IRIT at TREC'2002: Filtering Track

M. Boughanem, H. Tebri and M. Tmar

IRIT/SIG
Campus Univ. Toulouse III
118, Route de Narbonne
F-31062 Toulouse Cedex 4
Tel: 33 (0) 5 61 55 74 16

Email: $\{boughane, tebri, tmar\}@irit.fr$

Abstract

The experiments we undertaken this year for TREC2002 Filtering track, are focussed on threshold calibration. We proposed a new approach to calibrate the dissemination threshold in an adaptive information filtering. It consists of optimizing a utility function represented by a linearized form of the probability distributions of the scores of the relevant and the non-relevant documents already filtered. The profiles are learned using the same method used last year. It is based on a reinforcement algorithm. We submitted results on three tasks: adaptive, batch and routing.

1 Information representation and filtering

Our adaptive filtering model is inspired from the connectionist model Mercure [1]. The profile and the document are represented by a set of weighted terms. The filtering process consists of computing a relevance value *RSV Retrieval Status Value*. The document is delivered only if the *RSV* is greater than the dissemination threshold. A learning process is then carried out by modifying the profile and the dissemination threshold to be more efficient in the future.

1.1 System initialization

The user profile is represented by a set of terms :

$$p^{(0)} = \left(\left(tp_1, w_1^{(0)}\right), \left(tp_2, w_2^{(0)}\right) \ldots \left(tp_n, w_n^{(0)}\right)\right) \quad (1)$$

where tp_i is a term and $w_i^{(0)}$ is its weight in the initial profile (at $t = 0$), in the rest of this paper, the term-profile weight is noted $w_i^{(t)}$ where t represents the instant when the system receives a document. Initially, the term-profile weight is computed as follows:

$$w_i^{(0)} = \frac{tfp_i}{max_j\left(tfp_j\right)} \quad (2)$$

Where tfp_i is the term frequency of tp_i in the profile. The formula seems to be abusively simplistic, but at the beginning of the filtering process, no information is known but a set of terms and their occurrence frequencies in the initial user profile. However, this weight will be adjusted by learning.

1.2 Filtering incoming documents

Each incoming document at time t is indexed, to build a list of stemmed terms (Porter [3]), the terms belonging to a stop-list are removed. Each term is then weighted according to the following formula:

$$d_i^{(t)} = \frac{tf_i^{(t)}}{h_3 + h_4 * \frac{dl^{(t)}}{\Delta l^{(t)}} + tf_i^{(t)}} * log\left(\frac{N^{(t)}}{n_i^{(t)}} + 1\right) \quad (3)$$

Where $tf_i^{(t)}$: term frequency t_i in the document $d^{(t)}$, h_3, h_4: constant parameters, for the experiments $h_3 = 0.2$ and $h_4 = 0.7$, $dl^{(t)}$: document length $d^{(t)}$(number of index terms), $\Delta l^{(t)}$: average document length,

$N^{(t)}$:number of incoming documents until time t, $n_i^{(t)}$: number of incoming documents containing the term t_i. This weighting formula is a form of *Okapi* [5] term-query weight used in the information retrieval system *Mercure*.
$N^{(t)}$, $n_i^{(t)}$ and $\Delta l^{(t)}$ are collection based parameters that are computed on the cumulative documents filtered at time t. They can not be known while the document stream does not stop sending documents, the used values are recomputed for each incoming document.
A relevance value noted $rsv\left(d^{(t)}, p^{(t)}\right)$ is then computed corresponding to the document and the profile, as follows:

$$rsv\left(d^{(t)}, p^{(t)}\right) = \sum_{t_i \in d^{(t)}, tp_j \in p^{(t)} \text{ and } t_i = tp_j} d_i^{(t)} * w_j^{(t)} \tag{4}$$

The binary decision rule used for document filtering is the following:

$$\begin{cases} if\ rsv\left(d^{(t)}, p^{(t)}\right) \geq threshold^{(t)}\ accept\ the\ document \\ otherwise\ reject\ the\ document \end{cases}$$

The threshold value is initially fixed to a low value, 0 in our experiment. It is modified while filtering.

1.3 Profile adaptive learning

The learning process adopted in our case is incremental. It is processed at each filtered document judged as relevant by the user. The basic idea of our learning process is based on the relevance reinforcement process. When a document is judged as relevant, it is necessary to be able to find a new representation of the profile which makes it possible to find the document with a strong score. In other words, one will be brought to improve the profile such as $rsv\left(d^{(t)}, p^{(t)}\right) = \beta$ where β is the desired score rsv. This technique was used in TREC-10, we do not give more details, the reader can be refer to TREC-10 [1].
The problem is assimilated to a linear equation resolution. Each solution of this system is a set of weights affected to the document terms, knowing that there is an infinite number of solutions, we add a constraint to obtain only one solution. The system to resolve is given by :

$$\begin{cases} \sum_{\substack{t_i \in d^{(t)} \\ tp_j \in p^{(t)} \\ t_i = tp_j}} d_i^{(t)} w_j^{(t)} = \beta \\ \frac{w_i^{(t)}}{f\left(d_i^{(t)}, r_i^{(t)}, s_i^{(t)}\right)} = \frac{w_j^{(t)}}{f\left(d_j^{(t)}, r_j^{(t)}, s_j^{(t)}\right)} \forall\, (t_i, t_j) \in {d^{(t)}}^2 \end{cases} \tag{5}$$

The solution of the system 5 is a set of "provisional" weights, let n be the number of different terms in the processed document at time t, $f_i^{(t)} = f\left(d_i^{(t)}, r_i^{(t)}, s_i^{(t)}, R^{(t)}, S^{(t)}\right) \forall i \in \{1 \ldots n\}$ where $r_i^{(t)}$(resp. $s_i^{(t)}$) is the number of relevant (resp. non-relevant) documents containing the term t_i until the time t. For each term appearing in the document, the provisional weight solution of the system 5 is the following:

$$\forall i,\ pw_i^{(t)} = \frac{\beta f\left(d_i^{(t)}, r_i^{(t)}, s_i^{(t)}\right)}{\sum_j f\left(d_j^{(t)}, r_j^{(t)}, s_j^{(t)}\right) * d_j^{(t)}} \tag{6}$$

The function f is proportional to the term importance. The function f used for the experiments is the following:

$$\forall i, f\left(d_i^{(t)}, r_i^{(t)}, s_i^{(t)}\right) = d_i * log \frac{\frac{r_i^{(t)} + 0.5}{R^{(t)} - r_i^{(t)} + 0.5}}{\frac{s_i^{(t)} + 0.5}{S^{(t)} - s_i^{(t)} + 0.5}} \tag{7}$$

$R^{(t)}$ (resp. $S^{(t)}$) is the total number of relevant (resp. non-relevant) documents until the time t.
The "provisional" weight $pw_i^{(t)}$ contributes in learning the term-profile weight corresponding to the term t_i, we use the following gradient propagation formula:

$$w_i^{(t+1)} = w_i^{(t)} + log\left(1 + pw_i^{(t)}\right) \tag{8}$$

We add 1 to $pw_i^{(t)}$ to avoid adding negative value to the term-profile weight.

1.4 Threshold calibration

To find the best threshold, the system should follow the document score evolution and regulate the threshold in order to select as maximum as possible the relevant documents and reject as maximum as possible the non-relevant documents. The threshold regulation can be made by maximizing a utility function[1]. According to the sampling theory, the behavior of a random sample is the same of all the population, so the threshold allowing to maximize the utility function in a random sample of documents allows to maximize the same utility function in all the documents of the stream. The approach we propose for threshold calibration consists of estimating the discrete probabilities of the scores of the relevant and the non-relevant documents already filtered and then using a kind of probability plotting method to build a linearized probability density distribution. The threshold that maximizes a utility function, represented by both distributions, is then selected.

1.4.1 Score distribution modelling

The probability that a random document has a particular score is equal to the number of documents having the same score divided by the total number of documents:

$$p(X = score) = \frac{|\{d|rsv(d,p) = score\}|}{|\{d\}|} \tag{9}$$

As score values are very distinct, they tend to be equiprobable ($|\{d|rsv(d,p) = score\}| = 1$ or 0). Indeed, it is very difficult to find two or many documents having exactly the same score. Consequently the score probability distribution tends to be uniform.
Instead of computing the probability that a document has a score, we compute the probability that the document score belongs to an interval [2]. We define a set of intervals enough reduced so that the document scores belonging to the same interval are really close. We define n adjacent intervals $I_1, I_2 \dots I_n$ having the same ray:

$$I_i \quad = \quad [score_{i-1}, score_i] \tag{10}$$

Where : $score_0 = \min_d rsv(d,p)$ and $score_n = \max_d rsv(d,p)$.
The number of intervals is proportional to the sample size, indeed the great is the number of documents, the great is the definition field of the document scores. We define n as the half of the total number of documents: $n = \frac{|\{d\}|}{2}$.
The probability that a document score belongs to an interval is given by:

$$p(score_i < X < score_{i+1}) = \frac{|\{d|score(d,p) \in]score_i, score_{i+1}[\}|}{|\{d\}|} \tag{11}$$

The representation of the probability distribution based on formula 11 corresponding to documents scores is poissonnian. There is many methods to estimate the probability law parameters followed by the document scores: the parametric regression and the maximum likelihood estimation method [4]. In both cases, the law must be assumed to be known in advance, these methods allow to estimate the parameters of this function. But, in an experimental context many limits of these methods can be noted. In deed, even though the assumption that the scores follow a known distribution density could be acceptable, but it strongly depends on the experimental conditions such as the filtering and the learning approaches and the size (number of documents) of the sample used for deriving the parameters of the distributions, the sample size must be enough important to obtain non-skewed estimations. To resolve these problems we propose that instead of assuming that the distributions are known, they are built by estimating the discrete probabilities of the scores of the relevant and the non-relevant documents delivered at a certain time and then by linearizing these probabilities to obtain the corresponding probability density distributions. A utility function, to be optimized, is then represented using these distributions. The best threshold is the threshold that maximizes this utility.

[1] *a function used to evaluate the filtering systems performance*

[2] *There is a method to estimate the parameters of a probability law allowing to define a more or less precise interval containing this parameter called confidence interval rather than a value which will be less probably equal to this parameter*

1.4.2 Probability distribution linearization

Based on the regression theory, the representative curve of any function can be linearized in order to facilitate many tasks (computing surfaces, searching an extremum ...). The linearization consists of considering the domain of the function and divide it into a set of intervals such as the representative curve of the restriction of that function in each interval can be assimilated to a linear curve.

We use this technique to linearize the representative curve of the probability density distribution of the scores. We assume that this function exists and we try to linearize it. As we do not know this function, we propose to linearize it using the corresponding discrete probabilities. The first stage of this process is to identify a set of linear intervals. We define a linear interval by two scores $[score_x..score_y]$ where $score_x < score_y$ and all the points formed by (s_i, p_i) fit a straight line, s_i is a score in that interval and p_i is the discrete probability of s_i computed according to Formula 11. The linearity of a set of points is measured using the least squares method [4]. The least squares method requires that a straight line be fitted to a set of points such that the sum of the squares of the distance of the points to the fitted line is minimized. In our work the detection of linear interval is measured incrementally by considering all the points (s_i, p_i), ordered in increasing order of the scores. It consists of adding a point to a given set of points representing a straight line and computing an error which measures the standard deviation between that "new" set and a linear curve. If this error is below than a "linearity threshold", the considered point is definitely added to that set, the next point is then considered. Otherwise this point is removed from the set, and we continue the search of a new linear interval, and so on until the last point of the distribution. The following algorithm is applied:

1. $c = 1$ *(c is the index of the classes of the points with scores within a linear interval)*
2. $P = \emptyset$,
3. $threshold\_error = 0.0001$,
4. $< M + 1$ *: the total number of the points of the form* (s_i,p_i). /*We consider that the points are ranked in increasing order of their scores.* $i = 0$ *in the first score.*
5. *for* $i \in \{0 \dots M\}$,
   (a) $P \leftarrow P \cup \{i\}$,
   (b) *determine the equation of the line* $D_c : y(x) = a + bx$ *based on the linear regression for all points* $(s_j, p_j) \forall j \in P$,
   (c) *compute the standard deviation error between the points* $(s_j, p_j) \forall j \in P$ *and the line* D_c*:*

$$E = \sum_{j \in P} d^2((s_j, p_j), D_c) \quad (12)$$

$$d^2((s_j, p_j), D_c) = (\frac{a + b.s_j - p_j}{\sqrt{a^2 + 1}})^2 \quad (13)$$

   (d) *if* $E > threshold\_error$,
      i. /*a class of points is formed.
      $C_c = (d_c, f_c, a_c, b_c)$ *where,* $d_c = \min(s_j)$, $f_c = \max(s_j)$ $\forall j \in P$, a_c *and* b_c *are the coefficients of the equation of the line* $y = a_c + b_c x$ *derived using the linear regression of all the points* (s_j, p_j) *where* $j \in P|\{i\}$,
      ii. $P \leftarrow \{i\}$, /* *re-initialize* P
      iii. $c \leftarrow c + 1$.
   (e) *end if*
   (f) *end for*

A transformation is necessary at this stage such that all lines form a continuous representation:

1. *Transform the representation into a continuous one by relying the extremities of two adjacent classes. This liaison is done as follows: for two adjacent linear classes C_c and C_{c+1}, rely f_c and d_{c+1} with a line having as equation $y = \alpha_c + \beta_c x$. This line should pass through the points $(f_c, a_c + b_c f_c)$ and $(d_{c+1}, a_{c+1} + b_{c+1}.d_{c+1})$, so:*

$$\alpha_c = \frac{a_c + b_c.f_c - a_{c+1} + b_{c+1}.d_{c+1}}{f_c - d_{c+1}} \quad (14)$$

$$\beta_c = a_c + b_c.f_c - \frac{a_c + b_c.f_c - a_{c+1} + b_{c+1}.d_{c+1}}{f_c - d_{c+1}} f_c \quad (15)$$

2. *Normalize the coefficients a_c, b_c, α_c and β_c such that:*

$$\int_{score_0}^{score_M} f(x)dx = 1 \quad (16)$$

The equation 16 is the fundamental property of the probability density functions. As $\int_{score_0}^{score_M} f(x)dx$ represents the surface formed by the graphical representation of f and X-coordinate axis, the coefficients a_c, b_c, α_c and β_c are divided by this surface. This surface becomes unit. This surface is computed as the sum of the surfaces formed by all the linear intervals and the X-coordinate axis, let cl be the number of linear classes, so:

$$\int_{score_0}^{score_M} f(x)dx = \frac{1}{2}(\sum_{c=1}^{cl}(f_c - d_c)(2a_c + b_c.(f_c + d_c)) + \sum_{c=1}^{cl-1}(f_c - d_{c+1})(2\alpha_c + \beta_c(f_c + d_{c+1})))$$

1.4.3 Threshold optimization

The proposed method is based on utility maximization. Generally the utility function is done by :

$$F = aR_+ - bS_+ \quad (17)$$

Where R_+ (resp. S_+) is the number of relevant (resp. non-relevant) selected documents.
Our goal is to determine a threshold allowing to maximize the theoretical value of F :

$$threshold^* = arg \max_{threshold} F \quad (18)$$

Where a, b : positive constants, R_+ : number of relevant selected documents, S_+ : number of non-relevant selected documents. R_+ and S_+ are both inversely proportional according to the threshold,

$$R_+ = p(r|score > threshold) * R \quad (19)$$

$$S_+ = p(s|score > threshold) * S \quad (20)$$

R and S represent the total number of relevant and non-relevant documents examined.
Based on Bayes transformation rule, we obtain :

$$R_+ = \frac{p(score > threshold|r) * p(score > threshold}{p(r)} * R \quad (21)$$

$$S_+ = \frac{p(score > threshold|s) * p(score > threshold}{p(s)} * S \quad (22)$$

$p(score > threshold|r)$ (resp. $p(score > threshold|s)$) represents the probability that a document is selected when it is relevant (resp. non-relevant). It represents the surface formed by the curve of function f corresponding to the relevant (resp. non-relevant) documents and the X-coordinate axis.

However, $p(r) = \frac{R}{N}$ (resp. $p(s) = \frac{S}{N}$) is the probability that a document is relevant (resp. non-relevant). Utility done by equation 18 is equivalent to:

$$\begin{aligned} F &= p(score > threshold) * N * (a * p(score > threshold|r) \\ &\quad + b * p(score > threshold|s)) \end{aligned} \tag{23}$$

The retained threshold value allows to maximize F.

2 Experiments and results

2.1 Results of adaptive filtering task

The pre-test documents were used to learn the profile and to set the initial threshold. Then the filtering process is processed on the test data. At each selected the relevant document, the profile is learned and the new threshold is also computed. The algorithm that has been used for this experiment is the following:

- if a document d^t is selected and is judged as relevant,
- learn the profile,
- re-estimate the scores of all delivered relevant documents and of a sample of 1000 non relevant documents extracted from the training data that have been used in batch,
- build the linearized probability distributions of these samples,
- Compute the new threshold that maximizes the utility function $T11U = 2R_+ - S_+$. This threshold is measured by varying the threshold value from the smallest score of the relevant documents to the greatest score of the non-relevant documents.

Table 1 lists the comparative adaptive filtering results for all topics.

TREC adaptive filtering for topics 101-150				
Evaluation	$= max$	$\geq median$	$< median$	Avg
$T11U$	2	31	19	0.386
$T11F$	1	28	22	0.387
Set precision	0	24	26	0.261
Set recall	0	30	20	0.409
TREC adaptive filtering for topics 151-200				
$T11U$	1	43	7	0.282
$T11F$	3	44	6	0.054
Set precision	3	44	6	0.092
Set recall	0	41	9	0.031

Table 1: Comparative adaptive filtering results

2.2 Batch and Routing Experiments

In batch and routing tasks the profile and the threshold were learned from the training collection. The learned profile and threshold were applied to the test data.

2.2.1 Batch filtering

We built a sample of 1082 documents, which corresponds to the documents of all the profiles which have been labelled as relevant in the training dataset. This sample is then used to learn the profile and the dissemination threshold. We only consider the 40 top terms. The learned profiles and thresholds were applied to the test database.
Table 2 lists the comparative batch results for all topics.

TREC batch filtering of topics 101-150				
Evaluation	$= max$	$\geq median$	$< median$	Avg
$T11U$	15	47	3	0.485
$T11F$	10	46	4	0.454
Set precision	9	47	3	0.661
Set recall	0	33	17	0.321
TREC batch filtering of topics 151-200				
Evaluation	$= max$	$\geq median$	$< median$	Avg
$T11U$	8	31	19	0.236
$T11F$	12	46	4	0.090
Set precision	21	47	3	0.288
Set recall	0	36	14	0.044

Table 2: Comparative batch filtering results

2.2.2 Routing track

We experiment routing using a similar method then the batch filtering track. The new representing profile obtained from the sample documents is selected as routing profile, and applied on the test documents. The final output is the top 1000 ranked documents for each topic.
Table 3 shows the routing results at average uninterpolated precision for all topics.

TREC Routing for topics 101-150			
$= max$	$\geq median$	$< median$	$AvgP$
13	45	5	0.369
TREC Routing for topics 151-200			
$= max$	$\geq median$	$< median$	$AvgP$
11	42	8	0.004

Table 3: Comparative routing results

3 Conclusion

We described in this paper a learning and threshold updating method for information filtering. Adaptive learning is based on equation system resolution under constraints, a gradient propagation formula uses the system solution to improve the user profile representation. The threshold updating is done independently from learning, it controls perfectly the random variation of rsv values affected to incoming documents.
We have presented our experiments for TREC2002 who are focused on the Filtering (adaptive, batch and routing) tracks.

References

[1] M. Boughanem, C. Chrisment, M. Tmar, *Mercure and MercureFiltre applied for Web and Filtering tasks at TREC-10.* Proceedings of TREC-10, 2001.

[2] M. Boughanem and M.Tmar, *Incremental profile adaptive filtering: profile learning and threshold calibration*, Proceedings of 17th ACM Symposium on Applied Computing (SAC), pages 640-644, 2002.

[3] M. Mostafa, *An algorithm for suffix stripping.* Program, 14(3), pages 130-137, 1980.

[4] G. Saporta, *Probabilits, analyse des donnes et statistiques.* ed. Technip, 1996.

[5] S. Walker, S. E. Robertson, M. Boughanem, G. J. F. Jones, K. Sparck Jones, *Okapi/Keenbow at TREC-6 automatic and ad hoc, VLC, routing, filtering and QSDR.* Proceedings of TREC-6, 1997.