

# ***Exponential Differential Document Count***

*A Feature Selection Factor for Improving Bayesian Filters*

Fidelis Assis<sup>1</sup>

William Yerazunis<sup>2</sup>

Christian Siefkes<sup>3</sup>

Shalendra Chhabra<sup>2,4</sup>

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

*2: Mitsubishi Electric Research Labs- 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*

**CRM114 Team**

---

---

# *What is it?*

The EDDC, aka Confidence Factor, is an intuitively and empirically derived factor for automatically detecting and reducing the influence of features with low class separation power.

# *Motivation*

To mimic what we, humans, do when inspecting a message to tell if it is spam or not:

- Consider mainly the features which carry strong indications;
- Discard or reduce the importance of those which occur approximately equally in both classes.

# How is the Confidence Factor used?

It is used to adjust the *local probability* of a feature towards the don't care value (0.5), in the Bayes formula, inversely to its class separation power:

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

$P_a(F/C)$  is the adjusted value of  $P(F/C)$ , after applying the Confidence Factor:

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

Where  $CF(F)$  is the Confidence Factor of the feature  $F$ .

# How is it calculated?

- Goal: to derive a factor to reduce the influence of features with approximately the same frequency in both classes, *spam* and *ham*;
- First attempt: to use the difference of the normalized counts in both classes because it's zero when the frequencies are equal:

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

$ND_{f,h}$  - Normalized count of ham messages containing the feature  $f$

$ND_{f,s}$  - Normalized count of spam messages containing the feature  $f$

## Problem:

$CF(F)$  must be in the interval  $[0,1]$  because the adjusted *local probability*,  $P_a(F|C)$ , must also be in that interval!

# How is it calculated?

- The previous problem can be fixed with two simple steps:
  - dividing by the sum of the normalized counts: keep  $\leq 1$ ;
  - taking the square to avoid negative values: keep  $\geq 0$ ;

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

Or, in a simplified notation:

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

But we still have problems when  $ND_{f,h}$  and  $ND_{f,s}$  are small:

For  $ND_{f,h} = 1$  and  $ND_{f,s} = 0$ , the confidence factor is 1, completely unrealistic!

# How is it calculated?

- A new term is then subtracted from the numerator to smooth the results for small counts – the inverse of the sum of the *observed* (not normalized) counts in both classes:

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

$D_{f,h}$  - Count of ham messages containing the feature  $f$

$D_{f,s}$  - Count of spam messages containing the feature  $f$

Or, using the simplified notation:  $CF(F) = \frac{N\Delta_f^2 - \frac{1}{\sum_f}}{N\sum_f^2}$

which is the *core* of the EDDC formula.

# OSB Features and Intrinsic Weight

*Orthogonal Sparse Bigrams* (OSB) is a technique for feature generation that produces bigrams like the four below for the sentence “OSBF is a Bayesian filter”:

	<b>Bigram</b>				<b>Distance</b>
<b>OSBF</b>	<b>is</b>				<b>0</b>
<b>OSBF</b>	< skip >	<b>a</b>			<b>1</b>
<b>OSBF</b>	< skip >	< skip >	<b>Bayesian</b>		<b>2</b>
<b>OSBF</b>	< skip >	< skip >	< skip >	<b>filter</b>	<b>3</b>

For a 5-token sliding window the intrinsic weight of the feature, in OSBF, is given experimentally by  $(5-d)^{(5-d)}$  where  $d$  is the number of skipped tokens in the bigram. The closer the tokens, the greater the weight.

# Generic EDDC formula

Finally, adding a term to take into account the intrinsic weight of the feature, we have the generic EDDC formula:

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

$N \Delta_f$  - Difference of the normalized counts, in both classes, of documents containing  $f$

$N \Sigma_f$  - Sum of the normalized counts, in both classes, of documents containing  $f$

$\Sigma_f$  - Sum of the counts, in both classes, of documents containing  $f$

$W_f$  - Intrinsic weight of the feature

$K_1, K_2, K_3$  - Constants empirically defined

## Variant implemented in CRM114- OSBF and in OSBF- Lua

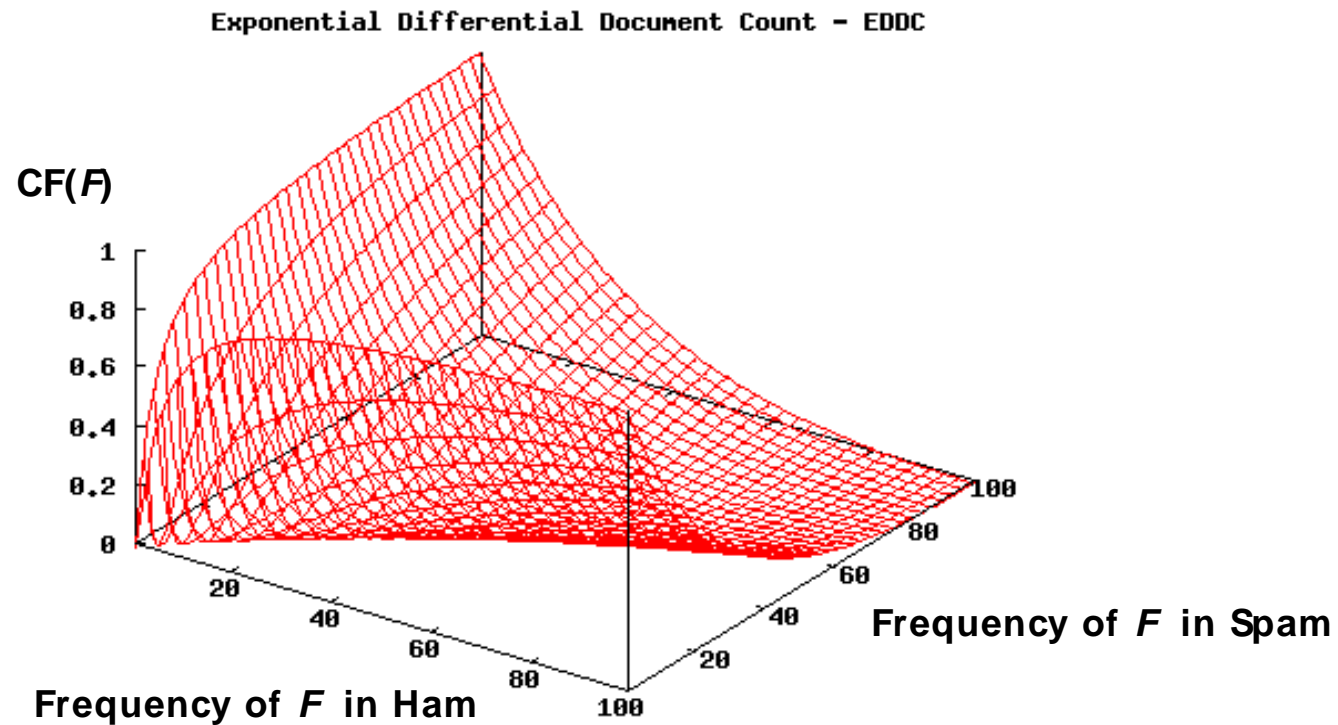
$$CF(F) = \left( \frac{N \Delta_f^2 + ND_{f,s} \times ND_{f,h} - \frac{K_1}{N \Sigma_f}}{N \Sigma_f^2} \right)^{K_2} \times \frac{W_f \times N \Sigma_f}{1 + K_3 \times W_f \times N \Sigma_f}$$

$ND_{f,s}$  - Normalized number of documents containing  $f$  in class spam

$ND_{f,h}$  - Normalized number of documents containing  $f$  in class ham

- Slower decay of the confidence factor as  $N \Delta_f^2$  decreases
- Requires higher  $K_2$  (best experimental value = 10, while only 2 in the original formula)

# Visualization of the Confidence Factor



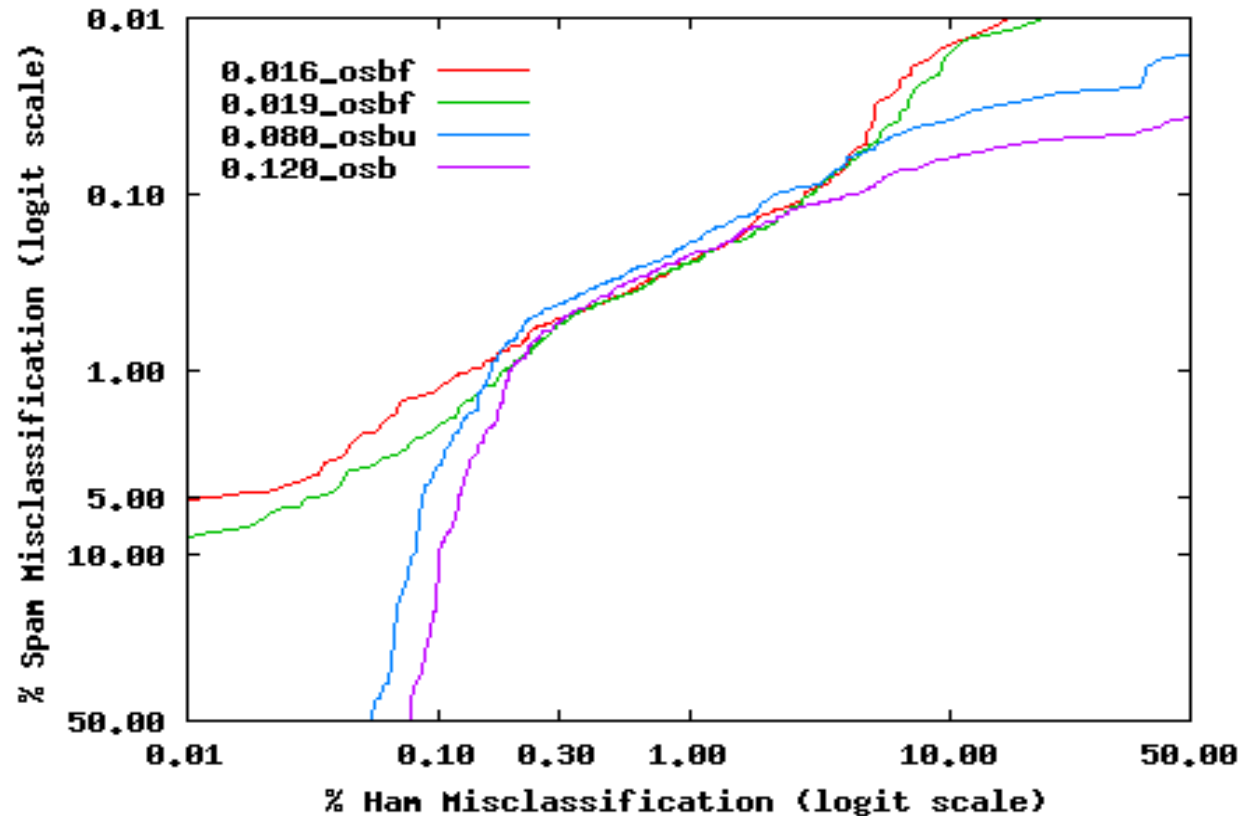
- Function of the difference between the frequencies of the feature in both classes;
- Features with small differences in frequency are strongly de-emphasized.

# ***Classifiers: OSB, OSB Unique and OSBF***

The three classifiers examined in this work have in common that they are all Bayesian and use Orthogonal Sparse Bigrams for feature generation.

- The OSB classifier takes its name from the feature generating technique - it's the standard CRM114 with OSB features;
- OSB Unique is a variant of the OSB classifier that ignores duplicate features;
- OSBF is similar to OSBU Unique but uses EDDC as a Confidence Factor for feature selection.

# *ROC Curves for the Full Corpus*



- The 0.016 OSBF curve was generated using the original formula;
- The 0.019 OSBF with the variant in the previous slide;
- The original formula has been showing better performance as the code improves
  - better pruning, bugs removed.

# Conclusions

- The EDDC factor clearly improved the overall performance of the OSB Bayesian filters presented;
- More importantly, it highly improved the performance in the most critical region for spam filtration, that where ham%  $\leq 0.1$ ;
- It doesn't impact speed – all 3 filters are really fast. Even with the large database used in these tests (2x46 Mbytes), they were able to process the Full corpus (92.189 messages) in 60-70 minutes on an AMD Athlon(TM) XP 2400+.

# ***Thanks!***

The CRM114- OSBF source is available at

<http://crm114.sourceforge.net>

The OSBF- Lua source is available at

<http://osbf-lua.luaforge.net>