Selection itérative de transformations pour la classification d'images

Mattis Paulin, Jérôme Revaud, Zaid Harchaoui, Florent Perronnin, Cordelia Schmid

Selection itérative de transformations pour la classification d'images

Mattis Paulin [1,*] Jérôme Revaud[1,*] Zaid Harchaoui[1,*] Florent Perronnin[2] Cordelia Schmid[1,*]

[1] Inria [2] Computer Vision Group, XRCE, France

Résumé

En classification d'images, une stratégie efficace pour apprendre un classifieur invariant à certaines transformations consiste à augmenter l'échantillon d'apprentissage par le même ensemble d'exemples mais auxquels les transformations ont été appliquéées. Néanmoins, lorsque l'ensemble des transformations possibles est grand, il peut s'avérer difficile de sélectionner un petit nombre de transformations pertinentes parmi elles tout en conservant une taille d'échantillon d'apprentissage raisonnable. optimal. En effet, toutes les transformations n'apportent pas le même impact sur la performance ; certains peuvent même dégrader la performance. Nous proposons un algorithme de sélection automatique de transformations : à chaque itération, la transformation qui donne le plus grand gain en performance est sélectionnée. Nous évaluons notre approche sur les images de la compétition ImageNet 2010 et améliorons la performance en top-5 accuracy de 70.1% à 74.9%.

Mots Clef

Sélection de variables, classification d'images, exemples virtuels.

Abstract

An approach to learning invariances in image classification is to augment the training set with transformed versions of the original images. However, given a large set of possible transformations, selecting an optimal subset of transformations is challenging. Indeed, transformations are not equally informative and adding uninformative transformations increases training time with no gain in accuracy. We propose a principled algorithm – Image Transformation Pursuit (ITP) – for the automatic selection of transformations. ITP works in a greedy fashion, by selecting at each iteration the one that yields the highest accuracy gain. ITP also allows to efficiently explore complex transformations, that are combinations of basic transformations. We report results on the ImageNet 2010 challenge dataset. We achieve an improvement of top-5 accuracy from 70.1% to 74.9%.

Keywords

Variable selection, image classification, virtual examples

*Équipe LEAR, Inria Grenoble Rhône-Alpes, Laboratoire Jean Kuntzmann, CNRS, Univ. Grenoble Alpes, France.

1 Introduction

La difficulté du problème de la classification d'images réside dans le fait que plusieurs objets d'une même classe peuvent présenter une apparence très différente. Sur ce point, nous distinguons deux sources de variabilité. La variabilité intrinsèque, tout d'abord, découle du fait que deux instances d'un même objet peuvent être visuellement différentes, même lorsqu'elles sont observées sous le même point de vue (ainsi, deux zèbres peuvent avoir des motifs de rayure différents). La variabilité extrinsèque relève elle de facteurs indépendant de l'objet en lui-même, tels que la différence de point de vue, les conditions d'illuminations ou la compression.

L'invariance d'un système d'apprentissage, c'est-à-dire sa capacité à prédire le même résultat quelles que soient les variations d'une image, doit être apprise en présentant au classifieur autant d'images que possible. Néanmoins, le coût d'annotation de celles-ci est souvent élevé et il existe, selon [4], trois familles d'approches pour construire un système invariant à partir d'un ensemble d'apprentissage fini : (i) créer des exemples dits virtuels en appliquant des transformations sur les exemples d'origine [15, 8, 9, 14], (ii) concevoir une représentation des exemples invariante aux transformations attendues [28, 26] et (iii) incorporer l'invariance au sein de la structure du système d'apprentissage, comme le font les réseaux de neurones convolutionnels [2].

Nous proposons ici d'explorer la création d'exemples virtuels (i) lors de l'entrainenement et du test. Si une telle tâche paraît difficile pour le cas de la variabilité intrinsèque (sauf dans certains cas comme la reconnaissance de piétons [19, 22]), il est possible de simuler la variabilité extrinsèque au travers de simples transformations géométriques et colorimétriques des images. Cette approche a fait ses preuves dans le domaine de la reconnaissance de chiffres [15, 8, 9], et récemment dans celle d'images [14].
Une sélection efficace d'un ensemble adapté de transformations est, à notre sens, indispensable à la réussite d'une telle approche. En effet, des transformations trop conservatrices (comme enlever la première rangée de pixels) n'ont que peu d'impact pour un coût d'extraction et d'apprentissage non négligeable, tandis que des transformations trop agressives (par exemple une symmétrie verticale) engendrent des images irréalistes et peuvent dégrader la performance. À ce jour et à notre connaissance, les transformations sont toujours sélectionnées à la main. Ce procédé est encore acceptable pour des images de chiffres en noir

et blanc de 256 pixels, mais très vite onéreux lorsque le nombre de transformations possibles augmente, comme en classification d'images. Par la suite, nous considérons par exemple quarante transformations de base ainsi que leurs combinaisons deux-à-deux, pour un total de plus d'un millier de transformations possibles.

Nous proposons une approche systématique de sélection de transformations, denommée Selection Itérative de Transformations (SIT), qui sélectionne de manière incrémentale un ensemble optimal de transformations à partir d'un dictionnaire. De fait, SIT permet d'explorer efficacement l'ensemble des transformations d'ordre deux, qui correspondent aux compositions de deux transformations de base.

Nous menons nos expériences sur les images du challenge ImageNet de 2010 ("ILSVRC") et notons des améliorations significatives. Nous apprenons cinq transformations avec SIT sur 30000 images, améliorant la performance en *top-5 accuracy* (meilleur résultat parmi cinq propositions de l'algorithme) de 40.4% à 49.5%. En appliquant ces mêmes transformations à l'ensemble entier, nous améliorons celle-ci de 70.1% à 74.9%. Une conclusion importante de nos travaux est qu'il est indispensable d'appliquer les transformations choisies aussi bien à l'entrainement qu'à l'apprentissage.

2 Travaux antérieurs

Nous passons en revue les travaux antérieurs traitant de l'enrichissement des données par l'ajout d'exemples virtuels.

Exemples virtuels

La première stratégie, consistant à injecter un bruit dans les données, revient souvent à appliquer une certaine fonction de régularisation. En particulier, engendrer un bruit gaussien pour une fonction de perte quadratique est équivalent à une régularisation en norme L_2 [3]. Se référer à [1, 29, 20, 30, 6, 18] pour des résultats d'équivalence plus généraux, et en particulier pour le récent bruit par omission (*drop-out*) [14, 31]. Si cette stratégie d'injection de bruit jouit d'une grande simplicité d'implémentation, elle s'avère souvent difficile d'interprétation du point de vue de la vision par ordinateur. En effet, pour certains bruits, il est impossible de recréer l'image virtuelle qui aurait engendré le descripteur perturbé.

Images virtuelles

La seconde stratégie travaille directement dans l'espace des images. Ainsi, les auteurs de [27] créent des images virtuelles à partir de celles d'origine en utilisant des transformations plausibles telles que découpage ou symmétrie horizontale. Par opposition avec les exemples virtuels, cette stratégie est plus intuitive et facile d'interprétation, mais transformer les images et en extraire des descripteurs peut s'avérer onéreux, à l'exception de la reconnaissance de chiffres, pour laquelle le calcul de champs de déformation élastiques peut être effectuée de manière élégante et bon marché [16]. Notre approche SIT utilise cette stratégie d'images virtuelles en sélectionnant de manière incrémentale un petit nombre de transformations à partir d'un dictionnaire, améliorant de ce fait les performances sans dégrader la capacité à passer à l'échelle. De plus, du fait de sa capacité à opérer sur un ensemble arbitrairement large de transformations, aucune connaissance a priori n'est nécessaire.

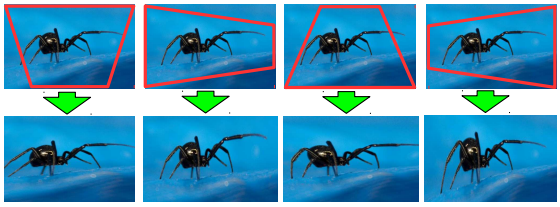


FIGURE 1 – 4 parmi les 8 homographies considérées.

3 Apprendre avec transformations

Dans ce qui suit, nous traitons en premier lieu des familles de transformations que nous appliquons aux images (partie 3.1). Dans un deuxième temps, nous effectuons un court rappel sur la classification par Descente de Gradient Stochastique (partie 3.2), Dans la partie 3.3, nous présentons l'algorithme de Sélection Itérative de Transformations (SIT). Enfin, nous comparons différentes stratégies d'aggrégation de scores au moment de la prédiction (partie 3.4).

3.1 Transformations d'images

Nous donnons pour commencer une liste des transformations que nous employons. Nous utilisons quarante transformations de base, appartenant à sept familles distinctes.

Symmétrie. Il s'agit de la réflexion horizontale, qui tire sa validité de la symétrie naturelle inhérente à la majorité des scènes et objets. Notons que de nombreux auteurs utilisent cette transformation pour augmenter leur ensemble d'apprentissage sans connaissance a priori [14, 11].

Découpages. Ces transformations restreignent l'image à l'une de ses sous-fenêtres. Nous utilisons dix découpages différentes, dont les paramètres (x_0, y_0, x_1, y_1) sont tirés préalablements de manière aléatoire avec : $(x_0, y_0) \in [0, 0.25]^2$ et $(x_1, y_i) \in [0.75, 1]^2$.

Homographies. Pour modéliser les changements de point de vue, nous considérons huit homographies, découlant de rotations horizontales ou verticales d'une caméra (Figure 1).

Homothéties. Nous réduisons la tailles des images par interpolation bilinéaire en utilisant des échelles de la forme $\sqrt{1.5^n}$, avec $n \in \{1, \ldots, 5\}$.

Colorimétrie. De manière similaire à [14], nous calculons la matrice de covariance des valeurs RVB de toutes les images d'apprentissage, puis notons $(\lambda_1, \lambda_2, \lambda_3)$ (resp. p_1, p_2, p_3) ses valeurs propres (resp. vecteurs propres). Préablablement à toute expérience, nous tirons trois triplets $(\varepsilon_1, \varepsilon_2, \varepsilon_3)$ aléatoirement avec une probabilité de loi $\mathcal{N}(0, 0.1)$ et ajoutons à tout pixel $p \in [0, 255]^3$ de l'image la valeur $\varepsilon_1\lambda_1 p_1 + \varepsilon_2\lambda_2 p_2 + \varepsilon_3\lambda_3 p_3$.

Compression JPEG. La compression JPEG, bien que conçue pour minimiser la perturbation visible à l'œil nu, introduit des perturbations non-négligeables pour les descripteurs locaux. Nous considérons trois valeurs d'encodage JPEG : 30, 50 et 70.

Rotations. Pour caractériser le changement d'orientation de la caméra, nous introduisons dix rotations d'angle $\{-15, -12, \ldots, -3, 3, \ldots, 12, 15\}$.

Transformations d'ordre K. Par transformations d'ordre K, nous entendons toute transformation obtenue comme composition de K transformations parmi celles définies précédemment.

3.2 Descente de Gradient Stochastique

Nous rappellons brièvement notre méthode d'apprentissage. Nous utilisons une fonction de perte de Séparateur à Vaste Marge (SVM) binaire, en un-contre-tous. Pour un ensemble de couples (descripteur, étiquette) $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, $x_i \in \mathbb{R}^d$, $y_i \in \{1, \ldots, C\}$, elle s'écrit sous la forme :

$$\ell(\mathcal{D}, w) = \sum_{i=1}^{n} \sum_{k=1}^{C} \ell_{\mathrm{SVM}}(x_i, y_{i,k}, w_k) + \frac{\lambda}{2}\|w\|_2^2, \quad (1)$$

avec $\ell_{\mathrm{SVM}}(x, y, w)) = \max(0, 1 - y(w^\top x))$ et $y_{i,k} = 1$ si $y_i = k$ et $y_{i,k} = 0$ sinon.

Le poids w minimisant la fonction de perte précédente est le résultat de l'apprentissage, la prédiction est ensuite réalisée sur un exemple x avec la formule :

$$y = \underset{k}{\mathrm{argmax}}\, w_k^\top x. \quad (2)$$

Nous choisissons d'optimiser ce problème par une méthode de descente de gradient stochastique (SGD), laquelle consiste à déterminer w itérativement en tirant à l'étape t un couple (exemple, étiquette) (x_t, y_t) et en mettant à jour pour chaque classe k :

$$w_k^{(t+1)} = w_k^{(t)} - \eta_t \left[\nabla_w \ell_{SVM}(x_t, y_{t,k}, w) + \lambda w\right] \quad (3)$$

Pour un pas d'apprentissage $\eta_t = \eta_0/(1 + \eta_0 \lambda t)$, $w^{(t)}$ converge vers le minimum global du risque [5]. Pour un nombre élevé de classes en un-contre-tous, il est parfois nécessaire de rééquilibrer le tirage des positifs par rapport aux négatifs, au moyen d'un hyperparamètre entier β. Au moment du tirage d'exemples, un positif est pris avec un probabilité $1/\beta$ et un négatif $(\beta - 1)/\beta$.

3.3 Sélection Itérative de Transformations

Nous décrivons maintenant notre algorithme. Soit $\mathbb{T} = \{\mathcal{T}_1, \ldots, \mathcal{T}_{|\mathbb{T}|}\}$ un ensemble de transformations avec la convention que $\mathcal{T}_1$ est l'identité. Nous cherchons un sous-ensemble $S \subset \mathbb{T}$ tel qu'entrainer sur les données transformées :

$$\mathcal{D}^t = \bigcup_{\mathcal{T} \in S} \mathcal{T}(\mathcal{D}) \quad (4)$$

donne les meilleures performances possibles. Le nombre de sous-ensembles possibles étant exponentiel en $|\mathcal{T}|$, il est impossible de tous les essayer. Nous proposons une alternative incrémentale.

Algorithme. Nous initialisons S avec l'identité $\mathcal{T}_1$. Puis, nous cherchons itérativement parmi les transformations restantes celle qui apporte le plus haut gain (par apprentissage sur une partie D_{entr} de D^t et validation sur son complémentaire D_{val}) et l'ajoutons à S. L'algorithme s'arrête lorsqu'un nombre T fixé de transformations a été sélectionné, ou lorsque le gain à l'ajout d'une nouvelle transformation tombe en dessous d'un seuil Δ.

Algorithm 1 Sélection Itérative de Transformations (SIT)

ENTRÉES : Ensemble d'images $\mathcal{D}$. Ensemble $\mathbb{T}$ de transformations de base. Seuil Δ et nombre T de transformations.
INITIALISATION : $S = \{\mathcal{T}_1\}$.
Tant que $|S| \leq T$ et gain inférieur à Δ **faire**
- Séparer $\mathcal{D}$ en $\mathcal{D}_{\mathrm{entr.}}$ et $\mathcal{D}_{\mathrm{val.}}$.
- **Pour** $\mathcal{T} \in \mathbb{T}, \mathcal{T} \notin S$ **faire**
  - Soit $S^+ = S \cup \{\mathcal{T}\}$.
  - Entrainer avec $S^+(\mathcal{D}_{\mathrm{entr.}})$
  - Calculer le gain $\mathcal{P}_\mathcal{T}$ sur $S^+(\mathcal{D}_{\mathrm{val.}})$.
- $S = S \cup \{\mathrm{argmax}_\mathcal{T}\, \mathcal{P}_\mathcal{T}\}$.

SORTIE : S.

Transformations d'ordre K. Pour opérer une sélection au sein d'un grand nombre de transformations d'ordre K, nous proposons une dérivation de SIT. Elle s'appuie sur le postulat que si deux transformations sont bénéfiques à la classification, leur composition est aussi susceptible de l'être. Une stratégie similaire est utilisée en statistique pour l'analyse en produit tensoriel ANOVA [12].

Notre approche commence par sélectionner une liste de T transformations d'ordre $K - 1$, puis engendre toutes les compositions d'ordre K de deux de ses éléments. Une nouvelle procédure SIT est alors lancée, opérant sur la concaténation des transformations d'ordre $K-1$ avec les nouvelles transformations. Comparée à une approche qui continuerait le processus incrémental à partir des transformations d'ordre $K - 1$, cette approche renvoie une meilleure séquence à T fixé.

3.4 Agrégation des scores de prédiction

Il est aussi possible de créer des exemples virtuels sur les images de test, mais il existe plusieurs manières d'agréger les scores obtenus par chacune de ces réalisations. Nous en proposons trois.

Moyenne. De manière analogue à [14], nous prenons comme consensus final la moyenne des scores de chacune des images virtuelles. En notant $s_t^{(k)}$ le score attribué à la t-ième transformation par le classifieur relatif à la classe k, nous définissons $s_{\mathrm{moy}}^{(k)} := \sum_{t=1}^{T} s_t^{(k)}$.

Maximum. Notre deuxième schéma consiste à prendre la transformation qui renvoie le meilleur score, c'est-à-dire celui de confiance maximale : $s_{\max} := \max_t s_t^{(k)}$.

Entropie Un intermédiaire entre les approches ci-dessus est la suivante : $s_{\mathrm{entropie}}^{(k)} := \log \sum_{t=1}^{T} \exp s_t^{(k)}$.

4 Expériences

4.1 Protocole expérimental

Images. Nous utilisons l'ensemble des images issues du challenge ImageNet 2010 [1] (ILSVRC) réparties en mille classes et 1.2 millions d'images d'entrainement, cinquante mille de validation et cent cinquante mille de test. En accord avec la pratique courante, nous utilisons la mesure de précision au rang 5, c'est-à-dire le pourcentage de quintuplets renvoyés par le système contenant la classe correcte. Pour nos expériences de sélection de transformations, nous utilisons un sous-ensemble des exemples d'entrainement de trente images par classe, que nous nommons ILSVRC-30. A l'exception des résultats finaux sur ILSVRC tout entier, nous prédisons sur l'ensemble de validation.

Descripteurs. Nous utilisons deux types de descripteurs d'images. Le premier est le Vecteur de Fisher (VF) [25] appris sur des descripteurs locaux SIFT [17] et couleur [7], projeté par ACP en dimension 61. A l'instar de [24], nous concaténons à ces descripteurs locaux leur position relative (x, y) et leur échelle σ, pour un total de 64 dimensions. Cette stratégie s'avère moins coûteuse que les pyramides spatiales [24]. Nos VFs sont appris avec 256 Gaussiennes, pour une réprésentation finale de dimension 32K. Nous utilisons ensuite la compression PQ [13], qui offre une perte de performance minime en classification d'images à grande échelle. Nous utilisons les paramètres de [23] et séparons nos VFs en sous-vecteurs de taille 8, encodés sur 8 bits.

Dans un second temps, nous travaillons sur des descripteurs DeCAF [10] appris sur les images du challenge ImageNet 2012. Notons tout d'abord qu'il existe une forte intersection entre les images des deux challenges, et que l'apprentissage a été réalisé en utilisant découpages, symmétrie et colorimétrie.

SGD Notre algorithme de descente de gradient stochastique possède trois hyperparamètres [21], que nous déterminons par validation croisée : la régularisation λ, le déséquilibre β et le nombre d'époques.

Fusion. Pour les VF, la moyenne des scores de confiance renvoyés par les canaux SIFT et couleur est utilisée pour la prédiction finale, notée "fusion" par la suite.

4.2 Aggrégation des scores

En supposant qu'après une première phase de sélection, nous disposons de $T - 1$ transformations, nous élaborons plusieurs scénarios dont les résultats sont décrits dans le Tableau 1. :

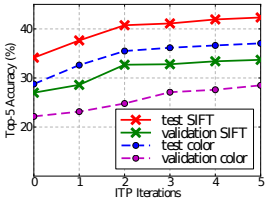
- Train 1/ Test 1 : entrainement et test sont effectués sur les images originales
- Train T/ Test 1 : l'entrainement est réalisé sur les images d'origine et transformées, tandis que le test n'utilise que les images originales.
- Train 1/ Test T : La prédiction est réalisée sur la moyenne des scores des images d'origine et de leurs transformées ; le classifeur est appris sur les données brutes.

1. http://www.image-net.org/challenges/LSVRC/2010/index

train	test	SIFT	couleur	fusion
1	1	34.1	28.9	40.4
1	T	36.2	30.6	42.5
T	1	38.7	31.5	44.4
T	T	**42.3**	**37.1**	**48.7**

Schéma	SIFT	couleur	fusion
moy.	42.3	37.1	48.7
max	42.1	36.8	48.9
entropie	42.6	37.3	49.2

TABLE 1 – Gauche : comparaison des différents schémas d'utilisation des transformations pour l'entrainement et le test sur ILSVRC-30 avec $T = 6$. Droite : différentes méthodes d'aggrégation des scores pour la prédiction.



	SIFT	Couleur
1	symétrie	découpage2
2	découpage0	color1
3	homog2	symétrie
4	découpage6	découpage1
5	découpage1	color0

FIGURE 2 – Evolution des performances en validation et en test en fonction des itérations de SIT sur ILSVRC-30 (gauche). Droite : liste des transformations sélectionnées.

- Train T/ Test T : Entrainement et test sont effectués sur les images transformées.

Les résultats obtenus montrent que le schéma Train T/ Test T est nettement supérieur aux autres, ce qui prouve qu'il est indispensable d'appliquer les transformtations à l'entrainement comme au test. Par la suite, sauf mention explicite du contraire, nous appliquons cette méthode.

Aggrégation des scores. Dans le Tableau 1, nous comparons les différentes méthodes d'aggrégation des scores des transformations présentées dans la partie 3.4. Tous les schémas donnant des valeurs similaires, nous gardons le plus simple, c'est-à-dire la moyenne.

4.3 Sélection de Transformations

Nous apprenons cinq transformations d'ordre un sur ILSVRC-30 avec SIT et rapportons dans la Figure 2 l'évolution de l'exactitude en validation (mesure utilisée pour sélectionner les transformations) et en test (sur un jeu de données indépendant). Ces deux mesures varient de manière similaire, ce qui valide le principe de notre algorithme.

La Figure 2 donne aussi la liste des transformations sélectionnées, qui montre une nette préférence pour les découpages. L'observation de ces découpages montre qu'ils se concentrent majoritairement sur le centre de l'image. On pourrait penser que l'on ne ferait qu'apprendre un biais de l'ensemble de données, par l'apprentissage d'une carte de saillance [24]. Pour tester cette hypothèse, nous extrayons tous les descripteurs locaux d'une image et de ses transformées et les agrégeons en un seul VF. Les descripteurs locaux présents dans de multiples instances sont ainsi pondérés par leur fréquence d'apparition. En fusion, nous obtenons sur ILSVRC-30 une performance en *top-5 accuracy* de 44.6% ce qui constitue un gain significatif par rapport aux 40.4% appris sur les images d'origine. Néanmoins, cela reste en dessous des performances de SIT : 48.7%. Nous n'apprenons donc pas simplement une distribution spatiale a priori sur la position de l'objet.

4.4 Transformations d'ordre deux

Nous raffinons notre stratégie de sélection en incorporant des transformations d'ordre deux, c'est-à-dire des transformations de transformations (composition par exemple d'une symétrie avec une découpage). Les résultats, présentés dans le Tableau 2, montrent une légère amélioration avec des transformations d'ordre deux.

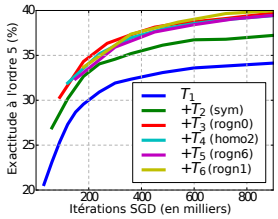
	SIFT	Couleur
1	symétrie×découpage0	color1
2	découpage6	color1× découpage1
3	symétrie	symétrie× color0
4	homographie2	découpage2×color0
5	symétrie×découpage1	découpage2×symétrie

	SIFT	Couleur	Fusion
Origine	34.1	28.9	40.4
SIT	41.8	37.1	48.4
2-SIT	42.6	38.3	**49.5**

TABLE 2 – Transformations sélectionnées par l'algorithme 2-SIT (haut) et amélioration en exactitude apportée par l'ordre deux (bas).

4.5 Précision pour un budget donné

Nous rapportons la précision au cours des itérations de la SGD (une itération correspond au traitement d'une image). De manière intéressante, nous observons que la convergence de la SGD s'accélère lorsque le nombre T de transformations augmente (Figure 3). Pour mieux en faire l'expérience, nous rapportons par ailleurs le nombre d'exemples nécessaires à la SGD pour atteindre une précision cible. Nous observons que le temps nécessaire pour obtenir une certaine exactitude décroit lorsque le nombre de transformations augmente, et ce, en dépit de l'augmentation de la taille de l'ensemble de données. Ainsi, obtenir une exactitude à l'ordre cinq de 45% sur ILSVRC-30 se réalise trois fois plus rapidement lorsque $T = 5$ que lorsque $T = 2$. Cela signifie qu'il faut parcourir trente-cinq fois l'ensemble d'apprentissage pour $T = 2$ et seulement cinq fois pout $T = 5$.



	Top-5 accuracy cible			
T	35%	40%	45%	48%
1	129	**683**	X	X
2	165	344	**2104**	X
3	**215**	307	886	**7200**
5	213	298	782	3508

FIGURE 3 – Gauche : Exactitude à l'ordre cinq sur ILSVRC-30 en fonction du nombre d'itérations de la SGD. Droite : Nombre d'exemples d'entrainement (en milliers) nécéssaires à la SGD pour atteindre une exactitude donnée. Le symbole 'X' signifie que l'exactitude cible ne peut tout simplement pas être atteinte.

4.6 Comparaison avec l'état de l'art

Il est couteux du point de vue du temps de calcul de sélectionner des transformations sur ILSVRC tout entier, ce pourquoi nous utilisons celles obtenues sur ILSVRC-30. Avec cet ensemble, nous apprenons un classifieur avec toutes les images de ILSVRC. Les résultats sont détaillés dans le Tableau 3. 2-SIT améliore significativement les performances, de 70.1% en fusion à 74.9%. Pour le canal couleur, cette amélioration est encore plus significative (+7%). En comparaison, les gagnants du challenge en 2010, NEC-UIUC-Rutgers obtinrent 71.8%. Nos résultats sont légèrement meilleurs que [23], qui ont 74.3%, notons toutefois que leurs descripteurs sont de dimension 524K, c'est-à-dire seize fois plus grands que les nôtres (32K). Dans de très récents travaux, [14] rapporte une exactitude de 83%, mais en utilisant un autre système de classification (Apprentissage profond).

	SIFT	Color	Fusion
Images d'origine	64.6	57.5	70.1
SIT ($T = 6$)	68.7	64.5	74.8
2-SIT ($T = 6$)	69.2	64.5	**74.9**

TABLE 3 – Exactitude à l'ordre cinq sur ILSVRC tout entier.

4.7 Descripteurs DeCAF

Pour montrer la dépendance des transformations au type de descripteurs, nous rapportons des résultats en utilisant ceux de [10]. En sélectionnant les transformations : symétrie, découpage7, rotation3° jpg70% homothétie50%, l'algorithme affiche une moindre préférence pour les découpages et fait appraître des transformations précédemment peu usitées. Sur ILSVRC tout entier, une exactitude à l'ordre cinq d'origine de 77.9% est améliorée en 81.4% par SIT.

5 Conclusion

L'algorithme proposé, Sélection Itérative de Transformations (SIT), permet de sélectionner efficacement un ensemble de transformations informatives tout en maintenant un temps de calcul modéré. De plus, il permet de traiter élégamment les transformations complexes d'ordre supérieur, compositions de plusieurs transformations de base. SIT améliore sensiblement l'exactitude de la classification avec un petit nombre de transformations. Il est intéressant de constater que les transformations des images au moment de l'entrainement ou du test sont complémentaires et apportent des gains significatifs.

6 Remerciements

Les auteurs souhaitent remercier le soutien financier apporté par le projet ERC "Allegro", le projet CNRS-Mastodons "Gargantua" et le projet ANR Fire-Id.

Références

[1] Yaser S. Abu-Mostafa. Hints. *Neural Computation*, 7(4), 1995.

[2] Yoshua Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2009.

[3] C. Bishop. Training with noise is equivalent to Tikhonov regularization. In *Neural computation*, 1995.

[4] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford UP, 1995.

FIGURE 4 – Exemples d'images de ILSVRC transformées. Transformations sélectionnées avec 2-SIT sur SIFT et couleur.

[5] Léon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In *Advances in Neural Information Processing Systems*, pages 161–168. NIPS Foundation (http ://books.nips.cc), 2008.

[6] M. Chen, Z. Xu, K. Weinberger, and F. Sha. Marginalized denoising autoencoders for domain adaptation. In *ICML*, 2012.

[7] Stephane Clinchant, Gabriela Csurka, Florent Perronnin, and Jean-Michel Renders. XRCE's participation to Imageval. *ImageEval workshop at CVIR*, 2007.

[8] D. DeCoste and M. Burl. Distortion-invariant recognition via jittered queries. In *CVPR*, 2000.

[9] Dennis Decoste and Bernhard Schölkopf. Training invariant support vector machines. *Machine Learning*, 2002.

[10] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf : A deep convolutional activation feature for generic visual recognition. *CoRR*, abs/1310.1531, 2013.

[11] E. Gavves, B. Fernando, C. G. M. Snoek, A. W. M. Smeulders, and T. Tuytelaars. Fine-grained categorization by alignments. In *ICCV*, 2013.

[12] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, 2001.

[13] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE TPAMI*, 2011.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012.

[15] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proc. of the IEEE*, 1998.

[16] Gaëlle Loosli, Stéphane Canu, and Léon Bottou. Training invariant support vector machines using selective sampling. In *Large Scale Kernel Machines*. MIT Press, 2007.

[17] David G Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004.

[18] Laurens Maaten, Minmin Chen, Stephen Tyree, and Kilian Q Weinberger. Learning with marginalized corrupted features. In *ICML*, 2013.

[19] J. Marín, D. Vázquez, D. Gerónimo, and A. López. Learning appearance in virtual scenarios for pedestrian detection. In *CVPR*, 2010.

[20] P. Niyogi, F. Girosi, and T. Poggio. Incorporating prior information in machine learning by creating virtual examples. *Proceedings of the IEEE*, 1998.

[21] Florent Perronnin, Zeynep Akata, Zaïd Harchaoui, and Cordelia Schmid. Towards good practice in large-scale learning for image classification. In *CVPR*, 2012.

[22] L. Pishchulin, A. Jain, C. Wojek, M. Andriluka, T. Thormählen, and B. Schiele. Learning people detection models from few training samples. In *CVPR*, 2011.

[23] J. Sánchez and F. Perronnin. High-dimensional signature compression for large-scale image classification. In *CVPR*, 2011.

[24] Jorge Sánchez, Florent Perronnin, and Teófilo de Campos. Modeling the spatial layout of images beyond spatial pyramids. *Pattern Recognition Letters*, 2012.

[25] Jorge Sánchez, Florent Perronnin, Thomas Mensink, and Jakob J. Verbeek. Image classification with the fisher vector : Theory and practice. *IJCV*, 2013.

[26] B Schölkopf and A J Smola. *Learning with Kernels*. 2002.

[27] J. Sietsma and R. Dow. Creating artificial neural networks that generalize. *Neural Networks*, 1991.

[28] Tinne Tuytelaars and Krystian Mikolajczyk. Local invariant feature detectors : A survey. *Foundations and Trends in Computer Graphics and Vision*, 3(3), 2007.

[29] Vladimir N. Vapnik. *Statistical learning theory*. Wiley, 1998.

[30] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, 2008.

[31] L. Wan, M. Zeiler, S. Zhang, Y. LeCun, and R. Fergus. Regularization of neural networks using dropconnect. In *ICML*, 2013.