Gestion de données manquantes dans des cascades de boosting : application à la détection de visages

Pierre Bouges

N° d'ordre : DU 2303
EDSPIC : 591

UNIVERSITÉ BLAISE PASCAL - CLERMONT II

Ecole Doctorale
Sciences Pour L'Ingénieur De Clermont-Ferrand

Thèse
présentée par :
PIERRE BOUGES

pour obtenir le grade de

DOCTEUR D'UNIVERSITÉ

Spécialité : Informatique

Gestion de données manquantes dans des cascades de boosting : Application à la détection de visages

---

Soutenue publiquement le 6 décembre 2012 devant le jury :

M.	Vincent	CHARVILLAT	Pr INP Toulouse	Président du jury
Mme	Alice	CAPLIER	Pr INP Grenoble	Rapporteur
M.	Christophe	GARCIA	Pr INSA Lyon	Rapporteur
M.	Jean-Luc	DUGELAY	Pr Eurecom	Examinateur
Mlle	Gaëlle	LOOSLI	MCF Univ. Blaise Pascal	Examinateur
M.	Christophe	BLANC	MCF Univ. Blaise Pascal	Examinateur
M.	Thierry	CHATEAU	Pr Univ. Blaise Pascal	Directeur de thèse

Remerciements

La thèse est souvent décrite comme un parcours en solitaire. Il n'en est rien. De nombreuses personnes participent directement ou indirectement à la réussite d'une thèse. Je profite donc de cette page pour remercier les personnes qui ont contribué à cette réussite.

Je débute ces remerciements par mon jury de thèse. Je remercie Vincent Charvillat pour avoir accepté la présidence du jury et pour avoir accepté de faire 4 heures de route suite à un imprévu sur ses billets d'avion. Je remercie aussi Jean-Luc Dugelay pour sa participation. Je remercie particulièrement Alice Caplier et Christophe Garcia pour avoir accepté de rapporter mes travaux. La pertinence de leurs remarques m'ont permis d'avancer dans mes réflexions.

Je continue ces remerciements par mon encadrement de thèse. Je commence par mon directeur de thèse, Thierry Chateau, qui a été celui que j'ai le plus sollicité pendant ces 3 années. Je le remercie pour son accessibilité, sa bonne humeur et pour les conseils et pistes de réflexion qu'il m'a soumis. Je le remercie aussi pour m'avoir fait confiance dans les initiatives que j'ai pris. Je termine en remerciant mes deux autres encadrants, Gaëlle Loosli et Christophe Blanc, pour leurs conseils et leurs soutiens.

Pendant ces 3 années, j'ai pu découvrir avec plaisir le métier d'enseignant. Je tiens à remercier Romain, Benoit et Khouloud qui ont partagé avec moi l'organisation et l'animation des cours, TD et TP de bases de données. J'en profite également pour m'excuser auprès de Romain et Benoit pour les avoir abandonné dans cette tâche pendant ma troisième année de thèse !

Je remercie également toutes les personnes de l'équipe comsee. Merci à ceux qui m'ont éclairé sur des questions de tracking, de reconstruction 3D ou toutes autres techniques de traitement d'image. De façon générale, merci à tous pour la bonne ambiance qui règne au sein de notre couloir. Merci à tous mes collègues du bureau 4016, Laetitia, Manu, Bertrand, Alex ainsi que Alexis, Vadim et Clément. Merci aussi à Christophe pour ses compétences en réanimation d'ordinateurs.

Je n'oublie pas non plus ma famille. Merci à mes parents de m'avoir soutenu tout au long de cette thèse. De façon plus générale, merci à tous les membres de ma famille et belle-famille qui ont pu se déplacer le jour de ma soutenance. Cela m'a fait très plaisir de pouvoir partager ce jour important avec eux.

Pour finir, un grand merci à ma femme Marie qui m'a soutenu dans les moments difficiles et m'a encouragé à continuer d'avancer malgré les échecs. Merci également à ma fille Loélia qui a attendu sagement que je rende mon manuscrit avant de nous rejoindre.

Résumé

Ce mémoire présente les travaux réalisés dans le cadre de ma thèse. Celle-ci a été menée dans le groupe ISPR (*ImageS, Perception systems and Robotics*) de l'Institut Pascal au sein de l'équipe ComSee (*Computers that See*). Ces travaux s'inscrivent dans le cadre du projet Bio Rafale initié par la société clermontoise Vesalis et financé par OSEO. Son but est d'améliorer la sécurité dans les stades en s'appuyant sur l'identification des interdits de stade.

Les applications des travaux de cette thèse concernent la détection de visages. Elle représente la première étape de la chaîne de traitement du projet. Les détecteurs les plus performants utilisent une cascade de classifieurs boostés. La notion de cascade fait référence à une succession séquentielle de plusieurs classifieurs. Le boosting, quant à lui, représente un ensemble d'algorithmes d'apprentissage automatique qui combinent linéairement plusieurs classifieurs faibles. Le détecteur retenu pour cette thèse utilise également une cascade de classifieurs boostés. L'apprentissage d'une telle cascade nécessite une base d'apprentissage ainsi qu'un descripteur d'images. Cette description des images est ici assurée par des matrices de covariance.

La phase d'apprentissage d'un détecteur d'objets détermine ces conditions d'utilisation. Une de nos contributions est d'adapter un détecteur à des conditions d'utilisation non prévues par l'apprentissage. Les adaptations visées aboutissent à un problème de classification avec données manquantes. Une formulation probabiliste de la structure en cascade est alors utilisée pour incorporer les incertitudes introduites par ces données manquantes. Cette formulation nécessite l'estimation de probabilités a posteriori ainsi que le calcul de nouveaux seuils à chaque niveau de la cascade modifiée. Pour ces deux problèmes, plusieurs solutions sont proposées et de nombreux tests sont effectués pour déterminer la meilleure configuration.

Enfin, les applications suivantes sont présentées : détection de visages tournés ou occultés à partir d'un détecteur de visages de face. L'adaptation du détecteur aux visages tournés nécessite l'utilisation d'un modèle géométrique 3D pour ajuster les positions des sous-fenêtres associées aux classifieurs faibles.

Mots-clés : reconnaissance de forme, détection d'objets, apprentissage supervisé, classification, base d'apprentissage, visage, données manquantes, adaptation.

Abstract

This thesis has been realized in the ISPR group (*ImageS, Perception systems and Robotics*) of the Institut Pascal with the ComSee team (*Computers that See*). My research is involved in a project called Bio Rafale. It was created by the compagny Vesalis in 2008 and it is funded by OSEO. Its goal is to improve the security in stadium using identification of dangerous fans.

The applications of these works deal with face detection. It is the first step in the process chain of the project. Most efficient detectors use a cascade of boosted classifiers. The term cascade refers to a sequential succession of several classifiers. The term boosting refers to a set of learning algorithms that linearly combine several weak classifiers. The detector selected for this thesis also uses a cascade of boosted classifiers. The training of such a cascade needs a training database and an image feature. Here, covariance matrices are used as image feature.

The limits of an object detector are fixed by its training stage. One of our contributions is to adapt an object detector to handle some of its limits. The proposed adaptations lead to a problem of classification with missing data. A probabilistic formulation of a cascade is then used to incorporate the uncertainty introduced by the missing data. This formulation involves the estimation of a posteriori probabilities and the computation of new rejection thresholds at each level of the modified cascade. For these two problems, several solutions are proposed and extensive tests are done to find the best configuration.

Finally, our solution is applied to the detection of turned or occluded faces using just an uprigth face detector. Detecting the turned faces requires the use of a 3D geometric model to adjust the position of the subwindow associated with each weak classifier.

Key-words : pattern recognition, object detection, supervised learning, classification, training database, face, missing data, adaptation.

Table des matières

Liste des figures

Liste des tableaux

Notations

$\mathbf{x}$	un vecteur
$\mathbb{1}_{\{pred\}}$	fonction indicatrice qui vaut 1 si le prédicat $pred$ est vrai et 0 sinon
I_i	image numéro i
$\mathcal{S}$	un ensemble d'apprentissage
N	nombre d'exemples dans $\mathcal{S}$
y	un scalaire correspondant à la classe d'un objet
θ	un angle
H	un classifieur fort
h_t	classifieur faible à l'itération t d'un algorithme de boosting
$\omega_t(i)$	poids de l'exemple i à l'itération t d'un algorithme de boosting
ϵ	une erreur d'apprentissage
α	poids d'un classifieur faible
T	nombre de classifieurs faibles
C	une image de caractéristiques
R	une région dans une image
I_{int}	une image intégrale
τ_j	seuil appliqué au classifieur fort du niveau j d'une cascade
d_{min}	taux de détection minimale d'un classifieur fort
f_{max}	taux de faux positifs maximum d'un classifieur fort
K	nombre de niveaux dans une cascade
$\mathcal{S}^p$	ensemble d'images d'apprentissage positives
$\mathcal{S}^n$	ensemble d'images d'apprentissage négatives
$\mathcal{B}$	ensemble d'images de fond ne contenant pas de positifs
d	nombre de caractéristiques
C_R	matrice de covariance de la région R
m_R	moyenne des points de la région R
I_x	dérivée horizontale de l'intensité au premier ordre
I_y	dérivée verticale de l'intensité au premier ordre
I_{xx}	dérivée horizontale de l'intensité au second ordre
I_{yy}	dérivée verticale de l'intensité au second ordre

$p(\mathbf{x})$	probabilité que le vecteur $\mathbf{x}$ représente un visage
fe	facteur d'échelle
Δ_p	nombre de pixels entre deux positions testées successives pour une échelle de 1
s	une échelle
w	largeur d'une fenêtre
h	hauteur d'une fenêtre
h_{d}	classifieur faible disponible
h_{i}	classifieur faible indisponible
$(y_{1:j} = 1)$	évènement $(y_1 = 1, \ldots, y_j = 1)$
β_j	seuil appliqué au classifieur du niveau j d'une cascade probabiliste
R_y	matrice de rotation autour de l'axe y

Chapitre 1

Introduction

Ce premier chapitre présente tout d'abord la notion d'apprentissage artificiel. En vue de bien appréhender les travaux de cette thèse, les concepts de l'apprentissage supervisé sont ensuite détaillés. Puis, les spécificités du domaine d'application visé, à savoir la détection de visages, sont présentées. Ce chapitre se termine par la présentation du plan de ce manuscript et par les contributions de cette thèse.

1.1 De l'apprentissage naturel à l'apprentissage artificiel

La vie d'un enfant est rythmée par de multiples apprentissages. Il apprend tout d'abord à reconnaître l'odeur de sa mère, puis sa voix. Puis, il apprend à marcher et à parler. Une fois à l'école, il apprend à lire. En premier lieu, il apprend par cœur des sons associés à des lettres, puis à des syllabes. Il généralise ensuite à des mots inconnus en identifiant des groupements syllabiques. Cet apprentissage est en partie supervisé par des professeurs qui préparent les tâches d'apprentissage (leçons, exercices) et qui félicitent ou sanctionnent l'enfant en fonction des résultats observés.

Comme l'enfant qui comprend petit à petit le monde qui l'entoure, l'apprentissage artificiel a pour but de comprendre tout phénomène naturel. Et cette compréhension se fait par la construction de modèles à partir de données. Le domaine de la reconnaissance de forme est l'objet de cette étude. Son but est la construction de modèle permettant d'associer une étiquette à une forme. Le terme très générique de « forme » représente les données à traiter. Par exemple, cela pourrait être du texte, les caractéristiques d'un client, les résultats d'un test clinique ou encore du son. Dans le cas de cette thèse, le terme de « forme » fera référence à une image et le terme « étiquette » fera référence au fait que l'image à traiter représente un visage ou pas. En reconnaissance de forme, trois types d'apprentissage existent :

– L'apprentissage supervisé : l'apprentissage est réalisé à partir de nombreux exemples annotés, c'est-à-dire que l'étiquette de chaque exemple d'apprentissage est connue ;
– L'apprentissage non-supervisé : l'apprentissage est réalisé sur des exemples non annotés.

Ici, les méthodes utilisées vont chercher à extraire des invariances au sein des données ;
– L'apprentissage semi-supervisé (ou apprentissage par renforcement) : un premier modèle est appris à partir de quelques exemples annotés. Ce modèle va ensuite être enrichi en incorporant de nouveaux exemples qui auront été préalablement annotés automatiquement.

Les travaux réalisés dans cette thèse utilisent de l'apprentissage supervisé.

1.2 L'apprentissage supervisé

Dans le domaine de la détection d'objet, l'apprentissage supervisé a souvent été utilisé avec succés. Il a permis, par exemple, de détecter des voitures [51, 64, 92], des piétons [45, 72, 10, 70] ou encore des visages [67, 74, 16, 90, 22]. L'ensemble de ces systèmes reposent sur deux étapes : une étape d'apprentissage hors ligne et une étape de classification en ligne.

1.2.1 L'apprentissage hors ligne

La phase d'apprentissage hors ligne se déroule de la façon suivante :

1. Constitution d'une base d'apprentissage. Cette base est composée d'images de l'objet à détecter, représentant l'ensemble des exemples positifs, et d'images de tout ce qui n'est pas l'objet à détecter, représentant l'ensemble des exemples négatifs. Le choix des exemples négatifs n'est pas simple et des techniques ont été développées pour résoudre ce problème (technique de bootstrap notamment) ;
2. Description des exemples d'apprentissage. Un vecteur de descripteurs est associé à chaque exemple d'apprentissage ;
3. Apprentissage d'une fonction de décision. Un algorithme d'apprentissage est appliqué sur l'ensemble d'apprentissage. Chaque élément de cet ensemble est un couple consitué d'un vecteur de descripteurs et d'un label qui représente la classe de l'élément. Généralement, le label 1 représente la classe de l'objet et le label -1 représente la classe non-objet.

Enfin, il faut noter que les algorithmes d'apprentissage peuvent être classés en deux approches :

– Les approches *discriminatives* cherchent à modéliser les relations qui existent entre les entrées (ici, les vecteurs de descripteurs) et les sorties (ici, les labels), tout en faisant un minimum d'hypothèses sur la structure des données d'entrée ;
– Les approches *génératives* cherchent à modéliser la structure des données d'entrée (on cherche souvent à estimer la distribution des données d'entrée). Une fonction de décision est ensuite déduite de ce modèle.

Dans cette thèse, des approches discriminatives sont utilisées. Ces approches sont généralement plus performantes lorsque la base d'apprentissage est conséquente [48]. La figure 1.1 illustrent les différentes étapes d'un apprentissage hors ligne. Pour la dernière étape, la frontière

de décision obtenue est représentée. Elle sépare la zone de la classe positive et la zone de la classe négative.

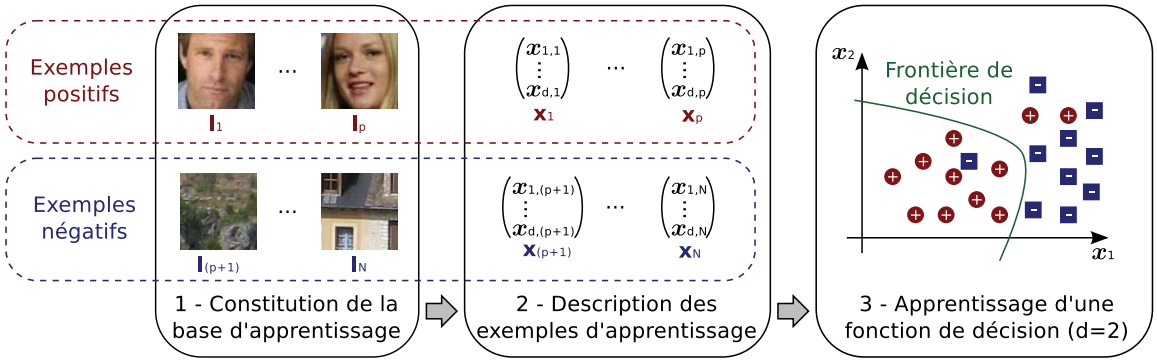


FIGURE 1.1 – Différentes étapes de l'apprentissage hors ligne. Une base d'apprentissage, constituée d'exemples positifs et négatifs, est tout d'abord constituée. Puis, chaque exemple d'apprentissage est décrit par un vecteur de descripteurs. Enfin, un algorithme d'apprentissage est appliqué sur l'ensemble des vecteurs de descripteurs annotés pour produire une fonction de décision. Elle permet de déterminer la frontière de décision entre les deux classes

La base d'apprentissage

Pour que l'apprentissage soit efficace, la base d'apprentissage se doit d'être la plus exhaustive possible. Dans le cas des exemples positifs, cette exhaustivité est approchée en collectant plusieurs milliers d'images de l'objet à détecter. Dans le cas des exemples négatifs, on utilisera plutôt une technique de bootstrap qui consiste à générer des exemples négatifs pertinents en réalisant plusieurs apprentissages. La mise en place de cette technique nécessite de collecter des images ne contenant pas l'objet à détecter. Ces images sont appelées « images de fond » par la suite.

Description des exemples

Une fois la base d'apprentissage constituée, on pourrait utiliser l'image brute comme descripteur. Mais cette approche comporte des inconvénients. Tout d'abord, l'agorithme d'apprentissage risque d'avoir des difficultés à apprendre la fonction de décision du fait de la grande variabilité qu'il peut y avoir au sein des exemples négatifs et positifs. Ensuite, dans le cas où l'algorithme parvient à apprendre une fonction de décision, celle-ci risque d'être lente lors de la phase de classification (pour une image de taille 20×20, la fonction de décision prendrait 400 valeurs en entrée).

Une autre approche consiste à associer un vecteur de descripteurs à chaque image d'apprentissage, puis à appliquer l'algorithme d'apprentissage sur ces vecteurs de descripteurs. Cette

approche a pour but de réduire les variabilités intra-classe tout en augmentant la variabilité inter-classe. Elle permet également d'amener des invariances à certaines caractéristiques des images (par exemple, l'illumination) ou d'obtenir un système plus rapide lors de la phase de classification. La notion de variabilité intra-classe est liée à la répartition spatiale des exemples d'une même classe (positive ou négative) dans l'espace des descripteurs. Dans cet espace, les exemples d'une même classe forment un nuage de points. Si ce nuage est compact, on parle d'une variabilité intra-classe faible alors qu'un nuage étendu est associé à une variabilité intra-classe forte. La variabilité inter-classe est liée à la répartition spatiale des nuages de points des deux classes. Si les deux nuages sont disjoints, on parle d'une variabilité inter-classe forte. À l'opposé, plus les nuages se recouvrent et plus la variabilité inter-classe diminue. De nombreux descripteurs existent. Parmi les plus utilisés, on trouve des ondelettes de Haar [50], les histogrammes de gradients orientés [10] et les Local Binary Pattern (LBP) [91].

L'algorithme d'apprentissage

La description des exemples d'apprentissage permet d'obtenir deux nuages de points dans l'espace des descripteurs. Chacun de ces points est associé à un label (1 ou -1). Le but d'un algorithme d'apprentissage, reposant sur une approche discriminative, est alors de construire une fonction de regression permettant de relier les labels aux vecteurs de descripteurs. Si on note $\{(\mathbf{x}_i, y_i)\}_{i=1,\dots,N}$ un ensemble d'apprentissage à N éléments, où $\mathbf{x}_i$ est un vecteur de descripteurs et y_i le label associé, alors le but de l'algorithme d'apprentissage est de construire une fonction F telle que $y_i = F(\mathbf{x}_i) \quad \forall i$. En pratique, il est généralement impossible d'avoir $y_i = F(\mathbf{x}_i) \quad \forall i$. On cherchera donc la fonction F qui vérifie $y_i = F(\mathbf{x}_i)$ pour le plus grand nombre d'exemples d'apprentissage. Le nombre d'exemples ne vérifiant pas $y_i = F(\mathbf{x}_i)$ permet de définir la notion d'erreur d'apprentissage.

Une fois la fonction de décision définie, on s'intéresse à sa capacité de généralisation, c'est-à-dire sa capacité à prédire des labels corrects sur des exemples inconnus (non utilisés pendant l'apprentissage). Pour cela, on utilise un ensemble de tests contenant des exemples inconnus dont on connait le label. On applique ensuite F sur ces exemples et on compare les labels prédits avec les labels attendus. On peut ainsi calculer une erreur de classification. Si cette erreur est élevée, cela reflète un des deux phénomènes propres à l'apprentissage artificiel : le sous-apprentissage et le sur-apprentissage. Deux causes peuvent expliquer le sous-apprentissage. D'une part, la base d'apprentissage n'est pas assez exhaustive et ne reflète pas assez la réalité. D'autre part, les classes positives et négatives sont trop difficiles à discerner. Ce dernier point se traduit par des nuages de points se chevauchant fortement dans l'espace des descripteurs. Le sur-apprentissage signifie que la fonction apprise F est trop spécifique aux données d'apprentissage. Le sur-apprentissage peut être évité en limitant la complexité de la fonction de décision. Par exemple, il peut être préférable de chercher F sous la forme d'une fonction linéaire plutôt que sous la forme d'une fonction non-linéaire.

1.2.2 La classification en ligne

Une fonction de décision F est désormais à disposition. Lorsqu'on souhaite classifier une image I inconnue, on commence tout d'abord par décrire cette image. On obtient donc un vecteur de descripteurs $\mathbf{x}$. Il suffit ensuite d'appliquer la fonction F sur $\mathbf{x}$ pour obtenir son label $y = F(\mathbf{x})$. Graphiquement, le label prédit y dépend de la position du point $\mathbf{x}$ dans l'espace des descripteurs. Si ce point se trouve dans la zone de classe positive ou sur la frontière de décision, alors le label sera 1. À l'opposé, si le point se trouve dans la zone de la classe négative, alors le label sera -1. Ce processus de classification en ligne est illustré sur la figure 1.2.

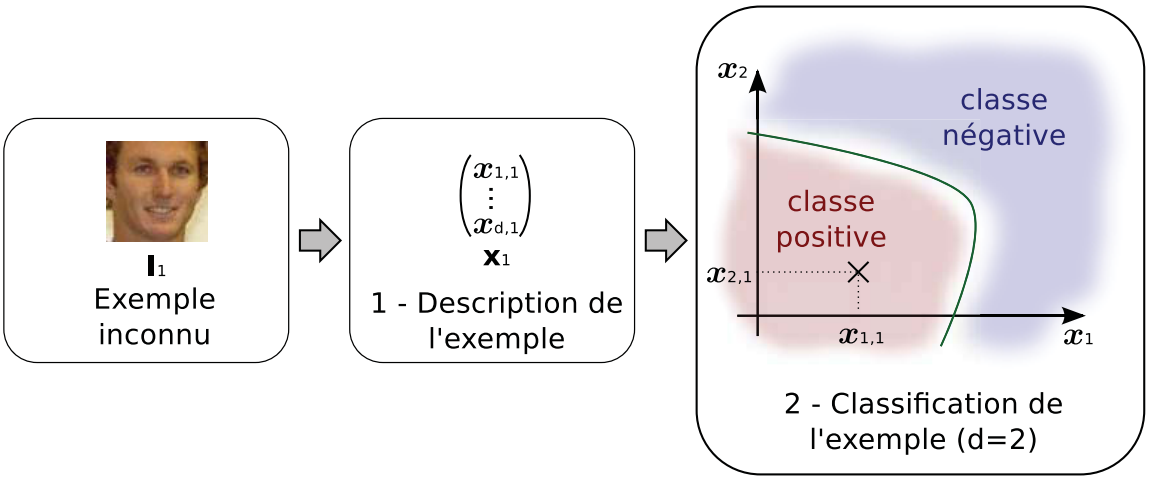


FIGURE 1.2 – Différentes étapes de la classification en ligne. Pour déterminer le label d'une image, on commence par calculer son vecteur de descripteurs. Puis, on applique la fonction de décision sur ce vecteur

Cette section a permis de présenter les différents concepts de l'apprentissage supervisé qui sont mis en jeu dans cette thèse. Les applications de cette thèse couvrent la détection de visage. La section suivante présente les spécificités de ce domaine.

1.3 Les challenges de la détection de visages et les problèmes associés

Les challenges associés à la détection de visages peuvent être attribués aux facteurs suivants :

– **Pose :** les visages peuvent être vus sous différents angles (de face, de profil, à $45°$, ...). D'une pose à l'autre, les relations entre les composantes faciales (yeux, nez, bouche, ...) varient et les composantes faciales peuvent être occultées ou disparaître ;

- **Présence ou abscence de composantes structurelles :** des caractéristiques faciales comme une moustache, une barbe ou des lunettes peuvent être présentes et ces caractéristiques présentent une grande variabilité de forme, de couleur et de taille ;
- **Expression faciale :** les visages peuvent véhiculer différentes émotions comme la peur, la joie ou la surprise. Ces émotions impactent l'apparence d'un visage (bouche ouverte, sourire, sourcils froncés, ...) ;
- **Occultation :** un visage peut être occulté par divers objets comme une écharpe, une casquette ou des lunettes de soleil. D'autres situations entraînent des occultations comme la détection de visages au sein d'une foule (les visages sont alors occultés par d'autres visages) ;
- **Orientation :** un visage peut présenter une rotation par rapport à l'axe optique du système d'acquisition. Par exemple, un visage peut être de face mais retourné (les yeux se retrouvent en bas et la bouche en haut) ;
- **Caractéristiques de l'image :** l'apparence d'un visage est également affectée par différentes caractéristiques liées à l'image comme l'illumination, le contraste, le bruit ou le flou.

La figure 1.3 illustre certains de ces facteurs.

(a) (b) (c) (d)

FIGURE 1.3 – Challenge de la détection de visages. L'apparence d'un visage de face (a) peut être modifée par une expression faciale (b), une occultation (c) ou une illumination particulière (d)

Le but de la détection de visages consiste à déterminer l'éventuelle présence d'un ou de plusieurs visages dans l'image et en cas de présence, à localiser le ou les visages dans l'image. De nombreux problèmes sont proches de la détection de visages :

- **La localisation de visages :** c'est un problème simplifié de la détection de visages dont le but est de localiser un seul visage dans une image. De plus, on suppose que l'image à traiter contient un seul visage ;
- **La détection des composantes faciales :** on souhaite ici déterminer la présence de composantes faciales comme le nez, la bouche, les yeux ou les sourcils. Le but est également de localiser ces composantes faciales ;
- **La reconnaissance faciale :** un visage est comparé à une base de visages et on cherche à savoir si ce visage correspond à un visage de la base ;
- **L'authentification de visages :** le but est de vérifier l'identité d'un visage présent dans

une image ;
- **Le suivi de visages :** on cherche à estimer la position d'un ou de plusieurs visages dans une séquence d'images ;
- **La reconnaissance d'expression faciale :** le but est d'associer un état émotionnel (heureux, triste, énervé, ...) à une image d'un visage.

Dans la plupart de ces problèmes dérivés, la détection de visages constitue la première étape de traitement. Un détecteur de visages sert par exemple à focaliser la recherche des composantes faciales ou permet d'initialiser une nouvelle piste dans le cas du suivi de visages.

1.4 Le contexte

Les travaux de cette thèse sont menés dans le cadre du projet Bio Rafale. Ce projet a débuté en décembre 2008 pour une durée de 4 ans. Il a pour but d'améliorer la sécurité dans les stades en permettant l'identification des interdits de stade. Pour atteindre ce but, les problématiques de détection de visages, de suivi de visages et de reconnaissance faciale doivent être abordées. Cette thèse couvre les problématiques de détection de visages.

1.5 Dans cette thèse

1.5.1 Problématique

La problématique de cette thèse concerne l'adaptation d'un détecteur de visages de face à des conditions d'utilisation critiques. Un détecteur de visages de face est prévu pour fonctionner dans des conditions où les visages à détecter sont de face. Lorsque ce n'est pas le cas (les visages sont tournés ou occultés par exemple), le détecteur est rapidement mis en défaut. Notre problématique consiste donc à détecter des visages qui ne sont pas de face en s'appuyant uniquement sur un détecteur de visages de face. Les challenges de la détection de visages présentés à la section 1.3 représentent différentes conditions d'utilisation critiques pour un détecteur de visages de face. Parmi ces conditions critiques, la détection de visages tournés (challenge de la pose) et la détection de visages occultés constituent les applications principales de cette étude. Des solutions relatives à ces deux problèmes existent et les plus performantes utilisent un apprentissage comme pour un détecteur de visages de face. Malgré de bonnes performances, ces méthodes présentent les inconvénients suivants :

1. Une base d'apprentissage doit être constituée. De nombreuses bases de visages de face sont accessibles en ligne (FERET, AR, CBCL, ...), ce qui facilite la construction des détecteurs de visages de face. Par contre, les bases d'images de visages qui ne sont pas de face sont beaucoup moins nombreuses et parfois, le nombre d'images proposé en ligne est insuffisant pour construire un détecteur associé. Par exemple, si l'on souhaite construire

un détecteur de visages tournés de $+45°$, on peut s'appuyer sur la base FERET mais celle-ci ne propose que 322 images de visages tournés de $+45°$, ce qui est insuffisant pour obtenir un détecteur performant. Il faut donc collecter des images pour constituer une base plus riche, ce qui représente un travail très fastidieux et qui demande généralement beaucoup de temps ;

2. Un apprentissage doit être réalisé. Les techniques les plus performantes dans le domaine de la détection de visages nécessitent généralement un temps d'apprentissage de plusieurs jours, voir plusieurs semaines. De plus, cet apprentissage est souvent réalisé plusieurs fois afin d'obtenir de bonnes performances, ce qui augmente d'autant plus le temps d'apprentissage ;
3. Les méthodes proposées sont spécifiques à un problème donné. La plupart des méthodes dédiées à la détection de visages qui ne sont pas de face fonctionnent sur le même principe : plusieurs détecteurs sont appris et ensuite combinés entre eux. Chaque détecteur est spécialisé pour un problème donné : détecter des visages tournés de $+45°$, détecter des visages occlutés par une écharpe, ... Ces systèmes sont donc difficilement évolutifs.

Notre objectif est de proposer une solution capable de détecter des visages qui ne sont pas de face avec les contraintes suivantes :

1. La base d'apprentissage est seulement constituée de visages de face ;
2. Une fois que le détecteur de visages de face est construit, aucun nouvel apprentissage ne doit être réalisé ;
3. La méthode doit pouvoir s'adapter facilement à de nouvelles conditions d'utilisation.

1.5.2 Contributions

Les contributions de cette thèse sont constituées d'une contribution principale et de deux contributions pratiques.

Gestion de classifieurs faibles manquants dans une cascade

Une cascade de classifieurs boostés est supposée être à disposition. Le problème traité porte sur la gestion de classifieurs faibles manquants dans les différents niveaux de cette cascade. Pour gérer ces classifieurs faibles manquants, la structure de la cascade est modifiée à l'aide d'une formulation probabiliste. Cette formulation prend en compte l'incertitude introduite par l'abscence de certains classifieurs faibles. Ces abscences entraînent également les deux problèmes suivants :

1. Les probabilités a posteriori utilisées à chaque niveau de la cascade ne peuvent plus être calculées. Elles doivent donc être estimées. Trois stratégies d'estimation sont proposées dont une s'appuyant sur l'algorithme des k-plus proches voisins ;

2. Du fait de l'utilisation d'une formulation probabiliste, des nouveaux seuils sont introduits à chaque niveau de la cascade. Pour les estimer, une procédure itérative est proposée. Elle utilise uniquement les données d'apprentissage de la cascade initiale.

Détection de visages tournés par modification de la position des sous-fenêtres associées aux classifieurs faibles

La solution de gestion de classifieurs faibles manquants dans une cascade est appliquée à la détection de visages tournés en s'appuyant uniquement sur un détecteur de visages de face. Le détecteur de visages de face est modifié pour lui permettre de détecter des visages tournés. Une des modifications consiste à ajuster la position de toutes les sous-fenêtres associées à chaque classifieur faible. Pour réaliser cet ajustement, un modèle géométrique 3D simple est utilisé.

Détection de visages occultés par association de McCascades

Le principe de McCascade est également appliqué à la détection de visages occultés en s'appuyant uniquement sur un détecteur de visages de face. Plusieurs cascade d'occultations sont créées. Chacune gère un type d'occultation spécifique et utilise un sous-ensemble des classifieurs faibles appris initialement. Ces cascades sont ensuite combinées et permettent de détecter des visages avec différentes occultations.

1.5.3 Plan du mémoire

Ce manuscript est organisé de la façon suivante :

Chapitre 2 : État de l'art. La détection de visages de face est un problème qui est énormément étudié depuis une dizaine d'années. Ce chapitre propose un état de l'art exhaustif des méthodes développées au cours de ces dix dernières années. En particulier, ce chapitre se focalise sur les solutions développées après les travaux de Viola et Jones [74] ;

Chapitre 3 : Détection de visages de face. Le détecteur retenu pour ces travaux est présenté dans ce chapitre. Des détails sont tout d'abord donnés sur les descripteurs utilisés, à savoir les matrices de covariance. L'algorithme d'apprentissage utilisé est ensuite détaillé. Enfin, une série de tests est menée pour évaluer différentes caractéristiques du détecteur ;

Chapitre 4 : Classification avec données manquantes. Un des objectifs de ces travaux consiste à détecter des visages tournés et des visages occultés en utilisant seulement un détecteur de visages de face. Il en découle la nécessité de devoir gérer des cascades de classifieurs dont certains sont manquants. Ce chapitre se focalise sur les solutions proposées pour gérer ce type de cascade ;

Chapitre 5 : Détection de visages tournés ou occultés. Les solutions présentées au chapitre précédent sont ici appliquées aux deux applications envisagées : la détection de visages tournés et la détection de visages occultés ;

Chapitre 6 : Le projet Biorafale. Dans ce chapitre, le projet constituant le cadre de cette thèse est présenté. Le contexte de ce projet est ensuite abordé ainsi que les différents partenaires et leurs rôles respectifs ;

Chapitre 7 : Conclusion et perspectives. Ce dernier chapitre fait le bilan du travail accompli et propose des pistes de développement à envisager.

Chapitre 2

État de l'art

De nombreuses solutions de détection de visages de face existent. Dans ce chapitre, les méthodes les plus récentes sont présentées. Les méthodes basées apprentissage les plus importantes sont d'abord exposées. Les travaux marquants de Viola et Jones en 2001 [74] sont ensuite détaillés. Ces travaux ont été suivis de nombreuses extensions qui sont présentées à la fin de ce chapitre.

2.1 Vue d'ensemble

Les premiers travaux sur la détection de visages datent du début des années 70, où des heuristiques simples et des techniques basées sur des données anthropométriques étaient utilisées [29, 61, 28]. Il faudra attendre les années 90 pour voir apparaitre des solutions plus robustes. Dès lors, l'intérêt de la communauté pour la détection de visages ne cessera de grandir. Il en découlera un nombre impressionnant de méthodes qui peuvent être classées en deux catégories :

les approches basées connaissances s'appuient explicitement sur les connaissances du visage. Parmi ces connaissances, on peut citer l'utilisation de la couleur de la peau [43] ou l'utilisation de la présence des composantes faciales [87] (la bouche, le nez, les deux yeux, ...) et des relations spatiales entre elles. On peut aussi penser à l'utilisation d'un template prédéfini [7] ou de façon plus poussée, à un template déformable [30] ;

les approches basées apprentissage considèrent le problème de détection de visage comme un problème de reconnaissance de forme à deux classes : visage et non-visage, où chaque classe est représentée par une base d'apprentissage. Ces méthodes s'appuient sur des apprentissages statistiques permettant de construire une fonction de décision qui intègre implicitement les caractéristiques d'un visage.

Parmi ces deux catégories, les approches basées apprentissage ont prouvé leur supériorité en terme de performance et de robustesse. C'est pourquoi nous nous focalisons sur ces approches dans la suite de ce chapitre. Les lecteurs intéressés par un historique complet sur la détection de

visages peuvent se référer à trois études. Les études de Yang et al. [85] et de Hjelmas et Low [20] couvrent les méthodes des années 70 jusqu'au début des années 2000. Enfin, l'étude de Zhang et Zhang [89] permet d'avoir une vue d'ensemble des méthodes développées entre 2000 et 2010.

2.2 Approches basées apprentissage

Dans le cadre de la détection d'objet, les approches basées apprentissage s'appuient sur des images d'apprentissage pour construire un modèle permettant de discriminer des instances de la classe objet par rapport à toutes les instances de la classe non-objet. Les performances de ces méthodes sont entre autres conditionnées par la qualité de la base d'apprentissage qui se doit d'être la plus représentative possible. Autrement dit, cette base doit permettre de capter la variabilité d'apparence présente au sein de la classe objet et non-objet. Dans le cas de cette étude, la classe objet est représentée par des images de visage alors que la classe non-objet correspond à toutes les images qui ne sont pas un visage. La constitution de la base d'apprentissage est un point important et est abordée ultérieurement. Une fois la base d'apprentissage constituée, un algorithme d'apprentissage est appliqué sur l'ensemble des images d'apprentissage ou sur des descripteurs associés. Différents algorithmes d'apprentissage existent comme des SVM [49, 60], des réseaux de neurones [57, 56, 17], des réseaux bayésiens [53] ou des classifieurs bayésiens naïfs [64]. Parmi l'ensemble des solutions proposées, quelques unes des plus significatives sont présentées ci-dessous.

Turk et Pentland [68] ont utilisé l'analyse en composantes principales (ACP) sur les images d'apprentissage de visages pour générer des vecteurs propres qui représentent un sous-espace des visages. Les images d'apprentissage sont ensuite projetées dans ce sous-espace et forment deux clusters (celui des visages et celui des non-visages). La classification d'un exemple se décompose alors en deux phases : 1) projection dans le sous-espace des visages et 2) calcul de la distance entre l'exemple projeté et les deux clusters. La distance minimum donne la classe de l'exemple.

Sung et Poggio [67] ont développé un système basé sur l'estimation de la distribution des visages et des non-visages et sur l'utilisation d'un perceptron multi-couche. Les images d'apprentissages sont d'abord regroupées en clusters (six pour les visages et six pour les non-visages) qui sont approchés par des gaussiennes multidimensionnelles. Une ACP est également appliquée à chaque cluster pour leur associer un sous-espace de plus petite dimension. Un exemple peut ensuite être classé en calculant douze couples de distances en s'appuyant sur ces sous-espaces. Ces douze couples de distances sont finalement passés à un perceptron multicouche (préalablement entraîné) pour classer l'exemple.

Rowley et al. [57] se sont appuyés sur un ensemble de réseaux de neurones entraîné sur les intensités des pixels des visages et des non-visages. Chaque réseau est structuré de façon à pouvoir détecter des caractéristiques faciales comme la bouche ou les yeux. Les scores de chaque réseau sont ensuite fusionnés pour créer la décision finale. Plusieurs stratégies de fusion

sont testées.

Schneiderman et Kanade [64] ont proposé un classifieur bayésien naïf qui estime la probabilité jointe de l'apparence locale et de la position de parties de visages à diférentes résolutions. Ils proposent également un détecteur de visages multivues (face, profil droit et profil gauche) en combinant différents classifieurs, chacun d'eux étant spécialisé pour une vue spécifique.

Roth et al. [56] ont créé le système *SNoW* (Sparse Network of Winnow). Ce système est un réseau épars de fonctions linéaires qui utilise la règle de mise à jour de Winnow [39]. Ce type de réseau est particulièrement adapté pour les problèmes où le nombre de descripteurs prenant part à la décision finale est important. Leur système détecte des visages en s'appuyant sur des descripteurs binaires qui encodent la position et l'intensité des pixels.

L'article le plus marquant dans le domaine de la détection de visage est celui de Viola et Jones [74] en 2001 qui présente le premier détecteur de visages temps-réel. Celui-ci utilise des descripteurs simples ainsi qu'une technique rapide de calcul de ces descripteurs. L'agorithme Adaboost est utilisé à la fois pour sélectionner les meilleurs descripteurs mais aussi pour former plusieurs classifieurs forts associés en cascade pour permettre une exécution en temps-réel. Ces travaux sont les premiers d'une nouvelle catégorie de méthode, nommé approches basées bosting, qui sont détaillées dans la section suivante.

Enfin, un autre système très performant a été proposé par Garcia et Delakis [17] où un réseau de neurones convolutionnels à six couches est utilisé. L'avantage de ce type de réseau est d'extraire automatiquement les caractéristiques importantes dans les premières couches puis de les classer dans les dernières. Leur système est construit pour détecter des visages avec des rotations dans le plan de $\pm 20°$ et tournés de $\pm 60°$ au maximum.

2.3 Approches basées boosting

Les performances du détecteur de Viola et Jones sont dues à la combinaison de trois éléments détaillés par la suite :

1. un apprentissage de classifieurs par une méthode de boosting ;
2. des descripteurs simples et rapides à caculer ;
3. une structure de classifieurs en cascade.

2.3.1 Principe du boosting

Les méthodes de boosting sont des algorithmes d'apprentissage supervisé, i.e. qui utilisent une base d'apprentissage labellisée. Notons $\mathcal{S} \doteq \{(\mathbf{x}_i, y_i)\}_{i=1,\dots,N}$ cette base où $\mathbf{x}_i \in \mathbb{R}^n$ est un vecteur de données représentant un exemple d'apprentissage et $y_i \in \{-1, 1\}$ est le label associé à $\mathbf{x}_i$ (généralement, le label 1 représente la classe des positifs et le label -1 la classe des négatifs). Le but de ces méthodes est de construire un classifieur $H(\mathbf{x}) : \mathbb{R}^n \to \{-1, 1\}$ permettant d'associer un label à un exemple inconnu $\mathbf{x}$. Les méthodes issues du boosting se basent

sur l'observation suivante : il est rare d'avoir à sa disposition un expert omniscient permettant de prendre la meilleure décision et par conséquent, on a plutôt recours à un comité d'experts plus ou moins compétents pour ensuite combiner leurs avis et prendre une décision. De manière étonnante, des recherches en apprentissage artificiel datant du début des années 90 montrent qu'il est possible d'atteindre une décision aussi précise que souhaitée par une combinaison judicieuse d'experts imparfaits mais correctement entraînés. Plusieurs algorithmes d'apprentissage ont été développés à la suite de ces travaux. Les méthodes issues du boosting sont des méthodes capables de générer des règles de décision précises en utilisant des règles de décision produites par des classifieurs faibles, i.e. des classifieurs ayant un taux de réussite un peu meilleur que le hasard.

Le premier algorithme de boosting a été proposé par Schapire en 1990 [62] et permet d'obtenir un classifieur après avoir entraîné un classifieur faible sur trois sous-ensembles d'apprentissage. En 1997, Freund et Schapire [12] proposent une amélioration de l'algorithme proposé par Schapire en 90 qui est aujourd'hui très utilisé : l'algorithme *Adaboost*. Nous présentons ici l'algorithme Adaboost dit « discret ». L'algorithme, présenté dans l'algorithme 1, se base toujours sur l'idée d'utiliser un comité d'experts pour prendre une décision et rajoute deux autres idées :

1. La pondération adaptative des votes par une technique de mise à jour multiplicative ;
2. La modification de la distribution des exemples disponibles pour entraîner chaque expert, en surpondérant au fur et à mesure les exemples mal classés.

Comme pour les SVM (Machine à Vecteurs supports [71]), les méthodes de boosting sont issues de considérations théoriques, ce qui permet de connaître certaines propriétés. En particulier, on sait que l'erreur d'apprentissage diminue exponentiellement avec t si le classifieur faible est un peu meilleur que le hasard (cela correspond à avoir $\epsilon_t < 0,5 \, \forall t$). Le comportement de l'erreur en généralisation est plus difficile à prédire. Il est cependant observé empiriquement que celle-ci tend à diminuer même après que l'erreur d'apprentissage soit devenue nulle. Pour avoir une vue d'ensemble sur les méthodes de boosting et notamment sur les propriétés de l'algorithme adaboost, on peut se référer à l'article [13] de Freund et Schapire. Les algorithmes *Adaboost discret*, *Adaboost réel* [63], *LogitBoost* [14] et *Gentle Adaboost* [14] sont les variantes les plus connues.

2.3.2 Descripteurs de type ondelette de Haar

Dans les premières approches basées apprentissage, l'algorithme d'apprentissage était entraîné avec l'ensemble des intensités des pixels des images d'apprentissage [68, 67, 57]. Une autre démarche consiste à utiliser des descripteurs. Le terme « descripteur » désigne tout vecteur de données $\mathbf{x} \in \mathbb{R}^n$ permettant de décrire une image ou une partie de celle-ci (par exemple, la moyenne des intensités des pixels d'une image représente un descripteur). Détaillons le processus de calcul d'un descripteur. Soit I une image en niveaux de gris de taille $W \times H$ et C une

Algorithme 1 : L'algorithme Adaboost Discret

Entrées : $\mathcal{S} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ une base d'apprentissage.
Sorties : $\text{sign}(H(\mathbf{x}))$ où $H(\mathbf{x})$ est un classifieur fort entraîné sur $\mathcal{S}$.

$\omega_1(i) = 1/N, \quad \forall i = 1, \ldots, N$;
pour $t = 1$ **à** T **faire**
  Normaliser les poids $\omega_t(i) = \dfrac{\omega_t(i)}{\sum_{i=1}^{N} \omega_t(i)}$;
  Estimer un classifieur faible $h_t(\mathbf{x}) : \mathbb{R}^n \to \{-1, 1\}$ sur $\mathcal{S}$ en utilisant les poids ω_t ;
  Calculer l'erreur d'apprentissage $\epsilon_t = \sum_{i=1}^{N} \omega_t(i) \mathbb{1}_{\{y_i \neq h_t(\mathbf{x}_i)\}}$;
  Calculer $\alpha_t = ln\left(\dfrac{1-\epsilon_t}{\epsilon_t}\right)$;
  Mise à jour des poids : $w_{t+1}(i) = \omega_t(i) e^{\alpha_t \mathbb{1}_{\{y_i \neq h_t(\mathbf{x}_i)\}}}$;
finpour
Le classifieur final est donné par : $\text{sign}(H(\mathbf{x})) = \text{sign}\Big(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})\Big)$;

image de caractéristiques de taille $W \times H \times d$ extraite à partir de I :

$$C(x, y) = \Phi(I, x, y) \tag{2.1}$$

où Φ est une fonction qui associe à chaque pixel de coordonnées (x, y) de I un ensemble de d caractéristiques ($C(x, y, i)$ est donc la i-ème caractéristique du pixel (x, y)). Une caractéristique est une donnée caractérisant un pixel comme son intensité, sa couleur dans l'espace RGB ou son gradient. Une fois C calculé, on peut calculer n'importe quel vecteur de descripteurs $\mathbf{x}$ associé à une région R de I :

$$\mathbf{x} = f(C, R) \tag{2.2}$$

où R est défini par les coordonnées du coin supérieur gauche, une largeur et une hauteur. Le type de descripteur utilisé est ici représenté par f. Ce processus est illustré sur la figure 2.1.

Dans leurs travaux, Viola et Jones proposent d'utiliser comme caractéristique l'intensité des pixels de I, on a donc $C(x, y) = I(x, y)\ \forall (x, y)$. Ils proposent également des descripteurs dérivés des ondelettes de Haar utilisés précédemment par Papageorgiou et al. [50]. Ces descripteurs sont basés sur des sommes d'intensité de pixels présent dans des zones rectangulaires adjacentes de même taille. Des descripteurs à deux, trois et quatre rectangles sont utilisés (illustrés sur la figure 2.2). En faisant varier leurs tailles et leurs positions, Viola et Jones obtiennent un ensemble de descripteurs potentiels. Ils utilisent ensuite Adaboost pour sélectionner à chaque itération le meilleur descripteur. En effet, à chaque itération, un classifieur faible est calculé pour chaque

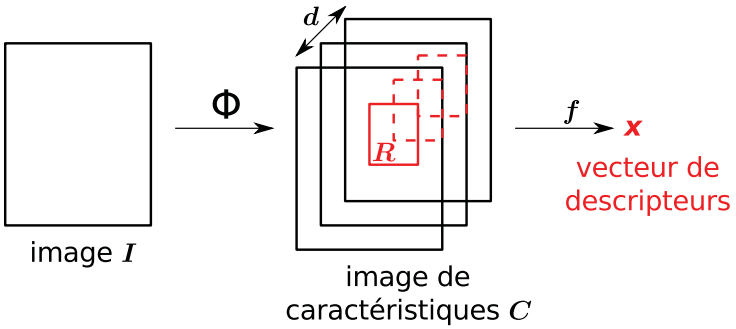


FIGURE 2.1 – Processus de calcul des descripteurs : une image de caractéristiques C est tout d'abord calculée à l'aide la fonction Φ, puis on associe un vecteur de descripteurs $\mathbf{x}$ à une région R de C à l'aide de la fonction f

descripteur et celui présentant l'erreur minimum est conservé. Un classifieur faible est ici défini par un simple seuillage sur la valeur du descripteur. Un classifieur fort est donc une somme de fonctions de seuillage où chaque fonction correspond à un descripteur de Haar.

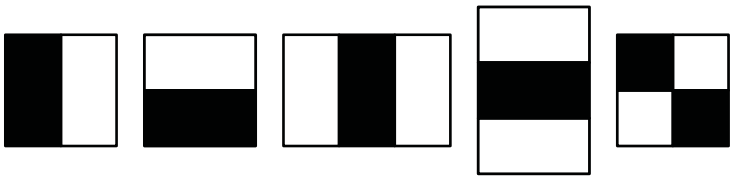


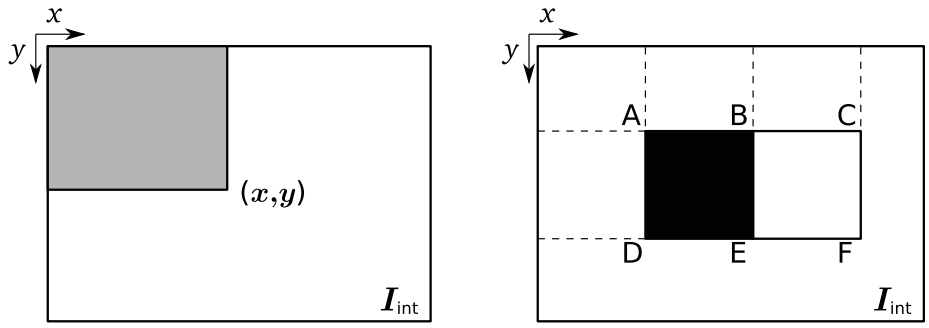
FIGURE 2.2 – Descripteurs de Haar. La valeur de chaque descripteur de Haar est égale à la somme des intensités des pixels présents dans les rectangles blancs auquelle on soustrait la somme des intensités des pixels des rectangles gris

S'inspirant des « Summed-Area Tables » présentés par Crow [9], Viola et Jones proposent une représentation intermédiaire de l'image, appelée image intégrale (voir figure 2.3(a)) et notée I_{int}, pour calculer rapidement les descripteurs de Haar :

$$I_{\text{int}}(x,y) = \sum_{x' \leq x, y' \leq y} I(x',y') \tag{2.3}$$

En utilisant cette représentation, chaque descripteur peut être calculé en un nombre constant d'accès mémoire (entre six et neuf). Par exemple, la valeur du descripteur de Haar de la figure 2.3(b) est donnée par :

$$I_{\text{int}}(x_F,y_F) + 2 * I_{\text{int}}(x_B,y_B) - I_{\text{int}}(x_C,y_C) - 2 * I_{\text{int}}(x_E,y_E) - I_{\text{int}}(x_A,y_A) + I_{\text{int}}(x_D,y_D) \tag{2.4}$$



(a) $I_{\text{int}}(x, y)$ est égal à la somme des intensités des pixels de la zone grisée

(b) Le calcul de ce descripteur de Haar par image intégrale nécessite six accès mémoire seulement

FIGURE 2.3 – (a) Principe de l'image intégrale et (b) calcul d'un descripteur de Haar

2.3.3 Structure en cascade

Lorsque l'on souhaite tester la présence de visages dans une image, le concept de fenêtre glissante est utilisé. Le principe consiste à tester différentes fenêtres de l'image dont la taille et la position varient. Ces fenêtres sont ensuite passées à un classifieur pour déterminer la présence éventuelle d'un visage. On peut ainsi imaginer entraîner un classifieur à l'aide de l'algorithme Adaboost et appliquer ce classifieur sur toutes les fenêtres potentielles. Le but est alors d'obtenir un taux de vrais positifs très élevés (>0,9) tout en ayant un taux de faux positifs très faible ($<10^{-6}$). Le taux de vrais positifs correspond au nombre de visages correctement classés par rapport au nombre total de fenêtres testées et le taux de faux positifs correspond au nombre de fenêtres négatives (i.e. ne contenant pas de visages) classées comme visage par rapport au nombre total de fenêtres testées. Pour atteindre cet objectif, une solution consiste à augmenter le nombre de classifieurs faibles du classifieur fort. En contrepartie, la classification de toutes les fenêtres de l'image devient très couteuse en temps de calcul.
Pour gérer le compromis entre performance et temps de cacul, Viola et Jones proposent d'utiliser plusieurs classifieurs forts successifs organisés en cascade. Les classifieurs forts des premiers niveaux sont chargés de rejeter les négatifs les plus simples alors que les classifieurs forts des derniers niveaux essayent de discriminer les visages des négatifs les plus difficiles (ceux qui ressemblent aux positifs). Le classifieur fort du niveau j est de la forme :

$$\text{sign}(H_j(\mathbf{x}) - \tau_j) = \text{sign}(\sum_{t=1}^{T_j} \alpha_{jt} h_{jt}(\mathbf{x}) - \tau_j) \tag{2.5}$$

où τ_j est un seuil fixé pendant l'apprentissage pour obtenir un taux de vrais positifs très élevé d_{min} (de l'ordre de 0,99) tout en assurant un taux de faux positifs modeste f_{max} (de l'ordre de 0,5). Le taux de détection D d'une cascade de K niveaux ainsi que son taux de faux positifs F

peuvent ainsi être estimés par :

$$D = \prod_{j=1}^{K} d_{\text{min}}, \quad F = \prod_{j=1}^{K} f_{\text{max}} \tag{2.6}$$

Par exemple, un taux de détection de 0,9 et un taux de faux positifs de 10^{-6} peuvent être atteint avec une casacade de 10 niveaux où $d_{\text{min}} = 0,99$ ($0,99^{10} \approx 0,9$) et $f_{\text{max}} = 0,3$ ($0,3^{10} \approx 6 \times 10^{-6}$).

La classification d'un fenêtre consiste alors à appliquer les classifieurs forts successifs tant que la fenêtre est classée comme visage. Dès qu'un classifieur fort classe la fenêtre comme non-visage, le processus s'arrête. Cette approche, illustrée sur la figure 2.4, permet de rejeter la majorité des fenêtres dans les premiers niveaux de la cascade, ce qui assure un temps d'exécution faible, tout en conservant un maximum de visages, ce qui assure des performances élevées. Une cascade est ainsi définie par l'ensemble des classifieurs $\{\text{sign}(H_1(\mathbf{x}) - \tau_1), \ldots, \text{sign}(H_K(\mathbf{x}) - \tau_K)\}$.

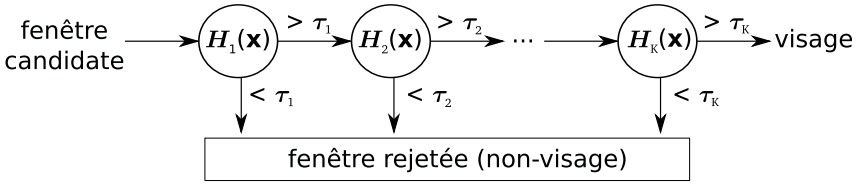


FIGURE 2.4 – Principe d'une structure en cascade où plusieurs classifieurs forts H_j sont associés. À chaque niveau, la fenêtre candidate est classée par un classifieur fort qui, soit la rejette (le processus de classification s'arrête), soit l'accepte (le processus de classification continue)

Chaque niveau de cascade utilise les mêmes images de visage tandis que les images de négatifs sont renouvelées au début de chaque niveau par une technique de bootstrap. Avant l'apprentissage du niveau j, la cascade de classifieurs des niveaux1 à $j-1$ est appliquée sur un ensemble d'images de fond ne contenant aucun visage et les faux positifs générés sont utilisés comme négatifs au niveau j. L'algorithme 2 présente le processus d'apprentissage d'une cascade exposé par Viola et Jones [74]. Dans le contexte de la détection de visages, il n'est pas concevable de constituer un ensemble de validation du fait de la structure très complexe de l'espace des négatifs. L'évaluation de la cascade sur un ensemble de validation a donc été remplacée par la contrainte suivante : pour chaque niveau de cascade, des classifieurs faibles sont ajoutés jusqu'à ce que le taux de positifs bien classés soit au moins de d_{min} et que le taux de négatifs mal classés soit au maximum de f_{max}.

2.4 Extensions des approches basées boosting

Suite aux travaux de Viola et Jones, de nombreuses améliorations vont apparaître. Ces améliorations peuvent être classées suivant trois axes :

Algorithme 2 : L'algorithme d'apprentissage d'une cascade de classifieurs boostés

Entrées :
- f_{max} : taux maximum de faux positifs par niveau ;
- d_{min} : taux minimum de détection par niveau ;
- F_{target} : taux de faux positifs à atteindre à la fin de l'apprentissage ;
- $\mathcal{S}^p$: ensemble d'images d'apprentissage positives ;
- $\mathcal{S}^n$: ensemble d'images d'apprentissage négatives ;
- $\mathcal{B}$: ensemble d'images de fond ne contenant pas de positifs.

Sorties : $\{\text{sign}(H_1(\mathbf{x}) - \tau_1), \ldots, \text{sign}(H_K(\mathbf{x}) - \tau_K)\}$, une cascade de classifieurs boostés

$F_0 = 1.0$;
$D_0 = 1.0$;
$j = 0$;
si $\mathcal{S}^n = \emptyset$ **alors**
    Remplir $\mathcal{S}^n$ avec des zones aléatoires provenant des images de $\mathcal{B}$;
finsi
tant que $F_j > F_{target}$ **faire**
    $j = j + 1$;
    $T_j = 0$;
    $F_j = F_{j-1}$;
    tant que $F_j > f_{max} \times F_{j-1}$ **faire**
        $T_j = T_j + 1$;
        En utilisant $\mathcal{S}^p$ et $\mathcal{S}^n$, entraîner un classifieur $\text{sign}(H_j(\mathbf{x}) - \tau_j)$ constitué de T_j classifieurs faibles à l'aide d'un algorithme de boosting;
        Appliquer le classifieur en cascade courant sur un ensemble de validation pour déterminer le taux de faux positifs F_j et le taux de détection D_j;
        Ajuster le seuil τ_j jusqu'à ce que le classifieur en cascade courant ait un taux de détections d'au moins $d_{\text{min}} \times D_{j-1}$ (cela affecte aussi F_j);
    fintq
    $\mathcal{S}^n = \emptyset$;
    si $F_j > F_{target}$ **alors**
        Appliquer le classifieur en cascade courant sur les images de $\mathcal{B}$ et remplir $\mathcal{S}^n$ avec les fausses détections;
    finsi
fintq

1. Modification de l'algorithme d'apprentissage ;
2. Modification de l'ensemble des descripteurs utilisé ;
3. Modification de la structure en cascade.

2.4.1 Les algorithmes d'apprentissage alternatifs

La détection de visages est considérée comme une détection d'évènements rares. De façon général, le nombre de visages dans une image est extrèmement faible. Il en découle que lors de la phase d'apprentissage, le nombre d'images de visages est très inférieur au nombre d'images de fonds. Pour prendre en compte cette asymétrie dans l'apprentissage et donner plus d'importance aux positifs, Viola et Jones [73] ont proposé l'algorithme *Asymboost*.

En 2002, Lienhart et al. [35] ont effectué une étude comparative des algorithmes Adaboost discret, Adaboost réel et Gentle Adaboost dont il ressort qu'il est préférable d'utiliser Gentle Adaboost. Ce résultat sera par la suite nuancé par Brubaker et al. [4].

Un nouvel algorithme de boosting, nommé *FloatBoost*, est proposé par Lin et al. [33]. Après certaines itérations de boosting, le classifieur faible le moins pertinent est retiré du classifieur fort. Le but est d'obtenir les même performances que l'agorithme Adaboost en utilisant moins de classifieurs faibles et ceci, au prix d'un temps d'apprentissage augmenté.

Wu et al. [80] ont travaillé sur une nouvelle approche pour créer les classifieurs des différents niveaux d'une cascade. Ils proposent de découpler la phase de sélection des descripteurs et la phase de combinaison de ces descripteurs pour obtenir un classifieur fort. La première phase est assuré par l'algorithme *Forward Feature Selection* (FFS) tandis que la deuxième phase est assurée par l'algorithme *Linear Asymmetric Classifier* (LAC) qui prend en compte le caractère asymétrique du problème d'apprentissage de visage. Leur stratégie permet d'obtenir des cascades performantes tout en reduisant grandement le temps d'apprentissage. Pham et Cham [52] ont également travaillé sur la réduction du temps d'apprentissage. En s'appuyant sur des statistiques calulées sur les pondérations des données d'apprentissage, ils parviennent à fortement diminuer le temps d'apprentissage d'un classifieur faible.

2.4.2 Présentation des autres ensembles de descripteurs de Haar

Une autre façon d'améliorer les performances d'une approche basée boosting consiste à étendre l'expressivité des descripteurs utilisés. En ce sens, Lienhart et Maydt [36] ont proposé d'enrichir l'ensemble des descripteurs de Haar considéré en y incorporant des ondelelettes tournées (voir figure 2.5(a)). Ils proposent également d'utiliser une image intégrale tournée pour calculer rapidement ces descripteurs.

Viola et Jones [27] ont remarqué que les ondelelettes utilisées dans leurs premiers travaux [74] ne permettaient pas d'apprendre correctement des visages qui ne sont pas de face. Pour combler cette limitation, ils proposent des descripteurs diagonaux (voir figure 2.5(b)) qui se

calculent en seize accès mémoire. Zhang et al. [90] ont également mis en avant que les négatifs générés dans les derniers niveaux de cascade devenaient très difficiles à classer en utilisant les descripteurs de Haar proposés par Viola et Jones. Ils proposent donc une structure en cascade où les premiers niveaux sont appris sur les ondelelettes de Haar (description locale des images) alors que les derniers niveaux sont appris sur des informations provenant d'une ACP (description globale des images).

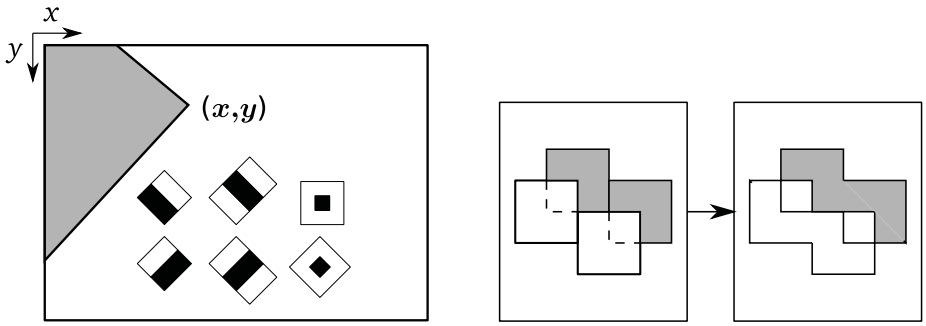


FIGURE 2.5 – (a) Descripteurs de Haar tournés et image intégrale tournée proposés par Lienhart et Maydt [36] et (b) Un des descripteurs de Haar diagonal proposé par Viola et Jones [27]

Dans le but de capter les caractéristiques non symétriques des visages qui ne sont pas de face, Li et Zhang [33] ont utilisé un ensemble de descripteurs rectangulaires non adjacents. Chaque rectangle est de taille $x \times y$ et des distances (dx, dy) les séparent les uns des autres (voir figure 2.6(a)).

Mita et al. [44] ont proposé des descripteurs de Haar joints (voir figure 2.6(b)). Chaque classifieur faible est ici appris sur plusieurs descripteurs de Haar. Basé sur les réponses de chaque descripteur, un mot binaire est formé et le classifieur faible est entraîné sur la valeur de ces mots en base 10.

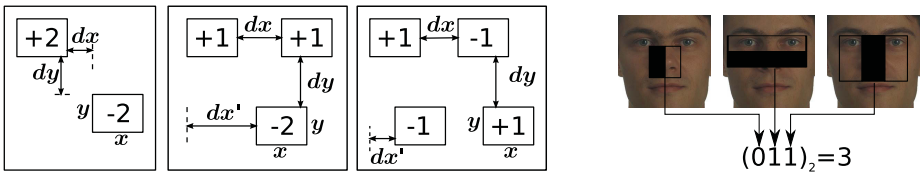


FIGURE 2.6 – (a) Descripteurs de Haar non adjacents proposés par Li et Zhang [33] et (b) Un exemple de descripteurs de Haar joint proposé par Mita et al. [44]

Pour finir, on peut citer certains descripteurs binaires appliqués avec succés au problème de détection de visages comme les descripteurs issus de la transformation modifié de Census [15], les Local Binary Pattern (LBP) [91] ou encore les Locally Assembled Binary (LAB) [83]. On

peut également citer les histogrammes de gradients orientés de Dalal et Triggs [10] utilisés dans le domaine de la détection de piétons.

2.4.3 Présentation des architectures en cascade différentes

Les deux limitations du sytème de Viola et Jones [74] concernent le temps important nécessaire à l'apprentissage d'une cascade et le choix des paramètres d'une cascade. Dans la formulation classique d'une cascade, les classifieurs forts de chaque niveau sont indépendants. Plusieurs travaux ont ensuite montré qu'il était préférable de les ré-utiliser dans l'entrainenemt de chaque niveau. Xiao et al. [82] ont proposé le concept de *chaîne de boosting*. À chaque niveau de cascade, le classifieur fort précédemment appris est intégré au classifieur du niveau courant. Huang et al. [23] ont ensuite proposé la struture de *cascade imbriquée*. Au lieu d'intégrer le classifieur fort précédemment appris, ils utilisent le score de classification du niveau précédent pour former le premier classifieur faible du niveau courant. Enfin, Yan et al. [84] ont proposé le principe de *descripteurs accumulés* où les descripteurs sélectionnés au niveau précédent sont utilisés pour entraîner les premiers classifieurs faibles du niveau courant. Ces trois concepts permettent de diminuer grandement le temps d'apprentissage.

En vue de fixer les paramètres d'une cascade, Brubaker et al. [3] ont introduit un critère pour sélectionner le seuil de chaque niveau ainsi que le nombre de classifieurs faibles de chaque niveau. Ce critère est basé sur un modèle probabiliste des performances globales d'une cascade. Xiao et al. [81] ont proposé le concept de *cascade dynamique* qui suppose que le taux de faux positifs change exponentiellement à chaque niveau.

Enfin, d'autres structures de cascade ont été proposées comme les *cascades souples* de Bourdev et Brandt [2]. Une cascade est ici une longue succession de classifieurs faibles où chaque classifieur faible est associé à un seuil de rejet. L'ensemble des seuils de rejet est calculé à l'aide d'un algorithme de calibration une fois que les classifieurs faibles sont tous sélectionnés. Enfin, Le et Satoh [31] ont proposé d'associer des cascades apprises sur des descripteurs de Haar avec plusieurs SVM. Ils arrivent ainsi à accélérer le temps de détection nécessaire pour scanner une image.

2.5 Bilan

Dans ce chapitre, différents systèmes de détection de visages ont été présentés. En particulier, les systèmes apparus après les travaux de Viola et Jones [74] ont été détaillés. Ces derniers ont établi les bases de la majorité des détecteurs de visages récents :

- Apprentissage par un algorithme de boosting. En plus d'apprendre une fonction de décision, ce type d'algorithme permet également de sélectionner les descripteurs les plus pertinents parmi un ensemble de descripteurs potentiels ;
- Des descripteurs rapides à calculer. Les ondeletettes de Haar sont majoritairement utilisées car elles sont très rapides à calculer ;

– Une structure en cascade. Pour accélérer le temps de classification, plusieurs classifieurs sont séquentiellement associés.

Chapitre 3

Détection de visages de face

Dans ce chapitre, le détecteur de visages retenu dans le cadre de cette thèse est abordé en détail. Les descripteurs utilisés, à savoir des matrices de covariance, sont tout d'abord présentés. Des détails sur l'algorithme d'apprentissage sont ensuite donnés. Enfin, une série de tests est menée pour évaluer différents aspects du détecteur.

3.1 Détection de visages par matrices de covariance

Les matrices de covariance constituent un autre type de descripteurs récemment appliqué à la détection de piétons [70]. Nous avons choisi d'utiliser ce descripteur pour nos travaux pour trois raisons :

1. Ses performances en détection d'objet sont parmi les meilleures actuellement (ce point est abordé ultérieurement dans les expérimentations) ;
2. Il est rapide à calculer. En effet, le principe des images intégrales a été étendu aux matrices de covariance ;
3. Il est évolutif. En effet, les caractéristiques utilisées pour calculer les matrices de covriance peuvent être facilement remplacées par de nouvelles caractéristiques plus pertinentes.

3.1.1 Les matrices de covariance comme descripteurs

L'utilisation des matrices de covariance comme descripteur a été proposée par Tuzel et al. [69]. Pour une région rectangulaire donnée $R \subset C$, où C est l'image des caractéristiques, on note $\{\mathbf{z}_i\}_{i=1,\ldots,N}$ l'ensemble des points de R (chaque point est un vecteur de dimension d). On peut alors représenter la région R par une matrice de covariance C_R de taille $d \times d$:

$$C_R = \frac{1}{N-1} \sum_{i=1}^{N} (\mathbf{z}_i - \boldsymbol{\mu})(\mathbf{z}_i - \boldsymbol{\mu})^{\mathrm{T}} \tag{3.1}$$

où $\boldsymbol{\mu}$ est le point moyen des $\mathbf{z}_i$.
Dans le cadre de la détection de piétons, Tuzel et al. [70] ont proposé d'utiliser les caractéristiques suivantes :

$$\Phi(I,x,y) = \left(x\ y\ |I_x|\ |I_y|\ \sqrt{I_x^2 + I_y^2}\ \arctan\frac{|I_y|}{|I_x|}\ |I_{xx}|\ |I_{yy}| \right)^{\mathrm{T}} \tag{3.2}$$

où I_x, I_y, I_{xx} et I_{yy} représentent les dérivées de l'intensité au premier et au second ordre, $\sqrt{I_x^2 + I_y^2}$ représente la norme du gradient et $\arctan\frac{|I_y|}{|I_x|}$ représente son orientation. Basée sur ces caractéristiques, la région R peut donc être décrite à l'aide d'une matrice de covariance de taille $d \times d$. Cette matrice encode des informations sur les variances des caractéristiques et leurs corrélations entre elles. Elle encode également la disposition spatiale des caractéristiques dans la région car la corrélation avec la position des pixels (x, y) est aussi calculée. Cette matrice étant symétrique, on peut la représenter à l'aide d'un vecteur de taille $\frac{d(d+1)}{2}$ (en réalité, seuls 33 éléments sont retenus car les variances en x, y et $x \times y$ sont ignorées). Pour gagner en efficacité, Tuzel et al. [69] ont également étendu le principe d'image intégrale aux matrices de covariance. Ce principe est détaillé en annexe A. Par construction, ces matrices sont robustes aux changements constants d'illumination. Pour ajouter une robustesse aux changements linéaires locaux de l'illumination, Tuzel et al. [70] appliquent la normalisation suivante : soit r une sous-fenêtre à l'intérieur d'une fenêtre R. Les matrices C_R et C_r sont d'abord calculées, puis la matrice de covariance normalisée $C_{r'}$ est calculée par :

$$C_{r'}(i,j) = \frac{C_r(i,j)}{\sqrt{C_R(i,i)C_R(j,j)}} \tag{3.3}$$

3.1.2 L'algorithme Logitboost

Pour l'apprentissage des classifieurs forts, nous utilisons l'algorithme *LogitBoost*. Celui-ci a été utilisé par Tuzel et al. [70] dans leur système de détection de piétons. Cet algorithme est une variante de l'algorithme Adaboost et a été proposé par Friedman et al. [14]. À la différence de l'algorithme Adaboost discret présenté, les labels y_i des exemples d'apprentissage $\mathbf{x}_i$ appartiennent à l'ensemble $\{0, 1\}$ (1 est la classe positive). De plus, on représente la probabilité d'un exemple $\mathbf{x}$ d'appartenir à la classe positive par :

$$p(\mathbf{x}) = \frac{e^{F(\mathbf{x})}}{e^{F(\mathbf{x})} + e^{-F(\mathbf{x})}} \text{ où } F(\mathbf{x}) = \frac{1}{2}\sum_{t=1}^{T} h_t(\mathbf{x}) \tag{3.4}$$

LogitBoost apprend itérativement l'ensemble des classifieurs faibles $\{h_t\}_{t=1,\ldots,T}$ en minimisant la log-vraissemblance binomiale négative des données d'apprentissage :

$$-\sum_{i=1}^{N}\Big[y_i\, ln(p(\mathbf{x}_i)) + (1-y_i)\, ln(1-p(\mathbf{x}_i))\Big] \tag{3.5}$$

À chaque itération t, cela est obtenu en résolvant un problème de régression par moindres carrés pondérés :

$$\sum_{i=1}^{N} \omega_t(i)||h_t(\mathbf{x}_i) - z_i||^2 \tag{3.6}$$

où $z_i = \frac{y_i - p(\mathbf{x}_i)}{p(\mathbf{x}_i)(1-p(\mathbf{x}_i))}$ sont des réponses attribuées aux exemples d'apprentissage et les poids sont donnés par $\omega_t(i) = p(\mathbf{x}_i)(1 - p(\mathbf{x}_i))$. Ces poids sont proches de 0 pour les exemples dont la probabilité est proche de 0 ou de 1 et ils sont maximums pour les exemples dont la probabilité est 0,5, i.e. ceux qui sont difficiles à classer. L'algorithme LogitBoost est présenté dans l'algorithme 3. Dans notre cas, la base d'apprentissage est représentée par l'ensemble $\{(I_i, y_i)\}_{i=1,\ldots,N}$ où I_i représente une image d'apprentissage (visage ou non-visage) et l'étape d'estimation d'un classifieur faible (voir ligne 4) est remplacée par l'estimation de plusieurs classifieurs faibles. Dans cette étape, chaque classifieur faible testé est appris en calculant les matrices de covariance d'une sous-fenêtre r dans toutes les images I_i. Le classifieur faible le plus performant est ensuite conservé, i.e. on conserve celui minimisant L_r, la log-vraissemblance binomiale négative calculée sur la sous-fenêtre r dans les images I_i. Tester toutes les sous-fenêtres possibles à chaque itération de LogitBoost rendrait l'apprentissage extrêmement long. C'est pourquoi un nombre N_{sf} de sous-fenêtres sont aléatoirement sélectionnées. Pour éviter de considérer des sous-fenêtres trop petites, il est imposé que la largeur minimale doit être d'au moins 1/10 de la largeur des images I_i. La même contrainte est appliquée sur la hauteur des sous-fenêtres. L'algorithme LogitBoost permet donc à chaque itération d'apprendre un classifieur faible mais également, de retenir la zone la plus discriminante parmi l'ensemble des images d'apprentissage I_i.

3.1.3 Ajout de la moyenne et apprentissage sur des sous-ensembles de caractéristiques

Le système de Tuzel et al. [70] a par la suite été perfectionné par Yao et Odobez [86]. Diverses améliorations ont été proposées :

- Corrections des distorsions dans les images provenant de caméras grand angle ;
- Remplacement des dérivées secondes dans les caractéristiques par deux valeurs issues d'une technique d'extraction de fond ;
- Ajout de la moyenne des caractéristiques dans le vecteur de descripteurs ;
- Apprentissage sur des sous-ensembles de caractéristiques.

Parmi ces améliorations, les deux dernières sont utilisées dans ces travaux.

Ajout des moyennes des caractéristiques

Les matrices de covariance fournissent les moments d'ordre deux des caractéristiques. Yao et Odobez proposent d'utiliser également les moments d'ordre un, à savoir les moyennes des

Algorithme 3 : L'algorithme LogitBoost

Entrées : $\mathcal{S} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ une base d'apprentissage avec $\mathbf{x}_i \in \mathbb{R}^n$ et $y_i \in \{0, 1\}$.

Sorties : $\text{sign}(H(\mathbf{x}))$ où $H(\mathbf{x})$ est un classifieur fort entraîné sur $\mathcal{S}$.

1 $\omega_1(i) = 1/N,\ p(\mathbf{x}_i) = 1/2,\ F(\mathbf{x}_i) = 0 \quad \forall i = 1, \ldots, N$;

2 **pour** $t = 1$ **à** T **faire**

3 Calculer les réponses $z_i = \dfrac{y_i - p(\mathbf{x}_i)}{p(\mathbf{x}_i)(1 - p(\mathbf{x}_i))}$ et les poids $\omega_t(i) = p(\mathbf{x}_i)(1 - p(\mathbf{x}_i))$;

4 Estimer un classifieur faible $h_t(\mathbf{x})$ en résolvant le problème de régression par moindres carrés pondérés $\displaystyle\sum_{i=1}^{N} \omega_t(i)||h_t(\mathbf{x}_i) - z_i||^2$;

5 Mise à jour de $F(\mathbf{x}_i) = F(\mathbf{x}_i) + \dfrac{1}{2}h_t(\mathbf{x}_i)$ et de $p(\mathbf{x}_i) = \dfrac{e^{F(\mathbf{x}_i)}}{e^{F(\mathbf{x}_i)} + e^{-F(\mathbf{x}_i)}}$;

6 **finpour**

7 Le classifieur final est donné par : $\text{sign}(H(\mathbf{x})) = \text{sign}\Big(\displaystyle\sum_{t=1}^{T} h_t(\mathbf{x})\Big)$;

caractéristiques. En notant m_R le vecteur des moyennes des caractéristiques d'une région R, on a :

$$m_R = \frac{1}{N}\sum_{i=1}^{N} \mathbf{z}_i \tag{3.7}$$

Le vecteur de descripteurs d'une sous-fenêtres r est donc constitué par la concaténation des éléments de la matrice $C_{r'}$ et des éléments du vecteur m_r. Les expérimentations de Yao et Odobez montrent que l'ajout des moyennes permet d'améliorer légèrement les performances et permet surtout d'améliorer le temps d'exécution du détecteur d'environ 30%.

Apprentissage sur des sous-ensembles de caractéristiques

Pour accélérer le temps de détection, Yao et Odobez proposent de ne pas calculer les matrices de covariance sur l'ensemble des caractéristiques mais plutôt de les calculer sur des sous-ensembles de caractéristiques de taille 2, 3 ou 4 parmi les d caractéristiques possibles. Ainsi, un classifieur faible est appris sur une sous-fenêtre r mais également sur un sous-ensemble de caractéristiques. Pour chaque sous-fenêtre r, la procédure d'apprentissage consiste à estimer un classifieur faible par sous-ensemble de caractéristiques et à conserver le plus performant. Pour chaque taille m de sous-ensembles, il existe $\mathrm{C}_m^d = \frac{d!}{m!(d-m)!}$ sous-ensembles possibles. Par exemple, il existe 28 sous-ensembles de 2 caractéristiques choisis parmi 8 caractéristiques. Pour les sous-ensembles de taille 2, une recherche exhaustive est effectuée, c'est-à-dire que les 28

sous-ensembles sont testés. Pour $m > 2$, cela devient trop coûteux en temps de calcul de tester l'ensemble des sous-ensembles possibles. Pour pallier ce problème, Yao et Odobez proposent de tout d'abord calculer les classifieurs faibles de chaque sous-ensemble de taille 2, puis d'utiliser les performances de ces classifieurs faibles pour estimer les k meilleurs sous-ensembles de taille m (ceux susceptibles d'amener de bonnes performances en classification). Les k meilleurs sous-ensembles sont ensuite testés. Cette procédure est décrite plus en détails dans l'annexe B. Les tests effectués par Yao et Odobez montrent que l'utilisation des sous-ensembles de caractéristiques permet de grandement accélérer le temps de détection mais permet également des performances légèrement meilleures.

3.2 Évaluation des performances

3.2.1 Mesure des performances

Pour une base de test donnée, les performances d'un détecteur de visage sont caractérisées par son taux de détections TD et par le nombre de faux positifs générés $nbFP$. Le couple ($nbFP$, TD) représente le point de fonctionnement du détecteur. Le taux de détections représente le nombre de visages qui sont détectés par rapport au nombre total de visages contenus dans la base de test. Le nombre de faux positifs correspond au nombre de mauvaises détections, i.e. le nombre de fenêtres classées en visage alors que ce ne sont pas des visages. Ces deux critères sont liés. En effet, augmenter (resp. diminuer) le taux de détections TD implique généralement d'augmenter (resp. de diminuer) le nombre de faux positifs $nbFP$. C'est pourquoi deux détecteurs de visages sont comparés par un ensemble de points de fonctionnement. Un ensemble de points $\{(nbFP_i, TD_i)\}$ représente une courbe FROC (pour Free Receiver Operator Characteristic). Si la base de test consiste à tester N fenêtres, alors l'abscisse d'une courbe FROC sera compris entre 0 et N tandis que l'ordonnée sera compris entre 0 et 1. Un classifieur parfait aurait comme point de fonctionnement $(0, 1)$. Un détecteur de visages sera plus performant qu'un autre si sa courbe FROC est au-dessus de l'autre courbe FROC. Dans le cas de cascade de classifieurs boostés, une courbe FROC s'obtient en appliquant plusieurs fois le classifieur sur la base de test et en modifiant à chaque fois le seuil de décision τ_K du dernier niveau. Augmenter ce seuil permet d'obtenir des points où TD et $nbFP$ sont faibles (le classifieur est conservateur) alors que le diminuer permet d'augmenter TD et $nbFP$. Cette augmentation se poursuit jusqu'à un certain point de la courbe. En effet, certains visages n'atteignent pas le dernier niveau, de même que de nombreux négatifs sont rejetés avant le dernier niveau. Pour obtenir plus de points, il faut supprimer le dernier niveau de la cascade. Ensuite, on applique à nouveau le classifieur en modifiant à chaque fois le seuil τ_{K-1} du dernier niveau courant. On répète ces opérations (suppression du dernier niveau + modification du seuil) jusqu'à obtenir suffisamment de points. Une courbe très similaire est la courbe ROC (Receiver Operator Characteristic) dont l'abscisse représente le taux de faux positifs et l'ordonnée le taux de détections. Les courbes ROC ne sont pas appropriées pour comparer plusieurs détecteurs de visages car le

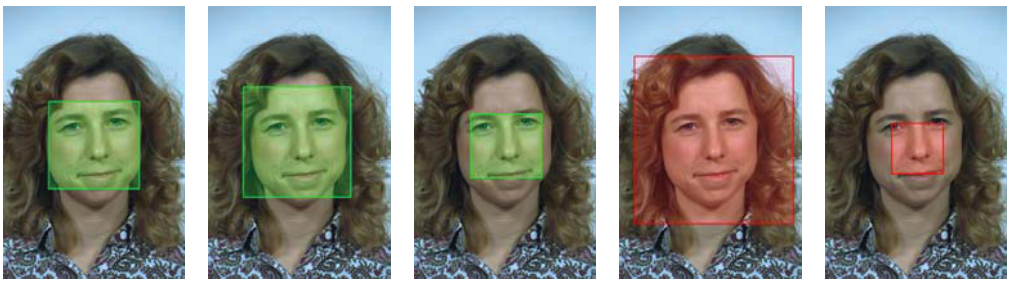
taux de faux positifs dépend du nombre de fenêtres testées. Ainsi, on peut imaginer que deux détecteurs A et B aient la même coube ROC mais que le détecteur A ait testé plus de fenêtres pour obtenir ces résultats. Au final, le détecteur A produira plus de faux positifs que le système B. Si les deux systèmes testent le même nombre de fenêtres, une courbre ROC est aussi pertinente qu'une courbe FROC.

3.2.2 Critère de validation d'une bonne détection

Pour permettre une comparaison équitable entre plusieurs détecteurs, il est nécessaire de définir un critère de validité d'une bonne détection. Une bonne détection est une détection qui correspond à la vérité terrain (la vérité terrain représente les zones d'une image qui contiennent des visages). Le but d'un critère de validation est de traduire mathématiquement la bonne correspondance entre une détection et une vérité terrain. Le critère utilisé s'appuie sur le recouvrement entre la détection et la vérité terrain et a été proposé par Yao et Odobez [86]. Ce recouvrement est calculé à l'aide de la F-mesure $F_{\text{recouvrement}}$:

$$F_{\text{recouvrement}}(VT, D) = \frac{2\rho\pi}{\rho + \pi} \quad \text{où} \quad \rho = \frac{|VT \cap D|}{|VT|} \quad \text{et} \quad \pi = \frac{|VT \cap D|}{|D|} \tag{3.8}$$

ρ représente la zone de précision et π la zone de rappel. VT représente la vérité terrain et D la détection. Enfin, l'opérateur $|R|$ représente le nombre de pixels dans la région R. Une détection correspond à la vérité terrain si $F_{\text{recouvrement}} > 0,5$. La figure 3.1 présente un exemple de verité terrain ainsi que deux détections acceptées et deux détections refusées.



(a) (b) (c) (d) (e)

FIGURE 3.1 – (a) vérité terrain (b)-(c) exemples de deux détections acceptées ($F_{\text{recouvrement}}$ est supérieur à 0,5) (d)-(e) exemples de deux détections rejetées ($F_{\text{recouvrement}}$ est inférieur à 0,5)

3.3 Méthodologie expérimentale

3.3.1 Les données d'apprentissage

Les performances d'un détecteur de visages dépendent fortement de la base d'apprentissage utilisée. Celle-ci se doit d'être la plus exhaustive possible. Pour les visages, cette exhaustivité est approchée en considérant plusieurs milliers d'images de visages différents. Dans le cas des négatifs en revanche, il est très difficile de constituer une base d'images exhaustives du fait de l'énorme variabilité au sein de cette classe (tout ce qui n'est pas un visage est un négatif potentiel). Pour résoudre ce problème, on utilise des milliers d'images ne contenant aucun visage et on applique une technique de bootstrap pendant l'apprentissage pour générer des négatifs pertinents (voir section 2.3.3).

La base des positifs

De nombreuses bases d'images de visages peuvent être utilisées comme base d'apprentissage comme la base FERET [54], la base AR [42] ou la base CBCL [1]. Nous avons choisi d'utiliser la base *Labeled Faces in the Wild-a* [77], notée LFW-a, pour la grande variété d'images de visage qu'elle propose. Cette base contient les images en niveau de gris de la base *Labeled Faces in the Wild* [25], noté LFW, après avoir été alignées par un logiciel commercial. La base LFW est constituée de 13233 images de visages de 5749 personnes différentes dont 1680 ont plus d'une image dans la base. Le but premier de cette base est de tester des algorithmes de reconnaissance faciale. Pour créer la base de positifs, une seule image par personne a tout d'abord été sélectionnée, ce qui représente 5749 images. Quand une personne dispose de plusieurs images, la première est simplement retenue. Par exemple, Aaron Peirsol dispose de 4 images, l'image « Aaron_Peirsol_0001.jpg » est conservée. Chacune de ces images est de taille 250×250 et représente un visage sur un arrière plan. Les visages ont ensuite été extraits en suivant les conclusions de Viola et Jones [74], confirmées ensuite par Cristinacce [8]. Ces dernières indiquent qu'il est préférable de considérer l'ensemble du visage au lieu de se focaliser sur la région des caractéristiques faciales. En pratique, l'extraction des visages s'est faite en considérant la région dont le coin supérieur gauche est $(56, 68)$ et dont la taille est 130×130. Enfin, chaque image est redimensionnée à la taille 24×24. Le processus de création d'un exemple positif est décrit sur la figure 3.2. Chaque image a ensuite été inspectée et les images des visages qui n'étaient pas de face ont été supprimées. Sur les 5749 images de visages, seules 3300 ont été conservées. Pour permettre de détecter des visages non centrés, des positifs artificiels sont créés en déplaçant chaque positif de quelques pixels vers la gauche, la droite, le haut et le bas, ce qui donne quatre positifs artificiels pour chaque positif initial. On obtient donc une base de $3300 + 3300 \times 4 = 16500$ positifs.

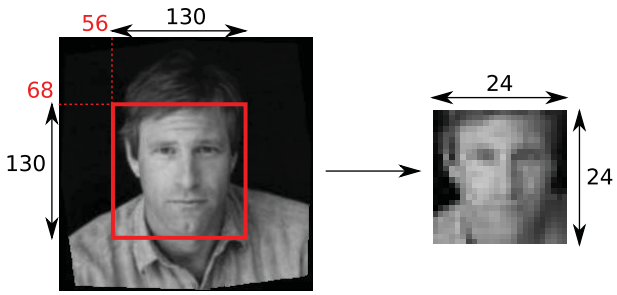


FIGURE 3.2 – Pour chaque image de la base LFW-a, un exemple positif est créé en extrayant la zone de taille 130×130 dont le coin supérieur gauche est $(56, 68)$. Puis la zone extraite est redimensionnée à 24×24

La base des négatifs

5795 images ne contenant aucun visage ont été collectées. En considérant des zones de différentes tailles et à différentes positions dans ces images, des millions de négatifs peuvent être générés. Cette génération de négatifs est effectuée pendant l'apprentissage d'une cascade au début de chaque nouveau niveau. Parmi l'ensemble des images possibles, les images représentant des corps de personnes sont pertinentes. Une partie de ces négatifs a donc été créée en utilisant des images de piétons de la base INRIA [10]. Pour chaque piéton, l'image du corps est conservée pour devenir une image de génération de négatifs. Un exemple de ce processus est donné sur la figure 3.3.

(a) (b) (c) (d)

FIGURE 3.3 – En extrayant les corps de piétons (a), on peut créer des images ne contenant pas de visages (b), (c) et (d) qui seront pertinentes pour la génération de négatifs

3.3.2 Les ensembles de tests

La base AR

La base AR [42] contient plus de 4000 images couleur de 126 personnes prises dans des conditions d'illumination contrôlées et sur fond blanc. Cette base a été créée pour tester la robustesse des algorithmes de reconnaissance faciale. Les images varient suivant les critères suivant :

- l'expression : neutre, sourire, peur et cri ;
- l'illumination : éclairage droit, gauche et des deux côtés ;
- les occultations : présence d'écharpe et de lunettes de soleil.

Dans cette thèse, cette base a été utilisée pour tester la robustesse des détecteurs de visages aux occultations. Seules les images de visages occultés par une écharpe (765 images) et par des lunettes de soleil (765 images) ont été considérées. La figure 3.4 montre deux exemples d'images de visages occultés.

(a) (b)

FIGURE 3.4 – (a) visage occulté par des lunettes de soleil (b) visage occulté par une écharpe

La base FERET couleur

Cette base contient 11338 images de visages dans des conditions d'illumination contrôlées et avec un arrière-plan uniforme. Ces images ont été obtenues en photographiant 994 personnes sous différents angles de vue et à différents âges. Au total, 15 sessions de prise de vue ont été organisées entre 1993 et 1996. Cette base est, en grande partie, une version couleur de la base FERET originale proposée par Phillips et al. [54]. Cette base est, à l'origine, destinée à tester les performances des algorithmes de reconnaissance faciale. Elle n'est, en général, pas utilisée pour tester des détecteurs de visages de face du fait de sa simplicité. Cependant, elle fournit des images de visages tournés de différents angles ($\pm 15°$, $\pm 22.5°$, $\pm 45°$, $\pm 67.5°$ et $\pm\ 90°$). Ainsi, elle devient une base intéressante pour tester la robustesse d'un détecteur de visages de face ou pour tester un détecteur de visages multivue. Des exemples de visages tournés de cette base sont donnés sur la figure 3.5.

(a) (b) (c) (d)

FIGURE 3.5 – Exemple de visages tournés de la base FERET. Le visage est tourné de $90°$ en (a), de $67.5°$ en (b), de $45°$ en (c) et de $22.5°$ en (d)

La base CMU-MIT

La base CMU-MIT est une base publique [1] qui a été utilisée à de nombreuses reprises pour comparer différents détecteurs. Elle a été créée par Rowley et al. [57]. Les images de la base contiennent des visages de face avec des arrière-plans complexes et très variés. Les conditions d'illumination varient d'une image à l'autre et les images sont généralement de mauvaise qualité. Tout cela fait que la base est une base de test difficile. Elle contient 130 images avec 507 visages au total. Les images sont divisées en trois groupes : A, B et C. Les ensembles A et C ont été collectés par Rowley et al. [57] tandis que l'ensemble B a été fourni par Sung et Poggio [67]. Cet ensemble B est aussi connu sous le nom de base MIT (la base CMU-MIT correspond à une extension de la base MIT). La figure 3.6 montre quelques images de la base.

En plus de la base CMU-MIT, il existe deux ensembles d'images dédiés aux tests des détecteurs invariants aux rotations et aux détecteurs multivues (face et profil). Le premier ensemble est l'ensemble D constitué de 50 images contenant 223 visages avec diverses rotations dans le plan. Cet ensemble a été constitué par Rowley et al. [58] pour tester leur détecteur de visages invariant aux rotations. Enfin, il existe un dernier ensemble de test, nommé « CMU profil », pour les détecteurs multivues. Celui-ci a été proposé par Schneiderman et Kanade [64]. Cet ensemble contient 208 images avec 441 visages dont 347 de profil.

3.3.3 Scan de l'image

Pour détecter des visages dans une image, le concept de fenêtre glissante est utilisé. Cela consiste à appliquer le détecteur sur différentes fenêtres de l'image. Chaque fenêtre est passée au détecteur qui la classifie en visage ou non-visage. Pour détecter des visages de différentes tailles, deux approches sont possibles :

1. l'image est itérativement sous-dimensionnée tandis que la taille de la fenêtre à tester reste constante. C'est une approche basée *pyramide d'images* ;

1. téléchargeable à l'adresse : http ://vasc.ri.cmu.edu//idb/html/face/frontal_images/index.html

FIGURE 3.6 – Quelques images de la base CMU-MIT

2. la fenêtre à tester est redimensionnée et la taille de l'image reste constante. C'est une approche basée *fenêtre multi-échelle*.

Quand le calcul des descripteurs ne dépend pas de la taille de la fenêtre à tester (on parle alors de technique invariante à l'échelle), l'approche basée fenêtre multi-échelle est à priviligier car elle est plus rapide. Les descripteurs basés sur des images intégrales comme les ondelettes de Haar ou les matrices de covariance peuvent être calculés en temps constant quelque soit la taille de la fenêtre. L'approche basée fenêtre multi-échelle est donc retenue. Cette approche comporte les paramètres suivants :

- $(w_{\min}, h_{\min})$: la taille minimale de la fenêtre à tester. Cette taille minimale correspond à la taille des exemples d'apprentissage (24×24 dans notre cas) ;
- fe : le facteur d'échelle qui représente le ratio entre deux échelles successives ;
- Δ_p : le nombre de pixels entre deux positions testées successives pour une échelle de 1. Ce nombre s'applique pour des déplacements horizontaux ou verticaux. Le nombre réel de pixels entre deux positions successives dépend de l'échelle : si s est l'échelle courante, alors le déplacement sera de $[s\Delta_p]$ où $[.]$ représente l'opérateur d'arrondi.

Le scan d'une image de taille $W \times H$ commence avec une fenêtre de taille $(w_0, h_0) = (w_{\min}, h_{\min})$ positionnée en haut à gauche de l'image. L'échelle initiale est $s_0 = 1$. Une fois testée, la fenêtre est déplacée en fonction de $[s_0\Delta_p]$. Une fois toutes les positions testées, l'échelle devient $s_1 = fe \times s_0$ et la taille devient $(w_1, h_1) = ([s_1 w_0], [s_1 h_0])$ et on teste à nouveau toutes les positions possibles. De façon générale, on a :

$$s_i = fe \times s_{i-1} \ , \quad w_i = [s_i w_{i-1}] \ , \quad h_i = [s_i h_{i-1}] \tag{3.9}$$

Ce processus continue tant que $w_i \leq W$ et que $h_i \leq H$.

3.3.4 Fusion des détections multiples

Après avoir scanné une image par fenêtre glissante, on observe généralement que chaque visage de l'image est associé à plusieurs détections. Une étape de post-traitement est donc nécessaire pour fusionner ces détections multiples. La solution proposée par Yao et Odobez [86] est ici utilisée. Soit $\{D_p\}_{p=1,\dots,P}$, l'ensemble des détections et soit (x_p^c, y_p^c) le centre de la boîte englobante de la détection D_p. Pour chaque détection, Yao et Odobez définissent la fiabilité d'une détection comme :

$$f_{\text{det}}(D_p) = \sum_{j=1}^{K} \Big((f_{\text{max}})^{K-j} \times p_j(\mathbf{x}_p) \Big) \tag{3.10}$$

où $\mathbf{x}_p$ est le vecteur de descripteurs de la détection D_p et $p_j(\mathbf{x}_p)$ est la probabilité de $\mathbf{x}_p$ d'appartenir à la classe visage au niveau j de la cascade (voir équation (3.4)). Une image de fiabilité F_{det} est ensuite construite :

$$F_{\text{det}}(x_p^c, y_p^c) = f_{\text{det}}(D_p) \quad \forall p = 1, \dots, P \tag{3.11}$$

Pour les positions (x, y) ne contenant pas de détections, on fixe $F_{\text{det}}(x, y)$ à 0. De plus, si deux détections (de tailles différentes) ont le même centre, alors seul le maximum des deux valeurs de fiabilité est conservé. L'image de fiabilité est ensuite lissée par un noyau gaussien de paramètres (σ_w, σ_h) qui sont égaux à 1/10 de la taille moyenne des détections. Tout maximum local de l'image lissée est considéré comme une détection potentielle. La taille de chaque détection potentielle est obtenue en faisant une moyenne pondérée des tailles des détections présentes dans leur voisinnage. La pondération dépend de la fiabilité de chaque détection et de leur distance au maximum local. Soit $\hat{D}_k$ une détection potentielle avec $(\hat{x}_k^c, \hat{y}_k^c)$ les coordonnées du centre de sa boîte englobante et soit $\{(x_{k,i}^c, y_{k,i}^c)\}_{i=1,\dots,P_k}$ l'ensemble des centres des boîtes englobantes contenues dans le voisinnage de $\hat{D}_k$ ainsi que $\{(w_{k,i}^c, h_{k,i}^c)\}_{i=1,\dots,P_k}$ leurs tailles associées. La taille finale $(\hat{w}_k, \hat{h}_k)$ de la boîte englobante de $\hat{D}_k$ est donnée par :

$$\hat{w}_k = \frac{1}{A} \sum_{i=1}^{P_k} \lambda_i w_{k,i}^c \quad \hat{h}_k = \frac{1}{A} \sum_{i=1}^{P_k} \lambda_i h_{k,i}^c \quad \text{où} \quad A = \sum_{i=1}^{P_k} \lambda_i \tag{3.12}$$

avec :

$$\lambda_i = \frac{F_{\text{det}}(x_{k,i}^c, y_{k,i}^c)}{\sqrt{(\hat{x}_k^c - x_{k,i}^c)^2 + (\hat{y}_k^c - y_{k,i}^c)^2}} \tag{3.13}$$

Une dernière étape est appliquée sur les détections potentielles. Chaque détection potentielle est triée en fonction de sa fiabilité. Ensuite, une détection potentielle est supprimée si elle recouvre trop une autre détection potentielle avec une fiabilité plus élevée. On commence par examiner la détection potentielle avec la plus faible fiabilité et on considère que deux détections D_1 et D_2 se recouvrent trop si $F_{\text{recouvrement}}(D_1, D_2) > 0.4$.

3.3.5 Le détecteur utilisé

Le détecteur utilisé comporte 10 niveaux. Chaque niveau est entraîné avec 5000 positifs et 5000 négatifs et chaque niveau est conçu pour détecter au moins 99,8% des positifs tout en faisant au plus 50% de faux positifs. À chaque itération de LogitBoost, 100 sous-fenêtres étaient testées. Pour générer des négatifs pendant l'apprentissage, 2453 images de fond ont été utilisées. Enfin, chaque classifieur faible est appris sur un sous-ensemble de deux caractéristiques.

3.4 Résultats de la détection de visages de face

3.4.1 Le choix des matrices de covariance

Pour quantifier les performances des matrices de covariance, il est nécessaire d'utiliser un protocole de test standardisé. Dans le domaine de la détection de visages, un tel protocole n'existe pas. Il existe des ensembles de tests comme l'ensemble CMU-MIT mais il n'existe pas d'ensemble d'apprentissage associé. Comparer deux détecteurs sur la même base de test n'est pas suffisant car les performances d'un détecteur sont grandement liées à l'apprentissage. Nous avons choisi d'utiliser le protocole de test proposé par Munder et Gavrila [47] dans le domaine de la détection de piétons. Trois ensembles d'apprentissage sont fournis : $\mathcal{A}_1$, $\mathcal{A}_2$ et $\mathcal{A}_3$ ainsi que deux ensembles de tests : $\mathcal{T}_1$ et $\mathcal{T}_2$. Chaque ensemble d'apprentissage est constitué de 4800 exemples positifs, 5000 exemples négatifs et plus de 1200 images de fond. Chaque ensemble de test est constitué de 4800 positifs et 5000 négatifs. Le test d'un classifieur nécessite trois apprentissages :

1. Apprendre un classifieur $C_{1,2}$ sur $\mathcal{A}_1 \cup \mathcal{A}_2$ et utiliser $\mathcal{A}_3$ comme ensemble de validation ;
2. Apprendre un classifieur $C_{1,3}$ sur $\mathcal{A}_1 \cup \mathcal{A}_3$ et utiliser $\mathcal{A}_2$ comme ensemble de validation ;
3. Apprendre un classifieur $C_{2,3}$ sur $\mathcal{A}_2 \cup \mathcal{A}_3$ et utiliser $\mathcal{A}_1$ comme ensemble de validation ;

Pour augmenter le nombre de négatifs, on peut également utiliser les images de fond pendant l'apprentissage. Ensuite, chacun des classifieurs est appliqué sur $\mathcal{T}_1$ et $\mathcal{T}_2$. On obtient finalement six courbes ROC. Les moyennes et les écarts-types de différents points sont calculés pour produire la courbre ROC finale. Trois cascades de classifieurs boostés ont été créées. Chacune d'entre elles possède les caractéristiques suivantes :

- La cascade est constituée de 8 niveaux ;
- Au niveau j, des classifieurs faibles sont ajoutés jusqu'à ce que le taux de vrais positifs sur la base de validation soit au moins de $0,995^j$ avec un taux de faux positifs d'au plus $0,5^j$;
- Chaque niveau est appris avec l'ensemble des 8 caractéristiques définies à la section 3.1.1 ;
- Pour limiter la durée de l'apprentissage, la technique de boostrap n'a pas été utilisée.

Les performances de la cascade avec matrices de covariance sont ensuite comparées à celles d'une cascade avec des ondelettes de Haar, d'un SVM avec des ondelettes de Haar et d'un SVM

avec des LRF (*Local Receptive Fields*). Les descripteurs LRF sont des descripteurs spécifiques à l'objet détecté qui nécessite l'apprentissage d'un réseau de neurones. Dans leur article, Munder et Gavrila notent que la combinaison SVM avec des LRF présente les meilleurs résultats. Les résultats présentés sur la figure 3.7 montrent qu'une cascade avec des matrices de covariance se situe au niveau des meilleures performances. Seule l'utilisation d'un SVM avec des LRF présente des résultats légèrement meilleurs. Rappelons toutefois que l'utilisation des LRF est plus contraingnante car elle nécessite l'apprentissage d'un réseau de neurones.

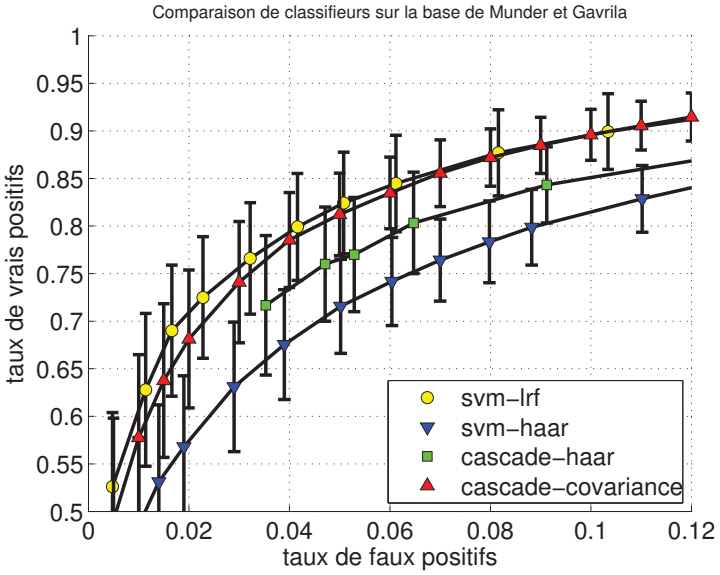


FIGURE 3.7 – Courbes ROC de différents classifieurs sur la base de Munder et Gavrila. La cascade avec des matrices de covariance est comparée à une cascade avec des ondelettes de Haar, un SVM avec des ondelettes de Haar et un SVM avec des LRF

3.4.2 L'espace mathématique des matrices de covariance

Dans leurs travaux, Tuzel et al. [70] ont proposé de prendre en compte la structure de l'espace mathématique des matrices de covariance. En représentant chaque matrice de covariance par la forme vectorisée de sa partie triangulaire supérieure (voir section 3.1.1), on considère que les matrices de covariance appartiennent à un espace euclidien. Hors, l'ensemble des matrices de covariance forme une variété Riemannienne. Pour prendre cette particularité en compte, Tuzel et al. [70] s'appuient sur la propriété suivante : une variété est un espace topologique qui se comporte localement comme un espace euclidien. Ils proposent donc d'appliquer à chaque matrice de covariance une fonction qui permet de passer de la variété à un espace euclidien. Pour étudier l'impact de cette fonction de passage dans le cadre de la détection de visages, deux détecteurs de visages ont été entraînés dont l'un des deux utilise cette fonction de passage. Le

premier détecteur est celui décrit à la section 3.3.5 et le second possède les mêmes caractéristiques et utilise la fonction de passage. Les deux détecteurs sont ensuite appliqués sur la base CMU-MIT et les courbes FROC sont calculées. Le scan de chaque image de test s'est effectué avec les paramètres suivants : $fe = 1,2$ et $\Delta_p = 1$. Les deux courbes FROC sont présentées sur la figure 3.8. On note tout d'abord une augmentation du taux de vrais positifs d'environ 15%, qui se réduit à environ 5% à partir d'une centaine de faux positifs.

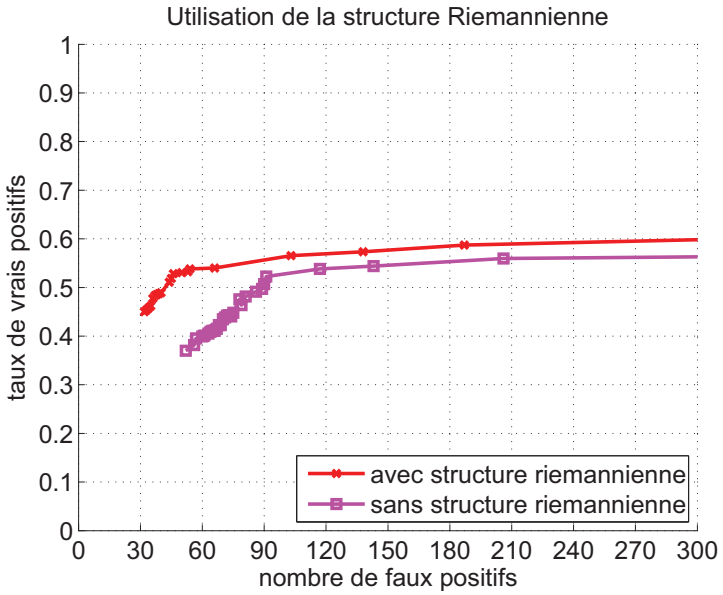


FIGURE 3.8 – Influence de la prise en compte de la structure de l'espace des matrices de covariance. La courbe « avec structure riemanienne » représente le détecteur qui prend en compte le fait que les matrices de covariance appartiennent à une variété riemannienne. Le détecteur « sans structure riemannienne » fait l'hypothèse que les matrices de covariance appartiennent à un espace euclidien.

Le temps d'exécution des deux détecteurs a également été comparé. Pour cela, cinq images de la base CMU-MIT ont été utilisées :

- aerosmith-double de taille 392×272 ;
- baseball de taille 302×204 ;
- cnn1085 de taille 320×240 ;
- cnn1160 de taille 320×240 ;
- cnn1630 de taille 320×240.

Les deux détecteurs ont été appliqués dix fois sur chaque image et les temps d'exécution ont ensuite été moyennés. Les résultats sont présentés dans le tableau 3.1. Le détecteur avec variété Riemanienne est toujours plus lent que le détecteur sans variété Riemanienne. Sur les images choisies, on constate une augmentation maximale du temps d'exécution de 35%. L'impact sur le temps d'exécution peut donc être important alors que le gain sur le taux de détection peut

être minime comme nous l'avons vu précédemment. Dans la suite, la structure de variété Riemanienne n'a donc pas été incorporée dans le détecteur.

	aerosmith-double	baseball	cnn1085	cnn1160	cnn1630
sans variété (s.)	9,74	5,62	5,84	5,72	5,92
avec variété (s.)	13,07	6,77	6,4	5,73	6,04
différence (%)	34,2	20,4	9,7	0,2	2,1

TABLE 3.1 – Temps d'exécution entre le détecteur avec variété riemannienne et le détecteur sans variété riemannienne

3.4.3 Analyse de sensibilité

Il est généralement intéressant de connaître le comportement d'un détecteur de visages face à divers facteurs influençant la qualité des images traitées comme le contraste, le flou ou encore le bruit. Par exemple, si on souhaite appliquer un détecteur de visages sur des images de vidéo surveillance, il est bon de savoir si le détecteur est robuste au flou. D'autres facteurs liés à la pose du visage sont également intéressants. Les travaux de Garcia et Delakis [17] ont inspiré la réalisation de l'analyse de sensibilité présentée. 20 images (voir figure 3.9) provenant de la base LFW-a, non utilisées pendant l'apprentissage, ont été sélectionnées. Pour tous les facteurs étudiés, le processus de scan utilise les paramètres suivants : $fe = 1,2$ et $\Delta_p = 1$. De plus, pour les facteurs « rotation », « contraste », « flou » et « bruit », les images sont redimensionnées à 63×63 pour faire en sorte que le visage présent dans chaque image ait une taille d'environ 24×24 qui représente la taille des images d'apprentissage.

FIGURE 3.9 – Images utilisées pour l'analyse de sensibilité

Rotation

Pour déterminer la robustesse du détecteur aux rotations, des rotations de $-30°$ à $+30°$ ont été appliquées à chaque image. Des exemples d'images générées sont présentés sur la figure 3.10.

FIGURE 3.10 – Exemples d'images utilisées pour la sensibilité aux rotations dans le plan

L'évolution du taux de vrais positifs en fonction de l'angle est présentée sur la figure 3.16(a). Le taux de détection reste très bon (supérieur à 85%) entre $-15°$ et $+15°$. Pour les angles supérieurs à $20°$ ou inférieurs à $-20°$, le taux de détection chute sous les 50%.

Contraste

Pour étudier l'influence du contraste, des images I_c sont générées à l'aide de l'équation suivante :

$$I_c = \alpha \bar{I} + (1 - \alpha) I \tag{3.14}$$

où I représente l'image originale, $\bar{I}$ représente l'image de la moyenne des niveaux de gris de I et α est un paramètre variant de $-2,0$ à $1,0$ qui permet de modifier le contraste de I : on obtient une augmentation de contraste pour $\alpha < 0$ et une diminution du contraste pour $\alpha > 0$. Des exemples d'image I_c sont présentées sur la figure 3.11.

FIGURE 3.11 – Exemples d'images utilisées pour la sensibilité au contraste

La figure 3.16(b) présente l'évolution du taux de détection en fonction du paramètre α. On note qu'une augmentation du contraste n'est pas bénéfique pour le détecteur. Il est par contre plus intéressant de noter que des images peu contrastées ne pénalisent pas les performances du détecteur. En effet, le taux de détection reste à 100% jusqu'à $\alpha = 0,7$.

Flou

Pour déterminer l'impact du flou, des images ont été générées par application successive d'une convolution avec un noyau gaussien de taille 3x3 et d'écart-type $1,0$. Par exemple, une image générée après 2 itérations signifie que l'image originale a été convoluée une fois, puis que

l'image convoluée a été convoluée une seconde fois. Des exemples d'images sont présentées sur la figure 3.12.

FIGURE 3.12 – Exemples d'images utilisées pour la sensibilité au flou

Les résultats sont présentés sur la figure 3.16(c). On trouve en abscisse le nombre de convolutions effectuées pour générer les images de test. Le détecteur présente une bonne robustesse au flou, on remarque que le taux de détection reste supérieur à 70% jusqu'à 6 convolutions.

Bruit

L'effet du bruit est étudié par application d'un bruit blanc gaussien d'écart-type σ compris entre 0,0005 et 0,005. La figure 3.13 présente des exemples d'images utilisées.

FIGURE 3.13 – Exemples d'images utilisées pour la sensibilité au bruit

Les résultats présentés sur la figure 3.16(d) montre que le taux de détection reste important (supérieur à 80%) pour $\sigma \leq 0.003$. Pour des valeurs de σ plus importantes, le taux chute progressivement jusqu'à 60%.

Échelle

Pour tester l'influence de la taille des visages sur les performances, les images utilisées ont été redimensionnées plusieurs fois pour obtenir des tailles de visages différentes : 24, 29, 35, 41, 50, 60, 72, 86, 103 et 124. Un facteur d'échelle de $1,2$ sépare deux tailles successives. La figure 3.14 présente des exemples de ces images.

L'influence de la taille des visages sur le taux de détection est présenté sur la figure 3.16(e). Jusqu'à une taille de 103×103, i.e. environ quatre fois plus grande que la taille d'apprentissage, le taux de détection reste supérieur à 90%. Pour la taille maximale 124×124, le taux de détection chute à 75%. Le détecteur présente donc des performances élevées tant que la taille des visages reste inférieure à une centaine de pixels.

FIGURE 3.14 – Exemples d'images utilisées pour la sensibilité à l'échelle

Position de la fenêtre de test

L'impact de la position de la fenêtre de test sur les performances a également été quantifié. Pour cela, différentes zones de taille 24×24 dans chaque image ont été découpées. À chaque zone est associée un vecteur de translation $\mathbf{v}_{xy} = (v_x, v_y)^{\mathrm{T}}$. La zone avec le vecteur $(0, 0)^{\mathrm{T}}$ est centrée sur le visage. La zone avec le vecteur $(-1, 0)^{\mathrm{T}}$ est décalée d'un pixel vers la gauche par rapport à la zone centrée. Les zones découpées sont telles que $-4 \leq v_x \leq 4$ et $-4 \leq v_y \leq 4$ (voir figure 3.15). Ensuite, pour chaque vecteur $(v_x, v_y)^{\mathrm{T}}$, les zones associées sont envoyées au classifieur et le taux de réponse positive est noté. Les résultats sont présentés sur la figure 3.16(f) où une couleur rouge indique un taux de 100% alors qu'une couleur bleu indique un taux de 0%. Cette carte de réponse indique que, pour une taille de visage de 24×24, la tolérance est de ± 2 pixels.

FIGURE 3.15 – Exemples d'images utilisées pour la sensibilité à la position de la fenêtre de test

Visages tournés

Le comportement du détecteur face à des visages tournés a aussi été étudié. Pour cela, le détecteur a été appliqué sur trois séries d'images provenant de la base FERET. La première série comporte des visages tournés de $\theta_y = 22,5°$ par rapport à l'axe y. La deuxième série comporte des visages tournés de $\theta_y = 45°$ et la troisième série comporte des visages tournés de $\theta_y = 67,5°$. Initialement, les visages présents dans les images de la base FERET ont une taille très grande (supérieure à 250×250). Pour limiter le problème de la taille des visages, chaque image est redimensionnée à 10% de sa taille initiale et le facteur d'échelle fe est fixé à $1,1$. Sur les visages tournés de $22,5°$, le détecteur détecte 70% des visages sans faire de faux

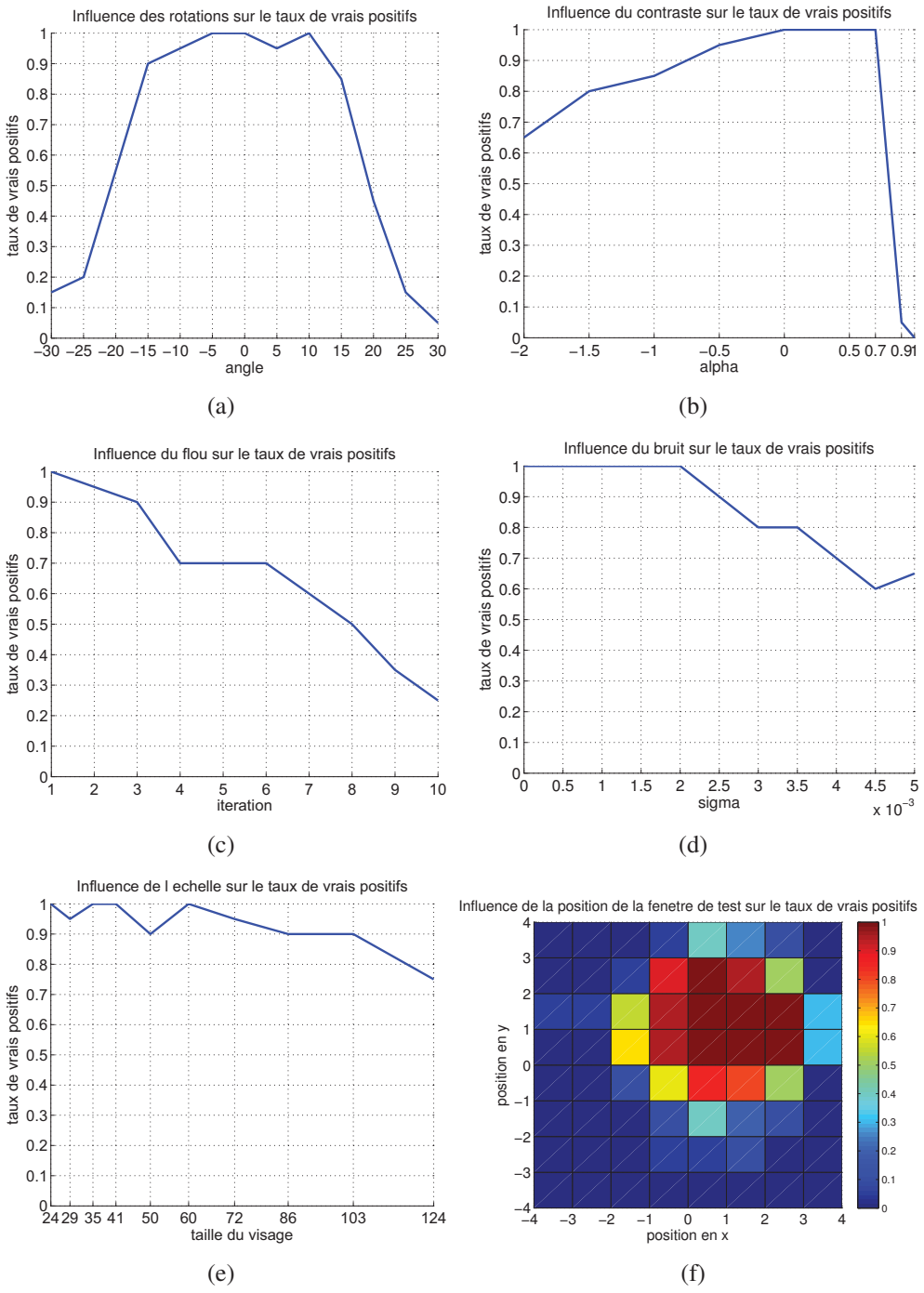


FIGURE 3.16 – Résultats de l'analyse de sensibilité. Différents facteurs sont étudiés : les rotations en (a), le contraste en (b), le flou en (c), le bruit en (d), la taille des visages en (e) et la position de la fenêtre de test en (f)

positifs. Pour les visages tournés de $45°$, seuls 14% des visages sont détectés sans aucun faux positifs. Enfin, pour les visages tournés de $67,5°$, seuls 15% des visages sont détectés et 1 faux positif est obtenu. Les courbes FROC du détecteur sur ces trois séries d'images sont également présentées sur la figure 3.17. On note que les visages tournés mettent rapidement en défaut le détecteur. Cela confirme que la détection de visages tournés necessite des solutions spécifiques. Celles-ci sont détaillées au chapitre 5.

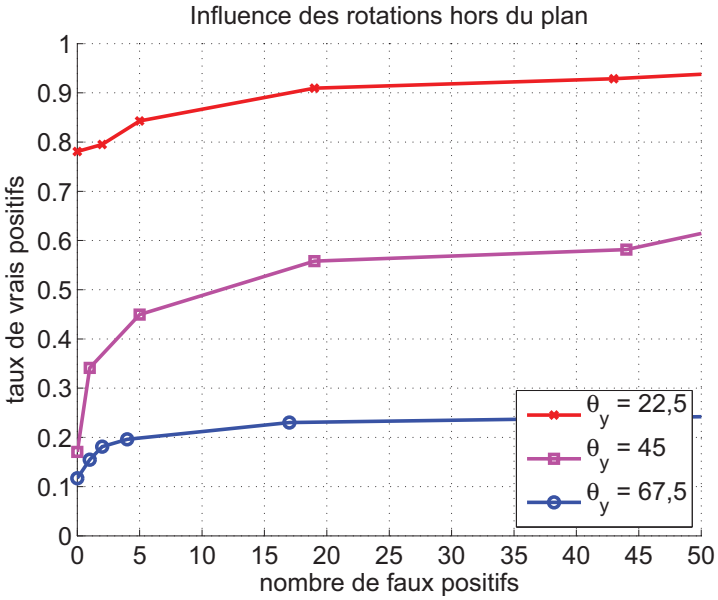
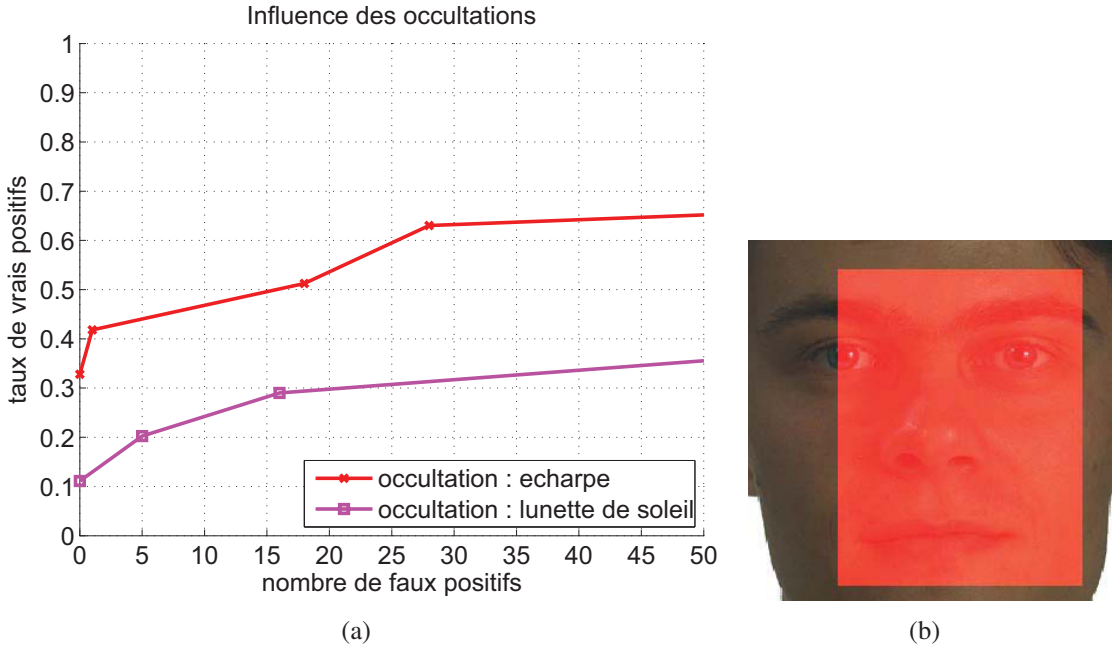


FIGURE 3.17 – Courbes FROC du détecteur de visages de face sur des visages tournés. Chaque visage est tourné d'un angle θ_y autour de l'axe y. Le détecteur a été appliqué sur trois séries d'images de visages tournés. Dans la première, les visages sont tournés de $22,5°$; dans la seconde, les visages sont tournés de $45°$ et dans la troisième, les visages sont tournés de $67,5°$

Occultations

Enfin, l'impact des occultations a été évalué en utilisant des images de la base AR. Deux ensembles d'images ont été considérés : le premier contient les images des visages occultés par une écharpe et le second contient les images des visages occultés par des lunettes de soleil. Comme pour la base FERET, les visages sont de très grande taille (environ 250×250). Les images ont donc été redimensionnées à 10% de leur taille initiale et le facteur d'échelle a été fixé à $1,1$. Dans les deux cas, le détecteur détecte difficilement les visages : 32% des visages portant une écharpe sont détectés et 4% des visages portant des lunettes de soleil sont détectés. Dans les deux cas, aucun faux positif n'est obtenu. Les courbes FROC du détecteur sur ces images sont présentées sur la figure 3.18(a). Comme pour les visages tournés, la détection de

visages occultés nécessite des solutions spécifiques (exposées au chapitre 5). Ces performances extrèmement faibles s'expliquent par la structure du détecteur utilisé. En effet, le premier niveau de la cascade n'est constitué que d'un seul classifieur faible qui utilise l'ensemble du visage pour prendre une décision (voir figure 3.18(b)). Ainsi, un visage occulté sera très probablement rejeté dès le premier niveau de cette cascade.



(a) (b)

FIGURE 3.18 – (a) - Courbes FROC du détecteur de visages de face sur des visages occultés. Le détecteur de visage de face est appliqué sur deux séries d'images : dans la première, les visages sont occultés par une écharpe et dans la deuxième, les visages sont occultés par des lunettes de soleil. (b) - Les résultats médiocres s'expliquent par le fait que le premier niveau de la cascade est constitué d'un seul classifieur faible qui utilise l'ensemble du visage pour prendre une décision. Ainsi, une majorité des visages occultés est rejetée dès le premier niveau

3.4.4 Étude des paramètres de scan d'une image

Dans cette dernière partie, l'impact des paramètres de scan d'une image, à savoir le facteur d'échelle fe et le déplacement entre deux fenêtres successives Δ_p, sont évalués. Pour l'étude des deux paramètres, l'ensemble de test A de la base CMU-MIT est utilisé.

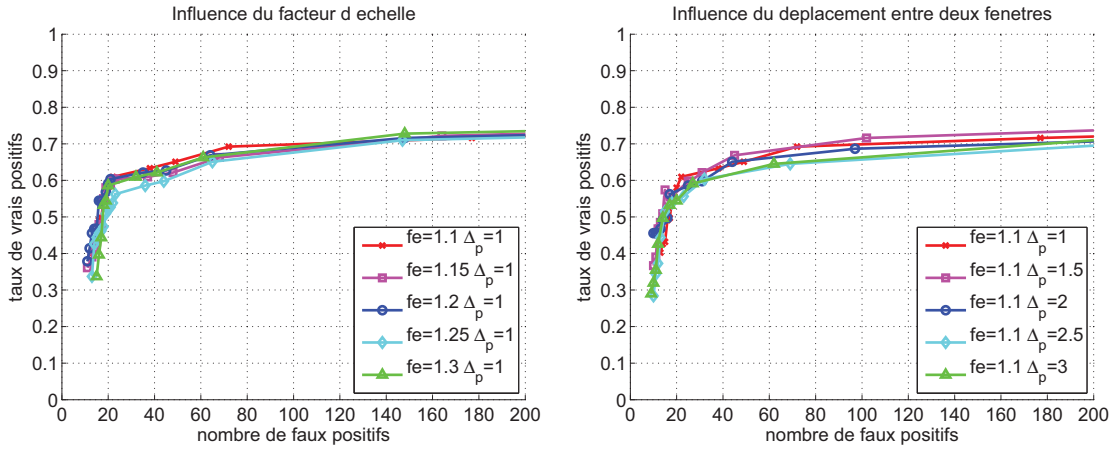
Le facteur d'échelle

Pour étudier l'influence du facteur d'échelle, Δ_p est fixé à 1 et les valeurs suivantes pour fe sont testées : 1,1, 1,15, 1,2, 1,25 et 1,3. Les différentes courbes FROC sont présentées sur la

figure 3.19(a). La valeur $fe = 1,25$ obtient les moins bonnes performances. Les autres valeurs de fe présentent des performances similaires.

Le déplacement entre deux fenêtres successives

Dans le cas du déplacement entre deux fenêtres successives, fe est fixé à $1,1$ et Δ_p varie parmi les valeurs suivantes : 1, 1,5, 2, 2,5 et 3. La figure 3.19(b) présente les courbes FROC pour les différentes valeurs de Δ_p. Pour $\Delta_p \geq 2,5$, les résultats se dégradent. Les meilleures performances sont obtenues pour $\Delta_p = 1,5$. Enfin, la valeur $\Delta_p = 2$ présente des résultats proches de $\Delta_p = 1$ tout en offrant un gain sur le temps d'exécution.



(a) (b)

FIGURE 3.19 – Influence des paramètres du scan d'une image. (a) - différentes valeurs pour fe sont testées et $\Delta_p = 1$ (b) - différentes valeurs pour Δ_p sont testées et $fe = 1,1$

3.5 Bilan

L'utilisation des matrices de covariance a été choisie pour leurs performances supérieures aux ondelettes de Haar. Les différentes séries d'expérimentations ont montré que les matrices de covariance présentent une bonne robustesse vis à vis des rotations dans le plan, des images peu contrastées, du flou et du bruit. Par contre, la taille des visages à détecter est à prendre en compte car des tailles importantes seront plus difficiles à détecter. Enfin, les rotations hors du plan et les occultations dégradent fortement les performances du détecteur.

Chapitre 4

Classification avec données manquantes

Ce chapitre traite de classification avec données manquantes, qui est un problème rencontré lors de l'adaptation d'un détecteur de visages de face à des situations non prévues par l'apprentissage. Les adaptations envisagées sont tout d'abord présentées ainsi que les problèmes associés. Plusieurs solutions visant à permettre la classification avec données manquantes sont ensuite détaillées. Enfin, diverses séries d'expérimentation sont exposées dans le but d'évaluer et de valider les différents aspects des solutions proposées.

4.1 Motivation

Une des problématiques de cette thèse concerne l'adaptation de détecteurs de visages de face à des conditions d'utilisation non couvertes par l'apprentissage initial. Ces conditions d'utilisation non couvertes peuvent être listées en trois catégories :

1. détection de visages de face avec rotation dans le plan (voir figure 4.1(a)) ;
2. détection de visages avec rotation hors du plan (voir figure 4.1(b)) ;
3. détection de visages de face partiellement occultés (voir figure 4.1(c)).

Les applications envisagées concernent les deux derniers points : la détection de visages avec rotation hors du plan et la détection de visages occultés. Dans la suite, le terme « visage tourné » est utilisé pour faire référence à un visage avec une rotation hors du plan. Dans chaque cas, l'objectif est de modifier un détecteur de visage de face existant sans faire d'apprentissage supplémentaire. Les solutions proposées pour chaque cas sont les suivantes :

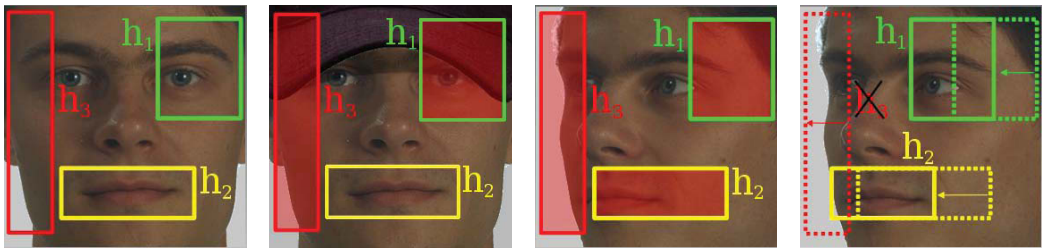
détection de visages tournés : un visage tourné n'est, en général, pas détecté par un détecteur de visage de face du fait de la modification de la pose du visage et donc de son apparence (voir figure 4.2(c)). Pour prendre en compte la modification de la pose, un modèle géométrique 3D est utilisé pour ajuster la position des sous-fenêtres associées à chaque classifieur faible. Comme on peut le voir sur la figure 4.2(d), la modification de la position

(a) (b) (c)

FIGURE 4.1 – Conditions mettant en défaut un détecteur de visages de face : (a) Plusieurs visages présentant diverses rotations dans le plan, (b) Visage avec une rotation hors du plan, (c) Visage occulté par une écharpe

des sous-fenêtres peut entraîner la disparition de certaines sous-fenêtres et les classifieurs faibles associés deviennent donc indisponibles ;

détection de visages de face occultés : ces visages ne sont, en général, pas détectés car certaines sous-fenêtres sont occultées et celles-ci seront classées en non-visage (voir figure 4.2(b)). La solution envisagée ici consiste à utiliser uniquement les sous-fenêtres non occultées lors de la classification. Les sous-fenêtres occultées ne sont plus utilisées et les classifieurs faibles associés sont considérés comme indisponibles.



(a) (b) (c) (d)

FIGURE 4.2 – Problèmes et solutions pour adapter un détecteur de visages de face. En (a), un exemple de classifieurs faibles appris. Chacun est en charge de classer une sous-fenêtre. En (b), le visage est occulté par une casquette et les sous-fenêtres de h_1 et h_3, en rouge, risquent d'être classées comme non-visage. La solution retenue consiste à utiliser uniquement h_2 qui n'est pas occulté. En (c), le visage est tourné de $45°$ et toutes les sous-fenêtres risquent d'être classées comme non-visage. La solution envisagée s'appuie sur un ajustement des positions des sous-fenêtres (voir figure (d)). Notez que la modification des positions des sous-fenêtres peut entraîner la disparition de certaines sous-fenêtres comme celle de h_3

Ces deux solutions sont détaillées au chapitre suivant. Dans les deux cas, les solutions envisagées reposent sur une cascade où seul un sous-ensemble des classifieurs faibles est utilisé. En effet, pour chaque niveau, seuls les classifieurs faibles disponibles sont utilisés. Chaque cascade modifiée représente donc une cascade avec classifieurs faibles manquants car on supprime les classifieurs faibles indisponibles de la cascade initiale (celle du détecteur de visage de face).

4.2 Les données manquantes en reconnaissance de forme

Il arrive fréquemment que des données soient manquantes dans des applications de modèles prédictifs. Par exemple, il manque souvent des résultats de test clinique quand on souhaite prédire l'efficacité d'un traitement pour un patient donné. Dans le cas de questionnaires concernant les préférences d'achat, les clients ne répondent parfois pas à toutes les questions. De façon générale en reconnaissance de forme, il est important de distinguer deux cas : les données peuvent manquer au moment de l'apprentissage et/ou au moment de la classification. Dans le cas de l'étude présentée ici, l'apprentissage est réalisé avec la totalité des données et certaines données (certains classifieurs faibles) manquent au moment de la classification. Nous nous concentrons donc sur les méthodes de traitement des données manquantes pendant la classification.

Données manquantes pendant la classification

Une étude des différentes méthodes de traitement des données manquantes lors de la classification par arbre de décision a été réalisée par Saar-Tsechansky et Provost [59]. Dans cette étude, on se place dans le cadre de la prédiction d'une variable cible y en appliquant un modèle F, issu d'un apprentissage, sur un vecteur d'attribut $\mathbf{x} = (x_1 \dots x_N)^{\mathrm{T}}$ ($y = F(\mathbf{x})$) et on suppose que certains attributs x_i sont manquants. Dans ce contexte, plusieurs approches existent pour traiter les données manquantes :

1 - Ignorer l'instance à traiter. Si l'instance présente des attributs manquants, on décide de ne pas faire de prédiction. Une prédiction est faite seulement si tous les attributs sont présents ;

2 - Acquérir les attributs manquants. En pratique, un attribut manquant peut être acquis moyennant un coût, tel que le coût d'effectuer un test clinique ;

3 - Estimer les attributs manquants par une valeur de substitution. L'attribut manquant est remplacé par une valeur estimée à partir des valeurs de cet attribut dans l'ensemble d'apprentissage. Une pratique courante consiste à prendre la moyenne des valeurs de l'attribut ;

4 - Estimer la variale cible en se basant sur les distributions des attributs. Connaissant la distribution (estimée) des valeurs d'un attribut, on peut estimer la distribution de la variable cible. L'algorithme C4.5 [55], basé sur un arbre de décision, s'appuie sur ce principe : quand un attribut est manquant au niveau d'un noeud, plusieurs pseudo-instances sont

créées avec des valeurs différentes pour l'attribut manquant ainsi qu'un poids correspondant à la probabilité de la valeur affectée. Chaque pseudo-instance est traitée par l'arbre de décision et la décision finale est une combinaison des différentes décisions ;

5 - Estimer les attributs manquants par une valeur arbitraire. Au lieu d'estimer une valeur (comme la moyenne), on remplace systématiquement l'attribut manquant par une valeur arbitraire.

6 - Utiliser des modèles prédictifs réduits. Dans cette approche, on construit plusieurs modèles prédictifs capables de prédire la variable cible en utilisant seulement un sous-ensembles des attributs. Par exemple, si l'attribut 1 manque, on utilisera le modèle appris sur les attributs 2 à N. L'idée de cette approche est d'utiliser uniquement les attributs présents pour prédire la variable cible.

Parmi toutes ces approches, les approches 4 et 5 ont été retenues. L'approche 6 représente les solutions existantes dans le domaine de la détection de visages tournés ou occultés. Elles sont abordées au chapitre suivant. La section suivante expose une solution naïve au problème de classifieurs faibes manquants inspirée de l'approche 5. Quant aux solutions solutions proposées, elles s'appuient sur les distributions des scores fournis par les classifieurs faibles pour estimer la variable cible, qui est ici la probabilité d'appartenance d'un exemple à la classe visage. L'étude de Saar-Tsechansky et Provost [59] ne concerne pas le traitement des données manquantes dans une cascade de classifieurs boostés. Quelques solutions ont été proposés dans le cas d'autres classifieurs. Pour des algorithmes de type SVM, on peut citer Globerson et Roweis [18] qui proposent un algorithme d'apprentissage sous la forme d'un problème quadratique qui prend en compte la possibilité que des attributs soient manquants lors de la classification. Cette idée est par la suite améliorée par Dekel et Shamir [11] qui proposent un algorithme d'apprentissage robuste à la suppression de attributs pendant la classification mais également robuste à la corruption (par du bruit) de ces attributs. Dans le cadre du boosting, Wang et Feng [75] proposent d'estimer les attributs manquants en utilisant des régressions linéaires entre chaque attribut manquant et l'ensemble des attributs présents. Enfin, Smeraldi et al. [66] proposent une version modifiée de l'agorithme Adaboost discret dans laquelle les classifieurs faibles donnent des réponses dans $\{-1, 0, 1\}$. Le cas 0 correspond au cas où l'attribut est manquant pendant la classification.

Aucun article sur le traitement de classifieurs faibles manquants dans une cascade de classifieurs boostées n'a été trouvé et la solution proposée est, à notre connaissance, la première existante. Celle-ci est uniquement valable pour des cascades utilisant des algorithmes de boosting réels, i.e. des algorithmes de boosting dont les classifieurs faibles sont des fonctions à valeurs réelles. On ne peut donc pas l'appliquer à l'algorithme Adaboost discret (les classifieurs faibles donnent des réponses binaires : 1 ou -1). Dans notre cas, les attributs seront les scores faibles fournis par les classifieurs faibles et le modèle prédictif sera le classifieur fort de chaque niveau qui combine les scores faibles et les compare à un seuil.

4.3 Solution naïve de gestion des classifieurs faibles manquants

Supposons que l'on souhaite classer un exemple $\mathbf{x}$ avec un classifieur fort $\text{sign}(H - \tau)$ où H est composé d'un ensemble de classifieurs faibles $\{h_1, \ldots, h_T\}$. Chaque classifieur faible h_t est une fonction à valeur réelle où $\text{sign}(h_t(\mathbf{x}))$ représente le label prédit par h_t pour $\mathbf{x}$. De plus, $|h_t(\mathbf{x})|$ représente la confiance du classifieur faible dans sa prédiction. Une valeur $h_t(\mathbf{x})$ nulle peut donc s'interpréter comme une prédiction neutre du classifieur faible ($\mathbf{x}$ est alors situé sur la frontière de décision associée à h_t). Supposons maintenant que seulement $p < T$ classifieurs faibles soient disponibles, donné par $\{h_{\mathrm{d}_1}, \ldots, h_{\mathrm{d}_p}\}$. L'ensemble des classifieurs indisponibles (et donc manquants) est défini par $\{h_{\mathrm{i}_1}, \ldots, h_{\mathrm{i}_q}\}$ où $q = T - p$. La stratégie la plus simple pour classer $\mathbf{x}$ malgré les classifieurs manquants consiste à considérer que tous les classifieurs faibles manquants donnent une réponse nulle, i.e. $h_{\mathrm{i}_1}(\mathbf{x}) = \cdots = h_{\mathrm{i}_q}(\mathbf{x}) = 0$. Si on note $H_{\mathrm{d}}(\mathbf{x}) = \sum_{t=1}^{p} h_{\mathrm{d}_t}(\mathbf{x})$, alors le classifieur fort devient $\text{sign}(H_{\mathrm{d}} - \tau)$. En appliquant ce principe à tous les classifieurs forts d'une cascade $\{\text{sign}(H_1(\mathbf{x}) - \tau_1), \ldots, \text{sign}(H_K(\mathbf{x}) - \tau_K)\}$, on obtient une nouvelle cascade composée des classifieurs forts $\{\text{sign}(H_{1\mathrm{d}} - \tau_1), \ldots, \text{sign}(H_{K\mathrm{d}} - \tau_K)\}$. Cette stratégie naïve consiste donc à considérer que les classifieurs faibles manquants n'interviennent pas dans la décision finale. Elle représentera la méthode de base dans les expérimentations. Nous chercherons donc à faire toujours mieux que cette stratégie, en se rapprochant au maximum des performances du classifieurs sans classifieurs faibles manquants.

4.4 Formulation probabiliste d'une cascade boostée

Dans les algorithmes de boosting réels, le label prédit $y \in \{-1, 1\}$ d'un exemple $\mathbf{x}$ peut être vu comme une variable aléatoire qui suit une loi de Bernoulli et $H(\mathbf{x})$ peut être interprétée comme la probabilité de y d'être un visage sachant l'exemple $\mathbf{x}$ (également appelée probabilité a posteriori) en utilisant la fonction sigmoïde suivante [14] :

$$P(y = 1|\mathbf{x}) = \frac{1}{1 + \mathrm{e}^{-H(\mathbf{x})}} \tag{4.1}$$

Chaque niveau d'une cascade calcule donc $P(y_j = 1|\mathbf{x}, y_1 = 1, \ldots, y_{j-1} = 1)$ où y_j est le label prédit par le niveau j. Le terme $y_1 = 1, \ldots, y_{j-1} = 1$ présent dans la probabilité signifie que si un exemple $\mathbf{x}$ atteint le niveau j, alors il a passé tous les niveaux précédents (les niveaux 1 à $j - 1$).

La présence de classifieurs faibles manquants dans les différents niveaux introduit de l'incertitude sur chaque label prédit y_j. Cette incertitude n'est pas prise en compte dans la probabilité $P(y_j = 1|\mathbf{x}, y_1 = 1, \ldots, y_{j-1} = 1)$ car les labels $y_1, \ldots, y_{j-1}$ sont supposés positifs. C'est pourquoi nous proposons de calculer $P(y_1 = 1, \ldots, y_j = 1|\mathbf{x})$ au niveau j de la cascade. Ainsi, le label prédit au niveau j dépendra aussi des labels prédits par les niveaux 1 à $j - 1$. Dans la suite de ce manuscript, l'évènement $y_1 = 1, \ldots, y_j = 1$ sera noté $y_{1:j} = 1$. En appliquant la

règle de Bayes [1] sur $P(y_{1:j} = 1|\mathbf{x})$, on obtient :

$$P(y_{1:j} = 1|\mathbf{x}) = P(y_j = 1|\mathbf{x}, y_{1:j-1} = 1) \times P(y_{1:j-1} = 1|\mathbf{x}) \quad \forall j > 1 \tag{4.2}$$

L'application récursive de cette règle donne :

$$P(y_{1:j} = 1|\mathbf{x}) = \prod_{i=2}^{j} P(y_i = 1|\mathbf{x}, y_{1:i-1} = 1) \times P(y_1 = 1|\mathbf{x}) \quad \forall j > 1 \tag{4.3}$$

Cette formulation probabiliste est très proche de celle proposée par Lefakis et Fleuret [32]. Les motivations restent néanmoins différentes car ils proposent un nouvel algorithme d'apprentissage basé sur une formulation probabiliste d'une cascade. Dans le cas de cette étude, une formulation probabiliste est utilisée pour gérer le fait que des classifieurs faibles peuvent manquer lors de la classification.

Dans la formulation classique d'une cascade, chaque niveau j applique un classifieur fort H_j sur $\mathbf{x}$ et compare $H(\mathbf{x})$ à un seuil τ_j. Avec la formulation probabiliste, les seuils τ_j disparaissent et de nouveaux seuils β_j sont introduits. En effet, $P(y_j = 1|\mathbf{x}, y_{1:j-1} = 1) \leq 1$ et donc :

$$\begin{aligned} P(y_{1:j} = 1|\mathbf{x}) &= \prod_{i=2}^{j} P(y_i = 1|\mathbf{x}, y_{1:i-1} = 1) \times P(y_1 = 1|\mathbf{x}) \\ &\leq \prod_{i=2}^{j-1} P(y_i = 1|\mathbf{x}, y_{1:i-1} = 1) \times P(y_1 = 1|\mathbf{x}) \\ &\leq \cdots \\ &\leq P(y_1 = 1|\mathbf{x}) \end{aligned} \tag{4.4}$$

L'équation (4.4) montre que si $P(y_{1:j} = 1|\mathbf{x})$ est inférieur à une valeur β_j, le processus de classification devrait s'arrêter car les valeurs $P(y_{1:j+1} = 1|\mathbf{x}), \ldots, P(y_{1:K} = 1|\mathbf{x})$ seront également inférieures à β_j. Dans la formulation probabiliste retenue, un classifieur est donc défini par $\text{sign}(P(y_{1:j} = 1|\mathbf{x}) - \beta_j)$. La cascade modifiée devient alors $\{\text{sign}(P(y_1 = 1|\mathbf{x}) - \beta_1), \text{sign}(P(y_{1:2} = 1|\mathbf{x}) - \beta_2), \ldots, \text{sign}(P(y_{1:K} = 1|\mathbf{x}) - \beta_K)\}$. Dans la suite, le terme de *McCascade* [2] est utilisé pour faire référence à une cascade qui aura été modifiée selon la formulation proposée. La figure 4.3 récapitule les différences entre une structure en cascade et une structure en McCascade. La section 4.6 détaille le calcul des seuils β_j et la section suivante se focalise sur l'estimation des probabilités $P(y_j = 1|\mathbf{x}, y_{1:j-1} = 1)$ pour chaque niveau de cascade.

1. Si A, B et C sont trois évènements, alors $P(A, B|C) = P(B|C, A)P(A|C)$
2. fait référence en anglais aux termes « **cascade** with **m**issing **c**lassifiers »

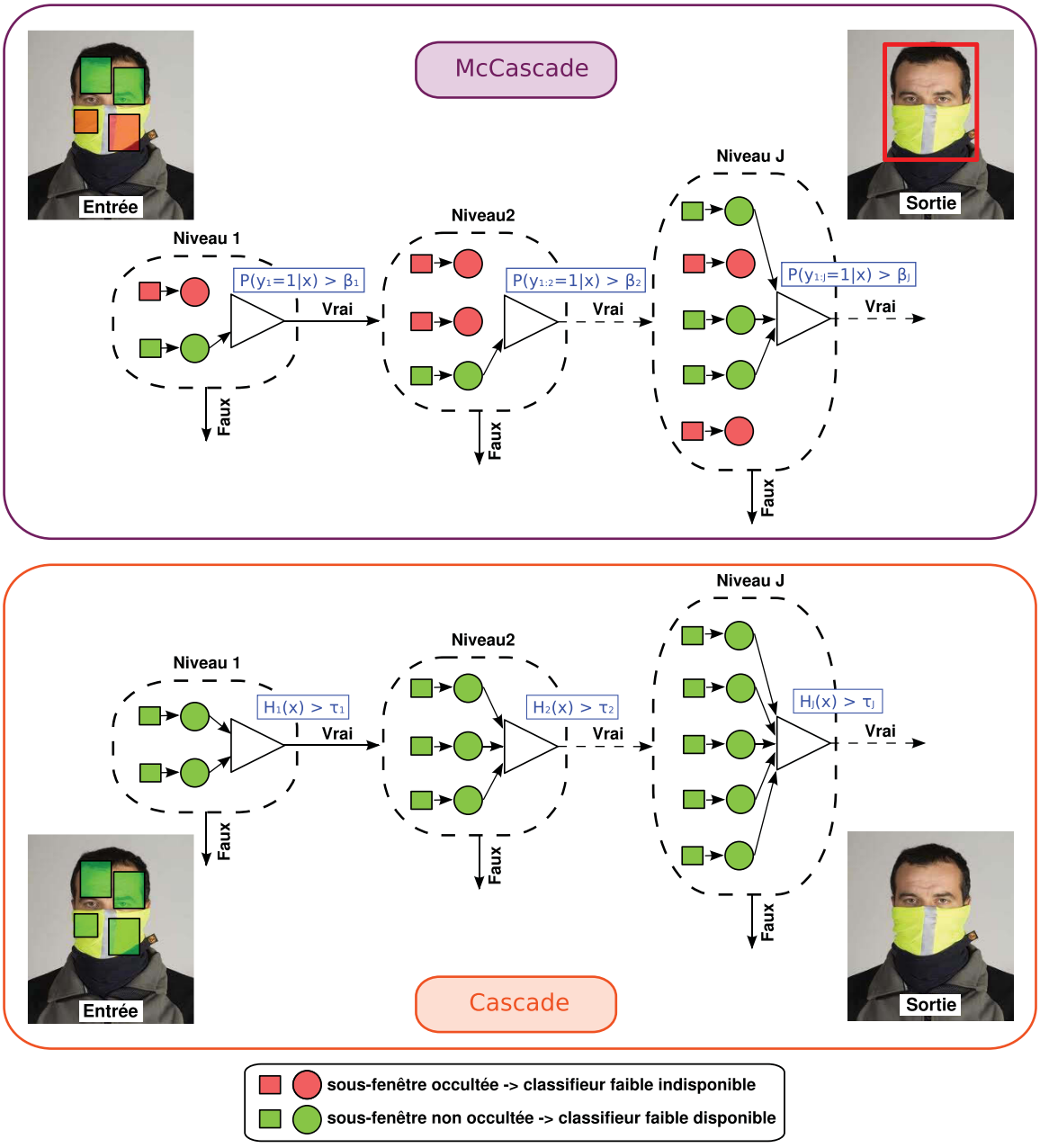


FIGURE 4.3 – Différences entre une cascade et une McCascade pour la classification d'un exemple $\mathbf{x}$. Dans une cascade, l'ensemble des classifieurs faibles sont utilisés. Au niveau j, $H_j(\mathbf{x})$ est calculé et est comparé au seuil τ_j. Un visage occulté n'est généralement pas détecté car les sous-fenêtres occultées pénalisent la décision à chaque niveau. Dans une McCascade, seules certains classifieurs faibles sont utilisés. Sur la figure, seuls les classifieurs faibles en charge de classifier le haut du visage sont utilisés. Au niveau j, $P(y_{1:j} = 1|\mathbf{x})$ est calculée et est comparée au seuil β_j. Un visage occulté est ici généralement détecté. Contrairement à la décision prise au niveau j d'une cascade, la décision prise au niveau j d'une McCascade fait intervenir l'ensemble des niveaux précédents

4.5 Estimation de la probabilité a posteriori

Nous venons de voir que la probabilité calculée à chaque niveau d'une cascade probabiliste faisait intervenir les probabilités a posteriori calculées dans une cascade traditionnelle. Cette section présente plusieurs stratégies d'estimation de la probabilité a posteriori $P(y_j = 1|\mathbf{x}, y_{1:j-1} = 1)$ pour un niveau j donné. En vue de faire abstraction du niveau de la cascade, la probabilité $P(y_j = 1|\mathbf{x}, y_{1:j-1} = 1)$ sera simplement notée $P(y = 1|\mathbf{x})$. Cette simplification est possible car les stratégies présentées sont indépendantes du niveau de la cascade. Quand des classifieurs faibles manquent, la probabilité $P(y = 1|\mathbf{x})$ ne peut plus être calculée et une approximation doit être utilisée. Trois stratégies d'approximation différentes sont proposées :

– La stratégie la plus simple pour estimer $P(y = 1|\mathbf{x})$ consiste à calculer une probabilité uniquement sur les classifieurs faibles disponibles. $P_{\text{boost}}(y = 1|\mathbf{x})$ est donc défini comme :

$$P_{\text{boost}}(y = 1|\mathbf{x}) \doteq \frac{1}{1 + \mathrm{e}^{-H_{\mathrm{d}}(\mathbf{x})}} \tag{4.5}$$

– Une seconde stratégie, notée $P_{\text{knn}}(y = 1|\mathbf{x})$, s'appuie sur un ensemble d'apprentissage (qui peut être celui utilisé pour l'apprentissage de la cascade initiale). Chaque exemple d'apprentissage $\mathbf{x}_i$ fournit un ensemble de scores de classifieurs faibles $h_{\mathbf{x}_i}$ ainsi qu'un label associé y_i avec $h_{\mathbf{x}_i} = (h_1(\mathbf{x}_i), \ldots, h_T(\mathbf{x}_i))$. Toutes ces valeurs de classifieurs faibles forment un ensemble $\mathcal{H} = \{(h_{\mathbf{x}_i}, y_i)\}_{i=1,\ldots,N}$ et le sous-ensemble des valeurs des classifieurs faibles disponibles forme $\mathcal{H}_{\mathrm{d}} = \{(h_{\mathrm{d}_{\mathbf{x}_i}}, y_i)\}_{i=1,\ldots,N}$ où $h_{\mathrm{d}_{\mathbf{x}_i}} = (h_{\mathrm{d}_1}(\mathbf{x}_i), \ldots, h_{\mathrm{d}_p}(\mathbf{x}_i))$. Cet ensemble $\mathcal{H}_{\mathrm{d}}$ est utilisé comme un ensemble d'apprentissage pour estimer $P(y = 1|\mathbf{x})$ en utilisant l'agorithme des k-plus proches voisins, noté k-ppv par la suite. Sachant un exemple $\mathbf{x}$ et ces scores associés $h_{\mathrm{d}_{\mathbf{x}}}$ fournis par les classifieurs faibles disponibles, l'algorithme des k-ppv donne les labels $\{y_1^*, \ldots, y_k^*\}$ des k plus proches voisins dans l'ensemble $\mathcal{H}_{\mathrm{d}}$. On calcule alors $P_{\text{knn}}(y = 1|\mathbf{x})$ comme :

$$P_{\text{knn}}(y = 1|\mathbf{x}) \doteq \sum_{i=1}^{k} \frac{\mathbb{1}_{\{y_i^*=1\}}}{k} \tag{4.6}$$

où $\mathbb{1}_{\{pred\}}$ est une fonction qui vaut 1 si le prédicat $pred$ est vrai et 0 sinon. Le principe de cette stratégie est illustré sur la figure 4.4.

– Une dernière stratégie, notée $P_{\text{comb}}(y = 1|\mathbf{x})$, consiste à combiner les deux stratégies précédentes de la façon la plus simple :

$$P_{\text{comb}}(y = 1|\mathbf{x}) \doteq \frac{P_{\text{boost}}(y = 1|\mathbf{x}) + P_{\text{knn}}(y = 1|\mathbf{x})}{2} \tag{4.7}$$

4.6 Calcul des seuils d'une McCascade

La dernière étape dans la contruction d'une McCascade consiste à calculer les seuils β_j de chaque niveau. Généralement, les seuils d'une cascade sont fixés pendant l'apprentissage.

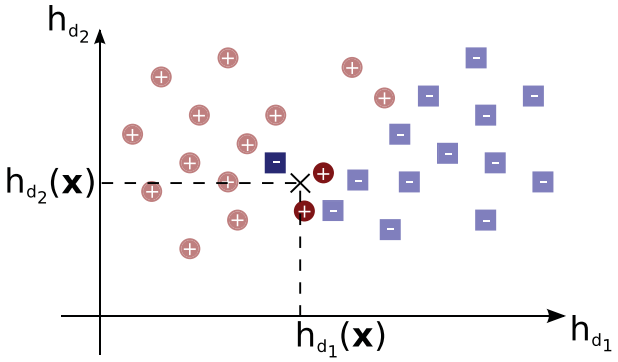


FIGURE 4.4 – Estimation de la probabilité a posteriori par les k-plus proches voisins. Supposons que deux classifieurs faibles soient disponibles h_{d_1} et h_{d_2}. Pour calculer la probabilité $P_{\mathrm{knn}}(y = 1|\mathbf{x})$, on calcule $h_{\mathrm{d}_1}(\mathbf{x})$ et $h_{\mathrm{d}_2}(\mathbf{x})$ puis on cherche les plus proches voisins du point $(h_{\mathrm{d}_1}(\mathbf{x}), h_{\mathrm{d}_2}(\mathbf{x}))$ dans l'ensemble $\{(h_{\mathrm{d}_1}(\mathbf{x}_i), h_{\mathrm{d}_2}(\mathbf{x}_i))\}_{i=1,\ldots,N}$ où x_i est un exemple d'apprentissage. Dans le cas de 3 voisins, on trouve 2 positifs et 1 négatifs. On a donc $P_{\mathrm{knn}}(y = 1|\mathbf{x}) = 2/3$

Quelques travaux relatifs au calcul des seuils d'une cascade après l'apprentissage ont été menés. La structure de cascade souple proposée par Bourdev et Brandt [2] impose une phase de calibration post apprentissage durant laquelle les seuils de rejet sont calculés. Cette phase de calibration est une procédure itérative dont le but de chaque itération est de sélectionnner le meilleur classifieur faible et de fixer le seuil de rejet de la cascade souple courante. Luo [40] propose deux méthodes de calcul de seuils post apprentissage pour des cascades de classifieurs boostés. La première méthode utilise la relaxation Lagrangienne et suppose que les niveaux sont indépendants (optimisation locale). La deuxième méthode suppose que les niveaux ne sont pas indépendants entre eux et aboutit à deux algorithmes gloutons qui visent à optimiser l'ensemble des seuils de façon globale. Enfin, Zhang et Viola [88] proposent également d'utiliser une procédure itérative pour fixer les seuils d'une cascade. Leur méthode s'appuie le fait qu'un détecteur de visages produit généralement plusieurs détections par visage et qu'en pratique, une seule de ces détections est réellement nécessaire. Leur méthode nécessite donc une base d'images contenant des visages ainsi que les vérités terrain associées.

4.6.1 Procédure itérative de calcul des seuils

Pour calculer les seuils $\beta_1, \ldots, \beta_K$, une procédure itérative, inspirée par celle proposée par Bourdev et Brandt [2], est utilisée. Celle-ci est décrite par l'algorithme 4. Cette procédure nécessite l'ensemble d'apprentisssage positif $\mathcal{S}^p$, l'ensemble d'image de fond $\mathcal{B}$ ainsi que l'ensemble des lois de probabilité $\{P(y_1 = 1|\mathbf{x}), \ldots, P(y_{1:K} = 1|\mathbf{x})\}$. À chaque itération j, toutes les probabilités $p_{ji} = P(y_{1:j} = 1|\mathbf{x}_i)$ sont calculées et β_j est choisi parmi l'ensemble fini des valeurs $\tilde{p}_{ji} = 0.5(p_{ji} + p_{j(i+1)})$, $i \in \{1, \ldots, N \quad 1\}$. La fonction `find_optimal_threshold` (voir ligne 14) retourne le seuil qui minimise une fonction de coût définie sur le taux de faux positifs

et de vrais positifs.

Algorithme 4 : Estimation des seuils β_j d'une McCascade

Entrées : – $\mathcal{S}^p$: ensemble d'images d'apprentissage positives ;
– $\mathcal{B}$: ensemble d'images de fond ne contenant pas de positifs ;
– $\{P(y_1 = 1|\mathbf{x}), \ldots, P(y_{1:K} = 1|\mathbf{x})\}$: ensemble de lois de probabilité ;

Sorties : $\beta_1, \ldots, \beta_K$: les seuils de la McCascade.

pour $j = 1$ **à** K **faire**
  si $j = 1$ **alors**
    Créer l'ensemble d'apprentissage des négatifs $\mathcal{S}^n$ en extrayant aléatoirement des zones dans les images de $\mathcal{B}$;
  sinon
    Appliquer la McCascade $\{\text{sign}(P(y_1 = 1|\mathbf{x}) - \beta_1), \ldots, \text{sign}(P(y_{1:j-1} = 1|\mathbf{x}) - \beta_{j-1})\}$ sur les images de $\mathcal{B}$ pour générer des faux positifs qui sont ajoutés à l'ensemble des négatifs $\mathcal{S}^n$;
  finsi
  $\mathrm{P} = \emptyset$;
  $\mathrm{Y} = \emptyset$;
  pour chaque *exemple* $(x_i, y_i) \in \mathcal{S}^p \cup \mathcal{S}^n$ **faire**
    Calculer la probabilité $p_{ji} = P(y_{1:j} = 1|\mathbf{x}_i)$;
    $\mathrm{P}[i] = p_{ji}$;
    $\mathrm{Y}[i] = y_i$;
  finprch
  $\beta_j =$ `find_optimal_threshold(`P, Y`)` ;
finpour

4.6.2 Les différentes fonctions de coût

Contrairement à la cascade initiale où chaque niveau assure d'atteindre un taux de vrais positifs supérieur à d_{min} avec un taux de faux positifs inférieur à f_{max}, la McCascade ne peut pas garantir les mêmes performances. Le but d'utiliser une fonction de coût est d'assurer que chaque seuil calculé amène des performances proches de celles de la cascade initiale. Trois fonctions de coût sont proposées :

– FP_cost est définie sur le taux de faux positifs f_β associé à un seuil β :

$$\text{FP\_cost}(f_\beta) = \max(0, f_\beta - f_{\max}) \tag{4.8}$$

Cette fonction signifie que le seuil obtenu amène un taux de faux positifs aussi proche que possible de $f_{\max}$.

– TP_cost est définie sur le taux de vrais positifs d_β associé à un seuil β :

$$\text{TP\_cost}(d_\beta) = \max(0, d_{\min} - d_\beta) \tag{4.9}$$

Un seuil calculé avec cette fonction amènera un taux de vrais positifs proche de $d_{\min}$.

– FP_TP_cost est définie à la fois sur le taux de vrais positifs et sur le taux de faux positifs :

$$\text{FP\_TP\_cost}(f_\beta, d_\beta) = \text{FP\_cost}(f_\beta) + \text{TP\_cost}(d_\beta) \tag{4.10}$$

Cette dernière fonction représente un compromis entre un taux de faux positifs de $f_{\max}$ et un taux de détection de $d_{\min}$.

Une version détaillée de `find_optimal_threshold` avec la fonction de coût FP_TP_cost est donnée dans l'algorithme 5.

Algorithme 5 : La fonction `find_optimal_threshold` est utilisée pour calculer les seuils de chaque niveau d'une McCascade

Entrées : – P : tableau de probabilités ;
– Y : tableau des lables $y_i \in \{-1, 1\}$ associés à chaque valeur P$[i]$.

Sorties : β^* : seuil de décision optimal entre les probabilités des positifs et des négatifs.

$c^* = +\infty$;
$\beta^* = 0$;
N = length(P);
pour $i = 1$ **à** $N - 1$ **faire**
  $\beta = 0.5 \times (\text{P}[i] + \text{P}[i+1])$;
  Calculer le taux de vrais positifs d et le taux de faux positifs f associés à β sur (P,Y);
  c = FP_TP_cost(f, d) // `ou c =` FP_cost(f, d) `ou c =` TP_cost(f, d) ;
  si $c < c^*$ **alors**
    $c^* = c$;
    $\beta^* = \beta$;
  finsi
finpour

4.7 Méthodologie expérimentale

4.7.1 Le détecteur utilisé

Pour tester les différents aspects de la méthode proposée, un détecteur de visages à trois niveaux est utilisé. Le nombre de classifieurs faibles de chaque niveau a été fixé à 5, 10 et 25. Le premier et second niveau sont appris sur des sous-ensembles de 2 caractéristiques tandis que le troisième niveau utilise des sous-ensembles de 3 caractéristiques. Chaque niveau est appris avec 4000 positifs et 8000 négatifs. De plus, 1600 images de fond sont utilisées pour générer des négatifs pendant les phases de bootstrap.

4.7.2 Le calcul des k-plus proches voisins

L'estimation des probabilités a posteriori P_{knn} et P_{comb} nécessite un calcul de k-ppv pour chaque niveau d'une McCascade. Un calcul exact de ces voisins pénaliserait grandement le processus de classification en terme de temps d'execution. La méthode de Muja et Lowe [46], qui permet une estimation rapide des k-ppv, a donc été retenue. Cette méthode est implémentée dans la librairie FLANN [3].

4.7.3 Les classifieurs faibles manquants

Pour évaluer les différents points d'une McCascade (la formulation probabiliste, l'estimation des probabilités a posteriori et le calcul des seuils), différents taux de classifieurs faibles manquants ont été considérés. Pour un taux donné et pour chaque niveau, 2 ensembles de classifieurs faibles à considérer comme manquant sont aléatoirement définis. Par exemple, un taux de 40% correspond à 4 classifieurs faibles manquants dans le deuxième niveau du détecteur. Les 2 ensembles de classifieurs faibles manquants pourraient être $\{h_{21}, h_{23}, h_{24}, h_{27}\}$ et $\{h_{22}, h_{23}, h_{26}, h_{28}\}$. Une fois ces ensembles définis, on obtient $2 \times 2 \times 2 = 8$ configurations différentes à tester.

4.8 Résultats

La base de test utilisée dans cette section est l'ensemble A de la base CMU-MIT qui consiste en 42 images contenant 169 visages de face avec des fonds variés [57]. Pour comparer différents systèmes, des courbes ROC sont utilisées car le nombre de sous-fenêtres testées reste le même entre les différents systèmes. Pour chaque taux de classifieurs faibles manquants, les 8 configurations testées amènent 8 courbes ROC différentes. Les moyennes et les écarts types sont ensuite calculés pour produire la courbe ROC finale. De plus, les performances exposées dans cette section sont brutes, i.e. que l'étape de fusion des détections multiples n'est pas réalisée. Nous avons fait ce choix pour ne pas biaiser les performances des différents systèmes testés. Pour un visage donné, si un classifieur produit plusieurs détections correctes, alors seule la plus pertinente est conservée (celle avec le meilleur score de classification). Les autres bonnes détections sont simplement ignorées et ne sont pas prises en compte dans la courbe ROC.

4.8.1 Évaluation de la formulation probabiliste

L'apport de la formulation probabiliste par rapport à la formulation classique est tout d'abord évalué. Pour cela, la solution naïve est comparée à une McCascade où les probabilités a posteriori sont estimées par P_{boost} mais les seuils de la McCascade ne sont pas calculés. Chaque β_j

3. disponible à l'adresse `http://people.cs.ubc.ca/ mariusm/index.php/FLANN/FLANN`

est simplement déduit des seuils τ_j :

$$\beta_j = \prod_{i=1}^{j} \frac{1}{1 + \mathrm{e}^{-\tau_i}} \tag{4.11}$$

Les deux classifieurs se différencient alors simplement sur un point : les niveaux sont indépendants dans la solution naïve alors qu'ils sont liés dans la McCascade. La solution naïve est notée « Cascade partielle » alors que le classifieur utilisant une structure en McCascade est noté « McCascade ». La figure 4.5 montre les performances des deux classifieurs pour des taux de classifieurs manquants compris entre 20% et 70%. Dans tous les cas, la formulation probabiliste améliore les performances. L'augmentation du taux de détection est minime lorsque le taux de classifieurs faibles manquants est inférieur à 30%. Au delà de 40% de classifieurs faibles manquants, on note une augmentation significative du taux de détection. Au plus, le taux de détection augmente de 45%. On note également une autre propriété intéressante de la formulation probabiliste : celle-ci offre des performances plus stables par rapport à la solution naïve. En effet, les écarts-types associés aux courbes de la McCascade sont généralement plus petits que les écarts-types des courbes de la solution naïve.

4.8.2 Évaluation des différentes stratégies de calcul des seuils β_j

Cette section se focalise sur l'évaluation des trois fonctions de coût proposées pour calculer les seuils β_j d'une McCascade : TP_cost, FP_TP_cost et FP_cost. Pour cela, plusieurs taux de classifieurs faibles manquants sont considérés : entre 20% et 70%. Pour chaque taux, les seuils sont d'abord calculés et le classifieur obtenu est ensuite appliqué sur la base de test (ceci est répété pour les 8 configurations). Les tests sont effectués pour les trois stratégies d'estimation des probabilités a posteriori : P_{boost}, P_{knn} et P_{comb}. Pour les deux dernières stratégies, le nombre de voisins est fixé à $k = 3$. Les résultats obtenus avec la stratégies P_{boost} sont donnés sur la figure 4.6. Les figures 4.7 et 4.8 donnent les résultats des stratégies P_{knn} et P_{comb}. On note que :

- la fonction TP_cost mène toujours aux meilleurs résultats ;
- la fonction FP_cost amène parfois des résultats inférieurs aux deux fonctions de coût ;
- les courbes ROC de la stratégie P_{knn} sont toujours identiques pour les trois fonctions de coût.

Les courbes ROC présentées suggèrent donc d'utiliser la fonction de coût TP_cost pour fixer les seuils d'une McCascade.

Les courbes ROC et FROC sont utiles pour mesurer les performances globales d'un classifieur. Le point de fonctionnement des classifieurs produits par les différentes fonctions de coût est un autre critère d'évaluation. Ce point de foncionnement est représenté par un taux de faux positifs, notés FP, et un taux de vrais positifs, noté VP. Les résultats pour un taux de classifieurs faibles manquants compris entre 20% et 70% sont regroupés dans le tableau 4.1. Dans chaque tableau, le nombre moyen de niveau de cascade évalué par exemple négatif, noté $\overline{nb}_{\text{niv}}$,

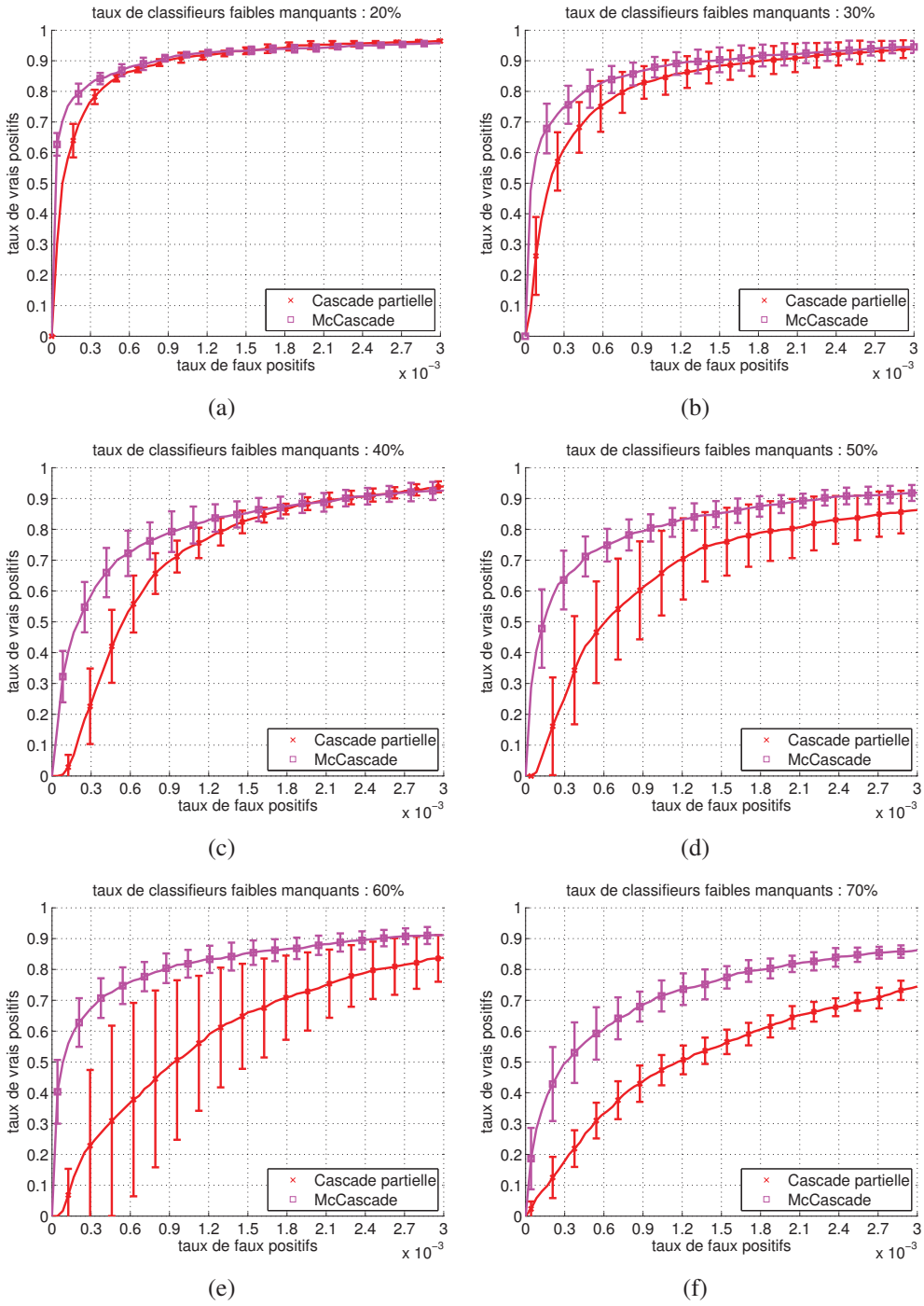


FIGURE 4.5 – Intérêt de l'utilisation d'une McCascade. La différence entre les deux classifieurs « Cascade partielle » et « McCascade » se situe seulement au niveau de la structure en cascade. Le premier classifieur utilise une structure classique tandis que le deuxième utilise une McCascade. Les seuils de la McCascade n'ont pas été recalculés, ils ont simplement été déduits des seuils de la cascade initiale. Différents taux de classifieurs faibles manquants sont testés : 20% en (a), 30% en (b), 40% en (c), 50% en (d), 60% en (e) et 70% en (f)

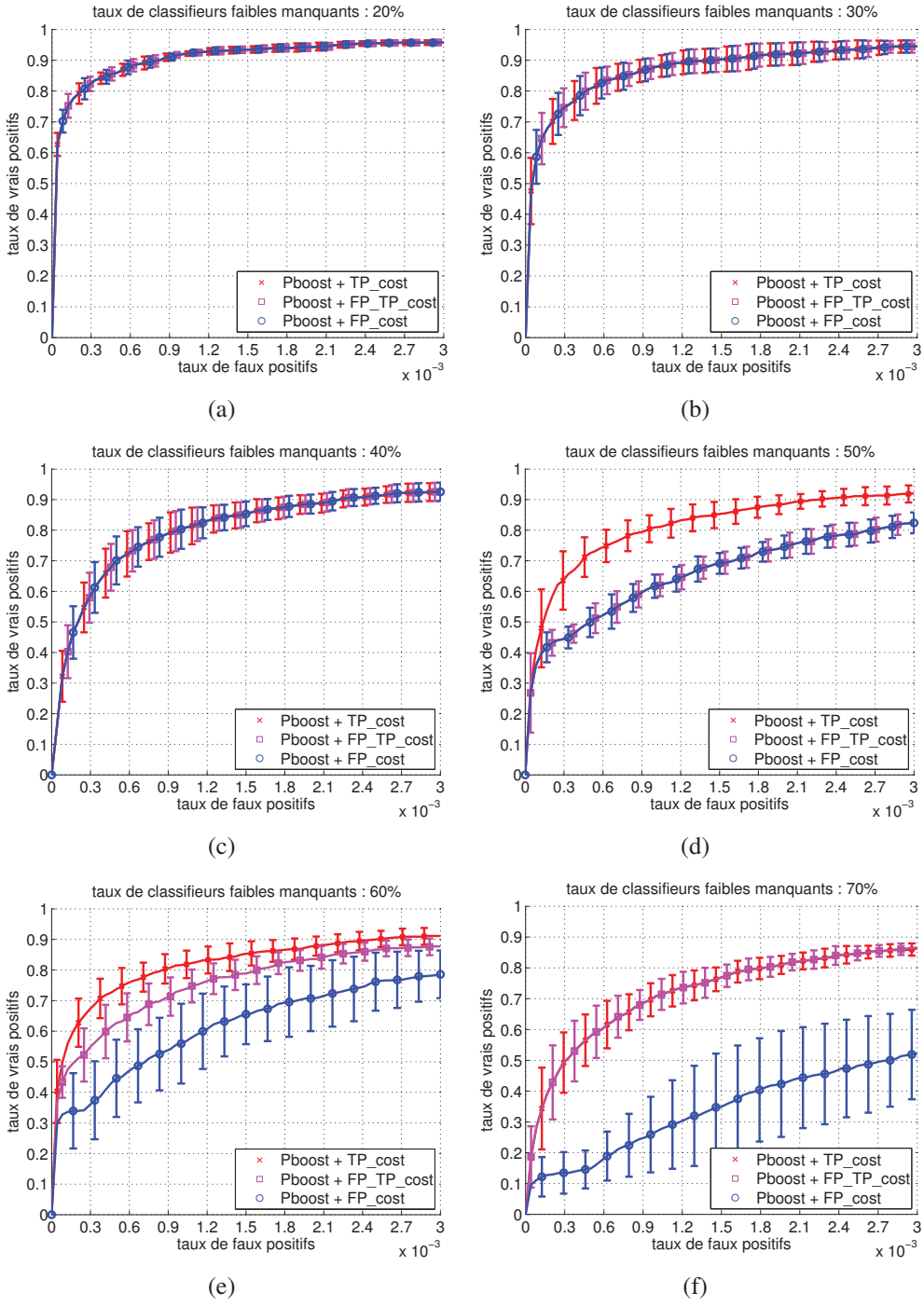


FIGURE 4.6 – Comparaison des trois fonctions de coûts TP_cost, FP_TP_cost et FP_cost pour la stratégie P_{boost}. Différents taux de classifieurs faibles manquants sont testés : 20% en (a), 30% en (b), 40% en (c), 50% en (d), 60% en (e) et 70% en (f)

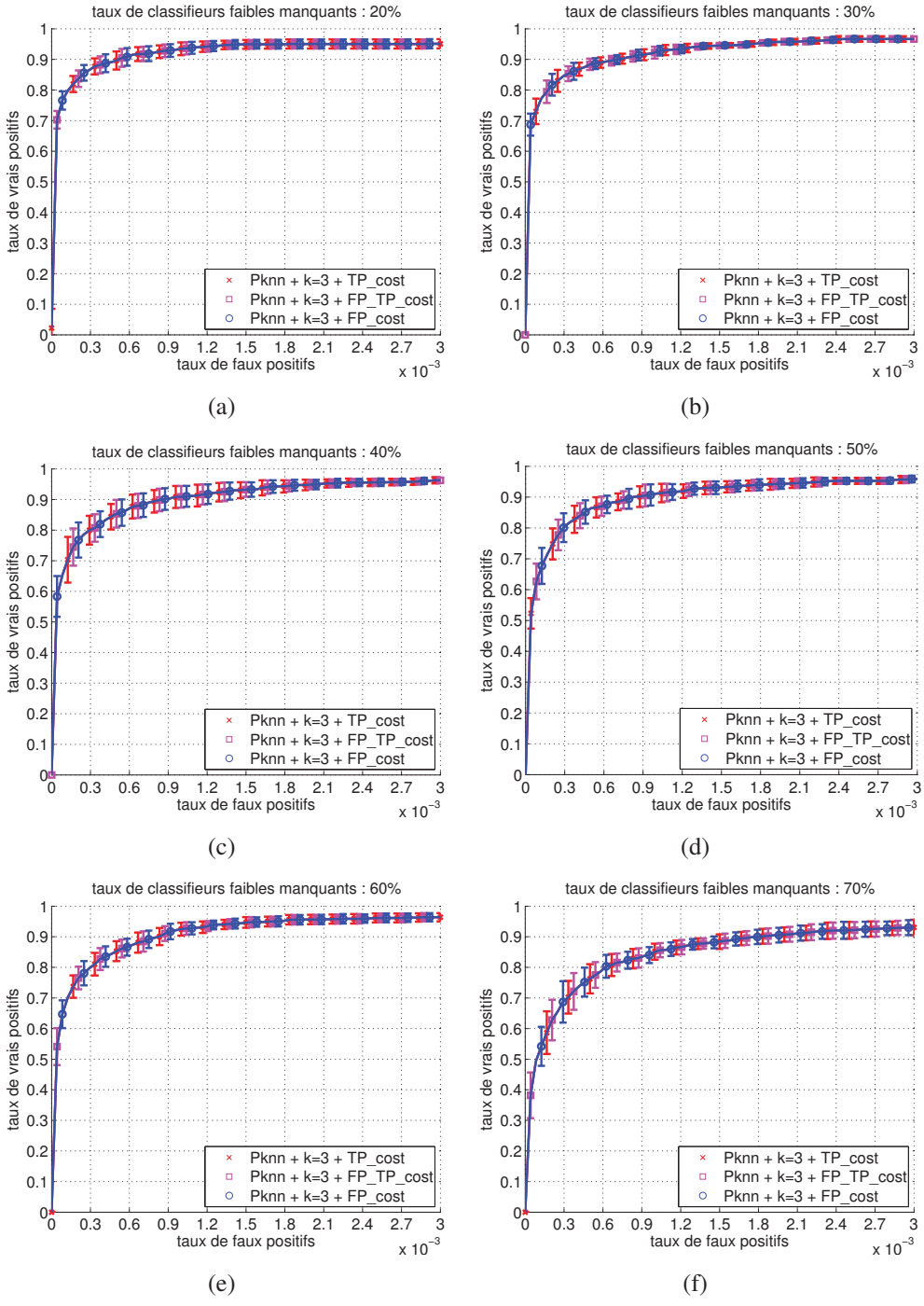


FIGURE 4.7 – Comparaison des trois fonctions de coûts TP_cost, FP_TP_cost et FP_cost pour la stratégie P_{knn} avec $k = 3$ voisins. Différents taux de classifieurs faibles manquants sont testés : 20% en (a), 30% en (b), 40% en (c), 50% en (d), 60% en (e) et 70% en (f)

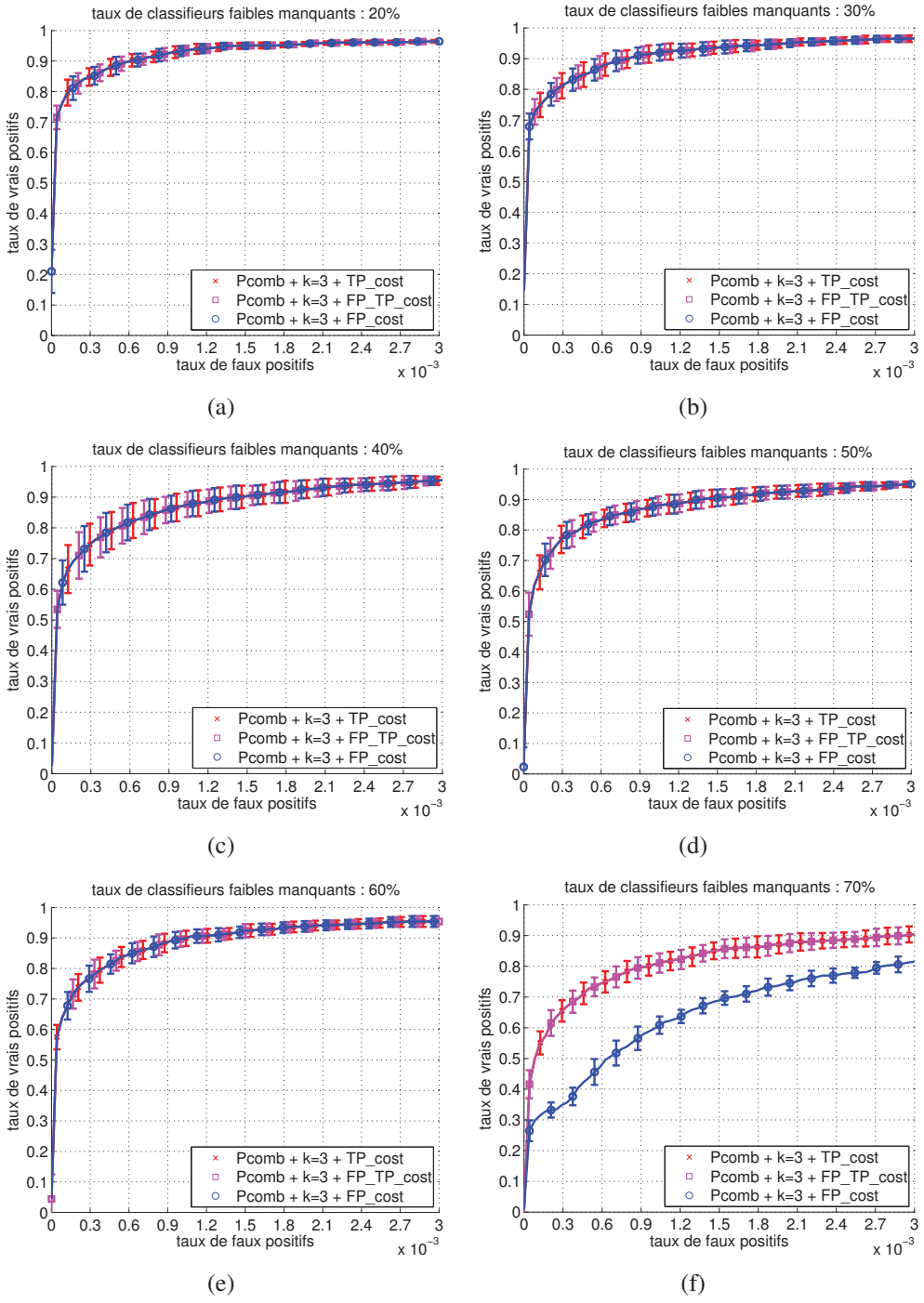


FIGURE 4.8 – Comparaison des trois fonctions de coûts TP_cost, FP_TP_cost et FP_cost pour la stratégie P_{comb} avec $k = 3$ voisins. Différents taux de classifieurs faibles manquants sont testés : 20% en (a), 30% en (b), 40% en (c), 50% en (d), 60% en (e) et 70% en (f)

est également présenté. Ce critère reflète l'impact de la fonction de coût sur le temps d'execution du classifieur. En effet, un nombre important de niveau évalué par négatif amènera un temps d'execution important.

Pour la stratégie P_{boost}, il est préférable d'utiliser la fonction FP_cost tant que le taux de classifieurs faibles manquants est strictement inférieur à 50%. En effet, les taux de détection VP sont très proches de ceux fournis par la fonction TP_cost alors que les taux de faux positifs FP sont bien inférieurs. À partir de 50% de classifieurs faibles manquants, seule la fonction TP_cost fournit un classifieur avec un taux de détection supérieur à 90%. Les classifieurs fournis par les fonctions FP_cost et FP_TP_cost sont inutilisables en pratique du fait de leurs trop faibles taux de vrais positifs.

Pour la stratégie P_{knn}, les trois fonctions de coût aboutissent au même point de fonctionnement pour des taux de classifieurs faibles manquants compris entre 20% et 50%. Pour un taux de 60% et de 70%, il devient préférable d'utiliser la fonction FP_cost car elle permet d'obtenir un classifieur avec un taux de détection supérieur à 90% tout en présentant le plus faible taux de faux positifs. De façon générale, la fonction FP_cost est donc préférable lorsque la stratégie P_{knn} est utilisée.

Enfin, pour la stratégie P_{comb}, on note qu'il est préférable d'utiliser la fonction FP_TP_cost pour un taux de classifieurs faibles manquants compris entre 20% et 60%. Pour un taux de 70%, seule la fonction TP_cost permet d'obtenir un taux de détection supérieur à 90% au prix d'un taux de faux positifs très élevés.

Pour une stratégie donnée, le critère $\overline{nb_{\text{niv}}}$ varie peu entre les différentes fonctions de coût. On remarque cependant que ce critère est toujours le plus bas avec la stratégie P_{knn} traduisant ainsi une meilleure faculté à rejeter les négatifs plus tôt dans la McCascade.

4.8.3 Évaluation de l'estimation de la probabilité a posteriori

Dans cette section, les trois stratégies proposées pour estimer les probabilités a posteriori : P_{boost}, P_{knn} et P_{comb}, sont comparées. Pour chaque taux de classifieurs faibles manquants (entre 20% et 70%), cinq classifieurs sont comparés :

1. « Cascade ». Ce classifieur est la cascade utilisant l'ensemble des classifieurs faibles. Ces performances représentent une borne supérieure pour les classifieurs qui n'utilisent qu'un sous-ensemble des classifieurs faibles. De plus, ces performances permettent de quantifier l'impact de la suppression d'une partie des classifieurs faibles ;
2. « Cascade partielle ». Ce classifieur représente la solution naïve présentée à la section 4.3 qui revient à utiliser la cascade initiale uniquement sur l'ensemble des classifieurs faibles disponibles ;
3. « McCascade + P_{boost} ». Ce classifieur est une McCascade utilisant l'ensemble des classifieurs faibles disponibles où les probabilités a posteriori sont estimées avec la stratégie P_{boost} ;

	fonction de coût	FP $\times 10^{-3}$	VP %	$\overline{nb_{\text{niv}}}$
P_{boost}	TP	3,08	96	1,36
	FP_TP	1,89	94	1,36
	FP	1,9	94	1,36
P_{knn}	TP	1,48	95	1,16
	FP_TP	1,48	95	1,16
	FP	1,48	95	1,16
P_{comb}	TP	1,8	96	1,27
	FP_TP	1,76	96	1,27
	FP	1,8	96	1,27

(a) taux de classifieurs faibles manquants : 20%

	fonction de coût	FP $\times 10^{-3}$	VP %	$\overline{nb_{\text{niv}}}$
P_{boost}	TP	10,98	97	1,51
	FP_TP	2,74	94	1,5
	FP	2,72	94	1,5
P_{knn}	TP	4,13	97	1,27
	FP_TP	4,13	97	1,27
	FP	4,13	97	1,27
P_{comb}	TP	6,22	98	1,45
	FP_TP	4,74	97	1,45
	FP	4,81	97	1,45

(b) taux de classifieurs faibles manquants : 30%

	fonction de coût	FP $\times 10^{-3}$	VP %	$\overline{nb_{\text{niv}}}$
P_{boost}	TP	7,06	97	1,47
	FP_TP	3,21	92	1,47
	FP	3,19	92	1,47
P_{knn}	TP	2,83	97	1,26
	FP_TP	2,83	97	1,26
	FP	2,83	97	1,26
P_{comb}	TP	5,46	97	1,44
	FP_TP	4,3	97	1,44
	FP	4,3	97	1,44

(c) taux de classifieurs faibles manquants : 40%

	fonction de coût	FP $\times 10^{-3}$	VP %	$\overline{nb_{\text{niv}}}$
P_{boost}	TP	3,53	93	1,62
	FP_TP	0,04	21	1,49
	FP	0,02	14	1,49
P_{knn}	TP	6,37	97	1,31
	FP_TP	6,37	97	1,31
	FP	6,37	97	1,31
P_{comb}	TP	22	98	1,91
	FP_TP	2,65	94	1,63
	FP	2,69	94	1,63

(d) taux de classifieurs faibles manquants : 50%

	fonction de coût	FP $\times 10^{-3}$	VP %	$\overline{nb_{\text{niv}}}$
P_{boost}	TP	9,34	97	1,66
	FP_TP	0,1	39	1,5
	FP	0,003	11	1,49
P_{knn}	TP	9,27	98	1,33
	FP_TP	6,19	98	1,33
	FP	6,19	98	1,33
P_{comb}	TP	39,4	98	1,96
	FP_TP	3,92	96	1,66
	FP	4	96	1,66

(e) taux de classifieurs faibles manquants : 60%

	fonction de coût	FP $\times 10^{-3}$	VP %	$\overline{nb_{\text{niv}}}$
P_{boost}	TP	23,44	97	1,88
	FP_TP	0,72	64	1,61
	FP	0,002	0,3	1,55
P_{knn}	TP	17,3	98	1,53
	FP_TP	9,33	97	1,53
	FP	6,07	94	1,53
P_{comb}	TP	30,07	98	1,92
	FP_TP	0,65	76	1,66
	FP	0,0002	1,7	1,54

(f) taux de classifieurs faibles manquants : 70%

TABLE 4.1 – Influence des fonctions de coût TP_cost (notée « TP »), FP_TP_cost (notée « FP_TP ») et FP_cost (notée « FP ») sur le point de fonctionnnement d'une McCascade pour différents taux de classifieurs faibles manquants. Ce point de fonctionnement est caractérisé par le taux de faux positifs « FP », le taux de vrais positifs « VP » et le nombre moyen de niveau de cascade évalué par exemple négatif « $\overline{nb_{\text{niv}}}$ »

4. « McCascade + P_{knn} ». Ce classifieur est une McCascade utilisant l'ensemble des classifieurs faibles disponibles où les probabilités a posteriori sont estimées avec la stratégie P_{knn} ;
5. « McCascade + P_{comb} ». Ce classifieur est une McCascade utilisant l'ensemble des classifieurs faibles disponibles où les probabilités a posteriori sont estimées avec la stratégie P_{comb} ;

Les seuils des trois McCascades sont calculés avec la fonction de coût TP_cost. Plusieurs valeurs du nombre de voisins k ont été testées (3, 7 et 13) pour les stratégies P_{knn} et P_{comb} et seuls les meilleurs résultats sont présentés. Les résultats sont disponibles sur la figure 4.9. Pour chaque taux de classifieurs faibles manquants considéré, les meilleures performances sont obtenues avec la stratégie P_{knn}. On peut également noter que les performances d'une McCascade utilisant la stratégie P_{knn} restent très proches des performances de la cascade utilisant l'ensemble des classifieurs faibles tant que le taux de classifieurs manquants ne dépassent pas 60%. À partir de 70%, les performances commencent à chuter mais restent néanmoins très supérieures aux performances de la solution naïve. Enfin, on peut retenir que la stratégie P_{boost} représente un bon compromis entre performance et vitesse. En effet, celle-ci propose des performances supérieures à la solution naïve et ne nécessite pas de recherche des k-ppv contrairement aux stratégies P_{knn} et P_{boost}.

4.8.4 Influence du nombre de voisins

Dans cette dernière série de test, l'influence du nombre de voisins utilisés pour calculer les k-ppv dans la stratégie P_{knn} est évalué. Comme dans les sections précédentes, nous considérons un taux de classifieurs faibles manquants compris entre 20% et 70%. Pour chaque taux, trois valeurs pour le nombre de voisins k sont testées : 3, 7 et 13. La figure 4.10 expose l'ensemble des résultats. On contaste que les meilleurs résultats sont obtenus pour $k = 3$ et que les performances se dégradent losque le nombre de voisins est augmenté.

4.9 Bilan

Dans ce chapitre, une solution originale de gestion de classifieurs faibles manquants dans une cascade de classifieurs forts a été présentée. Cette solution repose sur trois éléments :

1. une formulation probabiliste de la structure en cascade. Les tests ont montré que cette formulation permettait d'obtenir des performances supérieures à la solution naïve ;
2. une estimation des probabilités a posteriori à chaque niveau. Trois stratégies d'estimation ont été proposées et les tests ont permis de conclure que la stratégie P_{knn} était à préférer. De plus, les meilleures performances sont obtenues pour un nombre de voisins k égal à 3 ;

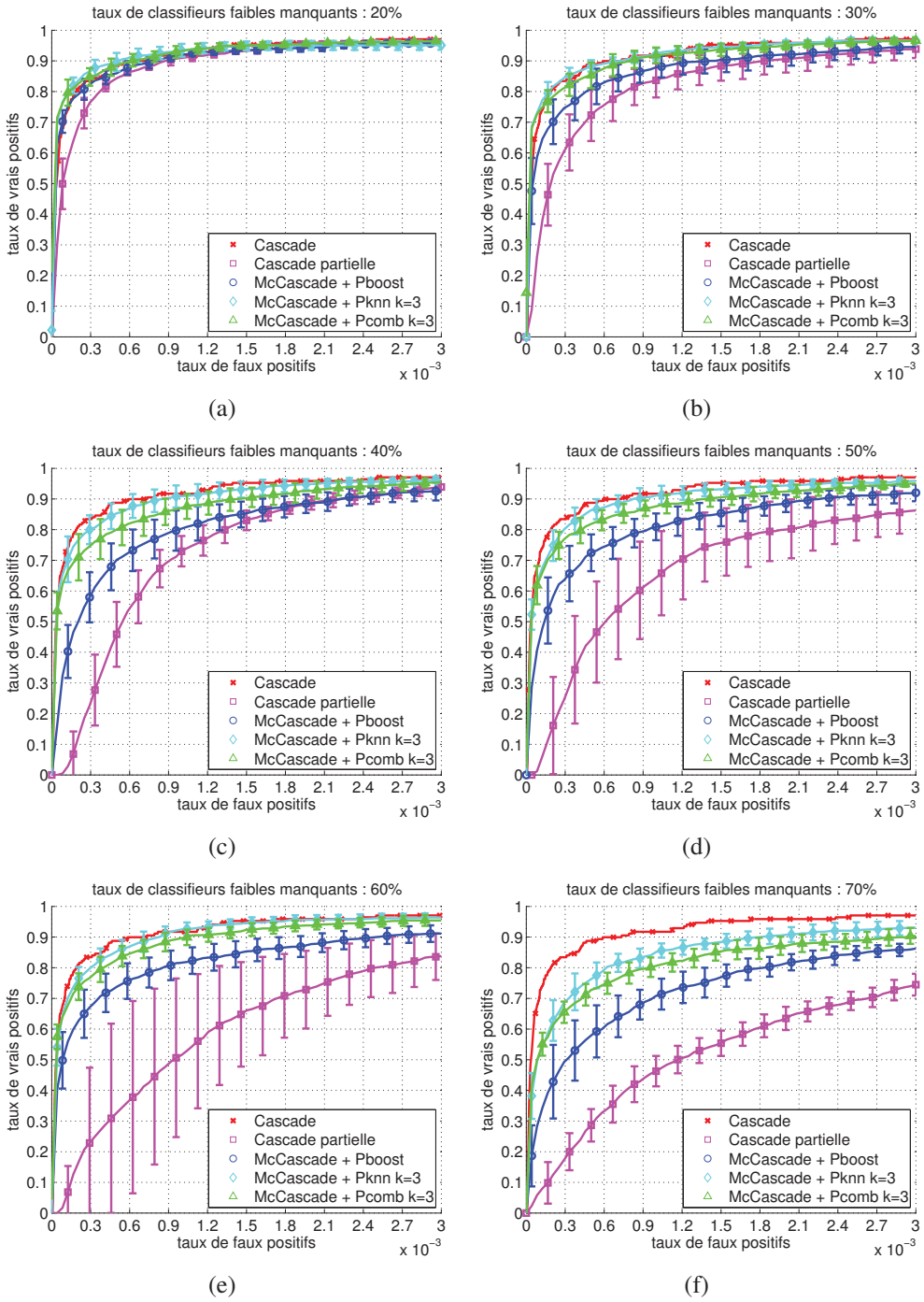


FIGURE 4.9 – Comparaison de différentes stratégies d'estimation des probabilités a posteriori dans une McCascade. Différents taux de classifieurs faibles manquants ont été testés : 20% en (a), 30% en (b), 40% en (c), 50% en (d), 60% en (e) et 70% (f). Trois McCascades sont présentées : la première utilise P_{boost}, la seconde P_{knn} et la troisième P_{comb}. Chacune peut être comparée à la solution naïve, notée « Cascade partielle ». De plus, sur chaque courbe, les performances de la cascade utilisant l'ensemble des classifieurs faibles, notée « Cascade », sont présentées

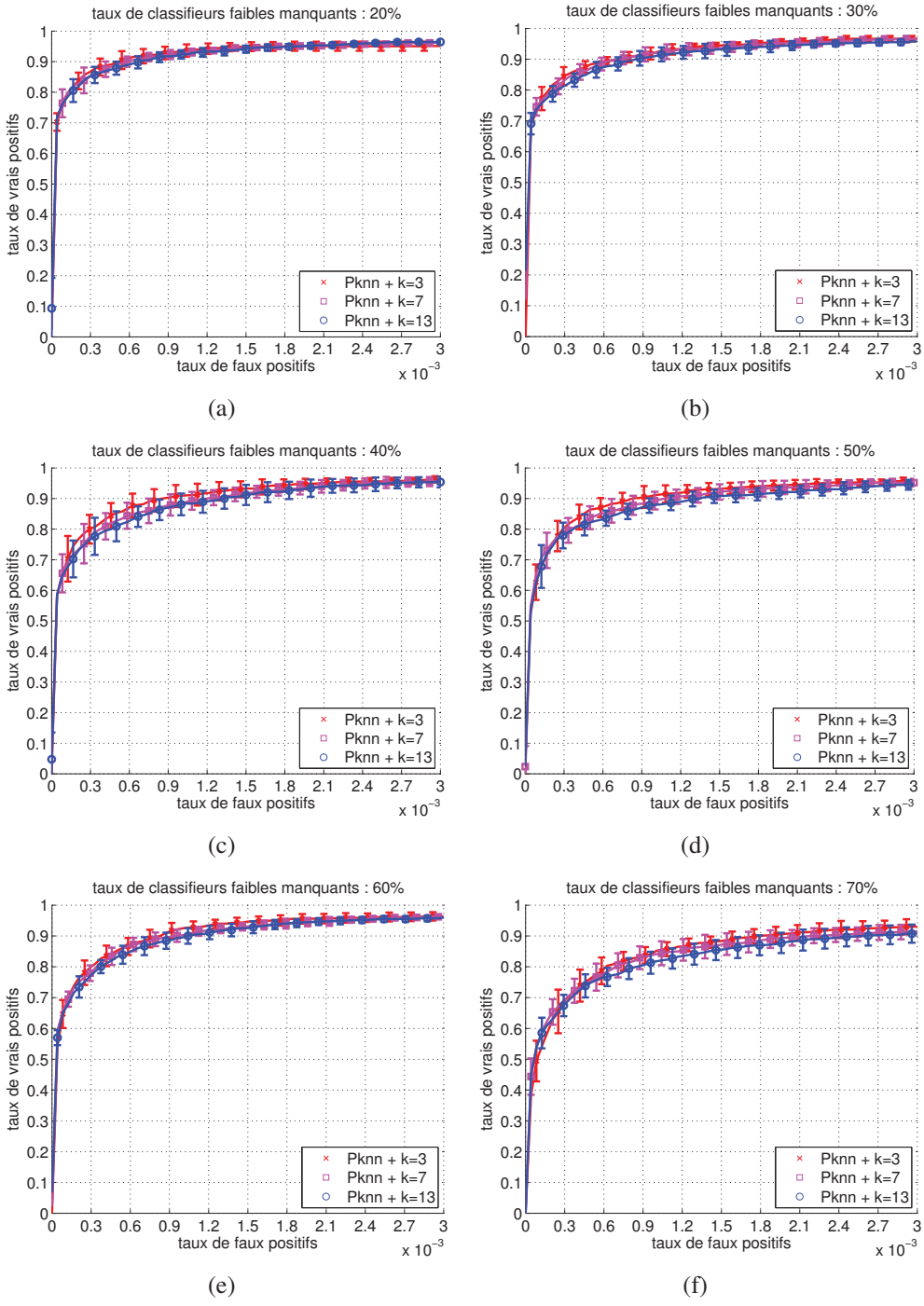


FIGURE 4.10 – Influence du nombre de voisins dans la stratégie P_{knn} pour différentes taux de classifieurs faibles manquants : 20% en (a), 30% en (b), 40% en (c), 50% en (d), 60% en (e) et 70% en (f)

3. un calcul itératif des seuils de la McCascade. Ce calcul s'appuie sur une fonction de coût. Trois fonctions de coût ont été proposées : FP_cost, FP_TP_cost et TP_cost. Lorsque la stratégie P_{knn} est utilisée, le choix de cette fonction de coût n'est pas critique vis à vis des performances. Cependant, pour un taux de classifieurs faibles manquants supérieur à 60%, la fonction FP_cost est préférable.

Chapitre 5

Détection de visages tournés ou occultés

Dans ce chapitre, les deux applications visées sont présentées, à savoir la détection de visages tournés et la détection de visages occultés. Dans les deux cas, un détecteur de visages de face est modifié pour s'adapter aux nouvelles conditions de détection. Les modifications envisagées conduisent à une disparition de certains classifieurs faibles dans la cascade initiale. Cette dernière est alors transformée en une McCascade pour gérer les classifieurs faibles manquants. Tout d'abord, l'application de détection de visages tournés est présentée dans la section 5.1. La section 5.2 traite ensuite de la détection de visages occultés. Les différents systèmes sont ensuite évalués dans la section 5.4.

5.1 Détection de visages tournés

Un visage tourné est un visage ayant subi une rotation hors du plan, comme illustré sur la figure 5.1. Le problème de la détection de visages avec rotation n'est ici pas abordé. Son but est de créer un détecteur invariant aux rotations (les tests effectués au chapitre 3 montrent qu'un détecteur utilisant des matrices de covariance peut détecter des visages avec des rotations maximales de $\pm 15°$). Un détecteur multi-vues est capable de détecter des visages tournés de différents angles. La première idée que l'on peut avoir pour créer un détecteur multi-vues est d'enrichir la base d'apprentissage avec des visages tournés. Malheureusement, cette approche conduit généralement à des situations où l'algorithme d'apprentissage ne fonctionne pas du fait de la trop grande variabilité au sein de la classe des positifs (notons toutefois que les réseaux de neurones convolutionnels fonctionnent sur des bases contenant des visages tournés entre $-60°$ et $+60°$ [17]). La solution consiste alors à apprendre différents détecteurs, chacun d'eux étant spécialisé pour détecter des visages dans une pose donnée, puis de les combiner pour obtenir un détecteur multi-vues. Cette approche est appelée approche basée vue.

(a) (b) (c) (d)

FIGURE 5.1 – Exemples de visages avec différentes rotations hors du plan. Les visages sont tournés de différents angles autour de l'axe y : $90°$ en (a), $67.5°$ en (b), $45°$ en (c) et $22.5°$ en (d)

5.1.1 État de l'art

Le premier détecteur multi-vues est celui de Schneiderman et Kanade [64]. Il permet de détecter des visages de face ainsi que de profil. Pour chaque vue, un détecteur est appris : un détecteur de visages de face et un détecteur de visages de profil droit (le détecteur de profil gauche est ensuite obtenu en appliquant le détecteur de profil droit sur la version miroir de l'image à scanner). Chacun est construit à l'aide de la modélisation des distributions $P(\text{image}|\text{object})$ et $P(\text{image}|\text{non-object})$ (chaque distribution est représentée par plusieurs histogrammes). Les trois détecteurs sont ensuite appliqués en parallèle et si plusieurs répondent positivement, la détection avec le plus grand score de classification est conservée. Pour valider leur approche, ils créent la base de test CMU profil encore utilisée aujourd'hui.

Li et al. [34] ont appliqué leur algorithme FloatBoost au problème de détection de visages multi-vues. Ils utilisent une structure en pyramide (voir figure 5.2(a)) à trois niveaux. Le niveau 1 est un classifieur détectant des visages entre $-90°$ à $+90°$. Le niveau 2 est constitué de trois classifieurs 1) de $-90°$ à $-30°$, 2) de $-30°$ à $+30°$ et 3) de $+30°$ à $+90°$. Enfin, le niveau 3 est constitué de sept classifieurs 1) de $-90°$ à $-60°$, 2) de $-60°$ à $-30°$, 3) de $-30°$ à $-10°$, 4) de $-10°$ à $+10°$, 5) de $+10°$ à $+30°$, 6) de $+30°$ à $+60°$ et 7) de $+60°$ à $+90°$. Ils aboutissent à un détecteur multi-vues nécessitant 200ms pour scanner une image 320x240.

Jones et Viola [27] ont proposé une détection multi-vues en deux étapes 1) estimation de la pose par un arbre de décision et 2) classification par un des trois classifieurs spécialisés : profil droit, face ou profil gauche (voir figure 5.2(b)). Ils remarquent que les ondelettes de Haar qu'ils avaient précédemment utilisées [74] ne sont pas suffisamment discriminantes pour estimer la pose et apprendre des visages tournés. Ils introduisent donc des ondelettes diagonales.

Huang et al. [23] ont appliqué le principe de cascade imbriquée à la détection multi-vues. Cinq vues dont définies : profil droit/gauche, demi-profil droit/gauche et face, chacune étant associée à un détecteur. Leur détecteur multi-vues utilise le même principe que celui proposé par Jones et Viola [27], à savoir une première étape permet d'estimer la pose et la seconde étape consiste à classer l'exemple par le détecteur le plus adapté (voir la figure 5.2(c)). L'estimation de la pose consiste à appliquer les trois premiers niveaux des cinq détecteurs puis à conserver le détecteur avec le plus fort score de classification.

Lin et Liu [37] ont introduit l'algorithme MBHboost qui est un algorithme d'apprentis-

sage multi-classes. Les classifieurs faibles sont des fonctions vectorielles dont chaque composante représente une classe. Chacun est appris sur une même ondelette de Haar mais chaque composante dispose de sa propre fonction de décision. Plusieurs applications sont présentées : détecteur multi-vues, détecteur invariant aux rotations, détecteur robuste aux conditions d'illumination et détecteur robuste aux occultations. Le détecteur multi-vues peut détecter neuf types de vue à une vitesse de 13,8 images par seconde pour des images 320x240.

Huang et al. [22] ont présenté un détecteur de visages multi-vues invariant aux rotations. Ils s'appuient sur un nouvel algorithme d'apprentissage : Vector Boosting et sur une nouvelle structure en arbre : Width-First-Search (voir figure 5.2(d)). Contrairement aux structures existantes qui considèrent qu'un exemple est uniquement traité par le détecteur le plus probable, ils proposent qu'un exemple soit traité par plusieurs détecteurs si plusieurs détecteurs répondent positivement dans l'arbre.

5.1.2 Prise en compte de la pose dans un classifieur

Dans cette thèse, l'approche basée vue est adoptée afin de concevoir un détecteur multi-vues. Cependant, contrairement aux solutions existantes, seul un détecteur de visages de face est à disposition. Par construction, les détecteurs de visages de face sont généralement robustes aux faibles rotations hors du plan (de l'ordre de $\pm 20°$). Au delà de cet angle, l'apparence des visages change trop pour être prise en charge par le détecteur de visages de face (voir figure 5.3(b)). La solution proposée, illustrée sur la figure 5.3(c), consiste à ajuster la position de toutes les sous-fenêtres sélectionnées pendant l'apprentissage. Elle permet de créer un nouveau classifieur capable de détecter des visages avec un angle donné. Il faut savoir que que le classifieur faible associé à chaque sous-fenêtre n'est pas modifié. Seule la position de chaque sous-fenêtre est modifiée.

Pour modifier la position d'une sous-fenêtre, nous proposons d'utiliser la transformation 3D entre un visage de face et un visage tourné. Les transformations concernées sont l'ensemble des rotations autour de l'axe vertical. Pour représenter ces rotations, un modèle 3D de visage est nécessaire. La construction d'un modèle 3D précis nécessite au moins deux images par visage (une de face et une de profil par exemple). Cependant, l'objectif est de n'utiliser qu'un détecteur de visages de face. Ainsi, le recours à des images d'apprentissage de visages qui ne sont pas de face est proscrit. Un visage est donc représenté par un modèle très simple : une ellipsoïde. L'idée est ensuite de placer la sous-fenêtre sur l'ellipsoïde, de faire tourner l'ellipsoïde et de récupérer la position de la sous-fenêtre après rotation. Plus formellement, considérons un point $p_1^i = (u_1\ v_1)^T$ dans une image de taille $w \times w$ (cette taille correspond à la taille des images d'apprentissage). Les coordonnées de p_1^i sont exprimées dans le repère image $\mathcal{R}_i$. Le processus permettant de calculer la position de ce point après une rotation d'angle θ_y autour de l'axe y est composé des trois étapes suivantes :

1. Un point P_1^i appartenant à l'ellipsoïde est associé au point p_1^i. Il suffit de calculer la coordonnée w_1 suivant l'axe z en utilisant l'équation de l'ellipsoïde exprimée dans $\mathcal{R}_i$

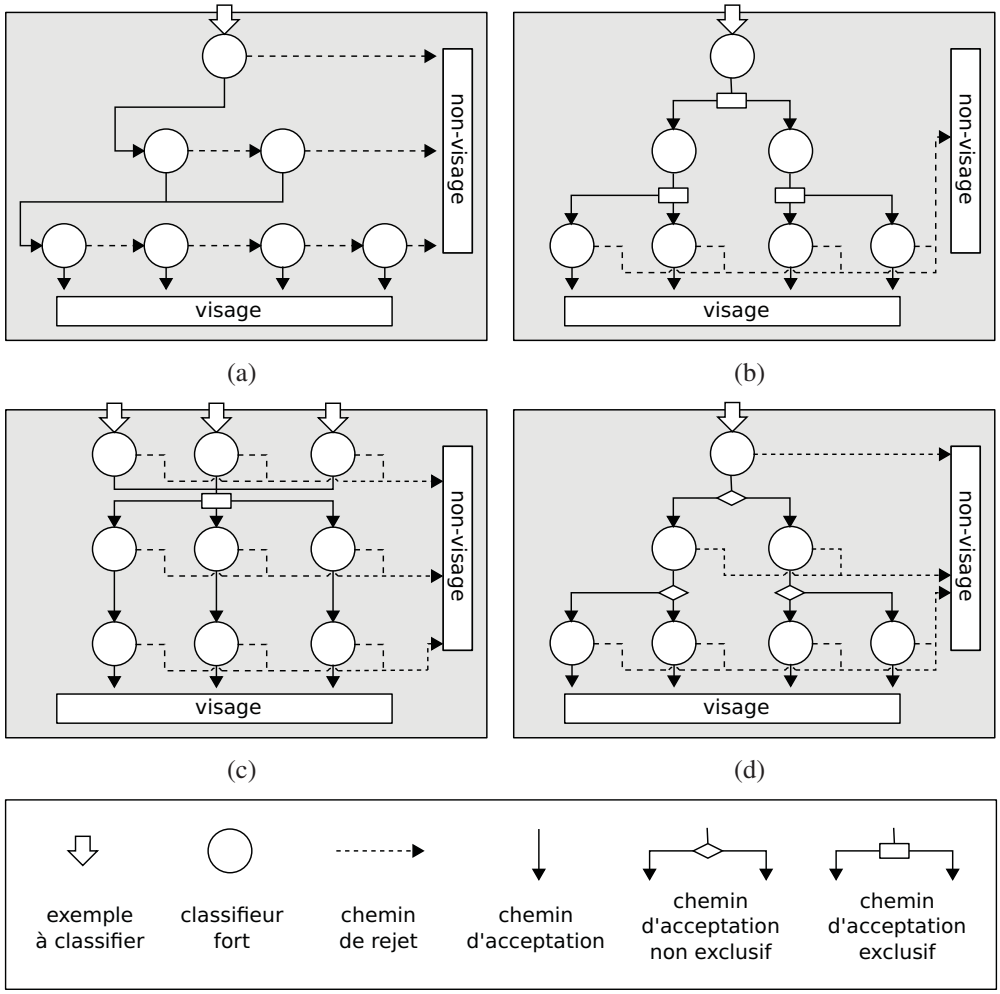
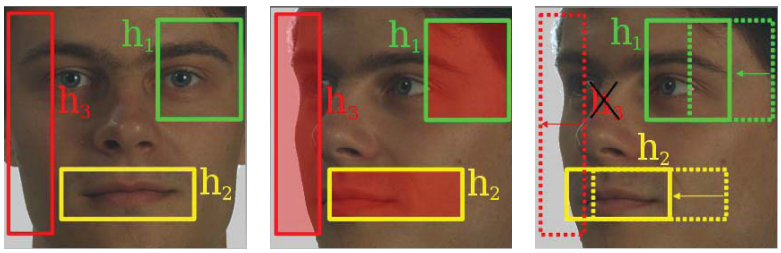


FIGURE 5.2 – Illustration de différentes structures de détecteurs multi-vues. (e) - légende (a) pyramide de Li et al. [34] (b) - arbre de décision de Jones et Viola [27] (c) - Estimateur de pose de Huang et al. [23] (d) - Arbre WFS (*Width-First-Search*) de Huang et al. [22]



(a) (b) (c)

FIGURE 5.3 – Mise en défaut d'un détecteur de visages de face sur un visage à moitié de profil. (a) - un exemple de trois sous-fenêtres intéressantes sélectionnées pendant l'apprentissage du détecteur de visages de face accompagnées de leurs classifieurs faibles h_1, h_2 et h_3 (b) - les trois sous-fenêtres ne sont plus intéressantes et le visage à moitié de profil est rejeté (c) - pour pouvoir détecter des visages à moitié de profil, les positions de toutes les sous-fenêtres sont ajustées. On remarque que la modification des positions des sous-fenêtres entraîne la disparition du classifieur h_3

(voir figure 5.4(a)) :

$$\frac{(u \quad u_0)^2}{a^2} + \frac{(v \quad v_0)^2}{b^2} + \frac{(w \quad w_0)^2}{c^2} = 1 \tag{5.1}$$

où $u_o = w/2$, $v_o = w/2$, $w_o = 0$ et a, b et c représentent les paramètres de l'ellipsoïde.

2. P_1^i est exprimé dans le repère $\mathcal{R}_e$ dont l'origine est le centre de l'ellipsoïde. Cela donne le point P_1^e. Le passage du repère $\mathcal{R}_i$ à $\mathcal{R}_e$ s'effectue par une translation de vecteur $(\quad w/2 \quad w/2 \; 0)^{\mathrm{T}}$:

$$\begin{bmatrix} \tilde{x}_1 \\ \tilde{y}_1 \\ \tilde{z}_1 \\ \tilde{d}_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & w/2 \\ 0 & 1 & 0 & w/2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ v_1 \\ w_1 \\ 1 \end{bmatrix} \tag{5.2}$$

et on obtient $P_1^e = (\tilde{x}_1/\tilde{d}_1 \; \tilde{y}_1/\tilde{d}_1 \; \tilde{z}_1/\tilde{d}_1)^T = (x_1 \; y_1 \; z_1)^T$. La rotation est ensuite appliquée à ce point pour obtenir le point P_2^e (voir figure 5.4(b)) :

$$P_2^e = (x_2 \; y_2 \; z_2)^T = R_y(\theta_y) \times P_1^e \tag{5.3}$$

où $R_y(\theta_y)$ est la matrice de rotation autour de l'axe y :

$$R_y(\theta_y) = \begin{pmatrix} \cos(\theta_y) & 0 & \sin(\theta_y) \\ 0 & 1 & 0 \\ \sin(\theta_y) & 0 & \cos(\theta_y) \end{pmatrix} \tag{5.4}$$

3. Pour finir, P_2^e est exprimé dans $\mathcal{R}_i$ pour obtenir le point P_2^i (voir figure 5.4(c)) :

$$\begin{bmatrix} \tilde{u}_2 \\ \tilde{y}_2 \\ \tilde{z}_2 \\ \tilde{d}_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & w/2 \\ 0 & 1 & 0 & w/2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^1 \begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} \tag{5.5}$$

et $P_2^i = (\tilde{u}_2/\tilde{d}_2\ \tilde{v}_2/\tilde{d}_2\ \tilde{w}_2/\tilde{d}_2)^T = (u_2\ v_2\ w_2)^T$. Le point final recherché est $p_2^i = (u_2\ v_2)^T$.

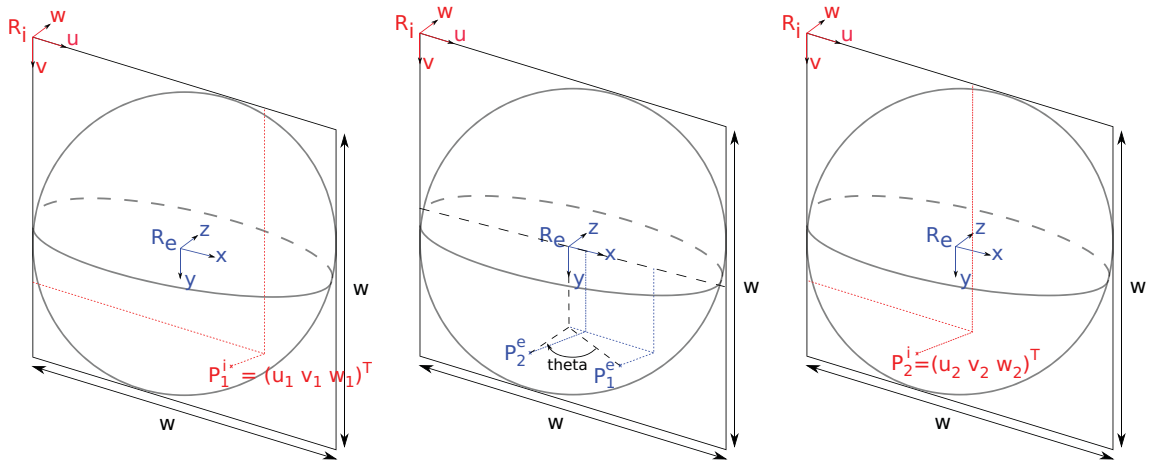
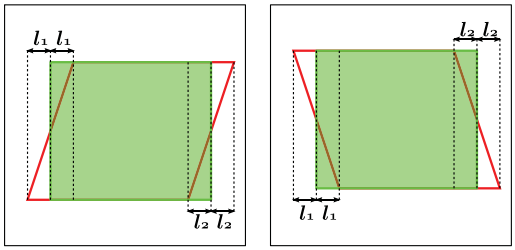


FIGURE 5.4 – Processus de rotation d'un point en utilisant une ellipsoïde. (a) - le point image $p_1^i = (u_1, v_1)$ est associé avec le point $P_1^i = (u_1, v_1, w_1)$ sur l'ellipsoïde en s'appuyant sur l'équation de l'ellipsoïde (b) - P_1^i est exprimé dans le repère de l'ellipsoïde, ce qui donne le point P_1^e. Le point après rotation P_2^e est calculé en utilisant la matrice de rotation autour de l'axe y (c) - P_2^e est exprimé dans le repère image, ce qui donne le point $P_2^i = (u_2, v_2, w_2)$ et le point image recherché $p_2^i = (u_2, v_2)$

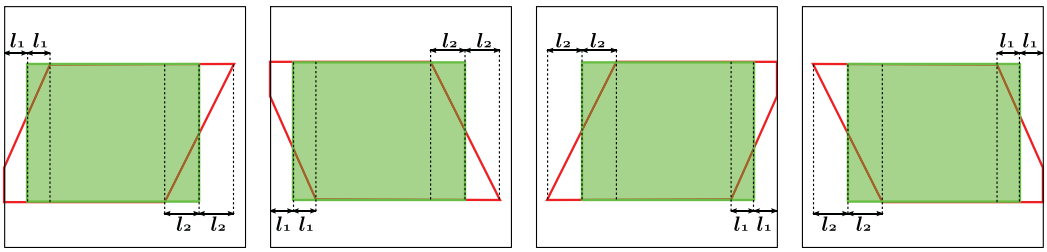
Pour connaître la position d'une sous-fenêtre r_{jt} après une rotation, le processus décrit ci-dessus est appliqué aux quatres coins de r_{jt}. Du fait du modèle 3D utilisé, les quatres coins obtenus après rotation ne forment pas une sous-fenêtre. En effet, plus l'ordonnée v_1 est proche de $w/2$ et plus l'effet de la rotation est important. De plus, la sous-fenêtre après rotation n'a pas toujours ces quatre coins visibles. Dans le cas où quatre, trois ou deux coins sont visibles, des heuristiques sont utilisées pour déduire une sous-fenêtre à partir des coins obtenus après rotation. Ces dernières sont illustrées dans le cas de quatre coins visibles sur la figure 5.5. La figure 5.6 illustre les cas pour lesquels trois coins sont visibles. Enfin, la figure 5.7 illustre les cas où seulement deux coins sont visibles.

La possible disparition de certaines sous-fenêtres pose problème (comme sur la figure 5.3(c) avec la sous-fenêtre de h_3). On considère qu'une sous-fenêtre disparaît dans deux situations : 1)



(a) (b)

FIGURE 5.5 – Déduction d'une sous-fenêtre avec quatre coins visibles. Deux cas possibles sont présentés en (a) et (b). Dans chaque cas, la sous-fenêtre en rouge est celle obtenue après rotation tandis que la sous-fenêtre verte est celle déduite et utilisée dans le classifieur



(a) (b) (c) (d)

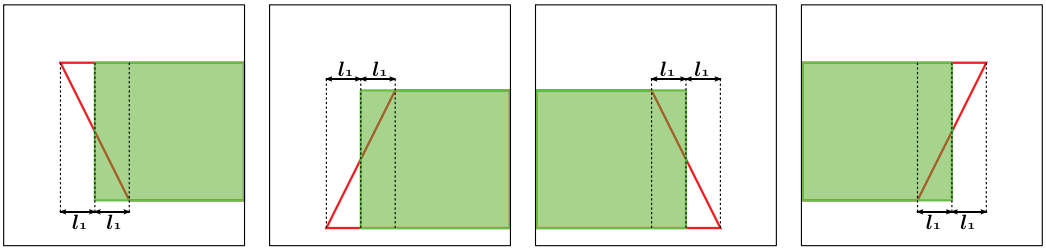
FIGURE 5.6 – Déduction d'une sous-fenêtre avec trois coins visibles. Quatre cas possibles sont présentés en (a), (b), (c) et (d). Dans chaque cas, la sous-fenêtre en rouge est celle obtenue après rotation tandis que la sous-fenêtre verte est celle déduite et utilisée dans le classifieur

un seul coin est visible après rotation et 2) aucun coin n'est visible après rotation. Si une sous-fenêtre r_{jt} disparaît, alors le classifieur faible associé h_{jt} devient indisponible. En examinant l'ensemble des sous-fenêtres, l'ensemble des classifieurs faibles disponibles peut être défini et une McCascade peut être construite en se basant sur cet ensemble. Ainsi, créer un classifieur pouvant détecter des visages tournés d'un certain angle nécessite trois étapes :

1. modification de la position de toutes les sous-fenêtres en utilisant le modèle ellipsoïdique ;
2. définition de l'ensemble des classifieurs faibles disponibles en vérifiant que les sous-fenêtres associées n'ont pas disparu après rotation ;
3. création d'une McCascade utilisant les classifieurs faibles disponibles.

5.1.3 Proposition d'un système multi-vues

La solution présentée à la section précédente permet de détecter des visages tournés d'un certain angle θ_y. Lorsqu'on souhaite détecter des visages tournés selon un angle compris dans un intervalle $[\ \theta_y^{\min}, +\theta_y^{\max}]$, une solution consiste à combiner plusieurs détecteurs. Chacun des



(a) (b) (c) (d)

FIGURE 5.7 – Déduction d'une sous-fenêtre avec deux coins visibles. Quatre cas possibles sont présentés en (a), (b), (c) et (d). Dans chaque cas, la sous-fenêtre en rouge est celle obtenue après rotation tandis que la sous-fenêtre verte est celle déduite et utilisée dans le classifieur

détecteurs est spécialisé dans la détection de visages selon un angle donné θ_y. En réalité, on suppose généralement que chacun est capable de détecter des visages dans l'intervalle $[\theta_y$ $15, \theta_y + 15]$. Si on souhaite détecter des visages tournés d'un angle $\theta_y \in [\ \ 45, +45]$, trois détecteurs sont alors nécessaires : un détecteur de visages de face H^0, un détecteur de visages tournés de $+30°$ H^{+30} et un détecteur de visages tournés de $\ \ 30°$ $H\ \ ^{30}$. Les détecteurs $H\ \ ^{30}$ et H^{+30} sont obtenus en modifiant la position des sous-fenêtres du détecteur H^0. Pour combiner les trois détecteurs, le principe proposé par [23], illustré sur la figure 5.8, est appliqué. Pour accélérer le traitement, un estimateur de pose est utilisé. L'estimation de la pose consiste à appliquer les trois premiers niveaux de chaque détecteur et à continuer avec le détecteur qui accepte l'exemple $\mathbf{x}$ avec le plus fort score de classification. La fonction d'estimation de pose est ainsi définie par :

$$\text{pose}(\mathbf{x}) = \underset{\theta_y \in \{\ \ 30,0,30\}}{\text{argmax}} \left(H_3^{\theta_y}(\mathbf{x}) \right) \tag{5.6}$$

5.2 Détection de visages occultés

Comme la détection de visages tournés, la détection de visages occultés nécessite des solutions spécifiques car un détecteur de visage de face sera facilement mis en échec sur des images de visages occultés. En effet, les descripteurs calculés sur les zones de visage occulté vont contribuer au rejet du visage par le classifieur. Pour les mêmes raisons que les visages tournés, l'enrichissement de la base d'apprentissage avec des visages occultés ne représente pas une solution intéressante.

5.2.1 État de l'art

Le problème des visages occultés a fait l'objet de nombreux travaux dans le cas de la reconnaissance faciale [5, 24, 41, 26, 78]. Dans le cas de la détection de visages, les travaux effectués

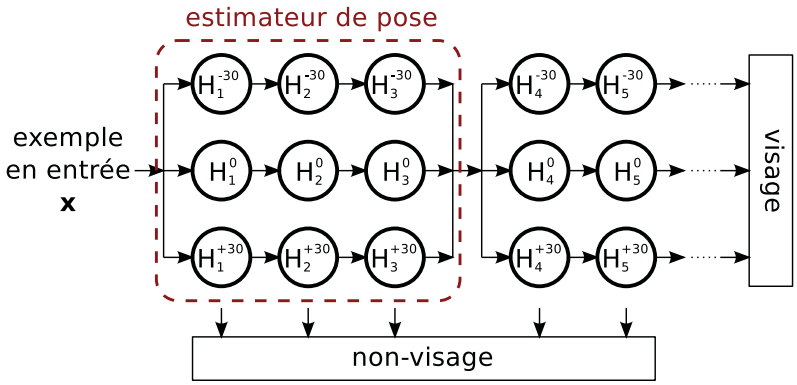


FIGURE 5.8 – Principe du système multi-vues retenu. L'exemple à classifier $\mathbf{x}$ passe d'abord par les trois premiers niveaux des trois détecteurs H^{-30}, H^{0} et H^{+30}. La pose estimée de $\mathbf{x}$ est obtenue en considérant le détecteur acceptant $\mathbf{x}$ avec le plus fort score de classification. Puis, $\mathbf{x}$ continue le processus de classification avec le détecteur retenu pour estimer la pose

sont beaucoup moins nombreux. En 2004, Hotta [21] a utilisé un SVM pour détecter des visages occultés. La particularité de son appoche réside dans l'utilisation de noyaux locaux où chacun est associé à une partie restreinte du visage.

La même année, Lin et al. [38] ont construit un détecteur de visages occultés qui s'inspire de l'approche basée vue. En effet, plusieurs classifieurs cascadés sont appris, chacun d'entre eux étant spécialisé pour gérer un type d'occultation (huit types d'occultation sont définis). En plus de ces classifieurs spécialisés, une cascade principale est apprise. Enfin, les différentes cascades sont combinées à l'aide du principe de *cascading with evidence* qui permet d'aiguiller un exemple entre la cascade principale et une des cascade gérant les occultations.

Le système de Lin et Liu [37] présenté à la section 5.1.1 a également été appliqué à la détection de visages occultés. Leur algorithme d'apprentissage multi-classes est utilisé sur huit classes de visages correspondant à huit types d'occultations différentes. Ils obtiennent des performances comparables à leur précédent système présenté en [38].

Chen et al. [6] ont adapté le détecteur de Viola et Jones [74] pour obtenir un détecteur de visages robuste aux occultations. Pour cela, ils divisent la fenêtre de test en plusieurs parties et classifient chaque partie. L'inconvénient de leur approche est la structure en cascade qui est perdue, ce qui limite leur solution à des applications non temps-réel.

La solution de Chen et al. [6] s'inspire des approches basées parties dans lesquelles l'objet à détecter est représenté par différentes parties pertinentes. Un détecteur est appris pour chaque partie et les scores de classification associés à chaque partie sont finalement fusionnés pour produire la décision finale. Dans le domaine du visage, on peut citer les travaux de Heisele et al. [19] qui représentent le visage par 14 parties. À chaque partie est associé un SVM linéaire et un dernier SVM linéaire permet de fusionner les différents scores de classification. Même si leur solution présente de bonnes performances dans le cas de visages occultés, elle présente

également deux inconvénients spécifiques aux approches basées parties 1) la taille minimale des visages détectée est supérieure à celle des approches globales (58×58 dans leur cas contre 24×24 pour notre détecteur) et 2) le temps d'exécution est trop important pour viser des applications temps-réel.

Des travaux dans le domaine de la détection de piétons ont également été menés comme celui de Wu et Nevatia [79]. Ce dernier s'appuie sur la décomposition d'un piéton en plusieurs parties permettant une robustesse aux occultations. Celui de Wang et al. [76] combine des histogrammes de gradients et des LBP pour obtenir un détecteur de piétons robuste aux occultations.

5.2.2 Proposition d'un système robuste aux occultations

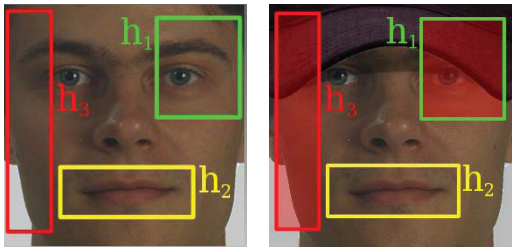
Le problème des visages occultés pour un détecteur de visages de face est illustré sur la figure 5.9. Supposons que l'on teste une fenêtre contenant un visage occulté : si les sous-fenêtres sélectionnées pendant l'apprentissage du détecteur de visages de face sont occultées (voir figure 5.9(b)), les classifieurs faibles associés risquent de répondre négativement. De façon plus globale, le visage sera probablement rejeté. Un système inspiré de celui de Lin et al. [38] est proposé pour détecter des visages occultés :

- la cascade principale $\mathcal{C}$ est constituée par le détecteur de visages de face ;
- en plus de la cascade principale, plusieurs cascades d'occultations sont créées. Chacune gère un type d'occultation particulière (occultation haute, basse, droite, gauche, ...). Chaque cascade d'occultation est créée à partir du détecteur de visage de face. En effet, pour chaque type d'occultation, un ensemble de classifieurs faibles disponibles est défini (par exemple pour les occultations basses, les classifieurs faibles disponibles sont ceux situés sur la partie supérieure du visage) et une McCascade est créée en se basant sur cet ensemble de classifieurs faibles ;
- les différentes cascades sont associées à l'aide du principe de *cascading with evidence* [38].

5.2.3 Création des cascades d'occultations

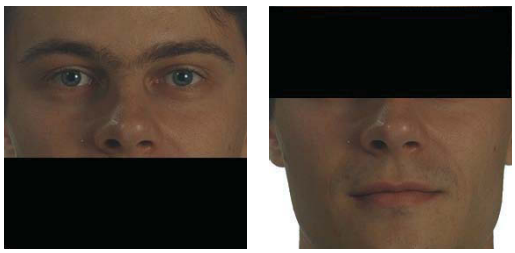
Plusieurs cascades d'occultations sont créées, chacune étant destinée à gérer un type d'occultation particulière. Pour limiter la complexité, le cas de deux types d'occultations à gérer est présenté : basse (occultation de type $\mathcal{A}$) et haute (occultation de type $\mathcal{B}$). Ils sont présentés sur la figure 5.10. Pour l'occultation $\mathcal{A}$, on considère que le tiers inférieur du visage est occulté alors que pour l'occultation $\mathcal{B}$, c'est le tiers supérieur qui est occulté.

Soit $\mathcal{O}_{\mathcal{I}}$ la région occultée avec $\mathcal{I} \in \{\mathcal{A}, \mathcal{B}\}$ l'ensemble des configurations d'occultation. De plus, soit $\mathcal{S}_{jt}$ la région couverte par la sous-fenêtre associée au classifieur faible h_{jt} (voir figure 5.11). Pour chaque type d'occultation $\mathcal{I}$, la construction de la cascade d'occultation associée nécessite la définition de l'ensemble des classifieurs faibles disponibles. Un classifieur faible h_{jt} est disponible pour l'occultation $\mathcal{I}$ si la région $\mathcal{S}_{jt}$ n'intersecte pas $\mathcal{O}_{\mathcal{I}}$. Pour $\mathcal{I} \in \{\mathcal{A}, \mathcal{B}\}$,



(a) (b)

FIGURE 5.9 – Mise en défaut d'un détecteur de visages de face sur un visage occulté. (a) - trois sous-fenêtres sélectionnées pendant l'apprentissage du détecteur de visages de face ainsi que leurs trois classifieurs faibles associés h_1, h_2 et h_3. (b) - les classifieurs faibles h_1 et h_3 en charge de classifier des zones occultées rejettent ces deux zones et le visage risque d'être rejeté par le classifieur fort



(a) (b)

FIGURE 5.10 – Définition de deux types d'occultation (1/3 occultation). (a) - occultation basse (occultation de type $\mathcal{A}$) (b) - occultation haute (occultation de type $\mathcal{B}$)

on définit donc deux ensembles de classifieurs faibles disponibles $\mathcal{H}^{\mathcal{A}}$ et $\mathcal{H}^{\mathcal{B}}$:

$$\mathcal{H}^{\mathcal{A}} = \{h_{jt} | \mathcal{S}_{jt} \cap \mathcal{O}_{\mathcal{A}} = \emptyset\} \tag{5.7}$$

$$\mathcal{H}^{\mathcal{B}} = \{h_{jt} | \mathcal{S}_{jt} \cap \mathcal{O}_{\mathcal{B}} = \emptyset\} \tag{5.8}$$

Connaissant ces deux ensembles, on peut créer deux McCascades $\mathcal{C}^{\mathcal{A}}$ et $\mathcal{C}^{\mathcal{B}}$ qui utilisent les classifieurs faibles de $\mathcal{H}^{\mathcal{A}}$ et $\mathcal{H}^{\mathcal{B}}$. D'après les tests effectués au chapitre précédent, les performances d'une McCascade commençent à diminuer lorsque le taux de classifieurs faibles manquants dépasse 70%. Ainsi, il faut s'assurer que le taux de classifieurs faibles manquants de chaque niveau de $\mathcal{C}^{\mathcal{A}}$ et $\mathcal{C}^{\mathcal{B}}$ ne dépasse pas 70%. Si un niveau présente un taux de classifieurs faibles manquants supérieur à 70%, alors celui-ci est supprimé de la McCascade. Enfin, les seuils β_j de chaque McCascade sont fixés à l'aide de l'algorithme 4 présenté à la section 4.6.

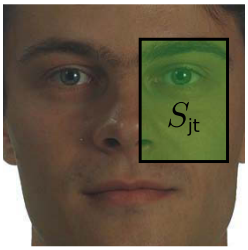


FIGURE 5.11 – Région couverte par une sous-fenêtre associée à un classifieur faible. Le classifieur faible h_{jt} doit classer la région $\mathcal{S}_{jt}$, en vert sur l'image

5.2.4 Cascading With Evidence

Pour combiner la cascade principale et les deux cascades d'occultations, le principe de *cascading with evidence* proposé par Lin et al. [38] est mis en application. Lorsqu'un exemple $\mathbf{x}$ doit être testé, celui-ci est tout d'abord envoyé à la cascade principale $\mathcal{C}$. Au niveau j de cette cascade, en plus d'appliquer le classifieur fort H_j, un vecteur additionnel $\mathbf{o}_j(\mathbf{x})$ est calculé :

$$\mathbf{o}_j(\mathbf{x}) = (H_j^{\mathcal{A}}(\mathbf{x}), H_j^{\mathcal{B}}(\mathbf{x})) \tag{5.9}$$

où

$$H_j^{\mathcal{I}}(\mathbf{x}) = \sum_{t|\mathcal{S}_{jt}\cap\mathcal{O}_{\mathcal{I}}=\emptyset} h_{jt}(\mathbf{x}) \quad \text{avec} \quad \mathcal{I} \in \{\mathcal{A}, \mathcal{B}\} \tag{5.10}$$

L'équation (5.10) signifie que $H_j^{\mathcal{I}}$ implique seulement les classifieurs faibles h_{jt} de $\mathcal{C}$ qui sont en charge de classer une région S_{jt} qui n'intersecte pas avec $\mathcal{O}_{\mathcal{I}}$. Le vecteur $\mathbf{o}_j(\mathbf{x})$ est appelé vecteur d'occultations. On remarque que ce vecteur se déduit facilement du classifieur fort $H_j(\mathbf{x}) = \sum_{t=1}^{T_j} h_{jt}(\mathbf{x})$. Connaissant le vecteur d'occultations de $\mathbf{x}$, les classifieurs faibles peuvent désormais être définis comme disponibles ou pas en fonction de l'occultation rencontrée. En effet, supposons que $\mathbf{x}$ soit un visage occulté de type $\mathcal{A}$ et supposons que la cascade principale le rejette au niveau j car $H_j(\mathbf{x}) < \tau_j$. Avant de le rejeter, son vecteur d'occultations est vérifié. En particulier, la majorité des valeurs $H_1^{\mathcal{A}}(\mathbf{x}), \ldots, H_j^{\mathcal{A}}(\mathbf{x})$ doivent être positives, indicant que $\mathbf{x}$ peut être un visage occulté de type $\mathcal{A}$. Sachant cela, $\mathbf{x}$ est envoyé au niveau j de la McCascade $\mathcal{C}^{\mathcal{A}}$ en charge des occultations de type $\mathcal{A}$. Si $\mathbf{x}$ est réellement un visage occulté de type $\mathcal{A}$, il passe tous les niveaux de $\mathcal{C}^{\mathcal{A}}$. Sinon, il est rejeté par $\mathcal{C}^{\mathcal{A}}$.

En utilisant les trois cascades $\mathcal{C}$, $\mathcal{C}^{\mathcal{A}}$ et $\mathcal{C}^{\mathcal{B}}$ ainsi que le principe de cascading with evidence, la détection de visages occultés est possible. Pour cela, il faut suivre la procédure de test décrite par l'algorithme 6 où $\mathcal{C}^{\mathcal{I}}$ représente la McCascade gérant l'occultation de type $\mathcal{I}$. Cette procédure de test est aussi illustrée sur la figure 5.12.

Les explications données jusqu'ici restent valables pour un plus grand nombre d'occultations gérées. Ce nombre dépend exclusivement des classifieurs faibles appris par le détecteur de visages de face. Par exemple, si tous les classifieurs faibles appris sont associés à des sous-

fenêtres situées sur la partie supérieure du visage, il sera impossible de gérer une occultation de type $\mathcal{B}$ car aucun classifieur faible ne sera situé en bas du visage.

5.3 Méthodologie expérimentale

5.3.1 Les ensembles de tests

Deux bases de test ont été utilisées pour obtenir les résultats de cette section : la base FERET et la base AR. La base FERET a permis d'évaluer la détection de visages tournés et la base AR a permis d'évaluer la détection de visages occultés.

5.3.2 Le détecteur utilisé

Tous les résultats ont été obtenus avec le détecteur de visage de face présenté à la section 3.3.5. Celui-ci comporte 10 niveaux. Chaque niveau est entraîné avec 5000 positifs et 5000 négatifs. Chacun est conçu pour détecter au moins 99,8% des positifs tout en faisant au plus 50% de faux positifs. Pour générer des négatifs pendant l'apprentissage, 2453 images de fond ont été utilisées. Enfin, chaque classifieur faible est appris sur un sous-ensemble de deux caractéristiques. Dans les tests, ce détecteur sera noté $\mathcal{C}$.

5.3.3 Fusion des détections multiples

Lorsqu'on utilise le système multi-vues présenté à la section 5.1.3 ou encore le système de *cascading with evidence* présenté à la section 5.2.4, le processus de fusion des détections multiples doit être adapté. En effet, il faut pouvoir fusionner des détections provenant de cascades différentes. Par exemple, dans le cas du système multi-vues, il peut arriver que deux cascades gérant des angles proches répondent positivement en même temps comme illustré sur la figure 5.13(a). De la même façon, on peut imaginer que plusieurs cascades d'occultations répondent positivement sur un même visage. La fusion des détections se fait alors en deux étapes :

1. les détections provenant d'une même cascade sont fusionnées en suivant la procédure décrite à la section 3.3.4. La figure 5.13(b) présente un exemple de résultat après avoir fusionné les détections de deux cascades différentes ;
2. si plusieurs détections fusionnées se recouvrent, alors on conserve celle avec le score de classification le plus fort. Sur la figure 5.13(c), la détection verte est celle qui présente le plus fort score de classification. Plus formellement, soit d_1 et d_2 deux détections fusionnées. Si $F_{\text{recouvrement}}(d_1, d_2) > 0,5$, alors on conserve la détection d_i telle que :

$$d_i = \underset{d_j \in \{d_1, d_2\}}{\text{argmax}} \left(\text{score}(d_j)\right) \tag{5.11}$$

Algorithme 6 : Détection de visages occultés par association de plusiseurs cascades

Entrées : Un exemple $\mathbf{x}$

Sorties : Classe de $\mathbf{x}$: `Visage, Non-Visage,Visage occulté de type` $\mathcal{I}$

si $\mathbf{x}$ *passe* $\mathcal{C}$ **alors retourner** `Visage`;
si $\mathbf{x}$ *est rejeté au niveau* j *et que* $H_j^{\mathcal{I}}(\mathbf{x}) < 0 \; \forall \, \mathcal{I}$ **alors retourner** `Non-Visage`;
Envoyer $\mathbf{x}$ au niveau j de $\mathcal{C}^{\mathcal{I}}$ si $H_j^{\mathcal{I}}(\mathbf{x}) > \tau_j$ et si la valeur $\sum_{i=1}^{j} H_i^{\mathcal{I}}(\mathbf{x})$ est la plus élevée pour les différentes occultations $\mathcal{I}$;
si $\mathbf{x}$ *passe* $\mathcal{C}^{\mathcal{I}}$ **alors**
    retourner `Visage occulté de type` $\mathcal{I}$;
sinon
    retourner `Non-Visage`;
finsi

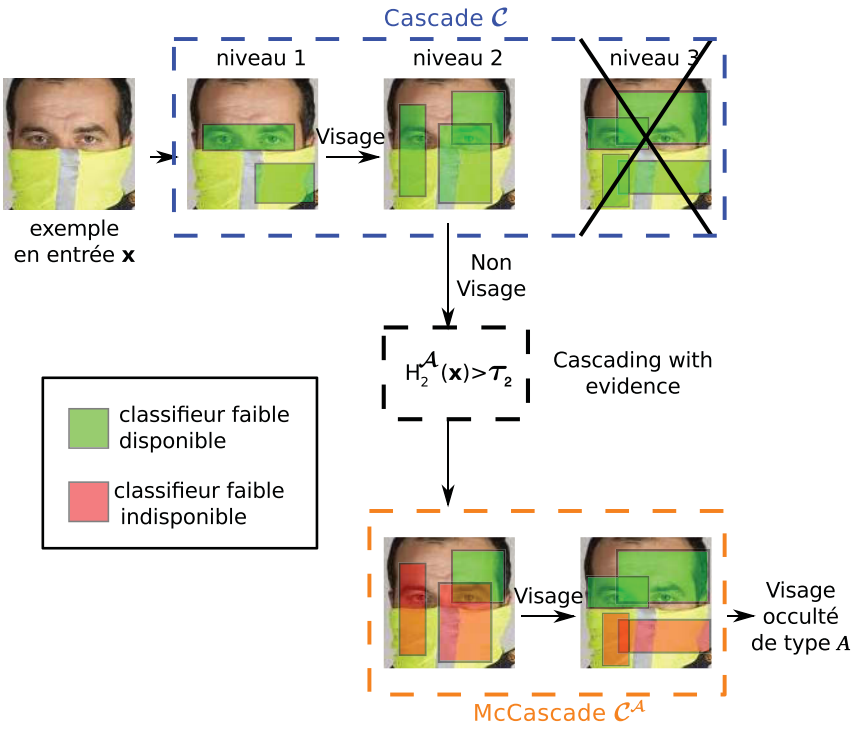
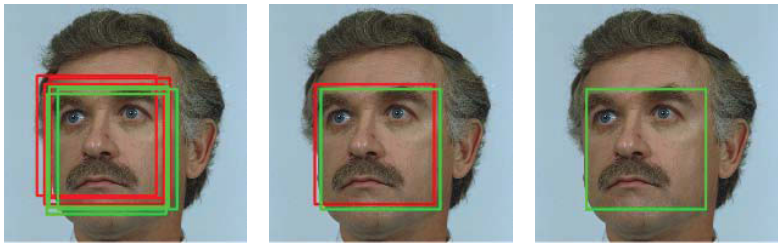


FIGURE 5.12 – Procédure de test de l'association d'une cascade et d'une McCascade par le principe de *cascading with evidence*. L'exemple $\mathbf{x}$ est d'abord traité par la cascade $\mathcal{C}$ et ensuite envoyé à la McCascade $\mathcal{C}^{\mathcal{A}}$ pour finalement être détecté comme un visage occulté de type $\mathcal{A}$



(a) (b) (c)

FIGURE 5.13 – Fusion de détections multiples de deux cascades différentes. (a) - Détections fournies par les deux cascades. (b) - Les détections provenant d'une même cascade sont fusionnées. (c) - Parmi les deux détections fusionnées se chevauchant, on conserve celle avec le score de classification le plus fort

5.4 Résultats du détecteur multi-vues

Cette première partie de résultats est consacrée à l'évaluation de la détection de visages tournés. Tout d'abord, différentes valeurs des paramètres de l'ellipsoïde sont étudiées. Puis, la modification de la position des sous-fenêtres couplée à une McCascade est évaluée. Enfin, les performances du système multi-vues proposé sont analysées.

5.4.1 Les paramètres de l'ellipsoïde

L'ellipsoïde utilisée pour modifier la position des sous-fenêtres est caractérisée par quatre paramètres : a, b, c et w. Le paramètre w correspond à la taille des images d'apprentissage, à savoir 24. Les paramètres a, b et c sont tout d'abord exprimés en fonction de $w/2$ afin de les fixer :

$$a = f_a \times w/2 \tag{5.12}$$
$$b = f_b \times w/2 \tag{5.13}$$
$$c = f_c \times w/2 \tag{5.14}$$

Une recherche exhaustive sur f_a, f_b et f_c a ensuite été réalisée. Les valeurs suivantes ont été testées :

- $1,3 \leq f_a \leq 2,4$ avec un pas de $0,1$, ce qui donne 12 valeurs ;
- $1,5 \leq f_b \leq 2,4$ avec un pas de $0,1$, ce qui donne 10 valeurs ;
- $0,3 \leq f_c \leq 1,4$ avec un pas de $0,1$, ce qui donne 12 valeurs.

On obtient ainsi $12 \times 10 \times 12 = 1440$ configurations différentes. Pour les tester, des images de la base FERET ont été utilisées : 100 images de visages tournés de $22,5°$ et 100 images de visages tournés de $45°$. Chacune est ensuite redimensionnée à 10% de sa taille initiale. Pour chaque ensemble de paramètres (a_i, b_i, c_i), la méthodologie suivante est appliquée :

1. Le détecteur de visage de face est modifié pour obtenir deux détecteurs $\mathcal{C}^{22,5}$ et $\mathcal{C}^{45}$. Pour les obtenir, les positions des sous-fenêtres de $\mathcal{C}$ sont ajustées en utilisant l'ellipsoïde de paramètres (a_i, b_i, c_i). Les sous-fenêtres qui disparaissent sont gérées par la solution naïve présentée à la section 4.3, i.e. que les classifieurs faibles associés à ces sous-fenêtres sont simplement ignorés ;
2. $\mathcal{C}^{22,5}$ est appliqué sur les 100 images de visages tournés de $22,5°$ avec $\Delta_p = 1$ et $fe = 1,1$ afin d'obtenir la courbe ROC associée. Le critère AUC de la courbe est ensuite calculé, noté $auc_i^{22,5}$. AUC signifie *Area Under the ROC curve* que l'on peut traduire par aire sous la courbe ROC. Ce critère consiste à calculer l'aire sous une courbe ROC et permet de comparer plusieurs courbes ROC. Plus ce critère est élevé et plus la courbe ROC associée présente des performances élevées ;
3. De la même façon, $\mathcal{C}^{45}$ est appliqué sur les 100 images de visages tournés de $45°$ et la valeur auc_i^{45} est calculée ;
4. Finalement, la valeur $auc_i = auc_i^{22,5} + auc_i^{45}$ est calculée.

Les paramètres présentant la plus forte valeur auc_i sont ensuite conservés. Les paramètres retenus sont les suivants :

$$a = 2,0 \times w/2 \tag{5.15}$$
$$b = 2,0 \times w/2 \tag{5.16}$$
$$c = 1,0 \times w/2 \tag{5.17}$$

Dans la suite des résultats sur la détection de visages tournés, toutes les ellipsoïdes utilisent les paramètres ci-dessus. Pour information, le tableau 5.0(a) donne les meilleures valeurs f_a, f_b et f_c pour les visages tournés de $22,5°$. Les meilleurs valeurs pour les visages tournés de $45°$ sont regroupés dans le tableau 5.0(b) et le tableau 5.0(c) donne les meilleurs valeurs f_a, f_b et f_c en fonction de la valeur auc_i.

5.4.2 Modification de la position de sous-fenêtres

Dans cette partie, l'utilisation d'une ellipsoïde dans l'ajustement des positions des sous-fenêtres est évaluée. Pour cela, trois classifieurs sont construits à partir de $\mathcal{C}$:

1. $\mathcal{C}^{22,5}$, un détecteur de visages tournés de $22,5°$;
2. $\mathcal{C}^{45}$, un détecteur de visages tournés de $45°$;
3. $\mathcal{C}^{67,5}$, un détecteur de visages tournés de $67,5°$;

Chaque classifieur est obtenu en modifiant les positions des sous-fenêtres de $\mathcal{C}$. La solution naïve est utilisée pour gérer les classifieurs faibles indisponibles associés aux fenêtres qui disparaissent. Ils sont donc simplement ignorés et ceci afin de ne pas biaiser les performances liées à l'utilisation du modèle géométrique retenu. Ces classifieurs sont ensuite appliqués sur les images de visages tournés de la base FERET en enlevant, pour chaque angle, les 100 images

f_a	f_b	f_c	$auc^{22,5}$
2,4	2,1	1,3	1085,91
2,3	2,1	1,3	1085,87
2,1	2,1	1,3	1085,82
2,1	2,3	1,3	1085,78
2,2	2,1	1,3	1085,78
2,4	2,3	1,3	1085,77
2,2	2,2	1,3	1085,77
2,3	2,2	1,3	1085,75
2,2	2,3	1,3	1085,74
2,1	2,2	1,3	1085,74
⋮	⋮	⋮	⋮
2,0	**2,0**	**1,0**	**1080,01**

(a) Visages tournés de $\theta_y = 22,5°$

f_a	f_b	f_c	auc^{45}
1,7	2,0	0,8	1085,1
1,8	2,0	0,8	1084,91
1,4	2,0	0,8	1084,49
1,6	2,0	1,0	1083,24
2,0	**2,0**	**1,0**	**1082,95**
1,3	2,0	0,8	1082,91
1,5	2,0	0,8	1082,79
1,9	2,0	1,0	1082,77
1,8	2,0	1,0	1082,64
1,7	2,0	1,0	1082,51

(b) Visages tournés de $\theta_y = 45°$

f_a	f_b	f_c	auc
2,0	**2,0**	**1,0**	**2162,96**
2.4	2.2	1.1	2162.7
1.9	2.0	1.0	2162.69
1.4	2.0	0.8	2162.59
2.2	2.2	1.1	2162.42
2.1	2.4	1.1	2162.413
2.3	2.2	1.1	2162.406
2.4	2.1	1.1	2162.38
2.2	2.3	1.1	2162.35
2.1	2.3	1.1	2162.19

(c) Cumul des tableaux (a) et (b) : $auc = auc^{22,5} + auc^{45}$

TABLE 5.1 – Meilleurs critères AUC obtenus avec différents paramètres d'ellipsoïde. Le tableau (a) contient les meilleurs résultats obtenus sur des visages tournés de $22,5°$ alors que le tableau (b) contient les meilleurs résultats pour des visages tournés de $45°$. Enfin, le tableau (c) contient le cumul des deux autres tableaux

utilisées pour fixer les paramètres de l'ellipsoïde. Leurs performances sont indiquées dans les figures 5.14(a), 5.14(b) et 5.15. Dans chacune des figures, la cascade $\mathcal{C}$ est notée « Cascade » et le détecteur obtenu en utilisant notre modèle géométrique est noté « MaCascade ». Pour les visages tournés de $22,5°$, l'amélioration obtenue est minime. Ceci s'explique par le fait que l'apparence de ces visages est très proche de celle des visages de face. L'amélioration est bien plus significative dans le cas des visages tournés de $45°$. On note une augmentation du taux de vrais positifs comprise entre 30% et 40%. Enfin, on remarque que la détection des visages tournés de $67,5°$ représente un cas limite de l'utilisation du modèle géométrique. En effet, le taux de vrais positifs est grandement amélioré (jusqu'à 60%) mais au prix d'un nombre très élevé de faux positifs.

5.4.3 Intérêt d'une McCascade

Les trois classifieurs de la section précédente $\mathcal{C}^{22,5}$, $\mathcal{C}^{45}$ et $\mathcal{C}^{67,5}$ comportent des classifieurs faibles manquants :

- $\mathcal{C}^{22,5}$ comporte en moyenne 18% de classifieurs faibles manquants par niveau (taux de manque minimum : 0% et taux de manque maximum : 28%) ;
- $\mathcal{C}^{45}$ comporte en moyenne 27% de classifieurs faibles manquants par niveau (taux de manque minimum : 0% et taux de manque maximum : 41%) ;
- $\mathcal{C}^{67,5}$ comporte en moyenne 44% de classifieurs faibles manquants par niveau (taux de

manque minimum : 0% et taux de manque maximum : 61%).
Au lieu de gérer les classifieurs faibles manquants par la solution naïve comme à la section précédente, il peut être intéressant d'utiliser une McCascade associée à la stratégie P_{knn}. Dans cette section, les trois classifieurs $\mathcal{C}^{22,5}$, $\mathcal{C}^{45}$ et $\mathcal{C}^{67,5}$ sont repris mais cette fois, la structure des classifieurs est changée en une McCascade pour gérer les classifieurs faibles manquants. L'estimation des probabilités a posteriori se fait à l'aide de la stratégie P_{knn} (avec $k = 3$ voisins) et les seuils de la McCascade sont calculés avec la fonction de coût FP_cost. Les classifieurs sont ensuite appliqués sur les images de visages tournés utilisées dans la section précédente. Sur les figures 5.14(a), 5.14(b) et 5.15, ces trois nouveaux classifieurs sont notés « MaMcCascade ». Pour les visages tournés de $22,5°$ et $45°$, l'amélioration obtenue par rapport à l'utilisation de la solution naïve est faible (augmentation du taux de vrais positifs entre 2% et 5%). L'impact de l'utilisation d'une structure en McCascade est plus marquée dans le cas des visages tournés de $67,5°$. En effet, contrairement à l'utilisation de la solution naïve, l'utilisation d'une McCascade permet d'améliorer le taux de vrais positifs tout en conservant un faible nombre de faux positifs. Cependant, les performances restent limitées : on détecte, par exemple, 55% des visages en faisant 12 faux positifs alors que ce taux est de 90% dans le cas des visages tournés de $22,5°$ et $45°$.

5.4.4 Le système multi-vues

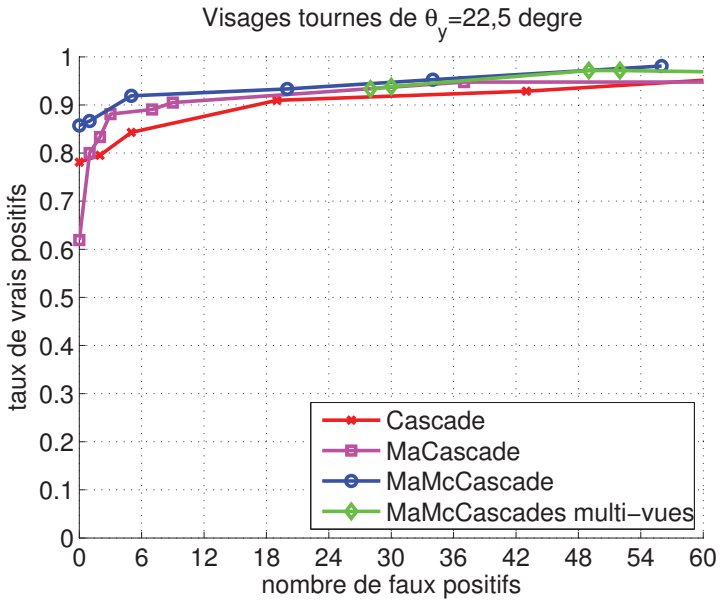
Le système multi-vues proposé est évalué dans cette section. Les trois classifieurs $\mathcal{C}^{22,5}$, $\mathcal{C}^{45}$ et $\mathcal{C}^{67,5}$ sont associés pour former un détecteur multi-vues en suivant le principe décrit à la section 5.1.3. Les classifieurs faibles manquants sont gérés par une structure en McCascade. Le détecteur multi-vues est ensuite appliqué sur les images de visages tournés comme précédement. Sur les figures 5.14(a), 5.14(b) et 5.15, le détecteur multi-vues est noté « MaMcCascade multivues » et obtient des performances similaires aux classifieurs spécifiques à chaque angle (noté « MaMcCascade » sur chaque courbe).

5.5 Résultats du détecteur sur visages occultés

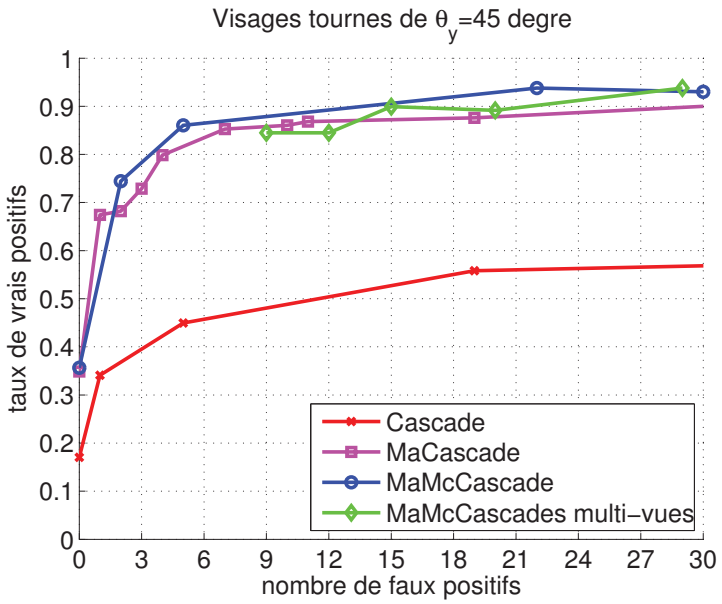
Cette seconde partie de résultats est consacrée à l'évaluation de la détection de visages occultés. Tout d'abord les performances de deux cascades d'occultation sont étudiées. Puis, ces deux cascades ainsi que le détecteur de visages de face sont combinés pour obtenir un système capable de détecter des visages occultés de différentes manières.

5.5.1 Les cascades d'occultations

Dans cette section, les performances de cascades dédiées à la détection de visages présentant un type particulier d'occultation sont évaluées. Dans la section 5.2.3, ces cascades sont nommées des cascades d'occultations. Chacune est créée à partir du détecteur de visage de



(a)



(b)

FIGURE 5.14 – Performance de différents classifieurs sur des visage tournés de $22,5°$ en (a) et de $45°$ en (b). Sur les deux figures, les légendes sont identiques. Le classifieur « Cascade » représente le détecteur de visage de face. Le classifieur « MaCascade » représente le détecteur de visage de face modifié à l'aide du modèle géométrique. Les classifieurs faibles manquants sont gérés par la solution naïve. Le classifieur « MaMcCascade » représente le détecteur de visage de face modifié à l'aide du modèle géométrique. Les classifieurs faibles manquants sont ici gérés par l'utilisation d'une McCascade associée à la stratégie P_{knn}. Enfin, le classifieur « MaMcCascade multi-vues » représente un détecteur multi-vues associant trois détecteurs de type « MaMcCascade » : un détecteur pour l'angle $22,5°$, un pour l'angle $45°$ et un pour l'angle $67,5°$

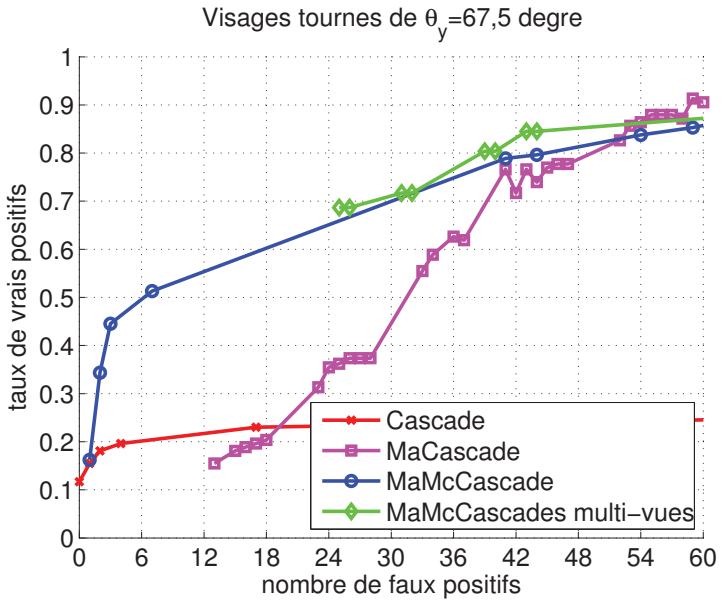


FIGURE 5.15 – Performance de différents classifieurs sur des visage tournés de $67,5°$. Le classifieur « Cascade » représente le détecteur de visage de face. Le classifieur « MaCascade » représente le détecteur de visage de face modifié à l'aide du modèle géométrique. Les classifieurs faibles manquants sont gérés par la solution naïve. Le classifieur « MaMcCascade » représente le détecteur de visage de face modifié à l'aide du modèle géométrique. Les classifieurs faibles manquants sont ici gérés par l'utilisation d'une McCascade associée à la stratégie P_{knn}. Enfin, le classifieur « MaMcCascade multi-vues » représente un détecteur multi-vues associant trois détecteurs de type « MaMcCascade » : un détecteur pour l'angle $22,5°$, un pour l'angle $45°$ et un pour l'angle $67,5°$

face $\mathcal{C}$ en considérant seulement les classifieurs faibles qui ne sont pas occultés. Deux cascades d'occultations sont créées :

1. $\mathcal{C}^{\mathcal{A}}$: cascade d'occultation basse. Celle-ci considère que le tiers inférieur du visage est occulté. Les classifieurs faibles associés à des sous-fenêtres intersectant cette zone sont considérés comme indisponibles. Elle comporte en moyenne 46% de classifieurs faibles indisponibles par niveau (taux d'indisponibilité minimum : 38% et taux d'indisponibilité maximum : 52%) ;

2. $\mathcal{C}^{\mathcal{B}}$: cascade d'occultation haute. Celle-ci considère que le tiers supérieur du visage est occulté. Les classifieurs faibles associés à des sous-fenêtres intersectant cette zone sont considérés comme indisponibles. Elle comporte en moyenne 42% de classifieurs faibles indisponibles par niveau (taux d'indisponibilité minimum : 33% et taux d'indisponibilité maximum : 60%) ;

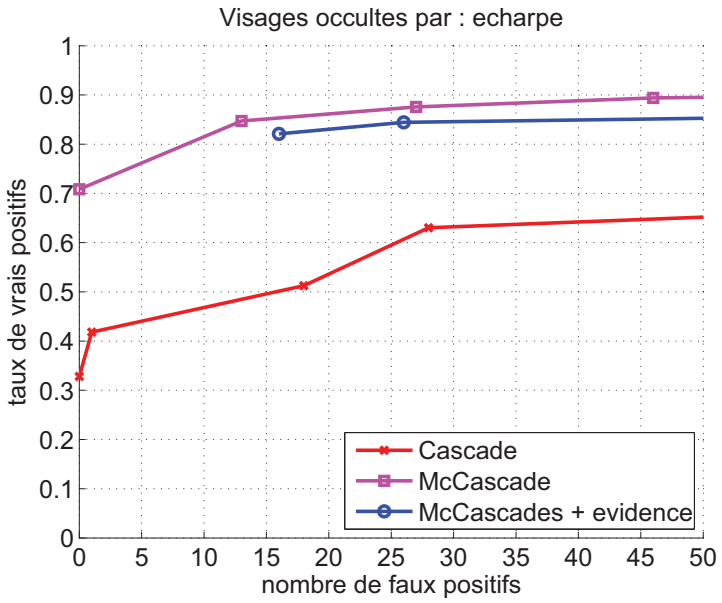
Les classifieurs faibles indisponibles de $\mathcal{C}^{\mathcal{A}}$ et $\mathcal{C}^{\mathcal{B}}$ sont gérés à l'aide de l'utilisation d'une McCascade associée à la stratégie P_{knn} (avec $k = 3$ voisins) et les seuils de la McCascade sont calculés avec la fonction de coût FP_cost. Contrairement au classifieur $\mathcal{C}$ qui comporte 10 niveaux, les cascades $\mathcal{C}^{\mathcal{A}}$ et $\mathcal{C}^{\mathcal{B}}$ ne comportent que 9 niveaux. En effet, le premier niveau de $\mathcal{C}$ ne comporte qu'un seul classifieur faible qui utilise l'ensemble du visage pour classer un exemple (voir figure 3.18(b) située à la page 46) et donc, celui-ci se retrouve indisponible dans $\mathcal{C}^{\mathcal{A}}$ et $\mathcal{C}^{\mathcal{B}}$, ce qui aboutit à un niveau avec 100% de classifieurs faibles manquants. Le premier niveau de $\mathcal{C}^{\mathcal{A}}$ et $\mathcal{C}^{\mathcal{B}}$ correspond donc au second niveau de $\mathcal{C}$. Pour tester les performances de $\mathcal{C}^{\mathcal{A}}$ et $\mathcal{C}^{\mathcal{B}}$, la base AR est utilisée. En particulier, les 765 images de visages occultés par une écharpe et les 765 images de visages occultés par des lunettes de soleil sont isolées. La figure 5.16(a) expose les performances de $\mathcal{C}^{\mathcal{A}}$ sur les visages occultés par des écharpes. La figure 5.16(b) donne les performances de $\mathcal{C}^{\mathcal{B}}$ sur les visages occultés par des lunettes de soleil. Sur ces deux figures, les cascades d'occultations $\mathcal{C}^{\mathcal{A}}$ et $\mathcal{C}^{\mathcal{B}}$ sont notées « McCascade ». Le classifieur noté « Cascade » représente les performances du détecteur de visages de face $\mathcal{C}$ sur les visages occultés. Pour ceux occultés par une écharpe, $\mathcal{C}^{\mathcal{A}}$ obtient de bien meilleures performances que $\mathcal{C}$. En effet, le taux de détection de $\mathcal{C}^{\mathcal{A}}$ est généralement supérieur de 30%. Par contre, les visages occultés par des lunettes de soleil sont moins détectés par $\mathcal{C}^{\mathcal{B}}$ que par $\mathcal{C}$. On note que le taux de détection de $\mathcal{C}^{\mathcal{B}}$ est en moyenne inférieur de 10%. Ces mauvaises performances sont analysées dans une prochaine section.

5.5.2 Association de plusieurs cascades d'occultations

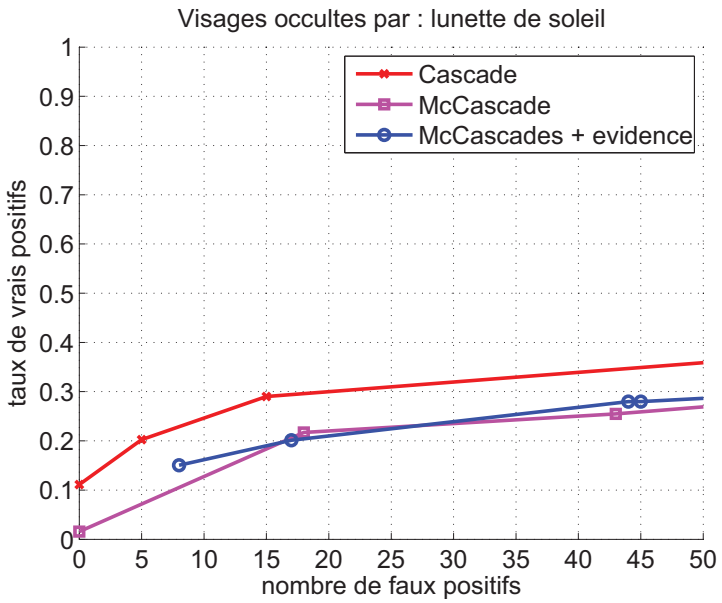
Dans la section précédente, un fort a priori sur les occultations rencontrées est connu. Par exemple, la cascade $\mathcal{C}^{\mathcal{A}}$ est appliquée sur des visages présentant une occultation basse (une écharpe) alors que $\mathcal{C}^{\mathcal{B}}$ est appliquée sur des visages présentant une occultation haute (des lunettes de soleil). Lorsque le type d'occultation à gérer n'est pas connu, il est proposé d'associer un détecteur de visages de face à plusieurs cascades d'occultation à l'aide du principe de *cascading with evidence*. Ici ce principe est testé en créant un détecteur associant $\mathcal{C}$, $\mathcal{C}^{\mathcal{A}}$ et $\mathcal{C}^{\mathcal{B}}$. Ce dernier est donc théoriquement capable de détecter des visages de face mais aussi des visages avec une occultation haute ou basse. Ensuite, ce détecteur est appliqué sur les images de la base AR. Sur les figures 5.16(a) et 5.16(b), ce détecteur est noté « McCascades + evidence ». Sur les visages occultés par une écharpe (figure 5.16(a)), le détecteur obtient des performances légèrement inférieures à celles de la cascade $\mathcal{C}^{\mathcal{A}}$ mais elles restent nettement supérieures à celles de $\mathcal{C}$. Sur les visages occultés par des lunettes de soleil (voir figure 5.16(b)), le détecteur obtient des performances similaires à celles de la cascade $\mathcal{C}^{\mathcal{B}}$ et elles restent inférieures à celles de $\mathcal{C}$.

5.5.3 Analyse des performances en fonction des occultations

À la section 5.5.1, on constate que la cascade $\mathcal{C}^{\mathcal{A}}$, gérant les occultations basses, présentent des résultats supérieurs à ceux la cascade $\mathcal{C}^{\mathcal{B}}$, gérant les occultations hautes. Cette différence de performance est due à une observation : la zone située en bas du visage, qui comprend la



(a)



(b)

FIGURE 5.16 – Performance de différents classifieurs sur des visages occultées par une écharpe en (a) ou des lunettes de soleil en (b). Sur les deux courbes, le détecteur noté « Cascade » représente le détecteur de visages de face. Le détecteur « McCascade » représente une cascade d'occultation dédiée à la détection de visages avec une occultation donnée. Enfin, le détecteur « McCascades + evidence » associe le détecteur de visages de face avec deux cascades d'occultations grâce au principe de *cascading with evidence*

bouche et le nez, est moins discriminante que la zone supérieure du visage. Il en résulte que les classifieurs faibles situés en bas du visage sont généralement moins performants que ceux s'appuyant sur le haut du visage. La cascade $\mathcal{C}^{\mathcal{B}}$ est donc pénalisée : les classifieurs faibles qu'elle utilise sont moins performants que ceux utilisés par $\mathcal{C}^{\mathcal{A}}$.

Afin de vérifier cette différence de performance entre les classifieurs faibles, une carte de performance $\mathcal{M}$ de la cascade $\mathcal{C}$, disponible sur la figure 5.17, est construite. Initialement, cette carte est une image de taille 24×24 dont chaque pixel est initialisé à 0. Ensuite, pour chaque classifieur faible h_{jt} de $\mathcal{C}$, son taux de classification sur les données d'apprentissage, noté TC_{jt}, est calculé et la carte est mise à jour :

$$\mathcal{M}(x, y) = \mathcal{M}(x, y) + TC_{jt} \quad \forall (x, y) \in \mathcal{S}_{jt} \subset \mathcal{M} \tag{5.18}$$

Chaque taux de classification TC_{jt} est obtenu en comptant le nombre de positifs et de négatifs correctement classé divisé par le nombre d'exemples d'apprentissage. Une fois que l'ensemble des classifieurs faibles est pris en compte, les valeurs de $\mathcal{M}$ sont normalisées entre 0 et 1. On obtient finalement la carte de la figure 5.17. Graphiquement, cette carte s'interprète de la façon suivante : plus un pixel est rouge et plus il est couvert par des classifieurs faibles performants. À l'opposé, un pixel bleu foncé signifie qu'aucun classifieur faible ne le couvre. Cette carte illustre bien le décalage de performance qu'il existe entre le haut du visage, où de nombreux pixels sont rouges, et le bas du visage, où aucun pixel rouge n'est présent. On comprend ainsi que la cascade $\mathcal{C}^{\mathcal{B}}$ obtient des performances inférieures à celles de $\mathcal{C}^{\mathcal{A}}$.

5.6 Bilan

Dans ce chapitre, le principe de McCascade est utilisé dans deux applications concrètes :

détection de visages tournés. Les positions des sous-fenêtres du détecteur sont ajustées à l'aide d'un modèle géométrique afin de détecter des visages tournés en utilisant seulement un détecteur de visages de face. Les tests ont tout d'abord permis de fixer les paramètres de ce modèle. Ensuite plusieurs cascades capables de détecter des visages tournés d'un angle θ_y sont créées. Plus θ_y est important et plus le nombre de classifieurs faibles manquants augmente. Les classifieurs faibles manquants sont dus au fait que leurs sous-fenêtres associées disparaissent après ajustement de leurs positions. Des tests ont montré que l'utilisation d'une ellipsoïde pour ajuster la position des sous-fenêtres permet de détecter des visages tournés. Cependant, le système atteint ces limites au délà d'un angle θ_y de $67,5°$. De plus, l'utilisation d'une McCascade pour gérer les classifieurs faibles manquants permet d'augmenter les performances de ces détecteurs. Enfin, un détecteur multi-vues est proposé et testé. Celui-ci combine plusieurs détecteurs de visages tournés, chacun étant spécialisé pour un angle θ_y donné ;

détection de visages occultés. Pour détecter des visages occultés en utilisant seulement un détecteur de visages de face $\mathcal{C}$, des régions d'occultation ont été définies et des cascades

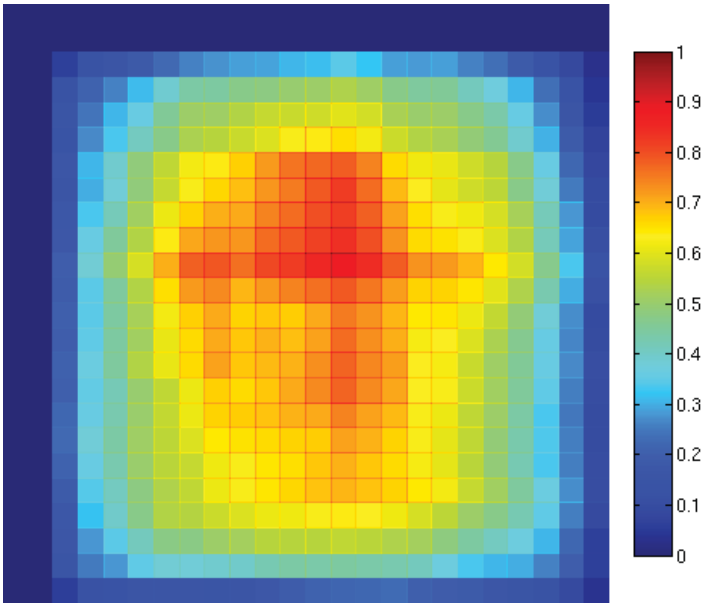


FIGURE 5.17 – Carte de performance des classifieurs faibles du détecteur de visage de face $\mathcal{C}$. Cette carte est construite en cumulant les taux de classification sur les données d'apprentissage de chaque classifieur faible de $\mathcal{C}$. Les valeurs de la carte sont ensuite normalisées entre 0 et 1. Il ressort que le haut du visage est couvert par des classifieurs faibles plus performants que ceux du bas du visage

d'occultations créées. Chaque cascade d'occultation est associée à une région d'occultation et elle utilise seulement les classifieurs faibles de $\mathcal{C}$ qui n'intersectent pas cette région. Les classifieurs faibles intersectant cette région sont considérés comme indisponibles. Ils sont gérés en utilisant une structure en McCascade. Des tests ont tout d'abord montré que l'utilisation de cascades d'occultations permet d'obtenir des meilleures performances que $\mathcal{C}$ sur des visages occultés. Le principe de *cascading with evidence* [38] a ensuite été utilisé pour associer plusieurs cascades d'occultations. Les résultats montrent que cette approche permet d'obtenir de bons résultats lorsque le type d'occultation rencontré n'est pas connu. Cependant, on constate que certaines occultations sont mieux gérées que d'autres. Ce résultat s'explique par les performances des classifieurs faibles qui ne sont pas uniformes sur l'ensemble du visage.

Chapitre 6

Le projet Bio Rafale

Ce chapitre présente le projet dans lequel s'inscrit cette thèse. Le contexte du projet ainsi que ces objectifs sont tout d'abord détaillés. Les différents partenaires, ainsi que leurs rôles respectifs, sont ensuite présentés. Ce chapitre se termine par des tests effectués sur des séquences de mise en situation.

6.1 Contexte

Cette thèse se déroule dans le cadre du projet Bio Rafale qui a été initié par la société clermontoise Vesalis. Ce projet est financé par OSEO et a débuté en décembre 2008 pour une durée de 4 ans. Le nombre de caméras de vidéosurveillance dans les lieux publics continue d'augmenter. En parallèle, le ministère de l'intérieur souhaite un développement de solutions logicielles permettant de traiter l'énorme quantité de données qui résulte de l'ensemble de ces caméras. L'association de l'ensemble des caméras et de solutions logicielles permettrait ainsi à la police judiciaire d'obtenir un outil de vidéo-protection performant. Parmi les solutions logicielles envisagées, on trouve le recours à la reconnaissance faciale. À ce jour, les solutions de reconnaissance faciale sont peu utilisées en France pour des applications civiles de sécurité. Ceci est dû à deux facteurs :

1. La maturité des techniques de reconnaissance faciale est relativement récente et cela reste un domaine très actif de la recherche ;
2. La règlementation française est très stricte quant à l'utilisation de la reconnaissance faciale. En effet, la reconnaissance faciale appliquée à la sécurisation des accès dans les entreprises ou dans les lieux publics est soumise à l'autorisation de la Commission Nationale de l'Informatique et des Libertés (CNIL)

Depuis 2006, dans le cadre du programme Concepts, Systèmes et Outils pour la Sécurité Globale (COSG), l'Agence Nationale de la Recherche a lancé 5 projets autour des technologies de la biométrie liée à l'image :

CANADA : ce projet, lancé en 2006, a pour but de fournir un ensemble d'outils et d'approches pour détecter et gérer, en temps-réel, des comportements pouvant compromettre la sécurité des personnes et des biens à partir de données vidéo ;

VIDEO-ID : ce projet a débuté en 2007 avec pour objectif de développer un système de vidéosurveillance intelligent (détection des situations anormales, détection et suivi de visages dans les vidéos, identification dans une « watch list » via le visage et l'iris, ...) ;

KIVAOU : débuté en 2007 également, le but est ici de développer des outils d'analyse vidéo (valise d'identification et d'indexation biométrique faciale par analyse temps réel vidéo, plateforme d'analyse de vidéos multiples enregistrées lors d'un évènement) ;

SCAR-FACE : le but de ce projet, lancé en 2008, est de réaliser des outils d'aide à la recherche d'individus dans des lieux publics équipés de réseaux de caméras de vidéosurveillance standard, dans un contexte d'élucidation de délits ;

QuIAVU : ce dernier projet, débuté en 2008, a pour but de maîtriser les critères de qualité des images prises par les systèmes de vidéosurveillance, afin de garantir l'analyse a posteriori des images, par l'établissement d'une méthode d'évaluation et la mise au point des métriques associées.

Bio Rafale est également un projet traitant de la biométrie liée à l'image.

6.2 Les objectifs

Le but du projet Bio Rafale est d'améliorer la sécurité dans les stades en s'appuyant sur l'identification des interdits de stade. L'idée est de développer une solution logicielle permettant de traiter les images provenant des caméras qui filment les différentes entrées d'un stade. Cette solution logicielle doit inclure les traitements suivants :

- Détection et suivi des visages ;
- Reconnaissance des interdits de stade en s'appuyant sur une base de données qui contient leurs identités.

L'ensemble des traitements ci-dessus doit se dérouler en temps-réel. De plus, il y a obligation d'utiliser le matériel existant. Il n'est donc pas question de rajouter des caméras. Le contexte du stade amène plusieurs difficultés : problématique de la foule, conditions d'illumination et de prise de vue difficile. De plus, les personnes filmées ne sont pas en situation coopérative. La figure 6.1 présente un exemple d'image d'une caméra filmant une entrée au parc des princes.

6.3 La chaîne de traitement

De nombreux partenaires interviennent au sein du projet Bio Rafale : Vesalis, l'Institut Pascal, le Gipsa-lab, Eurecom, Effidence, SPIE, IBM et l'INT. Ce nombre important de partenaires

FIGURE 6.1 – Exemple d'image provenant d'une caméra du Parc des Princes. Les visages des deux personnes entrantes ont été floutés

est lié aux nombreux défis techniques du projet. Arriver à reconnaître des personnes à la volée dans une foule sur des images de caméras de vidéosurveillance n'est pas chose aisée. Cela implique tout d'abord d'arriver à détecter des visages dans des conditions non contrôlées, c'est-à-dire que les images sont potentiellement de mauvaise qualité (illumination non uniforme, présence de flou) ou encore que la taille des visages à détecter peut varier. Une fois les visages détectés, il faut être capable d'associer un nom à une image de visage. Basiquement, cette association consiste à comparer le visage détecté à des images de visages stockés dans une base de données. Cette étape est la plus complexe du projet Bio Rafale. En effet, de nombreux facteurs, comme la pose, l'illumination, l'expression, la présence d'occultation ou de flou, sont à prendre en compte si l'on souhaite obtenir des performances correctes. La chaîne de traitement des données est constituée de nombreuses étapes :

1. **Détermination du contexte de perception.** Les systèmes développés dans le cadre du projet doivent fonctionner dans des conditions non contrôlées. Avant d'effectuer tout traitement, il est intéressant de récupérer des informations sur l'environnement dans lequel les différents systèmes vont fonctionner. En ce sens, *Effidence* a travaillé sur différents systèmes comme une calibration semi-automatique des caméras qui apporte une connaissance sur la géométrie de la scène. Une méthode qui apprend automatiquement les zones d'entrée des personnes dans la scène filmée a aussi été élaborée. Ils ont également proposé une technique permettant d'estimer la taille des visages à détecter ainsi qu'une autre

fournissant une carte d'occupation des visages dans la scène ;

2. **Détection des visages.** Cette étape permet de localiser les visages dans chaque image. Les travaux développés dans cette thèse au sein de *l'Institut Pascal* couvrent les problématiques de cette étape. En particulier, cette étude se focalise sur la détection de visages tournés et la détection de visages occultés. La majorité des personnes qui rentrent dans un stage présentent un visage de face, ce qui facilite la détection. Mais rien ne les oblige à être de face, il faut donc être capable de détecter des visages qui ne sont pas de face. De plus, les supporters portent souvent des casquettes ou des écharpes aux couleurs de l'équipe supportée. Il en résulte que les visages des supporters peuvent être partiellement occultés. Il faut également être capable de détecter de tels visages ;
3. **Suivi des visages.** Une fois les visages détectés, leurs localisations dans chaque image sont connues. Pour constituer une vidéo de visages pour chaque personne, il faut être capable de lier les différentes détections entre les différentes images : c'est le but du suivi de visages. Une autre approche consiste à utiliser le détecteur de visages pour initialiser les pistes de l'algorithmes de suivi, ce qui permet de limiter le temps de calcul. Des essais ont été menés par *Effidence* ainsi que par *l'Institut Pascal*. De plus, une thèse en rapport avec cette problématique est en cours à *l'Institut Pascal* ;
4. **Sélection des meilleures images.** Chaque vidéo de visage peut contenir plusieurs dizaines d'images alors qu'en pratique, seulement quelques unes (voir une seule) sont nécessaires pour l'étape de reconnaissance. Il faut donc sélectionner les meilleures images pour chaque visage. Pour cela, *Eurecom* a travaillé sur la mise au point d'une mesure de la qualité d'une image. Elle prend en compte différentes caractéristiques comme le flou, le contraste ou encore la taille du visage. De plus, elle permet d'associer un score de qualité compris entre 0 et 100 à chaque image et autorise ainsi la sélection des images de meilleure qualité. *Eurecom* a également travaillé sur des méthodes analysant la présence d'occultation dans les images de visage ;
5. **Normalisation des images.** Les méthodes de reconnaissance faciale sont très sensibles aux conditionnements des données d'entrée. Cela signifie qu'une méthode est généralement prévue pour fonctionner sur des visages d'une taille donnée dans une position définie et avec une illumination fixée. Des méthodes ont donc été développées pour normaliser les images de visage. Le *Gipsa-lab* a notamment proposé une méthode de normalisation de l'illumination et a travaillé sur des techniques d'alignement des images de visage ;
6. **Reconnaissance faciale.** Une fois les images normalisées, on peut envisager la dernière étape de la chaîne de traitement, à savoir la reconnaissance faciale. Les méthodes existantes de reconnaissance faciale proposent de bonnes performances dans des conditions contrôlées. Par contre, dans des conditions non contrôlées, ces mêmes méthodes ne fonctionnent pas aussi bien. Pour remédier à ce constat, le *Gipsa-lab* a élaboré des méthodes de reconnaissance faciale plus performantes. *Eurecom* a également travaillé sur les as-

pects pouvant améliorer les performances des méthodes de reconnaissance faciale qui utilisent uniquement l'apparence.

Les différents partenaires ainsi que leurs principales tâches sont regroupés dans la figure 6.2.

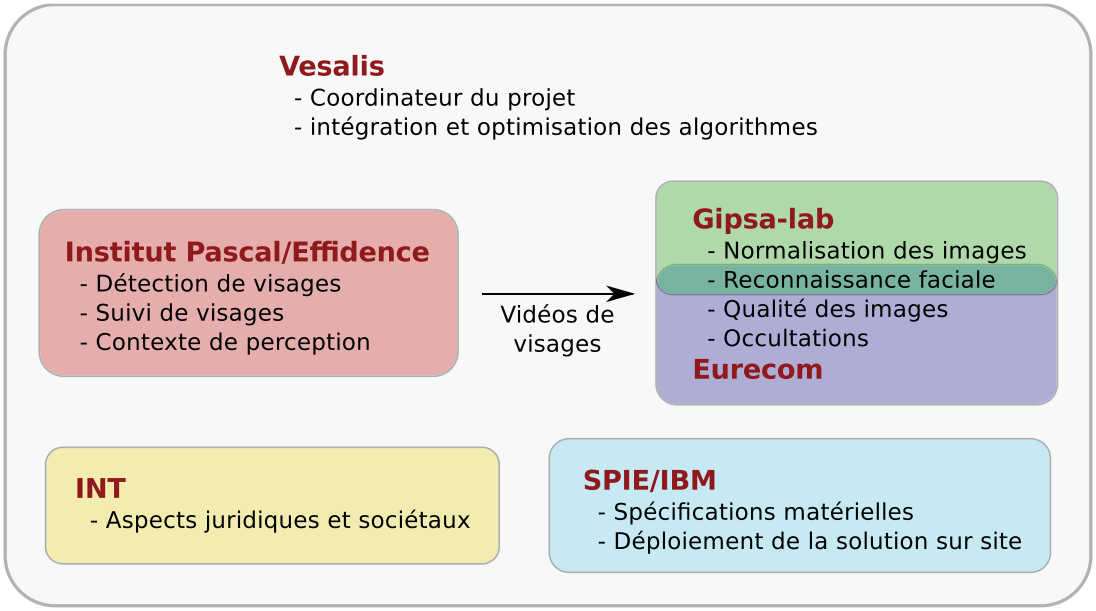


FIGURE 6.2 – Rôles des différents partenaires dans le projet Bio Rafale. Les noms des différents partenaires sont en rouge

6.4 Association avec un tracker

Le détecteur utilisé dans cette thèse a été associé avec un algorithme de suivi développé à l'Insitut Pascal [65]. Cette association a été testée sur une séquence du Parc des Princes. Le détecteur est appliqué sur chaque image de la séquence. Si des détections sont présentes, elles sont envoyées à l'algorithme de suivi pour que celui-ci les intègre aux pistes en cours. La figure 6.3 présente une image de la séquence en cours de traitement. Sur celle-ci, on peut voir que l'algorithme de suivi maintient deux pistes à jour qui correspondent à deux personnes entrantes dans le stade. On remarque que la zone du visage de la piste 13 (en orange) est remplie en rouge, ce qui signifie qu'il a été repéré par le détecteur de visage. À l'opposé, la zone du visage de la piste 12 (en rouge) n'est pas remplie en rouge signifiant que le détecteur n'a pas trouvé ce visage. À chaque piste créée par l'algorithme de suivi est associée une vidéo des visages suivis. La figure 6.4 présente des images qui constituent les vidéos des deux visages. Pour préserver l'anonymat des personnes, les visages ont été floutés .

FIGURE 6.3 – Exemple de suivi de visages. Sur cet exemple, deux visages sont suivis. Un numéro de piste est associé à chaque visage : 12 et 13. Les points colorés représentent les trajectoires des deux visages. Les visages des deux personnes entrantes ont été floutés

6.5 Méthodologie expérimentale

6.5.1 Les acquisitions sur le site PAVIN

Pour tester les différents algorithmes développés pendant le projet, différentes acquisitions ont été réalisées sur le site PAVIN situé à proximité de l'institut Pascal à Clermont-Ferrand. PAVIN signifie Plateforme Auvergnate pour Véhicules INtéligents. Pour réaliser les acquisitions, deux caméras étaient placées sur un mât : la première filmait les personnes de face et la seconde était placée en hauteur sur le mât pour obtenir une vue de dessus des personnes. Dans le cas de cette étude, seules les acquisitions provenant de la seconde caméra ont été utilisées. Elles reproduisent les conditions de prise de vue rencontrées dans un stade. Plusieurs scénarios ont été prévus. Quatre ont été retenus :

1. « Passage un par un » : six personnes différentes passent devant les caméras les unes après les autres en étant de face (voir figures 6.5(a), 6.5(b) et 6.5(c)) ;
2. « Passage deux par deux avec croisements éventuels » : six personnes différentes passent deux par deux devant les caméras. Deux passages sont effectués en modifiant les groupes de deux personnes au deuxième passage. À chaque passage, les personnes sont de face et leurs trajectoires se croisent au milieu de leur passage devant les caméras (voir figures

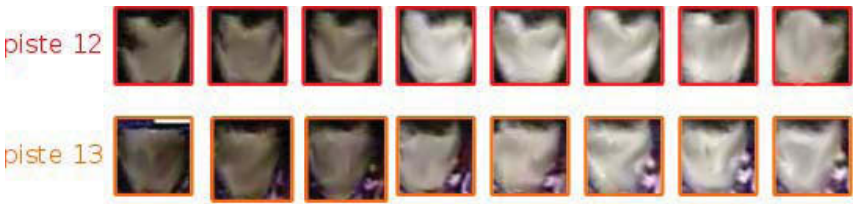


FIGURE 6.4 – Vidéos associées aux visages suivis. Les visages des deux personnes ont été floutés

6.5(d), 6.5(e) et 6.5(f)) ;

3. « Passage groupé » : un groupe de huit personnes passe devant les caméras. Les visages des personnes ne sont pas forcément de face (voir figures 6.5(g), 6.5(h) et 6.5(i)) ;
4. « Passage groupé avec occultations » : un groupe de quinze personnes passe devant les caméras et la plupart d'entre elles ont leur visage occulté. Ce sont des occultations basses (écharpe ou manteau) ou hautes (capuche) (voir figures 6.5(j), 6.5(k) et 6.5(l)).

Chaque séquence a été annotée, i.e. que pour chaque image, la position de chaque visage a été manuellement déterminée.

6.5.2 Les détecteurs utilisés

Le détecteur DV_{cov}

Le détecteur à base de matrices de covariance est le même que celui présenté à la section 3.3.5. Celui-ci comporte 10 niveaux. Chaque niveau est entraîné avec 5000 positifs et 5000 négatifs et chaque niveau est conçu pour détecter au moins 99,8% des positifs tout en faisant au plus 50% de faux positifs. Pour générer des négatifs pendant l'apprentissage, 2453 images de fond ont été utilisées. Enfin, chaque classifieur faible est appris sur un sous-ensemble de deux caractéristiques.

Le détecteur DV_{haar}

Le détecteur DV_{cov} est comparé au détecteur inclus dans la librairie *OpenCV* disponible à l'adresse : `http://sourceforge.net/projects/opencvlibrary/`. Cette implémentation correspond au détecteur de Lienhart et al. [35]. Le modèle *alt tree* [1] est utilisé. Celui-ci est une cascade de 47 niveaux où chacun utilise plusieurs arbres de décision binaires définis sur des descripteurs de type ondelette de Haar. La cascade complète est constituée 8468 arbres de décisions. Aucune information n'est disponible sur l'apprentissage de ce classifieur (base utilisée, nombre d'exemples d'apprentissage, ...).

1. le fichier xml correspondant est `haarcascade_frontalface_alt_tree.xml`

(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

FIGURE 6.5 – Images des acquisitions sur PAVIN. (a) (b) (c) - Images du scénario 1. (d) (e) (f) - Images du scénario 2. (g) (h) (i) - Images du scénario 3. (j) (k) (l) - Images du scénario 4

6.6 Résultats

6.6.1 Comparaison avec OpenCV

Le détecteur DV_{haar} ne renvoie que les détections finales, celles obtenues après avoir fusionné les détections multiples. Le score de classification de chaque détection n'est pas connu. Il est donc difficile de construire une courbe FROC. Pour comparer les deux détecteurs sur une séquence donnée, la méthodologie suivante est utilisée :

1. Le détecteur DV_{haar} est appliqué sur chaque image de la séquence. À l'aide des annotations, le taux de détection pour chaque personne est calculé. Le nombre de faux positifs $nbFP_{haar}$ est aussi déterminé ;
2. Le détecteur DV_{cov} est ensuite appliqué sur chaque image de la séquence. Le taux de détection par personne et le nombre de faux positifs $nbFP_{cov}$ sont calculés. Si $nbFP_{cov}$ est différent de $nbFP_{haar}$, alors le seuil du détecteur DV_{cov} est ajusté pour que $nbFP_{cov}$ soit égal à $nbFP_{haar}$. Cet éventuel ajustement affecte également le taux de détection par personne.

Passage un par un

Les résultats du premier scénario sont disponibles sur la figure 6.6(a). Sur cette figure, le taux de détection est indiqué en ordonnée. Les numéros en abscisse représentent les six personnes différentes présentes dans la vidéo. Le trait rouge correspond au taux de détection moyen du détecteur DV_{haar} et le trait vert à celui du détecteur DV_{cov}. On remarque que :

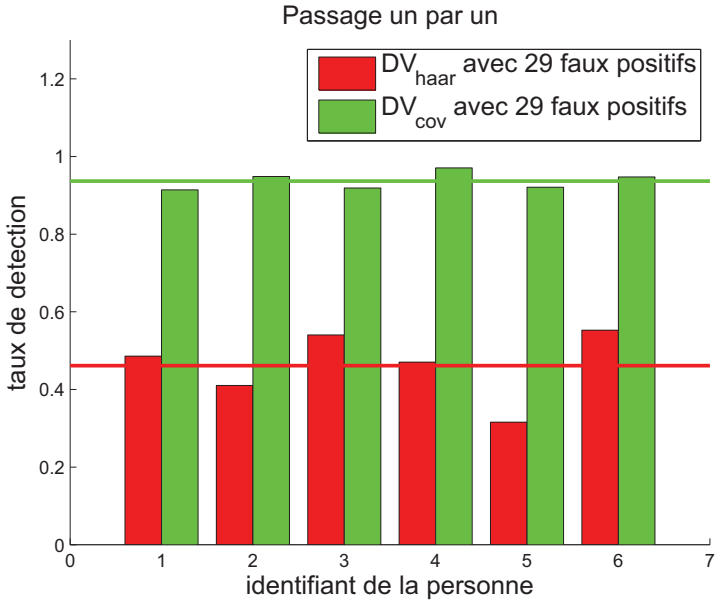
- Le taux de détection moyen du détecteur DV_{cov} est supérieur de plus de 45% à celui du détecteur DV_{haar} ;
- Toutes les personnes sont mieux détectées par le détecteur DV_{cov} ;

Passage deux par deux

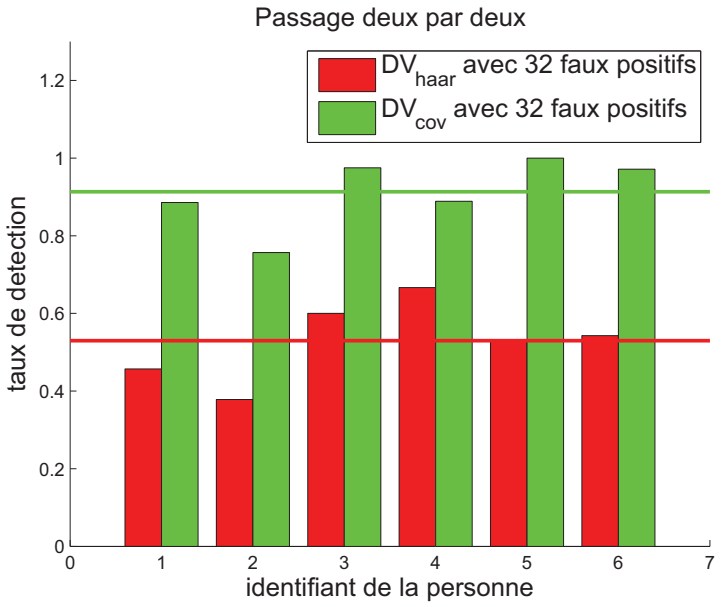
La figure 6.6(b) contient les résultats du second scénario. Par rapport au premier scénario, on note que l'écart entre les deux taux de détection moyen est plus faible (celui du détecteur DV_{cov} reste tout de même supérieur de 40% environ). Toutes les personnes sont encore une fois mieux détectées par le détecteur DV_{cov}.

Passage en groupe

Les résultats du scénario 3 sont donnés sur la figure 6.7. De façon générale, le détecteur DV_{cov} obtient encore de meilleures performances avec un taux de détection moyen supérieur de 40%. On peut cependant noter que la personne 6 n'est pas mieux détectée par le détecteur DV_{cov}. Celui-ci obtient le même taux de détection que le détecteur DV_{haar}. Par contre, toutes les autres personnes sont mieux détectées par le détecteur DV_{cov}.



(a)



(b)

FIGURE 6.6 – Comparaison de DV_{haar} et DV_{cov} sur le scénario 1 en (a) et sur le scénario 2 en (b). Chaque personne présente dans la vidéo est représentée par un numéro en abscisse à laquelle on associe son taux de détection pour chaque détecteur. Le trait vert correspond au taux de détection moyen du détecteur DV_{cov} et le trait rouge au taux de détection moyen de DV_{haar}

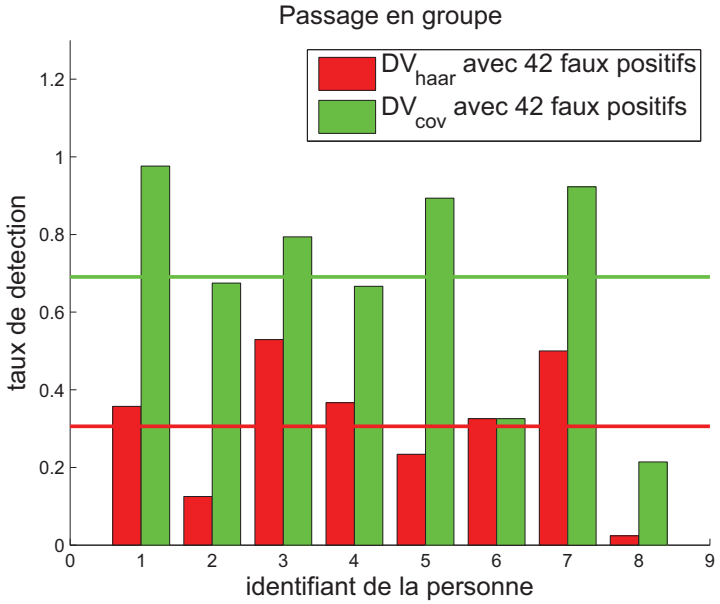


FIGURE 6.7 – Comparaison de DV_{haar} et DV_{cov} sur le scénario 3. Chaque personne présente dans la vidéo est représentée par un numéro en abscisse à laquelle on associe son taux de détection pour chaque détecteur. Le trait vert correspond au taux de détection moyen du détecteur DV_{cov} et le trait rouge au taux de détection moyen de DV_{haar}

Passage en groupe avec occultations

La figure 6.8 donne les résultats obtenus sur le scénario 4. Pour ce scénario, un troisième détecteur est également utilisé. Il s'agit du détecteur présenté à la section 5.5.2. Celui-ci associe le détecteur DV_{cov} avec deux cascades d'occultations. La première gère les occultations hautes et la seconde les occultations basses. Ils sont ensuite associés grâce au principe de *cascading with evidence*. Ce détecteur est noté « DV_{cov} + adaptation » sur la figure. Le terme « adaptation » reflète le fait que le détecteur initial a été adapté pour détecter des visages occultés. Globalement, le détecteur DV_{haar} obtient les moins bonnes performances avec une moyenne de 38% de bonnes détections. Il est suivi par le détecteur DV_{cov} qui obtient une moyenne de 47% de bonnes détections. Enfin, on trouve le détecteur DV_{cov} avec adaptation qui obtient 75% de bonnes détections. De plus, on note que le détecteur DV_{haar} ne détecte pas les personnes 11, 12 et 14. Celles-ci sont détectées par les deux autres détecteurs. Des exemples de détections de ces trois personnes sont fournis sur la figure 6.9. On remarque que la personne 11 (figure 6.9(a)) présente une occultation haute (présence d'une capuche) et que la personne 14 (figure 6.9(c)) présente une occultation basse (écharpe). Quant à la personne 12 (figure 6.9(b)), elle est difficile à détecter car elle présente des composantes structurelles (lunettes et bouc).

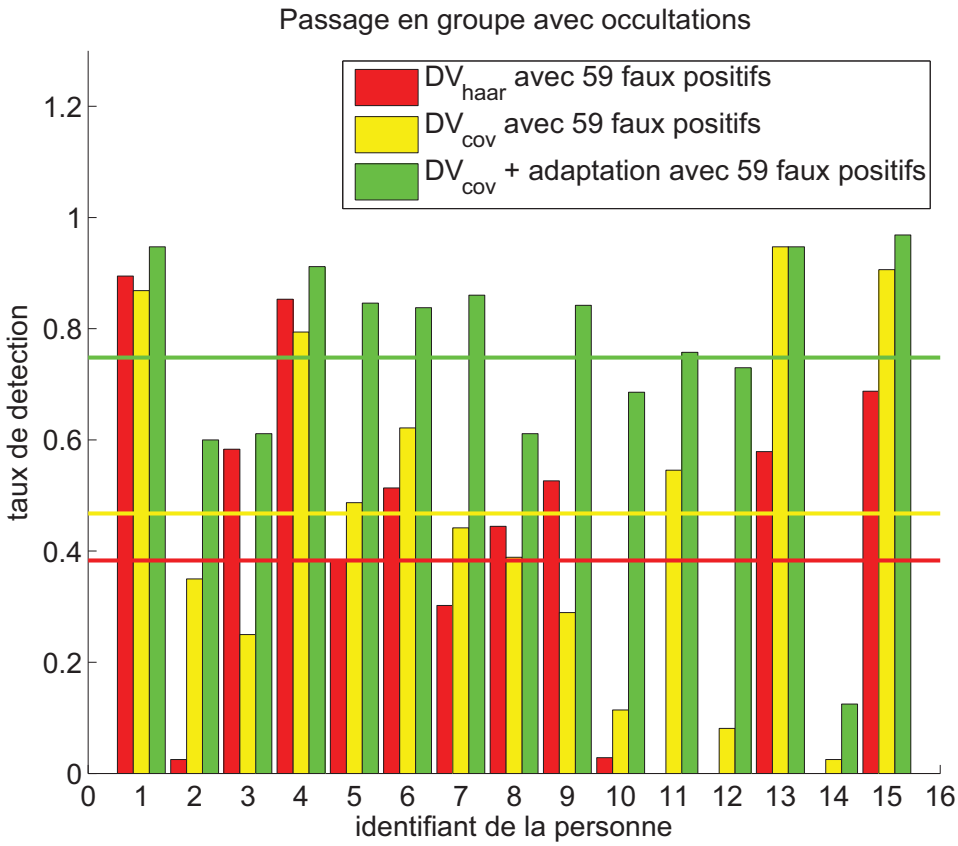


FIGURE 6.8 – Comparaison de DV_{haar}, DV_{cov} et DV_{cov} avec adaptation sur le scénario 4. Chaque personne présente dans la vidéo est représentée par un numéro en abscisse à laquelle on associe son taux de détection pour chaque détecteur. Le trait jaune correspond au taux de détection moyen du détecteur DV_{cov} et le trait rouge au taux de détection moyen de DV_{haar}. Le détecteur « DV_{cov}+adaptation » représente l'association du détecteur DV_{cov} avec deux cascades d'occultation (occultation haute et basse). Le trait vert correspond à son taux de détection moyen

(a) (b) (c)

FIGURE 6.9 – Personnes non détectées par OpenCV. En (a), la personne est occultée par une capuche. En (b), la personne présente des composantes structurelles (lunette, bouc). En (c), la personne est occultée par une écharpe

6.6.2 Vitesse d'exécution et améliorations envisagées

Dans cette partie, des temps d'exécution sont donnés. Ils ont été mesurés sur un processeur Intel Core 2 duo E8500 (3,16 Ghz). Chaque temps donné est en réalité un temps moyen sur dix exécutions. Le code est en C++ et l'exécution n'est pas parallélisée. La structure en cascade ne permet pas de donner un temps d'exécution par image. En effet, dans le cas le plus rapide, toutes les sous-fenêtres sont rejetées par le premier niveau de la cascade. À l'opposé, dans le pire des cas, toutes les sous-fenêtres passent tous les niveaux de la cascade. Le temps nécessaire au scan d'une image est composé du temps de calcul des images intégrales et de la somme des temps de calcul de chaque sous-fenêtre testée.

Le temps moyen de calcul des images intégrales d'une image de taille 360×288 est de $186,5418$ ms. Si on souhaite traiter 25 image/s, on dispose de 40 ms de traitement pour chaque image. On remarque que le calcul des images intégrales doit être optimisé puisque ce calcul dépasse déjà largement les 40 ms.

Une fois les images intégrales calculées, on peut tester différentes sous-fenêtres. Le tableau 6.1 donne différentes indications sur le temps de traitement d'une sous-fenêtre en fonction du niveau de la cascade. En plus du temps de traitement sur chaque niveau, le temps cumulé est également donné. Il est obtenu en faisant la somme des temps du niveau 1 au niveau courant. Si on note k le niveau courant, alors le temps de traitement de ce niveau est noté t_k et le temps cumulé du niveau 1 à k est noté $t_{1:k}$. Logiquement, plus le nombre de classifieurs faibles augmente et plus le temps de traitement augmente. On constate aussi l'écart qu'il y a entre le temps de traitement du niveau 1 : $0,0517$ ms, et le temps du dernier niveau : $1,7525$ ms. Le premier niveau est ainsi 34 fois plus rapide que le dernier niveau. Au mieux, il faut $0,0517$ ms pour classer une sous-fenêtre et au pire, il faut $9,8376$ ms. (ce qui est 190 fois plus lent que le cas le plus rapide).

Connaissant le taux de rejet théorique par niveau f_{max}, il est possible de donner un temps moyen $\bar{t}$ pour classer x sous-fenêtres :

$$\bar{t} = \sum_{k=1}^{10} t_k (f_{\text{max}})^{k-1} x \tag{6.1}$$

où f_{max} est le taux maximum de faux positifs de chaque niveau. Dans le cas du détecteur DV_{cov}, $f_{\text{max}} = 0.5$. Il est plus intéressant de connaître le nombre moyen de fenêtres x pouvant être traité en temps-réel (25 image/s). Il est obtenu en résolvant l'équation :

$$\sum_{k=1}^{10} t_k (f_{\text{max}})^{k-1} x = 40 \tag{6.2}$$

ce qui donne :

$$x = \frac{40}{\sum_{k=1}^{10} t_k (f_{\text{max}})^{k-1}} \tag{6.3}$$

niveau k	nombre de classifieurs faibles	temps t_k (ms)	temps cumulé $t_{1:k}$ (ms)
1	1	$0,0517$	$0,0517$
2	5	$0,088$	$0,1398$
3	12	$0,1747$	$0,3144$
4	25	$0,3735$	$0,6879$
5	54	$0,7194$	$1,4073$
6	124	$1,7287$	$3,1360$
7	120	$1,6296$	$4,7656$
8	105	$1,4648$	$6,2304$
9	134	$1,8547$	$8,0851$
10	131	$1,7525$	$9,8376$

TABLE 6.1 – Temps de traitement d'une sous-fenêtre en fonction du niveau de la cascade. Pour chaque niveau k, le temps de traitement cumulé est également donné. Il correspond à la somme des temps de traitement des niveaux 1 à k

Dans ce cas, on obtient $x = 240$.

On peut également donner le nombre minimal $x_{\min}$ et le nombre maximal $x_{\max}$ de sous-fenêtres pouvant être traité en temps réel :

$$x_{\min} = \frac{40}{t_{1:10}} = 4 \tag{6.4}$$

$$x_{\max} = \frac{40}{t_1} = 773 \tag{6.5}$$

En plus de ces temps d'exécution, on peut signaler que le temps d'apprentissage nécessaire pour obtenir le détecteur DV_{cov} n'est pas négligeable. En effet, deux semaines ont été nécessaires et le code d'apprentissage, en C++, était parallélisé sur quatre CPU.

Quelques pistes pour améliorer le temps de détection existent :

- Le nombre de sous-fenêtres à tester peut être grandement réduit en utilisant les outils développés par Effidence. En connaissant les tailles potentielles des visages, on peut réduire le nombre d'échelles parcourues dans l'image. De plus, en connaissant la zone de présence des visages, on peut limiter les recherches à cette zone ;
- Le détecteur n'a pas besoin d'être appliqué sur chaque image. En effet, une fois qu'un visage est détecté, l'algorithme de suivi peut être utilisé. Sur les 25 images par seconde, on peut imaginer n'appliquer le détecteur que toutes les 5 ou 10 images par exemple ;
- Il est aussi possible d'intervenir sur la phase d'apprentissage. Des techniques de cascade imbriquée [23] ou de descripteurs accumulés [84] permettent de réduire le nombre de classifieurs faibles et de diminuer le temps de détection pour chaque sous-fenêtre.

6.7 Bilan

La détection des visages dans la chaîne de traitement est l'étape nécessitant le plus de temps de calcul. Les tests ont montré les bonnes performances du détecteur proposé. Elles sont en particulier bien meilleures que celles de la librairie OpenCV. Il reste cependant à améliorer le temps d'exécution. Pour cela, différentes pistes ont été évoquées.

Chapitre 7

Conclusion et perspectives

7.1 Conclusion

Les travaux développés dans cette thèse étudient la problématique de la détection d'objets par des cascades de boosting lorsqu'un sous-ensemble des observations n'est pas disponible. Les applications explorées sont la détection de visages tournés et la détection de visages occultés par adaptation d'un détecteur de visages de face. Elles ont nécessité de mettre en place des techniques de gestion de classifieurs faibles manquants au sein d'une cascade de classifieurs boostées. Les techniques développées s'appuient sur une formulation probabiliste d'une cascade de classifieurs. Cette formulation, nommée McCascade, prend en compte l'incertitude introduite par l'abscence de certains classifieurs faibles. En particulier, la décision prise au niveau j dépend également des décisions des niveaux précédents. Ce n'est pas le cas dans une cascade classique. Cette nouvelle formulation nécessite tout d'abord d'estimer des probabilités a posteriori. Trois stratégies d'estimation ont été proposées. De plus, de nouveaux seuils de décision apparaissent. Nous proposons de les fixer à l'aide d'une procédure itérative. Au sein de celle-ci, chaque seuil est déterminé en minimisant une fonction de coût définie sur les performances à atteindre. De nombreux tests ont permis de déterminer la configuration optimale d'une McCascade. En particulier, la meilleure stratégie d'estimation des probabilités a posteriori utilise l'algorithme des k-plus proches voisins. Cette stratégie permet d'obtenir des performances proches de celles de la cascade initiale jusqu'à 60% de classifieurs faibles manquants par niveau. Au-delà, les performances diminuent.

Les techniques développées ont ensuite été appliquées à la détection de visages tournés en utilisant uniquement un détecteur de visages de face. Un modèle 3D de visages simple est tout d'abord utilisé pour adapter le détecteur de visages de face. En effet, les positions des sous-fenêtres associées à chaque classifieur faible sont ajustées pour prendre en compte la modification d'apparence. Cet ajustement des positions entraîne la disparition de certaines sous-fenêtres. Les classifieurs faibles associés deviennent donc indisponibles. Ils sont gérés par une McCas-

cade. Les tests effectués ont permis de fixer les paramètres du modèle 3D. Ils ont également montré que l'approche proposée obtient de bonnes performances. Le système atteint ces limites pour des visages tournés de $\pm 67,5°$.

La détection de visages occultés en utilisant uniquement un détecteur de visages de face a également été abordée. Des zones d'occultations sont tout d'abord définies. À chaque zone est associée une cascade d'occultation. Cette dernière est une McCascade qui utilise uniquement les classifieurs faibles non occultés. Plusieurs cascades d'occultations peuvent ensuite être combinées pour obtenir un système capable de gérer différents types d'occultation. Les tests ont montré que les performances des cascades d'occultation dépendent fortement des classifieurs faibles utilisés. Dans le cas des visages, les classifieurs faibles situés en haut du visage sont généralement plus performants que ceux situés en bas du visage.

Enfin, des comparaisons avec la librairie OpenCV ont été faites. Celles-ci ont montré que la détection de visages par matrices de covariance surpasse la détection par ondelettes de Haar. De plus, des tests ont montré que la détection de visages occultés peut être grandement améliorée par l'association de plusieurs cascades d'occultations. Dans le cadre du projet Bio Rafale, il reste désormais à améliorer le temps d'exécution du détecteur à base de matrices de covariance. Dans leur article, Yao et Odobez [86] rapportent des temps d'exécution de 5 à 20 images par seconde sur des images de taille 384×288.

7.2 Perspectives

Les perspectives présentées concernent différentes parties de cette thèse. Des perspectives pour la gestion des classifieurs faibles manquants sont tout d'abord abordées. Puis, des idées pour la détection des visages tournés sont présentées. Enfin, des pistes pour la détection de visages occultés sont évoquées.

Gestion des classifieurs faibles manquants

La construction d'une McCascade nécessite d'estimer des probabilités a posteriori et de fixer des seuils β_j. Ces deux points ont été abordés dans ces travaux. Cependant, d'autres stratégies d'estimation des probabilités a posteriori peuvent être envisagées. Par exemple, on peut estimer les classifieurs faibles indisponibles $\{h_{\mathrm{i}_1}, \ldots, h_{\mathrm{i}_q}\}$ à l'aide des classifieurs faibles disponibles $\{h_{\mathrm{d}_1}, \ldots, h_{\mathrm{d}_p}\}$. Soit $\mathbf{x}$ un exemple à classifier. On calcule tout d'abord les scores

$\{h_{\mathrm{d}_1}(\mathbf{x}), \ldots, h_{\mathrm{d}_p}(\mathbf{x})\}$ et on estime les scores $\{h_{\mathrm{i}_1}(\mathbf{x}), \ldots, h_{\mathrm{i}_q}(\mathbf{x})\}$ par regression linéaire :

$$\hat{h}_{\mathrm{i}_1}(\mathbf{x}) = \sum_{i=1}^{p} a_{1,i} h_{\mathrm{d}_i}(\mathbf{x}) + b_1 \tag{7.1}$$

$$\vdots$$

$$\hat{h}_{\mathrm{i}_q}(\mathbf{x}) = \sum_{i=1}^{p} a_{q,i} h_{\mathrm{d}_i}(\mathbf{x}) + b_q \tag{7.2}$$

Ensuite, on peut définir une nouvelle stratégie $P_{\mathrm{lin}}(y = 1|\mathbf{x})$ par :

$$P_{\mathrm{lin}}(y = 1|\mathbf{x}) \doteq \frac{1}{1 + \mathrm{e}^{-(H_{\mathrm{d}}(\mathbf{x}) + H_{\mathrm{lin}}(\mathbf{x}))}} \tag{7.3}$$

où $H_{\mathrm{d}}(\mathbf{x}) = \sum_{t=1}^{p} h_{\mathrm{d}_t}(\mathbf{x})$ et $H_{\mathrm{lin}}(\mathbf{x}) = \sum_{i=1}^{q} \hat{h}_{\mathrm{i}_i}(\mathbf{x})$. Les régressions linéaires sont estimées à l'aide d'un ensemble d'exemples positifs et négatifs. La figure 7.1 illustre ce processus pour deux classifieurs faibles h_1 et h_2. Dans cet exemple, on suppose que h_2 est indisponible lors de la classification et on souhaite l'estimer à l'aide de h_1 qui est disponible.

Dans cette thèse, les seuils β_j sont calculés à l'aide d'une procédure itérative. L'inconvénient de cette procédure est qu'elle réalise une optimisation locale. En effet, les seuils sont calculés niveau par niveau. Une autre approche serait de fixer les seuils de façon globale. La valeur de β_j dépendrait alors des valeurs des autres seuils $\{\beta_1, \ldots, \beta_{j-1}, \beta_{j+1}, \ldots, \beta_K\}$. Dans ses travaux, Luo [40] propose deux méthodes (une locale et une globale) pour calculer les seuils d'une cascade. La méthode globale amène de meilleures performances.

Détection des visages tournés

Dans cette thèse, la détection de visages tournés s'appuie sur un ajustement des positions des sous-fenêtres. En particulier, des heuristiques sont utilisées pour déterminer les sous-fenêtres après rotation (voir figure 5.5 page 79, figure 5.6 page 79 et figure 5.7 page 80). Des tests approfondis doivent être menés sur ces heuristiques pour quantifier leur impact sur les performances obtenues. De plus, des modifications peuvent être envisagées. Par exemple, lorsque deux coins sont visibles après rotation d'une sous-fenêtre, celle-ci est conservée dans nos travaux. Il serait intéressant de tester des contraintes sur la surface de la sous-fenêtre après rotation. Par exemple, soit S la surface de la sous-fenêtre avant rotation et S_{rot} la surface après rotation. Si $S_{\mathrm{rot}} < \gamma S$, alors la sous-fenêtre n'est pas conservée où $\gamma \in [0, 1]$. Une telle contrainte traduit le fait que si une sous-fenêtre est beaucoup modifiée après rotation, alors elle n'est pas conservée.

Détection des visages occultés

Dans cette étude, la détection de visages occultés repose sur l'association de plusieurs McCascade. L'utilisation de McCascades présente certaines limitations dans la gestion des occul-

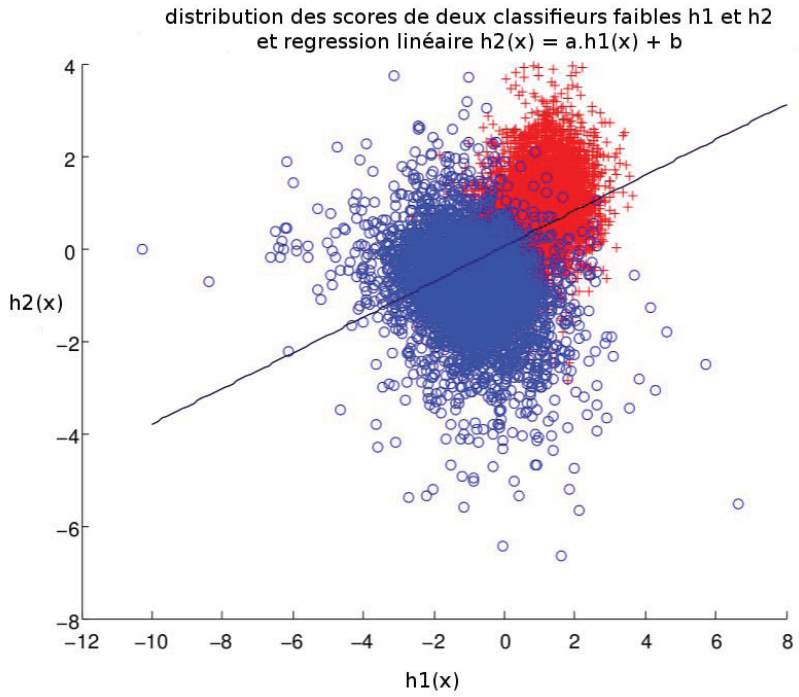


FIGURE 7.1 – Distribution des scores de deux classifieurs faibles h_1 et h_2. Les ronds bleus représentent des exemples négatifs et les croix rouges des exemples positifs. Une regression linéaire $h_2(\mathbf{x}) = ah_1(\mathbf{x}) + b$ peut être estimée. Si h_2 est indisponible lors de la classification, on peut alors l'estimer à l'aide de h_1

tations. En effet, les occultations pouvant être gérées dépendent exclusivement de la répartition spatiales de sous-fenêtres apprises lors de l'apprentissage initiale. En pratique, on remarque que les sous-fenêtres apprises sont généralement situées dans la partie supérieure du visage (autour des yeux). Ainsi, il est difficile, voir impossible, de gérer les occultations hautes car les sous-fenêtres restantes (celles situées en bas du visage) sont trop peu nombreuses et trop peu discriminantes. Un framework d'apprentissage simple est alors proposé pour obtenir un classifieur robuste aux occultations. L'idée est de partitionnner le visage en zones disjointes et d'imposer des contraintes de performances sur les différentes zones. Le but recherché est une répartition plus homogène des performances des classifieurs faibles sélectionnés pendant l'apprentissage sur l'ensemble du visage. Des tests préliminaires ont été effectués. Deux zones $\mathcal{Z}_1$ et $\mathcal{Z}_2$ sont définies : la moitié inférieure du visage et la moitié supérieure (voir figure 7.2). Les classifieurs faibles sont ensuite sélectionnés dans $\mathcal{Z}_1$ ou $\mathcal{Z}_2$. La contrainte suivante est ensuite appliquée : les classifieurs faibles dans la zone $\mathcal{Z}_i$ doivent amener un taux de détection d'au moins d_{min} et un taux de faux positifs d'au plus f_{max}. Ainsi, des classifieurs faibles sont ajoutés dans chaque zone tant que la contrainte n'est pas vérifiée.

Un apprentissage a été réalisé en prenant $d_{\text{min}} = 0,998$ et $f_{\text{max}} = 0,2$. Le classifieur obtenu comporte 12 niveaux. Chaque niveau est entraîné avec 5000 positifs et 5000 négatifs. Pour générer des négatifs pendant l'apprentissage, 2453 images de fond ont été utilisées. Enfin, chaque classifieur faible est appris sur un sous-ensemble de deux caractéristiques. Dans la suite, ce classifieur est noté $\mathcal{C}_{\text{framework}}$. Il est ensuite appliqué sur les images de visages occultés de la base AR. La figure 7.3(a) présente les résultats sur les visages occultés par une écharpe. Sur la figure 7.3(b), les visages sont occultés par des lunettes de soleil. Enfin, les visages sont occultés par une écharpe ou des lunettes de soleil sur la figure 7.3(c). Sur chaque figure, on trouve les performances des classifieurs suivants :

- « Cascade » : correspond au classifieur présenté à la section 3.3.5 page 37. Il a été entraîné sans le framework proposé ;
- « Cascade + framework » : correspond au classifieur $\mathcal{C}_{\text{framework}}$;
- « McCascade + framework » : correspond à une cascade d'occultation créée à partir de $\mathcal{C}_{\text{framework}}$. Dans le cas des écharpes, celle-ci gère les occultations basses. Dans le cas des lunettes de soleil, il s'agit des occultations hautes ;
- « McCascades + evidence + framework » : association des deux cascades d'occultations par le principe de *cascading with evidence*.

Sur les trois figures, on note que le détecteur $\mathcal{C}_{\text{framework}}$ présente de très bonnes performances, démontrant sa robustesse aux occultations hautes et basses. Le détecteur « McCascades + evidence + framework » obtient des performances similaires et ne parvient pas à améliorer les résultats. Dans le cas des cascades d'occultations, les performances sont améliorées pour les occultations basses (écharpe). Elles sont par contre dégradées pour les occultations hautes (lunette de soleil).

Ces résultats préliminaires montrent qu'il est possible de modifier légèrement l'apprentissage du détecteur de visages de face pour que celui-ci s'adapte plus facilement à des conditions critiques (ici, des occultations). Cette piste reste à explorer. Notamment, comment généraliser ce principe de zones ? Quels sont les impacts sur des visages tournés ?

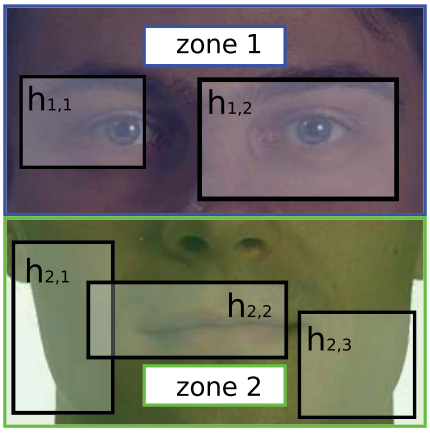


FIGURE 7.2 – Framework d'apprentissage robuste aux occultations. Deux zones $\mathcal{Z}_1$ ou $\mathcal{Z}_2$ sont définies. Les classifieurs faibles sont sélectionnés dans ces zones. Au total, cinq classifieurs faibles ont été sélectionnés. Les performances des classifieurs faibles $\{h_{1,1}, h_{1,2}\}$ sont équivalentes à celles des classifieurs $\{h_{2,1}, h_{2,2}, h_{2,3}\}$

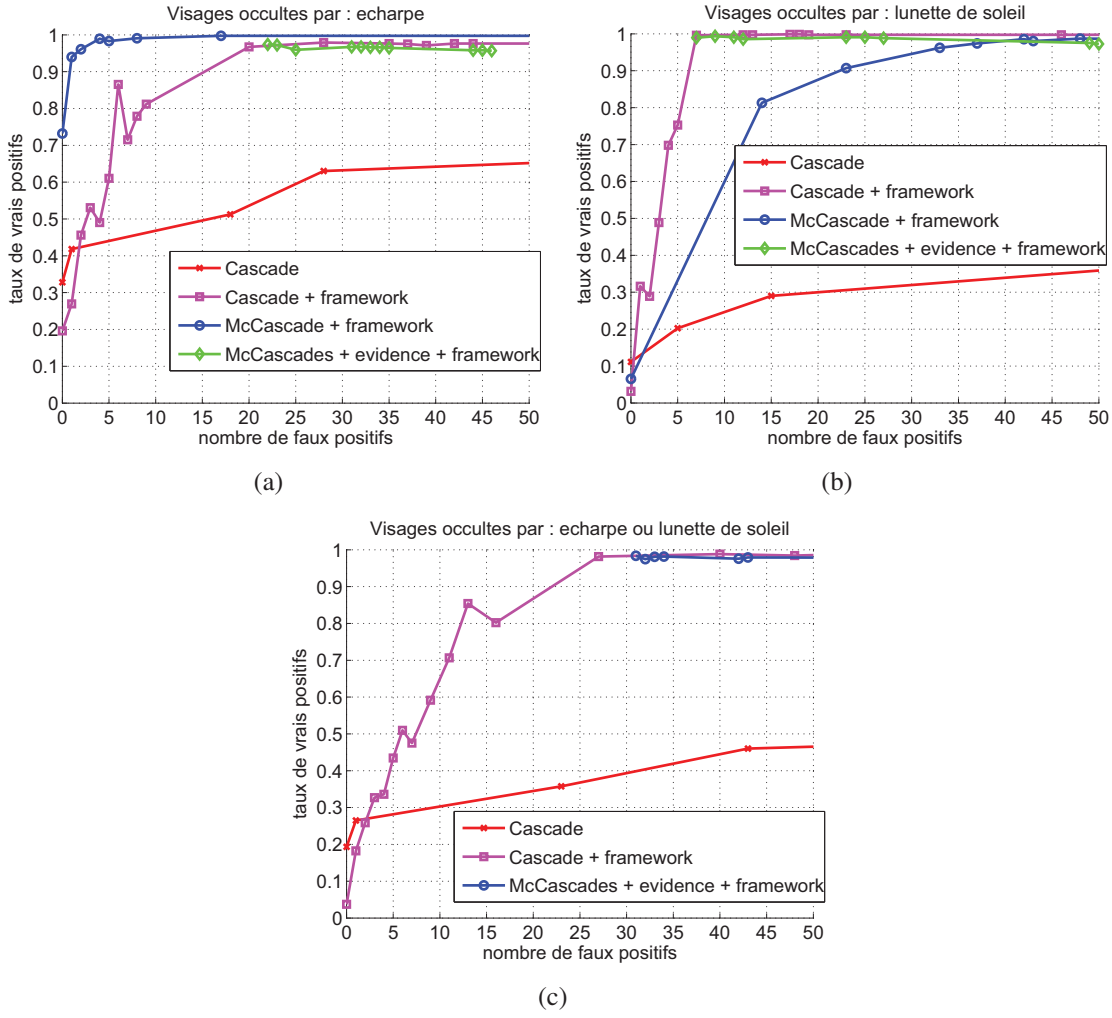


FIGURE 7.3 – Évaluation du framework d'apprentissage sur des visages occultées par une écharpe ou des lunettes de soleil. Le classifieur « Cascade » est un détecteur de visages de face entraîné sans le framework. Le classifieur « Cascade + framework » est un détecteur de visages de face entraîné avec le framework. Le classifieur « McCascade + framework » est une cascade d'occultation créée à partir du détecteur « Cascade + framework ». Elle gère les occultations basses en (a) et les occultations hautes en (b). Enfin, le détecteur « McCascades + evidence + framework » associe le détecteur « Cascade + framework » et les deux cascades d'occultation à l'aide du principe de *cascading with evidence*

Annexe A

Images intégrales pour les matrices de covariances

Le concept d'image intégrale appliqué aux matrices de covariance est ici présenté. Celui-ci a été proposé par Tuzel et al. [69]. Pour une image en niveau de gris I, Tuzel et al. définissent l'image intégrale I_{int} de I par :

$$I_{\text{int}}(x,y) = \sum_{x'<x, y'<y} I(x',y') \tag{A.1}$$

On remarque une différence avec la définition donnée par Viola et Jones [74] dans laquelle les pixels d'abscisse x et d'ordonnée y sont inclus dans le calcul (voir équation (2.3)). En reprenant la définition de la matrice de covariance donnée à la section 3.1.1, on a :

$$C_R = \frac{1}{N-1}\sum_{k=1}^{N}(\mathbf{z}_k - \boldsymbol{\mu})(\mathbf{z}_k - \boldsymbol{\mu})^{\mathrm{T}} \tag{A.2}$$

où R est une région de l'image des caractéristiques C. L'élément (i,j) de cette matrice s'écrit :

$$C_R(i,j) = \frac{1}{N-1}\sum_{k=1}^{N}(\mathbf{z}_k(i) - \boldsymbol{\mu}(i))(\mathbf{z}_k(j) - \boldsymbol{\mu}(j)) \tag{A.3}$$

En développant la moyenne $\boldsymbol{\mu}$ et en réarrangeant les termes, on obtient :

$$C_R(i,j) = \frac{1}{N-1}\Big[\sum_{k=1}^{N}\mathbf{z}_k(i)\mathbf{z}_k(j) - \frac{1}{N}\sum_{k=1}^{N}\mathbf{z}_k(i)\sum_{k=1}^{N}\mathbf{z}_k(j)\Big] \tag{A.4}$$

Calculer la matrice C_R nécessite de calculer les sommes sur chaque caractéristique $\mathbf{z}_k(i)_{i=1,\ldots,d}$ ainsi que les sommes sur chaque multiplication de couple de caractéristiques $\mathbf{z}_k(i)\mathbf{z}_k(j)_{i,j=1,\ldots,d}$.

On construit ainsi d images intégrales pour chaque caractéristique $\mathbf{z}_k(i)$ et d^2 images intégrales pour chaque couple de caractéristiques $\mathbf{z}_k(i)\mathbf{z}_k(j)$.

Soit P le tenseur de taille $W \times H \times d$ des images intégrales des caractéristiques :

$$P(x, y, i) = \sum_{x'<x, y'<y} C(x', y', i) \quad i = 1, \ldots, d \tag{A.5}$$

et Q le tenseur de taille $W \times H \times d \times d$ des images intégrales sur les couples de caractéristiques :

$$Q(x, y, i, j) = \sum_{x'<x, y'<y} C(x', y', i)C(x', y', j) \quad i, j = 1, \ldots, d \tag{A.6}$$

L'image intégrale d'une image en niveau de gris de taille $W \times H$ se calcule en un passage sur l'image. Ici, d passages sur les canaux de C de taille $W \times H$ sont nécessaires pour calculer P et $(d + d^2)/2$ passages sont nécessaires pour calculer Q (car Q est symétrique), ce qui conduit à une complexité de $O(d^2WH)$ pour le calcul de l'ensemble des images intégrales. Soit $\mathbf{p}_{x,y}$ un vecteur de taille d et $\mathbf{Q}_{x,y}$ une matrice de taille $d \times d$:

$$\begin{aligned} \mathbf{p}_{x,y} &= [P(x, y, 1) \ldots P(x, y, d)]^{\mathrm{T}} \\ \mathbf{Q}_{x,y} &= \begin{pmatrix} Q(x, y, 1, 1) & \ldots & Q(x, y, 1, d) \\ \vdots & & \vdots \\ Q(x, y, d, 1) & \ldots & Q(x, y, d, d) \end{pmatrix} \end{aligned} \tag{A.7}$$

Avec ces notations, la matrice de covariance d'une région $R(1, 1; x'', y'')$, où $(1, 1)$ est la coordonnée du coin supérieur gauche et (x'', y'') est celle du coin inférieur droit, est donnée par :

$$C_{R(1,1;x'',y'')} = \frac{1}{N-1}\Big[\mathbf{Q}_{x'',y''} - \frac{1}{N}\mathbf{p}_{x'',y''}\mathbf{p}^{\mathrm{T}}_{x'',y''}\Big] \tag{A.8}$$

où $N = x'' \times y''$. De façon générale, la matrice de covariance d'une région $R(x', y'; x'', y'')$ est donnée par :

$$\begin{aligned} C_{R(x',y';x'',y'')} = \frac{1}{N-1}\Big[&\mathbf{Q}_{x'',y''} + \mathbf{Q}_{x',y'} - \mathbf{Q}_{x'',y'} - \mathbf{Q}_{x',y''} \\ &- \frac{1}{N}(\mathbf{p}_{x'',y''} + \mathbf{p}_{x',y'} - \mathbf{p}_{x',y''} - \mathbf{p}_{x'',y'})(\mathbf{p}_{x'',y''} + \mathbf{p}_{x',y'} - \mathbf{p}_{x',y''} - \mathbf{p}_{x'',y'})^{\mathrm{T}}\Big] \end{aligned} \tag{A.9}$$

avec $N = (x'' - x') \times (y'' - y')$. Une fois les images intégrales calculées, la complexité du calcul d'une matrice de covariance est en $O(d^2)$.

Annexe B

Sélection des sous-ensembles de caractéristiques pertinents

La procédure proposée par Yao et Odobez [86] est ici décrite. Le but est de sélectionner des sous-ensembles de caractéristiques de taille m parmi un ensemble de d caractéristiques ($m < d$). Soit $\mathcal{S}_d^m = \{S_{m,c}\}_{c=1,\ldots,\mathrm{C}_m^d}$ l'ensemble des sous-ensembles de taille m avec $S_{m,c}$ le c-ème sous-ensemble et $\mathrm{C}_m^d = \frac{d!}{m!(d-m)!}$ est le nombre de sous-ensembles de taille m. Le but de chaque itération de l'algorithme LogitBoost est de déterminer le meilleur couple (sous-fenêtre r^*,sous-ensemble c^*) qui minimise la log-vraissemblance binomiale négative, i.e. :

$$(r^*, c^*) = \underset{r,c}{\operatorname{argmin}}\Big(L_r(S_{m,c})\Big) \tag{B.1}$$

où $L_r(S_{m,c})$ est la log-vraissemblance binomiale négative définie à l'équation (3.5) après l'apprentissage d'un classifieur faible sur la sous-fenêtre r et sur le sous-ensemble de caractéristiques $S_{m,c}$. La recherche exhaustive du couple (r^*, c^*) nécessite d'apprendre et de tester $N_{sf} \times \mathrm{C}_m^d$ classifieurs faibles, ce qui devient rapidement infaisable lorsque $m > 2$. En effet, C_m^d augmente rapidement et le temps d'apprentissage d'un classifieur faible augmente également avec m. Pour $m > 2$, l'idée est donc de déterminer un nombre restreint de sous-ensembles de caractéristiques susceptibles d'amener de bonnes performances. Pour cela, on commence par tester tous les sous-ensembles de taille 2, ce qui donne $\{L_r(S_{2,c})\}_{c=1,\ldots,\mathrm{C}_2^d}$ où les plus petites valeurs indiquent que les paires de caractéristiques associées sont de bons choix pour la classification. Ensuite, pour chaque sous-ensemble $S_{m,c}$ de taille $m > 2$, une valeur de substitution $\tilde{L}_r(S_{m,c})$ pour la log-vraissemblance binomiale négative est calculée :

$$\tilde{L}_r(S_{m,c}) = \sum_{S_{2,s} \in S_{m,c}} L_r(S_{2,s}) \tag{B.2}$$

Connaissant ces valeurs de substitution, on teste seulement les q sous-ensembles présentant les plus faibles valeurs. Les tests de Yao et Odobez montrent qu'en sélectionnant $q = 8$ sous-

ensembles (parmis 56) pour $m = 3$ et $q = 12$ sous-ensembles (parmis 70) pour $m = 4$, la probabilité pour qu'un des sous-ensembles sélectionnés soit parmis les 3 meilleurs est de 0.94.

Publications dans le cadre de cette thèse

[Bouges 11] Pierre Bouges, Thierry Chateau, Christophe Blanc & Gaëlle Lossli. *Face detection in a pose different than the one learned*. In GRETSI, Bordeaux, France, Septembre 2011.

[Bouges 12a] Pierre Bouges, Thierry Chateau, Christophe Blanc & Gaëlle Lossli. *Improving existing cascaded face detector by adding occlusion handling*. In 21st IEEE International Symposium on Robot and Human Interactive Communication (Ro-Man), Paris, France, Septembre 2012.

[Bouges 12b] Pierre Bouges, Thierry Chateau, Christophe Blanc & Gaëlle Lossli. *Using k-nearest neighbors to handle missing weak classifiers in a boosted cascade*. In Proceedings of the 21th International Conference on Pattern Recognition (ICPR), Tsukuba, Japon, Novembre 2012.

[Bouges 12c] Pierre Bouges, Thierry Chateau, Christophe Blanc & Gaëlle Lossli. *Handling missing weak classifiers in boosted cascade : application to multiview and occluded face detection*. EURASIP Journal on Image and Video Processing, soumis le 2 Juillet 2012.

[Vu 13] Ngoc-Son Vu, Siméon Schwab, Pierre Bouges, Xavier Naturel, Christophe Blanc, Thierry Chateau & Laurent Trassoudaine. *Face recognition for video security applications*. In Interdiciplinary Workshop for the Global Security (WISG), Troyes, France, Janvier 2013.

Bibliographie

[1] MIT Center For Biological and Computation Learning. Cbcl face database #1.

[2] Lubomir Bourdev and Jonathan Brandt. Robust object detection via soft cascade. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) Volume 2*, CVPR '05, pages 236–243, 2005.

[3] S. Charles Brubaker, Matthew D. Mullin, and James M. Rehg. Towards optimal training of cascaded detectors. In *European Conference on Computer Vision (ECCV)*, pages 325–337, 2006.

[4] S. Charles Brubaker, Jianxin Wu, Jie Sun, Matthew D. Mullin, and James M. Rehg. On the design of cascades of boosted ensembles for face detection. Technical Report GIT-GVU-05-28, Georgia Institue of Technology, 2005.

[5] Roberto Brunelli and Daniele Falavigna. Person identification using multiple cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17 :955–966, 1995.

[6] Jie Chen, Shiguang Shan, Shengye Yang, Xilin Chen, and Wen Gao. Modification of the adaboost-based detector for partially occluded faces. In *Proceedings of the 18th International Conference on Pattern Recognition - Volume 02*, pages 516–519, 2006.

[7] Ian Craw, David Tock, and Alan Bennett. Finding face features. In *European Conference on Computer Vision (ECCV)*, pages 92–96, 1992.

[8] David Cristinacce. *Automatic detection of facial features in grey scale images*. PhD thesis, University of Manchester, 2004.

[9] Franklin C. Crow. Summed-area tables for texture mapping. *SIGGRAPH Computer Graphics*, 18(3) :207–212, January 1984.

[10] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *In Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society Conference on*, volume 1, pages 886–893, 2005.

[11] Ofer Dekel, Ohad Shamir, and Lin Xiao. Learning to classify with missing and corrupted features. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, pages 216–223, 2008.

[12] Yoav Freund et Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55 :119–139, 1997.

[13] Yoav Freund et Robert E. Schapire. A short introduction to boosting. *Journal of japanese society for artificial intelligence*, 14 :771–780, 1999.

[14] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression : a statistical view of boosting. *Annals of Statistics*, 28, 1998.

[15] Bernhard Fröba and Andreas Ernst. Face detection with the modified census transform. In *Proceedings of the Sixth IEEE international conference on Automatic face and gesture recognition*, FGR' 04, pages 91–96, 2004.

[16] C. Garcia and M. Delakis. A neural architecture for fast and robust face detection. In *ICPR'02 :Proceedings of the IEEE-IAPR International Conference on Pattern Recognition*, 2002.

[17] Christophe Garcia and Manolis Delakis. Convolutional face finder : A neural architecture for fast and robust face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 26(11) :1408–1423, november 2004.

[18] Amir Globerson and Sam Roweis. Nightmare at test time : robust learning by feature deletion. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 353–360, 2006.

[19] Bernd Heisele, Thomas Serre, and Tomaso Poggio. A component-based framework for face detection and identification. *International Journal of Computer Vision*, 74 :167–181, 2007.

[20] Erik Hjelmas and Boon Kee Low. Face detection : A survey. *Computer Vision and Image Understanding*, 83(3) :236–274, September 2001.

[21] Kazuhiro Hotta. robust face detector under partial occlusion. In *IEEE International Conference on Image Processing (ICIP 2004)*, Singapore, 2004.

[22] Chang Huang, Haizhou Ai, Yuan Li, and Shihong Lao. High-performance rotation invariant multiview face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(4) :671–686, 2007.

[23] Chang Huang, Haizhou Ai, Bo Wu, and Shihong Lao. Boosting nested cascade detector for multi-view face detection. *International Conference on Pattern Recognition*, 2 :415–418, 2004.

[24] Chien-Yuan Huang, Octavia I. Camps, and Tapas Kanungo. Object recognition using appearance-based parts and relations. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 878–884, 1997.

[25] Gary B. Huang, Marwan Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild : A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[26] Hongjun Jia and Aleix M. Martinez. Support vector machines in face recognition with occlusions. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 136–141, 2009.

[27] Michael Jones and Paul Viola. Fast multi-view face detection. Technical Report 96, Mitsubishi Electric Research Laboratories, 2003.

[28] Takeo Kanade. *Picture processing by computer complex adn recognition of human faces*. PhD thesis, University of Kyoto, 1973.

[29] Y. Kaya and K. Kobayashi. A basic study on human face recognition. In *International Conference on Frontiers of Pattern Recognition*, Hawaii, 1971.

[30] Andreas Lanitis, Chris J. Taylor, and Timothy F. Cootes. Automatic face identification system using flexible appearance models. *Image and Vision Computing*, 13(5) :393–401, 1995.

[31] Duy-Dinh Le and Shin'Ichi Satoh. A multi-stage approach to fast face detection. *IEICE Transactions on Information and Systems*, 89(7) :2275–2285, july 2006.

[32] Leonidas Lefakis and François Fleuret. Joint cascade optimization using a product of boosted classifiers. In *Conference on Neural Information Processing Systems (NIPS)*, 2010.

[33] Stan Z. Li and ZhenQiu Zhang. Floatboost learning and statistical face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9) :2004, 2004.

[34] Stan Z. Li, Long Zhu, ZhenQiu Zhang, Andrew Blake, HongJiang Zhang, and Harry Shum. Statistical learning of multi-view face detection. In *In Proceedings of the 7th European Conference on Computer Vision*, pages 67–81, 2002.

[35] Rainer Lienhart, Alexander Kuranov, and Vadim Pisarevsky. *Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection*, volume 2781/2003 of *Lecture Notes in Computer Science*, pages 297–304. Springer Berlin/Heidelberg, 2002.

[36] Rainer Lienhart and Jochen Maydt. An extended set of haar-like features for rapid object detection. In *IEEE ICIP 2002*, pages 900–903, 2002.

[37] Yen-Yu Lin and Tyng-Luh Liu. Robust face detection with multi-class boosting. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1 :680–687, 2005.

[38] Yen-Yu Lin, Tyng-Luh Liu, and Chiou-Shann Fuh. Fast object detection with occlusions. In *Computer Vision - ECCV 2004*, volume 3021 of *Lecture Notes in Computer Science*, pages 402–413. Springer Berlin / Heidelberg, 2004.

[39] Nick Littlestone. Learning quickly when irrelevant attributes abound : A new linear-threshold algorithm. *Machine Learning*, 2 :285–318, 1988.

[40] Huitao Luo. Optimization design of cascaded classifiers. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1 :480–485, 2005.

[41] Aleix M. Martínez. Recognizing imprecisely localized, partially occluded, and expression variant faces from a single sample per class. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24 :748–763, 2002.

[42] Aleix M. Martinez and Robert Benavente. The ar face database. Technical Report 24, CVC Technical Report, June 1998.

[43] Stephen J. McKenna, Shaogang Gong, and Yogesh Raja. Modelling facial colour and identity with gaussian mixtures. *Pattern Recognition*, 31(12) :1883–1892, 1998.

[44] Takeshi Mita, Toshimitsu Kaneko, and Osamu Hori. Joint haar-like features for face detection. In *ICCV '05 : Proceedings of the Tenth IEEE International Conference on Computer Vision*, pages 1619–1626, 2005.

[45] A. Mohan, C. Papageorgiou, and T. Poggio. Example-based object detection in images by components. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(4) :349–361, 2001.

[46] Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application (VISSAPP)*, pages 331–340. INSTICC Press, 2009.

[47] Stefan Munder and Dariu M. Gavrila. An experimental study on pedestrian classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11) :1863–1868, 2006.

[48] Andrew Y. Ng and Michael I. Jordan. On discriminative vs generative classifiers : A comparison of logistic regression and naive bayes. In *Advances in Neural Information Processing Systems 14*, pages 841–848. MIT Press, 2002.

[49] Edgar Osuna, Robert Freund, and Federico Girosi. Training support vector machines : an application to face detection. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0 :130, 1997.

[50] Constantine P. Papageorgiou, Michael Oren, and Tomaso Poggio. A general framework for object detection. *Computer Vision, IEEE International Conference on*, 0, 1998.

[51] Constantine P. Papageorgiou and Tomaso Poggio. A trainable object detection system : Car detection in static images, 1999.

[52] Minh-Tri Pham and Tat-Jen Cham. Fast training and selection of haar features using statistics in boosting-based face detection. *IEEE International Conference on Computer Vision (ICCV)*, pages 1–7, 2007.

[53] Thang V. Pham, Marcel Worring, and Arnold W. M. Smeulders. Face detection by aggregated bayesian network classifiers. *Pattern Recognition Letters*, 23(4) :451–461, February 2002.

[54] P. Jonathon Phillips, Hyeonjoon Moon, Syed A. Rizvi, and Patrick J. Rauss. The feret evaluation methodology for face-recognition algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10) :1090–1104, 2000.

[55] J. Ross Quinlan. *C4.5 : programs for machine learning*. Morgan Kaufmann, 1998.

[56] Dan Roth, Ming hsuan Yang, and Narendra Ahuja. A snow-based face detector. In *Advances in Neural Information Processing Systems 12*, pages 855–861. MIT Press, 2000.

[57] Henry A. Rowley, Shumeet Baluja, and Takeo Kanade. Neural network-based face detection. *IEEE Transactions On Pattern Analysis and Machine intelligence*, 20 :23–38, 1998.

[58] Henry A. Rowley, Shumeet Baluja, and Takeo Kanade. Rotation invariant neural network-based face detection. In *Computer Vision and Pattern Recognition*, pages 38–44, 1998.

[59] Maytal Saar-Tsechansky and Foster Provost. Handling missing values when applying classification models. *Journal of Machine Learning Research*, 8 :1623–1657, December 2007.

[60] Hichem Sahbi, Donald Geman, and Nozha Boujemaa. Face detection using coarse-tofine support vector classifiers. In *Proceedings of the IEEE International Conference on Image Processing (ICIP*, pages 925–928, 2002.

[61] Toshiyuki Sakai, Makoto Nagao, and Takeo Kanade. Computer analysis and classification of photographs of human faces. In *First USA-Japan Computer Conference*, pages 2–7, 1972.

[62] Robert E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2) :197–227, 1990.

[63] Robert E. Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37 :297–336, 1999.

[64] Henry Schneiderman and Takeo Kanade. A statistical method for 3d object detection applied to faces and cars. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1 :1746, 2000.

[65] Siméon Schwab, Thierry Chateau, Christophe Blanc, and Laurent Trassoudaine. Suivi de visages par regroupement de détections : traitement séquentiel par blocs. In *Reconnaissance des Formes et Intelligence Artificielle (RFIA)*, Lyon, France, January 2012.

[66] Fabrizio Smeraldi, Michael Defoin-Platel, and Mansoor Saqi. Handling missing features with boosting algorithms for protein-protein interaction prediction. In *Proceedings of the 7th international conference on Data integration in the life sciences*, DILS'10, pages 132–147, 2010.

[67] Kah-Kay Sung and Tomaso Poggio. Example-based learning for view-based human face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20 :39–51, 1998.

[68] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1) :71–86, january 1991.

[69] Oncel Tuzel, Fatih Porikli, and Peter Meer. Region covariance : A fast descriptor for detection and classification. In *In European Conference on Computer Vision (ECCV)*, volume 2, pages 589–600, 2006.

[70] Oncel Tuzel, Fatih Porikli, and Peter Meer. Human detection via classification on riemannian manifolds. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

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

[72] P. Viola, M. J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. *Computer Vision, IEEE International Conference on*, 2, 2003.

[73] Paul Viola and Michael Jones. Fast and robust classification using asymmetric adaboost and a detector cascade. In *Advances in Neural Information Processing System 14*, pages 1311–1318. MIT Press, 2001.

[74] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1, 2001.

[75] C. Y. Wang and Ziding Feng. Boosting with missing predictors. *Biostatistics*, 2009.

[76] Xiaoyu Wang, Tony X. Han, and Shuicheng Yan. An hog-lbp human detector with partial occlusion handling. In *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'09)*, Washington, DC, USA, 2009.

[77] Lior Wolf, Tal Hassner, and Yaniv Taigman. Similarity scores based on background samples. In *Asian Conference on Computer Vision (ACCV)*, Xi' an, September 2009.

[78] John Wright, Allen Y. Yang, Arvind Ganesh, S. Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31 :210–227, 2009.

[79] BO Wu and Ram Nevatia. Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. In *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, pages 90–97, 2005.

[80] Jianxin Wu, S. Charles Brubaker, Matthew D. Mullin, and James M. Rehg. Fast asymmetric learning for cascade face detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(3) :369–382, 2008.

[81] Rong Xiao, Huaiyi Zhu, He Sun, and Xiaoou Tang. Dynamic cascades for face detection. *IEEE International Conference on Computer Vision*, pages 1–8, 2007.

[82] Rong Xiao, Long Zhu, and Hong-Jiang Zhang. Boosting chain learning for object detection. In *IEEE International Conference on Computer Vision*, 2003.

[83] Shengye Yan, Shiguang Shan, Xilin Chen, and Wen Gao. Locally assembled binary (lab) feature with feature-centric cascade for fast and accurate face detection. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–7, 2008.

[84] Shengye Yan, Shiguang Shan, Xilin Chen, and Wen Gao. Fea-accu cascade for face detection. In *IEEE International Conference on Image Processing (ICIP)*, Cairo, Egypt, 2009.

[85] Ming-Hsuan Yang, David J. Kriegman, and Narendra Ahuja. Detecting faces in images : A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24 :34–58, 2002.

[86] Jian Yao and Jean-Marc Odobez. Fast human detection from jint appearance and foreground feature subset covariances. *Computer Vision and Image Understanding*, 115 :1414–1426, 2011.

[87] Kin Choong Yow and Roberto Cipolla. Feature-based human face detection. *Image and Vision Computing*, 15(9) :713–735, 1997.

[88] Cha Zhang and Paul Viola. Multiple-instance pruning for learning efficient cascade detectors. In *NIPS*, 2007.

[89] Cha Zhang and Zhengyou Zhang. A survey of recent advances in face detection. Technical Report 66, Microsoft Research, June 2010.

[90] Dong Zhang, S. Z. Li, and Daniel Gatica-Perez. Real-time face detection using boosting in hierarchical feature spaces. *International Conference on Pattern Recognition*, 2 :411–414, 2004.

[91] Lun Zhang, Rufeng Chu, Shiming Xiang, Shengcai Liao, and Stan Li. Face detection based on multi-block lbp representation. In *Advances in Biometrics*, volume 4642 of *Lecture Notes in Computer Science*, pages 11–18. Springer Berlin / Heidelberg, 2007.

[92] Wei Zheng and Luhong Liang. Fast car detection using image strip features. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

Glossaire

AdaBoost : (ou *Adaptative Boosting*) une des premières méthodes de *Boosting* introduite par Freund et Schapire.

Apprentissage artificiel : ensemble de méthodes dont le but est la compréhension de tout phénomène naturel par la construction de modèles mathématiques.

Apprentissage supervisé : ensemble de méthodes d'apprentissage artificiel qui s'appuie sur une base d'apprentissage constituée d'une grande quantité d'exemples labellisés.

***Boosting* :** domaine de la reconnaissance des formes regroupant de nombreux algorithmes qui combinent des classifieurs faibles pour construire un classifieur fort.

Cascade : association séquentielle de plusieurs fonctions de décision.

Classification : action d'associer un label à un exemple.

Classifieur faible : fonction de décision qui associe un label correct pour un peu plus de 50% des exemples.

Classifieur fort : fonction de décision qui est une combinaison linéaire de plusieurs classifieurs faibles.

Détection : action permettant de déceler la présence d'un objet, et éventuellement de préciser sa position.

En ligne : un processus est dit « en ligne » lorsqu'il s'exécute au fur et à mesure du travail principal qu'il doit réaliser.

Exemple : ensemble de données représentant un objet.

Fonction de décision : fonction associant un label à un exemple.

FROC : acronyme de *« Free Receiver Operating Curves »* , courbe couramment utilisée en apprentissage automatique pour évaluer les performances d'une fonction de décision, représentant le taux de bonnes détections en fonction du nombre de fausses alarmes.

Hors ligne : un processus hors-ligne réalise des tâches en différé du travail principal, soit avant en prévision d'une opération particulière à effectuer pour le travail principal, soit après pour traiter par exemple des informations récoltées pendant le travail principal.

label : mot ou nombre désignant un groupe d'objets cohérent suivant un ou plusieurs critères.

Reconnaissance des formes : ensemble de techniques visant à associer automatiquement un label à un exemple.

ROC : acronyme de *« Receiver Operating Curves »* , courbe couramment utilisée en apprentissage automatique pour évaluer les performances d'une fonction de décision, représentant le taux de bonnes détections en fonction du taux de fausses alarmes.

Temps réel : en informatique industrielle, on parle d'un système temps réel lorsque celui-ci contrôle un procédé physique à une vitesse adaptée à l'évolution de ce procédé.