Normalizing Flows pour éviter le problème de pré-image

Clément GLÉDEL[1] Benoît GAÜZÈRE[1] Paul HONEINE[1]

[1]Univ Rouen Normandie, INSA Rouen Normandie, Normandie, Normandie Univ,
LITIS UR 4108, F-76000 Rouen, France

Résumé – Les méthodes d'apprentissage statistique génèrent souvent un espace de représentation dans lequel la tâche de classification ou de régression peut être effectuée efficacement. Alors qu'une telle transformation des données est en général non-inversible, il peut être intéressant de faire le chemin inverse afin d'interpréter les résultats dans l'espace des observations. Ce problème, dit de pré-image, est un problème difficile, mal posé, notamment quand les transformations sont implicites (méthodes à noyaux) ou non linéaires multiples (réseaux de neurones profonds). Dans cet article, nous proposons une méthode de classification ne souffrant pas de ce problème permettant ainsi une meilleure interprétabilité des représentations. La méthode proposée repose sur une nouvelle famille de méthodes génératives que sont les Normalizing Flows. Les expérimentations montrent de bons résultats de classification et d'interprétabilité.

Abstract – Statistical learning methods often embed the data in a latent space where the classification or regression task can be efficiently carried out. For several reasons such as interpretability, it is interesting to represent the results back into the input space. This so-called pre-image problem is a hard ill-posed problem, as one seeks to inverse implicit transformations in kernel-based machines, or multiple nonlinear embeddings in deep neural networks. In this paper, we propose a classification method that does not suffer from the pre-image problem, thanks to a novel class of generative neural networks called Normalizing Flows. Experiments show good results in classification and interpretability.

1 Introduction

En apprentissage statistique, les transformations non linéaires consistant à représenter les données dans un espace pertinent (appelé espace latent) sont essentielles. La transformation non linéaire peut être implicite, comme dans les méthodes à noyaux, ou explicite en combinant des opérations non linéaires dans les réseaux neuronaux profonds. Pour pouvoir interpréter et comprendre ces transformations, il peut être intéressant d'estimer la transformation inverse afin de représenter les résultats dans l'espace des observations. Il s'agit du problème dit de pré-image. Différents algorithmes ont été proposés pour la résolution du problème de pré-image [5]. De récents travaux ont montré l'intérêt de résoudre le problème de pré-image pour l'interprétation en analyse de séries temporelles [16] ou de données graphes [6], ou encore en apprentissage de dictionnaires non linéaires [18, 19].

Dans cet article, nous proposons une nouvelle classe de méthodes de classification qui évitent la malédiction de la pré-image. L'idée principale est d'apprendre une fonction de transformation non linéaire qui est inversible par conception ; des pré-images peuvent ainsi être générées efficacement en appliquant la transformation inverse sur n'importe quel point d'intérêt de l'espace latent. Notre transformation non linéaire réversible s'appuie sur les progrès récents des modèles génératifs dans le domaine de l'apprentissage statistique, et en particulier les Normalizing Flows (NF) [9, 12]. Un NF génère un espace latent tout en garantissant l'inversibilité de la transformation, grâce à des relations exactes entre les deux distributions des données, dans l'espace d'observation et celui généré. Alors que les NF ont été proposés initialement pour générer un espace latent à distribution gaussienne, nous proposons d'apprendre cet espace latent afin que les échantillons des différentes classes soient linéairement séparables. Nous démontrons l'efficacité de notre méthode sur 4 familles de NF différentes : RealNVP basé sur des couches de couplage [3], FFJORD sur NF continu [4], Glow sur des convolutions pour les images [8] et MoFlow sur des données graphes [17]. Les résultats expérimentaux sur des jeux de données bien connus montrent la pertinence de notre méthode pour traiter les tâches de classifications tout en générant un bon espace de représentation exempt du problème de la pré-image.

Le reste de l'article est organisé comme suit. La section suivante introduit succinctement les modèles génératifs NF. La section 3 présente la méthode proposée dans le cadre général, avant de l'étendre aux données graphes. La section 4 présente les résultats expérimentaux sur des jeux de données de référence. La section 5 conclut l'article.

2 Préliminaires sur les NF

Les récentes avancées en modèles génératifs incluent les auto-encodeurs variationnels (VAE), les réseaux génératifs antagonistes (GAN) et plus récemment les NF. Les NF reposent sur la formule de changement de variable, qui définit la relation entre les deux densités de probabilité dans les espaces d'observation $\mathcal{X}$ et latent $\mathcal{Z}$:

$$P_{\mathcal{X}}(x) = P_{\mathcal{Z}}(z) \left|\det \left(\frac{\partial z}{\partial x}\right)\right| = P_{\mathcal{Z}}(\Phi(x)) \left|\det \left(\frac{\partial \Phi(x)}{\partial x}\right)\right|, \tag{1}$$

où le déterminant de la jacobienne de la transformation Φ représente le facteur de déformation entre les deux distributions. En considérant la fonction log-vraisemblance à maximiser, on obtient pour N échantillons $\mathbf{x}_1, \ldots, \mathbf{x}_N$:

$$\log P_{\mathcal{X}}(X) = \sum_{i=1}^{N} \log P_{\mathcal{Z}}(\Phi(\mathbf{x}_i)) + \log \left|\det \left(\frac{\partial \Phi(\mathbf{x}_i)}{\partial \mathbf{x}_i}\right)\right|. \tag{2}$$

L'optimisation qui en découle de la vraisemblance est exacte, ce qui diffère des approximations réalisées par les VAE où une borne inférieure est optimisée.

Différentes stratégies ont été proposées dans la littérature pour dériver des architectures de NF, en définissant la transformation Φ de $\mathcal{X}$ de $\mathcal{Z}$ par la composition de ℓ fonctions élémentaires bijectives

$$\Phi = \Phi_\ell \circ \Phi_{\ell-1} \circ \cdots \circ \Phi_2 \circ \Phi_1,$$

permettant ainsi de calculer le déterminant de la jacobienne de Φ à partir du produit de ceux des Φ_i. Les principales familles de NF reposent sur des couches de couplage, comme RealNVP [3], ou sur un NF continu, comme FFJORD [4]. Pour le traitement d'images, une approche par convolution est proposée par Glow [8]. Finalement, sur des données graphes, la stratégie proposée par MoFlow [17] repose sur des couches de couplage affine appliquées à la fois sur la matrice de caractéristiques des nœuds et sur la matrice d'adjacence, après vectorisation.

3 Méthode proposée

Cette section présente la double contribution du présent article. D'une part, nous revisitons les NF, initialement développés pour transformer des données évoluant selon une distribution complexe en une distribution gaussienne, facilitant ainsi le processus génératif de nouvelles données. Toutefois, une telle distribution n'est pas adaptée à une tâche de classification. Nous proposons donc d'apprendre un espace latent où les données sont linéairement séparables selon leurs classes. D'autre part, nous utiliserons la fonction inverse associée au modèle NF afin de générer la pré-image de tout point de l'espace latent où le modèle prédictif opère. Ces deux axes sont décrits dans la suite. La Figure 1 résume les différents blocs de la méthode présentée dans cet article.

Considérons un jeu de données $\{(\mathbf{x}_1, y_1) \dots (\mathbf{x}_N, y_N)\}$ avec $\mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^d$ les données d'entraînement et $y_i \in \mathcal{Y} = \{1 \dots C\}$ leurs classes. Nous associons à chaque classe $c \in \mathcal{Y}$ une distribution gaussienne dans l'espace latent $\mathcal{Z}$ définie par

$$P_{\mathcal{Z}}(\mathbf{z}, c) = \frac{1}{\sqrt{(2\pi)^d \det(\Sigma_c)}} e^{-\frac{1}{2}(\mathbf{z}-\boldsymbol{\mu}_c)^\top \Sigma_c^{-1}(\mathbf{z}-\boldsymbol{\mu}_c)},$$

de matrice de covariance Σ_c et de moyenne $\boldsymbol{\mu}_c$. La log-vraisemblance à utiliser dans (2) est alors définie par

$$\log P_{\mathcal{Z}}(\mathbf{z}, c) = -\frac{1}{2}\left(d\log(2\pi) + (\mathbf{z}-\boldsymbol{\mu}_c)^\top \Sigma_c^{-1}(\mathbf{z}-\boldsymbol{\mu}_c)\right) - \log(\det(\Sigma_c)). \quad (3)$$

Dans la suite, nous utiliserons ces C distributions, qu'il faudrait organiser dans l'espace latent, afin d'apprendre un NF qui sépare les différentes classes. Ainsi, le problème d'optimisation est composé de deux termes

$$\mathcal{L}(\mathbf{x}, y) = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}_{nf}(\mathbf{x}_i, y_i) - \beta \mathcal{L}_\mu, \quad (4)$$

avec β est un hyperparamètre de compromis entre ces deux termes : le premier correspond à la fonction de log-vraisemblance associée à la définition des distributions selon les NF, avec

$$\mathcal{L}_{nf}(\mathbf{x}, y) = \log P_{\mathcal{Z}}(\Phi(\mathbf{x}), y) + \log \left| \det \left(\frac{\partial \Phi(\mathbf{x})}{\partial \mathbf{x}} \right) \right|,$$

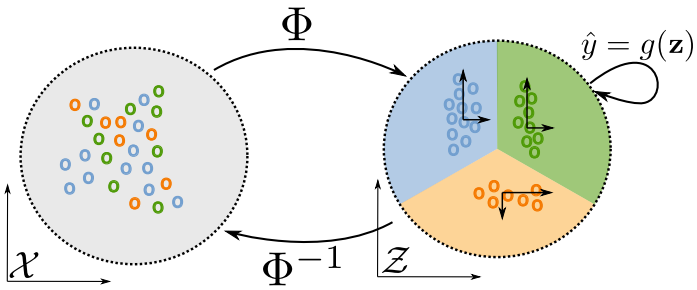

FIGURE 1 : Illustration de notre méthode. L'entraînement du NF vise à séparer linéairement les classes dans l'espace latent $\mathcal{Z}$. Un classifieur linéaire $g : \mathcal{Z} \to \mathcal{Y}$ déduit la classe pour chaque échantillon. La pré-image de tout point $\mathbf{z} \in \mathcal{Z}$ peut être retrouvée grâce à Φ^{-1}.

où $\log P_{\mathcal{Z}}(\Phi(\mathbf{x}), y)$ fait référence à (3), et le second terme est celui de séparation des centres des gaussiennes, avec

$$\mathcal{L}_\mu = \log\left(1 + \frac{1}{C^2}\sum_{i=1}^{C}\sum_{j=1}^{C} \| \boldsymbol{\mu}_i - \boldsymbol{\mu}_j \|_2\right).$$

En plus de différencier les gaussiennes par leurs moyennes, nous associons différentes dimensions de $\mathcal{Z}$ à différentes classes, grâce aux C matrices de covariance. Pour ce faire, nous répartissons les dimensions de $\mathcal{Z}$ en C classes, en associant $k = \frac{d}{C}$ valeurs propres à une valeur élevée, selon

$$(\underbrace{\lambda_1, \lambda_2, \dots, \lambda_k}_{c=1}, \underbrace{\lambda_{k+1}, \lambda_{k+2}, \dots, \lambda_{2k}}_{c=2}, \dots, \lambda_d) \quad (5)$$

Les $d-k$ valeurs propres restantes, pour chaque classe, sont fixées à $\epsilon = \sqrt[d-k]{1/\lambda_1^k}$. Ainsi, chaque classe possède sa propre matrice de covariance Σ_c construite selon

$$\Sigma_c = \sum_{i=(c-1)k+1}^{ck} \lambda_i \mathbf{v}_i^\top \mathbf{v}_i, \quad (6)$$

où $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_d$ forment la base canonique de l'espace.

Ce processus conduit à un espace latent $\mathcal{Z}$ où les classes sont linéairement séparables. Nous pouvons alors définir un modèle prédictif $f : \mathcal{X} \to \mathcal{Y}$ en apprenant un classificateur linéaire $g : \mathcal{Z} \to \mathcal{Y}$ comme les machines à vecteurs de support (SVM). L'inférence est ensuite réalisée par la combinaison de Φ, qui plonge les données dans l'espace $\mathcal{Z}$, et de g pour dériver une classe prédite, c'est à dire $f(\mathbf{x}) = g(\Phi(\mathbf{x}))$. Grâce à l'existence de Φ^{-1}, nous pouvons obtenir la pré-image de tout point d'intérêt de l'espace latent, fournissant ainsi un algorithme de classification sans la malédiction de la pré-image.

Extension aux données graphes

Dans le cas de données graphes, chaque graphe est représenté par n nœuds de d dimensions et un ensemble d'arêtes entre les nœuds caractérisés par e dimensions. Ces informations peuvent être représentées dans la matrice de caractéristiques des nœuds $\mathbf{X} \in \mathbb{R}^{n \times d}$ et le tenseur d'adjacence $\mathbf{A} \in \mathbb{R}^{n \times n \times e}$. Sans perte de généralité, nous appliquons (5) à la fois aux représentations latentes associées à $\mathbf{X}$ et à $\mathbf{A}$ après les avoir vectorisé, selon la même stratégie proposée par MoFlow [17]. Nous obtenons ainsi un espace latent biparti de dimension $D = n^2 \times e + n \times d$.

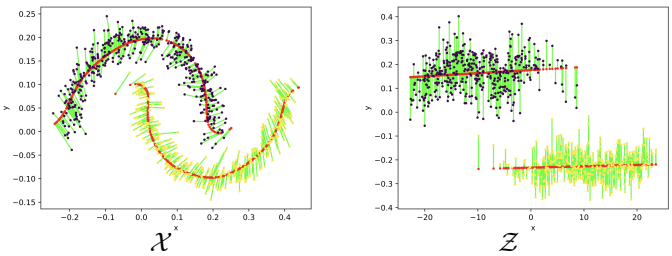

FIGURE 2 : Les données *double lune* dans l'espace d'entrée $\mathcal{X}$ et leurs représentations dans l'espace latent $\mathcal{Z}$ obtenus par la transformation apprise Φ. Les points rouges dans $\mathcal{Z}$ sont obtenues par une simple projection sur la première composante principale de chaque distribution, et les pré-images de ces points sont données par les points rouges dans $\mathcal{X}$. Le lien entre chaque échantillon et sa projection est en vert.

TABLE 1 : Performances en classification obtenues par notre méthode basée sur différent NF (RealNVP, FFJORD, et Glow) en comparaison aux SVM utilisant les noyaux RBF, Polynomial et Sigmoid. Les meilleurs résultats sont mis en évidence en gras.

Datasets	SVM RBF	SVM Poly	SVM Sigmoid	Méthode proposée RealNVP	Méthode proposée FFJORD	Méthode proposée Glow
Double lune	**100,00%**	93, 00%	90, 00%	**100,00%**	**100,00%**	-
Iris	93, 34%	93, 34%	80, 00%	**100,00%**	**100,00%**	-
Breast Cancer	94, 64%	94, 64%	96, 43%	**100,00%**	98, 21%	-
MNIST	97.31%	97.68%	90.98%	-	-	**99.47%**

4 Expérimentations

Nous évaluons les performances de la méthode proposée sur une multitude de jeux de données de différentes natures. Mais avant, nous démontrons l'intérêt et la pertinence de la pré-image.

Illustration de la pré-image

La Figure 2 illustre la méthode proposée sur un jeu de données *double lune* comportant 1000 échantillons en 2D répartis selon 2 lunes. En modifiant FFJORD comme décrit dans le présent article, nous avons obtenu l'espace latent permettant une séparabilité linéaire des deux classes. La pré-image a permis d'interpréter les résultats dans l'espace des observations ; ainsi, les axes principaux linéaires dans $\mathcal{Z}$ deviennent non linéaires dans $\mathcal{X}$.

Afin d'illustrer la pré-image sur des données de type image, nous avons utilisé les jeux de données MNIST et Olivetti. En entraînant le modèle avec le NF Glow [8], particulièrement adapté aux images, nous avons généré des pré-images de points arbitraires de $\mathcal{Z}$. Les 10 premières colonnes de la Figure 3 montrent la pré-image sur MNIST de 10 échantillonnages aléatoires appliqués à chaque distribution gaussienne paramétrée par Σ_c et $\boldsymbol{\mu}_c$, en fonction de la classe. Aussi, les pré-images de chaque prototype de classe sont présentées dans la dernière colonne. Sur le jeu de données Olivetti, la Figure 4 illustre la génération de visage à partir des pré-images de points d'interpolation spécifiques entre deux représentations de visages dans l'espace latent $\mathcal{Z}$. Ces 2 figures nous permettent de mettre en évidence la cohérence de l'espace latent et de la génération par pré-image.

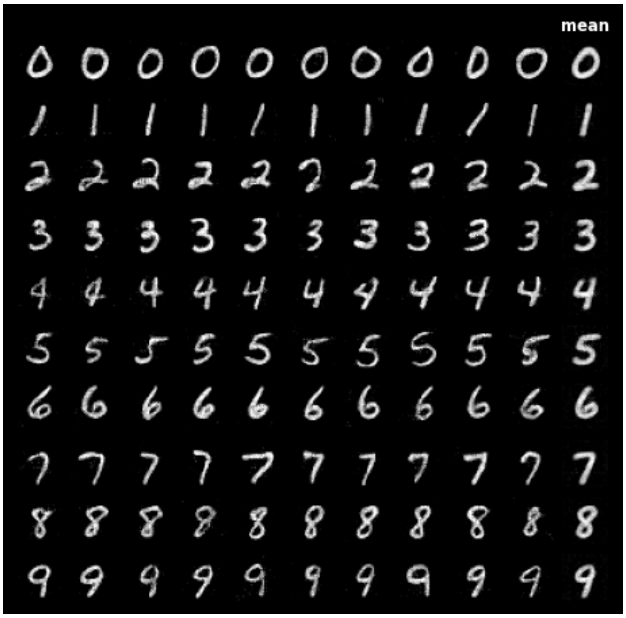

FIGURE 3 : Pré-images obtenues en échantillonnant selon chaque distribution gaussienne associée à chaque classe.

Performances en classification

Les performances du classifieur ont été étudiées sur 4 jeux de données : double lune, Iris, Breast cancer [15] et MNIST [2]. Nous avons utilisé RealNVP et FFJORD pour les données vectorielles, et Glow pour les images de MNIST. Les résultats en classification présentés dans la Table 1 montrent la supériorité en termes de performance de prédiction de notre méthode comparée aux SVM avec différents noyaux non linéaires[1].

Performances sur des données graphes

Pour déterminer la pertinence de notre méthode sur des données graphes, nous avons considéré les jeux de données MUTAG et Letter-med. MUTAG contient 188 molécules chimiques, chacune représentée par un graphe non orienté, étiqueté selon les 7 types d'atomes disponibles, avec jusqu'à 28 atomes par molécule et 3 différents types de liaisons. Letter-med [13] est composé de 2300 graphes représentant les 15 lettres majuscules A, E, F, H, I, K, L, M, N, T, V, W, X, Y, Z. Chaque graphe contient jusqu'à 9 nœuds où chacun a 2 attributs numériques qui sont ces coordonnées.

Nous avons comparé notre méthode aux SVM avec différents types de noyaux. Nous avons utilisé des noyaux classiques, appliqués à la concaténation des représentations aplaties des matrices des nœuds et d'adjacence. Pour les graphes à nœuds étiquetés comme MUTAG, nous avons utilisé les noyaux de graphes Weissfeiler-Lehman (WL) [14], Shortest-path [1], Hadamard Code (Hadcode) [7] ; pour les graphes à nœuds à attributs numériques, les noyaux de Propagation [11] et Multiscale Laplacian (MSLap) [10]. Les résultats donnés dans la Table 2 montrent la pertinence de notre méthode.

[1] À l'exception du jeu de données MNIST qui est pré-partagé pour l'entraînement et l'évaluation, les autres jeux de données ont été partagés selon le rapport 90%-10%. Les hyperparamètres ont été fixés par validation croisée selon les valeurs candidates suivantes : 10 values de 10^{-5} à 10^{3} pour C et pour la largeur de bande des noyaux RBF et polynomial, et le degré du dernier sur 4 valeurs 1 à 4.

TABLE 2 : Taux de bonne classification ($\pm$ écart type) sur les jeux de données graphes de nœuds étiquetés (MUTAG) ou à attributs numériques (Letter-med).

	SVM à noyaux classiques				SVM à noyaux de graphes					Notre méthode
	Linear	RBF	Polynomial	Sigmoid	WL	Shortest-path	Hadcode	Propagation	MSLap	MoFlow
MUTAG	0.761 ± 0.070	0.772 ± 0.039	0.717 ± 0.046	0.683 ± 0.056	0.778 ± 0.0	0.778 ± 0.0	0.778 ± 0.0	-	-	$\mathbf{0.939} \pm 0.016$
Letter-med	0.414 ± 0.022	0.419 ± 0.022	0.427 ± 0.028	0.281 ± 0.024	-	-	-	0.298 ± 0.135	0.410 ± 0.039	$\mathbf{0.752} \pm 0.012$

FIGURE 4 : Illustration des pré-images d'interpolation dans $\mathcal{Z}$. Chaque ligne illustre deux visages sélectionnés (le visage le plus à gauche et le plus à droite) et les pré-images de 10 interpolations entre les représentations de ces deux visages dans l'espace latent.

5 Conclusion

Dans cet article, nous avons présenté une nouvelle méthode de classification non linéaire qui ne souffre pas de la malédiction de la pré-image. Les résultats expérimentaux ont montré que notre méthode donne de très bons résultats en classification de jeux de données de référence. L'intérêt de la pré-image pour l'interprétabilité de l'espace latent a été également démontré sur plusieurs jeux de données. Nos travaux futurs seront consacrés à étendre notre méthode à un cadre plus général, aussi bien en régression qu'en apprentissage non supervisé.

Remerciements

Ce travail a été financé par l'ANR, projet « APi : Apprivoiser la pré-image » (ANR-18-CE23-0014).

Références

[1] Karsten M BORGWARDT et Hans-Peter KRIEGEL : Shortest-path kernels on graphs. *In Fifth IEEE international conference on data mining (ICDM'05)*, pages 8–pp. IEEE, 2005.

[2] Li DENG : The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[3] Laurent DINH, Jascha SOHL-DICKSTEIN et Samy BENGIO : Density estimation using real nvp. *arXiv preprint arXiv :1605.08803*, 2016.

[4] Will GRATHWOHL, Ricky T. Q. CHEN, Jesse BETTENCOURT et David DUVENAUD : Scalable reversible generative models with free-form continuous dynamics. *In International Conference on Learning Representations*, 2019.

[5] Paul HONEINE et Cédric RICHARD : Preimage problem in kernel-based machine learning. *IEEE Signal Processing Magazine*, 28(2):77 – 88, mars 2011.

[6] Linlin JIA, Benoît GAÜZÈRE et Paul HONEINE : A graph pre-image method based on graph edit distances. *In* Andrea et al. TORSELLO, éditeur : *Proceedings of the IAPR Joint International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (S+SSPR)*, pages 216–226, Venice, Italy, 21 - 22 janvier 2021. Springer International Publishing.

[7] Tetsuya KATAOKA et Akihiro INOKUCHI : Hadamard code graph kernels for classifying graphs. *In ICPRAM*, pages 24–32, 2016.

[8] Durk P KINGMA et Prafulla DHARIWAL : Glow : Generative flow with invertible 1x1 convolutions. *In Advances in neural information processing systems*, pages 10215–10224, 2018.

[9] Ivan KOBYZEV, Simon JD PRINCE et Marcus A BRUBAKER : Normalizing flows : An introduction and review of current methods. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):3964–3979, 2020.

[10] Risi KONDOR et Horace PAN : The multiscale laplacian graph kernel. *Advances in neural information processing systems*, 29, 2016.

[11] Marion NEUMANN, Roman GARNETT, Christian BAUCKHAGE et Kristian KERSTING : Propagation kernels : efficient graph kernels from propagated information. *Machine Learning*, 102:209–245, 2016.

[12] George PAPAMAKARIOS, Eric T NALISNICK, Danilo Jimenez REZENDE, Shakir MOHAMED et Balaji LAKSHMINARAYANAN : Normalizing flows for probabilistic modeling and inference. *J. Mach. Learn. Res.*, 22(57):1–64, 2021.

[13] Kaspar RIESEN, Horst BUNKE *et al.* : Iam graph database repository for graph based pattern recognition and machine learning. *In SSPR/SPR*, volume 5342, pages 287–297, 2008.

[14] Nino SHERVASHIDZE, Pascal SCHWEITZER, Erik Jan VAN LEEUWEN, Kurt MEHLHORN et Karsten M BORGWARDT : Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(9), 2011.

[15] W Nick STREET, William H WOLBERG et Olvi L MANGASARIAN : Nuclear feature extraction for breast tumor diagnosis. *In Biomedical image processing and biomedical visualization*, volume 1905, pages 861–870. SPIE, 1993.

[16] Thao TRAN THI PHUONG, Ahlame DOUZAL, Saeed Varasteh YAZDI, Paul HONEINE et Patrick GALLINARI : Interpretable time series kernel analytics by pre-image estimation. *Artificial Intelligence*, 286:103342, septembre 2020.

[17] Chengxi ZANG et Fei WANG : Moflow : an invertible flow model for generating molecular graphs. *In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 617–626, 2020.

[18] Fei ZHU et Paul HONEINE : Online kernel nonnegative matrix factorization. *Signal Processing*, 131:143 – 153, février 2017.

[19] Fei ZHU, Paul HONEINE et Jie CHEN : Pixel-wise linear/nonlinear nonnegative matrix factorization for unmixing of hyperspectral data. *In Proc. 45th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4737–4741, Barcelona, Spain, 4 - 8 mai 2020.