Classification by resonance in an associative network

Christophe Brouard
Laboratoire d'Informatique de Grenoble,
BP 53 - 38041 Grenoble, France
Christophe.Brouard@iut2.upmf-grenoble.fr

Abstract. This paper presents the system we designed for our participation to the LSHTC Pascal Challenge. In the training phase, this system builds an associative network linking the terms present in the different documents to the different classes. In the classification phase, each new document is represented by an activation pattern. The activation spreads in the built network and a degree of "resonance" between the considered document and the different classes is computed. This system can be tuned with different parameters and the results of the different experiments aiming at tuning these parameters are given. At last, practical considerations concerning the computational complexity and the execution times are also discussed.

Keywords: classification, associative network, resonance.

1 Introduction

The LSHTC Pascal Challenge[1] gave the opportunity to its participants to experiment their classification systems and to compare them on a corpus extracted from the Open Directory Project[2]. The Challenge proposed four different tasks. We focused on the Basic task taking into account the content of the documents and ignoring the human description of the documents. The aim of our participation was to test in a classification task the general principles of a system that we previously used in an adaptive filtering task [1]. This system evaluates the relevance of a document for particular query or a particular class by computing a degree of "resonance" in an associative network.

In the next section, we briefly present related works. In the third section we describe more precisely our training and classification algorithms. In the fourth and fifth section, we present the results we obtained respectively on the dry-run and the large datasets for the Basic task. We also explain how the different parameters of the system are tuned. In another section, practical considerations concerning the computational complexity and the execution times are discussed. In the last section, the perspectives of this work are given.

[1] http://lshtc.iit.demokritos.gr
[2] http://www.dmoz.org/

2 Related works

The system we designed in order to participate to the LSHTC Challenge is based on a spreading activation method. The idea of information selection based on spreading activation methods in networks is not new. In the context of semantic networks, numerous methods have been developed and even if the idea seems interesting, the results have not been always very satisfying [2]. Numerous problems have been encountered. One of these concerns the artificial constraints necessary to avoid the activation of all the nodes of the network. We believe that the concept of resonance can be a natural way of controlling spreading activation. This concept has been introduced in the Adaptive Resonance Theory (ART) in the context of neural networks by Grossberg [3]. It has been applied to numerous problems from classification tasks to cognitive models. Our goal is to apply this mechanism with an analogy less developed with the neural system. However, we keep the architecture of the network, i.e., two layers linked by oriented connections, one for representing the inputs, the other one for representing the classes. We keep also the notion of Hebbian learning (reinforcement of connections between nodes simultaneously activated). Lastly, we consider a kind of resonance, comparing a back-propogated activation with the initial pattern activation corresponding to an input and selecting the category which maximizes the similarity of these activations.

3 Algorithm description

The training phase corresponds to the construction of the network and the classification phase corresponds to the utilization of the built network.

3.1 Network construction (training)

In the training phase, the system connects the nodes representing the terms of the documents to the nodes representing the classes of these documents. The connections between the nodes are oriented and weighted (Fig. 1).

The weight of the connection from a node i to a node j corresponds to the relative frequency of the concept represented by j given of the concept represented by i. For the calculation of these frequencies, we consider degrees of membership [4]. Let $\mu_D(i)$ denotes the degree of membership in the real unit interval [0,1] of a term or a class i to a document D. Then, the weight is defined as follows:

$$W_{ij} = \frac{\sum_D \mu_D(i).\mu_D(j)}{\sum_D \mu_D(i)} \quad . \qquad (1)$$

So, if *i* corresponds to a term and *j* corresponds to a class then W_{ij} corresponds to the proportion of documents in class *j* among all documents in which the term *i* occurs. If *i* corresponds to a class and *j* corresponds to a term then W_{ij} corresponds to the proportion of documents containing the term *j* among all the documents in the class *i*. A membership degree different from 0 or 1 is only used for the terms.

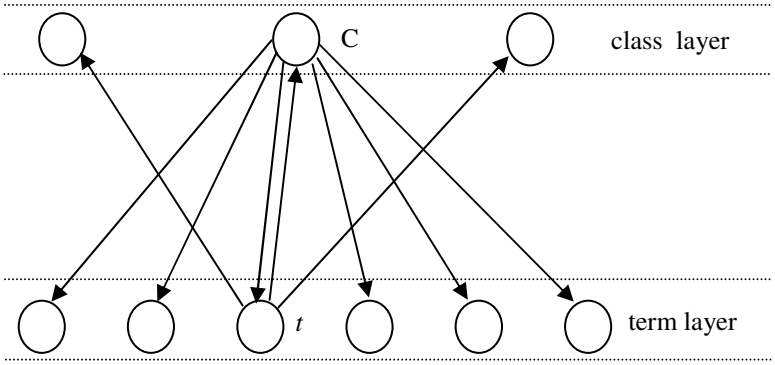

Fig. 1. The built network. The connections are oriented from a term to a class or from a class to a term. For readability reasons, only the connections of a particular term *t* and a particular class C are represented in the figure.

3.2 Network utilization (classification)

3.2.1 Resonance computation

In the classification phase, each new document is represented by an activation pattern in the term layer, i.e., the nodes representing the terms of the document are activated. The activation spreads in the built network and a degree of resonance between the considered document D and each class C is computed. This degree of resonance measures the similarity of the initial activation pattern representing the document D with the activation pattern resulting from a propagation to C and a back-propagation from C to the term layer. This degree of resonance is defined as follows:

$$R(D, C) = \sum_{t \in D} a(t).\frac{W_{Ct} \cdot \sum_{t \in D} a(t).W_{tC}}{\sum_{t' \in V} W_{Ct'}} \quad . \qquad (2)$$

where *a(t)* denotes the initial activation of the term *t* and V denotes the set of all the terms. This formula simply corresponds to the dot product between the vector representing the document (initial activation pattern) and the vector representing the activation pattern in the term layer after propagation to C and back-propagation from

C. The calculation of the activation of the term *t* after propagation and back-propagation corresponds simply to the product of the sum of the activation received by C from the term layer with the proportion of the activation back-propagated from C to *t*. In this calculation we always consider that the activation of a node is spread and divided according to the weight of the connections. We also consider that the activation of a node corresponds to the sum of the activation received from the connected nodes.

3.2.2 Top-down versus bottom-up pathways

The previous formula can be rewritten as follows:

$$R(D, C) = \left[\frac{\sum_{t \in D} a(t).W_{Ct}}{\sum_{t' \in V} W_{Ct'}} \right] . \left[\sum_{t \in D} a(t).W_{tC} \right] . \quad (3)$$

This new formulation makes appear two different factors. The first factor corresponds to the top-down pathway from C to D. It measures the presence of the terms which are the most frequent in the documents of this category. The second factor corresponds to the bottom-up pathway from D to C. It measures the presence of terms specific to the class C. We give the possibility to vary the importance of each factor by introducing a new parameter *p* as follows :

$$R(D, C) = \left[\frac{\sum_{t \in D} a(t).W_{Ct}}{\sum_{t' \in V} W_{Ct'}} \right]^{p} . \left[\sum_{t \in D} a(t).W_{tC} \right]^{2-p} . \quad (4)$$

3.2.3 Class decision

A score is computed for all the classes and the document is assigned to the class corresponding to the largest score.

$$Decision(D) = \underset{C}{ArgMax}\,(R(D,C)) \ . \quad (5)$$

4 Results on the dry-run dataset

Here we present the results of the experiments that we have done in order to tune the parameters. All the parameters are tuned using the validation file. The evaluation measure used in order to tune the parameters corresponds simply to the proportion of correct classifications (hit rate). The validation file is also used in the training phase, i.e., the training set and the validation set are put together. For readability reasons, we present the results concerning the different parameters independently. Thus, for the presentation of the results concerning one particular parameter we fix the others to their optimal value. No dependencies have been observed between the different parameters.

4.1 General Parameters tuning

4.1.1 Membership degree in the training phase

As we mentioned above, a membership function must be defined in the learning phase. Concerning the class nodes, the situation is binary and the solution is simple, i.e., the membership degree is 1 when the document is in the class, the degree is 0 when it is not. Concerning the term nodes, after different tests we have chosen to define it as follows :

$$\mu_D(t) = \min\left(1, \frac{nb\_occ(D,t)}{nb\_terms(D).pm}\right) . \qquad (\mathbf{6})$$

where *nb_occ(D,t)* corresponds to the number of occurrences of the term *t* in the document D, *nb_terms(D)* corresponds to the number of terms in the document D and *pm* is a constant in [0,1]. So if the proportion of the term *t* is larger than *pm*, then the membership degree is 1 (we consider that the term is totally present). Otherwise, the membership degree corresponds to the ratio between the proportion of the term with *pm*. The value of *pm* has been learned on the validation file. The best result is obtained with *pm*= 1/60 (see table 1). If *pm* is very little, the membership degree is 1 if the term occurs in the document and 0 otherwise.

Table. 1. Impact of value of *pm* parameter on the percentage of correct classifications (hit rate) for the validation file (Basic task).

1/pm	**1**	**10**	**40**	**50-60**	**70**	**80**	**100**	**10000**
hit rate	41.4	43.0	47.8	48.1	47.8	47.8	47.0	42.7

4.1.2 Activation function in the classification phase

As we mentioned above, an activation function must be defined in the classification phase. After different tests we have chosen to use the same kind of function as for the membership degree. So, the activation of a term *t* for a document D is defined as follows:

$$a(t) = \min\left(1, \frac{nb\_occ(D,t)}{nb\_terms(D).pa}\right) . \quad (7)$$

where *nb_occ(D,t)* corresponds to the number of occurrences of the term *t* in the document D, *nb_terms(D)* corresponds to the number of terms in the document D and *pa* is a constant in [0,1]. So if the proportion of the term *t* is larger than *pa*, then the activation is 1. Otherwise, the activation corresponds to the ratio between the proportion of the term with *pa*. The value of *pa* has been learned on the validation file. The best result is obtained with *pa*= 1/15 (see table 2).

Table 2. Impact of value of *pa* parameter on the percentage of correct classifications (hit rate) for the validation file (Basic task).

1/pa	**1**	**5**	**10**	**15**	**20**	**50**	**100**	**10000**
hit rate	46.8	46.9	47.8	48.1	47.7	44.5	43.3	42.4

4.1.3 Feature selection

We remove the most frequent terms. In order to do this selection, the traditional *idf* weight is computed for each term and only the terms with an *idf* weight larger than a particular threshold *th_idf* are kept. The *idf* weight of a term *t* is defined as follows :

$$idf(t)=Log\left(\frac{N}{df(t)}\right) . \quad (8)$$

where N is the number of documents and *df(t)* corresponds to the number of documents containing *t*. After different tests, the threshold *th_idf* has been fixed to 2.2 (see table 3). Since we use the natural logarithm, this means that we discard all the terms that occur in more than 11% of all documents. We also tried odds-ratio and mutual information measures without improving the results obtained with the *idf* measure.

Table 3. Impact of feature selection on the percentage of correct classifications (hit rate) for the validation file (Basic task).

threshold	**0**	**1.0**	**1.5**	**2.0**	**2.1**	**2.2-2.5**	**2.8**	**3.0**
hit rate	45.9	45.9	46.9	47.2	47.8	48.1	46.6	46.1

4.2 Top-down versus bottom-up pathways: results

For the Basic task, the largest number of correct classifications is obtained for a value of *p* respectively equal to 1.5 and 1.55. For these values we obtain 46.8% of correct classifications (895 documents among 1860) for the validation file and 49.9% (927 documents among 1858) for the test file. Thus, the results are clearly better when we favour the top-down pathway. The worse results are obtained when we only consider bottom-up pathway (*p*=0) or top-down pathway (*p*=2).

Table 4. Percentage of correct classifications for the different values of *p* and for the validation and the test files (Basic task).

p	**0.0**	**0.5**	**1.0**	**1.2**	**1.4**	**1.5**	**1.6**	**1.8**	**2.0**
validation	23.4	30.8	39.8	44.1	47.4	48.1	47.0	39.5	26.0
test	24.3	32.0	41.4	44.9	48.2	49.5	49.2	44.3	33.0

5 Results on the large dataset

For the large dataset, we considered the following values *pm*=1/60, *pa*=1/10, *p*=1.6 and *th_idf*=2.2. These values have been chosen after different tests on the validation file with values corresponding to the best results for the dry-run dataset. The results uploaded to the site of the LSHTC Pascal Challenge do not correspond to the system described above. For our new system, we obtained 39.6% of correct classifications on the validation file. The application of the system to the test file (via Oracle) with the same parameters gives the following results :

Table 5. Results on the large dataset (Basic task).

Accuracy	**F-measure**	**Precision**	**Recall**	**Tree Induced Error**
0.415711	0.31604	0.29429	0.34126	3.57142

6 Practical considerations

6.1 Computational complexity

The training phase consists in building the network. As we can guess from the formula 1, the construction method can be incremental, i.e., the connections can be updated for each new document. Since only the two oriented connections concerning the class of the document must be updated for each term, if n_t denotes the average number of terms in a document, the algorithm takes $O(n_t)$ time for taking into account a new document. If n_d denotes the number of documents, the algorithm takes $O(n_d)$ time for building the network.

For each new document, the classification phase consists in computing a score of the document for each class. The resonance computation is done by a single traverse of the list of the connections for each activated term node. So if n_c denotes the average number of connections for activated nodes, the algorithm takes $O(n_c)$ time for computing all the scores. A connection linking a term to a category (and the reverse connection) is built when a term occurs in a document of the category. However n_c is not easy to formulate. The number of the connections for activated nodes is larger than for other nodes since the most frequent terms are more connected. It depends on the number of classes, the number of documents, the average number of terms in each document and the homogeneity of documents in the same classes (do they contains the same terms ?). Moreover, in order to increase the speed of the algorithm (without loss of performance), we remove all the connections between i and j such that $p(i\&j)<p(i).p(j)$ where $p(i)$ denotes the frequency of i. In these cases, the connections do not correpond to a "positive" dependence between i and j, the value of the connection is simply due to the frequency of i and j. In our experiments this corresponds to 20% of the connections and saves roughly 20% of the classification time.

Table 6. Numbers of nodes and connections for the dry-run dataset (6 323 documents train+validation) and large dataset (128 710 documents train+validation) after simplifications (Basic task).

nodes		connections	
dry-run	large	dry-run	large
56 856	393 558	496 326	9 205 980

6.2 Training and classification times

The system is implemented in Java. It is run on a PC with an Intel Core Duo 2Ghz CPU with 2Go of RAM. The training and classification times for the dry-run and large datasets are given below. We can observe that the training time for the large

dataset is approximatively 20 times larger than the training time for the dry-run dataset as we expected. This corresponds to the ratio of datasets sizes. If we compare the classification times for one document in the two cases, it appears that the classification is 13 times faster for the dry-run dataset. It corresponds to the ratio of the average number of the connections for activated nodes as explained above.

Table 7. Times in seconds of the training and classification phases for the dry-run and large datasets (Basic task).

Training		Classification	
dry-run	**large**	**dry-run**	**large**
8	155	6	1480

7 Conclusions

The method we have developed must be refined. Other activation functions could be chosen. The parameter p controlling the relative importance of top-down and bottom-up must be compared on different corpus. More generally, both linear classifiers corresponding respectively to top-down and bottom-up pathways could be combined differently. The method must also be compared with other well-known methods. All these issues will be addressed in the future.

However the comparison with the results obtained by the other participants (in particular on the dry-run dataset) are pretty good. The computations are relatively simple and consequently the training and classification times are short. This appears as an important advantage. Moreover, both propagations in the resonance computation can individually be done in parallel. We could exploit it by implementing the computation on an adapted hardware architecture.

References

1. Brouard, C., Nie, J.Y.: Relevance as resonance: a new theoretical perspective and a practical utilization in information filtering. Inf. Proc. & Manag. 40, 1--19 (2004)
2. Crestani, F.: Application of Spreading activation Techniques in Information Retrieval, Artificial Intelligence Review, 11, 453--482 (1997)
3. Grossberg, S.: Adaptive pattern classification and universal recoding. I. Parallel development and coding of neural feature detectors. Biol. Cyber., 23, 117--140 (1976)
4. Zadeh, L.A.: Fuzzy Sets. Information and Control, 8, 338--353 (1965)