Autoencodeurs discriminants pour la détection et la reconnaissance de véhicules en imagerie aérienne

Sébastien Razakarivony [1], **Frédéric Jurie** [2]

1. *72 Rue de la Tour Billy*
   95100 Argenteuil, France
   srazakarivony@gmail.com
2. *Université de Caen, U.F.R. Sciences*
   Boulevard du Maréchal Juin
   14032 Caen Cedex, France
   frederic.jurie@unicaen.fr

RÉSUMÉ. *Les autoencodeurs, qui permettent de modéliser des données au moyen de variétés, peuvent être utilisés dans un contexte de détection d'objets pour modéliser l'apparence des classes d'objets à détecter. La distance entre un vecteur à classer et la variété peut alors être utilisée comme une mesure de probabilité d'appartenance du vecteur à la classe. Cependant, en construisant la variété de manière à ce que les vecteurs de la classe appartiennent à la variété, rien ne garantit que des vecteurs d'autres classes ne lui appartiennent pas également. Nous cherchons à lever cette limitation en proposant un nouveau type d'autoencodeurs, les* autoencodeurs discriminants*, qui ont la propriété de construire des variétés éloignant les formes n'appartenant pas à la classe d'objets à détecter de la variété. Une validation expérimentale dans un contexte de détection et reconnaissance de véhicules en imagerie aérienne permet de conclure sur la pertinence de la méthode proposée.*

ABSTRACT. *The autoencoders allow to model data with manifolds. In an object detection task, they model the appearance of objects to detect. The distance between a vector to classify and the manifold can then be used as a measure of the probability that the vector belong to it. However, if the learnt manifold is such that all vectors of the class belong to it, nothing garanties that the vectors of other classes will not. We propose to remove this limitation with a new kind of encoders, the discriminative autoencoders, which have the property to build manifolds that move away the negative examples from the positive ones. An experimental validation on the context of detection and recognition of vehicles allows to conclude on the method.*

MOTS-CLÉS : *vision par ordinateur, détection d'objets, variétés, apprentissage statistique.*

KEYWORDS: *computer vision, object detection, manifold, machine learning.*

DOI:10.3166/TS.32.245-264

1. Introduction

Cet article adresse la question de la détection de véhicules faiblement résolus dans des images aériennes (illustration figure 1). Il s'agit d'une tâche ancienne, non résolue à ce jour, dont la difficulté réside dans la petitesse des objets à détecter mais également dans les multiples changements d'apparence possibles de ceux-ci (variations de couleurs, d'illumination, présence d'occultations, d'ombres). Bien que difficile, cette tâche est importante pour de nombreuses applications en lien avec la surveillance ou la sécurité.

Détecter des petites cibles ne peut se faire directement en utilisant les méthodes standard de la littérature (par exemple, approche de Dalal et Triggs pour la détection de piétons (Dalal, Triggs, 2005) ou l'utilisation de modèles à parties (Felzenszwalb *et al.*, 2009)), ces méthodes n'étant pas adaptées à la détection d'objets d'une vingtaine de pixels de large. Par ailleurs, il est assez difficile d'obtenir des données d'apprentissage pour cette tâche, en raison du coût de collecte des images aériennes. Finalement, la tâche est également rendue difficile par le fait que les positions des cibles ne sont généralement pas corrélées avec leur voisinage immédiat dans l'image : un véhicule peut se trouver sur une route, un parking, dans une forêt. Dans certains cas, les véhicules peuvent même être partiellement camouflés, et, en conséquence, encore plus difficiles à détecter.

Une imagette 20x20 pixels peut être vue comme un vecteur dans un espace à 400 dimensions, dont les composantes peuvent s'expliquer en réalité par un petit nombre de paramètres (souvent appelés variables latentes), tels que la pose 3D ou l'illumination. Il a été montré que la théorie des variétés est un bon cadre formel pour représenter les objets de petite taille, pour lesquels des descripteurs standard peuvent difficilement être extraits ((Feraud *et al.*, 2001 ; Pentland, 1994 ; Zhang *et al.*, 2012)). Cette théorie permet de représenter la variété de grande dimension par un sous-espace de dimension inférieure. Par exemple, le travail de Zhang (Zhang *et al.*, 2012) montre que les images d'objets 3D vus depuis différents points de vue peuvent être représentées par des points sur une variété de dimension réduite. Différents travaux utilisent les variétés comme modèle génératif, tel que le travail de Pentland (Pentland, 1994), basé sur des variétés linéaires obtenues par analyse en composantes principales, ou le travail de Feraud (Feraud *et al.*, 2001), basé sur de l'apprentissage de variétés non linéaires.

Cependant, bien qu'elles modélisent fidèlement les apparences des objets, les approches basées sur les variétés ne se concentrent pas sur les informations discriminantes, contrairement aux approches de l'état de l'art, comme le boosting (Viola *et al.*, 2005), les séparateurs à vastes marges (SVM) (Cortes, Vapnik, 1995), ou encore certains réseaux de neurones (RN) (LeCun *et al.*, 1998). Cet article vise à lever cette limitation en proposant un nouveau type d'autoencodeurs, que nous désignons comme des *autoencodeurs discriminants*, qui permet la construction d'un modèle génératif de l'information discriminante. Contrairement aux autoencodeurs standard, notre autoencodeur discriminant apprend une variété qui, par construction, permet de bien représenter les apparences des cibles à détecter tout en assurant que les apparences

Figure 1. Images représentatives d'un problème de détection en imagerie aérienne, extraites de la base VeDAI

des fonds seront mal représentées. Une fois cette variété construite, une classification simple se basant sur l'erreur de reconstruction peut être utilisée. Nous développons également dans cet article les équations permettant d'étendre ce concept au cas multi-classe.

Nos expériences sur une base de données présentant des véhicules dans des images aériennes (la base VeDAI, décrite section 4) montrent que l'approche proposée est non seulement meilleure que les autoencodeurs standard, mais qu'elle permet d'améliorer significativement les résultats d'un classifieur discriminant classique tel que le SVM. De plus, en classification, les autoencodeurs discriminants donnent des résultats similaires à un SVM non linéaire, mais avec un temps d'exécution plus réduit, et ont de meilleures propriétés en ce qui concerne l'apprentissage.

Cet article est une version étendue de (Razakarivony, Jurie, 2014). Elle présente davantage de résultats qualitatifs sur les autoencodeurs discriminants, leur extension multiclasse et les expériences correspondantes.

Le reste de l'article est construit comme suit : tout d'abord nous présentons dans la section 2 les études en relation avec le travail proposé, puis nous présentons les

autoencodeurs discriminants et leur extension multiclasse dans la section 3. Enfin, les validations expérimentales sont présentées dans la section 4.

2. État de l'art

Même si la détection d'objets dans les images a une longue histoire dans la littérature de la vision par ordinateur, les travaux récents se concentrent principalement sur la détection de grands objets dans des images issues de la vie quotidienne. Ils sont pour la plupart basés sur l'utilisation de *fenêtres glissantes*, combinant des descripteurs tels que les histogrammes de gradient orientés (HOG) (Dalal, Triggs, 2005) ou les *local binary pattern* (Wang *et al.*, 2009), avec de puissants classifieurs discriminants, tels que les techniques de boosting (Viola *et al.*, 2005) ou le SVM (Cortes, Vapnik, 1995). Beaucoup d'améliorations ont été proposées, par exemple le développement de stratégies d'élagage (Lampert *et al.*, 2008) ou les modèles déformables à base de parties (Felzenszwalb *et al.*, 2009), optimisé par ajout de cascades dans (Benenson *et al.*, 2012). Les approches plus récentes sont basées sur des réseaux convolutionnels (Sermanet *et al.*, 2012 ; Oquab *et al.*, 2014 ; LeCun *et al.*, 1998), qui sont généralement utilisés pour traiter des objets assez bien résolus (221 par 221 pixels), même si quelques travaux ont été effectués pour la détection de visages ((Garcia, Delakis, 2004 ; Osadchy *et al.*, 2007)).

Plus proches de notre problème, certaines approches ont été développées spécifiquement pour la détection de véhicules. Dans (Zhao, Nevatia, 2003), les auteurs envisagent la détection de voitures comme un problème de détection d'objets 3D, prenant en compte les variations de points de vue et d'ombre. Dans (Stilla *et al.*, 2004), les auteurs proposent différents algorithmes adaptés aux différents capteurs utilisés (couleur, imagerie infrarouge, radar). De son côté, (Kembhavi *et al.*, 2011) montre des résultats intéressants reposant sur un ensemble de descripteurs développés spécifiquement pour la détection de véhicules, basés sur leurs couleurs et leurs attributs géométriques. Les résultats obtenus par les différentes approches mentionnées dans ce paragraphe ne peuvent être comparées, chacune utilisant des bases d'images et des métriques différentes. En outre, les différentes bases utilisées ne sont pas publiques.

Peu de travaux portent sur la détection de petits objets en environnement ouvert. Notons toutefois ceux de (Munder, 2006) qui s'intéresse à la détection de piétons (36×18 pixels) en utilisant des ondelettes de Haar ou des HOG combinés avec un classifieur SVM (Enzweiler, Gavrila, 2009). Les autres travaux utilisent généralement des techniques de saillance ((Rutishauser *et al.*, 2004 ; Seo, Milanfar, 2010)), inexploitables dans notre cas en raison de la complexité des fonds.

Les variétés, qui sont des sous-espaces plongés dans des espaces de grande dimension approximés localement par un espace euclidien (espace latent ou tangent), ont été utilisées de manière efficace pour modéliser les objets. Une variété peut être apprise de différente manière. Les techniques linéaires sont les plus simples, telles que l'analyse linéaire discriminante (Fisher, 1936), ou l'analyse en composante principale (Hotelling, 1933). Pour les techniques non linéaires, certaines méthodes sont

basées sur la propriété de conservation des distances géodésiques, comme Isomap (Tenenbaum *et al*., 2000). D'autres, telles que *local linear embedding* (Saul, Roweis, 2003) et ses variantes, apprennent des approximations locales de la variété. Enfin, d'autres approches apprennent la variété d'une manière globale, comme les autoencodeurs (Kramer, 1991). Il est intéressant de noter que la plupart de ces algorithmes ont été conçus pour visualiser des données de grandes dimensions en les projetant dans un espace 2D ou 3D et ne donnent que la fonction f, qui va de l'espace des descripteurs vers l'espace latent et non son inverse (requise pour notre approche de détection, comme expliqué plus loin). En revanche, l'ACP et les autoencodeurs permettent d'apprendre f et f^{-1}.

Les variétés ont déjà été utilisées par de nombreux auteurs sur des tâches de détection d'objets. Dans (Pentland, 1994), Pentland *et al.* introduisent le concept d'*eigenfaces*, en utilisant l'ACP pour construire une variété linéaire des visages. Dans le même esprit, (Carvalho *et al*., 2011) utilise une ACP pour modéliser non seulement les objets mais aussi les fonds. Des résultats intéressants ont été obtenus sur des images de voitures et de piétons. L'utilisation de deux variétés est nécessaire pour contrebalancer l'absence d'aspect discriminatif. Dans (Feraud *et al*., 2001), les auteurs utilisent des autoencodeurs pour construire un modèle de visage dans le but d'effectuer de la détection, mais le taux de fausse alarme est élevé. Dans (Razakarivony, Jurie, 2013), les auteurs proposent une méthode limitant ce taux en utilisant un modèle de fond.

Notre approche se base sur ces travaux récents, en combinant des descripteurs classiques (HOG, LBP) avec une technique d'apprentissage de variété. La contribution de nos travaux réside dans un nouveau modèle, les *autoencodeurs discriminants* et son extension, les *autoencodeurs discriminants multiclasse*.

La technique développée s'approche des techniques d'optimisation de descripteurs, telle que l'approche de (Snoek *et al*., 2012), qui apprennent une représentation par des autoencodeurs dans l'espace latent qui permet d'obtenir des descripteurs efficaces pour la discrimination. A l'inverse, nous nous proposons d'utiliser les autoencodeurs directement comme classifieurs par la reconstruction dans l'espace des descripteurs.

3. Autoencodeurs discriminants

Avant de présenter les autoencodeurs discriminants, nous commençons par expliquer comment les variétés peuvent être utilisées dans un contexte de classification, puis comment apprendre les variétés avec des autoencodeurs.

3.1. Les variétés comme modèles génératifs

Soit $\mathcal{H}$ l'espace des descripteurs et $\mathbf{x} \in \mathcal{H}$ un descripteur visuel extrait d'une image (par exemple la signature d'une région de l'image). Nous rappelons que construire une

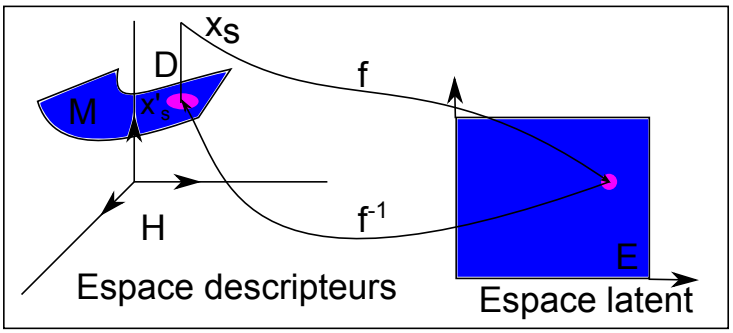

Figure 2. Illustration du concept de distance à la variété. *Soit X_s un vecteur et $X'_s = f^{-1} \circ f(X_s)$ sa projection sur la variété. $||X'_s - X_s||$ est la distance à la variété*

variété Riemannienne $\mathcal{M}$ représentative des descripteurs est équivalent à trouver une fonction f telle que :

$$\forall \mathbf{x} \in \mathcal{M}, \exists! \bar{\mathbf{x}} \in \mathcal{R}^n, \bar{\mathbf{x}} = f(\mathbf{x}) \quad (1)$$

f est le plongement de $\mathcal{M}$ et est une fonction isométrique.

Si $\mathbf{x}$ est sur la variété, $f^{-1} \circ f(\mathbf{x}) = \mathbf{x}$. $f^{-1} \circ f$ projette les points de l'espace des descripteurs sur la variété $\mathcal{M}$. Soit $P_{\mathcal{M}} = f^{-1} \circ f$, nous pouvons définir la distance à la variété comme $D_{\mathcal{M}}(\mathbf{x}) = ||\mathbf{x} - P_{\mathcal{M}}(\mathbf{x})||$ où $||x||$ représente la norme Euclidienne de $\mathbf{x}$. Le principe de cette projection est illustré figure 2. Cette distance peut être utilisée pour modéliser une catégorie, en considérant que plus un descripteur est près de la variété, plus la probabilité qu'il fasse partie de la classe modélisée est importante. Le descripteur testé est étiquetté du label de la classe la plus probable.

3.2. Autoencodeurs

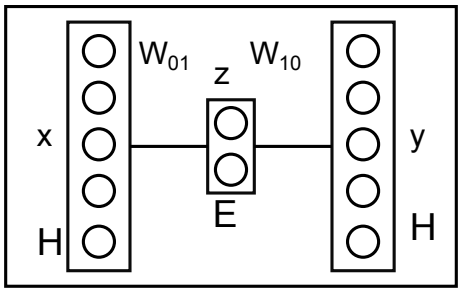

Figure 3. Représentation d'un autoencodeur à 3 couches où H est l'espace des descripteurs et E l'espace latent

Les autoencodeurs sont des réseaux de neurones symétriques, qui apprennent la fonction identité sous contrainte. Un autoencodeur simple est montré figure 3 mais

des architectures plus complexes peuvent être utilisées. Nous nous sommes limités à cette architecture, cependant, les équations restent valables pour des architectures plus complexes. Un neurone de la couche i est connecté à la couche $i+1$ et seulement à ces neurones. Soit W_{ij} la matrice des poids entre la couche i et j. Les couches sont numérotées de 0 (entrée) à N (couche centrale) puis de nouveau de N (couche centrale) à 0 (sortie), tel que montré figure 3. Comme le réseau est symétrique $dimension(W_{ji}) = dimension(W_{ij}^T)$. Chaque couche j a sa sortie $\mathbf{r}(\mathbf{x})$, complètement définie par la couche précédente $\mathbf{x}$ et la matrice des poids W_{ij} par $\mathbf{r}(\mathbf{x}) = h(W_{ij}\mathbf{x})$. h est appelée la fonction d'activation et est typiquement la fonction sigmoïde. Quand la fonction d'activation h est linéaire pour toutes les couches, l'autoencodeur effectue une ACP (Kramer, 1991). A l'inverse, utiliser une ou plusieurs couches avec des fonctions d'activation non linéaires permet au réseau d'approximer n'importe quelle fonction (Cybenko, 1989).

Soit χ l'ensemble des vecteurs d'entrainements $\mathbf{x}$. L'autoencodeur standard minimise la fonction de coût (Kramer, 1991) :

$$L(\chi) = \sum_{\mathbf{x} \in \chi} ||\mathbf{x} - \tilde{\mathbf{x}}||, \tag{2}$$

minimisant ainsi l'erreur de reconstruction des exemples positifs, $\tilde{\mathbf{x}}$ étant la reconstruction de $\mathbf{x}$ donnée par l'autoencodeur. Cette erreur est généralement minimisée en utilisant une descente de gradient stochastique, dans le cadre d'une rétropropagation de l'erreur (Rumelhart *et al.*, 1985). f et son inverse sont ainsi apprises simultanément. L'espace latent E est disponible à la sortie de la couche centrale. Des techniques pour avoir une meilleure convergence existent, telles que l'utilisation de *restricted boltzmann machine* (Ackley *et al.*, 1985) et de la *contrastive divergence* (Hinton, 2000), ou bien l'utilisation du "dropout". Le lecteur intéressé peut se reporter à (Hinton *et al.*, 2012) et (Hinton, Salakhutdinov, 2006).

Dans le contexte de l'apprentissage de variété, le réseau est utilisé pour apprendre f, donnant ainsi un plongement des données (Hinton, Salakhutdinov, 2006). Ici à l'inverse, le réseau est entièrement appris, ce qui donne la projection $P_{\mathcal{M}}(x)$ dont nous avions besoin. La distance à la classe, qui peut être utilisé en tant que score de classification, peut être calculée simplement par les équations données dans la section précédente.

3.3. Autoencodeurs discriminants

Nous introduisons ici le concept d'*autoencodeurs discriminants* (AED), qui permettent d'utiliser les données de deux classes (notées par la suite χ^+, pour les positifs et χ^- pour les négatifs) et apprennent une variété qui sera adaptée à la reconstruction des positifs tout en assurant une mauvaise reconstruction des négatifs. Ainsi, nous essayons de tirer également parti de l'information des exemples négatifs.

Soit $t(\mathbf{x})$ le label de l'exemple $\mathbf{x}$, avec $t(\mathbf{x}) \in \{-1, 1\}$ et $e(\mathbf{x})$ la distance de cet exemple à la variété, avec $e(x) = ||\mathbf{x} - \tilde{\mathbf{x}}||$. La nouvelle fonction à optimiser est :

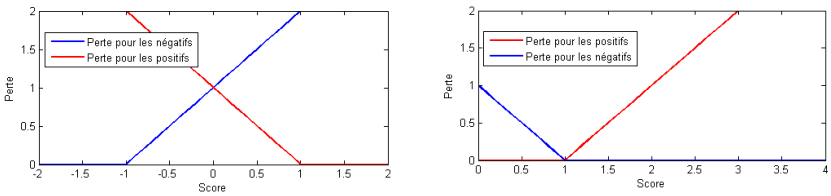

Figure 4. Hinge Loss, à gauche Hinge loss standard, à droite Hinge loss pour l'apprentissage de métrique

$$L_d(\chi^+ \cup \chi^-) = \sum_{\mathbf{x} \in \chi^+ \cup \chi^-} \max\left(0, t(\mathbf{x}) \cdot (e(\mathbf{x}) - 1)\right) \tag{3}$$

qui n'est rien d'autre que la fonction de coût Hinge Loss utilisée dans de nombreux algorithmes de classifications tels que le SVM. En pratique, nous utilisons une version légèrement différente de la Hinge Loss classique, comme proposé par (Mignon, Jurie, 2012) – la fonction standard étant $L_d = \sum_{\mathbf{x} \in \chi^+ \cup \chi^-} \max(0, 1 - t(\mathbf{x}) \cdot e(\mathbf{x}))$ – plus adaptée à notre problème, étant donné que les erreurs de reconstruction sont toutes positives. En un sens, notre problème se rapproche plus des algorithmes d'apprentissage de métrique que de ceux de classification. En effet, nous travaillons sur des paires que nous cherchons à approcher ou éloigner (bien que ces paires ne correspondent pas à celle utiliser pour des apprentissages de métrique). Quand le minimum est atteint, les exemples positifs (resp. négatifs) ont une erreur de reconstruction plus petite (resp. plus grande) que 1. Une illustration de ces fonctions de coût est présentée figure 4.

Pour optimiser la fonction de coût, nous utilisons une rétropropagation de l'erreur. Repartant des équations de l'autoencodeur, nous avons $\mathbf{y} = h(W_{10}\mathbf{z})$ et $\mathbf{z} = k(W_{01}\mathbf{x})$. Comme proposé par (Hinton, Salakhutdinov, 2006), nous prenons comme fonction k la fonction identité et une sigmoïde pour h. k étant l'identité, la transformation apprise de l'espace latent vers l'espace des descripteurs est linéaire. k est introduit pour conserver la généralité des équations. Pour simplifier les notations, notons $\mathbf{u}$ et $\mathbf{v}$ tels que : $\mathbf{u} = W_{10}\mathbf{z}$ et $\mathbf{v} = W_{01}\mathbf{x}$. L'objectif alors d'estimer les coefficients w_{ki} de W_{01} et W_{10} en minimisant $L(\chi^+ \cup \chi)$. La valeur optimale des w_{ki} vérifie : $\frac{\partial L}{\partial w_{ki}} = 0$, qui peut être résolue par une descente de gradient. Les dérivées partielles peuvent s'écrire :

$$\frac{\partial L}{\partial w_{ki}} = \frac{\partial L}{\partial \mathbf{e}^i} \cdot \frac{\partial \mathbf{e}^i}{\partial \mathbf{y}^i} \cdot \frac{\partial \mathbf{y}^i}{\partial \mathbf{u}^i} \cdot \frac{\partial \mathbf{u}^i}{\partial w_{ki}} \tag{4}$$

avec :

$$\frac{\partial \mathbf{e}^i}{\partial \mathbf{y}^i} = -1; \quad \frac{\partial \mathbf{y}^i}{\partial \mathbf{u}^i} = \frac{\partial h(\mathbf{u}^i)}{\partial \mathbf{u}^i}; \quad \frac{\partial \mathbf{u}^i}{\partial w_{ki}} = z^k \tag{5}$$

De plus, dans le cas d'une fonction sigmoïde, $\frac{\partial h(\mathbf{u}^i)}{\partial \mathbf{u}^i} = \mathbf{y}^i \cdot (1 - \mathbf{y}^i)$. Jusqu'ici, les équations sont identiques à la rétro-propagation d'erreur classique. Ensuite, nous introduisons la fonction hinge loss avec :

$$\frac{\partial L}{\partial \mathbf{e}^i} = \begin{cases} \mathbf{e}^i & \text{si } t(\mathbf{x}) \cdot (e(\mathbf{x}) - 1) > 0 \\ 0 & \text{sinon} \end{cases} \tag{6}$$

Nous obtenons la descente de gradient suivante : $\Delta w_{ki} = -\eta \delta_i \mathbf{z}^k$, avec $\delta_i = \mathbf{e}^i \frac{\partial h(\mathbf{u}^i)}{\partial \mathbf{u}^i}$ si $t(\mathbf{x}) \cdot (e(\mathbf{x}) - 1) > 0$ et 0 sinon.

Pour la couche cachée (nous ne donnons les dérivations que pour une couche cachée, cependant les équations sont les mêmes pour les couches suivantes) :

$$\frac{\partial L}{\partial w_{lk}} = \frac{\partial L}{\partial \mathbf{z}^k} \cdot \frac{\partial \mathbf{z}^k}{\partial \mathbf{v}^l} \cdot \frac{\partial \mathbf{v}^l}{\partial w_{lk}} \tag{7}$$

Les deux derniers termes ne changent pas, cependant le troisième devient :

$$\frac{\partial L}{\partial \mathbf{z}^k} = \sum_n e^n \frac{\partial e^n}{\partial \mathbf{z}^k} \text{ si } t(\mathbf{x}) \cdot (e(\mathbf{x}) - 1) > 0$$

ce qui nous donne :

$$\frac{\partial L}{\partial \mathbf{z}^k} = \sum_n e^n \frac{\partial (\mathbf{x}^k - h(\mathbf{u}^n))}{\partial \mathbf{u}^n} \cdot \frac{\partial \mathbf{u}^n}{\partial \mathbf{z}^k}$$

$$= -\sum_n e^n \frac{\partial (h(\mathbf{u}^n))}{\partial \mathbf{u}^n} w_{kn} = -\sum_n \delta_n w_{kn} \tag{8}$$

L'incrément devient $\Delta w_{lk} = -\eta \delta_k \mathbf{x}^l$ avec $\delta_k = \frac{\partial h(\mathbf{v}^k)}{\partial \mathbf{v}^k} \sum_n \delta_n w_{kn}$ si $t(\mathbf{x}) \cdot (e(\mathbf{x}) - 1) > 0$ et 0 sinon. Pour les deux incréments, η est le pas d'apprentissage. Différentes manières existent pour l'optimiser. Le lecteur intéressé pourra se référer à (Y. LeCun *et al.*, 1998) pour plus d'information.

Ces équations peuvent être rendues plus robustes par l'ajout d'une marge w à l'équation $t(\mathbf{x}) \cdot (e(\mathbf{x}) - 1) > 0$, qui devient $t(\mathbf{x}) \cdot (e(\mathbf{x}) - 1) + w > 0$. w est déterminée par validation croisée. Il est possible d'utiliser la fonction de régression logistique généralisée afin d'avoir de meilleures propriétés de convergence. Il est à noter que la constante 1 est arbitraire. Cependant, étant donné que les sorties du réseaux sont entre -1 et 1, il est préférable d'utiliser une valeur de 0.2 afin de correspondre à la dynamique de l'erreur de reconstruction.

Finalement, $e(x_{new})$, l'erreur de reconstruction pour un nouveau vecteur x_{new} peut être utilisée directement comme score de classification.

3.4. Extension multiclasse

Afin d'être utilisé dans des conditions de classification plus avancées, nous avons également mis en place une architecture multiclasse, que nous désignerons par *autoencodeur discriminant multiclasse* (AEDMC). Celle-ci est illustrée figure 5. Le principe

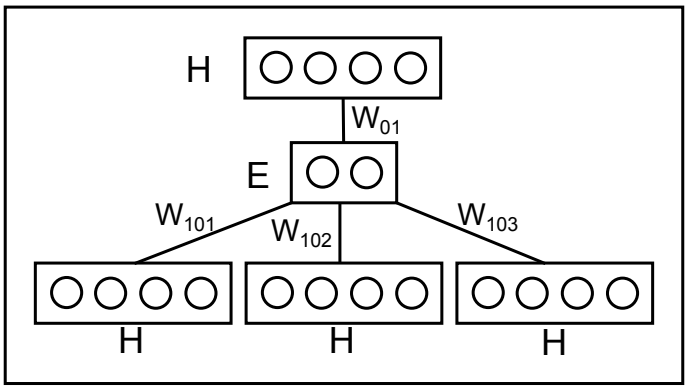

Figure 5. Autoencodeur discriminant multiclasse, illustration pour 3 classes

est le même que l'autoencodeur discriminant proposé précédemment, cependant, au lieu d'avoir un espace de sortie égal à l'espace d'entrée, l'espace de sortie a une dimension de $C \times n$, où C est le nombre de classe et n la dimension de l'espace d'entrée.

Soit $\chi = \bigcup \chi_C$ l'ensemble regroupant toutes les instances de toutes les classes et $t(\mathbf{x}) \in \{-1, 1\}^C$ le label de $\mathbf{x}$. Nous avons $t^j(\mathbf{x}) = 1$ si la classe de $\mathbf{x}$ est j, -1 sinon. En reprenant les notations précédentes, nous introduisons la fonction f qui correspond à la première couche d'entrée, telle que :

$$f(\mathbf{x}) = k(W_{01}\mathbf{x}) \tag{9}$$

puis les fonctions g^j telles que :

$$g^j(\mathbf{z}) = h(W_{10j}\mathbf{z}) \tag{10}$$

avec W_{10j} comme sur la figure 5. Nous définissons alors $\tilde{\mathbf{x}}^{\mathbf{j}}$ tel que :

$$\tilde{\mathbf{x}}^{\mathbf{j}} = g^j \circ f(\mathbf{x}) \tag{11}$$

Enfin, nous définissons $e^j(\mathbf{x})$ tel que :

$$e^j(\mathbf{x}) = ||\mathbf{x} - \tilde{\mathbf{x}}^{\mathbf{j}}|| \tag{12}$$

La fonction à minimiser est alors la suivante :

$$L(\chi) = \sum_{\chi} \sum_{j} max(0, t^j \mathbf{x}(e^j(\mathbf{x} - 1))) \tag{13}$$

De même que précédemment, cette fonction est optimisée par descente de gradient. Il est possible de l'adapter avec la fonction logistique généralisée, d'ajouter une marge et d'utiliser un paramètre d'échelle de 0.2.

4. Expériences

Dans cette section, nous présentons les données utilisées pour valider l'approche développée, puis les architectures utilisées, que ce soit pour la tâche de détection (classification binaire objet / fond) ou de la tâche de reconnaissance (classification multi-classe d'une vignette contenant une objet inconnu).

4.1. La base VeDAI

Figure 6. Régions de 100×100 pixels centrées sur des voitures (base VeDAI). La taille, les reflets spéculaires, les ombres et occultations rendent la tâche de détection difficile

Nous avons effectué la validation expérimentale sur une base conçue par nos soins pour évaluer les algorithmes de détection de véhicules faiblement résolus . Elle est décrite en détail dans (Razakarivony, Jurie, 2015) et est téléchargeable à des fins de recherche [1]. Cette base contient plus de 1200 images (1024x1024 pixels, 3 couleurs), avec de nombreux fonds et véhicules différents (ainsi qu'illustré figure 6). Ces images sont extraites du site Utah ARGC [2], plus précisément du 2012-HRO-6-inch-orthophotography set. La résolution des images est de 12.5×12.5 cm par pixel. Les voitures ont donc une taille aux alentours de 20×40 pixels. Leur détection est difficile à cause des occultations, des réflexions, des ombres et des spécularités comme montré figure 6. La variation intra-classe est importante. Nous avons utilisé un processus de validation croisée à 10 plis (*folds* en anglais). Les images sont réparties en 10 ensembles, chacun contenant 134 voitures réparties sur 121 images. Durant l'évaluation, 9 sont utilisés pour l'apprentissage, tandis que le dernier ensemble est utilisé pour le test. Chaque pli est utilisé une fois en tant qu'ensemble test. Nous nous réfèrerons à cette expérience sous le nom de *détection*.

Nous avons également créé une base de données permettant d'évaluer la tâche de classification à partir de la base VeDAI. Des fenêtres de taille 64x64 pixels autour de vignettes représentant les 8 classes de véhicules ont été extraites et transformées en base de classification. Il s'agit des 8 classes "car", "Boat","Camping car", "Pickup", "Tracteur", "Truck", "Van" et une classe regroupant tous les autres véhicules que nous désignerons par "Other". Une illustration des différentes classes est proposée figure 7. Ce problème de classification sera appelé *reconnaissance*. Ce type de tâche correspond à la situation où un véhicule a été détecté (par exemple une désignation manuelle, ou au moyen un détecteur de mouvement) et que sa classe doit être déterminée

1. https://downloads.greyc.fr/vedai
2. http://gis.utah.gov/

Figure 7. Images illustrant les différentes catégories de la base de données. De gauche à droite : Truck, Camping car, Tractor, Boat, Other, Pick-up et Van.

automatiquement. Pour la mesure de la performance, nous donnerons les taux de reconnaissance par classe, le micro-averaging (moyenne des taux de reconnaissance) et le macro-averaging (taux de reconnaissance global).

4.2. Algorithmes pour la détection

Notre algorithme se situe dans le cadre classique des algorithmes dits à *fenêtre glissante* et utilise l'apprentissage de variété pour prédire le score de chaque fenêtre. Toutes les régions rectangulaires d'une proportion largeur/longueur donnée sont évaluées par notre classifieur. En pratique, cela est fait par le biais d'une grille multi-échelles sur l'image. Nous avons utilisé un pas de 8 pixels et un ratio entre les échelles de $2^{\frac{1}{10}}$, comme effectué dans (Felzenszwalb *et al.*, 2009) – on a un rapport $2^{1/10}$ pour la 1^{re} échelle, $2^{2/10}$ pour la 2^{e} etc. Comme le ratio largeur/longueur d'un véhicule varie selon son orientation, différents classifieurs ont été appris pour différents ratios. Ce ratio varie entre 0.5 et 2.0 pour la classe des voitures par exemple. Ces ratios ont été déterminés par clustering des aires des exemples d'entraînement positifs. 4 échelles ont été utilisées, la distance à la cible étant approximativement connue. Pour améliorer l'efficacité du détecteur, nous utilisons une cascade à deux étages. Le premier étage est constitué de 12 SVM linéaires basés sur des HOG ou des LBP (2 orientations $\times$ 6 ratios largeur/longueur), tandis que le second étage affine les scores de détection en utilisant 12 autoencodeurs discriminants (chacun apparié avec le SVM correspondant). Les détecteurs du premier étage utilisent toutes les données d'apprentissage, tandis que ceux du second étage utilisent les exemples négatifs difficiles obtenus après filtrage par ce premier étage. Les négatifs difficiles sont tous les exemples de la base de données que le SVM classe avec un bon score. Durant le test, les 12 SVM parcourent l'image. Seules les fenêtres avec un score supérieur à -1.0 sont gardées (soit environ

0.06 % des fenêtres). Ensuite, comme habituellement dans un algorithme par fenêtre glissante, une étape de filtrage des non-maxima est effectuée. Nous avons utilisé une stratégie gloutonne simple, qui consiste à prendre les fenêtres de meilleur score et enlever les fenêtres qui recouvrent celles-ci dans un rayon égal à la moitié de la largeur de la fenêtre. Finalement, les fenêtres sélectionnées sont reclassées avec l'autoencodeur (standard ou discriminant). Notre algorithme peut prendre en entrée n'importe quel descripteur. Afin d'illustrer ce fait, nous avons effectué les expériences avec les descripteurs HOG et les descripteurs LBP. Les dimensions de ces descripteurs sont comprises entre 2500 et 3100, selon les différents ratio hauteur/largeur. Nous avons combiné les deux scores issus des deux étapes de la cascade. Dans ce cas, le résultat final est obtenu par combinaison linéaire avec $\alpha S_{autoencodeur}(\mathbf{x}) + (1-\alpha) S_{SVM}(\mathbf{x})$, où α est fixé par validation croisée. La validation des paramètres (nombre de neurones, paramètre α) se fait sur un des plis d'entrainement : l'apprentissage du classifieur utilise alors 8 des plis et l'effet des paramètres est visualisé sur le 9ème pli. Une fois tous ces paramètres bloqués, un nouvel apprentissage est réalisé en utilisant les 9 plis et le résultat est issu d'un test sur le 10ème pli. Cette procédure est ensuite effectuée 10 fois pour obtenir les résultats. α et le nombre de neurones ne sont pas nécessairement les mêmes pour chaque pli, mais, en pratique, nous avons observé que ces paramètres sont proches.

4.3. Algorithmes pour la reconnaissance

Pour les expériences de classification, nous avons testé deux approches. La première consiste à apprendre un autoencodeur discriminant pour chaque classe, en prenant comme ensemble négatif l'union des autres classes. La seconde approche utilise l'extension de l'autoencodeur discriminant, en effectuant une optimisation multiclasse. Afin de comparer rigoureusement nos résultats, nous avons également testé un SVM linéaire et un SVM non linéaire (noyaux *radial basis function*, abrégé SVM RBF), ainsi qu'un réseau de neurones classique en utilisant l'architecture de réseau de neurones pour la classification de la toolbox Matlab (abrégé MLP pour *multi-layers perceptron*).

5. Résultats

Dans cette section, nous donnons dans un premier temps quelques résultats quant au paramétrage des autoencodeurs discriminants. Dans un deuxième temps, nous donnons les résultats quantitatifs, que ce soit pour les tâches de détection ou de reconnaissance.

5.1. Étude paramétrique

Le paramètre le plus important pour tout algorithme basé sur les variétés est le choix de la dimension de la variété. Nous avons choisi de fixer cette dimension par validation croisée.

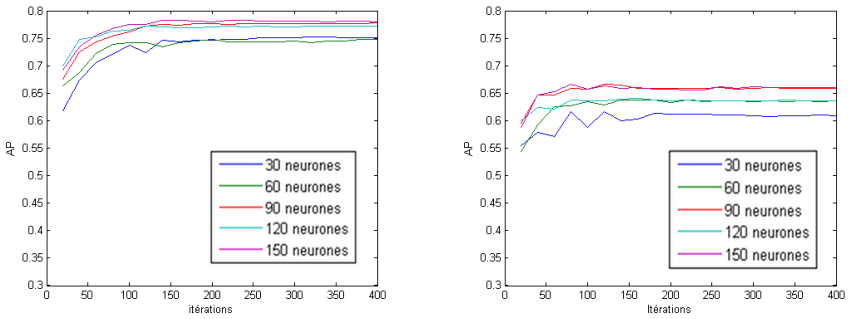

Figure 8. Average Precision en fonction du nombre de neurones et du nombre d'itérations, à gauche, sur un sous-ensemble ayant fait partie de l'apprentissage à droite, sur un sous-ensemble disjoint

En ce qui concerne la détection, il est visible sur la figure 8 que la performance change en fonction du nombre de neurones. Il est à voir que le nombre de neurones optimal sur la validation est en accord avec le nombre de neurones optimal sur l'apprentissage. Les performances varient en fonction de l'optimum local dans lequel aura convergé le réseau de neurones. Cependant, dès lors que suffisamment de neurones sont utilisés, de bonnes performances sont atteintes. À l'inverse, un nombre de neurones insuffisant (30 sur la figure) dégrade fortement les performances. Le nombre de neurones obtenu par validation croisée et qui correspond aux expériences présentées au paragraphe suivant est de 90.

Pour la reconnaissance, nous observons de même une grande stabilité des performances en fonction du nombre de neurones dès lors qu'ils sont en nombre suffisant, comme illustré en figure 9. Cette stabilité est valable tant pour l'autoencodeur discriminant à deux classes que pour son extension multiclasse.

Pour ce qui est du nombre d'itérations et de la convergence en général, que ce soit pour la détection (figure 8) ou la reconnaissance (figure 9), nous observons une courbe de croissance classique lors de l'apprentissage de réseaux de neurones, avec un fort gain au départ, qui se ralentit petit à petit. Les équations d'optimisation utilisées (moment, pas dégressif, etc), permettent de maintenir un plateau de performance suffisamment large pour que la sélection du nombre maximal d'itérations n'ait pas d'influence sur les performances (voir (Y. LeCun *et al.*, 1998) pour plus de précisions).

La figure 10 montre que le micro et le macro averaging augmentent avec le nombre d'itérations. De plus, pour certaines classes, les performances ont tendance à décroître avant de se stabiliser, tandis que pour d'autres elles augmentent. Ce comportement est tout à fait prévisible. En effet, au début de l'optimisation, certaines classes ont des scores plus élevés que d'autres, de par l'initialisation aléatoire des poids du réseau. Ces classes agissent comme classe par défaut et ont donc un score de bonne reconnaissance très élevé (Dans un cas extrême, si l'ensemble d'une classe répond toujours

plus fort que les autres, son taux de reconnaissance est de 100 %). Cependant, cette représentation erronée amène les autres classes à porter beaucoup de leurs fausses reconnaissances sur cette classe.

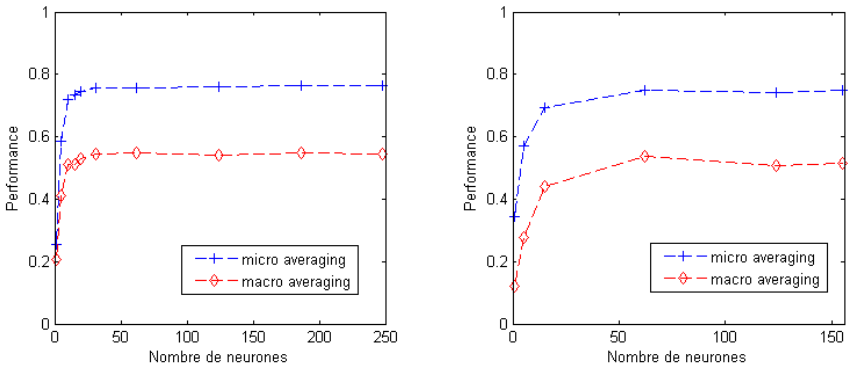

Figure 9. Robustesse du micro et macro averaging en fonction du nombre de neurones, à gauche pour 8 autoencodeurs discriminants et à droite, pour 1 autoencodeur multiclasse

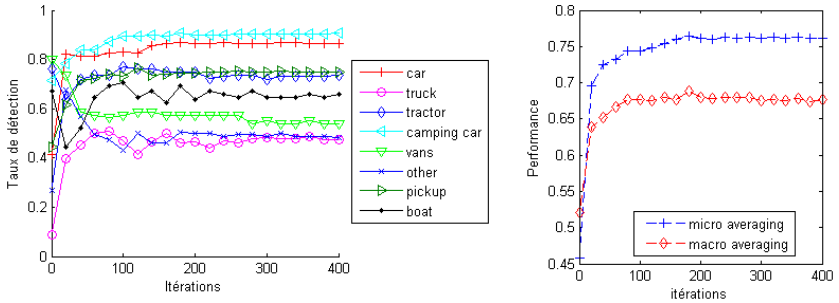

Figure 10. A gauche : taux de reconnaissance en fonction de la classe et du nombre d'itérations. A droite : micro et macro averaging sur l'ensemble des classes.

5.2. Résultats quantitatifs en détection

Les résultats obtenus sur VeDAI sont donnés en tableau 1. Premièrement, nous avons observé que l'autoencodeur standard (n'utilisant que des exemples positifs) ne donne pas de bons résultats. Même combiné au premier étage de la cascade, il n'améliore pas significativement les performances du SVM. Ceci confirme que l'autoencodeur seul n'est pas capable de discriminer correctement et il ne permet pas d'apporter une information complémentaire au SVM.

A l'inverse, pour l'autoencodeur discriminant, le gain est significatif pour les deux descripteurs, montrant que cette technique apporte en performance quel que puisse

être l'espace des représentations. En effet, l'autoencodeur discriminant donne de bien meilleurs résultats que l'autoencodeur classique (+38.0 de mAP avec HOG, +25.0 avec LBP) et est significativement meilleur que le SVM (+9.1 de mAP avec HOG et +6.3 avec LBP), alors qu'il utilise les mêmes exemples d'entrainement. Pour les différents points de fonctionnement, l'autoencodeur discriminant gagne en performance. Le grand écart type pour 0.01 FPPI rend la conclusion non fiable pour ce point, mais le gain est important pour les deux descripteurs.

La combinaison du score de l'autoencodeur discriminant et du score du SVM donne des résultats légèrement meilleurs que l'autoencodeur discriminant seul, montrant ainsi que l'autoencodeur a conservé la majorité de l'information du premier étage de la cascade.

Enfin, lorsque nous comparons notre approche avec le Deformable Part Model (DPM) de (Felzenszwalb *et al.*, 2009) (en utilisant la version 5 et en désactivant l'apprentissage des parties du fait de la petite taille des objets), le gain est d'environ $10\ \%$ de mAP également. Ces résultats montrent clairement que non seulement les autoencodeurs discriminants donnent de meilleurs résultats que les autoencodeurs classiques, mais qu'ils permettent d'améliorer significativement les performances d'un SVM ou du DPM dans une tâche de détection de petits objets.

Tableau 1. Comparaison de différentes chaînes algorithmiques en détection. (Deformable Part Model, Autoencodeur, Autoencodeur Discriminant)

Détecteur	mAP		Rappel @ 0.01 FPPI		Rappel @ 0.1 FPPI		Rappel @ 1 FPPI	
DPM (Felzenszwalb *et al.*, 2009)	60.5±4.2		13.4±6.8		31.4±5.8		74.5±4.5	
Descripteurs	HOG	LBP	HOG	LBP	HOG	LBP	HOG	LBP
SVM (1er étage)	58.9±3.5	52.9±3.5	13.2±5.1	10.5±6.3	30.4±3.9	30.8±3.9	72.1±4.1	63.2±5.0
SVM puis AE	30.0±3.9	34.2±4.4	1.5 ±1.6	3.6±1.6	6.8 ±1.8	13.3±3.9	39.5±4.1	43.2±7.1
SVM fusion AE	58.8±3.8	55.5±4.0	12.9±3.5	13.4±4.7	34.0±4.5	32.1±3.1	71.8±5.4	66.7±5.0
SVM puis AED	68.0±4.2	59.2±3.6	**21.2±6.9**	14.5±4.5	46.7±6.8	39.2±4.0	78.7±3.4	70.8±4.3
SVM fusion AED	**69.6±3.4**	**60.0±3.7**	20.4 ±6.2	**17.3±6.5**	**49.0±3.6**	**40.0±3.5**	**80.3±3.1**	**72.0±5.3**

5.3. Résultats quantitatifs en reconnaissance

Les résultats sont présentés tableau 2. De même qu'en détection, l'autoencodeur discriminant fait significativement mieux qu'un autoencodeur classique. Nous retrouvons ainsi ce qui avait été montré en détection.

De plus, les résultats sont similaires à un SVM RBF en termes de classification. Il est à noter que, si les résultats sont similaires, un SVM RBF a un nombre d'opérations non linéaires en $O(N)$, avec N le nombre de vecteurs support (environ 17 000 dans notre cas), tandis que l'autoencodeur discriminant a une complexité en $O(n)$, n étant la dimension de l'espace des descripteurs, aux alentours de 400. De plus, un SVM RBF a une complexité d'apprentissage en $O(n_e^2)$, avec n_e le nombre d'exemples d'apprentissage disponibles, tandis que l'autoencodeur discriminant a une complexité

Tableau 2. Résultats de reconnaissance comparés entre AE, AED, AEDMC, un réseau classique et différents SVM

Méthode	Boa	Cam	Car	Oth	Pic
8 SVM linéaires	21.4±18.3	83.9±4.6	82.8±5.6	24.2±7.6	53.0±8.4
8 AE	31.9±11.6	74.8±5.7	77.8±3.4	37.1±12.6	71.0±2.6
1 SVM MC RBF	56.1±13.4	83.7±7.7	84.6±3.9	47.8±6.5	74.5±1.9
8 SVM RBF	61.9±15.4	86.4±6.9	86.7±3.9	39.8±6.2	77.4±3.5
8 AED	58.3±12.4	87.9±6.4	86.0±3.6	41.6±5.9	76.7±2.0
1 AEDMC	50.4±10.1	87.0±5.5	86.7±4.7	38.9±6.5	74.1±2.8
MLP	1.4.±3.9	36.4±47.1	63.1±34.0	32.9±29.8	69.9±25.7
8 AED + 1 SVM RBF	57.7±13.1	86.8±6.5	86.5±3.8	47.1±7.4	76.6±2.6
Méthode	Tra	Tru	Van	Macro	Micro
8 SVM linéaires	48.6±8.2	11.9±7.2	27.1±8.0	44.1±3.4	60.3±3.2
8 AE	50.1±6.4	35.0±6.7	14.0±13.4	49.0±4.3	64.7±5.7
1 SVM MC RBF	68.4±11.6	57.7±7.8	60.9±12.0	66.8±3.3	75.1±3.2
8 SVM RBF	71.6±10.8	56.4±9.5	57.3±13.5	67.2±2.2	76.4±2.0
8 AED	69.2±8.6	53.5±9.6	57.4±11.6	66.3±5.8	75.7±6.9
1 AEDMC	67.9±8.2	52.3±9.0	48.5±10.0	63.2±2.3	74.4±2.4
MLP	30.7±39.4	18.7±26.0	0.3±0.8	31.7±8.3	50.7±11.3
8 AED + 1 SVM RBF	71.9±10.7	57.9±9.1	60.7±12.1	**68.1±3.0**	**76.7±2.7**

Tableau 3. Comparaison des temps d'exécution en Matlab, sur la même machine, pour 3 algorithmes différents. Le SVM RBF est l'implémentation proposée par libsvm

Méthode	SVM RBF	8 AED	1 AEDMC
Temps d'exécution (ms)	450.0	0.45	0.30

d'apprentissage en $O(n_e)$. L'autoencodeur discriminant permet donc d'obtenir des résultats similaires à un SVM RBF, mais avec un temps d'apprentissage et de test plus petit. Il peut également traiter plus de données d'entraînement.

Ensuite, l'utilisation d'un autoencodeur discriminant en multiclasse est tout aussi efficace que huit autoencodeurs. Les performances sont à peine dégradées, cependant, le nombre d'opérations est quasiment divisé par deux. Ainsi, il est possible de gagner à la fois sur le temps d'apprentissage et sur le temps de test.

Le tableau 3 présente les temps d'exécution en Matlab des trois algorithmes utilisés. Même si ces temps ne sont pas représentatifs dans l'absolu, nous retrouvons le fait que le SVM RBF est lent, tandis que l'autoencodeur multiclasse est à peu près 1,5 fois plus rapide que les 8 autoencodeurs discriminants.

Afin de voir si l'information encodée par un AED et par un SVM RBF était la même, nous avons effectué une expérience de fusion entre les deux classifieurs. Il apparait alors que les performances sont augmentées en ce qui concerne le macro et le micro averaging, d'environ 1 % en comparaison avec la meilleure performance précédente, mais sans être significativement supérieur. Nous pouvons donc en déduire

que, si les performances sont similaires, la frontière générée est toutefois différente et apporte une information supplémentaire utile.

6. Conclusions et perspectives

Cet article introduit le concept d'*autoencodeur discriminant*. En plus d'optimiser la reconstruction d'exemples positifs, cet autoencodeur éloigne les exemples négatifs de l'espace de reconstruction. Nous avons également présenté une extension multiclasse. Nous avons de plus montré comment apprendre de tels autoencodeurs, en s'inspirant de techniques d'apprentissage de métriques. Dans le contexte de la détection de petites cibles, nous avons montré que les autoencodeurs discriminants donnent de bien meilleurs résultats que les autoencodeurs classiques et permettent une amélioration significative des performances quand ils sont associés à un SVM linéaire. De plus, ils peuvent donner des résultats similaires à un SVM non linéaire, mais dans un temps plus restreint, tout en donnant la possibilité de passer à l'échelle en ce qui concerne le nombre d'exemples.

L'étude de l'apprentissage simultané de la marge et de la reconstruction est une des pistes à envisager. De même, une extension reprenant le formalisme discriminant mais en l'appliquant à des réseaux convolutionnels pourrait être envisagé, avec comme objectif d'apprendre, simultanément à l'apprentissage des classifieurs, un codage des images en indices visuels.

Remerciements

Ce travail a été financé par l'Agence nationale de la recherche et de la technologie, par le contrat CIFRE No2011/0850 et par SAGEM, entreprise du groupe SAFRAN.

Bibliographie

Ackley D., Hinton G., Sejnowski T. (1985). A learning algorithm for boltzmann machines. *Cognitive Science*, vol. 9, n° 1, p. 147–169.

Benenson R., Mathias M., Timofte R., Van Gool L. (2012). Pedestrian detection at 100 frames per second. In *Computer vision and pattern recognition (cvpr), 2012 ieee conference on*, p. 2903-2910.

Carvalho G., Moraes L., Cavalcanti G., Ren T. (2011). A weighted image reconstruction based on pca for pedestrian detection. In *International joint conference on neural networks.*

Cortes C., Vapnik V. (1995). Support-vector networks. *Machine Learning*, vol. 20, n° 3, p. 273-297.

Cybenko G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, vol. 2, n° 4, p. 303–314.

Dalal N., Triggs B. (2005). Histograms of oriented gradients for human detection. In *Ieee cvpr.*

Enzweiler M., Gavrila D. (2009). Monocular pedestrian detection: Survey and experiments. *IEEE PAMI*, vol. 31, p. 2179–2195.

Felzenszwalb P., Girshick R., Mcallester D., Ramanan D. (2009). Object detection with discriminatively trained part based models. *IEEE PAMI*, vol. 32, n° 9, p. 1627–1645.

Feraud R., Bernier O., Viallet J., Collobert M. (2001). A fast and accurate face detector based on neural networks. , vol. 23, p. 42–53.

Fisher R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of eugenics*, vol. 7, n° 2, p. 179–188.

Garcia C., Delakis M. (2004). Convolutional face finder: A neural architecture for fast and robust face detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, n° 11, p. 1408–1423.

Hinton G. (2000). Training products of experts by minimizing contrastive divergence. *Neural Computation*, vol. 14, p. 2002.

Hinton G., Srivastava N., Krizhevsky A., Sutskever I., Salakhutdinov R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Hinton G. E., Salakhutdinov R. R. (2006, juillet). Reducing the Dimensionality of Data with Neural Networks. *Science*, vol. 313, p. 504-507.

Hotelling H. (1933). Analysis of a complex statistical variable into principal components. *Journal of educational psychology*, vol. 24, n° 6, p. 417.

Kembhavi A., Harwood D., Davis L. S. (2011). Vehicle detection using partial least squares. *IEEE PAMI*, vol. 33, n° 6, p. 1250-1265.

Kramer M. (1991). Nonlinear principal component analysis using autoassociative neural networks. *Am. Inst. of Chem. Engineers Jour.*, vol. 37, n° 2, p. 233–243.

Lampert C., Blaschko M., Hofmann T. (2008). Beyond sliding windows: Object localization by efficient subwindow search. In *Ieee cvpr*.

LeCun, Bottou, Bengio, Haffner. (1998). Gradient-based learning applied to document recognition. *Proc. of the IEEE*, vol. 86, n° 11, p. 2278–2324.

LeCun Y., Bottou L., Orr G., Müller K. (1998). Efficient backprop. In *Neu. net.: Tricks of the trade.*

Mignon A., Jurie F. (2012). Pcca: A new approach for distance learning from sparse pairwise constraints. In *Cvpr*.

Munder S. (2006). An experiment study on pedestrian classification. In, vol. 28.

Oquab M., Bottou L., Laptev I., Sivic J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. In *Ieee cvpr*.

Osadchy M., Cun Y. L., Miller M. L. (2007). Synergistic face detection and pose estimation with energy-based models. *The Journal of Machine Learning Research*, vol. 8, p. 1197–1215.

Pentland A. (1994). Viewbased and modular eigenspaces for face recognition. In *Ieee cvpr*.

Razakarivony S., Jurie F. (2013). Small target detection combining foreground and background manifolds. In *Iapr international conference on machine vision and application.*

Razakarivony S., Jurie F. (2014). Autoencodeurs discriminants pour la détection de cibles faiblement résolues. In *Rfia.*

Razakarivony S., Jurie F. (2015, mars). *Vehicle Detection in Aerial Imagery : A small target detection benchmark*. Technical Report n° GREYC-2015-03-04. GREYC CNRS UMR 6072, Universite de Caen. Consulté sur https://hal.archives-ouvertes.fr/hal-01122605

Rumelhart D., Hinton G., Williams R. (1985). *Learning internal representations by error propagation*. Rapport technique. DTIC Document.

Rutishauser U., Walther D., Koch C., Perona P. (2004). Is bottom-up attention useful for object recognition? In *Ieee cvpr.*

Saul L., Roweis S. (2003). Think globally, fit locally: unsupervised learning of low dimensional manifolds. *JMLR*, vol. 4, p. 119–155.

Seo H., Milanfar P. (2010). Visual saliency for automatic target detection, boundary detection, and image quality assessment. In *Ieee icasp.*

Sermanet P., Chintala S., LeCun Y. (2012). Convolutional neural networks applied to house numbers digit classification. In *Icpr.*

Snoek J., Adams R. P., Larochelle H. (2012). Nonparametric guidance of autoencoder representations using label information. *The Journal of Machine Learning Research*, vol. 13, n° 1, p. 2567–2588.

Stilla U., Michaelsen E., Soergel U., Hinz S., Ender H. (2004). Airborne monitoring of vehicle activity in urban areas. *International Archives of Photogrammetry and Remote Sensing*, vol. 35, n° B3, p. 973–979.

Tenenbaum J. B., de Silva V., Langford J. C. (2000). A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science*, vol. 290, n° 5500, p. 2319–2323.

Viola P., Jones M. J., Snow D. (2005, juillet). Detecting pedestrians using patterns of motion and appearance. *IJCV*, vol. 63, n° 2, p. 153–161.

Wang X., Han T., Yan. S. (2009). An hog-lbp human detector with partial occlusion handling. In *Iccv*, p. 32–39.

Zhang X., Gao X., Caelli T. (2012). Parametric manifold of an object under different viewing directions. In *Eccv*, p. 186–199.

Zhao T., Nevatia R. (2003). Car detection in low resolution aerial images. *Image and Vision Computing*, vol. 21, n° 8, p. 693–703.

Article soumis le 7/12/2014
Accepté le 2/06/2015