CONE : Un algorithme d'optimisation de la F-Mesure par pondération des erreurs de classification

Kévin Bascol[1,2], Rémi Emonet[2], Elisa Fromont[3], Amaury Habrard[2], Guillaume Metzler[2,4], and Marc Sebban[2]

[1]Bluecime inc., France
[2]Univ Lyon, UJM-Saint-Etienne, CNRS, Institut d Optique Graduate School, Laboratoire Hubert Curien UMR 5516, F-42023, Saint-Etienne, France
[3]Univ. Rennes 1, IRISA/Inria, 35042 Rennes cedex, France
[4]Blitz inc., France

14 juin 2018

Résumé

Nous proposons un algorithme d'optimisation de la F-Mesure avec des garanties théoriques utilisable avec toute méthode d'apprentissage par pondération des erreurs. **CONE**, notre algorithme, génère itérativement un ensemble de coûts à partir de l'ensemble d'entraînement de telle sorte que le classifieur final ait une F-Mesure proche de l'optimale. L'optimalité de la F-Mesure obtenue est exprimée à l'aide d'une borne supérieure plus fine que celle présentée dans [Parambath et al. 2014] De plus, nous montrons que les coûts obtenus à chaque itération de **CONE** permettent de réduire drastiquement l'espace de recherche et ainsi de converger rapidement vers les paramètres optimaux. L'efficacité de notre méthode est montrée à la fois en terme de F-Mesure mais aussi de vitesse de convergence sur plusieurs jeux de données déséquilibrés.

Mots-clef : Algorithm, Supervised Learning, Theory.

1 Introduction

The F_β-measure [vR74] is a performance measure used in classification to evaluate the ability of a classifier to predict the labels of new instances with a good recall and a good precision (as an harmonic mean of these two measures). It is the most commonly used measure in imbalanced settings where optimizing the accuracy of the classifier would greatly favor the majority class [CBK09, LFG$^+$13]. The β parameter of the measure controls the relative weights of the precision and the recall. For $\beta < 1$ (resp. $\beta > 1$), more importance is given to the precision (resp. recall), with $\beta = 1$, they are considered equally important. The F_β-measure can be expressed in terms of the true positive rate and true negative rate of the model. These rates are count-based measures which, in addition to its non convexity, makes the F_β-measure unsuitable for direct optimization [NKJ15].

Several methods have been studied to solve the F_β-measure optimization problem. They can roughly be separated into two categories : the ones that optimize a "simpler" surrogate function (e.g., [Jan05, PCM13, DWCH11]) such as a loss based on maximizing the Expected Utility [Jan05], and the ones that learn multiple accurate models with different parameters and keep the model which maximizes the F-measure [MKO$^+$03, BFSDH15, ZEPB13, PUG14, Joa05]. In this second category, the parameters can be costs on the classification errors (cost-sensitive methods) [MKO$^+$03, PUG14] or different classification thresholds for probabilistic models [ZEPB13, Joa05, BFSDH15]. [YCLC12] have shown that both categories of methods give the same results asymptotically and propose heuristics to decide on the category to use depending on the context.

The work presented in this paper falls into the second category and within the cost-sensitive classification-based methods. By taking into account misclassification costs, cost-sensitive learning aims at dealing with problems induced by class imbalanced datasets. We build upon the work presented in [PUG14] which is one of the few recent papers addressing the F-measure optimization from a theoretical point of view (see also [BFSDH15, ZEPB13]). The authors proposed a grid-based approach to find the optimal costs for which a cost-sensitive classifier would give the best F-measure. They proved theoretically that with a sufficiently precise grid, they can approach the optimal F-measure up to a bound that depends only on the optimality of the learned classification model. In this paper, we go much further than [PUG14] from both the algorithmic and

the theoretical points of view. Our contributions are four-fold :

- we give a geometric interpretation of the theoretical guarantees derived in [PUG14]. They can be represented as exempting/unreachable cones in a 2D space where the x-axis gives the value of a parameter t that controls the relative class costs, and the y-axis gives the F-measure of the corresponding cost-sensitive classifier ;
- rather than testing exhaustively the whole grid, we take advantage of those cones to propose an iterative algorithm – called **CONE** – which uses this 2D parameter space to select which class weights t should be tested at every step ;
- we show that, in practice, the proposed bound from [PUG14] is loose and would require a huge number of grid points on t to provide guarantees ;
- we then provide a new tighter theoretical bound by refining the one from [PUG14] and we derive additional constraints allowing our iterative algorithm to drastically prune the search space.

In Section 2, we present our notations and the main results from [PUG14] which are the starting point of our study. Section 3 presents our visual optimality-analysis tool to iteratively choose with our algorithm **CONE**, costs that lead to a near-optimal F-measure. We also derive our tighter bound on the optimal F-measure. Section 4 is devoted to the experiments on benchmark real datasets to prove the efficiency and the effectiveness of the proposed method. We finally conclude in Section 5.

2 Notations and context

Let $\mathbf{X} = (x_1, ..., x_m)$, where $\boldsymbol{x}_i \in \mathbb{R}^n$, be the set of m training instances. We denote by L the number of classes, so that $S = \{(x_i, y_i) \in \mathbb{R}^n \times \{1, ..., L\}\}$ denotes our training sample. Let $\mathcal{H}$ be a family of hypothesis (e.g., linear separators). The following definitions are taken from [PUG14] which is the starting point of our work.

For a given hypothesis $h \in \mathcal{H}$ learned from $\mathbf{X}$, the errors that h makes can be summarized in an error profile defined as $\mathbf{E}(h) \in \mathbb{R}^{2L}$:

$$\mathbf{E}(h) = (\mathtt{FN}_1(h), \mathtt{FP}_1(h), ..., \mathtt{FN}_L(h), \mathtt{FP}_L(h))$$

where $\mathtt{FN}_i(h)$ (resp. $\mathtt{FP}_i(h)$) is the proportion of False Negative (resp. False Positive) that h yields for class i. We then denote as $\mathcal{E}(\mathcal{H})$ the set of all possible error profiles for a given set of hypotheses, and more precisely its closure : $\mathcal{E}(\mathcal{H}) = cl\left(\{\mathbf{E}(h), h \in \mathcal{H}\}\right)$.

Intuitively, an element e is in $\mathcal{E}(\mathcal{H})$ if there exists an hypothesis $h \in \mathcal{H}$ that yields these proportions of false negatives and false positives.

We additionally introduce a function $a : [0, 1] \mapsto \mathbb{R}^L$ which sets the cost of the different types of misclassification. The function a is given below and depends on the final measure that we want to optimize and the considered setting (either binary or multiclass classification). In the following, the F-Measure for any value of β will be denoted by F_β or simply F.

2.1 The binary classification setting

In a binary setting, P is the number of positive instances and N the number of negative examples. We also denote by $\boldsymbol{e}$ the vector (e_1, e_2) where e_1 and e_2 are respectively the number of False Negative examples and the number of False Positive ones.

The F_β-Measure, $F_\beta(\boldsymbol{e})$ is defined by :

$$F_\beta(\boldsymbol{e}) = \frac{(1+\beta^2)(P - e_1)}{(1+\beta^2)P - e_1 + e_2}. \tag{1}$$

and [PUG14] have shown that the function a that assigns the misclassification costs can be defined as : $a(t) = \left(1 + \beta^2 - t, t\right)$ with $t \in [0, 1]$

2.2 The multiclass classification setting

We now consider a classification setting with L classes where P_k, $k = 1, ..., L$ is the number of examples in class k. Given a reference class (taken as class 1), the vector $\boldsymbol{e} = (e_1, ..., e_{2L})$ denotes the proportions of misclassified examples, more precisely e_1 is the proportion of False Positives and e_{2k-1}, $k = 2, ..., L$ is the proportion of False Negatives in class k. Then the multiclass-micro F-Measure, $mcF_\beta(\boldsymbol{e})$ is defined by :

$$mcF_\beta(\boldsymbol{e}) = \frac{(1+\beta^2)(1 - P_1 - \sum_{k=2}^{L} e_{2k-1})}{(1+\beta^2)(1 - P_1) - \sum_{k=2}^{L} e_{2k-1} + e_1}. \tag{2}$$

Moreover, the function a that assigns the misclassification costs is shown by [PUG14] to be defined as, for all $t \in [0, 1]$:

$$a(t) = \begin{cases} 1 + \beta^2 - t & for\ e_{2k-1},\ k = 2, ..., L \\ t & for\ e_1. \end{cases}$$

2.3 Base Results

We recall two main results from [PUG14].

Proposition 1. *[Proposition 4 [PUG14]] Let* $F^\star = \max_{\boldsymbol{e'} \in \mathcal{E}(\mathcal{H})} F(e')$. *We have :* $\boldsymbol{e} \in \operatorname{argmin}_{\boldsymbol{e'} \in \mathcal{E}(\mathcal{H})} \langle a(F^\star), \boldsymbol{e'} \rangle \iff F(\boldsymbol{e}) = F^\star$.

Proposition 1 states that $a(F^\star)$ are the costs that should be assigned to the error profile in order to find the F-optimal classifier in the class of hypothesis $\mathcal{H}$. Hence, maximizing F amounts to minimizing $\langle a(F^\star), \boldsymbol{e'} \rangle$ with respect to $h \in \mathcal{H}$, that amounts to solving a cost-sensitive classification problem.

Proposition 2. *[Proposition 5 [PUG14]] Let $\varepsilon_0 \geq 0$ and $\varepsilon_1 \geq 0$, and assume that there exists $\Phi > 0$ such that for all $\boldsymbol{e}, \boldsymbol{e'}$ satisfying $F(\boldsymbol{e'}) > F(\boldsymbol{e})$, we have :*

$$F(\boldsymbol{e'}) - F(\boldsymbol{e}) \leq \Phi\langle a(F(\boldsymbol{e'})), \boldsymbol{e} - \boldsymbol{e'}\rangle. \tag{3}$$

Then , let us take $\boldsymbol{e}^\star \in argmax\ F(\boldsymbol{e'})$ and denote $\boldsymbol{a}^\star = a(F(\boldsymbol{e}^\star))$. Let furthermore $\boldsymbol{g} \in \mathbb{R}^d_+$ and $h \in \mathcal{H}$ satisfying the following two conditions :

$$(i)\ \|\boldsymbol{g} - \boldsymbol{a}^\star\|_2 \leq \varepsilon_0, \quad (ii)\langle \boldsymbol{g}, \mathbf{E}(h)\rangle \leq \min_{\boldsymbol{e'} \in \mathcal{E}(\mathcal{H})} \langle \boldsymbol{g}, \boldsymbol{e'}\rangle + \varepsilon_1.$$

We have :

$$F(\mathbf{E}(h)) \geq F(\boldsymbol{e}^\star) - \Phi(2\varepsilon_0 M + \varepsilon_1), \quad M = \max_{\boldsymbol{e'} \in \mathcal{E}(\mathcal{H})} \|\boldsymbol{e'}\|_2, \tag{4}$$

where $F(\boldsymbol{e}^\star)$ is the optimal value of the F-Measure.

Proposition 2 states that having near-optimal costs (up to ε_0) is sufficient to have a near-optimal (up to $\Phi(2\varepsilon_0 M + \varepsilon_1)$) F-measure. The value of the constant Φ is given in [PUG14] : it is equal to $\frac{1}{\beta^2 P}$ in the binary case and to $\frac{1}{\beta^2 \sum_{k=1}^{L} P_k}$ in the multiclass context. Leveraging Equation 4 and the Lipschitz-continuity of the function a (with a constant of 2, that we refine in Section 3.3), [PUG14] design a meta-algorithm : learning different cost-sensitive classifiers using a grid on t values, with step $\frac{\varepsilon_0}{2}$, gives a sub-optimality in F-measure that is inferior to $\Phi(2\varepsilon_0 M + \varepsilon_1)$.

In Figure 1 (left), we give an original geometric interpretation of this result in the 2-D space where t is the x-axis and F is the y-axis. In this (t, F) graph, the previous near-optimality result yields an upper cone of values where $F^\star$ cannot be found. Given ε_1, the sub-optimality of the cost-sensitive learning algorithm for the 0/1 loss (i.e., the learning bias), $\Phi\varepsilon_1$ corresponds to the vertical offset of this cone. The symmetric slope of this cone is $\frac{2\Phi M \varepsilon_0}{\varepsilon_0/2} = 4\Phi M$.

Note that this geometric interpretation in the form of cones will play a key role in the rest of this paper.

In terms of t, Equation 4 and the Lipschitz-continuity of a can be combined in the following bound (that we refine in Section 3.2), representing the effective cone :

$$F(\boldsymbol{e}^\star) \leq F(\mathbf{E}(h)) + 4\Phi M \|t - t^\star\|_2 + \Phi\varepsilon_1$$

3 Contributions

In this section, we introduce three contributions : an iterative algorithm for searching the parameter space of a cost-sensitive classifier, a tighter bound to improve this algorithm and an additional constraint that prunes the search space.

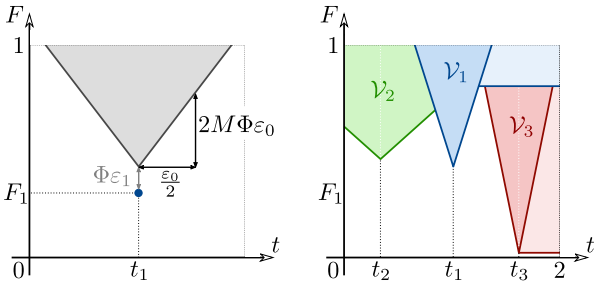

FIGURE 1 – Illustration of the geometric interpretation of the existing bounds on the optimal F-measure (left) and of the proposed **CONE** algorithm after 3 iterations (right). Filled cones (resp. lighter horizontal areas) represent the unreachable area due to the bound from Section 2 and 3.2 (resp. 3.3).

3.1 CONE Algorithm

In Section 2, we gave an interpretation of the bound as a cone of unreachable values in the (t, F) space. We leverage this interpretation to design **CONE** (Cone-based Optimal Next Evaluation), an iterative algorithm that wraps a cost-sensitive classification algorithm (e.g., a weighted SVM). At every iteration i, **CONE** proposes a new value t_i to be used by the cost-sensitive algorithm. **CONE** is described in Algo. 1, illustrated in Fig. 1 (right) and is explained below.

The choice of t_i is based on the area $\mathcal{Z}_{i-1}$ which we define as the union of all cones obtained from previous iterations. t_i is chosen to reduce the maximum value of F for which (t, F) is not in any previous cone. To achieve this goal, **CONE** selects a t which maximizes $F_{max}(t) = \max\{F, (t, F) \notin \mathcal{Z}_{i-1}\}$. The cost sensitive classification algorithm then provides a new value of F_i obtained from cost t_i, which is used to refine the unreachable area as $\mathcal{Z}_i = \mathcal{Z}_{i-1} \cup \mathcal{V}_i$, where $\mathcal{V}_i$ is the cone corresponding to (t_i, F_i). In the case where there are multiple values of t that maximize $F_{max}(t)$ (e.g., at the beginning, or when some range of t values yield $F = 1$), **CONE** selects as t_i the middle of the widest range (see Fig. 1).

From a practical perspective, $\mathcal{Z}_i$ can be represented as a very dense grid (a discretization of $[0, 1 + \beta^2] \times [0, 1]$, the (t, F) space) of binary values or as a set of linear constraints. The stopping criterion *shouldStop* can take different forms including a fixed number of iterations, or rules on the current best F-measure and the current upper bound $max_t\, F_{max}(t)$.

For the **CONE** algorithm to be significantly more efficient than a grid search on t, the cones need to be as spread as possible (the bound as tight as possible). In the next section, we derive a tight bound on the cone slope, which makes the algorithm more efficient as shown in Fig. 2.

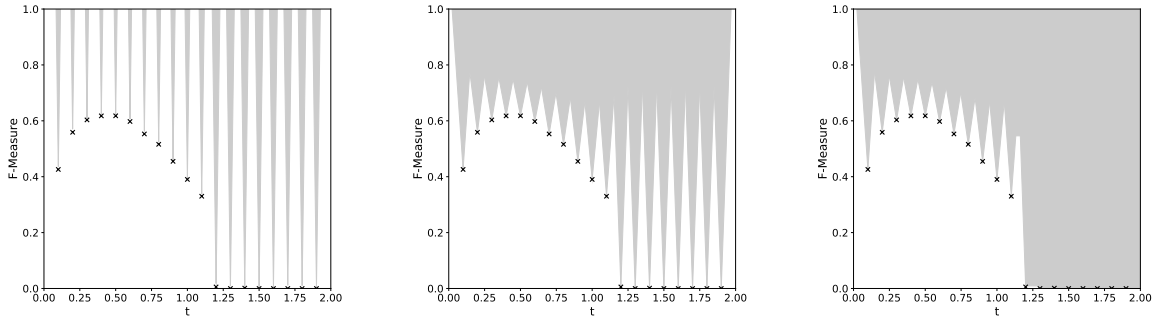

FIGURE 2 – Unreachable region obtained from the same 19 (t_i, F_i) points : on the left, cones from the bound from [PUG14] ; in the middle, cones from our tighter bound presented in Section 3.2 ; on the right, with the pruning constraints from Section 3.3. For an easier comparison, the points are equally spaced and are the same for the three graphs (none are obtained with our iterative algorithm).

Algorithm 1 CONE

Input : β, training set S,
Input : cost-sensitive learning algorithm $wLearn$,
Input : stopping criterion $shouldStop$.
Initialize $i = 0$.
Initialize $\mathcal{Z}_0 = \varnothing$.
repeat
  $i = i + 1$
  $t_i = findNextT(\mathcal{Z}_{i-1})$
  $classifier_i = wLearn(1 + \beta^2 - t_i, t_i)$
  $F_i = F_\beta(classifier_i, S)$
  $\mathcal{V}_i = unreachableZone(t_i, F_i, S)$
  $\mathcal{Z}_i = \mathcal{Z}_{i-1} \cup \mathcal{V}_i$
until $shouldStop(i, classifier_i, \mathcal{Z}_i)$

3.2 A tighter bound for the Optimal F-Score

To increase the spread of the exempting cones (i.e., lower their slope), we propose a tighter bound on the optimal F-Measure than the one given in Proposition 2.

Lemma 1. *Under the assumptions of Proposition 2 with* $\boldsymbol{e'}$ *and* $\boldsymbol{e} = \mathbf{E}(h)$ *the considered vectors of misclassified instances, we have :*

$$\langle \boldsymbol{a}^\star, \boldsymbol{e} \rangle \leq \langle \boldsymbol{a}^\star, \boldsymbol{e}^\star \rangle + \varepsilon_1 + \varepsilon_0(\|\boldsymbol{e}\|_2 + M')$$

with $M' = \max_{\substack{\boldsymbol{e'} \in \mathcal{E}(\mathcal{H}) \\ F_\beta(\boldsymbol{e'}) > F_\beta(\boldsymbol{e})}} \|\boldsymbol{e'}\|_2$

Proof. *We first bound* $\langle \boldsymbol{g}, \boldsymbol{e'} \rangle$ *as follows :*

$$\langle \boldsymbol{g}, \boldsymbol{e'} \rangle = \langle \boldsymbol{g} - \boldsymbol{a}^\star, \boldsymbol{e'} \rangle + \langle \boldsymbol{a}^\star, \boldsymbol{e'} \rangle \leq \langle \boldsymbol{a}^\star, \boldsymbol{e'} \rangle + \varepsilon_0 M',$$

where we have used the Cauchy-Schwarz inequality and condition (i) *of Proposition 2. This implies that :*

$$\min_{\boldsymbol{e'} \in \mathcal{E}(\mathcal{H})} \langle \boldsymbol{g}, \boldsymbol{e'} \rangle \leq \min_{\boldsymbol{e'} \in \mathcal{E}(\mathcal{H})} \langle \boldsymbol{a}^\star, \boldsymbol{e'} \rangle + \varepsilon_0 M' = \langle \boldsymbol{a}^\star, \boldsymbol{e}^\star \rangle + \varepsilon_0 M' \quad (5)$$

Then, by rewriting $\langle \boldsymbol{a}^\star, \boldsymbol{e} \rangle = \langle \boldsymbol{a}^\star - \boldsymbol{g}, \boldsymbol{e} \rangle + \langle \boldsymbol{g}, \boldsymbol{e} \rangle$. *We have the following bound :*

$$\langle \boldsymbol{a}^\star, \boldsymbol{e} \rangle \leq \langle \boldsymbol{g}, \boldsymbol{e} \rangle + \varepsilon_0 \|\boldsymbol{e}\|_2 \leq \langle \boldsymbol{a}^\star, \boldsymbol{e}^\star \rangle + \varepsilon_1 + \varepsilon_0(\|\boldsymbol{e}\|_2 + M').$$

Where we have successively applied the Cauchy Schwarz inequality and condition (i) *of Proposition 2 to obtain the first inequality. Then, for the second one we applied condition* (ii) *of Prop 2 (recall that* $\boldsymbol{e} = \mathbf{E}(h)$*) and Inequality (5) above.* □

Now, we would like to give an explicit value for M' that can be obtained by solving the following optimization problem :

$$\max_{\boldsymbol{e'} \in \mathcal{E}(\mathcal{H})} \|\boldsymbol{e'}\|_2, \qquad s.t.\ F_\beta(\boldsymbol{e'}) > F_\beta(\boldsymbol{e}).$$

In the binary case, setting $\boldsymbol{e} = (e_1, e_2)$ and $\boldsymbol{e'} = (e'_1, e'_2)$ (see Sec. 2). We can write $F_\beta(\boldsymbol{e'}) > F_\beta(\boldsymbol{e})$ as :

$$\frac{(1+\beta^2)(P - e'_1)}{(1+\beta^2)P - e'_1 + e'_2} > \frac{(1+\beta^2)(P - e_1)}{(1+\beta^2)P - e_1 + e_2},$$

We can then simplify this expression and have :

$$P[(e_2 - e_1) - (e'_2 - e'_1)] + (1+\beta^2)P(e_1 - e'_1) > e'_1 e_2 - e_1 e'_2.$$

Now, we set : $e'_1 = e_1 + \alpha_1$ and $e'_2 = e_2 + \alpha_2$.

We are now searching the values α_1, α_2 which are solution of the optimization problem. By replacing the above mentioned quantities in the last inequality we have :

$$\begin{aligned} P(\alpha_1 - \alpha_2) - (1+\beta^2)P\alpha_1 &> e_2\alpha_1 - e_1\alpha_2, \\ \Leftrightarrow \alpha_1(\beta^2 P + e_2) &< -\alpha_2(P - e_1). \end{aligned}$$

Thus we have the following condition on α_1 :

$$\alpha_1 < \frac{-\alpha_2(P - e_1)}{\beta^2 P + e_2}.$$

Remember that $\alpha_1 \in [-e_1, P - e_1]$ and $\alpha_2 \in [-e_2, N - e_2]$, where N (resp. P) is the proportion of

negative (resp. positive) instances. Moreover $\|e'\|_2^2 = (e_1+\alpha_1)^2 + (e_2+\alpha_2)^2$. In other words, we have to maximize a convex function under a linear constraint, we shall then study what happen at the limit of the set defined by the constraint, i.e., when :

$$\alpha_1 = \frac{-\alpha_2(P-e_1)}{\beta^2 P + e_2}. \quad (6)$$

We are looking for the set of values of α_2 such that α_1 belongs to $[-e_1, P-e_1]$ using Equation (6). We get :

$$\left[-(\beta^2 P + e_2), e_1 \frac{\beta^2 P + e_2}{P - e_1}\right].$$

Thus, the set of admissible values for α_2 is

$$\left[-e_2, \min\left(e_1 \frac{\beta^2 P + e_2}{P - e_1}, N - e_2\right)\right].$$

The norm $\|e'\|_2^2$ reaches its maximum at the limit of the predefined set, i.e., when $\alpha_2 = -e_2$ or $\alpha_2 = \min\left(e_1 \frac{\beta^2 P + e_2}{P - e_1}, N - e_2\right)$. We have the corresponding values of α_1 using (6).

Our tighter slope can now be derived from the computation of M', i.e., the value of $\|\boldsymbol{e'}\|_2$, in the following result :

Theorem 1. *Considering the assumptions from Proposition 2, for all $\boldsymbol{e} \in \mathcal{E}(\mathcal{H})$ and all t we have :*

$$F(\boldsymbol{e}^\star) \leq F(\mathbf{E}(h)) + \Phi(\sqrt{2}(\|\boldsymbol{e}\|_2 + M')\|t - t^\star\|_2 + \varepsilon_1).$$

In other words, we refined the slope of the cones to $\sqrt{2}\Phi\left(\|\boldsymbol{e}\|_2 + M'\right)$

Proof. *We simply plug the result of Lemma 1 into the inequality (3) and use $\sqrt{2}$ as the Lipschitz constant of a. Indeed, $\|\frac{\partial a(t)}{\partial t}\|_2 = \|(-1,1)\|_2 \leq \sqrt{2}$. The slope is just the interpretation of this result.* □

In a *multiclass setting*, the Lipschitz constant takes the more general value of $\sqrt{L}$. It is also harder to give an explicit expression of M', but it's value can be obtained by maximizing $\|\boldsymbol{e'}\|_2$, $\boldsymbol{e'} \in \mathcal{E}(\mathcal{H})$ under the following constraints :

$$\begin{aligned} F_\beta(\boldsymbol{e'}) &> F_\beta(\boldsymbol{e}), \\ s.t.\ \alpha_1 &< -\sum_{k=2}^{L} \alpha_k \frac{\beta^2(1-P_1)+e_1}{\sum_{k=2}^{L} e_{2k-1} - (1-P_1)}, \\ \alpha_1 &\in [e_1, P_1 - e_1], \\ \forall n \in \{2,...,L\}\ \alpha_n &\in [-e_{2n-1}, P_n - e_{2n-1}], \end{aligned}$$

where the values α_n are defined as in the binary setting.

In the next section, we will show that the space of research can be pruned and drastically reduced for low values of F-Measure.

3.3 Search space pruning

For the following section, we will only focus on the binary setting. Thus a, e_1 and e_2 will have the same meaning as the one introduced in Section 2. We introduce first the following preliminary result :

Lemma 2. *The difference $(e_1 - e_2)(t)$ is increasing when $\boldsymbol{e}(t)$ is obtained from an optimal classifier trained with costs $a(t)$.*

Proof. *Let $t > t'$, $\boldsymbol{e}(t)$ and $\boldsymbol{e}(t')$ the vector of misclassified examples obtained with an optimal classifier (in the bayes sense) trained with costs $a(t)$ and $a(t')$ respectively. We thus have the following inequalities :*

$$\begin{aligned} t \cdot e_2(t) + (1 + \beta^2 - t)\, e_1(t) \leq \\ t \cdot e_2(t') + (1 + \beta^2 - t)\, e_1(t'), \end{aligned}$$

and

$$\begin{aligned} t' \cdot e_2(t') + (1 + \beta^2 - t')\, e_1(t') \leq \\ t' \cdot e_2(t) + (1 + \beta^2 - t')\, e_1(t). \end{aligned}$$

By multiplying the second equation by -1 and summing the two equations, we get :

$$(t - t')(e_1(t) - e_2(t) \geq (t - t')(e_1(t') - e_2(t')).$$

Thus :

$$e_1(t') - e_2(t') \leq e_1(t) - e_2(t). \quad \square$$

The following result proposes a way to refine the area of research for the optimal weights, i.e., the range of values of t, we will test to optimize the F_β-Measure.

Proposition 3. *Let $t > t'$, $\boldsymbol{e}(t)$ and $\boldsymbol{e}(t')$ the vector of misclassified examples obtained with an optimal classifier trained with costs $a(t)$ and $a(t')$ respectively. We have :*

$$F_\beta(\boldsymbol{e}(t)) \leq (1+\beta^2)\frac{\frac{1+\beta^2}{t'} TP(t')}{\beta^2 \frac{1+\beta^2}{t'} TP(t') + P}. \quad (7)$$

Proof. *If we suppose that we have learned the best classifier in terms of weighted-error we have :*

$$\begin{aligned} e_2(t')t' + (1+\beta^2-t')e_1(t') &\leq (1+\beta^2-t')P, \\ e_2(t')t' - (1+\beta^2-t')TP(t') &\leq 0, \end{aligned}$$

where the first inequality comes from the optimality of the learned classifier compared to the one who classifies all the instances as negative. We can then write :

$$\begin{aligned} e_2(t') &\leq \frac{(1+\beta^2-t')TP(t')}{t'}, \\ e_1(t') - e_2(t') &\geq (P - TP(t') - \frac{1+\beta^2-t'}{t'} TP(t')), \\ &\geq P - \frac{1+\beta^2}{t'} TP(t'). \end{aligned}$$

We now use Lemma 2, so the fact that $e_1 - e_2$ is an increasing function of t to write :

$$P - \frac{1+\beta^2}{t'} TP(t') \leq e_1(t) - e_2(t) \leq P,$$

thus :

$$P - \frac{1+\beta^2}{t'} TP(t') + e_2(t) \leq e_1(t) \leq P + e_2(t).$$

We recall that $e_1(t) \in [0, P]$ and $e_2(t) \in [0, N]$. Moreover, the above inequation shows that we have to search for the best F-Measure we can reach under the constraint : $P - \frac{1+\beta^2}{t'} TP(t') + e_2(t) \leq e_1(t)$. We achieve the best F-Measure by minimizing $e_1(t)$ and $e_2(t)$. The left-hand side is minimized when $e_2(t)$ is equal to 0, then $e_1(t) = P - \frac{1+\beta^2}{t'} TP(t')$. We finally compute the corresponding F-Measure and get, for all $t > t'$:

$$F_\beta(\boldsymbol{e}(t)) \leq (1+\beta^2) \frac{\frac{1+\beta^2}{t'} TP(t')}{\beta^2 \frac{1+\beta^2}{t'} TP(t') + P}. \quad \square$$

This proposition shows that, in the particular case where the F_β-Measure is equal to 0 for a given value of t', it remains equal to 0 for all $t > t'$. This result follows our intuition as t' is the weight assigned to the False Positives : if the optimal learned algorithm (in term of weighted error) classifies all the instances as negative, assigning a higher cost ($t > t'$) to the False Positives (i.e., putting more weight on the negative class) will not give us the possibility to find a positive instance. In the general case, this proposition can be interpreted as a rectangle of unreachable values in the (t, F) space with a vertical offset illustrated in Figure 1 (right). This offset is smaller (and thus more useful) when the algorithm finds only a few True Positives (low recall).

4 Experiments

In this section, we present experiments to illustrate the behavior of our algorithm, its bound and its performance on various datasets.

4.1 Experimental settings

Table 1 describes the datasets we used for our experiments, including their Imbalance Ratio (I.R.). The higher this ratio, the more one should expect that optimizing the classification accuracy is a bad choice in terms of balance between precision and recall. The datasets *IJCNN'01* and *News20* are obtained from LIBSVM [1]. The other ones come from the UCI repository [2].

We reproduce the experimental settings from [PUG14] which we describe here. For datasets with no explicit test set, $\frac{1}{4}$ of the data is kept for testing. The training set is split at random, keeping $\frac{1}{3}$ as the validation set, used to select the hyper-parameters using the F_1-measure. The the penalty constraint of the classifiers (hyper-parameter C) is considered in $\{2^{-6}, 2^{-5}, ..., 2^6\}$. In the experiments $t \in [0, 1+\beta^2]$ in order to take into account the symmetry of the coefficients applied to each class (e.g. in binary both coefficients will be in $[0, 1+\beta^2]$). The maximal number of training iterations is set to 50000. Fitting the intercept of the classifiers is achieved by adding a constant feature with value 1. We report test-time F_1-measure averaged over 5 experiments.

TABLE 1 – Datasets details. The Imbalance Ratio (I.R.) corresponds to the number of negative instances for one positive in a binary dataset, and, in a multi-class dataset, to the number of instances of the largest class for one of the smallest class.

Dataset	Instances	Classes	I.R.	Features
Adult	48842	2	3.19	123
Abalone10	4174	2	5.64	10
IJCNN'01	141691	2	9.39	22
Abalone12	4174	2	15.18	10
Yeast	1484	2	27.48	8
Wine	1599	2	28.79	11
Letter	20000	26	1.32	16
News20	19928	20	1.12	62061

We compared two different cost-sensitive classification algorithms (linear SVM and Logistic Regression implemented in LIBLINEAR) and showed their performance in classification without using any wrapper (standard classification algorithms with meta-parameters tuned on the F-measure), with the wrapper proposed in [PUG14] (results with the * superscript) and with two versions of our **CONE** algorithm, one that does not use the additional pruning constraints defined in 3.3 but can be used in both binary and multiclass settings and one (with the $^+$ superscript) which uses additional pruning constraints but can be used only on binary classification problems. We also compare these methods with a last one, called $_{I.R.}$, which consists of setting the cost of each classes with their representation in the dataset. In other words, the cost c of a False Negative is the proportion of positive examples in the dataset and the cost of False Positive is $1 - c$.

4.2 Analysis of the convergence and the bounds on the F-measure

Figures 3 and 4 illustrate on three (arbitrarily chosen) datasets the behavior of our wrapper (**CONE**) compared to the one proposed in [PUG14] which defines a **grid** to find the best costs. Both methods wrap

1. https ://www.csie.ntu.edu.tw/ cjlin/libsvm/
2. https ://archive.ics.uci.edu/ml/datasets.html

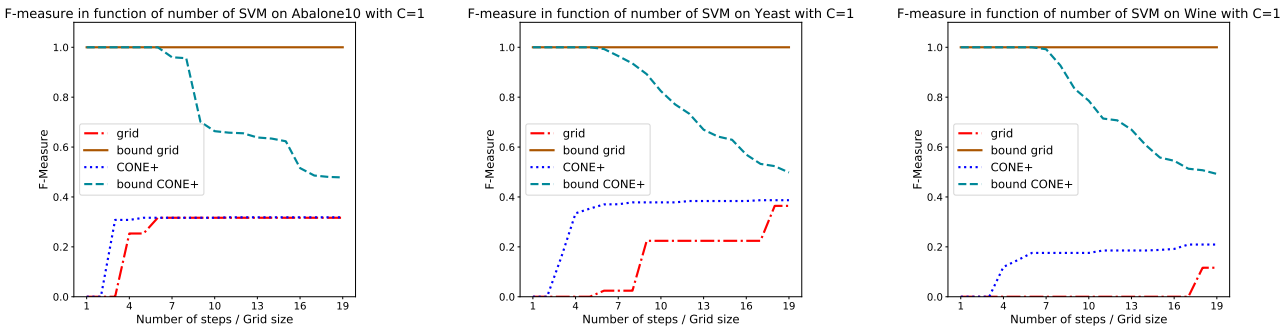

FIGURE 3 – Evolution of the F-Measure of a SVM classifier with $C = 1$ and of the considered bound as a function of the number of t values considered on three datasets.We compare the results of **CONE**$^+$ with the **grid** obtained with the wrapper presented in [PUG14].

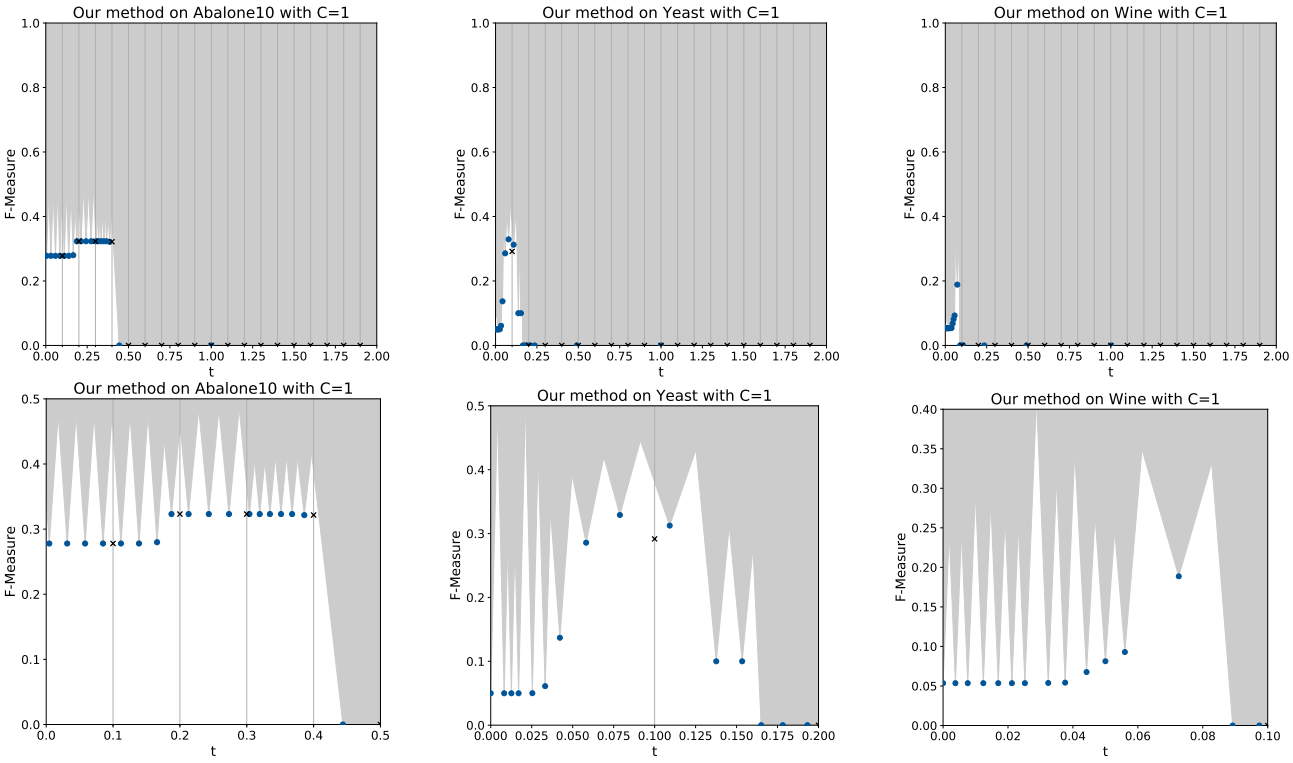

FIGURE 4 – Examples of runs of our method (blue points and shaded area) and of the **grid** wrapper (black crosses) both with a cost-sensitive SVM classifier with $C = 1$. The results are shown at different scales : full (top) and centered around the optimal F-measure (bottom).

here a cost-sensitive SVM classifier with $C = 1$. A step size of 0.1 for the value of t is taken for the **grid** method as suggested in [PUG14].

Figure 3 shows, for both methods (**CONE**$^+$ and **grid**), the evolution of the F-Measure and of the considered bounds (*bound grid* for [PUG14] and *bound* **CONE**$^+$ for us) as a function of the number of t values considered. While **CONE** is incremental (each additional point in the graph needs only to consider one new value of t), the **grid** method needs to know *a priori* the number of grid points it will use. This figure shows two main results : (i) while the bound of [PUG14] is constant, our algorithm **CONE**$^+$ allows us to refine the bound which decreases monotonically after each iteration ; (ii) it is worth noticing that **CONE** reaches its maximum F-measure after learning much fewer models than [PUG14]. This last point is also emphasized by the Table 3 where we show the obtained F-Measure after a limited number of iteration on both methods.

Figure 4 illustrates some runs on the same datasets of both methods in the (t, F) space. The crosses represent the values of the F-Measure obtained using the grid method (recall that they were computed from $t = 0.1$ to $t = 1.9$ in increasing order with a 0.1 step). The points

TABLE 2 – Classification F-Measure for $\beta = 1$ for Logistic Regression and SVM algorithms. Algorithms with a * are reproduced from [PUG14] and the subscript $_{I.R.}$ is used for the classifiers trained with a cost depending on the Imbalance Ratio. The $_C$ subscript indicates the use of our wrapper and the $^+$ superscript indicates the use of the pruning method introduced in 3.3. The presented values are obtained by taking the mean F-Measure over 5 experiments.

Dataset	SVM	$\text{SVM}_{I.R.}$	$\text{SVM}^\star$	SVM_C	SVM_C^+	LR	$\text{LR}_{I.R.}$	$\text{LR}^\star$	LR_C	LR_C^+
Adult	62.6 (0.2)	64.9 (0.3)	66.4 (0.2)	66.4 (0.1)	66.4 (0.1)	63.1 (0.1)	66.0 (0.1)	66.5 (0.1)	66.4 (0.2)	66.4 (0.2)
Abalone10	0.0 (0.0)	30.9 (1.2)	30.2 (2.5)	32.3 (1.3)	32.4 (1.1)	0.0 (0.0)	31.9 (1.4)	31.5 (0.3)	31.2 (1.7)	31.2 (1.5)
IJCNN'01	44.5 (0.4)	53.3 (0.4)	61.6 (0.4)	61.6 (0.6)	61.4 (0.7)	46.2 (0.3)	51.6 (0.3)	58.2 (0.4)	57.9 (0.5)	58.0 (0.4)
Abalone12	0.0 (0.0)	16.9 (2.7)	16.4 (3.6)	17.4 (3.3)	17.7 (4.1)	0.0 (0.0)	18.0 (3.5)	17.6 (2.5)	17.4 (2.7)	18.0 (3.3)
Yeast	0.0 (0.0)	29.3 (2.9)	33.8 (7.9)	32.9 (11.6)	39.0 (7.5)	2.5 (5.0)	29.0 (3.5)	33.0 (11.7)	32.1 (17.9)	31.2 (9.8)
Wine	0.0 (0.0)	15.6 (5.2)	19.8 (7.2)	17.5 (4.3)	24.7 (3.6)	0.0 (0.0)	14.6 (3.2)	22.3 (3.2)	19.4 (6.6)	24.1 (4.5)
Letter	64.4 (0.6)	63.5 (1.5)	69.6 (0.6)	69.7 (0.6)	-	71.8 (0.3)	71.7 (0.3)	71.8 (0.3)	71.8 (0.3)	-
News20	84.0 (0.2)	84.0 (0.3)	84.1 (0.3)	84.0 (0.2)	-	83.4 (0.1)	83.4 (0.1)	83.4 (0.2)	83.4 (0.2)	-

TABLE 3 – Classification F-Measure for $\beta = 1$ when limiting the number of iterations/grid steps to 9, then 4. $\text{SVM}^\star$ are results reproduced from [PUG14]. SVM_C is our wrapper method and SVM_C^+, our wrapper method when using the pruning method introduced in 3.3.

Steps	9			4		
Dataset	$\text{SVM}^\star$	SVM_C	SVM_C^+	$\text{SVM}^\star$	SVM_C	SVM_C^+
Adult	66.1 (0.1)	66.5 (0.1)	66.5 (0.1)	65.8 (0.3)	66.5 (0.03)	66.5 (0.04)
Abalone10	30.2 (2.5)	31.0 (1.1)	32.3 (1.2)	30.7 (2.8)	12.2 (14.5)	30.8 (1.1)
IJCNN'01	61.6 (0.4)	61.0 (0.6)	61.6 (0.6)	61.0 (0.5)	61.0 (0.6)	61.0 (0.6)
Abalone12	16.1 (3.5)	12.2 (7.0)	17.0 (3.5)	0.0 (0.0)	0.0 (0.0)	15.9 (3.7)
Yeast	24.5 (16.3)	34.8 (8.3)	32.3 (12.2)	33.0 (18.0)	14.7 (12.0)	35.0 (8.4)
Wine	11.7 (11.3)	11.3 (10.8)	19.4 (6.6)	0.0 (0.0)	0.0 (0.0)	17.7 (4.4)

are those computed iteratively with **CONE**. The shaded areas correspond to the unreachable regions of the space characterized by our algorithm. The top figures show the results in the full t scale. The most impressive result comes from the fact that the crosses quickly enter the shaded area, which means that the **grid** algorithm is learning models that have no chance to yield an optimal performance. The bottom part of the figure zooms on the white area considered by **CONE**. We can see that only a few crosses are in this area (even only one in the case of Yeast) while all our points are useful to try to get the best F-measure.

4.3 Performance

Tables 2 and 3 show the F-measure performance of each algorithm on all the datasets. Note that, in theory, given an infinite computation budget, our method should give equal (optimal) F-measure results to the grid method of [PUG14]. In practice however, [PUG14] have set the grid step to 0.1 and perform 19 model computations. To fairly compare the two methods, we have set the number of iterations of the **CONE** method to 19 too in Table 2. The results are not significantly better for any of the methods except for Yeast and Wine where $\textbf{CONE}^+$ dominates. This indicates that, on these datasets, the grid step chosen by [PUG14] is relevant to approach the optimal F-measure and that our method logically yields comparable final results. Note that the classical methods (denoted by the subscript $_{I.R.}$) which consists of weighting each classes with respect to their representation in the dataset is not giving the best F-Measure. Both grid and **CONE** methods are giving best results.

Table 3 shows the performance of both methods (except the $_{I.R.}$ one) with a limited budget : the number of iterations/computations is set to 9 and 4. With 9 iterations and for the first four datasets (Adult, Abalone10, IJCNN'01 and Abalone12), the results are not significantly better for any of the methods which indicates that a coarser grid is sufficient to find a good F-measure. For the most imbalanced datasets (Yeast and Wine) the results are significantly better but they are more unstable (the variance indicated between parenthesis is much higher). This last observation is emphasized when it comes to use only four iterations. On the three most imbalanced dataset (mainly on Abalone12 and Wine) our method $\textbf{CONE}^+$ is able to focus on the right space of research in the (t, F) space which is translated by a non-zero F-Measure measured on the test set. Furthermore, on the Yeast dataset the propo-

sed method is much stable than the grid one. When it comes to **CONE**, we think that the results are worst because it is taking another direction of reasearch compare to the algorithm which prunes the space.

5 Conclusion

We have presented **CONE**, a novel iterative approach for F-measure optimization based on cost-sensitive classification. This approach has strong theoretical guarantees showing that the F-measure of the output model is close to the optimal one with respect to the class of hypothesis considered. Furthermore, it has an efficient algorithmic strategy allowing us to prune drastically the search space of possible costs leading to a fast convergence to the true parameters. We experimentally demonstrated the efficiency of our approach on many imbalanced datasets and the interest of our theoretical framework. The results provided in this paper improve the ones obtained in [PUG14] both from a theoretical and practical standpoint in a context where there are relatively few papers that provide strong theoretical results on F-measure classification. Our perspectives include the refinement of our theoretical framework for exploring more efficiently the search space or extension to more difficult contexts such as the SGD-based algorithms used with neural networks. We also aim at working on harder settings where the imbalance ratio can be larger than 100 with very few positive data.

Références

[BFSDH15] Róbert Busa-Fekete, Balázs Szörényi, Krzysztof Dembczynski, and Eyke Hüllermeier. Online f-measure optimization. In *NIPS*, 2015.

[CBK09] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection : A survey. *ACM Comput. Surv.*, 2009.

[DWCH11] Krzysztof J Dembczynski, Willem Waegeman, Weiwei Cheng, and Eyke Hüllermeier. An exact algorithm for f-measure maximization. In *NIPS*, 2011.

[Jan05] M. Jansche. Maximum expected f-measure training of logistic regression models. In *EMNLP*, 2005.

[Joa05] T. Joachims. A support vector method for multivariate performance measures. In *ICML*, 2005.

[LFG+13] Victoria Lopez, Alberto Fernandez, Salvador Garcia, Vasile Palade, and Francisco Herrera. An insight into classification with imbalanced data : Empirical results and current trends on using data intrinsic characteristics. *Information Sciences*, 250 :113 – 141, 2013.

[MKO+03] David R Musicant, Vipin Kumar, Aysel Ozgur, et al. Optimizing f-measure with support vector machines. In *FLAIRS*, 2003.

[NKJ15] Harikrishna Narasimhan, Purushottam Kar, and Prateek Jain. Optimizing non-decomposable performance measures : A tale of two classes. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 199–208, 2015.

[PCM13] S.K. Shevade P.M. Chinta, P. Balamurugan and M.N. Murty. Optimizing f-measure with non-convex loss and sparse linear classifiers. In *IJCNN*, 2013.

[PUG14] Shameem Puthiya Parambath, Nicolas Usunier, and Yves Grandvalet. Optimizing f-measures by cost-sensitive classification. In *NIPS*, pages 2123–2131, 2014.

[vR74] C. J. van Rijsbergen. Further experiments with hierarchic clustering in document retrieval. *Information Storage and Retrieval*, 10(1) :1–14, 1974.

[YCLC12] Nan Ye, Kian Ming Adam Chai, Wee Sun Lee, and Hai Leong Chieu. Optimizing f-measure : A tale of two approaches. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, 2012.

[ZEPB13] Ming-Jie Zhao, Narayanan Edakunni, Adam Pocock, and Gavin Brown. Beyond fano's inequality : Bounds on the optimal f-score, ber, and cost-sensitive risk and their implications. *JMLR*, 2013.