MercureO2 : adhoc and routing tasks

M. Boughanem — Université de Limoges, 123 AV. Albert Thomas, 87060 Limoges France

C. Soule-Dupuy — IRIT Toulouse, 118 Route de Narbonne, 31062 Toulouse France

email : {bougha, soule, trec}@irit.fr tel (33) 61 55 63 22

1 MercureO2 system

MercureO2 is an object oriented information retrieval system based on connectionist approach. It allows the query formulation, the query evaluation based on propagation of the neuron activation and the query modification based on backpropagation of the user judgements of the document relevance.

1.1 The model

The neural information model used in MercureO2 is shown in figure 1. It represents a set of associated neurons. We distinguish in this model :

- neuron objects

- an input layer representing the user information needs

- a term neurons layer

- a document neurons layer

- an output layer representing the result of the query evaluation

- weighted links between the neurons.

- **The MercureO2 neuron object** is characterized by : identifier, input value, output value, output threshold value, methods allowing to compute the input values and output signals, to propagate the signal, to backpropagate the signal,... The functions of this neuron are :

  - Input function : $In(N_i) = \sum_{j=1} Out(N_j) * w_{ji}$

  - Activation (or state) of the neuron at (k+1)stage is :

  $$Activation^{k+1}(N_i) = Activation^{k}(N_i) + g(In(N_i))$$

  The initial stage at $k = 0$: $Activation^0(N_i) = 0$

  - Output of the neuron : $Out(N_i) = \begin{cases} Activation^k(N_i) & \text{si } Activation^k(N_i) > threshold \\ 0 & \text{otherwise} \end{cases}$

  Where N_i is a neuron i and w_{ij} represents the weight of the link between the neurons i and j.

- **The document neuron** represents a particular document descriptor. The document descriptor is a list of terms representing the important concepts in a document. These terms are extracted by an automatic indexing operation.

- **The term neuron** represents an indexing term.

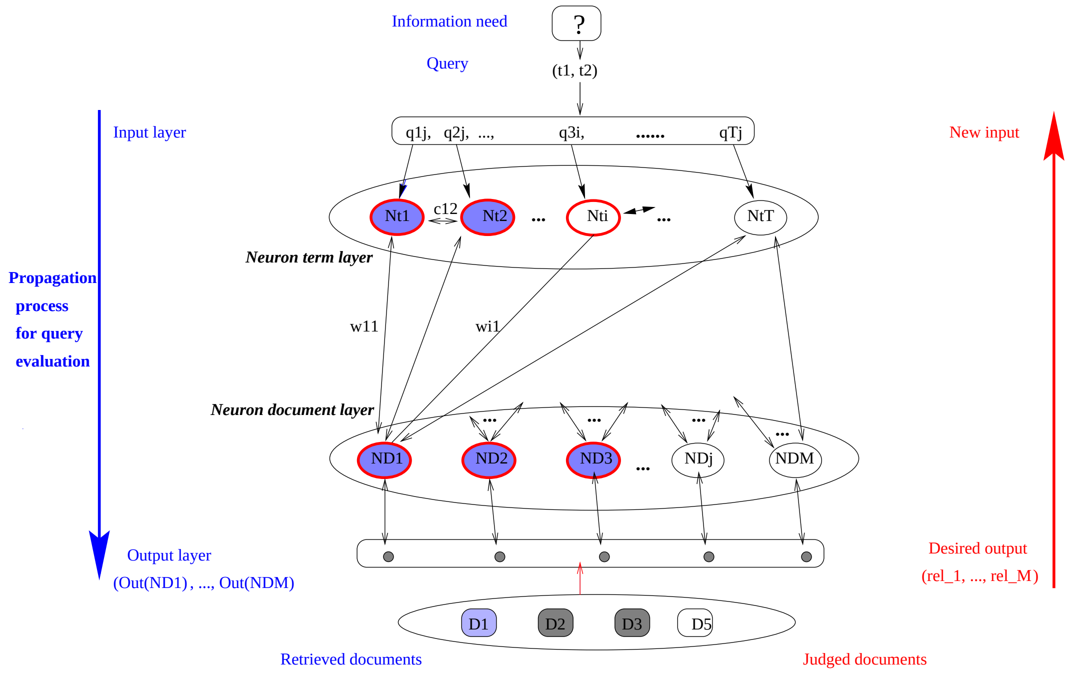

Figure 1: MercureO2 Model

- **The input layer** represents the user information needs. A set of weighted terms representing the query is associated to this information needs. The weight assigned to a term is :

$$q_{ij} = \frac{freq_{ij}}{max_j(freq_{ij})}$$

  The input layer is represented by the vector $Input_j = (q_{1j}, q_{2j}, ..., q_{Tj})$

- **The output layer** represents the obtained output for the applied Input. It is a vector of output values associated to the activated document neurons. This vector represents the retrieved documents.

- **The links :** The neurons are interconnected by weighted bidirectionnal links. Two kinds of associations are defined : term neuron to term neuron association and term neuron to document neuron association

  - **term neuron-document neuron link**
    This link represents an indexing link. The weight of this link is a combination of the term frequency (tf) and the inverse document frequency (IDF) [2]. We used the following formula :

$$w_{ij} = \frac{freq_{ij}}{\sqrt{\sum_{k=1}^{T}(freq_{kj}^2 .log(\frac{N}{N_k})^2)}} * \log(\frac{M}{m_i})$$

  - **term neuron-term neuron link**
    This weighted link represents a cooccurrence association [1] . It is defined as follows :

$$c_{ij} = \alpha . \frac{\sum_{k=1}^{T}(w_{ik} * w_{jk})}{\sqrt{\sum_{k=1}^{T} w_{ik}^2 + \sum_{k=1}^{T} w_{ik}^2 - \sum_{k=1}^{T}(w_{ik} * w_{jk})}}$$

    where :
    α : a positive constant,
    w_{ij} : the weight of the link between the term i and the document j,

$freq_{ij}$:the frequency of the term t_i in the document D_j,
M : the number of documents in the collection,
T : the number of indexing terms,
m_i : the number of documents containing the term t_i.

1.2 Query evaluation

The query evaluation is based on the neural propagation process. In MercureO2 this evaluation is performed as follows :

1. Transform the user information needs into a list of terms by series of query test processing modules. The result of query processing is the $Input_j = (q_{1j}, q_{2j}, ..., q_{Tj})$
2. Apply this input to the term layer. Each term neuron computes an input value:

$$In(N_{t_i}) = q_{ij}$$

   and then an activation value : $Activation^1(N_{t_i}) = g(In(N_{t_i}))$
   An output signal is then generated : $Out(N_{t_i}) = Activation^1(N_{t_i})$

3. These signals are propagated forwards through the network. Each neuron computes an input and an output value.

$$In(N_{D_i}) = \sum_{j=1}^{T} Out(N_j) * w_{ij}$$

   then, $Out(N_{D_i}) = Activation^1(N_{D_i}) = g(In(N_{D_i}))$

The output layer is : $Output_j(Out(N_{D_1}), Out(N_{D_2}), ..., Out(N_{D_M}))$

These output values computed by the document neurons allow to rank the list of retrieved documents. The output function we used is the sigmoid : $g(x) = \frac{e^x - 1}{e^x + 1}$

1.3 Query modification based on relevance backpropagation

The strategy we use for automatic query modification is based on the neural backpropagation algorithm. The top of retrieved documents are judged by the user. These judged documents will represent the called *Desired output*. To each judged document is assigned a *relevance value*, positive relevance value is assigned to relevant document, a negative value to non relevant document and zero for not judged document. So, the desired output is represented by the vector of the form $DesiredOutput = (rel_1, ..., rel_i, , ..., rel_M)$ This strategy consists to backpropagate the relevance values from the Output layer to input layer. Thus, The desired output will be applied to the document neurons, activations will be computed by each neurons in the document neurons and then in term neurons. The result of this backpropagation is a new input.

This strategy is defined as follows :

1. Build the desired output : $DesiredOutput = (rel_1, ..., rel_i, , ..., rel_M)$,
2. Apply this output to the neuron document layer. Each neuron computes an input value : $In(N_{D_i}) = rel_i$
   And then an ouput signal : $Out(N_{D_i}) = Activation^1(N_{D_i}) = g(In(N_{D_i}))$
3. The output signals are backpropagated to term neuron layer. Each neuron term computes an input value : $In(N_{t_i}) = \sum_{j=1}^{M}(w_{ij} * Out(N_{D_j}))$
   and then an output signal $Out(N_{t_i}) = Activation^1(N_{t_i}) = g(In(N_{t_i}))$
4. A new input is then computed according to this formula : $NewInput_j = Input_j + \alpha * Out(N_t)$
5. This new input is applied to the term neuron layer and a new query evaluation is then done. This process (propagation of the input and backpropagation of the desired output) can be repeated several times ; *the iteration number* is a parameter in MercureO2.

Parameters	Routing		Adhoc	
Run Ids	mer-r1	mer-r2	mer-a1	mer-a2
Items used	All	All	All	description
Training Doc	ALL(cat B)	All(catB)		
NbJudgedDocs	50	50	30	30
Coef_Rel (relevant doc)	0.5	0.5	0.5	0.5
Coef_Rel (non relevant)	-0.25	-0.25	-0.25	-0.25
β	0.5	0.5	0.5	0.5
Number of iteration	1	2	1	1
α	0.5	0.5	0.5	0.5
NbAdded terms	60	60	50	50

Table 1: Trec runs parameters

Routing results			
Run	Best	$\geq$median	$< median$
mer-r1	14	21	8
mer-r2	27	14	2

Adhoc results			
Run	Best	$\geq$median	$< median$
mer-a1	10	29	6
mer-a2	7	29	9

Table 2: Comparative TREC Results

For desired output several formulations can be used, for this experimentation we used :

$$rel_i = Coef\_Rel * log_2(\frac{NbJudgedDoc}{NbRel} + \beta)$$

where :
Coef_Rel : relevance coefficient of the documents (positive for relevant and negative for non relevant document)
NbRel, NbNRel : number of relevant and non relevant documents respectively
NbJudgedDoc : number of judged documents

2 Description of TREC Experiment

Our experiment is done with MercureO2. The initial queries are built automatically and then modified (reweighted and expanded) by using our relevance backpropagation strategy. The terms of the new query are deduced as explained previously. We easily note that after this process the initial query will be expanded and reweighted. For the two experiments (Adhoc and routing) we limited the number of terms to add. We consider the new input and we sort all terms do not appear in the initial query and we add only the *NbAdded* top terms. For TREC 5 experiment we submitted two runs for routing (mer-r1 and mer-r2 with 1 iteration and 2 iterations respectively) and two runs for adhoc (mer-a1 where we used all the items in the topic for query construction and mer-a2 we used only the description item). The values of the different parameters we used are listed in (Table 1).

The results we obtained in comparison with other TREC category B runs are shown in (Table 2).

References

[1] D., J. Harper, & C., V. Van Rijsbergen, *An evaluation of feedback in document retrieval using co-occurrence data.* Journal of Documentation, 34(4), 189-216, 1978.

[2] G. Salton & M., J. Allan, *Automatic Text Decomposition and structuring* RIAO'94 New York 1994. (USA).