

Then dissimilarity matrix  $\mathcal{D}(S^i) = D^i$  is constructed and rows are clustered. If more than one cluster is obtained then words in each cluster are considered as candidates for triggers for non-ambiguous meanings. Next step is to check whether trigger-candidates really create separate meaning. Suppose  $N_i$  is a set of all neighboring words of the  $i$ th word that pass the  $-2 \log \lambda$  test then  $\mathcal{S}_i = \{i_j : s_{i,i_j} \neq 0\}$ . After clustering  $S^i$  we have sub-sets  $\mathcal{S}_i^k \subset \mathcal{S}_i$  which are the trigger-candidates for  $k$ th meaning. A greedy algorithm is proposed to correct content of trigger-candidates.  $\mathcal{S}_i^k$  subsets are scored in order to find the winning subset. Let  $l$  be  $l$ th word in the learning corpus  $\mathbf{d}$  so that  $bd_l = i$  then we have a set of neighbors of this word in the corpus  $\mathcal{N}_l$ . We can score all  $\mathcal{S}_i^k$  respect to  $\mathcal{N}_l$  that is:

$$\mathcal{S}(\mathcal{S}_i^k, \mathcal{N}_l) = |\{i_j : i_j \in \mathcal{S}_i^k \wedge i_j \in \mathcal{N}_l\}| \quad (12)$$

The set that gets the best score is modified in the following way:

$$\begin{aligned} \text{if} \quad & \mathcal{S}(\mathcal{S}_i^k, \mathcal{N}_l) = \max_j \mathcal{S}(\mathcal{S}_i^j, \mathcal{N}_l) \\ \text{then} \quad \mathcal{S}_i^k := & \bigcup_{j: \mathcal{S}(\mathcal{S}_i^j, \mathcal{N}_l) > 0} \mathcal{S}_i^j \cup \{i_j : i_j \in \mathcal{N}_l \wedge s_{i,i_j} \neq 0\} \end{aligned} \quad (13)$$

Set  $\{i_j : i_j \in \mathcal{N}_l \wedge s_{i,i_j} \neq 0\}$  may contain some  $i_j$  element that after clustering were considered as outliers. It may also happened that there are two  $\mathcal{S}_i^k$  sets that have the same score then the one with the lower index is updated. Final step is to create semantic  $\mathbf{s}_{i^k}$  vectors for the ambiguous  $i$ th word where  $i^k$  are all new meanings for this word so that  $\mathbf{s}_i = \sum_k \mathbf{s}_{i^k}$ . New vectors are created in following way:

$$\mathbf{s}_{i^k j} = \begin{cases} \mathbf{s}_{ij} & \text{if } j \in \mathcal{S}_i^k \\ 0 & \text{if } j \notin \mathcal{S}_i^k \end{cases} \quad (14)$$

Ambiguous semantic vectors are discarded and  $\mathbf{s}_{i^k}$  vectors are used instead. This way new  $S' = (s_{i,j})_{m \times n}$  matrix is composed where  $m > n$ .

## 6.2 Example

# 7 Concept Discovery

## 7.1 Math

Discovery of concepts means simply clustering rows of  $S'$  matrix. Again dissimilarity matrix  $\mathcal{D}(S') = D'$  is constructed. Let assume that  $\mathcal{C}_i$  is a set of indices of semantic vectors belonging to the  $i$ th cluster. Then concept semantic vectors are mean vectors of all vectors belonging to a cluster that is:

$$\mathbf{c}_i = \frac{1}{|\mathcal{C}_i|} \sum_{j \in \mathcal{C}_i} \mathbf{s}_j \quad (15)$$

## 7.2 Example

# 8 Concept Vector Space Utilization

## 8.1 Math

To use concept vectors following information is needed:

1. all  $\mathcal{S}_i^k$  sets
2. all  $\mathcal{C}_i$  sets
3. all  $\mathbf{c}_i$  vectors

Let  $\mathbf{d}$  be a testing document then  $\mathbf{d}_l = i$  are words or collocations in that document. If  $\mathcal{S}_{\mathbf{d}_l}^k$  sets exist then  $i$  is an ambiguous word. If  $\mathcal{S}(\mathcal{S}_{\mathbf{d}_l}^k, \mathcal{N}_l) = \max_j \mathcal{S}(\mathcal{S}_{\mathbf{d}_l}^j, \mathcal{N}_l)$  then  $\mathbf{d}_l := i_k$ . This way all ambiguous words are exchanged with non-ambiguous. Next step is to exchange words with concepts. If  $\mathbf{d}_l \in \mathcal{C}_j$  then  $\mathbf{d}_l := j$ . After that  $C = (c_{i,j})_{n_a \times m_a}$  concept space matrix for the  $\mathbf{d}$  document can be constructed. If  $\exists_l \mathbf{d}_l = j$  then  $c_{j,j} = \mathbf{c}_{ij}$ . Final step is to cluster  $C$  matrix with the use of dissimilarity matrix  $\mathcal{D}(C) = D$ . Let  $\mathcal{D}_i$  be an  $i$ th cluster containing some concepts then  $\mathbf{d}$  is exchanged with several  $\mathbf{d}^i$  documents in following way:

$$\mathbf{d}_l^i = \begin{cases} \mathbf{d}_l & \text{if } \mathbf{d}_l \in \mathcal{C}_i \\ 0 & \text{if } \mathbf{d}_l \notin \mathcal{C}_i \end{cases} \quad (16)$$

where  $\mathbf{d} = \sum_j \mathbf{d}^j$ .  $\mathbf{d}^i$  is more homogenous document than  $\mathbf{d}$ . Division of  $\mathbf{d}$  into  $\mathbf{d}^i$  vectors is called *subjectization*. If occurrence of concepts in all  $\mathbf{d}_i$  will be counted then document-concept matrix can be obtained. Rows in this matrix have much smaller dimension then in document- word matrix.

## 8.2 Example