Improved Learning Method for Learning User Profiles and Analysis of the Former Methods

Matematiikan erikoistyö Mat- 1.139

Antti Rajala, 45363m, Ti
23.9.1998
(modified by Eerika Savia 30.9.1998)

Contents

1 Introduction

1.1 Background

This paper was developed as a part of the SmartPush project of HUT TAI Research Centre. We are developing a system which is capable of filtering interesting documents for the user from a feed that contains large amount of irrelevant documents. This is done with information provided by the user in the form of feedback she gives for some documents. From this the user profile is constructed and compared to the documents coming from the feed.

1.2 About This Paper

This paper is based on two formerly published papers of the project by Eerika Savia titled 'Adaptiivinen oppimismenetelmä käyttäjäprofiilien päivittämiseen' [1] and 'Etäisyysmittojen vertailu elektronisessa täsmäjakeluympäristössä' [2]. The former paper presents a learning algorithm which learns the user profile from the feedback given by the user on some set of documents. We are going to see an analysis of this method and then a new perspective on the learning which leads to a new method which has certain desirable attributes. Finally all of these methods are compared to the distance measure described in [2] and some conclusions are drawn.

2 Analysis of the Former Methods

In this chapter we'll see some analysis on the learning methods in [1]. In the first algorithm things are somewhat simplified in a way that only positive feedback is assumed. The improved algorithm presents the problem field as a whole. It gives the user a possibility for providing both negative and positive feedback and the notion of negative profile is introduced.

2.1 Algorithm, the First Version

Suppose that we have n concepts in which the user can be interested. These are numbered and the index set is called $\mathcal{I} = \{1, 2, \ldots, n\}$. We are given the initial user profile and the document on which the user wants to give positive feedback. Both the user profile $\mathbf{p}$ and the document $\mathbf{d}$ are presented as vectors in R^n. The ith element of $\mathbf{d}$ is written as $d_i, i \in \mathcal{I}$, which is the proportion of the concept i in the document. Similarly p_i is the proportion of the user interest in that concept. We are teaching the user profile $\mathbf{p}$ with the document $\mathbf{d}$. The algorithm could be interpreted as a function $(\mathbf{p}, \mathbf{d}) \to \mathbf{p}^\star \in R^n$, where $\mathbf{p}^\star$ is the adjusted profile.

In order to do this, the index set $\mathcal{I}$ is partitioned into two disjoint sets, namely hits and misses: $\mathcal{I}_H = \{i \in \mathcal{I}\colon\ d_i > 0\}$ and $\mathcal{I}_M = \{i \in \mathcal{I} :\ d_i = 0\}$. Then we sum the weights in the profile where the corresponding weights in the document are zero. The total miss weight $W = \sum_{i \in \mathcal{I}_M} p_i$. Suppose further that we are given the learning coefficients $\beta \in (0, 1)$ and $\lambda > 0$. Now the algorithm goes as follows:

Algorithm 2.1 *Depending on the total miss weight W use one of the following :*

$W > 0$*:*

$$\begin{cases} p_i^\star = p_i + \beta\, W\, d_i & i \in \mathcal{I}_H \\[1em] p_i^\star = p_i - \beta\, p_i & i \in \mathcal{I}_M \end{cases} \tag{1}$$

$W = 0$:

$$p_i^\star = \frac{p_i + \lambda\, d_i}{1 + \lambda} \quad i \in \mathcal{I} \tag{2}$$

2.1.1 What Happens?

First of all the index set is divided to those indices where there is weight in the document and those where there is not, $\mathcal{I}_H$ and $\mathcal{I}_M$. The idea is to increase those components of the profile that belong to $\mathcal{I}_H$ (formerly called: hit weights). But as our vector is always normalized so that $\sum_{i\in\mathcal{I}} p_i = 1$ there emerges a need to take that added weight from somewhere, from I_M that is. From (1) we can write:

$$p_i^\star = p_i - \beta\, p_i = (1-\beta)\, p_i \ , \ i \in \mathcal{I}_M$$

And so we gain $\beta\, p_i$ of weight per miss component. Total weight gained can be calculated like this:

$$\textstyle\sum_{i\in\mathcal{I}_M} (p_i - p_i^\star) = \sum_{i\in\mathcal{I}_M} \beta\, p_i = \beta \sum_{i\in\mathcal{I}_M} p_i = \beta\, W$$

Thus the net gain is the fraction β of W. In effect what happens is that we reduce the part of profile which doesn't coincide with the document by the fraction of β, i.e. $\beta\times 100\%$, which is added to the part that do coincide with the document in the proportion of $\mathbf{d}$, i.e.

$$p_i^\star = p_i + \beta\, W\, d_i, i \in \mathcal{I}_H$$

This is what happens when we have some miss weights to decrease from. In the case that we don't have them, i.e. $W = 0$ we apply (2). This can be written as

$$p_i^\star = \tfrac{p_i + \lambda\, d_i}{1+\lambda} = p_i + \tfrac{\lambda}{1+\lambda}\,(d_i - p_i)$$

And in the vector form:

$$\mathbf{p}^\star = \mathbf{p} + \tfrac{\lambda}{1+\lambda}\,(\mathbf{d} - \mathbf{p})$$

Thus as $0 < \frac{\lambda}{1+\lambda} < 1$ we can see that the new profile lies on the line joining $\mathbf{p}$ and $\mathbf{d}$, i.e. if $W = 0$ the algorithm just moves the profile in the direction of the document in the vector space R^n.

2.1.2 Where Does It Converge?

Let us suppose that the user supplied the same document $\mathbf{d}$ all the time and see what happens to the profile. Consider that we are given some initial profile $\mathbf{p}^0$ and construct a sequence $\{\mathbf{p}^k\}$ by applying the learning algorithm to $\mathbf{p}^k$ and $\mathbf{d}$ to get $\mathbf{p}^{k+1}$, $k = 1...\infty$.

Lemma 2.1 *Let us denote the initial profile* $\mathbf{p}^0$ *and the document* $\mathbf{d}$ *and the sum of miss weights by* $W^k = \sum_{i \in \mathcal{I}_M} p_i^k$ *and* $\hat{\mathbf{p}} = \lim_{k\to\infty} \mathbf{p}^k$. *Then the following hold*

1) $W^0 > 0$ *then*

$$\begin{cases} \hat{p}_i = p_i^0 + W^0\, d_i & i \in \mathcal{I}_H \\ \\ \hat{p}_i = 0 & i \in \mathcal{I}_M \end{cases}$$

2) $W^0 = 0$ *then*

$$\hat{\mathbf{p}} = \mathbf{d}$$

Proof See appendix.

2.1.3 Some Problems

As can be verified from (1), if $W > 0$ learning depends linearly on it. So when W becomes small learning slows down considerably until at $W = 0$ the whole method changes abruptly.

If we concentrate on some specific concept i it can be seen that if there is no weight in $\mathbf{d}$ on this specific concept, i.e. $d_i = 0$ then p_i is considered a miss weight. But even a small change in d_i will change the status of p_i. The way how the change in p_i depends on d_i in the neighborhood of $d_i = 0$ is somewhat discontinuous. It would be desirable that the methods used are stable in the sense that small changes in the data affect only small changes in the result.

Also the relation between parameters λ and β should be explored.

2.2 Algorithm, the Second Version (negative feedback)

Sometimes the user might want to tell the system that she greatly dislikes the documents that the system is pushing to her. The above algorithm is not capable of handling this desirable feature; that is – it has to be extended. What is needed is negative feedback. Let us have a look at what was written about this before in [1]. Now – in addition to the information of what the user is interested in, we want to keep track of information of what she is not. The user profile is split into two: a negative and a positive profile.

As the user cannot at the same time like and dislike any subject, there can be weight on some concept either in the negative or the positive profile, but not both. We define both of the profiles as vectors of R^n and denote the negative profile $\mathbf{q}$ and the positive profile $\mathbf{p}$. By vector component p_i we mean the proportion in which the user is interested in concept i compared to the other concepts she likes. Similarly q_i is the proportion of the user dislike in that concept. Now given $\mathbf{p}$ and $\mathbf{q}$ the user rates the document $\mathbf{d}$, that is – gives positive or negative feedback for it. We recall the algorithm from [1]. We review only the positive feedback part of the algorithm here, because in the case of negative feedback we need only change the roles of $\mathbf{p}$ and $\mathbf{q}$.

Again we want to teach the profiles $\mathbf{p}$ and $\mathbf{q}$ in favor of the document $\mathbf{d}$. This can be expressed mathematically as $(\mathbf{p}, \mathbf{q}, \mathbf{d}) \to (\mathbf{p}^\star, \mathbf{q}^\star)$, where $\mathbf{p}^\star$ and $\mathbf{q}^\star$ are the adjusted profiles. Here the situation is more complex than in section 2.1. We need to partition the index set $\mathcal{I}$ further to allow for own hit and miss sets for both the negative and the positive profile.

$$\mathcal{I}_H^P = \{i : d_i > 0 \; and \; q_i = 0\},$$

$$\mathcal{I}_M^P = \{i : d_i = 0 \; or \; q_i > 0\},$$

$$\mathcal{I}_H^Q = \{i : d_i > 0 \; and \; q_i > 0\},$$

$$\mathcal{I}_M^Q = \{i : d_i = 0 \; or \; q_i = 0\}$$

We calculate the total miss weight similarly as in the first algorithm but this time for both of the profiles.

$$W_p = \textstyle\sum_{i \in \mathcal{I}_M^P} p_i \quad W_q = \textstyle\sum_{i \in \mathcal{I}_M^Q} q_i$$

We need also an additional variable

$$q_{min} = \begin{cases} \min_{i \in \mathcal{I}_H^Q} \{q_i\} & \text{if} \quad \mathcal{I}_H^Q \neq \emptyset \\ \\ 0 & \text{else.} \end{cases}$$

With this notation the algorithm is:

Algorithm 2.2 *Depending on the total miss weights W_p and W_q use the right alternative from each category.*

POSITIVE PROFILE

$W_p > 0$ *(if $\mathcal{I}_H^P = \emptyset$, do nothing):*

$$\begin{cases} p_i^\star = p_i + \beta_1 \, W_p \, \frac{d_i}{\sum_{j \in \mathcal{I}_H^P} d_j} & i \in \mathcal{I}_H^P \\ \\ p_i^\star = p_i \, (1 - \beta_1) & i \notin \mathcal{I}_H^P \end{cases} \tag{3}$$

$W_p = 0$*:*

$$\begin{cases} p_i^\star = \frac{1}{1+\lambda_1} \; (p_i + \lambda_1 \, \frac{d_i}{\sum_{j \in \mathcal{I}_H^P} d_j}) & i \in \mathcal{I}_H^P \\ \\ \\ p_i^\star = 0 & i \notin \mathcal{I}_H^P \end{cases} \tag{4}$$

NEGATIVE PROFILE

$W_q > 0$ *(if $\mathcal{I}_H^Q = \emptyset$, do nothing):*

$$\begin{cases} q_i^\star = q_i - \beta_2 \, q_{min} \, \frac{d_i}{\sum_{j \in \; \mathcal{I}_H^Q} d_j} & i \in \mathcal{I}_H^Q \\ \\ q_i^\star = q_i \, (1 + \frac{\beta_2 \, q_{min}}{W_q}) & i \notin \; \mathcal{I}_H^Q \end{cases} \tag{5}$$

$W_q = 0$*:*

$$\begin{cases} q_i^\star = \frac{1}{1-\lambda_2 \, q_{min}} \; (q_i - \lambda_2 \, q_{min} \, \frac{d_i}{\sum_{j \in \mathcal{I}_H^Q} d_j}) & i \in \mathcal{I}_H^Q \\ \\ q_i^\star = 0 & i \notin \mathcal{I}_H^Q \end{cases} \tag{6}$$

2.2.1 What Happens?

First of all, we have now two profiles to cope with. The positive profile is adjusted almost the same way as before. The only difference is that in the place of d_i, we use $\frac{d_i}{\sum_{j \in \mathcal{I}_H^P} d_j}$. The idea is that we divide the indices into two sets, those belonging to the positive and those belonging to the negative profile. We could think that there are two independent processes, updating the positive and the negative profile, acting independently. The fraction above is just the same we would get if we normalized each subset of the document separately. This implies that the adjusting of the positive profile, restricting observation only to its own subset of the indices, is identical to the first algorithm.

When adjusting the negative profile some major differences occur. Because we cannot allow any weight to become over one, this implies quite severe restrictions on the method. The pace of learning is linearly dependent of minimum hit weight q_{min} instead of the total miss weight as is the case with the positive profile. The negative profile is adjusted to the opposite direction compared to the positive profile. The hit weights are decremented and the miss weights are incremented, which is intuitive.

2.2.2 Some Problems

Extending the system this way leads us to several new problems. As was the case earlier the effect of teaching on the positive profile is linearly dependent of the total miss weight W_p if it is greater than zero. The situation is much worse in the adjusting of the negative profile. As noted above the effect depends linearly on the minimum hit weight. In a random situation this could be arbitrarily small, which means that the teaching does not have any effect at all. As will be shown in the examples later, this implies very instable behaviour of the system.

The algorithm as presented above does not allow any weight to change from negative to positive. This can be modified so that if some weight becomes small enough, say smaller than some ϵ, it is drawn to zero. Even then as some weight approaches zero, the process of learning slows down until the weight reaches the critical value and is drawn to zero. It is also be very hard to say when the weight becomes so small that it could be treated as nonexistent. This is because weights are relational properties. If the number of concepts increases also the average weight value decreases. For this reason,

ϵ should not even be constant.

The determination of the other parameters is not straightforward either. There should be a relation between the negative and the positive profile. For this reason the processes of updating these profiles should not be separate, but the parameters should depend on each other in some way. This leads us to explore the idea that there might be some more general idea behind the algorithm, that could combine the negative and the positive profile and bring some reason to the parameters.

3 New Approach

Let us forget everything we have seen this far and consider a different point of view for a moment. What kind of object of the real world should the user profile be capable of modeling?

3.1 Optimal Profile

Assume that it is possible for the user to provide feedback for an infinite number of documents instantly without delay. At first, for simplicity, only positive feedback is considered. We have information about each document's coverage of some given concept i. A reasonable assumption would be that if the user gives positive feedback for a document, she gives the feedback for the concepts in the proportion of the distribution $\mathbf{d}$. Now at some point of time we let the user read and rate some infinite collection of documents. We collect ones that have got positive feedback and call them $\mathcal{D} = \{\mathbf{d}^k\}_{k=1}^{\infty}$. It is further assumed that each concept is equally present in those documents that the user has read. We take the first N documents and for each concept calculate the total positive feedback the user has given for it.

$$\mathbf{x}^N = \sum_{k=1}^{N} \mathbf{d}^k$$

Now we divide this by the sum over all concepts to get the distribution of how much feedback the user has given to any specific concept in proportion to the others.

$$\mathbf{p}^N = \frac{\mathbf{x}^N}{\sum_{i=1}^{n} x_i^N} = \frac{\mathbf{x}^N}{\sum_{i=1}^{n} \sum_{k=1}^{N} d_i^k}$$

$$= \frac{\mathbf{x}^N}{\sum_{k=1}^{N} \sum_{i=1}^{n} d_i^k} = \frac{\mathbf{x}^N}{\sum_{k=1}^{N} 1} = \frac{\mathbf{x}^N}{N}$$

Finally we take the limit as N tends to infinity and we should have a reasonable estimate of the user interest on the concept i. We define the optimal user profile:

Definition 3.1 (Optimal user profile) *Assume that at any point of time the user provides feedback for a collection of documents such that all of the concepts are uniformly present in the collection. From these the still infinite collection* $\mathcal{D} = \{\mathbf{d}^k\}_{k=1}^{\infty}$ *of those documents, which have got positive feedback, is collected. The optimal user profile:*

$$\mathbf{p}_{opt} = \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \mathbf{d}^k$$

Of course this definition is very idealistic, but nevertheless it gives some clue about the way how the profile could be constructed. In the real world enviroment the less the user is distracted by providing feedback the better. In some systems it might not be possible to get any explicit feedback at all but only some information how she behaves. This is called implicit feedback and is beyond the scope of this text. In this context it is assumed that the implicit feedback collected can be handled the same way as explicit.

3.2 Modeling Constant Interest

Let us suppose that at some given time the user has already rated some N documents. At this point we make the assumption that the user interest is constant and not varying with time. It should be obvious that in the general case we cannot hope to catch the true user interest with the limited data in hand. With the assumption of non-changing user, however, we should have a system that approaches the optimal state, the optimal user profile, that is

$$\mathbf{p}^N = \frac{1}{N} \sum_{k=1}^{N} \mathbf{d}^k \tag{7}$$

Now as the user interest is constant it is same when the feedback is collected. In particular, at any point of time we can use the documents that have been rated already. Clearly, as the user rates more documents $\mathbf{p}^N$ tends to the optimal profile. Now we could say that if we can calculate $\mathbf{p}^N$ at any time, we always have the best obtainable profile available. Before stating this, let us consider the assumptions made so far.

3.3 Varying Interest

In realistic system we cannot hope the user interest to stay constant but to vary slowly. So we have to drop the assumption of constancy. To handle this a forgetting function γ is introduced. Suppose $\mathcal{D} = \{\mathbf{d}^k\}_{k=0}^N$ is the collection of documents the user has given positive feedback this far, in the order of emergence. The first index is chosen to be 0 instead of 1, because of computational comfort.

$$\begin{cases} \mathbf{x}^N = \sum_{k=0}^N \gamma(k)\mathbf{d}^{N-k} \\ \\ \mathbf{p}^N = \frac{\mathbf{x}^N}{\sum_{i=1}^n x_i^N} \end{cases}$$

The meaning of function γ is to weight different documents so that preference is given to documents that are rated more recently. This implies that it should be non-increasing and non-negative function defined in the set $\{0, \ldots, N\}$. The effect of γ is that the user gradually forgets the feedback that she has given a long time ago. Choosing γ well we should now have a profile that resembles the user interest quite well. But to compute $\mathbf{p}^N$ we need all of the documents rated so far. This is not feasible in real world implementations.

3.4 Computing User Profile

It is obvious that we cannot compute $\mathbf{p}$ in the most straightforward way. It should be also obvious that there is no other possibility in the general case. But we could choose γ in such a way that $\mathbf{p}^N$ could be computed recursively as a function of some reasonable number of previous profiles and/or documents.

Theorem 3.1 *Let* $\mathbf{x}^N = \sum_{k=0}^N \gamma(k)\mathbf{d}^{N-k}$. *If* γ *is such that* $\gamma(k) = \frac{1}{a^k}$ *where* $a \in R, a \geq 1$ *then* $\mathbf{x}^N = f(\mathbf{x}^{N-1}, \ \mathbf{d}^N) = \frac{\mathbf{x}^{N-1}}{a} + \mathbf{d}^N$

Proof

$$\begin{aligned} \mathbf{x}^N &= \sum_{k=0}^N \gamma(k)\mathbf{d}^{N-k} = \sum_{k=0}^N \frac{1}{a^k} \ \mathbf{d}^{N-k} = \sum_{k=0}^N \frac{1}{a^{N-k}} \, \mathbf{d}^k = \\ &= \sum_{k=0}^{N-1} \frac{1}{a^{N-k}}\mathbf{d}^k + \mathbf{d}^N = \frac{1}{a} \sum_{k=0}^{N-1} \frac{1}{a^{N-1-k}}\mathbf{d}^k + \ \mathbf{d}^N = \\ &= \frac{1}{a} \sum_{k=0}^{N-1} \frac{1}{a^k}\mathbf{d}^{N-1-k} + \mathbf{d}^N = \ \frac{\mathbf{x}^{N-1}}{a} + \mathbf{d}^N \ \square \end{aligned}$$

As Theorem 3.1 shows, if we use exponential forgetting function we can compute profile recursively from only the previous profile and the current document. For practical purposes it sounds reasonable to use that kind of forgetting function ; the parameter a determines the rate of forgetting. One can easily see that if $a = 1$ we get a model of a user with no forgetting.

3.5 More General γ Function

Theorem 3.2 *Let* $\mathbf{x}^N = \sum_{k=0}^{N} \gamma(k)\mathbf{d}^{N-k}$. *If* γ *is such that* $\gamma(k) = \sum_{j=1}^{m} \frac{c_j}{a_j^k}$ *where* $\forall j\ a_j \in R, a_j \geq 1$ *and* $\forall i \neq j\ a_j \neq a_i$ *then there exist such* λ_j *and* α_j *that* $\mathbf{x}^N = \sum_{j=1}^{m} \lambda_j\, \mathbf{x}^{N-j} + \sum_{j=1}^{m} \alpha_j(\lambda_1, ..., \lambda_m)\, \mathbf{d}^{N-m+j}$

Proof See appendix.

So if we use more than one previous profile and document to compute the next profile we can use more general gamma function. We could also estimate some other function quite well using exponential functions as base functions and taking linear combinations of them. This is beyond the scope of this text and in practice might not even be necessary as even exponential function is behaving quite well.

3.6 Amount of Feedback

It can be made possible for the user to give various amounts of feedback for different documents. This is very easy to implement now.

$$\mathbf{x}^N = \sum_{k=0}^{N} \gamma(k)\, \alpha_{N-k}\, \mathbf{d}^{N-k} \tag{8}$$

where $\alpha_k > 0$ is the amount of feedback given to the document $\mathbf{d}^k$. Clearly the effect of α is linear. Especially when implicit feedback is used it might be desirable to map this implicit feedback to explicit of some real valued amount.

3.7 Negative Feedback

The restriction that $\alpha_k > 0$ is nothing fundamental. We can as well remove it and say that when $\alpha_k < 0$ the feedback is negative. This leads to some problems. What happens when some x_i gets below zero? Now we have to introduce also the negative user profile.

3.7.1 Negative Profile

As before we want normalized profile to model the distribution of the user interest into separate concepts. Here we want to tell which concepts are those that the user likes the most and which the least. That is why we separate those concepts that the user likes and those that she dislikes into two distinct sets:

$$\begin{aligned} \mathcal{I}_p^k &= \{i : x_i^k \geq 0\} \\ \mathcal{I}_q^k &= \{i : x_i^k < 0\} \end{aligned} \tag{9}$$

The total feedback $\mathbf{x}$ is as in (8). Now we divide $\mathcal{I}$ for the positive and negative profile according to (9) and normalize them separately.

$$\begin{cases} p_i^N = \frac{max\{x_i^N, 0\}}{\sum_{j \in \mathcal{I}_p^N} x_j^N} & \text{if} \quad \sum_{j \in \mathcal{I}_p^N} x_j^N > 0 \\ \mathbf{p}^N = 0 & \text{if} \quad \sum_{j \in \mathcal{I}_p^N} x_j^N = 0 \quad \text{or} \quad \mathcal{I}_p^N = \emptyset \end{cases} \tag{10}$$

$$\begin{cases} q_i^N = \frac{min\{x_i^N, 0\}}{\sum_{j \in \mathcal{I}_q^N} |x_j|^N} & \text{if} \quad \sum_{j \in \mathcal{I}_q^N} |x_j|^N > 0 \\ \mathbf{q}^N = 0 & \text{if} \quad \mathcal{I}_q^N = \emptyset \end{cases} \tag{11}$$

As can be observed from (10) and (11) there is a possibility that the user gives such feedback that we might lose the positive or the negative profile or even both. This is a question to which we will return in more detail later on.

4 Improved Algorithm

Finally we have all the tools to present the algorithm. We will use exponential forgetting function which is simple and reasonable. Instead of profiles, we will maintain the total feedback $\mathbf{x}$ and the positive and negative profile are computed from this when needed. As will be shown in the next chapter if only positive feedback is allowed, there exists a simple formula for calculating current profile as a function of the previous one, the current document and the total feedback weight, which will be explained in the next chapter. However – if the negative feedback is included, this is not straightforward. The idea of the algorithm is based on Theorem 3.1.

4.1 Algorithm, only positive feedback

We present first the special case, in which only positive feedback is allowed and we have only positive profile.

Algorithm 4.1 *Assume that we are given some initial feedback* $\mathbf{x}^N$.

$$\mathbf{x}^{N+1} = \frac{\mathbf{x}^N}{a} + \alpha_{N+1}\mathbf{d}^{N+1}$$

$$\begin{cases} p_i^N = \frac{max\{x_i^N, 0\}}{\sum_{j \in \mathcal{I}_p^N} x_j^N} & if \quad \sum_{j \in \mathcal{I}_p^N} x_j^N > 0 \\ \\ \mathbf{p}^N = 0 & if \quad \sum_{j \in \mathcal{I}_p^N} x_j^N = 0 \end{cases} \tag{12}$$

4.2 Algorithm, Positive and Negative Feedback

For both positive and negative feedback we need a negative profile $\mathbf{q}$ in addition to the positive profile $\mathbf{p}$.

Algorithm 4.2 *Assume that we are given some initial feedback* $\mathbf{x}^N$.

$$\mathbf{x}^{N+1} = \frac{\mathbf{x}^N}{a} + \alpha_{N+1}\mathbf{d}^{N+1}$$

$$\begin{cases} p_i^N = \frac{max\{x_i^N,0\}}{\sum_{j\in\mathcal{I}_p^N} x_j^N} & if \quad \sum_{j\in\mathcal{I}_p^N} x_j^N > 0 \\ \\ \mathbf{p}^N = 0 & if \quad \sum_{j\in\mathcal{I}_p^N} x_j^N = 0 \quad or \quad \mathcal{I}_p^N = \emptyset \end{cases} \tag{13}$$

$$\begin{cases} q_i^N = \frac{min\{x_i^N,0\}}{\sum_{j\in\mathcal{I}_q^N} |x_j|^N} & if \quad \sum_{j\in\mathcal{I}_q^N} |x_j|^N > 0 \\ \\ \mathbf{q}^N = 0 & if \quad \mathcal{I}_q^N = \emptyset \end{cases} \tag{14}$$

4.3 Forgetting with Time

In section 3.3 the argument of γ function is defined to be the order of appearance of the document, i.e. the user profile is forgotten by rating documents. This need not necessarily be the case. A more reasonable alternative might in some cases be that the user forgets her previous interests as time passes by. Let us divide the collecting of feedback into two separate processes, namely learning and forgetting. Suppose again that we already have some initial state $\mathbf{x}$ from which we calculate the next state using either forgetting or learning.

$$forgetting: \qquad \mathbf{x}^\star = \frac{\mathbf{x}}{a} \tag{15}$$

$$learning: \quad \mathbf{x}^\star = \mathbf{x} + \alpha\mathbf{d} \tag{16}$$

Now forgetting could be applied periodically in the intervals of B and learning every time the user rates some document. Because periodically updating profile is computationally expensive, we can as well use equivalent method. That is, it is the same what the state of the profile is when it is not used. Thus we carry out the updating only when the user needs the updated profile. This is equivalent to the following algorithm. All documents are assumed to have certain time stamp t_k, which is just the time of the rating of the document k.

Algorithm 4.3 *Assume that we are given some initial feedback* $\mathbf{x}^k$*.*

$$\mathbf{x}^{k+1} = \frac{\mathbf{x}^k}{a^{T_{k+1}-T_k}} + \alpha_{k+1}\mathbf{d}^{k+1} \quad \textit{where } T_k = \lfloor \tfrac{t_k}{B} \rfloor$$

the negative and positive profiles are calculated from $\mathbf{x}^k$ *as in (13) and (14).*

This algorithm implies the following forgetting function.

Theorem 4.1 *If we apply algorithm 4.3 N times starting from some* $\mathbf{x}^0$*, then* $\mathbf{x}^N$ *can be written as:*

$$\mathbf{x}^N = \sum_{k=0}^{N} \gamma(T_N - T_k)\, \alpha_k\, \mathbf{d}^k \quad \textit{where } \gamma(T_N - T_k) = \frac{1}{a^{T_N-T_k}}$$

Proof write $D^k = \alpha_k\, \mathbf{d}^k$

$$\begin{aligned}
\mathbf{x}^0 &= D^0 \\
\mathbf{x}^1 &= \frac{\mathbf{x}^0}{a^{T_1-T_0}} + D^1 \\
&= \frac{D^0}{a^{T_1-T_0}} + D^1 \\
\mathbf{x}^2 &= \frac{\mathbf{x}^1}{a^{T_2-T_1}} + D^2 \\
&= \frac{D^0}{a^{T_1-T_0}\, a^{T_2-T_1}} + \frac{D^1}{a^{T_2-T_1}} + D^2 \\
&= \frac{D^0}{a^{T_2-T_0}} + \frac{D^1}{a^{T_2-T_1}} + D^2 \\
&\vdots \\
\mathbf{x}^N &= \sum_{k=0}^{N-1} \frac{D^k}{a^{T_N-T_k}} + D^N \\
&= \sum_{k=0}^{N} \gamma(T_N - T_k)\, D^k \\
&= \sum_{k=0}^{N} \gamma(T_N - T_k)\, \alpha_k\, \mathbf{d}^k \quad \square
\end{aligned}$$

In the light of Theorem 4.1 we have an algorithm that implements exponential forgetting function where argument is the time difference of documents instead of the difference in the order of appearance.

5 Comparison with Previous Methods

5.1 Recursive Formula of Profile

As noted earlier it is possible to derive a simple recursive formula for the profile without the notion of total feedback $\mathbf{x}$ as in chapters 3 and 4. However we need to keep track of the total feedback weight.

$$\begin{aligned} &Total\ positive\ feedback\ weight \quad M_p^k = \textstyle\sum_{i \in \mathcal{I}_p^k} x_i^k \\ &Total\ negative\ feedback\ weight \quad M_q^k = \textstyle\sum_{i \in \mathcal{I}_q^k} |x_i^k| \end{aligned} \tag{17}$$

For a moment we allow only positive feedback and $\mathcal{I}_p^k$ becomes just $\mathcal{I}$. For now the total positive feedback weight will be notated just M^k and we also get a recursive formula for it :

$$\begin{aligned} &M^{k+1} = \textstyle\sum_{i \in \mathcal{I}} x_i^{k+1} = \sum_{i \in \mathcal{I}} (\frac{x_i^k}{a} + \alpha_{k+1}\, d_i^{k+1}) = \\ &\tfrac{M^k}{a} + \alpha_{k+1} \textstyle\sum_{i \in \mathcal{I}} d_i^{k+1} = \frac{M^k}{a} + \alpha_{k+1} \end{aligned} \tag{18}$$

Now given profile $\mathbf{p}^k$ we can derive $\mathbf{p}^{k+1}$:

$$\begin{aligned} &\mathbf{p}^{k+1} = \tfrac{\mathbf{x}^{k+1}}{M^{k+1}} = (\tfrac{\mathbf{x}^k}{a} + \alpha_{k+1}\, \mathbf{d}^{k+1}) \ : (\tfrac{M^k}{a} + \alpha_{k+1}) = \\ &(\tfrac{\mathbf{x}^k}{M^k} + \tfrac{\alpha_{k+1}\, a}{M^k}\, \mathbf{d}^{k+1}) \ : (1 + \tfrac{\alpha_{k+1}\, a}{M^k}) = \tfrac{\mathbf{p}^k + \lambda_{k+1}\, \mathbf{d}^{k+1}}{1 + \lambda_{k+1}} \end{aligned} \tag{19}$$

Where we have made a substitution:

$$\lambda_k = \frac{\alpha_k\, a}{M^{k-1}} \tag{20}$$

That is, if only positive feedback is allowed we arrive to the same method as in algorithm 2.1 and [1] when the total miss weight $W = 0$. Notice that (20) gives somewhat more intuitive meaning to the learning coefficient λ.

5.2 Total Weight

In the old algorithms there was no concept of total weight of feedback. We could only tell the relational weights between the concepts in the profile.

Particularly the total negative weight is very important. Suppose that we have given negative feedback to some set of concepts only once or a hundred times. It is obvious that a difference should be made between these kinds of events. Total negative weight could be used for this. The use of total positive weight is not as intuitive, but one use might be associated with confidence on the profile. If the user has rated only a few documents or has not used her profile for a long time, the total positive weight becomes small. This could also mean that the profile becomes less trustworthy. This could be used with learning. Suppose that we have a profile with high confidence level, that is high total positive weight. Then this profile gives very high recommendation to some document which is turned down by the user. Then the profile should be changed more drastically than in the situation, where system is not at all sure, whether the rating it has given to the document is valid or not.

5.3 Total Weight is Zero?

It is possible for the user to give such feedback that positive or negative total weight, or even both, get zero. This means that we don't have the respective profile to use. This is not any bad property of the model but a more fundamental problem. Suppose that the user has some initial profile and then begins to give continuous negative feedback to every concept in the profile equally. The positive profile gradually disappears. From this kind of feedback, we cannot possibly tell in what she is interested. What we can do is to wait until she gives some reasonable feedback again. And reconstruct the profile from that. In this situation, however, we could probably tell what terms she dislikes the most and make the decisions based on this. As can readily be seen this is the result of irrational user behavior and should not present any problem in practice.

5.4 Examples

Now we are ready to present some graphical examples of the properties of the new algorithm. We will see three examples, where the old algorithms fail to produce reasonable results and the improved algorithm is performing well. To maintain clarity only 5 concepts are used, but for this purpose it will be enough. The examples are represented so that the old algorithm comes first, then the improved algorithm follows. For the sake of convenience, the parameter values are held constant during all experiments.

$$\beta_1 = 0.1 \; \beta_2 = 0.1 \; \lambda_1 = 0.1 \; \lambda_2 = 0.1 \; \epsilon = 0.001$$

$$a = 1.05 \; \alpha = 5$$

5.4.1 Some Negative Weight Near Zero

This example shows that if there is some negative hit weight in the profile which is very small then the learning slows down considerably. The initial document and the profile are :

$$\mathbf{d} = (0, 0, 0.9999, 0, 0.0001)$$

$$\mathbf{p} = (1, 0, 0, 0, 0)$$

$$\mathbf{q} = (0, 0, 0.9, 0.099, 0.001)$$

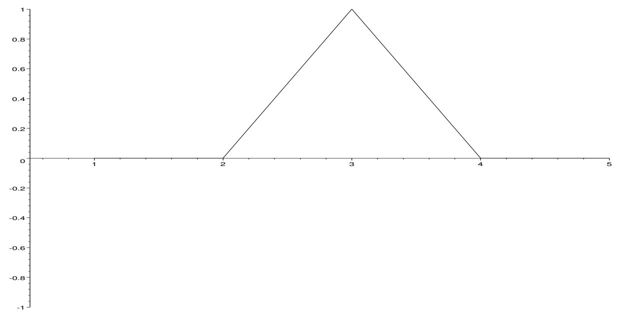

Figure 1: The document 1

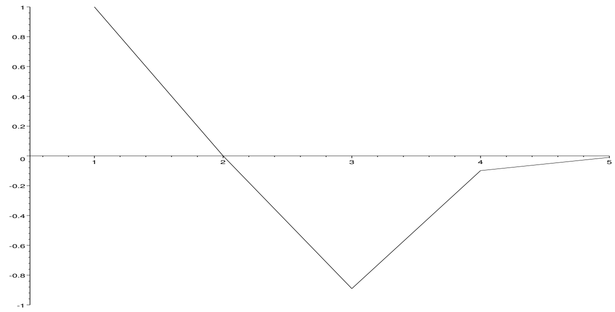

Figure 2: The combined initial profile

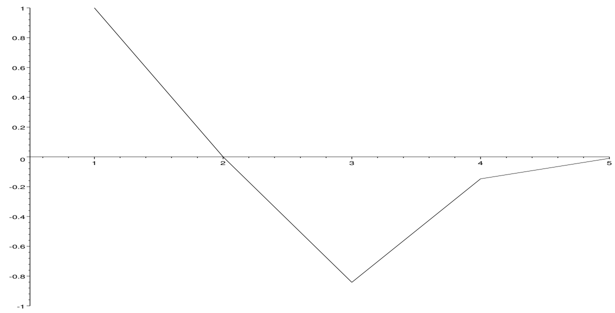

Figure 3: The combined profile after teaching it 50 times with the document

Even if we teach the profile 50 times, very little happens. This is because in the concept 5 we have very little negative weight and there is also a little weight in the document which makes this a hit weight. If we remove the seemingly meaningless weight from the document, the situation changes drastically. We try with a new document:

$$\mathbf{d} = (0, 0, 1, 0, 0)$$

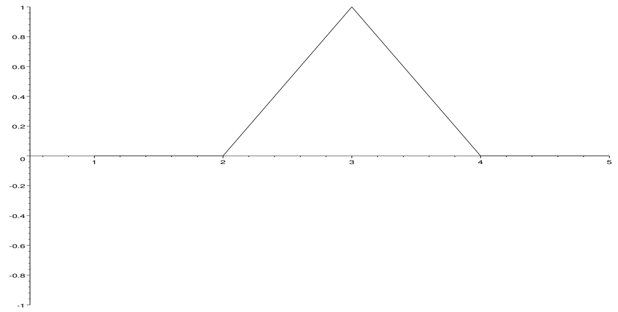

Figure 4: Document 2 with no weight in concept 5

With the new document we teach the same initial profile only 2 times:

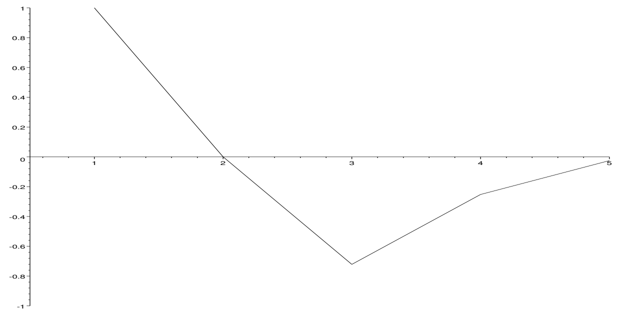

Figure 5: The combined profile after teaching it 2 times with the second document

Now with only two steps we get larger effect than before with 50 steps. The strangest thing is that we changed the document by only a minimal amount in one concept. This is due to the instability of dividing weights into hit and miss weights as stated already in chapter 2.

Now we apply the improved algorithm.

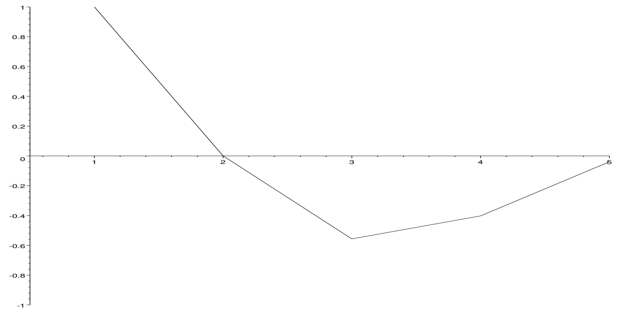

Figure 6: The combined profile after teaching it 5 times with the first document

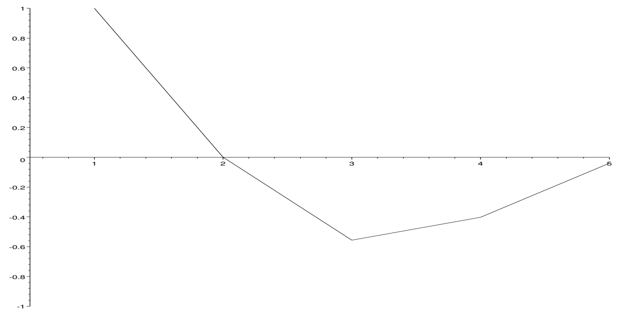

Figure 7: The combined profile after teaching it 5 times with the second document

5.4.2 Crossing the Zero Level

This example illustrates the situation where the user has given negative feedback for some concept. So initially we have negative weight on this concept. Then she changes her mind and begins to give positive feedback on the concept. There should be no problem of skipping from the negative to the positive profile. Let the initial profile and document be :

$$\mathbf{d} = (0, 0, 1, 0, 0)$$
$$\mathbf{p} = (1, 0, 0, 0, 0)$$
$$\mathbf{q} = (0, 0, 0.9, 0.1, 0)$$

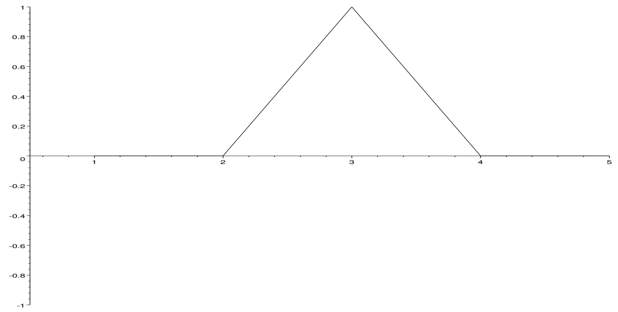

Figure 8: The document

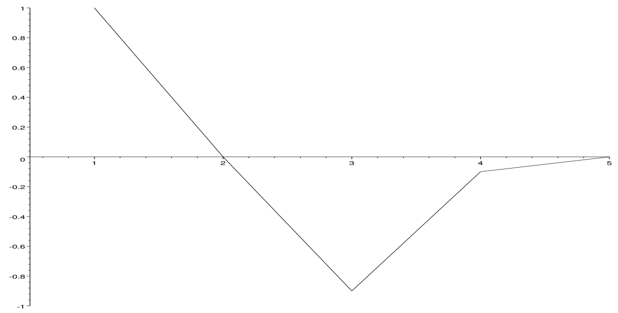

Figure 9: The combined initial profile

Now the initial profile is taught with the old algorithm. Notice that the process slows down when the weight approaches zero, and when it gets over zero it speeds up again.

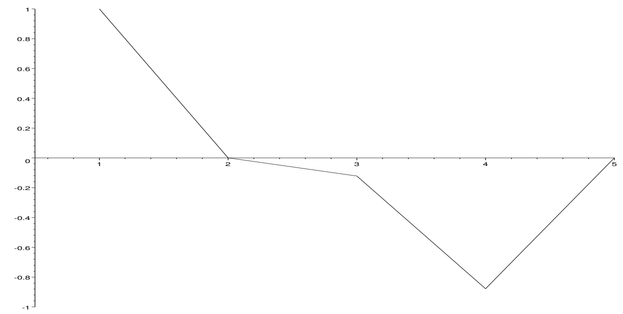

Figure 10: The combined profile after teaching it 20 times

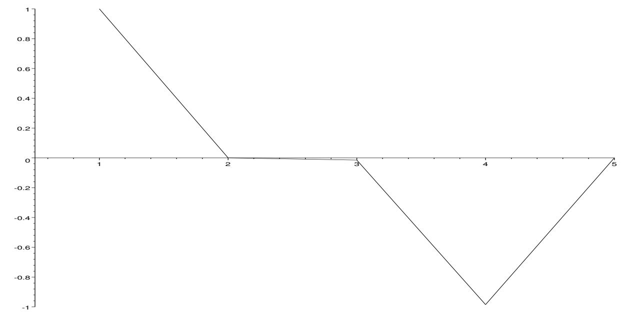

Figure 11: The combined profile after teaching it 40 times

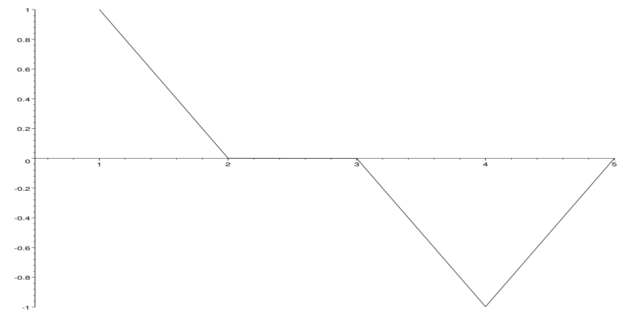

Figure 12: The combined profile after teaching it 60 times

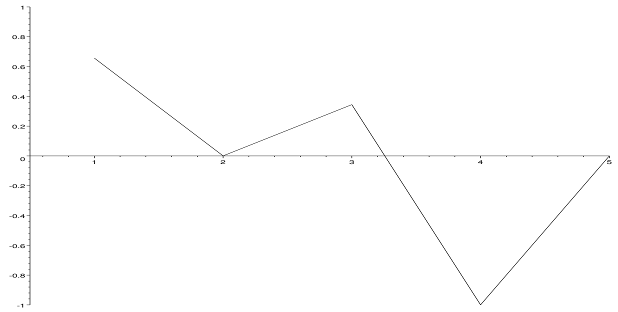

Figure 13: The combined profile after teaching it 70 times

Then the improved algorithm is applied.

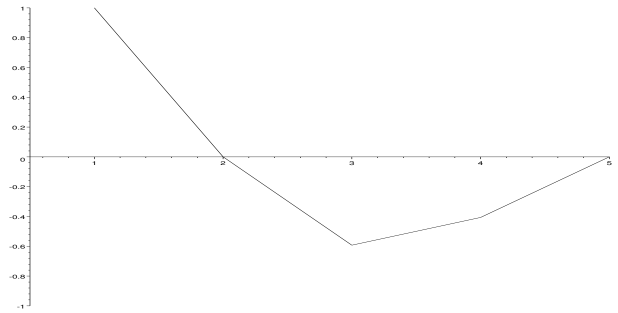

Figure 14: The combined profile after teaching it 5 times

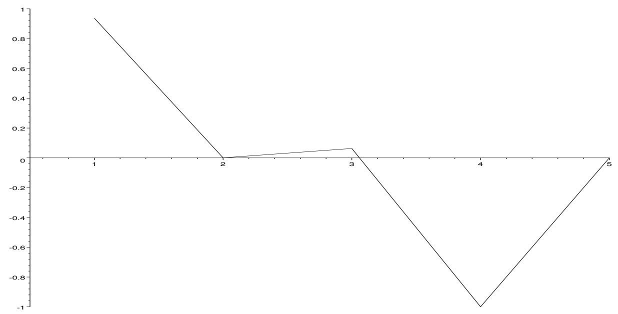

Figure 15: The combined profile after teaching it 6 times

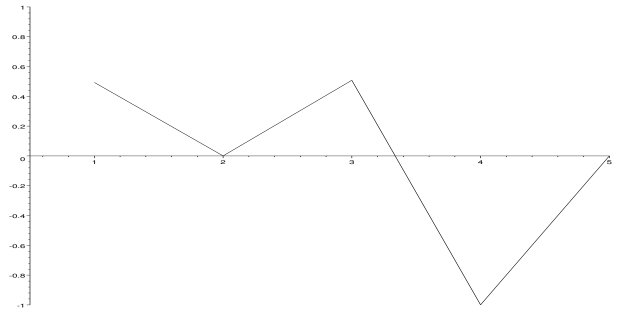

Figure 16: The combined profile after teaching it 10 times

5.5 The total negative weight

The last example will show us a situation, where the user has negative weight on some single concept. Then she decides to give positive feedback for it. But nothing happens. There is no place to put that negative weight! This illustrates the necessity of having additional information about the user dislikes than just the proportions. Let the initial profile and the document be:

$$\mathbf{d} = (1, 0, 0, 0, 0)$$

$$\mathbf{p} = (0, 0, 1, 0, 0)$$

$$\mathbf{q} = (1, 0, 0, 0, 0)$$

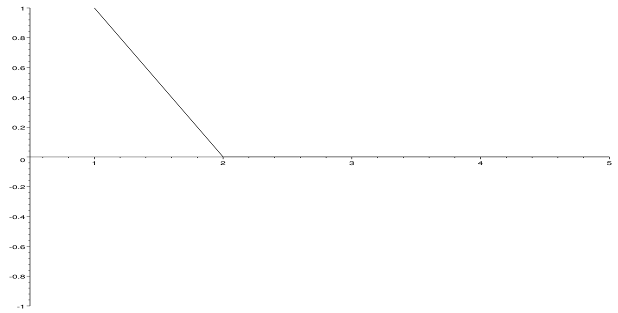

Figure 17: The document

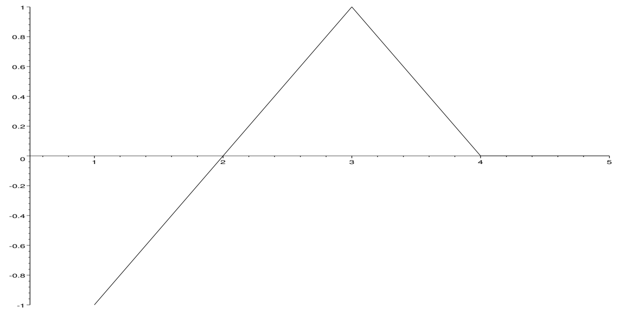

Figure 18: The combined initial profile

As the profile is taught with the old algorithm, nothing happens!

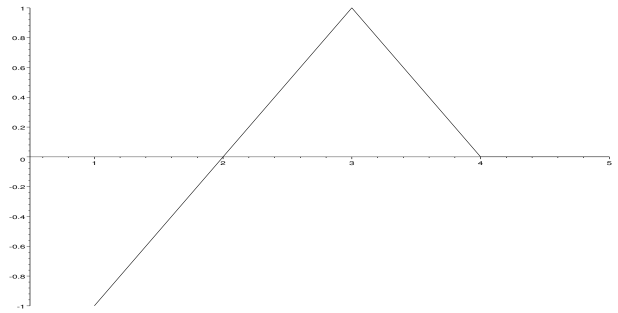

Figure 19: The combined profile after teaching it 100 times

Now we change the algorithm. Notice the small number above the picture. It is the total negative weight of the profile. As positive feedback is given for the negatively weighted concept, proportions stay constant. The total weight decreases until it reaches zero. Then we do not have negative profile at all.

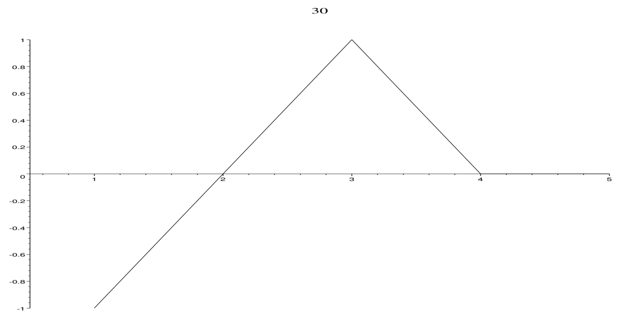

Figure 20: The initial combined profile

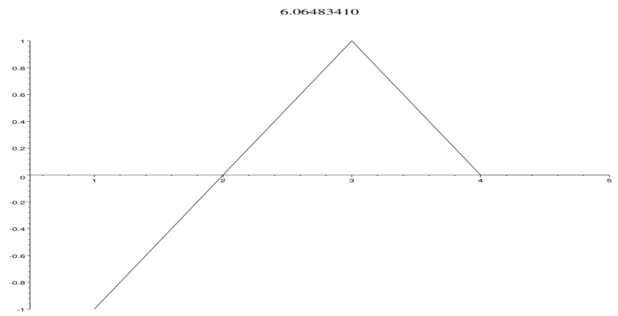

Figure 21: The combined profile after teaching it 5 times

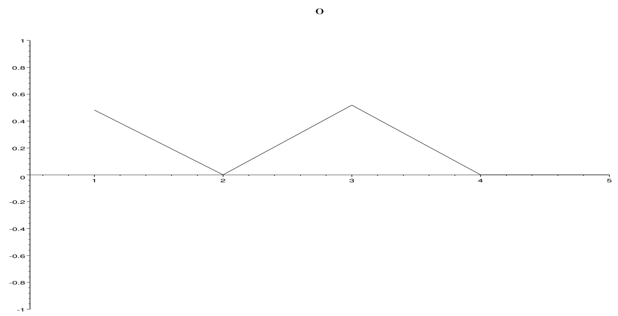

Figure 22: The combined profile after teaching it 10 times

To further show the instability of the old algorithm let us consider a small change to the document again.

$$\mathbf{d} = (0.999, 0, 0, 0, 0.001)$$

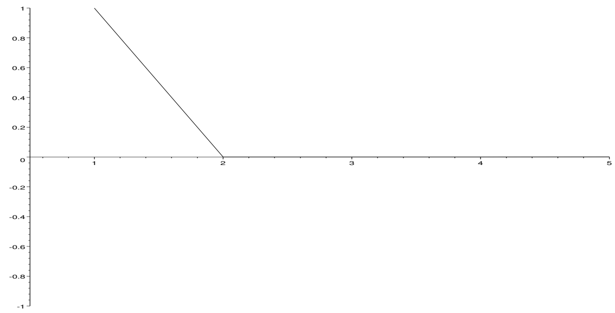

Figure 23: The modified document

Now when we teach the initial profile with this document, the only positive hit weight is in the concept 5. Although the user gives much more positive feedback to the concept 1, the result is that the profile changes in favor of the concept 5.

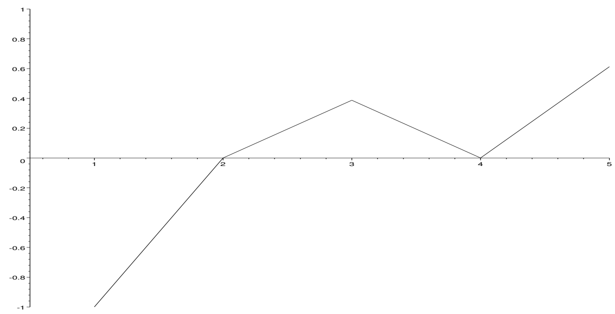

Figure 24: The combined profile after teaching it 10 times with the modified document

The result does not change notably with the improved algorithm.

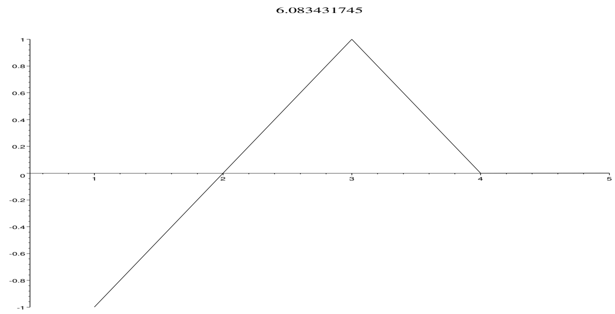

Figure 25: The combined profile after teaching it 5 times with the modified document

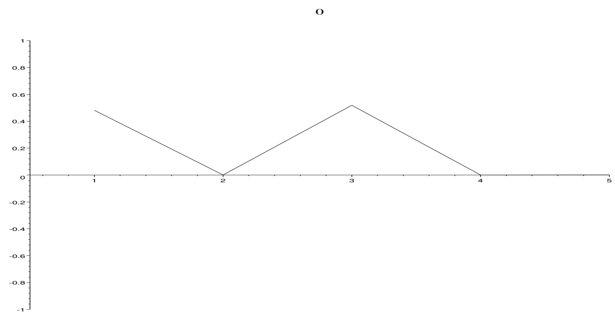

Figure 26: The combined profile after teaching it 10 times with the modified document

6 Comparison with Distance Measure

One possibility to analyze the learning method might be to compare it to the distance measure. In this chapter we have a look at what happens, when we teach profile $\mathbf{p}$ with some document $\mathbf{d}$ and measure the distance between the profile and the document, before and after the learning process. Here we have to restrict the observations to only positive feedback, because so far there is no method to compare documents with negative profile.

6.1 Review of Distance Measure

The distance measure divides the indices into two disjoint sets according to the profile that is measured. Those indices where there is weight in the profile and those where there is not. Please notice that the distinction between a hit weight and a miss weight is different from the one made in chapter 2.

$$\begin{aligned} \mathcal{I}_h &= \{i \in \mathcal{I} : p_i > 0\} \\ \mathcal{I}_m &= \{i \in \mathcal{I} : p_i = 0\} \end{aligned} \tag{21}$$

Coarse measure measures how much there is vain weight in the document whereas fine measure measures how well do the document and profile coincide. Finally these are combined into the combined distance measure.

Coarse measure:

$$D_c(\mathbf{p}, \mathbf{d}) = \sum_{i \in \mathcal{I}_m} d_i \tag{22}$$

Fine measure:

$$D_f(\mathbf{p}, \mathbf{d}) = 1 - \sum_{i \in \mathcal{I}_h} d_i\, p_i = 1 - \sum_{i \in \mathcal{I}} d_i\, p_i \tag{23}$$

Now combined - or the actual - distance measure divides documents to $N+1$ categories according to how well they match the profile in the sense of coarse measure. Then the fine measure ranks these documents inside these categories. (Exception : if the calculated measure exceeds 1, $D(\mathbf{p}, \mathbf{d}) = 1$.)

Distance measure

$$D(\mathbf{p}, \mathbf{d}) = \frac{1}{N} \lfloor\, N\, D_c(\mathbf{p}, \mathbf{d}) \;\; + 0.5 \rfloor + \frac{1}{N}\, D_f(\mathbf{p}, \mathbf{d}) \tag{24}$$

Actually, it might not be wise to separate indices according to whether $p_i = 0$ or not. Because the fact that some document contains only a slight bit of some concept and the user gives positive feedback to this document does not necessarily mean that the user is interested in this concept. And as the user rates more and more documents it is very probable that every weight of the profile will gradually end up non-zero. One solution to this problem could be to define some limit value δ such that if p_i is below δ it means that the user is not interested in that concept. This, however, is beyond the scope of this paper and will not be explored further here.

6.2 Algorithm, First Version vs Distance Measure

Recall the formulae (1) and (2) from the second chapter. We have some initial profile $\mathbf{p}$ and document $\mathbf{d}$ and teach the profile with the document. Let $\mathbf{p}^\star$ be the new profile. Suppose first that the total miss weight $W > 0$. It is clear that the coarse measure $\sum_{i:p_i^\star=0} d_i$ becomes zero. What about the fine measure?

$$\begin{aligned} D_f(\mathbf{p}^\star, \mathbf{d}) &= 1 - \textstyle\sum_{i\in\mathcal{I}} p_i^\star \cdot d_i = 1 - (\sum_{i\in\mathcal{I}} p_i\, d_i + \sum_{i\in\mathcal{I}} \beta\, W\, d_i^2) \\ &= D_f(\mathbf{p}, \mathbf{d}) - \textstyle\sum_{i\in\mathcal{I}} \beta\, W\, d_i^2 \leq D_f(\mathbf{p}, \mathbf{d}) \end{aligned}$$

Both the coarse measure and the fine measure decrease in the process. So also the distance between the profile and the document decreases. Then suppose that $W = 0$. As before it is obvious that the coarse measure becomes zero. To observe what happens with the fine measure, let us write it in the vector form with Euclidean inner product. That is, $D_f(\mathbf{p}, \mathbf{d}) = 1 - (\mathbf{p}, \mathbf{d})$.

$$\begin{aligned} D_f(\mathbf{p}^\star, \mathbf{d}) - D_f(\mathbf{p}, \mathbf{d}) &= (\mathbf{p}, \mathbf{d}) - (\mathbf{p}^\star, \mathbf{d}) = (\mathbf{p}, \mathbf{d}) - (\tfrac{1}{1+\lambda}\,(\mathbf{p} + \lambda \mathbf{d}), \mathbf{d}) \\ &= (\mathbf{p}, \mathbf{d}) - \tfrac{1}{1+\lambda}\,(\mathbf{p}, \mathbf{d}) - \tfrac{\lambda}{1+\lambda}\,(\mathbf{d}, \mathbf{d}) \\ &= \tfrac{\lambda}{1+\lambda}\,((\mathbf{p}, \mathbf{d}) - (\mathbf{d}, \mathbf{d})) \end{aligned}$$

From above we can see that the distance decreases if and only if

$$(\mathbf{p}, \mathbf{d}) \leq (\mathbf{d}, \mathbf{d}) \tag{25}$$

If these vectors were normalized so that $\|\mathbf{p}\|_2 = 1$ this would, according to Cauchy-Schwartz inequality always be true. This is not the case as our profiles and documents are distributions and so lie on the unit sphere of 1-norm. To see when (25) holds let us write it again in component form

$$\sum_{i \in \mathcal{I}} p_i \, d_i \leq \sum_{i \in \mathcal{I}} d_i^2. \tag{26}$$

If we fix $\mathbf{d}$ and change (26) into equality it defines a hyperplane in the subset of R^n where $\mathbf{p}$ lies in. So (26) tells us that the distance will decrease, if we are on the other side of the plane and increase if we are on the other. Let us see what happens if we teach the profile $\mathbf{p}$ twice with $\mathbf{d}$:

$$\begin{aligned}
\mathbf{p}^{\star} &= \tfrac{1}{1+\lambda}(\mathbf{p} + \lambda\,\mathbf{d}) \\
\mathbf{p}^{\star\star} &= \tfrac{1}{1+\lambda}(\mathbf{p}^{\star} + \lambda\,\mathbf{d}) = \tfrac{1}{1+\lambda}(\tfrac{1}{1+\lambda}(\mathbf{p} + \lambda\,\mathbf{d}) + \lambda\,\mathbf{d}) \\
&= \tfrac{1}{(1+\lambda)^2}(\mathbf{p} + \lambda\,\mathbf{d} + (\lambda^2 + \lambda)\,\mathbf{d}) = \tfrac{1}{1+(\lambda^2+2\lambda)}(\mathbf{p} + (\lambda^2 + 2\lambda)\,\mathbf{d}) \\
&= \tfrac{1}{1+\hat{\lambda}}(\mathbf{p} + \hat{\lambda}\,\mathbf{d}) \ \text{ where } \ \hat{\lambda} = \lambda^2 + 2\lambda
\end{aligned} \tag{27}$$

So the effect is the same if we would have taught the profile only once with learning coefficient $\lambda^2 + 2\lambda$. As (25) does not depend on λ, this implies that however many times the profile learns from the document $\mathbf{d}$ the distance never gets below $D_f(\mathbf{d}, \mathbf{d})$. In effect what happens, is that it either monotonically increases or decreases approaching $D_f(\mathbf{d}, \mathbf{d})$ depending on which side of the plane it is originally.

6.3 Improved Algorithm vs Distance Measure

As was shown in chapter 4, accepting only positive feedback, the improved algorithm is essentially the same as the algorithm presented in [1], when $W = 0$. So we can use the results presented above. That is, the coarse measure is made zero, but the fine measure decreases if and only if (25) holds.

6.4 Best Profile to a Given Document

Suppose that we are given some document **d**. For what kind of profile would **d** have the best possibility to be chosen? First of all we have to lower the coarse measure as much as possible. This can made zero by guaranteeing that there is at least a little weight in the profile in places where there is weight in the document. And then optimize the fine measure.

Theorem 6.1 *Let* $i_{max} = \arg\max_{i \in \mathcal{I}} d_i$ *and* M *be the number of indices such that* $d_i > 0$*. If it is demanded* $\forall i : d_i > 0$ *that* $p_i \geq \epsilon$ *then the profile that optimizes (23) is*

$$\begin{cases} p_i = 1 - (M-1)\,\epsilon & i = i_{max} \\ \\ p_i = \epsilon & i \neq i_{max} \end{cases}$$

6.5 What Does This Mean?

It seems that even if the user gives positive feedback to a document, it doesn't necessarily increase the document's chance to be presented to her. Intuitively this sounds like a contradiction. On the other hand, the optimal profile for a given document does not look very good either. In this paper we have adopted a view point of the profile, which is independent of the distance measure, but averages over the feedback given by the user. It could be different. Another approach would be to change the profile to the optimal direction in the sense of the distance measure. That is – to increase the document's possibility to become chosen next time. This, however, complicates the creation of the distance measure. How can we create an objective distance measure if the user profile which we want to measure depends on the measure created? The observations of chapter 3 show us that there is some point to define the user profile as it is defined now. If the distance measure doesn't change the right way, there may be something wrong with the measure.

7 Conclusions

Chapters 2 and 5 showed that algorithm 2.2 has several drawbacks. Let us summarize them here.

1. It is not enough to know the proportion how the user has given negative feedback on concepts. We need to know also the total amount of given feedback.

2. In the previous algorithm positive and negative profiles were handled totally separately. There is no intuitive reason for this to be so, but the weights on concepts should be allowed to move freely from the positive side to the negative, and vice versa.

3. In the previous methods there were many unfixed parameters which had no intuitive meaning.

In chapter 3, a new point of view is introduced. This leads directly to the new algorithm. The theoretical observations of chapter 3 provide some understanding of what kind of object of the real world the profile should try to model and also how it could be modeled. Now also every parameter gets an intuitive meaning. Basically the improved algorithm provides solution to each problem stated above. This was illustrated in chapter 5.

As the user should be distracted as little as possible by requiring explicit feedback the use of implicit feedback becomes necessary. Now it sounds intuitively possible to interpret this implicit feedback as explicit feedback of certain amount. Once a mapping from the user's behaviour to explicit feedback is made, it should be possible to adjust the parameters of the system to allow for this. This demands understanding of the meaning of the parameters.

8 Future Work

There are still several open questions remaining. In this paper the profiles and the documents are represented as vectors. The structure of these representations is actually hierarchical in our case, and these vectors are just the leaf level of the hierarchy tree. It might be reasonable to investigate the effect of learning on other levels of the tree also.

This paper is based only on theoretical inspections. These methods should be measured with real data, which were unfortunately not available at the time of writing of this paper.

References

[1] Savia, Eerika (1997). Adaptiivinen oppimismenetelmä käyttäjäprofiilien päivittämiseen. Matematiikan erikoistyö, TKK. http://smartpush.cs.hut.fi/pubdocs

[2] Savia, Eerika (1998). Etäisyysmittojen vertailu elektronisessa täsmäjakeluympäristössä. Matematiikan erikoistyö, TKK. http://smartpush.cs.hut.fi/pubdocs

9 Appendix

Lemma 2.1 *Let us denote the initial profile* $\mathbf{p}^0$ *and the document* $\mathbf{d}$ *and the sum of miss weights by* $W^k = \sum_{i \in \mathcal{I}_M} p_i^k$ *and* $\hat{\mathbf{p}} = \lim_{k\to\infty} \mathbf{p}^k$*. Then the following hold*
1) $W^0 > 0$ *then*

$$\begin{cases} \hat{p}_i = p_i^0 + W^0\, d_i & i \in \mathcal{I}_H \\ \hat{p}_i = 0 & i \in \mathcal{I}_M \end{cases}$$

2) $W^0 = 0$ *then*

$$\hat{\mathbf{p}} = \mathbf{d}$$

Proof

1) Dependent of only $\mathbf{d}$, $\mathcal{I}_M$ and $\mathcal{I}_H$ are the same at every step. Thus we can observe each set separately.

$i \in \mathcal{I}_M$

$$p_i^1 = (1-\beta)p_i^0$$

$$p_i^2 = (1-\beta)p_i^1 = (1-\beta)^2 p_i^0$$

by trivial induction:

$$p_i^k = (1-\beta)^k p_i^0 \to 0 \text{ as } k \to \infty$$

from this we can calculate also

$$W^k = \sum_{i \in \mathcal{I}_{\mathcal{M}}} p_i^k = \sum_{i \in \mathcal{I}_{\mathcal{M}}} (1-\beta)^k\, p_i^0 = (1-\beta)^k W^0$$

$i \in \mathcal{I}_H$

$$p_i^1 = p_i^0 + \beta\, W^0\, d_i$$

$$p_i^2 = p_i^1 + \beta\, W^1\, d_i = p_i^0 + \beta W^0 d_i + \beta\,(1-\beta)\, W^0\, d_i$$

again by trivial induction:

$$p_i^k = p_i^0 + W^0\,\beta\, d_i\, \textstyle\sum_{j=0}^k (1-\beta)^j \to p_i^0 + W^0\, d_i\, \beta\, \frac{1}{\beta} = p_i^0 + W^0\, d_i \text{ as } k \to \infty$$

2)

$$p_i^k = \frac{p_i^0}{(1+\lambda)^k} + \sum_{j=1}^{k} \frac{\lambda\, d_i}{(1+\lambda)^j}$$

as is seen by induction:

$$p_i^1 = \frac{p_i^0}{1+\lambda} + \frac{\lambda\, d_i}{1+\lambda}$$

$$\begin{aligned} p_i^{k+1} &= \tfrac{p_i^k}{1+\lambda} + \tfrac{\lambda\, d_i}{1+\lambda} \\ &= \tfrac{p_i^0}{(1+\lambda)^{k+1}} + \textstyle\sum_{j=1}^{k} \tfrac{\lambda\, d_i}{(1+\lambda)^{j+1}} + \tfrac{\lambda\, d_i}{1+\lambda} \\ &= \tfrac{p_i^0}{(1+\lambda)^{k+1}} + \textstyle\sum_{j=1}^{k+1} \tfrac{\lambda\, d_i}{(1+\lambda)^j} \end{aligned}$$

Calculate the limit value:

$$\begin{aligned} p_i^k &= \tfrac{p_i^0}{(1+\lambda)^k} + \textstyle\sum_{j=1}^{k} \tfrac{\lambda\, d_i}{(1+\lambda)^j} \\ &\to d_i \, \tfrac{\lambda}{1+\lambda} \tfrac{\lambda+1}{\lambda} = d_i, \text{ as } k \to \infty \quad \square. \end{aligned}$$

Theorem 3.2 *Let* $\mathbf{x}^N = \sum_{k=0}^{N} \gamma(k)\mathbf{d}^{N-k}$*. If* γ *is such that* $\gamma(k) = \sum_{j=1}^{m} \frac{c_j}{a_j^k}$ *where* $\forall j \; a_j \in R, a_j \geq 1$ *and* $\forall i \neq j \; a_j \neq a_i$ *then there exist such* λ_r *and* α_l *that* $\mathbf{x}^N = \sum_{r=1}^{m} \lambda_r \, \mathbf{x}^{N-r} + \sum_{l=1}^{m} \alpha_l(\lambda_1, ..., \lambda_m) \, \mathbf{d}^{N-m+l}$

Proof

$$\mathbf{x}^{N-r} = \sum_{k=0}^{N-r} \sum_{j=1}^{m} \frac{c_j}{a_j^{N-r-k}} \, \mathbf{d}^k$$

For a moment let

$$f = \sum_{l=1}^{m} \alpha_l(\lambda_1, ..., \lambda_m) \, \mathbf{d}^{N-m+l}$$

$$\mathbf{x}^N = \sum_{r=1}^{m} \lambda_r \, \mathbf{x}^{N-r} + f$$

$$\sum_{k=0}^{N} \sum_{j=1}^{m} \frac{c_j}{a_j^{N-k}} \, \mathbf{d}^k = \sum_{r=1}^{m} \lambda_r \sum_{k=0}^{N-r} \sum_{j=1}^{m} \frac{c_j}{a_j^{N-r-k}} \, \mathbf{d}^k + f$$

Both sides are linear combinations of the document vectors. These are same vectors if the coefficients of the document vectors are the same. By observing the documents $0, ..., N-m$, we can fix the λ_r. Coefficients of $\mathbf{d}^k$ from both sides, $0 \leq k \leq N-m$:

$$\begin{aligned}
& \textstyle\sum_{j=1}^{m} \frac{c_j}{a_j^{N-k}} = \sum_{r=1}^{m} \lambda_r \sum_{j=1}^{m} \frac{c_j}{a_j^{N-r-k}} \\
\Leftrightarrow \quad & \textstyle\sum_{j=1}^{m} \frac{c_j}{a_j^{N-k}} = \sum_{j=1}^{m} \sum_{r=1}^{m} \lambda_r \frac{c_j}{a_j^{N-r-k}} \\
\Leftarrow \quad & \textstyle\frac{1}{a_j^{N-k}} = \sum_{r=1}^{m} \frac{\lambda_r}{a_j^{N-r-k}} \\
\Leftrightarrow \quad & \textstyle 1 = \sum_{r=1}^{m} \lambda_r a_j^r
\end{aligned} \tag{28}$$

That is if we can find λ_r such that they solve

$$\begin{pmatrix} a_1 & a_1^2 & \dots & a_1^m \\ a_2 & a_2^2 & \dots & a_2^m \\ \vdots & & \vdots & \\ a_m & a_m^2 & \dots & a_m^m \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_m \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} \tag{29}$$

then the first $N - m$ documents have same coefficients. Indeed there is a unique solution to (29). We show that the columns of the matrix are linearly independent:

$$\begin{aligned} & b_1\, a_i + b_2\, a_i^2 + \ldots + b_m\, a_i^m = 0 && i = 1 \ldots m \\ \Rightarrow\ & a_i\,(b_1 + b_2\, a_i + \ldots + b_m\, a_i^{m-1}) = 0 && i = 1 \ldots m \\ a_i > 0 \Rightarrow\ & b_1 + b_2\, a_i + \ldots + b_m\, a_i^{m-1} = 0 && i = 1 \ldots m \end{aligned}$$

Now we have $m - 1$ -degree polynomial $b_1 + b_2\, x + \ldots + b_m\, x^{m-1}$ which is zero at m places, namely at $a_i \quad i = 1, ..., m$. This implies that it has to be the zero polynomial:

$$b_1 = b_2 = \ldots = b_m = 0 \Rightarrow \quad \text{columns are linearly independent}$$

So – now we can be sure that the coefficients of the first $N-m$ document vectors are the same on both sides of the equation. For the rest of them we need to fix the alpha functions. From (28)

$k = N - m + l \ \ (l = 1 \ldots m)$

$$\sum_{j=1}^{m} \frac{c_j}{a_j^{m-l}} = \sum_{j=1}^{m} \sum_{r=1}^{m-l} \lambda_r \frac{c_j}{a_j^{m-r-l}} + \alpha_l(\lambda_1, \ldots, \lambda_m)$$

$$\Leftrightarrow \quad \alpha_l(\lambda_1, \ldots, \lambda_m) = \sum_{j=1}^{m} \frac{c_j}{a_j^{m-l}} - \sum_{j=1}^{m} \sum_{r=1}^{m-l} \lambda_r \frac{c_j}{a_j^{m-r-l}}$$

Now we have seen that it is possible to find values to the λ_r and α_l so that equality holds in (28). The proof is completed. $\square$

Theorem 6.1 *Let* $i_{max} = \arg\max_{i \in \mathcal{I}} d_i$ *and* M *be the number of indices such that* $d_i > 0$*. If it is demanded* $\forall i : d_i > 0$ *that* $p_i \geq \epsilon$ *then the profile that optimizes (23) is*

$$\begin{cases} p_i = 1 - (M-1)\,\epsilon & i = i_{max} \\ \\ p_i = \epsilon & i \neq i_{max} \end{cases}$$

Proof We want to choose $\mathbf{p}$ so that it maximizes $D = \sum_{i \in \mathcal{I}} p_i \, d_i$.

$$\sum_{i \in \mathcal{I}} p_i = \sum_{i \in \mathcal{I}} d_i = 1$$

$$\mathcal{I}_M = \{i : d_i > 0\} = \{i_1, ..., i_M\}$$

$$p_i \geq \epsilon \; \forall i \in \mathcal{I}_M$$

It is obvious that p_i has to be zero for all i not in $\mathcal{I}_M$.

$$\Longrightarrow \sum_{i \in \mathcal{I}_M} p_i = 1 \Longrightarrow p_{i_1} = 1 - \sum_{i \in \mathcal{I}_M - \{i_1\}} p_i \quad , \quad \text{where } i_1 \in \mathcal{I}_M$$

$$\Longrightarrow D = \sum_{i \in \mathcal{I}_M - \{i_1\}} p_i \, d_i + (1 - \sum_{i \in \mathcal{I}_M - \{i_1\}} p_i) \, d_{i_1}$$

$$\epsilon \leq p_i \leq 1 - (M-1)\epsilon, \sum_{i \in \mathcal{I}_M - \{i_1\}} p_i \leq 1 - \epsilon \tag{30}$$

So we want to choose $p_i, i \in \mathcal{I}_M - \{i_1\}$ so that they maximize D. The domain in (30) is closed and bounded. The boundedness is obvious. It is closed if its complement is open.

Its complement is :

$$p_i > 1-(M-1)\epsilon \quad \text{or} \quad p_i < \epsilon \quad \text{for some } i \quad \text{or} \sum_{i\in\mathcal{I}_M-\{i_1\}} p_i > 1-\epsilon$$

Take $\mathbf{p} \in$ the complement. We have three alternatives.

1. $p_i > 1-(M-1)\epsilon \implies p_i = 1-(M-1)\epsilon+\delta$. Take an $\mathbf{x}$ from the δ neighborhood of $\mathbf{p}$. $\|\mathbf{x}-\mathbf{p}\|_\infty < \delta \implies x_i - p_i > -\delta \implies x_i - (1-(M-1)\epsilon+\delta) > -\delta \implies x_i > 1-(M-1)\epsilon$ $\implies \mathbf{x} \in$ the complement.
2. $p_i < \epsilon \implies p_i = \epsilon - \delta$. Take an $\mathbf{x}$ from the δ neighborhood of $\mathbf{p}$. $\|\mathbf{x}-\mathbf{p}\|_\infty < \delta \implies x_i - p_i < \delta \implies x_i - (\epsilon-\delta) < \delta \implies x_i < \epsilon$ $\implies \mathbf{x} \in$ the complement.
3. $\sum_{i\in\mathcal{I}_M-\{i_1\}} p_i > 1-\epsilon \implies \sum_{i\in\mathcal{I}_M-\{i_1\}} p_i = 1-\epsilon+\delta$. Take an $\mathbf{x}$ from the $\frac{\delta}{M-1}$ neighborhood of $\mathbf{p}$. $\|\mathbf{x}-\mathbf{p}\|_\infty < \frac{\delta}{M-1} \implies$ $x_i - p_i > -\frac{\delta}{M-1}\ \forall i \implies \sum_{i\in\mathcal{I}_M-\{i_1\}} x_i =$ $\sum_{i\in\mathcal{I}_M-\{i_1\}}(x_i-p_i) + \sum_{i\in\mathcal{I}_M-\{i_1\}} p_i =$ $\sum_{i\in\mathcal{I}_M-\{i_1\}}(x_i-p_i) + 1-\epsilon+\delta$ $> -\sum_{i\in\mathcal{I}_M-\{i_1\}} \frac{\delta}{M-1} + 1-\epsilon+\delta = 1-\epsilon$ $\implies \mathbf{x} \in$ the complement.

$\implies$ $\frac{\delta}{M-1}$ neighborhood fits around any point of the complement of the domain in (30) $\implies$ the complement of the domain is open $\implies$ the domain is closed.

The dimension of our space is finite which implies that the domain in (30) is compact. D is obviouly continuous, so the maximum point can be found either on the border or at some critical point of D.

1. $\nabla D = 0 \implies$

$$d_i = d_{i_1}\ \forall i \in \mathcal{I}_M - \{i_1\} \implies D = d_{i_1}\sum_{i\in\mathcal{I}_M} p_i = d_{i_1}$$

2. $p_{i_2} = 1-(M-1)\epsilon \implies$

$$D = (1-(M-1)\epsilon)\, d_{i_2} + \sum_{i\in\mathcal{I}_M-\{i_2\}} \epsilon\, d_i \tag{31}$$

3. $p_{i_2} = \epsilon \implies$

$$D = \epsilon\, d_{i_2} + \sum_{i \in \mathcal{I}_M - \{i_1, i_2\}} p_i\, d_i + \left(1 - \epsilon - \sum_{i \in \mathcal{I}_M - \{i_1, i_2\}} p_i\right) d_{i_1}$$

$$\epsilon \le p_i \le 1 - (M-1)\epsilon, \quad \sum_{i \in \mathcal{I}_M - \{i_1, i_2\}} p_i \le 1 - 2\,\epsilon$$

4.

$$\sum_{i \in \mathcal{I}_M - \{i_1\}} p_i = 1 - \epsilon \implies p_{i_2} = 1 - \epsilon - \sum_{i \in \mathcal{I}_M - \{i_1, i_2\}} p_i \implies$$

$$D = \epsilon\, d_{i_1} + \sum_{i \in \mathcal{I}_M - \{i_1, i_2\}} p_i\ d_i + \left(1 - \epsilon - \sum_{i \in \mathcal{I}_M - \{i_1, i_2\}} p_i\right) d_{i_2}$$

$$\epsilon \le p_i \le 1 - (M-1)\epsilon, \quad \sum_{i \in \mathcal{I}_M - \{i_1, i_2\}} p_i \le 1 - 2\,\epsilon$$

We had four alternatives, where the maximum point might be situated. The critical point exists only if all the document weights are the same, in which case the p_i may be chosen arbitrarily.

If some $p_i = 1 - (M-1)\epsilon$, all the other weights become fixed to ϵ and the process stops.

Otherwise, we are led to a recursive problem. Actually the last two of the alternatives are exactly the same. The first one fixes p_{i_2} and the other p_{i_1} as the numbering i_k is arbitrary, there is no difference.

The intermediate steps, after choosing k of the weights to be ϵ, are of the form:

$$D = \epsilon \sum_{i \in \{i_2, \ldots, i_{k+1}\}} d_i + \sum_{i \in \mathcal{I}_M - \{i_1, \ldots, i_{k+1}\}} p_i\, d_i + \left(1 - k\,\epsilon - \sum_{i \in \mathcal{I}_M - \{i_1, \ldots, i_{k+1}\}} p_i\right) d_{i_1}$$

$$\epsilon \leq p_i \leq 1-(M-1)\epsilon, \quad \sum_{i \in \mathcal{I}_M - \{i_1, \ldots, i_{k+1}\}} p_i \leq 1-(k+1)\,\epsilon$$

$$\nabla D = 0 \implies d_i = d_{i_1} \ \forall i \in \mathcal{I}_M - \{i_1, ..., i_{k+1}\} \tag{32}$$

This can can continued until what is left is:

$$\begin{aligned} D \ &= \epsilon \textstyle\sum_{i \in \{i_2, \ldots, i_M\}} d_i + \textstyle\sum_{i \in \mathcal{I}_M - \{i_1, \ldots, i_M\}} p_i \, d_i + \\ &+ \left(1-(M-1)\,\epsilon - \textstyle\sum_{i \in \mathcal{I}_M - \{i_1, \ldots, i_M\}} p_i\right) \ d_{i_1} \\ &= \epsilon \textstyle\sum_{i \in \{i_2, \ldots, i_M\}} d_i + \ (1-(M-1)\,\epsilon)\, d_{i_1} \end{aligned} \tag{33}$$

At every step we have only three real alternatives (two of them are same). The first one is the critical point. This occurs only if **d** is chosen as in (32). It is easy to verify that in this case, the rest of the p_i can be chosen whatever way to produce same result. So we don't have to worry about this alternative. The following two alternatives sets one of the p_i to ϵ and it means that we take the next step in the recursion. This leads ultimately to (33), where the choise of i_1 is arbitrary. It is also very easy to see with calculation as in (31) that if we choose the fourth alternative, the result is the same as in (33). And as it is of the same form as in the theorem, this completes the proof. □