Exponential Differential Document Count

A Feature Selection Factor for Improving Bayesian Filters Accuracy

***Fidelis Assis*[1], *William Yerazunis*[2], *Christian Siefkes*[3], *Shalendra Chhabra*[2,4]**

1: Empresa Brasileira de Telecomunicações – Embratel, Rio de Janeiro, RJ, Brazil

2: Mitsubishi Electric Research Laboratories, Cambridge MA

3: Database and Information Systems Group, Freie Universität Berlin,
Berlin-Brandenburg Graduate School in Distributed Information Systems

4: Computer Science and Engineering, University of California, Riverside CA

Abstract. This paper introduces the *Exponential Differential Document Count (EDDC)*, an intuitively and empirically derived factor for improving the accuracy of Bayesian filters by automatically detecting and reducing the influence of features with low class separation power.

1 Introduction

Feature extraction and feature selection are key points in text classification tasks and, in particular in spam filtering. To address these points two techniques have been developed in the Open Source project CRM114[1]: *Orthogonal Sparse Bigrams (OSB)* [Siefkes 2004] addresses the first point while the *Exponential Differential Document Count (EDDC)*, object of this paper, addresses the second.

Orthogonal Sparse Bigrams with confidence Factor (OSBF) is a Bayesian filter implemented in CRM114 and in OSBF-Lua[2]. It uses EDDC as a confidence factor for dynamically weighting the features produced by OSB, which are sparse bigrams like the four listed below for the sentence "OSBF is a Bayesian filter":

OSBF is
OSBF <skip> a
OSBF <skip> <skip> Bayesian
OSBF <skip> <skip> <skip> filter

2 How the Bayes' theorem is interpreted in OSBF

The Bayes' theorem allows one to calculate the conditional probability of an event *S* given that another event *F* has happened:

$$P(S|F) = \frac{P(F|S) \times P(S)}{P(F)}$$

[1] http://crm114.sourceforgef.net
[2] http://osbf-lua.luaforge.net

We can interpret *P(S|F)*, also known as the *a posteriori* probability, as the probability of a message being spam given that it contains the feature *F*. In this case, *P(S)* and *P(F)* are the unconditional, or *a priori*, probabilities of the message being spam and of the feature *F* being present in a message, respectively. *P(F|S)* is then the probability of the feature being present in a spam message, also known as the *local* probability.

Let's now see how the different parts of the formula, the *a priori*, *a posteriori* and unconditional probabilities are estimated. In the general case of N classes, the unconditional probability of a document *D* being in class $C_k \in \{C_1, C_2, ..., C_m\}$, where all C_i are disjoint, is estimated as the ratio between the number of documents in class C_k and the sum of the number of documents in each class:

$$P(D \in C_k|\emptyset) = \frac{|C_k|}{\sum_{i=1}^{m} |C_i|}$$

$P(D \in C_k|\emptyset)$ is the estimated probability of *D* being in C_k given that we don't know anything about *D*, that is, the unconditional probability of $D \in C_k$;

$|C_i|$ is the number of documents in class C_i.

For two classes, spam *S* and ham *H*, the above formula reduces to

$$P(D \in S|\emptyset) = \frac{|S|}{|S|+|H|} \quad \text{and} \quad P(D \in H|\emptyset) = \frac{|H|}{|S|+|H|}$$

Because it's not practical to keep a precise count of the number of documents in each class (we'd need a human confirmation for each document), OSBF uses the number of learned messages in each class, that is, every time a training on a class is done by a human, the learning count is updated for that class:

$$P(D \in S|\emptyset) = \frac{L_s}{L_s + L_h} \quad ; \quad P(D \in H|\emptyset) = \frac{L_h}{L_s + L_h}$$

This first estimate can be improved by inspecting the features in the document. For instance, if *D* contains the features $F_1, F_2, ..., F_n$, we could consider the first feature and recalculate the probability applying Bayes, here in its more generic form, to get the probability of D being in class C_k, given that it contains F_1:

$$P(D \in C_k|F1) = P(D \in C_k|\emptyset) \times \frac{P(F_1|C_k)}{\sum_{i=1}^{m} P(F_1|C_i)P(C_i)}$$

$P(D \in C_k|\emptyset)$ is the *a priori* probability;

$P(F_1|C_k)$ is the probability that a document in class C_k will contain feature F_1. As mentioned before, in OSBF this is estimated as the ratio of the number of documents containing the feature F_1 among all learned in the class C_k.

This better estimate, $P(D \in C_k|F_1)$, can be further improved using the next feature, F_2:

$$P(D \in C_k|F_1, F_2) = P(D \in C_k|F_1) \times \frac{P(F_2|C_k)}{\sum_{i=1}^{m} P(F_2|C_i)P(C_i)}$$

Repeating this process for all features we get the general formula for the Bayes chain rule

$$P(D \in C_k|F_1, F_2, ..., F_n) = P(D \in C_k|\emptyset) \times \prod_{j=1}^{n} \frac{P(F_j|C_k)}{\sum_{i=1}^{m} P(F_j|C_i)P(C_i)}$$

3 The Confidence Factor

The motivation for the confidence factor is to reduce the noise introduced by features with small counts and de-emphasize those with low class separation power. This is an attempt to mimic what we do when inspecting a message to tell if it is spam or not. We intuitively consider only a few features which carry strong indications, according to what we've learned and remember, and reduce the importance of those that occur approximately with the same frequency in all classes. So, the confidence factor is used to reduce the influence of the *local* probability of a feature, inversely to its class separation power. Once *P(F|C)* is estimated as above, the adjusted value, *P_a(F|C), is* calculated using the following formula:

$$P_a(F|C) = 0.5 + CF(F) * (P(F|C) - 0.5)$$

Where *CF(F)* is the confidence factor for feature *F*.

4 Empiric deduction of *CF(F)* for two classes, Spam and Ham (Non Spam)

From the previous formula we know that *CF(F)* must be in the interval [0, 1], 0 for minimum, which reduces *P_a(F|C)* to 0.5 for both classes, thus ignoring the feature, and 1 for maximum confidence, when the local probability of the feature is fully considered. This gives us a first guess for *CF(F)* as the difference between the counts of the feature *F* in both classes, because it is zero when the feature occurs equally in both classes. In such case, *F* is useless as a class indicator and its confidence factor is 0. Let's write $D_{f,s}$ and $D_{f,h}$ for the number of documents containing the feature *F* in the classes *Spam* and *Ham*, respectively, and $ND_{f,s}$ and $ND_{f,h}$ for the normalized counts. We can then express our first formula for *CF(F)* as

$$CF(F) = ND_{f,s} - ND_{f,h}$$

This first approach has a big problem because the result is not limited to the interval [0, 1]. It can be negative or greater than 1. We can easily fix this by dividing by the sum of the terms and taking the square of the result as in the next formula

$$CF(F) = \left(\frac{ND_{f,s} - ND_{f,h}}{ND_{f,s} + ND_{f,h}} \right)^2$$

This new formula looks much better but it produces unrealistic high values for *CF(F)* when $ND_{f,s}$ and $ND_{f,h}$ are small. For instance, $CF(F) = 1$ when $ND_{f,s} = 1$ and $ND_{f,h} = 0$, which is clearly unexpected. So, we need to introduce an extra term to reduce the value of *CF(F)* for small counts. A natural candidate for this is the inverse of the sum of the counts:

$$CF(F) = \frac{(ND_{f,s} - ND_{f,h})^2 - \frac{1}{D_{f,s} + D_{f,h}}}{(ND_{f,s} + ND_{f,h})^2}$$

Now, $CF(F) = 0$ when $ND_{f,s} = 1$ and $ND_{f,h} = 0$, which is more reasonable because one single occurrence is not enough to draw any conclusion. Nevertheless, a value just above zero would seem even more reasonable. We can experiment later on with values between 0 and 1 in the numerator of the inverse sum of counts. For simplicity, let's write $N\Delta_f$ for $ND_{f,s} - ND_{f,h}$ and $N\Sigma_f$ for $ND_{f,s} + ND_{f,h}$ and rewrite the formula as

$$CF(F) = \frac{N\Delta_f^2 - \frac{1}{\Sigma_f}}{N\Sigma_f^2}$$

Because the features can have different intrinsic weights - in OSB the bigrams have different weights depending on the distance between the tokens - we need a final adjustment to take the weight into account. We do that by multiplying *CF(F)* by a term which approaches 1 as the weight increases and 0 otherwise. The sum of the counts is also considered as a form of weight because the greater it is the greater the confidence in the feature indication. The new term combines the intrinsic weight W_i and the sum of the counts Σ_f as shown below

$$\frac{W_f \Sigma_f}{1 + W_f \Sigma_f}$$

We then have the basic formula for the Confidence Factor

$$CF(F) = \frac{N\Delta_f^2 - \frac{1}{\Sigma_f}}{N\Sigma_f^2} \times \frac{W_f \Sigma_f}{1 + W_f \Sigma_f}$$

Or the generic form below which uses the constants K_1, K_2 and K_3 for tuning the performance of the filter by adjusting the decay speed of the confidence factor as the difference in counts reduces and the influence of the weights.

$$CF(F) = \left(\frac{N\Delta_f^2 - \frac{K_1}{\Sigma_f}}{N\Sigma_f^2} \right)^{K_2} \times \frac{W_f \Sigma_f}{1 + K_3 W_f \Sigma_f}$$

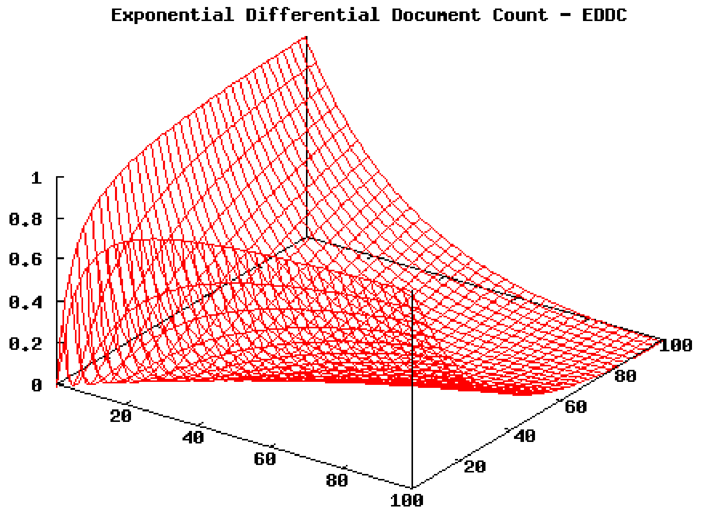
In order to keep greater significance for high-count features with significant but non-conclusive evidence, the product of the respective counts in each class is added to the numerator of the exponentiation base:

$$CF(F) = \left(\frac{N\Delta_f^{\,2} + ND_{f,s} \times ND_{f,h} - \frac{K_1}{\Sigma_f}}{N\Sigma_f^{\,2}} \right)^{K_2} \times \frac{W_f \Sigma_f}{1 + K_3 W_f \Sigma_f}$$

The values for *K1*, *K2* and *K3* which produced the best accuracy for different corpora were 0.25, 10 and 8, respectively. The above formula with the mentioned values for *K1, K2* and *K3*, except for the use of normalized counts only, is what is implemented in OSBF, both in CRM114 and in OSBF-Lua. The first formula, without the addition of the product of the counts, is under tests and the first results on the Full corpus are very good: 1-ROCAC% = 0.016 when K2 is set to 2.

Examining the above formulas and abstracting the details, we see that the confidence factor is basically an exponential of the difference between document counts, so we name it *Exponential Differential Document Count – EDDC.*

Here we have a graphical representation of the EDDC, which visually shows its action on the local probability. The two axes from 0 to 100, indicate the number of occurrences of the feature in each class and the vertical axis, from 0 to 1, the correspondent confidence factor:



4 Results and Discussion

By using the EDDC as a multiplicative factor in determining the per-term weightings in a OSB Bayesian classifier, as in OSBF, we find we can usually gain significant accuracy, when combined with the proper training regimen.

Testing against the NIST TREC 2005 test-set, (a single-pass training set) we find that EDDC improves most of the results (Ham 1% is the error rate for good email when the spam error rate is fixed at 1%; Spam 1% is the error rate for spam when the error rate for good email is fixed at 1%). The table below shows a comparison of the performances of the three OSB Bayesian filters OSB Unique, OSB and OSBF, presented in [Assis 2005]:

Corpus	***Total Messages***	***1-ROCA% OSBU, OSB -> OSBF***	***Ham 1% OSBU, OSB -> OSBF***	***Spam 1% OSBU, OSB -> OSBF***
SB	7.006	0.231, 0.393 -> 0.556	3.48, 3.74 -> 4.00	4.01, 5.62 -> 12.69[*]
MRX	49.086	**0.177, 0.218 ->0.078**	**1.07, 0.63 -> 0.29**	**1.07,** 0.56 -> **0.51**
Full	92.189	**0.042, 0.049 -> 0.019**[†]	0.35, 0.23 -> 0.25	0.21, 0.15 -> 0.19
TM	170.201	**0.195, 0.272 -> 0.120**[‡]	1.58, 1.04 -> 1.17	1.71, 1.07 -> 1.20

(†) Best value, together with IJS2, among all TREC 2005 participants.

(‡) Best value among all TREC 2005 participants.

(*) Although this looks like a large change, the 95% confidence interval on the value "12.69" is actually (3.47 to 37.02) and so this is not a statistically significant loss of accuracy.

Statistically significant changes are indicated in bold; it can be seen that EDDC significantly improves overall accuracy (1-ROCAC%) of spam filtration in most cases.

5 Acknowledgements

We wish to thank Professor Gordon Cormack of the University of Waterloo for the creation of the TREC spam filter testing corpus and his help in running these filters against that corpus.

6 References

[Assis 2005] Fidelis Assis, William S. Yerazunis, Christian Siefkes and Shalendra Chhabra "CRM114 versus Mr. X: CRM114 Notes for the TREC 2005 Spam Track", *The Fourteenth Text REtrieval Conference – TREC SpamTrack 2005*. Download at: http://trec.nist.gov/pubs/trec14/papers/crm.spam.pdf

[Siefkes 2004] Christian Siefkes, Fidelis Assis, Shalendra Chhabra, and William S. Yerazunis "Combining Winnow and Orthogonal Sparse Bigrams for Incremental Spam Filtering", *European Conference on Machine Learning (ECML) / European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, September 2004. Download at: http://page.mi.fu-berlin.de/~siefkes/papers/winnow-spam.pdf