Learning from Few Positives: a Provably Accurate Metric Learning Algorithm to deal with Imbalanced Data

Rémi Viola, Rémi Emonet, Amaury Habrard, Guillaume Metzler, Marc Sebban

Learning from Few Positives: a Provably Accurate Metric Learning Algorithm to deal with Imbalanced Data

Rémi Viola[1,2], Rémi Emonet[1], Amaury Habrard[1], Guillaume Metzler[1], and Marc Sebban[1]

[1]University of Lyon, UJM-Saint-Etienne, CNRS,, Institute of Optics Graduate School, Laboratoire Hubert Curien UMR 5516, Saint-Etienne, France.
firstname.name@univ-st-etienne.fr
[2]DGFiP, Ministry of Economy and Finance, Paris, France.
remi.viola@dgfip.finances.gouv.fr

Abstract

Learning from imbalanced data, where the positive examples are very scarce, remains a challenging task from both a theoretical and algorithmic perspective. In this paper, we address this problem using a *metric learning* strategy. Unlike the state-of-the-art methods, our algorithm **MLFP**, for *Metric Learning from Few Positives*, learns a new representation that is used only when a test query is compared to a minority training example. From a geometric perspective, it artificially brings positive examples closer to the query without changing the distances to the negative (majority class) data. This strategy allows us to expand the decision boundaries around the positives, yielding a better F-Measure, a criterion which is suited to deal with imbalanced scenarios. Beyond the algorithmic contribution provided by **MLFP**, our paper presents generalization guarantees on the false positive and false negative rates. Extensive experiments conducted on several imbalanced datasets show the effectiveness of our method.

1 Introduction

Fraud detection in bank or insurance applications Abdallah *et al.* [2016]; Schiller [2006], and anomaly identification for medical diagnosis Aggarwal [2017] are some societal challenges requiring to address the problem of learning from highly imbalanced data. When dealing with such a setting, one has to face two major issues: (i) the scarcity of the class of interest, only composed of a few positive data, which limits the efficiency of standard margin-based loss functions; (ii) the scattering of positive examples in the total mass of the training data, which makes the estimation of local densities much more complicated than in balanced scenarios. Several solutions have been proposed in the literature to address these two problems. Most of them consist in applying sampling strategies which aim to balance the dataset by reducing the number of negative examples and/or creating new synthetic positive data Sharma *et al.* [2018]; Pérez-Ortiz *et al.* [2019]. On the other hand, one can resort to cost-sensitive algorithms Khan *et al.* [2017] which assign a weight to each class (or even to each example) so that the classifier can focus better on the minority class. Other strategies include the use of ensemble methods Wu *et al.* [2017]; Frery *et al.* [2018] or the specific adaptation of existing approaches such as deep learning Huang *et al.* [2016]; Dumpala *et al.* [2018] or kernel methods Mathew *et al.* [2015]; Ding *et al.* [2018]; Zhang *et al.* [2019].

In this paper, we address the problem of learning from imbalanced data from a metric learning perspective Bellet *et al.* [2013]; Kulis and others [2013]. Learning a metric specifically designed for the application at hand may present several advantages in the context of imbalanced datasets: (i) the metric can be learned

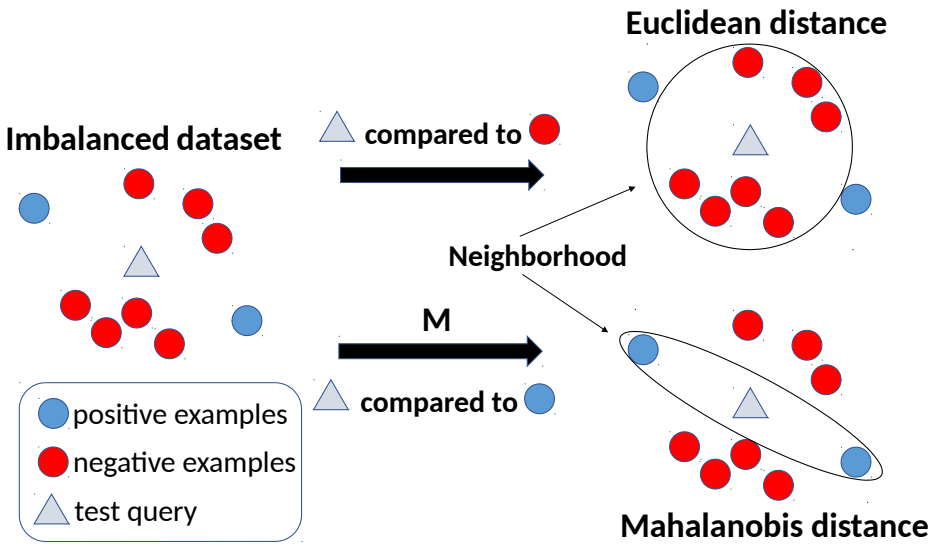

Figure 1: Intuition behind our method **MLFP**: a PSD matrix $\mathbf{M}$ is optimized under constraints, and is used only when a test query is compared to a positive example. The distance to the negative examples is kept unchanged. This allows the learned metric to expand the decision boundaries around the positives and thus to capture more examples of the class of interest.

under semantic constraints allowing us to expand the decision boundaries around the positives; (ii) this framework enables to design optimization problems based on the geometry of the data without suffering from the issues of standard accuracy-based loss functions (*e.g.*, hinge loss for SVMs, exponential loss for boosting, logistic loss for logistic regression); (iii) metric learning is a nice setting to derive theoretical guarantees on the learned transformation Bellet *et al.* [2015]. Surprisingly, despite these interesting features, metric learning has not received much attention to address the problem of learning from imbalanced data (see, *e.g.*, the recent papers Feng *et al.* [2018], Wang *et al.* [2018] and Gautheron *et al.* [2019]). The goal of this paper is to bridge this gap from both an algorithmic and a theoretical perspective. As illustrated in Figure 1, we propose the algorithm **MLFP** that optimizes a linear transformation (via a *Positive Semi Definite* (PSD) matrix $\mathbf{M}$ of a Mahalanobis distance) only when a test query is compared to a minority training example. A single metric $\mathbf{M}$ is learned for the whole space taking the geometry of the data into account. Unlike the standard metric learning algorithms (see, *e.g.*, **LMNN** Weinberger and Saul [2009] or **ITML** Davis *et al.* [2007]), our method boils down to artificially bringing positive examples closer to the query without challenging the features of the negatives. This has a direct impact on the decision boundaries around the positives allowing us to capture more examples of the class of interest yielding a better F-Measure (see Section 3 for a formal definition). By using the uniform stability framework, we derive theoretical guarantees on the learned matrix $\mathbf{M}$ showing the actual capability of **MLFP** to control the false positive and false negative rates.

The paper is organized as follows. In Section 2, we report some related work on metric learning for imbalanced data classification. Section 3 is dedicated to the presentation of our metric learning algorithm **MLFP**. Section 4 presents a theoretical analysis using the uniform stability framework and Section 5 illustrates the performance of **MLFP** compared to state-of-the-art algorithms.

2 Related Work

Most of the metric learning algorithms (see Bellet *et al.* [2013]; Kulis and others [2013] for a survey) are based on the optimization of the Mahalanobis distance between two points $\mathbf{x}_i$ and $\mathbf{x}_j \in \mathbb{R}^q$:

$$d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j)^2 = (\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{M} (\mathbf{x}_i - \mathbf{x}_j),$$

where $\mathbf{M}$ is a $q \times q$ *Positive Semi Definite* matrix. One can express $\mathbf{M}$ as $\mathbf{L}^T\mathbf{L}$ where $\mathbf{L}$ is a $r \times q$ matrix where r is the rank of $\mathbf{M}$. Thus, this distance can be seen as the Euclidean distance in a new feature space $\mathbf{Lx}$.

A well-known representative of this family of algorithms is the *Large Margin Nearest Neighbor* (**LMNN**) Weinberger and Saul [2009]. For each example of a training set of size m, the learned metric $\mathbf{M}$ aims to bring closer the neighbors of the same class (called target neighbors) while pushing away the examples of other classes (the impostors). This algorithm has been shown to be very efficient and to scale well with large datasets. However, it is worth noticing that **LMNN** is not designed to take into account some imbalance in the data. Indeed, the similarity constraints constructed from pairs of examples of the same class do not make any difference between the positive and negative examples. Therefore, in imbalanced scenarios, **LMNN**, as the other state-of-the-art methods, is prone to focus on the majority class and thus is subject to miss the positive examples.

The first attempts to address the problem of learning a metric from imbalanced datasets have been proposed very recently. Wang *et al.* [2018] introduce an *iterative metric learning* algorithm (**IML**) that aims to define a stable neighborhood used to predict the label of a new test data. The method repeats two main steps: (i) the learning of a linear transformation, *e.g.*, by using **LMNN**, and (ii) a training sample selection given a test example. The procedure is repeated until stabilization of the neighborhood. By repeating the process several times, **IML** is able to locally separate positives from negatives. However, the main issue comes from the algorithmic complexity of the method, which requires to apply **LMNN** and to update the pairs used for the training process at each iteration. Another approach to learn metrics from imbalanced datasets has been recently proposed Gautheron *et al.* [2019]. In their *Imbalanced Metric Learning* algorithm (**ImbML**), the authors take into account the nature of the pairwise constraints by using two different sub-losses, one for each label, weighted according to the number of positive and negative examples respectively. This intuitive and natural way to proceed prevents the algorithm from favoring the majority class. However, we will see that applying the learned metric $\mathbf{M}$ to all examples is not necessary, focusing only on the minority class appears to be much more efficient and allows us notably to better control the false negatives. Finally, Feng *et al.* [2018] introduce **DMBK** for *Distance Metric by Balancing KL-divergence*. This algorithm resorts to the KL-divergence to represent normalized between-class divergences. Combined with a geometric mean, **DMBK** is able to make these divergences balanced. Note that this method makes sense in the multi-class setting, but is meaningless for addressing binary problems, due to the use of the normalization while computing the KL-divergence.

Beyond the algorithmic limitations of the previous state-of-the-art algorithms, note that none of them comes with guarantees on the classification error. In this paper, we address this problem by studying the capability of **MLFP** to optimize a metric $\mathbf{M}$ which provides a good compromise between (i) expanding the decision boundaries around the positives which enables to reduce the false negative rate at test time (one of the main issues faced in imbalanced learning); (ii) controlling this expansion to prevent the algorithm from detecting too many false alarms, represented by the false positive rate. The theoretical results take the form of guarantees on the learned metric using the uniform stability framework Bousquet and Elisseeff [2002] which measures the stability of the output of the algorithm when the training set is subject to slight changes.

3 Metric Learning for Imbalanced Data

In this section, we present our algorithm **MLFP**, for *Metric Learning from Few Positives*. In the following, we denote by $S = \{\mathbf{z}_i = (\mathbf{x}_i, y_i)\}_{i=1}^m$ the set of m training examples drawn *i.i.d.* from an unknown joint distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, where $\mathbf{x}_i \in \mathcal{X}$ (here $\mathcal{X} = \mathbb{R}^q$) is a feature vector and $y_i \in \mathcal{Y}$ (here $\mathcal{Y} = \{-1, +1\}$) corresponds to its associated label. The label $+1$ is used to denote the positive or the minority class. We further note $S = S_+ \cup S_-$ with S_+ the set of m_+ positive examples and S_- the set of m_- negative examples, such that $m = m_+ + m_-$.

3.1 Problem Formulation

In our approach, we use the Euclidean distance when comparing a query point to a majority-class example. The originality comes from the use of an optimized Mahalanobis distance when comparing a query to a minority-class sample. The objective of this strategy is to formulate a metric learning problem leading to a classifier (a kNN here) which is accurate on both classes even in an imbalanced scenario.
In order to avoid the pitfall of classic metric learning algorithms that are prone to focus on the majority class, we propose to give more importance to the minority class composed of the positive instances. Our algorithm **MLFP** tries to control the false positive (FP) and false negative (FN) rates thanks to the following constrained optimization problem:

$$\min_{\mathbf{M} \in \mathbb{S}^+} \frac{1}{m^3} \left((1-\alpha) \sum_{\substack{(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) \\ y_i = y_j = 1 \neq y_k}} \ell_{\text{FN}}(\mathbf{M}, \mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_k) + \alpha \sum_{\substack{(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) \\ y_i = y_j = -1 \neq y_k}} \ell_{\text{FP}}(\mathbf{M}, \mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_k) \right) + \mu \|\mathbf{M} - \mathbf{I}\|_{\mathcal{F}}^2,$$

$$\text{such that } \lambda_{\max}(\mathbf{M}) \leq 1. \tag{1}$$

where $\mathbb{S}^+$ is the set of PSD matrices, $\lambda_{\max}(\mathbf{M})$ is the largest eigenvalue of the PSD matrix $\mathbf{M}$, ℓ_{FN} and ℓ_{FP} are defined by:

$$\ell_{\text{FN}}(\mathbf{M}, \mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_k) = [1 - c + d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j)^2 - d(\mathbf{x}_i, \mathbf{x}_k)^2]_+$$

and

$$\ell_{\text{FP}}(\mathbf{M}, \mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_k) = [1 - c + d(\mathbf{x}_i, \mathbf{x}_j)^2 - d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_k)^2]_+,$$

where $[a]_+ = \max(0, a)$, α is the positive rate $\dfrac{m_+}{m}$ and $\mu \|\mathbf{M} - \mathbf{I}\|_{\mathcal{F}}^2$ is a regularization term which penalizes a large deviation from the Euclidean distance. The hyper-parameter c controls the margin we want to preserve between pairs of dissimilar examples according to the Euclidean space and the learned one.
Problem (1) is composed of two terms where triplets are involved. Unlike standard metric learning algorithms, our method takes into account both the Euclidean distance d and the metric learned d_M. More precisely: the first term ℓ_{FN} aims to gather the minority class examples with respect to the learned metric such that the distance between two positives (using $\mathbf{M}$) is less than the distance to a negative example (using the Euclidean distance). This subloss can be seen as a way to prevent the model from generating false negatives (FN). The second term ℓ_{FP} works in a similar manner. The only difference lies in the fact that the query $\mathbf{x}_i$ is a negative example. Thus, we learn $\mathbf{M}$ such that the positive queries $\mathbf{x}_k$ are not bringing too close to $\mathbf{x}_i$, *i.e.* the Euclidean distance between two negatives $\mathbf{x}_i$ and $\mathbf{x}_j$ (with respect to the Euclidean distance) is lower than the distance between $\mathbf{x}_i$ and $\mathbf{x}_k$ (with respect to $\mathbf{M}$). This subloss can be seen as a way to prevent the model from generating false positives (FP).

Both FN and FP are important terms to optimize measures that are more suited to deal with imbalanced settings, such as the F-Measure Rijsbergen [1979] defined as follows:

$$F_1 = \frac{2(m_+ - FN)}{2m_+ - FN + FP}.$$

Minimizing the F-Measure boils down to finding a good trade-off between FP and FN. However, in a highly imbalanced setting, where m_+ is very low, missing only a few positives leads to a dramatic decrease of the F-Measure. That is why we constrain the largest eigenvalue $\lambda_{\max}(\mathbf{M})$ to be lower than 1, so that the learned matrix $\mathbf{M}$ aims to pay more attention to the positive class. In the next section, we provide a formal explanation of its use.

3.2 On the Impact of the Constraint

We study the impact of the $\lambda_{\max}(\mathbf{M})$ value on both FN and FP and, thus the influence of the constraint of our optimization problem.

Proposition 1. *Let $\mathbb{P}[FN_{\mathbf{M}}(\mathbf{x})]$ (resp. $\mathbb{P}[FP_{\mathbf{M}}(\mathbf{x})]$) be the probability of a positive query (resp. a negative query) $\mathbf{x}$ of being a false negative (resp. a false positive) using the 1-NN algorithm with the learned matrix $\mathbf{M}$ and $\mathbb{P}[FN(\mathbf{x})]$ (resp. $\mathbb{P}[FP(\mathbf{x})]$) the same probability using the Euclidean distance.*
Then, if $\lambda_{\max}(\mathbf{M}) \leq 1$, we have:

$$\mathbb{P}[FN_{\mathbf{M}}(\mathbf{x})] \leq \mathbb{P}[FN(\mathbf{x})] \text{ and } \mathbb{P}[FP_{\mathbf{M}}(\mathbf{x})] \geq \mathbb{P}[FP(\mathbf{x})].$$

Sketch of proof. Let ε be the distance from $\mathbf{x}$ to its nearest neighbor $N_{\mathbf{x}}$. The example $\mathbf{x}$ is a false negative if $N_{\mathbf{x}} \in S_-$, that is, all positives $\mathbf{x}' \in S_+$ are outside an ellipsoid $\mathcal{E}_{\varepsilon,\mathbf{M}^{-1}}(\mathbf{x})$, defined by ε and $\mathbf{M}$. Therefore, we have:

$$\mathbb{P}[FN_{\mathbf{M}}(\mathbf{x})] = (1 - \mathbb{P}\left[\mathbf{x}' \in \mathcal{E}_{\varepsilon,\mathbf{M}^{-1}}(\mathbf{x})\right])^{m_+}. \tag{2}$$

When the Euclidean distance is used, we deal with a standard sphere $\mathcal{S}_\varepsilon$ of radius ε, and we get:

$$\mathbb{P}[FN(\mathbf{x})] = (1 - \mathbb{P}\left[\mathbf{x}' \in \mathcal{S}_\varepsilon(\mathbf{x})\right])^{m_+}. \tag{3}$$

Having $\lambda_{\max}(\mathbf{M}) \leq 1$ implies Eq. (2) $\leq$ Eq. (3). Indeed $\lambda_{\max}(\mathbf{M}) \leq 1$ implies that the sphere $\mathcal{S}_\varepsilon$ is included in the ellipsoid $\mathcal{E}_{\varepsilon,\mathbf{M}^{-1}}$ as illustrated in Figure 2. By this choice, we expand the decision boundaries around positives and thus capture more minority class examples. Using a similar scheme, we can prove the second inequality of Proposition 1. When $\mathbf{x}$ is negative and $N_{\mathbf{x}} \in S_+$, we have

$$\mathbb{P}[FP_{\mathbf{M}}(\mathbf{x})] \quad = \quad (1 - \mathbb{P}\left[\mathbf{x}' \in \mathcal{E}_{\varepsilon,\mathbf{M}^{-1}}(\mathbf{x})\right])^{m_-}, \tag{4}$$

and

$$\mathbb{P}[FP(\mathbf{x})] \quad = \quad (1 - \mathbb{P}\left[\mathbf{x}' \in \mathcal{S}_\varepsilon(\mathbf{x})\right])^{m_-}. \tag{5}$$

□

From Equations (2) and (4), we can note that they are both exponentially decreasing *w.r.t.* to the number of positives and negatives respectively. However, in imbalanced scenarios, the number of negatives is supposed to be much higher than the number of positives. Thus, the probability of having a false positive is decreasing faster than the probability of having a false negative. We then choose to learn a matrix $\mathbf{M}$ under the constraint $\lambda_{\max}(\mathbf{M}) \leq 1$, so that our algorithm will focus first on reducing FN. An illustration of the impact of this constraint in terms of decision boundaries is shown in Figure 3. The experiments in Section 5 will confirm that the use of this constraint is very relevant from an F-Measure perspective and is able to reduce the number of FN at test time.

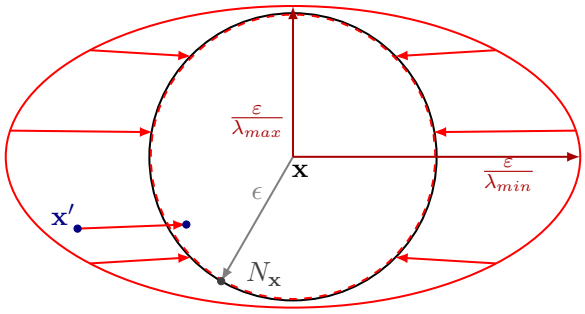

Figure 2: Illustration of the constraint $\lambda_{\max}(\mathbf{M}) \leq 1$. Without learning the matrix $\mathbf{M}$, the Euclidean distance is used both to compare a query $\mathbf{x}$ to a negative $N_{\mathbf{x}}$ and to a positive $\mathbf{x}'$. The isodistance curves are thus spherical and identical (one in solid black for $N_{\mathbf{x}}$, one in dashed red for $\mathbf{x}'$). By learning the matrix $\mathbf{M}$, we virtually change the distance of the query to the positive examples. The isodistance curves for the positives are now ellipses, like the one represented in red. In the example, the positive $\mathbf{x}'$, that is outside the sphere, is inside the ellipse and will thus be considered closer, with the constraint $\lambda_{\max}(\mathbf{M}) \leq 1$, than the negative $N_{\mathbf{x}}$ that lies on the black sphere. With this same constraint, we are sure that the ellipse is enclosing the circle (*i.e.* $\frac{\varepsilon}{\lambda_{max}} \geq \varepsilon$) and so that all positives will be brought closer to the query. In the end, this constraint ensures that we increase the influence of the positives and thus leads to the decrease of FN.

4 Theoretical Analysis

In this section, we provide generalization guarantees about the learned metric $\mathbf{M}$ using the uniform stability framework Bousquet and Elisseeff [2002] adapted to metric learning Bellet *et al.* [2015]. Then, we use this result to derive classification guarantees over a 1-Nearest Neighbor (1NN) classifier making use of this metric. Note that the whole study is conducted under the constraint $\lambda_{\max}(\mathbf{M}) \leq 1$ as used in Problem (1). First, we denote by ℓ the weighted combination of ℓ_{FN} and ℓ_{FP} as defined in Problem (1) and F_S the objective function to optimize over the training set $S = \{\mathbf{z}_i\}_{i=1}^m$. We have

$$F_S = \frac{1}{m^3} \sum_{i,j,k=1}^{m} \ell(\mathbf{M}, (\mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_k)) + \mu \|\mathbf{M} - \mathbf{I}\|_{\mathcal{F}}^2.$$

Let $\mathcal{R}_S$ be the associated empirical risk over S defined as

$$\mathcal{R}_S = \frac{1}{m^3} \sum_{i,j,k=1}^{m} \ell(\mathbf{M}, (\mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_k)),$$

and $\mathcal{R}$ be the corresponding expected true risk defined as

$$\begin{aligned} \mathcal{R} = \mathop{\mathbb{E}}_{S \sim \mathcal{D}^m} [\mathcal{R}_S] &= \mathop{\mathbb{E}}_{S \sim \mathcal{D}^m} \left[\frac{1}{m^3} \sum_{i,j,k=1}^{m} \ell(\mathbf{M}, (\mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_k)) \right] \\ &= \mathop{\mathbb{E}}_{\mathbf{z},\mathbf{z}',\mathbf{z}'' \sim \mathcal{D}} \left[\ell(\mathbf{M}, (\mathbf{z}, \mathbf{z}', \mathbf{z}'')) \right]. \end{aligned}$$

The last equality is due to the *i.i.d.* aspect of the expectation. We also suppose that for all $\mathbf{x}$, we have $\|\mathbf{x}\| \leq K$.

4.1 Uniform Stability

Intuitively, an algorithm is stable if its output, in terms of loss, does not change significantly under a small modification of the training sample. The supremum of this change must be bounded in $\mathcal{O}(1/m)$.

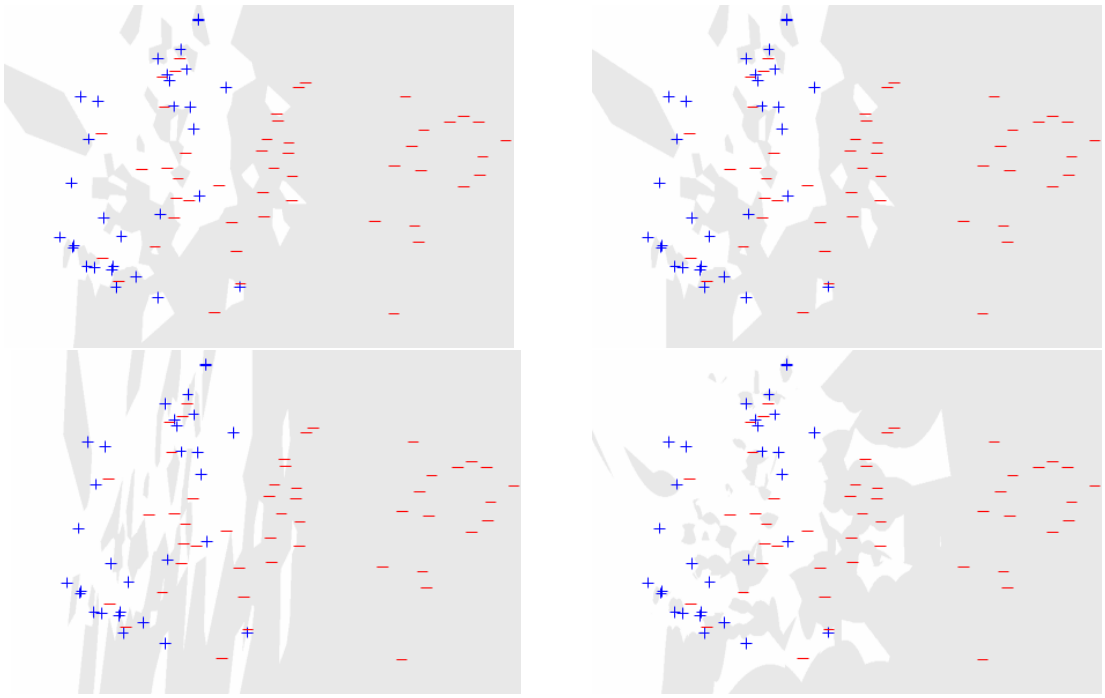

Figure 3: Illustration of the impact of the constraint $\lambda_{\max}(\mathbf{M}) \leq 1$ in **MLFP** (bottom right) compared to k**NN** (top left), **LMNN** (top right), **ImbML** (bottom left) on the *autompg* dataset with a 1NN classifier. We perform a PCA, keeping the two most relevant dimensions, and plot the test set on a mesh grid of the space. In light grey (*resp.* white), areas classified as negative (*resp.* positive).

Definition 1. *A learning algorithm $\mathcal{A}$ has a uniform stability in $\frac{\kappa}{m}$ with respect to a loss function ℓ and parameter set θ, with κ a positive constant if:*

$$\forall S,\ \forall i,\ 1 \leq i \leq m,\ \sup_{Z} |\ell(\theta_S, Z) - \ell(\theta_{S^i}, Z)| \leq \frac{\kappa}{m},$$

where S is a learning sample of size m, $Z = (\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3) = ((\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_3, y_3))$ is a triplet of labeled examples, θ_S the model parameters learned from S, θ_{S^i} the model parameters learned from the sample S^i obtained by replacing the i^{th} example $\mathbf{z}_i$ from S by another example $\mathbf{z}'_i$ independent from S and drawn from $\mathcal{D}$. Finally, $\ell(\theta_S, Z)$ is the loss suffered at Z.

In this definition, S^i represents the notion of small modification of the training sample. The next definition aims to study the evolution of the loss function according to the considered triplets Z and Z'.

Definition 2. *A loss function ℓ is said to be γ-admissible, w.r.t. the distance metric $\mathbf{M}$ if (i) it is convex w.r.t. its first argument and (ii) if the following condition holds:*

$$\forall Z, Z' \quad |\ell(\mathbf{M}, Z) - \ell(\mathbf{M}, Z')| \leq \gamma,$$

where $Z = (\mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_k)$ and $Z' = (\mathbf{z}'_i, \mathbf{z}'_j, \mathbf{z}'_k)$ are two triplets from a sample S and drawn from $\mathcal{D}$.

From the two above definitions, we can state the following generalization bound.

Theorem 1. *Let $\delta > 0$ and $m > 2$. Let S be a sample of m randomly selected training examples. Let $\mathbf{M}$ be the matrix learned from Problem* (1) *which has a uniform stability in $\frac{\kappa}{m}$. The loss function ℓ as defined*

above is γ-admissible. With probability $1-\delta$, the following bound on the true risk $\mathcal{R}$ of ℓ holds:

$$\mathcal{R} \leq \mathcal{R}_S + 2\frac{\kappa}{m} + (2\kappa + 2\gamma)\sqrt{\frac{\ln(2/\delta)}{2m}},$$

where

$$\kappa = \frac{12}{\mu} \times ((1-\alpha)K^2)^2 \text{ and } \gamma = (1-\alpha)(1-c+4K^2).$$

The derived bound provides guarantees on the generalization performances of the learned metric on the distribution $\mathcal{D}$ *w.r.t.* to the loss ℓ. We now make use of this bound to provide classification guarantees of a 1NN making use of the learned metric $\mathbf{M}$.

4.2 Classification Guarantees

We derive here generalization guarantees on the FP and FN rates for a 1NN classifier making use of the metric $\mathbf{M}$ learned by **MLFP**. Let S be the learning sample of size m used by a nearest-neighbor classifier. Let us define the empirical risks for FP and FN:

$$\mathcal{R}_{FP}(S) = \mathbb{E}_{\mathbf{z}=(\mathbf{x},y)\sim D} \mathbb{1}_{\{d_{\mathbf{M}}(\mathbf{x},\mathbf{x}_p)^2 \leq d(\mathbf{x},\mathbf{x}_n)^2\}} \times \mathbb{1}_{\{y=-1\}}.$$

where $\mathbf{x}_p, \mathbf{x}_n \in S$ are respectively the nearest positive and negative neighbors of $\mathbf{x}$ in S. Symmetrically, we have:

$$\mathcal{R}_{FN}(S) = \mathbb{E}_{\mathbf{z}=(\mathbf{x},y)\sim D} \mathbb{1}_{\{d(\mathbf{x},\mathbf{x}_n)^2 \leq d_{\mathbf{M}}(\mathbf{x},\mathbf{x}_p)^2\}} \times \mathbb{1}_{\{y=1\}}.$$

We consider then the expected true risks averaged over all the training samples of size m:

$$\mathcal{R}_{FP} = \mathbb{E}_{S\sim D^m} \mathcal{R}_{FP}(S) \text{ and } \mathcal{R}_{FN} = \mathbb{E}_{S\sim D^m} \mathcal{R}_{FN}(S).$$

We can now introduce our main result.

Theorem 2. *Let $\delta > 0$ and $m > 0$. Let S be a training sample of size m i.i.d. from a distribution D, z a new instance i.i.d. from D, and let $\mathbf{M}$ be the learned matrix from Problem* (1) *which has a uniform stability in $\frac{\kappa}{m}$ with respect to the loss ℓ. Considering that the loss function ℓ is γ-admissible, let us denote by $\mathcal{R}_S$ its empirical risk. With probability $1-\delta$, we have the following bounds for the FP and FN rates:*

$$\mathcal{R}_{FP} \leq \frac{1}{\alpha}\left[\mathcal{R}_{S\cup\{z\}} + \frac{2\kappa}{m+1} + (2\kappa+2\gamma)\sqrt{\frac{\ln(2/\delta)}{2(m+1)}}\right],$$

$$\mathcal{R}_{FN} \leq \frac{1}{1-\alpha}\left[\mathcal{R}_{S\cup\{z\}} + \frac{2\kappa}{m+1} + (2\kappa+2\gamma)\sqrt{\frac{\ln(2/\delta)}{2(m+1)}}\right].$$

By comparing these two bounds, one can observe that when the class imbalance becomes important, *i.e.* when α takes a low value, the guarantees on the FN rate become better than the guarantees on FP. This result provides a theoretical confirmation that our approach - thanks to the constraint $\lambda_{\max}(\mathbf{M}) \leq 1$ - is able to focus more on reducing FN. An illustration of this phenomenon will be shown in the next section.

5 Experiments

In this section, we compare **MLFP** to other metric learning algorithms, focusing on (highly) imbalanced datasets. For all experiments, we use a 3-Nearest Neighbor classifier as done in both Weinberger and Saul [2009] and Wang *et al.* [2018]. Note that the source code allowing the interested reader to reproduce these experiments is available[1].

[1] https://github.com/RemiViola/MLFP

DATASETS	SIZE	DIM	IR	**3NN**	**LMNN**	**ITML**	**IML**	**IMBML**	**MLFP** (OURS)
BALANCE	625	4	1.2	0.880 ± 0.018	0.874 ± 0.019	0.931 ± 0.032	0.886 ± 0.029	$\mathbf{0.960 \pm 0.019}$	0.874 ± 0.003
AUTOMPG	392	7	1.7	0.780 ± 0.054	0.792 ± 0.031	0.801 ± 0.018	0.785 ± 0.021	0.790 ± 0.044	0.805 ± 0.021
IONOSPHERE	351	34,	1.8	0.745 ± 0.015	0.803 ± 0.049	0.831 ± 0.054	0.823 ± 0.044	0.786 ± 0.053	$\mathbf{0.923 \pm 0.026}$
PIMA	768	8	1.9	0.601 ± 0.042	0.591 ± 0.037	0.583 ± 0.022	0.591 ± 0.037	0.575 ± 0.026	$\mathbf{0.635 \pm 0.032}$
WINE	178	13	2	0.968 ± 0.016	0.992 ± 0.016	0.992 ± 0.016	0.992 ± 0.016	$\mathbf{0.992 \pm 0.016}$	0.961 ± 0.041
GLASS	214	9	2.1	0.735 ± 0.049	0.710 ± 0.064	$\mathbf{0.759 \pm 0.051}$	0.710 ± 0.064	0.716 ± 0.043	0.747 ± 0.034
GERMAN	1000	23	2.3	0.407 ± 0.049	0.358 ± 0.029	0.430 ± 0.073	0.352 ± 0.029	0.388 ± 0.043	$\mathbf{0.511 \pm 0.006}$
VEHICLE	846	18	3.3	0.850 ± 0.045	0.928 ± 0.024	0.931 ± 0.019	0.933 ± 0.026	$\mathbf{0.937 \pm 0.014}$	0.859 ± 0.037
HAYES	132	4	3.4	0.581 ± 0.210	0.824 ± 0.089	0.829 ± 0.071	0.824 ± 0.089	0.908 ± 0.083	$\mathbf{0.930 \pm 0.109}$
SEGMENTATION	2310	19	6	0.882 ± 0.031	0.888 ± 0.011	0.866 ± 0.029	0.895 ± 0.020	$\mathbf{0.909 \pm 0.028}$	0.882 ± 0.024
ABALONE8	4177	10	6.4	0.223 ± 0.025	0.220 ± 0.040	0.213 ± 0.025	0.228 ± 0.021	0.200 ± 0.023	$\mathbf{0.336 \pm 0.018}$
YEAST3	1484	8	8.1	0.719 ± 0.028	0.734 ± 0.020	0.742 ± 0.034	0.717 ± 0.032	0.723 ± 0.023	0.725 ± 0.022
PAGEBLOCKS	5473	10	8.8	0.855 ± 0.027	0.844 ± 0.027	0.850 ± 0.023	0.842 ± 0.027	$\mathbf{0.865 \pm 0.021}$	0.860 ± 0.022
SATIMAGE	6435	36	9.3	0.688 ± 0.034	0.707 ± 0.038	0.710 ± 0.024	0.710 ± 0.039	$\mathbf{0.731 \pm 0.030}$	0.697 ± 0.030
LIBRAS	360	90	14	0.694 ± 0.188	0.725 ± 0.105	0.722 ± 0.204	0.690 ± 0.120	$\mathbf{0.729 \pm 0.157}$	0.694 ± 0.066
REDWINEQUALITY4	1599	11	29.2	0.062 ± 0.075	0.057 ± 0.114	0.027 ± 0.053	0.000 ± 0.000	0.031 ± 0.062	$\mathbf{0.083 \pm 0.039}$
YEAST6	1484	8	41.4	0.560 ± 0.205	0.578 ± 0.246	0.523 ± 0.205	$\mathbf{0.629 \pm 0.244}$	0.606 ± 0.148	0.527 ± 0.152
ABALONE17	4177	10	71	0.000 ± 0.000	0.000 ± 0.000	0.029 ± 0.057	0.000 ± 0.000	$\mathbf{0.073 \pm 0.000}$	0.053 ± 0.033
ABALONE20	4177	10	159.7	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.044 ± 0.089	0.000 ± 0.093	$\mathbf{0.078 \pm 0.029}$
MEAN				0.591	0.612	0.619	0.613	0.627	**0.643**

Table 1: Mean results (and standard deviations) in terms of F-Measure over 5 experiments for the different Metric Learning methods, with **3NN** as final classifier, on datasets *sorted by imbalance ratio* (IR=m_-/m_+). The properties of the considered datasets are given on the left hand part of the table: size, dimension and IR. The *mean* over all datasets among ML methods is given and the best results are in *bold*, the standard deviation is indicated with the $\pm$ sign.

5.1 Experimental Setup

We use several public datasets from the UCI[2] and KEEL[3] repositories. These datasets are diverse in terms of imbalance ratio (IR, number of majority examples per positive example), dimension, number of examples, as shown in Table 1. All the datasets are standardized by substracting the mean and dividing by the standard deviation.

We use the F-Measure as the performance criterion to compare the different methods.

Furthermore, 80% of the dataset is randomly selected in order to train the model and 20% to test it. The different hyper-parameters are tuned with a 10-fold-cross-validation over the training set. The sampling of the test set is repeated 5 times and we report the average results in terms of F-Measure (F_1).

For our **MLFP** method, the hyper-parameters μ for the regularization and c for the margin are both tuned in the range $[0, 1]$, using a Bayesian optimization with 400 calls. The Bayesian optimization is done with the Scikit-Optimize library[4]. As the matrix $\mathbf{M}$ can be expressed as $\mathbf{L}^T\mathbf{L}$ (Cholesky decomposition), we directly learn a diagonal matrix $\mathbf{L}$. Since we are not particularly interested, in this paper, in low rank matrices, we do not impose any constraint on the dimension of $\mathbf{L}$. At each iteration of the optimization process, the spectral radius of the matrix $\mathbf{L}$ is constrained to be less than one so that $\mathbf{M} = \mathbf{L}^T\mathbf{L}$ has its largest value less than one.

We compare **MLFP** with several methods: The 3-Nearest Neighbor algorithm (**3NN**), as a baseline. **LMNN**, where the hyper-parameter μ, which controls the trade-off between the two parts of the loss (see Weinberger and Saul [2009] for more details), is tuned in $[0, 1]$ using a Bayesian optimization with 20 calls. **ITML** Davis *et al.* [2007]. **IML** Wang *et al.* [2018] where we select $5k$ points for the sampling selection and we also tune the hyper-parameter of the **LMNN** algorithm in $[0, 1]$. We used 0.8 for the ratio of matching as suggested in the paper. **ImbML** Gautheron *et al.* [2019] where the parameter m is tuned in $\{1, 10, 100, 1000, 10000\}$, the parameter λ in $\{0, 0.01, 0.1, 1, 10\}$ and the parameter a in $[0, 1]$. We also use a Bayesian optimization with 400 calls.

[2]https://archive.ics.uci.edu/ml/datasets.html

[3]https://sci2s.ugr.es/keel/datasets.php

[4]https://scikit-optimize.github.io/

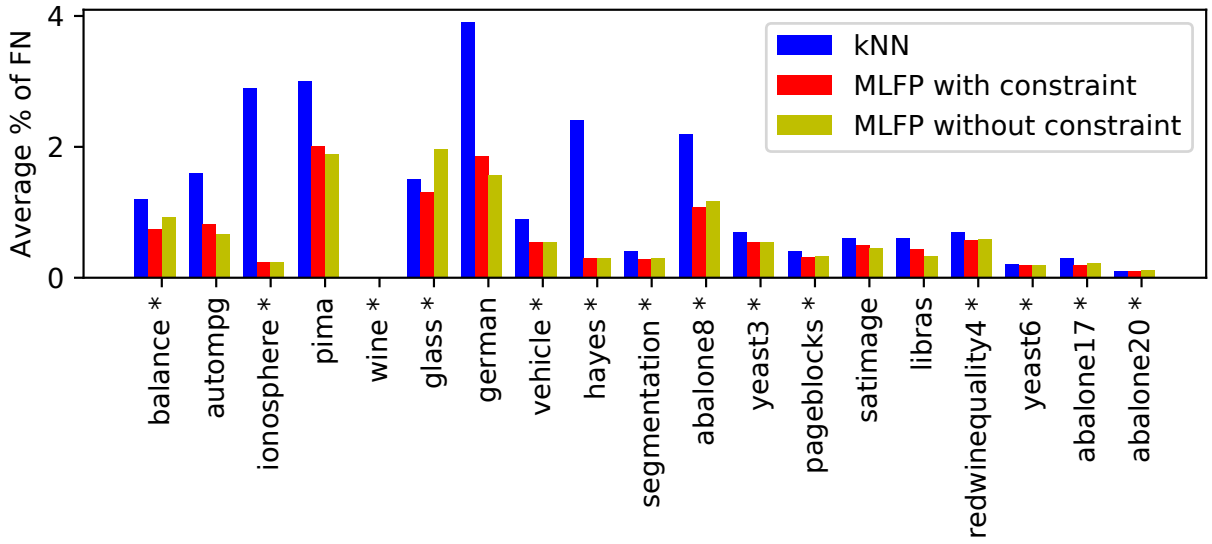

Figure 4: Average percentage of false negatives for each dataset at test time (see Section 5 for more details), for k**NN** and **MLFP** with or without the constraint on λ_{max}. On 14 datasets (with $*$) over 19, the number of FN is lower for the version with the constraint. Note that the number of FN is always lower with **MLFP** compared to k**NN**.

5.2 Results

The main results are reported in Table 1. Unsurprisingly, all metric learning methods perform better than a **3NN**. Furthermore, in terms of F-Measure, those which were designed to deal with imbalanced scenarios perform better than **LMNN** or **ITML**. However, the most competitive method is **MLFP**: the F-Measure is increased on average by 1.6 points compared to the second best method (**ImbML**). More precisely, our **MLFP** outperforms all the other methods on 8 (over 19) datasets. The fact that **MLFP** works better than **ImbML** shows the advantage of learning a specific metric when computing distances to positive examples. Furthermore, as shown on Figure 3, both **ImbML** and **MLFP** focuses on the minority class, but they perform this task in a different way. Our method tries to reduce the number of FN by increasing the decision boundaries around each of positive. In **ImbML**, the possibility of having large margins in the learned space has the disadvantage of creating larger areas of negative classification and this potentially increases the risk of FN.

In the theoretical part of this paper, we have proved that learning a matrix $\mathbf{M}$ under the constraint $\lambda_{\max}(\mathbf{M}) \leq 1$ allows our algorithm to focus first on reducing FN. An illustration of the impact of this contraint in terms of false negatives is shown in Figure 4 on the 19 datasets. This figure reports the percentage of false negatives **at test time** generated by the **3NN** algorithm and **MLFP** with or without the constraint. The results show that, compared to a **3NN** algorithm, **MLFP** systematically reduces the number of false negatives and thus has the desired effect. When comparing **MLFP** with and without the constraint, we can note that on 14 datasets out of 19, the use of the constraint $\lambda_{\max}(\mathbf{M}) \leq 1$ leads at test time to a smaller number of false negatives.

6 Conclusion

In this paper, we have proposed a new metric learning algorithm to deal with imbalanced datasets. In this setting, finding the good compromise between the false negative and false positive rates is still an open problem. The original contribution of this paper comes from the optimization in our algorithm **MLFP** of a

Mahalanobis distance which is *only* used to compare a new query to positive examples, while the Euclidean distance is still used when for comparing that query to negative samples. A constraint on the maximum eigenvalue of the learned matrix is introduced and has been shown to be provably efficient to reduce the false negative rate. Our paper is supported by a theoretical study and an extensive experimental evaluation showing that **MLFP** outperforms state-of-the-art metric-learning methods.
This work opens the door to two promising lines of research. First, in **MLFP** we learn a linear projection of the data. One interesting perspective would consist in kernelizing our metric learning algorithm or designing a deep learning version allowing us to capture non linearity. A simpler solution might also consist in learning different local metrics for different regions of the input space as done in Zantedeschi *et al.* [2016]. Second, as initiated in Sharma *et al.* [2018], combining a Mahalanobis distance with a sampling strategy might lead to a new family of imbalanced learning methods.

Acknowledgements

This work was supported by the following projects: AURA project TADALoT (Pack Ambition 2017, 17 011047 01), ANR project LIVES (ANR-15-CE23-0026) and IDEXLYON project ACADEMICS (ANR-16-IDEX-0005).

References

Aisha Abdallah, Mohd Aizaini Maarof, and Anazida Zainal. Fraud detection system: A survey. *Journal of Network and Computer Applications*, 68:90–113, 2016.

Charu C. Aggarwal. *Outlier Analysis*. Springer International Publishing, 2017.

Aurélien Bellet, Amaury Habrard, and Marc Sebban. A survey on metric learning for feature vectors and structured data. *CoRR*, abs/1306.6709, 2013.

Aurélien Bellet, Amaury Habrard, and Marc Sebban. *Metric Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2015.

Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of machine learning research*, 2(Mar):499–526, 2002.

Jason V. Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S. Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th international conference on Machine learning*, pages 209–216. ACM, 2007.

Shuya Ding, Bilal Mirza, Zhiping Lin, Jiuwen Cao, Xiaoping Lai, Tam V. Nguyen, and Jose Sepulveda. Kernel based online learning for imbalance multiclass classification. *Neurocomputing*, 277:139–148, 2018.

Sri Harsha Dumpala, Rupayan Chakraborty, and Sunil Kumar Kopparapu. A novel data representation for effective learning in class imbalanced scenarios. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 2100–2106, 7 2018.

Lin Feng, Huibing Wang, Bo Jin, Haohao Li, Mingliang Xue, and Le Wang. Learning a distance metric by balancing kl-divergence for imbalanced datasets. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, PP:1–12, 01 2018.

Jordan Frery, Amaury Habrard, Marc Sebban, and Liyun He-Guelton. Non-linear gradient boosting for class-imbalance learning. In *2nd International Workshop on Learning with Imbalanced Domains: Theory and Applications*, pages 38–51, 2018.

Léo Gautheron, Emilie Morvant, Amaury Habrard, and Marc Sebban. Metric learning from imbalanced data. In *arXiv*. 1909.01651, 2019.

Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. Learning deep representation for imbalanced classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5375–5384, 2016.

Salman H. Khan, Munawar Hayat, Mohammed Bennamoun, Ferdous A. Sohel, and Roberto Togneri. Cost-sensitive learning of deep feature representations from imbalanced data. *IEEE transactions on neural networks and learning systems*, 29(8):3573–3587, 2017.

Brian Kulis et al. Metric learning: A survey. *Foundations and Trends® in Machine Learning*, 5(4):287–364, 2013.

Josey Mathew, Ming Luo, Chee Khiang Pang, and Hian Leng Chan. Kernel-based smote for svm classification of imbalanced datasets. In *IECON 2015-41st Annual Conference of the IEEE Industrial Electronics Society*, pages 001127–001132. IEEE, 2015.

María Pérez-Ortiz, Peter Tiňo, Rafal Mantiuk, and César Hervás-Martínez. Exploiting synthetically generated data with semi-supervised learning for small and imbalanced datasets. *CoRR*, abs/1903.10022, 2019.

Cornelis J. Van Rijsbergen. *Information Retrieval*. 1979.

Jörg Schiller. The impact of insurance fraud detection systems. *Journal of Risk and Insurance*, 73(3):421–438, 2006.

Shiven Sharma, Colin Bellinger, Bartosz Krawczyk, Osmar R. Zaïane, and Nathalie Japkowicz. Synthetic oversampling with the majority class: A new perspective on handling extreme imbalance. *IEEE International Conference on Data Mining*, pages 447–456, 2018.

Nan Wang, Xibin Zhao, Yu Jiang, Yue Gao, and KLISS BNRist. Iterative metric learning for imbalance data classification. In *27th International Joint Conference on Artificial Intelligence*, pages 2805–2811, 2018.

Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244, 2009.

Fei Wu, Xiao-Yuan Jing, Shiguang Shan, Wangmeng Zuo, and Jing-Yu Yang. Multiset feature learning for highly imbalanced data classification. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 1583–1589. AAAI Press, 2017.

Valentina Zantedeschi, Rémi Emonet, and Marc Sebban. Metric learning as convex combinations of local models with generalization guarantees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1478–1486, 2016.

Xiaogang Zhang, Dingxiang Wang, Yicong Zhou, Hua Chen, Fanyong Cheng, and Min Liu. Kernel modified optimal margin distribution machine for imbalanced data classification. *Pattern Recognition Letters*, 125:325–332, 2019.

– Supplementary Material – Learning from Few Positives: a Provably Accurate Metric Learning Algorithm to deal with Imbalanced Data

Rémi Viola[1,2], Rémi Emonet[1], Amaury Habrard[1], Guillaume Metzler[1], and Marc Sebban[1]

[1]University of Lyon, UJM-Saint-Etienne, CNRS,, Institute of Optics Graduate School, Laboratoire Hubert Curien UMR 5516, Saint-Etienne, France.
firstname.name@univ-st-etienne.fr
[2]DGFiP, Ministry of Economy and Finance, Paris, France.
remi.viola@dgfip.finances.gouv.fr

1 Introduction and Notations

We will denote by $\mathbf{z} = (\mathbf{x}, y)$ the couple features-label where $\mathbf{x} \in \mathbb{R}^d$ and $y \in \{-1, 1\}$ and $S = \{\mathbf{z}_i\}_{i=1}^m$ a set of m training examples drawn from an unknown distribution $\mathcal{D}$. We denote by m_+ the number of positives and m_- the number of negatives. Thus the rate of positives α is equal to $\frac{m_+}{m}$. Suppose that $\mathbf{x}'$ is a test instance, we recall that:

- $d_{\mathbf{M}} = d_{\mathbf{M}}(\mathbf{x}', \mathbf{x}) = \sqrt{(\mathbf{x} - \mathbf{x}')^T M (\mathbf{x} - \mathbf{x}')}$ if $\mathbf{x}$ is a positive instance,
- $d = d_{\mathbf{I}}(\mathbf{x}', \mathbf{x}) = d_(\mathbf{x}', \mathbf{x}) = \sqrt{(\mathbf{x} - \mathbf{x}')^T (\mathbf{x} - \mathbf{x}')}$ otherwise.

We are considering the following optimization problem:

$$\min_{\mathbf{M} \in \mathbb{S}^+} \quad \frac{1}{m^3} \left((1-\alpha) \sum_{\substack{(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) \\ y_i = y_j \neq y_k = -1}} \ell_{\mathrm{FN}}(\mathbf{M}, \mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_k) + \alpha \sum_{\substack{(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) \\ y_i = y_j \neq y_k = 1}} \ell_{\mathrm{FP}}(\mathbf{M}, \mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_k) \right) + \mu \|\mathbf{M} - \mathbf{I}\|_{\mathcal{F}}^2 \quad (1)$$

Our loss function can thus be seen as :

$$\ell(\mathbf{M}, (z_1, z_2, z_3)) = \begin{cases} (1-\alpha) \times \ell_{\mathrm{FN}}(\mathbf{M}, \mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3) & \text{if } y_i = y_j = 1,\ y_k = -1, \\ \alpha \times \ell_{\mathrm{FP}}(\mathbf{M}, \mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3)] & \text{if } y_i = y_j = -1,\ y_k = 1, , \\ 0 & \text{otherwise,} \end{cases}$$

where ℓ_{FN} and ℓ_{FP} are defined by:

- $\ell_{\mathrm{FN}}(\mathbf{M}, \mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_k) = [1 - c + d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j)^2 - d(\mathbf{x}_i, \mathbf{x}_k)^2]_+$,

- $\ell_{\mathrm{FP}}(\mathbf{M}, \mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_k) = [1 - c + d(\mathbf{x}_i, \mathbf{x}_j)^2 - d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_k)^2]_+$.

In the following, we will also suppose that for all $\mathbf{x}$ we have: $\|\mathbf{x}\|_2 \leq K$. Furthermore, we will denote by $\mathcal{R}_S$ and $\mathcal{R}$ respectively the empirical risk of ℓ over the training sample S and the true risk. More precisely, the empirical risk $\mathcal{R}_S$ is evaluated using a training set of size m which is used to build all the triplets and the true risk $\mathcal{R}$ is its expectation over all the samples of size m, i.e. $\mathcal{R} = \underset{S \sim \mathcal{D}^m}{\mathbb{E}} [\mathcal{R}_S]$.

- $d = d_{\mathbf{M}}(\mathbf{x}', \mathbf{x}) = \sqrt{(\mathbf{x} - \mathbf{x}')^T M (\mathbf{x} - \mathbf{x}')}$ if $\mathbf{x}$ is a positive instance,
- $d = d_{\mathbf{I}}(\mathbf{x}', \mathbf{x}) = \sqrt{(\mathbf{x} - \mathbf{x}')^T (\mathbf{x} - \mathbf{x}')}$.

In the following, we will also use the following constraint on $\mathbf{M}$:

$$\lambda_{\max}(\mathbf{M}) \leq 1, \text{ where } \lambda_{\max} \text{ is the largest eigenvalue of } \mathbf{M}.$$

Finally, due to the context of our study, i.e. imbalanced setting, $\alpha < 1/2$. Thus, $\alpha < 1 - \alpha$.

2 Generalization Guarantees

The aim of this section is to provide some generalization guarantees on our loss function according to the used loss function. Note that the following results give guarantees on the learned metric $\mathbf{M}$ which aims to find a good compromise between achieving a low rate of False Negatives while keeping a reasonable rate of False Positives.

2.1 Uniform Stability

In this section, we briefly restate the definition of stability and the generalization bound based on this notion.
Roughly speaking, an algorithm is *stable* if its output, in terms of difference between losses, does not change significantly under a small modification of the training sample. This variation must be bounded in $O(1/m)$ in terms of infinite norm where m is the size of the training set S *i.i.d.* from an unknown distribution $\mathcal{D}$.

Definition 1. *[Definition 6 (Bousquet and Elisseeff, 2002)] A learning algorithm $\mathcal{A}$ has a uniform stability in $\frac{\kappa}{m}$ with respect to a loss function ℓ and parameter set θ, with κ a positive constant if:*

$$\forall S,\ \forall i,\ 1 \leq i \leq m,\ \sup_Z |\ell(\theta_S, Z) - \ell(\theta_{S^i}, Z)| \leq \frac{\kappa}{m},$$

where S is a learning sample of size m, $Z = (\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3) = ((\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_3, y_3))$ is a triplet of labeled examples, θ_S the model parameters learned from S, θ_{S^i} the model parameters learned from the sample S^i obtained by replacing the i^{th} example z_i from S by another example z_i' independent from S and drawn from $\mathcal{D}$. $\ell(\theta_S, \mathbf{x})$ is the loss suffered at $\mathbf{x}$.

In this definition, S^i represents the notion of small modification of the training sample. The following one aims to study the evolution of the loss function according to the label of the considered triplet.

Definition 2. *A loss function ℓ is said to be γ-admissible, with respect to the distance metric $\mathbf{M}$ if (i) it is convex with respect to its first argument and (ii) the following condition holds:*

$$\forall Z, Z' \quad |\ell(\mathbf{M}, Z) - \ell(\mathbf{M}, Z')| \leq \gamma,$$

where $Z = (z_i, z_j, z_k)$ and $Z' = (z'_i, z'_j, z'_k)$ are two triplets of examples.

2.2 Preliminary Results

We now introduce the results we need to derive our generalization guarantees:

Proposition 1. *Let $X_1, ..., X_m$ be m independent random variables taking values in $\mathbb{R}$ and let $U = f(X_1, ..., X_m)$. If for each $1 \leq i \leq m$, there exists a constant c_i such that:*

$$\sup_{x_1, ...x_m \in \mathbb{R}} |f(x_1, ..., x_m) - f(x_1, ...x'_i, ..., x_m)| \leq c_i,$$

then for any positive constant B, we have:

$$\mathbb{P}[|U - \mathbb{E}[U]| \geq B] \leq 2 \exp\left(\frac{-2B^2}{\sum_{i=1}^m c_i^2}\right).$$

In the following, we set $D_S = \mathcal{R} - \mathcal{R}_S$. We then introduce the two following lemmas, for which the proof can be found in (Bellet et al., 2015) (see the proofs of Lemma 8.9 and 8.10 respectively). However, note that results have been adapted to our context, i.e. for triplet based loss function. But the proofs can be easily adapted.

Lemma 1. *For any learning method of estimation error D_S and satisfying a uniform stability in $\frac{\kappa}{m}$, we have $\mathbb{E}_S[D_S] \leq \frac{2\kappa}{m}$.*

Lemma 2. *For any parameter matrix $\mathbf{M}$ using m training examples, and any loss function ℓ satisfying the γ-admissibility, we have the following bound:*

$$\forall i,\ 1 \leq i \leq m,\ |D_S - D_{S^i}| \leq \frac{2\kappa}{m} + \frac{2\gamma}{m}.$$

Using the above Proposition and the two Lemmas, we are able to get the following generalization bound:

Theorem 1. *Let $\delta > 0$ and $m > 1$. Let S be a sample of m randomly selected training examples and let $\mathbf{M}$ be the learned parameter matrix from an algorithm with uniform stability $\frac{\kappa}{m}$. Assuming that the loss function ℓ is k-Lipschitz and γ-admissible and let us denote by $\mathcal{R}_S$ its empirical risk. With probability $1 - \delta$, we have the following bound on the true risk $\mathcal{R}$ of our loss function ℓ:*

$$\mathcal{R} \leq \mathcal{R}_S + 2\frac{\kappa}{m} + (2\kappa + 2\gamma)\sqrt{\frac{\ln(2/\delta)}{2m}}.$$

2.3 Generalization Bound

We first prove that our function is k-Lipschitz according to the following definition.

Definition 3. *A loss function ℓ is k-Lipschitz with respect to its first argument if for any parameters matrices $\mathbf{M}$ and $\mathbf{M}'$, and for any triplets of labeled examples $Z = (\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3)$, we have:*

$$|\ell(\mathbf{M}, Z) - \ell(\mathbf{M}', Z)| \leq k\|\mathbf{M} - \mathbf{M}'\|_{\mathcal{F}}.$$

Lemma 3. *We now show that our loss function ℓ is k-Lipschitz with $k = 4(1-\alpha)K^2$*

Proof. We need to study two cases, according to the label of the triplets.

Case 1: $y_i = y_j = 1,\ y_k = -1$

$$\begin{aligned}
|\ell(\mathbf{M}, Z) - \ell(\mathbf{M}', Z)| &= (1-\alpha)|[1 - c + d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j)^2 - d(\mathbf{x}_i, \mathbf{x}_k)^2]_+ - [1 - c + d_{\mathbf{M}'}(\mathbf{x}_i, \mathbf{x}_j)^2 - d(\mathbf{x}_i, \mathbf{x}_k)^2]_+|, \\
&\leq (1-\alpha)|d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j)^2 - d_{\mathbf{M}'}(\mathbf{x}_i, \mathbf{x}_j)^2|, \\
&= (1-\alpha)|(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{M} - \mathbf{M}')(\mathbf{x}_i - \mathbf{x}_j)|, \\
&= (1-\alpha)|\|\mathbf{x}_i - \mathbf{x}_j\|_2^2 \|\mathbf{M} - \mathbf{M}'\|_{\mathcal{F}}, \\
|\ell(\mathbf{M}, Z) - \ell(\mathbf{M}', Z)| &\leq 4(1-\alpha)K^2\|\mathbf{M} - \mathbf{M}'\|_{\mathcal{F}}
\end{aligned}$$

where the second line uses the fact that the hinge loss is 1-Lipschitz, the third line uses the linearity of the difference with respect to $\mathbf{M}, \mathbf{M}'$, the fourth line uses usual properties on norms and the last line the fact that $\|\mathbf{x}\| \leq K$.

Case 2: $y_i = y_j = -1,\ y_k = 1$
The proof is similar to the proof given in the previous case and leads to the following result:

$$|\ell(\mathbf{M}, Z) - \ell(\mathbf{M}', Z)| \leq 4\alpha K^2\|\mathbf{M} - \mathbf{M}'\|_{\mathcal{F}}.$$

We conclude by taking the maximum of the three previous values. Thus $k = 4(1-\alpha)K^2$ □

Now, we have to prove that our loss function is γ-admissible according to the definition 2.

Lemma 4. *The loss function ℓ defined by (1) is γ-admissible with respect to the distance metric $\mathbf{M}$, with $\gamma = (1-\alpha)(1 - c + 4K^2)$.*

Proof. Needless to say that the loss function ℓ is convex with respect to $\mathbf{M}$ as the sum of two convex functions. Indeed, both of them are linear w.r.t. $\mathbf{M}$ and the maximum of two convex functions remains convex.
Furthermore, because our loss function can be equal to zero for some labels of our triplets, we are looking for the greatest value than our loss function ℓ can achieve.

Using our previous result, we can bound the first part ell_{FN} by: $(1-\alpha)(1 - c + 4K^2)$ and the last term ell_{FP} by: $\alpha(1 - c + 4K^2)$.

Finally:

$$\forall\ Z, Z'\ |\ell(\mathbf{M}, Z) - \ell(\mathbf{M}, Z')| \leq \max((1-\alpha)(1 - c + 4K^2), \alpha(1 - c + 4K^2)).$$

Thus, $\gamma = (1-\alpha)(1 - c + 4K^2)$.

□

Definition 4. *A learning algorithm has a uniform stability in $\frac{\kappa}{m}$ where κ is a positive constant, if given any training set S we have:*

$$\forall i,\ \sup_Z |\ell(\mathbf{M}, Z) - \ell(\mathbf{M}^i, Z)| \leq \frac{\kappa}{m},$$

where M^i is the matrix learned with a training set S^i which differs from S of only one example $(\mathbf{x}_i \to \mathbf{x}'_i)$.

For the sake of clarity for the following development, let us denote by F_S the objective function to optimize over the training set S, i.e. $F_S = \frac{1}{m^3}\sum_{\mathbf{x}_i,\mathbf{x}_j,\mathbf{x}_k} \ell(\mathbf{M}, Z) + \mu\|\mathbf{M}-\mathbf{I}\|^2_{\mathcal{F}}$.
To compute the constant of uniform stability, we first need the following technical lemma:

Lemma 5. *Let S be a learning sample, let F_S and F_{S^i} be two objective functions with respect to two samples S and S^i and let $\mathbf{M}$ and $\mathbf{M}^i$ be their respective minimizers. We also define $\Delta\mathbf{M} = \mathbf{M}^i - \mathbf{M}$ and recall that $N(\mathbf{M}) = \mu\|\mathbf{M}-\mathbf{I}\|^2_{\mathcal{F}}$. For all $t \in [0,1]$, we have:*

$$N(\mathbf{M}) - N(\mathbf{M}+t\Delta\mathbf{M}) + N(\mathbf{M}^i) - N(\mathbf{M}^i - t\Delta\mathbf{M}) \\ \leq \frac{2t}{\mu m^3}[3m(m-1)+1] \times \left(4(1-\alpha)K^2\right) \times \|\Delta\mathbf{M}\|_{\mathcal{F}}.$$

Proof. Since ℓ (the hinge loss) is convex, so is the empirical risk and thus for all $t \in [0,1]$ we have the two following inequalities:

$$\mathcal{R}_{S^i}(\mathbf{M}+t\Delta\mathbf{M}) - \mathcal{R}_{S^i}(\mathbf{M},R) \leq t\mathcal{R}_{S^i}(\mathbf{M}^i) - t\mathcal{R}_{S^i}(\mathbf{M}).$$

and

$$\mathcal{R}_{S^i}(\mathbf{M}^i - t\Delta\mathbf{M}) - \mathcal{R}_{S^i}(\mathbf{M}^i) \leq t\mathcal{R}_{S^i}(\mathbf{M}) - t\mathcal{R}_{S^i}(\mathbf{M}^i).$$

We get the second inequality by swapping the role of $\mathbf{M}$ and $\mathbf{M}^i$. If we sum these two inequalities, the right hand side vanishes and we obtain:

$$\mathcal{R}_{S^i}(\mathbf{M}+t\Delta\mathbf{M}) - \mathcal{R}_{S^i}(\mathbf{M}) + \mathcal{R}_{S^i}(\mathbf{M}^i - t\Delta\mathbf{M}) - \mathcal{R}_{S^i}(\mathbf{M}^i) \leq 0. \tag{2}$$

By assumption on $\mathbf{M}$ and $\mathbf{M}^i$ we have:

$$\begin{aligned} F_S(\mathbf{M},R) - F_S(\mathbf{M}+t\Delta\mathbf{M}) &\leq 0, \\ F_{S^i}(\mathbf{M}^i) - F_{S^i}(\mathbf{M}^i - t\Delta\mathbf{M}) &\leq 0, \end{aligned}$$

then, summing the two previous inequalities and using (2), we get:

$$\mathcal{R}_{S^i}(\mathbf{M}+t\Delta\mathbf{M}) - \mathcal{R}_S(\mathbf{M}+t\Delta\mathbf{M}) - \mathcal{R}_{S^i}(\mathbf{M}) + \mathcal{R}_S(\mathbf{M}) \\ + \mu[\|\mathbf{M}-\mathbf{I}\|^2_F + \|\mathbf{M}^i-\mathbf{I}\|^2_F - \|\mathbf{M}+t\Delta\mathbf{M}-\mathbf{I}\|^2_F - \|\mathbf{M}^i - t\Delta\mathbf{M}-\mathbf{I}\|^2_F] \leq 0. \tag{3}$$

We now focus on the first part of the previous inequality. For the sake of simplicity, let us set:

$$H = \mathcal{R}_S(\mathbf{M}+t\Delta\mathbf{M}) - \mathcal{R}_{S^i}(\mathbf{M}+t\Delta\mathbf{M}) + \mathcal{R}_{S^i}(\mathbf{M}) - \mathcal{R}_S(\mathbf{M}).$$

$$\begin{aligned} H &\leq \left|\mathcal{R}_S(\mathbf{M}+t\Delta\mathbf{M}) - \mathcal{R}_{S^i}(\mathbf{M}+t\Delta\mathbf{M}) + \mathcal{R}_{S^i}(\mathbf{M}) - \mathcal{R}_S(\mathbf{M})\right|, \\ &\leq \frac{1}{m^3}\left| \sum_{z_i,z_j,z_k \in S^l} \ell(\mathbf{M}, z_i^l, z_j^l, z_k^l) - \sum_{z_i,z_j,z_k \in S} \ell(\mathbf{M}, z_i, z_j, z_k) \right. \\ &\qquad \left. \sum_{z_i,z_j,z_k \in S} \ell(\mathbf{M}+t\Delta\mathbf{M}, z_i, z_j, z_k) - \sum_{z_i,z_j,z_k \in S^l} \ell(\mathbf{M}+t\Delta\mathbf{M}, z_i^l, z_j^l, z_k^l)\right|, \end{aligned}$$

where S and S^l differ from the l-th example, i.e. $\forall i, j, k \neq l,\ z_i = z_i^l,\ z_j = z_j^l$ and $z_k = z_k^l$.

We will now focus on the first difference in the previous expression, i.e. on:

$$\sum_{z_i, z_j, z_k \in S^l} \ell(\mathbf{M}, z_i^l, z_j^l, z_k^l) - \sum_{z_i, z_j, z_k \in S} \ell(\mathbf{M}, z_i, z_j, z_k).$$

This difference can be decomposed into two parts according to the value of the index i: when $i = l$ and when $i \neq l$:

$$\begin{aligned}
&\sum_{j=1}^{m}\sum_{k=1}^{m}\Big(\ell(\mathbf{M}, z_l^l, z_j^l, z_k^l) - \ell(\mathbf{M}, z_l, z_j, z_k)\Big) \\
&+\sum_{i\neq l}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m}\Big(\ell(\mathbf{M}, z_i^l, z_j^l, z_k^l) - \ell(\mathbf{M}, z_i, z_j, z_k)\Big)
\end{aligned}$$

The first part of the decomposition is composed of m^2 terms that are at least not equal to zero. We, thus have to work on the second part of the decomposition has it contains some terms that are equal to zero. We will have to do this process two times as follows:

$$\begin{aligned}
&\sum_{j=1}^{m}\sum_{k=1}^{m}\Big(\ell(\mathbf{M}, z_l^l, z_j^l, z_k^l) - \ell(\mathbf{M}, z_l, z_j, z_k)\Big) \\
&+\sum_{i\neq l}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m}\Big(\ell(\mathbf{M}, z_i^l, z_j^l, z_k^l) - \ell(\mathbf{M}, z_i, z_j, z_k)\Big), \\
=\ &\sum_{j=1}^{m}\sum_{k=1}^{m}\Big(\ell(\mathbf{M}, z_l^l, z_j^l, z_k^l) - \ell(\mathbf{M}, z_l, z_j, z_k)\Big) \\
&+\sum_{i\neq l}^{m}\sum_{k=1}^{m}\Big(\ell(\mathbf{M}, z_i^l, z_l^l, z_k^l) - \ell(\mathbf{M}, z_i, z_l, z_k)\Big) \\
&+\sum_{i\neq l}^{m}\sum_{j\neq l}^{m}\sum_{k=1}^{m}\Big(\ell(\mathbf{M}, z_i^l, z_j^l, z_k^l) - \ell(\mathbf{M}, z_i, z_j, z_k)\Big), \\
=\ &\sum_{j=1}^{m}\sum_{k=1}^{m}\Big(\ell(\mathbf{M}, z_l^l, z_j^l, z_k^l) - \ell(\mathbf{M}, z_l, z_j, z_k)\Big) \\
&+\sum_{i\neq l}^{m}\sum_{k=1}^{m}\Big(\ell(\mathbf{M}, z_i^l, z_l^l, z_k^l) - \ell(\mathbf{M}, z_i, z_l, z_k)\Big) \\
&+\sum_{i\neq l}^{m}\sum_{j\neq l}^{m}\Big(\ell(\mathbf{M}, z_i^l, z_j^l, z_l^l) - \ell(\mathbf{M}, z_i, z_j, z_l)\Big) \\
&+\sum_{i\neq l}^{m}\sum_{j\neq l}^{m}\sum_{k\neq l}^{m}\underbrace{\Big(\ell(\mathbf{M}, z_i^l, z_j^l, z_k^l) - \ell(\mathbf{M}, z_i, z_j, z_k)\Big)}_{=0}.
\end{aligned}$$

All these sums are respectively composed of $m^2,\ m(m-1)$ and $(m-1)^2$ terms and the last $(m-1)^3$ terms are all equal to zero. Furthermore: $m^2 + m(m-1) + (m-1)^2 = 3m(m-1) + 1$, so that we

have to find a bound on the supremum of the difference:

$$[3m(m-1)+1] \sup_{Z,Z'} |\ell(\mathbf{M}, Z) - \ell(\mathbf{M}, Z') + \ell(\mathbf{M} + t\Delta\mathbf{M}, Z) - \ell(\mathbf{M} + t\Delta\mathbf{M}, Z')|.$$

Thus, H can be upper-bounded by:

$$H \leq \frac{1}{m^3}(m-1)+1] \, (\sup_{Z,Z'} |\ell(\mathbf{M}, Z) - \ell(\mathbf{M}+, Z') + \ell(\mathbf{M} + t\Delta\mathbf{M}, Z) - \ell(\mathbf{M} + t\Delta\mathbf{M}, Z')|).$$

We can then write:

$$\begin{aligned} H &\leq \frac{1}{m^3}(m-1)+1] \, (\sup_{Z} |\ell(\mathbf{M} + t\Delta\mathbf{M}, Z) - \ell(\mathbf{M}, Z)| + (\sup_{Z'} |\ell(\mathbf{M} + t\Delta\mathbf{M}, Z') - \ell(\mathbf{M}+, Z')|), \\ &\leq \frac{2t}{m^3}[3m(m-1)+1] \times \|\Delta\mathbf{M}\|_{\mathcal{F}} \times \left(4(1-\alpha)K^2\right), \end{aligned}$$

where the last lines uses Lemma 3 and properties on norms. Finally, we have :

$$N(\mathbf{M}) - N(\mathbf{M}+t\Delta\mathbf{M}) + N(\mathbf{M}^i) - N(\mathbf{M}^i - t\Delta\mathbf{M}) \leq \frac{2t}{\mu m^3}[3m(m-1)+1] \times \left(4(1-\alpha)K^2\right) \times \|\Delta\mathbf{M}\|_{\mathcal{F}} \tag{4}$$

□

We are now able to prove the uniform stability of our algorithm.

Theorem 2. *Let S be a learning sample of size m, the algorithm (1) has a uniform stability in $\frac{\kappa}{m}$ with $\kappa = \frac{6}{\mu} \times \left(4(1-\alpha)K^2\right)^2$.*

Proof. Let us set $t = \frac{1}{2}$ in the result of Lemma 5 and we focus on the left hand side of this result. We have:

$$\begin{aligned} f(\mathbf{M}) &= \|\mathbf{M} - \mathbf{I}\|_F^2 + \|\mathbf{M}^i - \mathbf{I}\|_F^2 - \|\frac{1}{2}(\mathbf{M} + \mathbf{M}^i) - \mathbf{I}\|_F^2 - \|\frac{1}{2}(\mathbf{M} + \mathbf{M}^i) - \mathbf{I}\|_F^2, \\ &= \|\mathbf{M} - \mathbf{I}\|_F^2 + \|\mathbf{M}^i - \mathbf{I}\|_F^2 - \frac{1}{2}\|\mathbf{M} + \mathbf{M}^i - \mathbf{I}\|_F^2, \\ f(\mathbf{M}) &= \frac{1}{2}\|\mathbf{M} - \mathbf{M}^i\|_F^2. \end{aligned}$$

Then, using Lemma 5, we get the following bound on $\|\Delta\mathbf{M}\|_{\mathcal{F}}$.

$$\begin{aligned} \|\Delta\mathbf{M}\|_{\mathcal{F}}^2 &\leq \frac{8}{\mu m^3}[3m(m-1)+1] \times \left((1-\alpha)K^2\right) \times \|\Delta\mathbf{M}\|_{\mathcal{F}}, \\ \|\Delta\mathbf{M}\|_{\mathcal{F}} &\leq \frac{8}{\mu m^3}[3m(m-1)+1] \times \left((1-\alpha)K^2\right). \end{aligned}$$

To prove the uniform stability of our algorithm, it remains to find the value κ such that:

$$\forall S, \ \forall i, \ 1 \leq i \leq m, \ \sup_{Z} |\ell(\mathbf{M}, Z) - \ell(\mathbf{M}^i, Z)| \leq \frac{\kappa}{m}.$$

To do this, we use the fact that our loss function ℓ is k-Lipsichtz with $k = \left(4(1-\alpha)K^2\right)$ and our upper-bound on $\|\Delta\mathbf{M}\|_{\mathcal{F}}$. It gives:

$$\begin{aligned} |\ell(\mathbf{M}, Z) - \ell(\mathbf{M}^i, Z)| &\leq k\|\Delta\mathbf{M}\|_{\mathcal{F}}, \\ &\leq \frac{2k^2(3m^2-3m+1)}{\mu m^3}. \end{aligned}$$

Finally:

$$\forall S,\ \forall i,\ 1 \leq i \leq m,\ \sup_Z |\ell(\mathbf{M}, Z) - \ell(\mathbf{M}^i, Z)| \leq \frac{\kappa}{m^3},$$

with $\kappa = \dfrac{4(3m^2-3m+1)}{\mu} \times \left((1-\alpha)K^2\right)^2$. □

For the sake of simplicity, we will simplify this result in the following. Note that for all $m \geq 1$, $\dfrac{3m^2-3m+1}{m^3} \leq \dfrac{3}{m}$. Thus, our algorithm has a uniform stability in $\dfrac{\kappa}{m}$ with $\kappa = \dfrac{12}{\mu} \times \left((1-\alpha)K^2\right)^2$. We can now apply Theorem 1 to our algorithm and get the following result:

Theorem 3. *Let $\delta > 0$ and $m > 1$. With probability $1-\delta$, we have the following bound on the true risk $\mathcal{R}$ of our loss function ℓ:*

$$\mathcal{R} \leq \mathcal{R}_S + 2\frac{\kappa}{m} + (2\kappa + 2\gamma)\sqrt{\frac{\ln(2/\delta)}{2m}},$$

with:

$$\kappa = \frac{12}{\mu} \times \left((1-\alpha)K^2\right)^2.$$

and

$$\gamma = (1-\alpha)(1-c+4K^2).$$

Proof. The proof is consequence of Theorem 1 and Lemma 4. □

3 Classification Guarantees - Proof

We now give a proof of the Theorem 3 provided in the paper.

Proof. We first begin with the FP rate. We can note the hinge loss can be a surrogate for the indicator function as follows:

$$\mathbb{1}_{\{d_{\mathbf{M}}(\mathbf{x},\mathbf{x}_{p_i}) \leq d(\mathbf{x},\mathbf{x}_n)\}} = \mathbb{1}_{\{d_{\mathbf{M}}(\mathbf{x},\mathbf{x}_{p_i})^2 \leq d(\mathbf{x},\mathbf{x}_n)^2\}} \leq \left[1 + d(\mathbf{x},\mathbf{x}_n)^2 - d_{\mathbf{M}}(\mathbf{x}_i,\mathbf{x}_p)^2\right]_+,$$

We can recognize one of the term of our optimization Problem (1) with the hyper-parameter $c = 0$.

We recall that each labeled example is denoted as $\mathbf{z} = (\mathbf{x}, y)$. Then, we have:

$$\begin{aligned}
\mathcal{R}_{FP} &\leq \mathbb{E}_{S\sim D^m}\mathbb{E}_{\mathbf{z}\sim D}\left[1 + d(\mathbf{x}, \mathbf{x}_n)^2 - d_{\mathbf{M}}(\mathbf{x}, \mathbf{x}_p)^2\right]_+ \times \mathbb{1}_{\{y=-1\}} \\
&\leq \mathbb{E}_{S'\sim D^{m+1}}\mathbb{E}_{\mathbf{z}_i,\mathbf{z}_j,\mathbf{z}_k\in S'}\left[1 + d(\mathbf{x}_i, \mathbf{x}_j)^2 - d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_k)^2\right]_+, \times \mathbb{1}_{\{y_i=y_j=-1\neq y_k\}} \\
&\leq \mathbb{E}_{S'\sim D^{m+1}}\mathbb{E}_{\mathbf{z}_i,\mathbf{z}_j,\mathbf{z}_k\in S'}\left[\frac{\alpha}{\alpha}\left[1 + d(\mathbf{x}_i, \mathbf{x}_j)^2 - d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_k)^2\right]_+ \times \mathbb{1}_{\{y_i=y_j=-1\neq y_k\}} + \right. \\
&\quad \left. \frac{1-\alpha}{\alpha}\left(\left[1 + d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j)^2 - d(\mathbf{x}_i, \mathbf{x}_k)^2\right]_+ \times \mathbb{1}_{\{y_i=y_j=1\neq y_k\}}\right)\right], \\
&\leq \mathbb{E}_{S'\sim D^{m+1}}\mathbb{E}_{\mathbf{z}_i,\mathbf{z}_j,\mathbf{z}_k\in S'}\left[\frac{1}{\alpha}\left(\alpha\left[1 + d(\mathbf{x}_i, \mathbf{x}_j)^2 - d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_k)^2\right]_+ \times \mathbb{1}_{\{y_i=y_j=-1\neq y_k\}} + \right.\right. \\
&\quad \left.\left. (1-\alpha)\left[1 + d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j)^2 - d(\mathbf{x}_i, \mathbf{x}_k)^2\right]_+ \times \mathbb{1}_{\{y_i=y_j=1\neq y_k\}}\right)\right], \\
&\leq \frac{1}{\alpha}\mathcal{R}.
\end{aligned}$$

The second inequality is obtained by the i.i.d. aspect of the expectation. The third inequality is due to the fact that the second term in the sum is positive. Finally, one can note that the right-hand side of the last inequality corresponds to a weighted version of the true risk with respect to the loss used in Problem (1) with $c = 0$ and where we take an expectation over all the samples of size $m + 1$. The result is obtained by combining the results of Theorems 3 and 1 over the true risk defined above.

The bound for the false negative can be obtained in a similar way. Using the same arguments, one can show that:

$$\mathcal{R}_{FN} \leq \frac{1}{1-\alpha}\mathcal{R}.$$

Applying Theorems 3 and 1 to the above risk leads to the result. □

4 Other Results

In this section, we provide extended experiments using a Nearest-Neighbor classifier and other experiments on real fraud detection datasets provided by the French Ministry for the Economy and Finance.

4.1 Results for 3NN Without the Constraint

We present the results obtained with **MLFP** when we do not add the constraint on $\lambda_{\max}$. The results are shown in Table 1.
Adding the constraint leads to a small impact on the F-measure on average, but adding this constraint remains important if we want to increase the capacity of the model to capture the class of interest.

4.2 Results for 1NN

In Table 2, we provide the results obtained using a Nearest-Neighbor classifier and we also show that results of **MLFP** when the constrain is not added.
The results show that the proposed approach gives better results than its competitors even if $k = 1$. Also in with a 1-NN, we show that adding the constraint on $\lambda_{\max}(\mathbf{M})$ has a small impact on the F-measure of the method. It slightly increases the value of the F-measure.

Table 1: Results for MLFP with or without the constraint $\lambda_{max} \leq 1$, with 3−NN as final classifier, on public datasets sorted by imbalance ratio. The best results are in bold.

Datasets	MLFP + Cons	MLFP - Cons
balance	**0.954**	0.953
autompg	**0.827**	0.810
ionosphère	0.891	**0.928**
pima	**0.537**	0.524
wine	0.851	**0.884**
glass	0.718	**0.728**
german	**0.474**	0.417
vehicle	**0.886**	0.885
hayes	**0.565**	0.557
segmentation	**0.861**	0.853
abalone8	**0.345**	0.302
yeast3	**0.681**	0.655
pageblocks	0.842	0.842
libras	0.662	**0.722**
wine4	0.093	**0.095**
yeast6	**0.538**	0.482
abalone17	0.071	**0.113**
abalone20	0.062	**0.081**
Mean	**0.603**	0.602

4.3 Results on Private Datasets

This section provides the results (see Table 4) obtained on eight real fraud detection datasets provided by the General Directorate of Public Finances (DGFiP) which is part of the French central public administration related to the French Ministry for the Economy and Finance. These private datasets correspond to data coming from tax and VAT declarations of French companies and are used for tax fraud detection purpose covering declaration of over-valued, fictitious or prohibited charges, wrong turnover reduction or particular international VAT frauds such as "VAT carousels" and is described in Table 3.
The results are obtained using a 3-NN classifier.
The reported results are obtained using a 3-NN classifier. We note, that, on most of the datasets (7/8), **MLPF** reaches the highest performances in terms of F-measure, showing that the method is also interesting for real applications.

References

Bellet, A., Habrard, A., and Sebban, M. (2015). *Metric Learning*. Morgan & Claypool Publishers.

Bousquet, O. and Elisseeff, A. (2002). Stability and generalization. *Journal of machine learning research*, 2(Mar):499–526.

Table 2: Results for the different Metric Learning methods, with 1−NN as final classifier, on public datasets sorted by imbalance ratio. The best ML results are in bold.

Datasets	1NN	LMNN	ITML	IML	ImbML	MLFP + Cons	MLFP − Cons
balance	0.945	0.977	**0.986**	0.981	0.979	0.950	0.951
autompg	0.815	0.818	0.813	0.800	0.827	**0.834**	0.815
ionosphère	0.828	0.862	0.805	0.860	**0.902**	0.875	0.884
pima	0.415	0.413	0.475	0.435	0.449	0.533	**0.535**
wine	0.893	0.901	0.878	**0.902**	0.883	0.893	0.871
glass	0.745	0.733	0.770	0.709	0.722	**0.775**	0.740
german	0.354	0.341	0.391	0.331	0.349	0.454	**0.494**
vehicle	0.900	**0.953**	0.875	0.948	0.930	0.898	0.900
hayes	0.089	0.248	0.427	0.074	0.387	0.668	**0.728**
segmentation	0.881	0.900	0.873	**0.910**	0.906	0.881	0.881
abalone8	0.235	0.227	0.224	0.229	0.241	**0.297**	0.283
yeast3	0.622	0.621	0.643	0.639	**0.666**	0.630	0.621
pageblocks	0.847	0.842	**0.848**	0.833	0.846	**0.848**	**0.848**
libras	**0.803**	0.683	0.731	0.692	0.720	0.762	0.720
wine4	0.086	0.040	0.017	0.056	0.044	**0.137**	0.078
yeast6	**0.506**	0.460	0.460	0.354	0.489	0.470	**0.506**
abalone17	**0.212**	0.158	0.087	0.105	0.186	0.182	0.173
abalone20	0.000	0.000	**0.031**	0.000	0.000	0.011	**0.031**
Mean	0.565	0.565	0.574	0.548	0.585	**0.617**	0.614
Average Rank	3.9	4.7	4.3	4.9	3.4	**2.6**	3

Table 3: Information about the studied datasets sorted by imbalance ratio. The first part refers to the public datasets, the second one describes the *DGFiP* private datasets.

datasets	size	dim	%+	%−	IR
dgfip 9 2	440	173	24.8	75.2	3
dgfip 4 2	255	82	20.8	79.2	3.8
dgfip 8 1	1028	255	17.8	82.2	4.6
dgfip 8 2	1031	254	17.9	82.1	4.6
dgfip 9 1	409	171	16.4	83.6	5.1
dgfip 4 1	240	76	16.2	83.8	5.2
dgfip 16 1	789	162	10.3	89.7	8.7
dgfip 16 2	786	164	9.9	90.1	9.1

Table 4: Results for the different Metric Learning methods, with 3−NN as final classifier, on private datasets sorted by imbalance ratio. The best ML results are in bold.

Datasets	IR	kNN	LMNN	ITML	IML	ImbML	MLFP
DGFiP9_2	3	0.173	0.152	0.119	0.225	0.204	**0.400**
DGFiP4_2	3.8	0.164	0.241	0.141	0.186	0.098	**0.359**
DGFiP8_1	4.6	0.100	0.082	0.098	0.104	0.097	**0.305**
DGFiP8_2	4.6	0.140	0.220	0.122	0.254	0.240	**0.304**
DGFiP9_1	5.1	0.088	0.174	0.113	0.142	0.131	**0.291**
DGFiP4_1	5.2	0.073	0.130	0.097	0.120	0.067	**0.256**
DGFiP16_1	8.7	0.049	0.046	0.071	0.057	0.106	**0.192**
DGFiP16_2	9.1	**0.210**	0.142	0.176	0.172	0.153	0.199