



Université Mohamed Khider de Biskra  
Faculté des Sciences et de la Technologie  
Département de génie électrique

# MÉMOIRE DE MASTER

Sciences et Technologies  
Télécommunication  
Réseaux et Télécommunication

Réf. :

---

Présenté et soutenu par :  
**SAKHRI Ahmed Rami**

Le : samedi 23 juin 2018

## Optimisation de la détection pour la reconnaissance de visage

---

Jury :

Mlle. ZEHANI Soraya	MAA Univ. Biskra	Présidente
Mme. BELAHCENE Mebarka	MCA Univ. Biskra	Rapporteure
Mlle. MEDOUAKH Saadia	MAA Univ. Biskra	Examinatrice

Année universitaire : 2017/2018

# *Dédicaces*

*Au nom du dieu le clément et le miséricordieux louange à **ALLAH** le tout puissant.*

## *Dédicaces*

*Je dédie ce modeste travail en signe de respect, reconnaissance et de remerciement :*

*A mes chers parents,  
qui m'ont aidé de  
près et de loin.*

*A mes chers frères  
Halim, said,  
Akerem et yassine*

*A toutes mes chers amies  
Hocin, Abdelhafid, Baha, Abdou,  
Karim, Fares, Boura, Anwar  
Houssam, Wissal Nadji...*

*A tous mes collègues de spécialité  
Réseaux  
et Télécom Université Biskra*

*A tout ma famille, qui porte le nom **Sakhri**.*

*A tout ceux qui ont participé à l'élaboration de ce modeste travail et tous ceux qui nous  
sont chers.*

*M. SAKHRI AHMED R*

## Remerciement

*En tout premier lieu, je remercie le bon Dieu, tout puissant, de m'avoir donné la force pour survivre, ainsi que l'audace pour dépasser toutes les difficultés.*

*Nous voudrions adresser nos vifs remerciements*

*A notre enseignante et encadreur*

**M<sup>me</sup> BELAHCENE Mébarka**

*Maître de Conférences Habilité -A- Au Département de Génie Electrique*

*Nous vous reconnaissons la gentillesse et la spontanéité avec lesquelles vous avez bien voulu diriger ce travail.*

*Vous vous y êtes grandement impliqués par vos directives, vos remarques et suggestions, mais aussi par vos encouragements dans les moments clés de son élaboration.*

*Nous tenons à vous remercier aussi pour cette liberté que vous avez permise, votre manière de penser et de procéder, votre manière d'être, bref toute votre personnalité.*

*Nos remerciements s'adressent également à Monsieur le Président du jury et les membres du jury pour l'honneur qu'ils nous font d'avoir bien voulu étudier ce travail et de le juger.*

### *À ma très chère mère et à mon très cher père*

*Autant de phrases, aussi expressives soient-elles, ne sauraient montrer le degré d'amour et d'affection que j'éprouve pour vous.*

*Vos prières, vos encouragements et votre soutien m'ont toujours été d'un grand secours.*

*Vous ne cessé de me soutenir et de m'encourager avec tous les moyens et au prix de toutes les sacrifices durant toutes les années de mes études.*

*Vous étiez toujours présents pour me consoler quand il fallait.*

*En ce jour mémorable, pour moi ainsi que pour vous, recevez ce travail en signe de ma vive reconnaissance et de ma profonde estime.*

*Que le tout puissant vous donne la santé, le bonheur et une longue vie afin que je puisse vous combler à mon tour.*

*Je remercie Bilal Zoubiri de m'avoir aidé et de m'avoir soutenu et encouragé*

*Enfin, je remercie, mes très chers parents, mes frères et tous mes ami(e)s que j'aime tant*

*Afin de n'oublier personne, mes vifs remerciements s'adressent à tous ceux qui m'ont aidée à la réalisation de ce modeste mémoire*

# Liste des tableaux

## *Chapitre 1 Détection de Visage et Deep learning*

<b>TABLEAU 1. 1</b> CLASSIFICATION DES METHODES DE DETECTION DE VISAGE DANS UNE IMAGE ....	<b>9</b>
<b>TABLEAU 1. 2</b> EVALUATION DES PERFORMANCES DES PRINCIPALES METHODES DE DETECTION DE VISAGE (TAUX DE DETECTION / NOMBRE DE FAUX POSITIFS) SUR LES BASES CMU ET MIT ...	<b>11</b>
<b>TABLEAU 1. 3</b> PERFORMANCES DES DIFFERENTES METHODES DE DETECTION DE PEAU .	<b>18</b>

## *Chapitre 2 Etat de l'Art sur la détection et Deep Learning*

<b>TABLEAU 2. 1</b> (A GAUCHE) COMPARAISON DES APPROCHES SRNN PROPOSEES AVEC HIERARCHY LSTM V3 UTILISANT ALEXNET OU VGG 16 COMME CNN . (A DROITE) COMPARAISON DES ENTITES DE PERIPHERIE A L'AIDE DE SRNN-MAXNODE. ....	<b>41</b>
--	-----------

# Liste des figures

## *Chapitre 1 Détection de Visage et Deep learning*

<b>FIG. 1. 1</b> LES ETAPES DE LA RECONNAISSANCE DE VISAGE .....	6
<b>FIG. 1. 2</b> ILLUSTRATION DES COURBES DE PERFORMANCES .....	7
<b>FIG. 1. 3</b> ILLUSTRATION DES COURBES ROC ET CMC.....	8
<b>FIG. 1. 4</b> LA DETECTION DE LA COULEUR DE PEAU .....	10
<b>FIG. 1. 5</b> PROCESSUS DE LA MISE EN CORRESPONDANCE .....	11

## *Chapitre 2 Etat de l'Art sur la détection et Deep Learning*

<b>FIG. 2.1</b> LE MODELE PROFOND CONTEXTUEL MULTI-ETAPES .....	29
<b>FIG. 2. 2</b> LE SCHEMA DU SYSTEME DE DETECTION DE VISAGE PROPOSE SUR SFEW 2.0 .....	30
<b>FIG. 2. 3</b> EXEMPLES DE DETECTIONS DE VISAGE PAR JDA (ROUGE) ET DCNN (BLEU) .....	31
<b>FIG. 2. 4</b> (A) EXEMPLES DE DETECTION DE REPERES FACIAUX PAR UN SEUL CNN CONVENTIONNEL, LE CNN EN CASCADE , ET LE RESEAU CONVOLUTIF PROFOND A CONTRAINTES (TCDCN).....	33
<b>FIG. 2. 5</b> L'ALIGNEMENT DES FACES EST UN PROBLEME DE REGRESSION, MAIS RESOLU PAR UNE CLASSIFICATION A GRANDE ECHELLE . .....	36
<b>FIG. 2. 6</b> LES CONTRAINTES DE CALCUL DU MODELE AU MOMENT DU TEST .....	36
<b>FIG. 2. 7</b> UN EXEMPLE REEL .....	38
<b>FIG. 2. 8</b> DIFFERENTS CNN POUR LE TRAITEMENT D'IMAGE.....	39
<b>FIG. 2. 9</b> RESEAU ANTICIPE D'UN RNN STRUCTUREL (SRNN) FORMES EN CE QUI CONCERNE LES ETIQUETTES DE NŒUD . .....	40
<b>FIG. 2. 10</b> UN CADRE ETIQUETE COMME ACTIVITE DE GROUPE «SPIKE GAUCHE» ET ENCADRES ENTOURANT CHAQUE JOUEUR D'EQUIPE SONT ANNOTES DANS L'ENSEMBLE DE DONNEES AVEC DES ACTIONS INDIVIDUELLES . .....	40

## *Chapitre 3 Etude et Conception de la Détection*

<b>FIG. 3. 1</b> RESULTATS DE MSER SUR DES EXEMPLES D'IMAGES UNE PERSONNE .....	48
<b>FIG. 3. 2</b> RESULTAT DE MSER SUR IMAGE DE PLUSIEURS PERSONNES .....	48
<b>FIG. 3. 3</b> QUATRE DIRECTIONS DE BORD .....	49
<b>FIG. 3. 4</b> COMPARAISONS DE DETECTION DE LIGNE UTILISANT DIFFERENTES METHODES . .....	51
<b>FIG. 3. 5</b> (A) IMAGE ORIGINALE; (B) RESULTAT DE LA METHODE EDGE DETECTION; (C) RESULTAT D'AUTRES METHODES . .....	51
<b>FIG. 3. 6</b> LES LIGNES D'UNE LARGEUR DE PIXEL TRAVERSENT LE MASQUE EN TANT QUE LIGNE MEDIANE . .....	52
<b>FIG. 3. 7</b> LE MASQUE $3 \times 3$ . .....	54
<b>FIG. 3. 8</b> DETECTION DE CONTOURS .....	54
<b>FIG. 3. 9</b> IMAGES DETECTEES .....	55
<b>FIG. 3. 10</b> IMAGES NON DETECTEES.....	55
<b>FIG. 3. 11</b> LES QUATRE FILTRES DANS LE DOMAINE DE LA FREQUENCE SPATIALE .....	56
<b>FIG. 3. 12</b> UNE IMAGE D'ECHANTILLON DE VISAGE ET SES QUATRE IMAGES FILTRES DE GABOR . .....	57
<b>FIG. 3. 13</b> SCHEMA DE PRINCIPE DU GNN .....	58
<b>FIG. 3. 14</b> QUELQUES EXEMPLES D'IMAGES POUR TESTER LA STRUCTURE PROPOSEE .....	59
<b>FIG. 3. 15</b> EXEMPLES D'IMAGE DE MAUVAISE DETECTION DE LA POSE .....	59

<b>FIG. 3. 16</b>	EXEMPLE D'IMAGE DE MAUVAISE DETECTION DE LA COULEUR DE LA PEAU .....	60
<b>FIG. 3. 17</b>	EXEMPLE D'IMAGES DE MAUVAISE DETECTION AVEC L'ALGORITHME PROPOSE.....	60
<b>FIG. 3. 18</b>	LES DESCRIPTEURS DE HAAR . .....	61
<b>FIG. 3. 19</b>	CONTEXTE SUR L'ALGORITHME DE VIOLA-JONES .....	62
<b>FIG. 3. 20</b>	CASCADE DE CLASSIFIEURS .....	66
<b>FIG. 3. 21</b>	ALGORITHME D'APPRENTISSAGE ADABOOST . .....	67
<b>FIG. 3. 22</b>	DETECTION DE VISAGE AVEC LA METHODE DE VIOLA ET JONES. ....	70
<b>FIG. 3. 23</b>	EXEMPLE DE BONNE DETECTION SUR DES IMAGES DE PLUSIEURS PERSONNES.....	71
<b>FIG. 3. 24</b>	EXEMPLE DE MAUVAISE DETECTION SUR DES IMAGES DE PLUSIEURS PERSONNES .....	72
<b>FIG. 3. 25</b>	IMAGES DETECTEES .....	73
<b>FIG. 3. 26</b>	IMAGES DETECTEES ET DECOUPEES.....	73

### *Chapitre 4 Etude et Implémentation du RCNN pour la Détection*

<b>FIG. 4. 1</b>	SCHEMA D'UNE ARCHITECTURE DU CONVNET .....	78
<b>FIG. 4. 2</b>	EXEMPLE D'IMAGE (GAUCHE) ET DE SON TRAITEMENT PAR DEEPCREAM (DROITE) . .	79
<b>FIG. 4. 3</b>	ENSEMBLE DE NEURONES (CERCLES) CREANT LA PROFONDEUR D'UNE COUCHE DE CONVOLUTION (BLEU). ILS SONT LIES A UN MEME CHAMP RECEPTIF (ROUGE).....	82
<b>FIG. 4. 4</b>	MAX POOLING AVEC UN FILTRE $2 \times 2$ ET UN PAS DE 2 .....	86
<b>FIG. 4. 5</b>	ARCHITECTURE STANDARD D'UN RESEAU A CONVOLUTIONS.....	88
<b>FIG. 4. 6</b>	ARCHITECTURE DU RCNN .....	95
<b>FIG. 4. 7</b>	EXEMPLE DE RESULTATS DE TABLE DE VERITE DU TERRAIN AU FORMAT [X, Y, LARGEUR, HAUTEUR] .....	99
<b>FIG. 4. 8</b>	ARCHITECTURES UTILISEES POUR LA DETECTION RCNN.....	100
<b>FIG. 4. 9</b>	RESULTATS DE DETECTION PAR RCNN POUR DIFFERENTES ARCHITECTURES .....	101
<b>FIG. 4. 10</b>	RESULTATS DE BONNES DETECTIONS RCNN .....	102
<b>FIG. 4. 11</b>	RESULTATS DE MAUVAISES DETECTIONS RCNN .....	103
<b>FIG. 4. 12</b>	ECHANTILLON D'IMAGES D'UNE BDD .....	104
<b>FIG. 4. 13</b>	ECHANTILLON DE DETECTION DE VISAGE D'IMAGES D'UNE PERSONNE.....	105
<b>FIG. 4. 14</b>	EXEMPLE DE DETECTION DE VISAGE D'UN GROUPE DE PERSONNES SUR UNE IMAGE (EN MILIEUX INCONTROLES) .....	106
<b>FIG. 4. 15</b>	EXEMPLE DE DETECTION DE VISAGE SUR DES IMAGES EN MILIEUX INCONTROLES... ..	107

## Liste des Abréviations

- ADN** : Acide Désoxyribose Nucléique
- ASM** : modèles de formes actives
- BDD** : base de donnée
- CIE** : Commission Internationale de l'Eclairage
- CLM** : Modèles locaux
- CMC** : Cumulative Match Characteristics
- CNN** : Convolutional Neural Network
- CRN** : réseau de raffinement en cascade
- DCNN** : Deep-CNN
- DPM** : modèle de pièce déformable
- EER** : Error Equal Rate ( Taux d'égalité erreur )
- FAR** : False Accept Rate (Taux de fausse acceptation)
- FER** : reconnaissance d'expression faciale
- FP** : taux des faux positifs
- FRR** : False Reject Rate (Taux de faux rejet)
- GMM** : mélange de modèles gaussiens
- GNN** : Gabor Neural Network
- HOG** : Histogramme de gradient orienté
- HSL** : Hue, Saturation, Luminance
- JDA** : the joint cascade detection and alignment;
- LBP** : binaire local pattern
- LBP-TOP** : modèles binaires locaux sur trois plans orthogonaux
- LSTM** : Long Short-Term Memory
- MGS** : modèle gaussien simple
- MoT** : Mixtures of Trees
- MSER features** : Maximally stable extremal regions features
- MTL** : L'apprentissage multi-tâches
- PHOG** : l'histogramme pyramidale des gradients orientés

**RBM** : Restricted Boltzmann Machines  
**RCNN** : Region Convolutional Neural Network  
**RGB (RVB)** : Rouge vert bleu  
**ROC** : Receiver Operating Characteristic  
**ROR** : Rank One Recognition(Taux d'identification)  
**RPR** : Rank of Perfect Recognition  
**SOM** : Self-Organizing Map  
**SRNN** : réseau neuronal récurrent structurel  
**SVM** : Support Vector Machines  
**THG** : Transformation de Hough Généralisée  
**TP** : taux des vrais positifs  
**TSL** : Teinte, Saturation,Luminance  
**TV** : variation totale



## Résumé

La recherche dans le domaine de la détection de personne en milieux incontrôlés devient un sujet de plus en plus traité. Les algorithmes conventionnels utilisent exclusivement des modèles basés sur des vecteurs de caractéristiques. Ces vecteurs restent difficiles à définir et dépendent largement du type d'images observé. Ces méthodes donnent des résultats avec un faible taux de détection et un haut taux de fausse classification. Une approche innovante pour résoudre ce problème est d'utiliser un algorithme permettant de déterminer automatiquement les caractéristiques utiles pour reconnaître les personnes. Dans notre projet, nous proposons un réseau de neurones convolutif pour identifier les visages en temps réel et en milieux incontrôlés. Les réseaux de neurones convolutifs ont montré leur grande performance dans le domaine de la classification des objets. Testée sur des images de joueurs et de célébrités collectées sur le Web acquises en milieu réel, l'approche proposée atteint une meilleure performance de classification que les méthodes conventionnelles. Ces résultats indiquent clairement que l'utilisation des réseaux de neurones convolutifs pour la détection de visage de personne est très prometteuse.

### ملخص

أصبح البحث في مجال الكشف البشري في بيانات غير خاضعة للسيطرة موضوعاً يثير الفلق المتزايد. لا تستخدم الخوارزميات التقليدية سوى النماذج القائمة على المتجهات المميزة. ويظل من الصعب تحديد هذه النواقل وتعتمد إلى حد كبير على نوع الصور الملحوظة. هذه الأساليب تعطي نتائج مع معدل اكتشاف منخفض ومعدلات عالية من سوء التصنيف. النهج المبكر لحل هذه المشكلة هو استخدام خوارزمية لتحديد الخصائص التي من المفيد التعرف عليها. في مشروعنا ، نقترح شبكة عصبية تلافيفية لتحديد الوجوه في الوقت الحقيقي وفي البيانات غير المتحكم فيها. وقد أظهرت الشبكات العصبية التحويلية أدائها الكبير في مجال تصنيف الكائنات. تم اختباره على صور اللاعبين والمشاهير التي تم جمعها على الويب المكتسبة في الحياة الحقيقية ، فإن المنهج المقترح يحقق أداء تصنيف أفضل من الأساليب التقليدية. هذه النتائج تشير بوضوح إلى أن استخدام الشبكات العصبية التحويلية للكشف عن الوجه واعدة جداً.

# Table des matières

<b>DEDICACES</b> .....	<b>I</b>
<b>REMERCIEMENT</b> .....	<b>II</b>
<b>LISTE DES TABLEAUX</b> .....	<b>III</b>
<b>LISTE DES FIGURES</b> .....	<b>IV</b>
<b>LISTE DES ABREVIATIONS</b> .....	<b>VI</b>
<b>RESUME</b> .....	<b>VIII</b>
<b>TABLE DES MATIERES</b> .....	<b>IX</b>
<b>INTRODUCTION GENERALE</b> .....	<b>1</b>
<b>CHAPITRE 1 DETECTION DE VISAGE ET DEEP LEARNING</b> .....	<b>4</b>
INTRODUCTION .....	5
1.1 SYSTEMES BIOMETRIQUES BASES SUR LA RECONNAISSANCE DE VISAGE .....	5
1.2 MESURE DE PERFORMANCE D'UN SYSTEME BIOMETRIQUE .....	7
1.3 DETECTION DE VISAGE.....	9
1.3.1 Approche basée sur la reconnaissance .....	9
1.3.2 Approche basée sur les caractéristiques invariantes .....	10
1.3.3 La couleur de peau .....	10
1.3.4 Approche basée sur l'appariement de gabarits (Template matching) .....	10
1.3.4.1 Des faces prédéfinies de visages .....	11
1.3.4.2 Approche basée sur l'apparence.....	11
1.3.5 Détection de visages basée sur l'analyse de la couleur de la peau .....	12
1.4 LES METHODES DE DETECTION EXPLICITES .....	13
1.5 LES APPROCHES DE DETECTION NON PARAMETRIQUES.....	14
1.6 LES APPROCHES PARAMETRIQUES .....	14
1.7 APPROCHES SEMI-PARAMETRIQUES .....	16
1.8 AUTRES APPROCHES DE DETECTION.....	17
1.9 COMPARAISON DES DIFFERENTES APPROCHES .....	18
1.10 DETECTION AVEC DEEP LEARNING .....	19
CONCLUSION .....	19
<b>CHAPITRE 2 ETAT DE L'ART SUR LA DETECTION ET DEEP LEARNING</b> .....	<b>20</b>
INTRODUCTION .....	21
PARTIE A TRAVAUX ANTERIEURS SUR LA DETECTION.....	21
A.1 Travaux basiques sur la détection .....	21
A.2 Travaux antérieurs sur la détection avec Deep Learning .....	23
A.2.1 Détection des points de repère faciaux (Facial landmark detection) .....	24
A.2.2. Désembrouillage (Deblurring) et les structures CNN pour le traitement d'image.....	26
A.2.3 Reconnaissance d'activité d'individus .....	26
PARTIE B TRAVAUX RECENTS SUR L'APPRENTISSAGE PROFOND POUR LA DETECTION .....	28
B.1 Apprentissage en profondeur contextuel en plusieurs étapes pour la détection des piétons .....	28
B.2 Reconnaissance d'expression faciale statique basée sur l'image avec apprentissage en réseau profond multiple.....	29
B.3 R-CNN rapide .....	31
B.4 Apprendre la représentation en profondeur pour l'alignement du visage avec les attributs auxiliaires .....	32
B.5 Analyse de repères faciaux de force brute avec un classificateur de 140 000 voies.....	34
B.5.1 L'Approche.....	35
B.5.2 Évaluation .....	36

B.6 Réseau récurrent à l'échelle pour le désembrouillage d'image en profondeur .....	37
B.7 Réseau Neuronal Récurrent Structurel (SRNN) pour l'Analyse d'Activité de Groupe.....	39
CONCLUSION .....	42
<b>CHAPITRE 3 ETUDE ET CONCEPTION DE LA DETECTION .....</b>	<b>43</b>
INTRODUCTION .....	44
3.1 METHODES DETECTION DE VISAGE .....	44
3.2 METHODE DE DETECTION MSER FEATURES ( <i>MAXIMALLY STABLE EXTREMAL REGIONS FEATURES</i> ) .....	45
3.2.1 <i>Modèle Mathématique</i> .....	45
3.2.2 <i>Avantages de MSER</i> .....	45
3.2.3 <i>Comparaison avec d'autres détecteurs de région</i> : .....	46
3.2.4 <i>Extensions et adaptations</i> .....	47
3.2.5 <i>Implémentation de la méthode de détection MSER et résultats</i> .....	47
DISCUSSION : .....	48
3.3 METHODE EDGE DETECTION .....	49
3.3.1 <i>Méthode de détection des contours</i> .....	49
3.3.2 <i>Filtrage optimal</i> .....	52
3.3.3 <i>Dérivée première</i> .....	52
3.3.3.1 <i>Méthode de base</i> .....	52
3.3.3.2 <i>Filtre de Prewitt</i> .....	52
3.3.3.3 <i>Filtre de Sobel</i> .....	53
3.3.3.4 <i>Filtre de Canny</i> .....	53
3.3.4 <i>Implémentation de la méthode edge detection et résultats</i> .....	53
3.3.4.1 <i>Application d'edge detection aux images couleurs</i> .....	53
DISCUSSION.....	55
3.4 DETECTION DE VISAGE A L'AIDE DE RESEAUX NEURONAUX ET DE FONCTIONNALITES GABOR.....	56
3.4.1 <i>Principe du Filtre de Gabor</i> .....	56
3.4.2 <i>Convolution du filtre de Gabor et de l'image</i> .....	56
3.4.3 <i>Réseaux neuronaux</i> .....	57
3.4.4 <i>Implémentation de la méthode réseaux de neurones-Gabor (GNN)</i> .....	58
3.4.5 <i>Résultats pour la détection par GNN</i> .....	59
3.5 METHODE DE VIOLA-JONES.....	60
3.5.1 <i>Caractéristiques</i> .....	62
3.5.2 <i>Sélection de caractéristiques par boosting</i> .....	63
3.5.3 <i>Cascade de classifieurs</i> .....	65
3.5.4 <i>Étapes clés</i> .....	66
3.5.5 <i>Applications de la méthode de Viola Jones</i> .....	67
3.5.6 <i>D'autres implémentations de Viola Jones</i> .....	68
3.5.7 <i>Mécanisme d'évaluation de la qualité de la performance des algorithmes de recherche d'objets</i> .....	68
3.5.8 <i>Partie Experimentale</i> .....	69
3.5.9 <i>Résultats de la détection par Viola Jones</i> .....	70
DISCUSSION.....	73
CONCLUSION .....	73
<b>CHAPITRE 4 ETUDE ET IMPLEMENTATION DU RCNN POUR LA DETECTION .....</b>	<b>75</b>
INTRODUCTION .....	76
4.1 POURQUOI AVONS-NOUS BESOIN DE MACHINE LEARNING?.....	77
4.2 EN SAVOIR PLUS SUR LES RESEAUX DE NEURONES CONVOLUTIONNELS .....	77
4.3 RESEAU NEURONAL CONVOLUTIF.....	79
4.3.1 <i>Présentation</i> .....	79
4.3.2 <i>Traitement convolutif</i> .....	80
4.3.2.1 <i>Traitement de profondeur 1 et sans chevauchement (facilitant la compréhension)</i> .....	80
4.3.2.2 <i>Généralisation (profondeur et chevauchement)</i> .....	80
4.3.2.3 <i>Caractéristiques et avantages</i> .....	80

4.3.2.4 Blocs de construction.....	81
4.3.2.5 Filtres et foulée .....	82
4.3.2.6 Fonctionnalités .....	82
4.3.2.7 Zéro remplissage.....	82
4.3.2.8 Nombre de neurones.....	83
4.3.2.9 Paramètres d'apprentissage .....	83
4.3.3 Couche de normalisation des lots.....	83
4.3.4 ReLU Layer.....	84
4.3.5 Couche de normalisation de canal (normalisation de réponse locale).....	84
4.3.6 Couches Max et Moyenne-Pooling.....	85
4.3.7 Couche entièrement connectée .....	86
4.3.8 Couches de sortie.....	86
4.3.8.1 Softmax.....	86
4.3.8.2 couches de classification.....	87
4.3.9 Couche de régression.....	87
4.4 ARCHITECTURE DE MODELES DE CNN.....	88
4.5 CONFIGURATION DES PARAMETRES ET FORMATION D'UN RESEAU NEURONAL CONVOLUTIF .....	89
4.5.1 Spécification du solveur et du nombre maximum d'époques .....	89
4.5.2 Spécification et modification du taux d'apprentissage.....	89
4.5.3 Spécification des données de validation.....	90
4.5.4 Sélection d'une ressource matérielle .....	91
4.5.5 Enregistrer les réseaux de points de contrôle et reprendre la formation .....	91
4.5.6 Configurer les paramètres dans les couches convolutives et entièrement connectées .....	91
4.5.7 Entraîner le réseau .....	92
4.6 RECADRAGE INTELLIGENT.....	92
4.7 DETECTION DE VISAGE BASEE SUR LE RESEAU NEURONAL.....	93
4.8 DETECTION DE VISAGE DANS DES ENVIRONNEMENTS NON CONTROLES .....	93
4.9 DETECTION D'OBJET AVEC UN R-CNN.....	94
4.9.1 Architecture du RCNN.....	94
4.9.2 Principe de fonctionnement du RCNN .....	95
4.9.3 Description de l'implémentation du RCNN .....	96
4.9.3.1 Processus RCNN .....	97
4.9.4 Expériences menées sur la détection par RCNN .....	98
4.9.4.1 Définition d'arguments d'entrée.....	98
4.9.4.2 options - Options de formation.....	99
4.9.4.3 Résultats pour la détection RCNN.....	100
DISCUSSION.....	103
CONCLUSION .....	108
<b>CONCLUSION GENERALE .....</b>	<b>110</b>
<b>REFERENCES .....</b>	<b>114</b>

### Introduction Générale

En tant que technologie majeure dans le traitement de l'information sur le visage humain, la détection du visage humain est le processus le plus important dans des applications telles que la vidéosurveillance, la reconnaissance faciale et la fabrication assistée par ordinateur. Cependant, en raison des variations dans le fond, l'illumination, la pose et les expressions faciales, la détection de visage est toujours un grand défi dans le monde réel. La chose la plus importante est que la détection de visage est la première étape de la reconnaissance faciale, et sa précision affecte les résultats de la reconnaissance.

La détection des visages est un problème bien étudié en vision par ordinateur. Les détecteurs de visage modernes peuvent facilement détecter les faces proches de l'avant. Des recherches récentes dans ce domaine se concentrent davantage sur le problème incontrôlé de détection de visage, où un certain nombre de facteurs tels que les changements de pose, les expressions exagérées et les illuminations extrêmes peuvent entraîner de grandes variations visuelles de l'apparence du visage.

Les difficultés de détection de visage proviennent principalement de deux aspects: 1) les grandes variations visuelles des visages humains dans les arrière-plans encombrés; 2) le grand espace de recherche des positions possibles du visage et des tailles de visage. La première exige que le détecteur de visage traite avec précision un problème de classification binaire alors que le dernier impose en outre une exigence d'efficacité temporelle.

Depuis le travail fondateur de Viola-Jones [1], la cascade boostée avec des fonctionnalités simples devient la conception la plus populaire et la plus efficace pour la détection de visage pratique. La nature simple des fonctionnalités permet une évaluation rapide et un rejet précoce rapide des fausses détections positives. Pendant ce temps, la cascade boostée construit un ensemble de caractéristiques simples pour obtenir une classification précise du visage et du non-visage. Le détecteur de visage original de Viola-Jones utilise la fonction de Haar qui est rapide à évaluer mais suffisamment discriminante pour les visages frontaux. Cependant, en raison de la nature simple de la fonction de Haar, elle est relativement faible dans l'environnement incontrôlé où les visages sont dans des poses variées, des expressions sous un éclairage inattendu.

## Introduction Générale

Un certain nombre d'améliorations au détecteur de visage Viola-Jones ont été proposées au cours de la dernière décennie. La plupart d'entre eux suivent le cadre de cascade boosté avec plus fonctionnalités avancées.

Outre l'architecture en cascade, le modèle de pièce déformable (DPM) est une autre méthode de détection de visage. Les modèles DPM apprennent les filtres racines, les filtres de pièces et leurs relations spatiales via la machine à vecteurs de support (SVM). Les modèles DPM ont une grande robustesse pour l'occlusion, mais ils sont coûteux en calcul pour construire des pyramides de caractéristiques.

Récemment, le réseau neuronal convolutif (CNN) a obtenu des performances d'impression dans des tâches de vision par ordinateur, telles que la classification d'images et la reconnaissance. Yang et al. en [2] démontre que les modèles CNN d'attributs faciaux peuvent être appliqués pour trouver des propositions faciales et que les fenêtres proposées peuvent être traitées par un modèle CNN de type AlexNet.

Cependant, cette méthode prend beaucoup de temps en pratique en raison de sa structure CNN complexe. Li et al. en [3] propose une architecture en cascade appelée Cascade CNN pour la détection de visages réels. Cependant, son réseau d'étalonnage pour la boîte englobante nécessite des dépenses de calcul supplémentaires. Zhang et al. en [4] propose un CNN en cascade multi-tâche pour la détection et l'alignement des visages. Il a examiné la pertinence de la détection et de l'alignement des visages, mais le processus de formation est assez coûteux. Farfadi et al. introduit le détecteur de visage dense profond (DDFD), qui ajuste finement l'AlexNet pré-entraîné pour détecter des visages dans une large gamme d'orientations. Huang et al. en [5] propose un framework FCN de bout en bout appelé DenseBox pour la détection de visages. Cependant, DDFD et DenseBox n'adoptent qu'un seul modèle à échelle unique.

Dans ce travail, après étude et conception de détecteurs classiques pour leur simplicité mais non efficaces. Nous proposons d'appliquer le Region Convolutional Neural Network (RCNN) pour faire la détection de visages en milieux incontrôlés. Par rapport aux précédentes fonctionnalités, CNN peut apprendre automatiquement des fonctionnalités pour capturer des variations visuelles complexes en exploitant une grande quantité de données d'entraînement et sa phase de test peut être facilement parallélisée sur les cœurs GPU pour l'accélération. Compte tenu de la charge de calcul relativement élevée des CNN, l'analyse exhaustive de l'image complète à plusieurs échelles avec un CNN profond n'est pas une solution pratique. Pour obtenir une détection rapide des visages, nous présentons une méthode RCNN, qui se base sur la détection

## Introduction Générale

de régions d'intérêts dans l'image dans les premières étapes à faible résolution et vérifie soigneusement les détections dans les étapes ultérieures à haute résolution afin de sélectionner la région à détecter. Nous montrons que cette solution peut surpasser les méthodes de détection de visage classiques pour certaines applications.

Dans ce travail, nos principales contributions sont les suivantes: (1) La réalisation de plusieurs techniques de détection en milieux incontrôlés et (2) La conception d'une détection basé sur le RCNN (Région Convolutional Neural Network) réseau diminuant la complexité du modèle par rapport aux méthodes existantes du CNN.

Notre mémoire est organisé comme ceci :

Dans le **chapitre 1** nous présentons l'essentiel sur la détection en général notamment la détection basée sur le Deep Learning ;

Le **chapitre 2** est consacré à l'état de l'art des travaux récents sur la détection des objets et des visages par l'apprentissage en profondeur particulièrement l'architecture fondamentale du CNN.

Dans le **chapitre 3**, nous présentons l'étude et la conception de quatre types de détecteur que nous jugeons intéressants et candidats à notre application la reconnaissance.

Le **chapitre 4** est dédié à l'étude et implémentation du RCNN pour la détection de visage. Le détecteur de visage RCNN est proposé en détail, y compris l'architecture du réseau, les stratégies de formation et les paramètres.

Dans les deux **chapitres 3 et 4**, les conceptions sont suivies démonstration et analyse des résultats expérimentaux

La dernière partie du mémoire résume le document par une **conclusion générale** et propose le travail futur

Chapitre 1 Détection de Visage et Deep learning



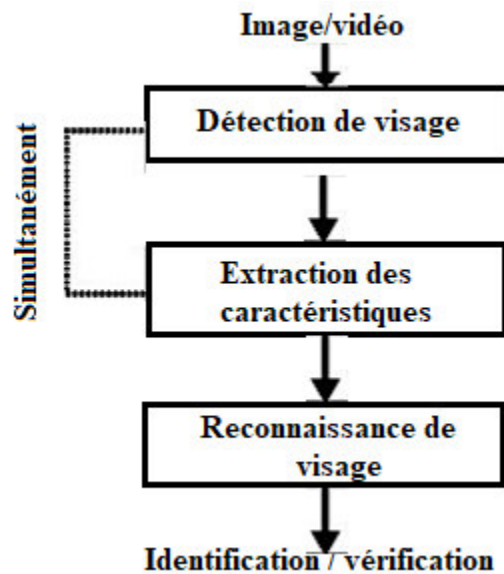
## Introduction

La détection joue un rôle de plus en plus important dans les systèmes de reconnaissance suscitant l'intérêt de la communauté scientifique ainsi que les industriels. En effet, plusieurs applications et technologies biométriques ont été développées basées sur l'usage des caractéristiques physiques, biologiques et comportementales (visages, empreintes digitales, voix, salive, sang, ADN, signature, etc.). Ces caractéristiques sont spécifiques pour chaque individu et demeurent pratiquement inchangées durant toute la vie. Dans le cas de la reconnaissance de visages, les chercheurs ne sont pas encore arrivés à une performance comparable à celle de l'Homme, mais plusieurs pistes sont en cours d'exploration et semblent prometteuses classée parmi les principales techniques de biométrie.

L'authentification de visages et l'identification de visages sont deux notions différents: dans des tâches d'authentification, le système sait a priori l'identité de l'utilisateur et il doit juste la vérifier; en d'autres termes, le système doit décider si l'utilisateur à priori est un imposteur ou non. Alors que dans l'identification de visages, l'identité a priori n'est pas connue: le système doit alors décider quelles sont les images parmi ceux stockées dans la base de données ressemblent le plus à l'image à identifier [6].

### 1.1 Systèmes biométriques basés sur la reconnaissance de visage

La reconnaissance automatique de visage s'effectue en trois étapes principales : (1) détection de visages, (2) extraction et normalisation des caractéristiques du visage, (3) identification et/ou vérification (**Fig. 1.1**). Certaines techniques de traitements d'images peuvent être communes à plusieurs étapes. Par exemple, l'extraction des caractéristiques faciales (yeux, nez, bouche) est utilisée aussi bien pour la détection que pour l'identification de visages. Par ailleurs, les étapes de détection de visage et d'extraction de caractéristiques peuvent être exécutées simultanément. Cela dépend notamment de la nature de l'application, de la taille de la base d'apprentissage, et des conditions de prise de vue (bruit, occultation, etc.). Enfin, les techniques de traitement utilisées dans chaque étape sont très critiques pour les applications biométriques, et doivent, par conséquent, être optimisées pour améliorer les performances du système global.



*Fig. 1. 1 Les étapes de la reconnaissance de visage [6]*

Dans ce qui suit nous allons détailler chaque étape du système de reconnaissance faciale, et nous présenterons les principales difficultés rencontrées.

Ce projet se concentre principalement sur la tâche de détection pour la reconnaissance de visages. Il existe dans la littérature une multitude de méthodes de détection de visage et d'identification. Nous passerons en revue, dans ce chapitre, les techniques les plus utilisées.

Le principal problème dans la détection et la reconnaissance d'un objet est relatif aux différentes représentations possibles de celui-ci. Ainsi la détection et la reconnaissance du visage dépendent de plusieurs facteurs, dont les plus étudiés sont :

- *La position* : sur une image, un visage peut être vu de face, de profil, ou d'un angle quelconque.
- *L'expression faciale* : l'apparence d'un visage dépend aussi de son expression.
- *La présence d'attributs* : une personne peut avoir un chapeau, des lunettes, une moustache, une barbe, une cicatrice....
- *Les conditions extérieures* : la couleur, l'intensité de l'éclairage, la taille, la texture sont différentes sur chaque image.
- *L'occultation* : une partie du visage peut être cachée par un autre objet ou par une autre personne.
- *La couleur* : les êtres humains ont des couleurs de peau différentes, d'où la différence de la valeur du pixel représentant la peau de chaque personne.

1.2 Mesure de performance d'un système biométrique

Tout d'abord, afin de comprendre comment déterminer la performance d'un système biométrique [6], il nous faut définir clairement trois critères principaux :

1. **Taux de faux rejet** (“*False Reject Rate*” ou **FRR**) : Ce taux représente le pourcentage de personnes censées être reconnues mais qui sont rejetées par le système ;

$$FRR = \frac{\text{nombre de faux rejet}}{\text{nombre clients presentes}} \tag{Equ. 1.1}$$

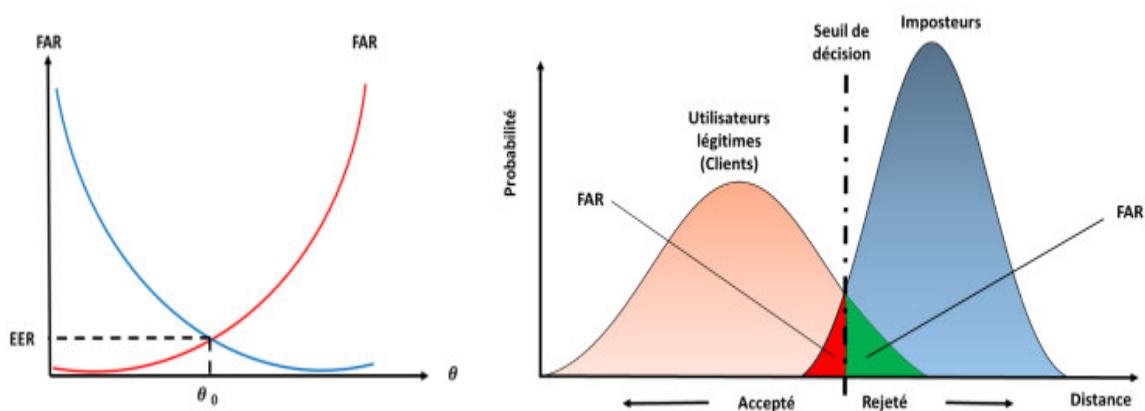
2. **Taux de fausse acceptation** (“*False Accept Rate*” ou **FAR**) : Ce taux représente le pourcentage de personnes censées ne pas être reconnues mais qui sont tout de même acceptées par le système :

$$FAR = \frac{\text{nombre de fausse acceptations}}{\text{nombre d'imposteurs presentes}} \tag{Equ. 1.2}$$

3. **Taux d'égal erreur** (“*Equal Error Rate*” ou **EER**) : Ce taux est calculé à partir des deux premiers critères et constitue un point de mesure de performance courant. Ce point correspond à l'endroit où **FRR = FAR**, c'est-à-dire le meilleur compromis entre les faux rejets et les fausses acceptations.

$$EER = \frac{\text{nombre de fausse acceptations} + \text{nombre de faux rejet}}{\text{nombre totale d'accès}} \tag{Equ. 1.3}$$

La Fig. 1.2 (a) illustre l'EER à partir des courbes FRR ( $\theta$ ) et FAR ( $\theta$ ) ou  $\theta$  est le seuil de décision.



(a) Graphe démonstratif de l'EER

(b) Illustration du FRR et du FAR.

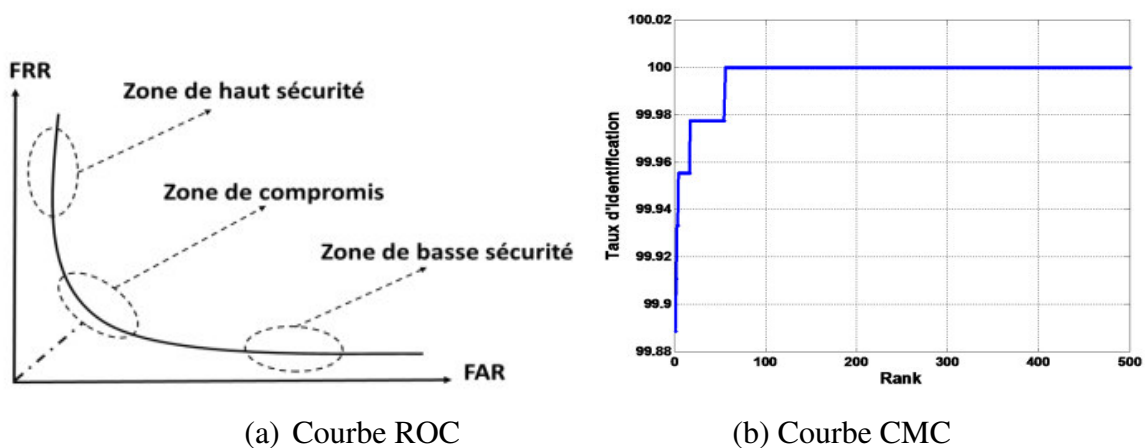
Fig. 1. 2 Illustration des courbes de performances

La courbe de distribution permet de visualiser la répartition des scores des imposteurs et des clients en fonction du seuil de décision. Le but fondamental de tout système biométrique opérant au niveau score, est de pouvoir séparer au maximum les distributions de score des imposteurs et des authentiques (voir la **Fig. 1.2 (b)**). En minimisant la zone de recouvrement entre ces deux distributions, on améliore la performance globale du système en augmentant le taux de reconnaissance.

Selon le mode (*vérification* ou *identification*) du système biométrique, il existe deux façons d'un mesurer la performance :

Lorsque le système opère en mode **vérification**, on utilise ce que l'on appelle une courbe (**ROC**) "*Receiver Operating Characteristic*" représentée à la **Fig. 1.3 (a)**. Elle permet de représenter graphiquement la performance d'un système de vérification pour les différentes valeurs du seuil de décision.

**Le taux d'erreur égal** (*Equal Error Rate* ou **EER**) correspond au point **FAR = FRR**, c'est-à-dire graphiquement à l'intersection de la courbe ROC avec la première bissectrice. Il est fréquemment utilisé pour donner un aperçu de la performance d'un système. Cependant, il est important de souligner que l'EER ne résume en aucun cas toutes les caractéristiques d'un système biométrique. Le seuil  $\theta$  doit donc être ajusté en fonction de l'application ciblée : haute sécurité, basse sécurité ou compromis entre les deux.



**Fig. 1.3** Illustration des courbes ROC et CMC

En mode **identification**, il peut être utile de savoir si le bon choix se trouve parmi les N premières réponses du système. On trace alors la courbe "*Cumulative Match Characteristics*" (**CMC**) qui représente la probabilité que le bon choix se trouve parmi les N premiers résultats. Comme l'illustre la **Fig. 1.3 (b)**.

1.3 Détection de visage

La détection du visage sert à reconnaître un objet dans une image comme étant un visage et de le distinguer du reste de l'image. La détection du visage humain est une tâche compliquée vus les facteurs exposés ci-dessus. Les méthodes de détection de visages ont été classifiées par Yang [7] en quatre approches, voir (**tableau 1.1**) :

- Approche basée sur la reconnaissance.
- Approche basée sur les caractéristiques invariantes.
- Approche basée sur l'appariement de gabarits (Template matching)
- Approche basée sur l'apparence

*Tableau 1. 1 Classification des méthodes de détection de visage dans une image [7]*

Approach	Representative Works
<b>Knowledge-based</b>	Multiresolution rule-based method
<b>Feature invariant</b>	
- Faciale feature	Grouping of edges
- Texture	Space Gray-Level Dependence matrix (SGLD) of face pattern
- Skin Color	Mixture of Gaussian
- Multiple Feature	Integration of skin color, size and shape
<b>Template matching</b>	
Predefined face templates	Shape template
Deformable Template	Actives Shape Model (ASM)
<b>Appearance-based method</b>	
Eigenface	Eigenvector decomposition clustering
Distribtion-based	Gaussien distribution and multilayer perceptron
Neural Network	Ensemble of nerual network and arbitration schemes
Support Vector Machine (SVM)	SVM with polynomial kemel
Naives Bayes Classifier	Joint statistics of local appearance and position
Hidden Markov Model (HMM)	Higher order statistics with HMM
Information-Theorectical Approach	Kullback relative information

1.3.1 Approche basée sur la reconnaissance

C'est une méthode fondée sur des règles qui représentent les composants principaux et représentatifs des visages humains. Les règles sont généralement constituées à partir de la relation entre les caractéristiques du visage. Par exemple, les visages sur les images ont souvent

deux yeux qui sont symétriques, un nez et une bouche. La relation entre ces membres peut être représentée par la distance entre ces derniers et leur position.

Un problème se pose avec cette approche, en l'occurrence la difficulté de traduire les connaissances relatives aux visages humains en règles bien définies, ce qui peut provoquer des erreurs de détection et rendre le système peu fiable.

### 1.3.2 Approche basée sur les caractéristiques invariantes

Cette famille d'algorithmes a pour objectif de trouver les caractéristiques structurelles même si le visage est dans différentes positions, conditions lumineuses ou angle de vue. Le problème que rencontre cette approche est que la qualité des images peut être sévèrement diminuée à cause de l'illumination, le bruit ou l'occlusion. Cependant, Il existe plusieurs propriétés ou caractéristiques invariables du visage dont les principales sont les suivantes :

#### 1.3.3 La couleur de peau

La couleur de la peau de l'être humain a été utilisée pour la détection des visages et sa pertinence a été prouvée comme caractéristique spécifique au visage.

Le principe de cette méthode est basé sur l'information couleur pour la discrimination des pixels de peau ou non-peau. Chaque pixel d'une image couleur est codé dans un espace couleur (par exemple RGB ou YCrCb, ..).

Cette méthode se résume en trois étapes : 1) Prétraitement de l'image ; 2) Choix d'un espace de couleurs ; 3) Seuillage et segmentation de la couleur de peau.



*Fig. 1. 4 la détection de la couleur de peau*

Cette méthode est caractérisée par la rapidité de traitement et par la simplicité de la décision. En effet le principe est simple et limité à la couleur de peau sans aucune considération des effets d'échelle et de position. Néanmoins, cette méthode affiche des détections des faux positifs et peut créer des conflits avec l'arrière plan.

#### 1.3.4 Approche basée sur l'appariement de gabarits (Template matching)

La détection de visages entiers ou de parties de visage se fait à travers un apprentissage d'exemples standards de visages. La corrélation entre les images d'entrées et les exemples enregistrés est calculée et utilisée pour la décision.

1.3.4.1 Des faces prédéfinies de visages

Cette technique est utilisée pour classer des objets, elle est très intéressante pour la détection de visage de par sa facilité d'application. Le principe de cette méthode est basé sur une comparaison effectuée entre une image quelconque et un modèle prédéfini, dont le but est de calculer la corrélation pour aboutir à une décision par oui/non. La correspondance est faite pixel par pixel.

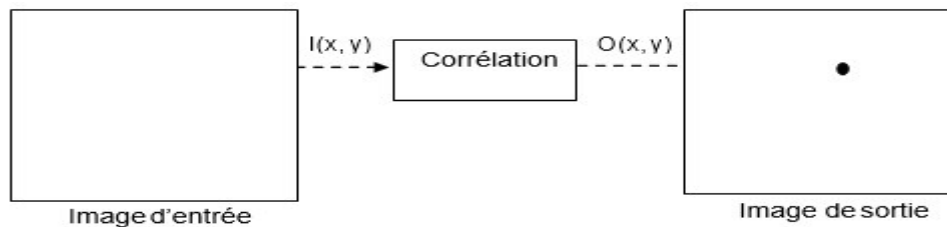


Fig. 1. 5 Processus de la mise en correspondance

Cette méthode a l'avantage d'être simple mais elle est sensiblement influencée par la variation d'échelle, de pose et de forme.

1.3.4.2 Approche basée sur l'apparence

Cette approche a pour objectif de déterminer les caractéristiques significatives des visages et des non visages à partir de techniques d'analyse statistique et d'apprentissage organisés par le biais de modèles de distribution ou par une fonction discriminante. La classification visage ou non visage est représentée par une variable aléatoire  $x$  (dérivée d'une image ou d'un vecteur caractéristique).

Parmi les méthodes utilisées dans ce contexte nous citons : Les réseaux de neurones, les machines à vecteur de support SVM, les modèles cachés de Markov HMM, Viola-Jones...

Tableau 1. 2 Evaluation des performances des principales méthodes de détection de visage (Taux de détection / Nombre de Faux Positifs) sur les bases CMU et MIT (Extrait de [8]).

Détecteur de visages	CMU	CMU-125	MIT	MIT-20
Rowley et al. [Row98],	86,2%/23		84,5%/8	
Schneiderman et Kanade [Sch00]		94,4%/65		
Féraud et al [Fer01]	86%/8			
Viola et jones [Vio01]	88,4%/31		77,8%/5	
Garcia et al [Gar04]	90,3/8	90,5/8	90,1/7	90,2/5

Dans le cadre de cette mémoire de fin d'études , nous nous intéressons en particulier aux techniques de détection de visages La section suivante, présentera une étude détaillée sur ces techniques, ce qui nous permettra de situer notre approche par rapport aux travaux existants.

## 1.3.5 Détection de visages basée sur l'analyse de la couleur de la peau

Les méthodes de détection basées sur l'analyse de la couleur de la peau sont des méthodes efficaces et rapides. Elles réduisent l'espace de recherche de la région visage dans l'image. De plus, la couleur de la peau est une information robuste face aux rotations, aux changements d'échelle, et aux occultations partielles. Plusieurs espaces couleur peuvent être utilisés pour détecter, dans l'image, les pixels qui ont la couleur de la peau. L'efficacité de la détection dépend essentiellement de l'espace couleur choisi. Les espaces couleur les plus utilisés sont :

- L'espace RVB, mis au point en 1931 par la Commission Internationale de l'Eclairage (CIE). Il consiste à représenter l'espace des couleurs à partir de trois rayonnements monochromatiques de couleurs : Rouge-Vert-Bleu. Cet espace correspond à la façon dont les couleurs sont généralement codées informatiquement, ou plus exactement à la manière dont les écrans à tubes cathodiques (ordinateurs, TV) représentent ces couleurs.
- L'espace HSL (Hue, Saturation, Luminance), appelé aussi TSL (Teinte, Saturation, Luminance) en Français, s'appuie sur les travaux du peintre Albert H. Munsell. C'est un modèle de représentation dit "naturel", car il est proche de la perception physiologique de la couleur par l'œil humain. En effet, le modèle RGB aussi adapté soit-il pour la représentation informatique de la couleur ou bien l'affichage sur les périphériques de sortie, ne permet pas de sélectionner facilement une couleur. Le modèle HSL consiste à décomposer la couleur selon des critères physiologiques :
  - la teinte (en Anglais Hue), correspondant à la perception de la couleur,
  - la saturation, décrivant la pureté de la couleur, c'est-à-dire son caractère vif ou terne,
  - la luminance, indiquant la quantité de lumière de la couleur, c'est-à-dire son aspect clair ou sombre.

Il existe d'autres modèles naturels de représentation proches du modèle HSL :

- HSB : Hue, Saturation, Brightness soit en Français Teinte, Saturation, Brilliance. La brillance décrit la perception de la lumière émise par une surface.
- HSV : Hue, Saturation, Value soit en Français Teinte, Saturation, Valeur.
- HSI : Hue, Saturation, Intensity soit en Français Teinte, Saturation, Intensité.
- HCI : Hue, Chrominance, Intensity soit Teinte, Chrominance, Intensité.
- Le modèle YCrCb est un signal non-linéaire codé à partir du signal RVB. Le paramètre Y représente la luminance (c'est-à-dire l'information en noir et blanc), tandis que Cr et Cb permettent de représenter la chrominance, c'est-à-dire l'information couleur.



- Le modèle colorimétrique  $L^*a^*b^*$  (aussi connu sous le nom de CIELab), dans lequel une couleur est repérée par trois valeurs :
  - $L^*$ , la luminance, exprimée en pourcentage (0 pour le noir à 100 pour le blanc).
  - $a^*$  et  $b^*$  deux gammes de couleur allant, respectivement, du vert au rouge et du bleu au jaune avec des valeurs allant de -120 à +120.

Le modèle  $L^*a^*b^*$  couvre ainsi l'intégralité du spectre visible par l'œil humain et le représente de manière uniforme. Il permet donc de décrire l'ensemble des couleurs visibles indépendamment de toute technologie graphique. Les techniques de détection du visage basées sur la couleur de la peau peuvent être classifiées en quatre catégories que nous allons maintenant aborder: les méthodes explicites, les méthodes non paramétriques, les méthodes paramétriques, et les méthodes semi paramétriques. Toutes ces approches pratiquent une phase d'apprentissage sur un nombre d'images représentatives pour calculer une densité de probabilité de la couleur peau.

#### 1.4 Les méthodes de détection explicites

Les méthodes explicites utilisent des règles de décision empiriques et/ou statistiques [9] pour la détection des pixels ayant la couleur de la peau. Les images doivent être acquises dans un environnement contrôlé (i.e. un éclairage réglable) [10]. Une méthode explicite est une méthode de classification qui consiste à définir explicitement les frontières de la région peau (cluster) dans l'espace couleur utilisé.

Dans [11], par exemple, la peau est classifiée dans l'espace couleur  $RGB$  en utilisant les règles suivantes :

$$R > 95 \text{ et } G > 40 \text{ et } B > 20 \text{ et } \max\{R, G, B\} - \min\{R, G, B\} > 15$$

$$\text{Et } |R - G| > 15 \text{ et } R > G \text{ et } R > B$$

Par ailleurs, Chai et Ngan [12] ont proposé un algorithme de segmentation de visage dans lequel ils ont employé les deux plans CbCr du modèle couleur YCrCb pour déterminer les régions ayant la couleur peau. Ils ont utilisé la base de données de visages ECU [13]. Ils ont trouvé que les valeurs de pixels dans les domaines  $DCb = [77, 127]$  et  $DCr = [133, 173]$  définissent bien les pixels peau. De même, Garcia et Tziritas [14] ont segmenté la peau en utilisant, huit plans dans l'espace YCbCr, ou six plans dans l'espace HSV.

L'avantage de ces méthodes réside dans la simplicité des règles de détection de la peau qu'elles utilisent, ce qui permet une classification rapide. Cependant, leur problème principal est la difficulté de déterminer empiriquement un espace couleur approprié ainsi que des règles de

décision adéquates qui assurent un taux de reconnaissance élevé. Une méthode utilisant des algorithmes d'apprentissage a été proposée pour résoudre ces problèmes [15]. Les auteurs commencent par choisir un espace RGB normalisé (où  $r = R / R+B+G$ ,  $g = G / R+B+G$  et  $b = B / R+B+G$  et la somme des trois composantes normalisées  $r+g+b= 1$ ) sur lequel ils appliquent un algorithme d'induction constructive afin de créer de nouveaux ensembles d'attributs pour les composantes RGB. Une règle de décision, semblable à l'équation qui réalise la meilleure identification possible, est estimée pour chaque ensemble d'attributs. Ils ont obtenu des résultats meilleurs que ceux qui sont obtenus avec un classifieur de Bayes défini dans l'espace RGB.

### 1.5 Les approches de détection non paramétriques

Les approches non paramétriques ne dépendent pas de la forme de la fonction de distribution de la teinte. Elles utilisent les histogrammes couleur 2D et 3D pour représenter la distribution (ou densité de probabilité) de la tonalité de la peau dans un espace couleur. L'avantage d'utiliser les histogrammes couleur est qu'ils sont robustes (invariants) aux occultations et aux changements de point de vue. Ils peuvent ainsi différencier un grand nombre d'objets [10] dans une tâche de reconnaissance. En général, les approches non paramétriques se déroulent en trois étapes :

- Construire les histogrammes de couleur de peau et de non peau à partir de l'espace couleur.
- Calculer la probabilité conditionnelle pour chaque couleur de peau et de non peau.
- Utiliser la règle de Bayes afin de calculer la probabilité pour qu'un pixel corresponde à la classe peau. Ce processus, permet ainsi de créer une carte de probabilité de la couleur peau.
- Un seuil de classification est enfin déterminé à partir de la relation entre les détections correctes et les détections fausses données par la courbe ROC (Receiver Operating Characteristic). Il est appliqué à la carte de probabilité et permet d'extraire les régions peau.

### 1.6 Les approches paramétriques

Les approches non-paramétriques basées sur les histogrammes couleur exigent un grand espace mémoire. De plus, leurs performances dépendent directement des images d'apprentissage. Le besoin d'un modèle plus compact pour représenter la peau, et qui peut être généralisé avec moins de données d'apprentissage, a poussé les chercheurs à développer des

modèles paramétriques de distribution de peau. Cette dernière est souvent représentée sous la forme d'un modèle gaussien simple (MGS) parfois elliptique, ou par un mélange de modèles gaussiens (GMM) pour traduire son caractère multimodal. Plusieurs travaux sur la modélisation de la distribution de la couleur de peau ont utilisé un mélange de Gaussiennes défini par :

$$p(c; \mu, \Sigma) = \sum_{k=1}^n \alpha_k \frac{1}{\sqrt{(2 * \pi)^d |\Sigma_k|}} \exp^{-0.5(c-\mu_k)^T (\Sigma_k)^{-1} (c-\mu_k)} \quad (\text{Equ. 1.4})$$

Où  $C$  est la représentation des espaces couleur utilisés,  $n$  le nombre de gaussiennes utilisées,  $\alpha$  le poids de la  $k^{ieme}$  gaussienne,  $\Sigma$  la matrice de covariance,  $\mu$  le vecteur moyen, et  $d$  la dimension des données de  $C$ . Pour une gaussienne simple (uni-modale), les valeurs de  $\alpha$  et de  $n$  sont égales à 1. Dans ce cas, les paramètres peuvent être estimés en utilisant le maximum de vraisemblance, alors que pour le mélange de gaussiennes ils sont déterminés en utilisant l'algorithme *EM* (Expectation - Maximization algorithm) [18][17][16]. Le choix du nombre  $n$  de gaussiennes dépend énormément des données d'apprentissage et du choix de l'espace couleur utilisé. Plusieurs techniques ont utilisé des valeurs de  $n$  comprises entre 2 et 16.

Yang and Ahuja [19] ont utilisé deux gaussiennes dans l'espace couleur LUV. Leurs tests ont démontré que le modèle MGS n'était pas suffisants pour modéliser la distribution de la peau avec la base de données de Michigan. Greenspan et al. [20] ont démontré qu'un modèle MMG est une représentation robuste qui s'adapte aux différents espaces couleur et aux variations d'illumination. Leur modèle MMG contient deux composantes, la première modélise la distribution de la lumière ordinaire de la couleur de peau, alors que la deuxième composante modélise la distribution des régions les plus lumineuses de la peau. En étudiant la distribution de la couleur de peau et de non-peau dans plusieurs espaces couleurs, Lee et Yoo [21] ont conclu que les régions de couleur de peau ont approximativement une forme elliptique qui ne peut pas être modélisée par un MGS. En effet, en raison de l'asymétrie du cluster de peau par rapport au pic de densité, l'utilisation du modèle gaussien symétrique (MGS) mène à un taux élevé de faux positifs. Afin de résoudre ce problème, ils ont proposé donc un nouveau modèle qu'ils ont appelé « modèle elliptique de frontière ». Les auteurs ont comparé leur modèle avec les modèles MGS et MMG à six composantes, appliqués sur la base de données de Compaq [19]. Ils ont obtenu des performances légèrement meilleures. Cependant, l'inconvénient du modèle elliptique de frontière réside dans le fait que son utilisation est limitée à la classification binaire.

Hsu [22] propose de combiner une technique de correction de couleur avec la détection de la

couleur de peau, pour localiser le visage dans une image. La correction de couleur permet d'éliminer l'effet de la réflexion et de la variation de l'illumination dans l'image. La technique de compensation d'éclairage utilise ce qu'on appelle un « blanc de référence ». Elle est basée sur l'hypothèse qu'une image contient toujours du blanc. Les pixels blancs sont reconnus en utilisant une correction non-linéaire Gamma de la valeur de luminance. Si le pourcentage des pixels blancs dépasse 5 % sur 100 pixels, on applique alors une correction de couleur sur l'image. Cette correction s'applique sur les coefficients des trois axes RGB.

Enfin, pour détecter la couleur de peau on utilise un MGS dans l'espace modifié CbCr, qui est obtenu à partir d'une transformation non linéaire appliquée à l'espace YCbCr. L'inconvénient de cette méthode de correction réside dans la difficulté à définir les hypothèses de départ [23].

### **1.7 Approches semi-paramétriques**

Décrit par Kohonen au début des années 80, le réseau SOM (Self-Organizing Map) [24] est un des plus populaires réseaux de neurones non supervisé. Il est principalement utilisé pour classifier des données de grande dimension, mais il fonctionne tout aussi bien pour des données de faible dimension. L'objectif du réseau SOM est de classifier les pixels d'entrée selon qu'ils correspondent ou non à des pixels de peau, ce qui permet au final d'extraire la région du visage. Brown et al. [25] entraînent deux réseaux SOM pour apprendre, à partir de 500 images, la distribution des pixels de la couleur peau et la couleur non peau. Les performances du SOM ont été testées sur l'ensemble des images d'apprentissage/test de la base de données Compaq [26]. Plusieurs espaces couleur (RGB normalisé, Teinte-Saturation, TLS) ont été utilisés avec le détecteur SOM. Les résultats ont montré que les performances des détecteurs de peau SOM ne dépendent pas des espaces couleurs utilisés, à l'inverse du modèle MMG. Par ailleurs, les performances du détecteur SOM sont inférieures à celles qui sont obtenues par la méthode basée sur les histogrammes RGB, développée dans [26]. Néanmoins, le détecteur SOM nécessite moins de paramètres et il est plus efficace pour les applications en temps réel.

Certains auteurs ont également proposé un apprentissage automatique des couleurs représentant la peau à l'aide de réseaux neuronaux. Ceux-ci peuvent être entraînés à partir d'échantillons de pixels représentant la peau (et non-peau), préalablement convertis dans l'espace de couleur YCrCb. Par ailleurs, d'autres espaces de couleurs ont aussi été utilisés dans ce contexte. Kakumanu et al. [27,28] ont employé un réseau de neurones pour modéliser la constance de la couleur « the color constancy ». Leur réseau de neurones possède trois couches et permet d'estimer l'illumination de la peau. Il prend en entrée un histogramme à deux composants  $r$  et

g et donne en sortie les illuminations de la peau dans l'espace  $r, g$ . Ce réseau de neurones est entraîné sur une base de données de 255 images, et testé sur 71. Les images utilisées fournissent un grand choix d'illuminations, de différents arrières plans et de sources de lumière non blanches. Enfin, une technique de seuillage simple est utilisée pour détecter la peau à partir de ce réseau de neurones.

### 1.8 Autres approches de détection

Sobottka et Pitas [29] proposent une méthode de localisation et d'extraction des caractéristiques du visage qui utilise à la fois la forme et la couleur [30]. La segmentation de l'image se fait dans l'espace HSV pour localiser la région couleur de la peau. Les composantes connexes sont ensuite déterminées en appliquant un algorithme de croissance de régions avec une résolution grossière. Pour chaque composante connexe, l'algorithme ajuste une ellipse afin de déterminer la région candidate qui correspond au visage. Enfin, une analyse plus fine des caractéristiques à l'intérieur de cette région permet de conclure sur la présence d'un visage ou non.

Dans [31,32], un modèle gaussien de la couleur de peau est utilisé pour classifier les pixels de couleurs de peau. Afin de caractériser la forme des « clusters » dans l'image binarisée, un ensemble de 11 moments géométriques d'ordre inférieur est calculé en utilisant la transformée de Fourier et la transformée radiale de Mellin. Afin de détecter la région visage, un réseau de neurones est entraîné à l'aide des moments géométriques extraits. L'expérimentation de cette méthode a démontré un taux de détection de 85% sur une base de test de 100 images. Belaroussi et al. [33] proposent de combiner trois détecteurs pour localiser le visage :

- un détecteur anthropomorphique basé sur un modèle d'apparence neuronal. Il s'agit d'un réseau de neurones auto-associateur (réseau Diabolo déjà utilisé en reconnaissance d'écriture). Ce réseau est entraîné à reconstruire la classe "visage" (pendant l'apprentissage, l'entrée sert de sortie désirée) en réalisant une compression des données (analyse en composantes principales non linéaire) dans sa couche cachée comportant un faible nombre de cellules. Une image de non-visage sera en principe mal compressée et donnera une erreur de reconstruction plus importante.
- un détecteur géométrique basé sur la Transformation de Hough Généralisée (THG). L'ellipse est une forme géométrique simple qui permet de modéliser grossièrement un visage. Une THG est donc réalisée sur l'image des orientations de gradients afin de détecter une ellipse verticale d'excentricité donnée. Il en résulte un tableau de vote dont le maximum correspond à la position

dans l'image du point le plus susceptible d'être le centre de l'ellipse.

- un détecteur colorimétrique basé sur une modélisation statistique de la teinte chair par mélange de gaussiennes (MMG) dans l'espace YCbCr.

Le traitement parallèle d'une image par les trois détecteurs produit trois cartes de probabilité qui sont combinées linéairement. Le maximum (cas mono-visage) ou les  $n$  premiers maxima (cas multi-visages) de la combinaison donne la localisation. Les expériences, conduites sur la base ECU (3000 images), montrent que la combinaison des détecteurs permet d'atteindre des taux de localisation supérieurs à 85%.

### 1.9 Comparaison des différentes approches

Les performances des différentes méthodes de modélisation de la couleur de la peau ont été évaluées sous des conditions identiques. Malheureusement, la plupart des méthodes de détection de la peau fournissent des résultats obtenus sur leurs propres bases de données qui ne sont pas disponibles. La base d'apprentissage et de test la plus utilisée pour la détection de la peau est la base de données Compaq [26].

Dans le tableau (1.3) les performances des différentes méthodes appliquées à cette base de données, sont présentés. Il donne le taux des vrais positifs (TP) et le taux des faux positifs (FP). Bien que les méthodes utilisent la base de données de manière différente pour définir les images d'apprentissage et les images de test, et emploient différentes stratégies d'apprentissage, ce tableau donne une image assez fidèle des performances obtenues par ces méthodes [28].

*Tableau 1. 3 Performances des différentes méthodes de détection de peau [6].*

Méthodes	Espace couleur	TP	FP
Bayes SPM [Jon99]	RGB	90%	14.2%
Bayes SPM [Bra00]	RGB	93.4%	19.8%
MMG [Jon99]	RGB	90%	~15.5%
SOM [Bro01]	TLS	78%	32%
Elliptical boundary model [Lee02]	CIE-xy	90%	20.9%
MGS [Lee02]	CbCr	90%	33.3%
MMG [Lee02]	IQ	90%	30%
Thresholding of I axis[Bra00]	YIQ	94.7%	30.2%
MGS [Hsu02]	YCbCr	96%	

## 1.10 Détection avec Deep Learning

Ces dernières années, les méthodes basées sur l'apprentissage en profondeur ont amélioré la précision et la robustesse des cas difficiles. (Fan et Zhou 2016) ont employé plusieurs CNN dans une façon grossier-à-fin. (Zhang et al., 2016b) ont adopté l'apprentissage multitâche dans leur cadre CNN en cascade. (Zhu et al., 2016) ont traité les coordonnées  $(x, y, z)$  en valeurs RVB et, avec l'image, les ont introduites dans un CNN, qui affine itérativement les paramètres faciaux sous-jacents. D'autres travaux ont intégré des modèles récurrents (Trigeorgis et al., 2016, Peng et al., 2016) ou des modèles génératifs (Zhang et al., 2016a).

La comparaison de ces différentes méthodes dans des scénarios vidéo réels plus difficiles était relativement inconnue. Il n'y avait pas de protocole d'évaluation communément accepté ou suffisamment de données annotées pour le suivi et l'alignement du visage avant la publication du repère de 300VW (Shen et al., 2015, Chrysos et al., 2017). Ce benchmark contient plus de 100 vidéos annotées et vise à évaluer le suivi des repères faciaux dans des contextes contraints et non contraints. Plusieurs méthodes ont été proposées pour répondre à cette tâche difficile. (Yang et al., 2015) ont utilisé une régression de forme en cascade spatio-temporelle qui combinait une régression multivue avec une régression de série temporelle pour améliorer la cohérence temporelle. (Uricar et Franc 2015) ont utilisé un détecteur de modèle déformable étendu avec un filtre de Kalman pour le lissage temporel (Xiao, Yan et Kassim 2015) a présenté une approche basée sur la régression multi étapes, qui initialise progressivement les contours plus complexes des points de repère stables. Rajamanoharan et Cootes 2015) ont proposé un CLM à vues multiples qui utilise un modèle de forme global avec des cartes de réponse spécifiques à la tête. (Wu et Ji 2015) ont proposé une approche pour mieux utiliser l'information de forme dans les régresseurs en cascade, en combinant explicitement la forme avec l'information d'apparence. Dans le contexte en ligne, (Sanchez Lozano et al., 2016) a proposé une régression continue en cascade qui peut être mise à jour de façon incrémentielle.

## Conclusion

Dans ce chapitre, nous avons présenté les systèmes biométriques basés sur la reconnaissance et détection de visage et nous avons aussi donné un aperçu sur les techniques de leurs performances. En effet, nous avons présenté un aperçu des approches les plus utilisées dans la détection de visage et nous avons basé notre étude sur l'apprentissage en profondeur pour la détection de visage dont l'étude détaillée fera objet du chapitre 3.

Chapitre 2 Etat de l'Art sur la détection et Deep Learning



### Introduction

Le Grand Défi de la reconnaissance des visages consiste en divers sous-défis basés respectivement sur les variantes émotions, poses, occlusions... Ces ensembles de données présentent toujours plus de difficultés que beaucoup de données conventionnelles en raison de leurs caractéristiques plus spontanées. Alors qu'un certain nombre de caractéristiques artisanales telles que les modèles binaires locaux sur trois plans orthogonaux (LBP-TOP), l'histogramme pyramidale des gradients orientés (PHOG) et les modèles quantized locaux (LPQ) ont été prouvés. Ils fonctionnent bien sur des ensembles de données conventionnels, ils obtiennent des performances nettement inférieures sur ces ensembles de données. Le réseau de neurones convolutionnels profonds (CNN Convolutional Neural Network) a récemment donné d'excellentes performances dans une grande variété de tâches de classification d'images. La conception soignée de l'apprentissage de fonctionnalités local à global avec la convolution, la mise en pool et l'architecture en couches rend la capacité de représentation visuelle très forte, ce qui en fait un outil puissant pour la reconnaissance faciales.

### Partie A Travaux antérieurs sur la détection

#### A.1 Travaux basiques sur la détection

Les nouvelles fonctionnalités améliorent constamment les performances de détection. Les travaux sur la conception de caractéristiques incluent des caractéristiques de Haar, l'invariance invariable (SIFT), le SIFT, l'histogramme de gradients (HOG), l'histogramme de gradient, un binaire local pattern (LBP), l'histogramme des couleurs et l'auto-similarité des couleurs (CSS). D'autres indices comme la profondeur, la segmentation et le mouvement améliorent également les performances de détection. On utilise aussi les fonctionnalités HOG et CSS modifiées. La plupart des approches considèrent par exemple la détection des piétons comme une tâche de classification en scannant une image avec des fenêtres coulissantes dont les dimensions sont variables. Un grand nombre d'approches de classification génératives et discriminatives ont été développées. Les méthodes génératives calculent la probabilité d'une fenêtre entourant un piéton. Les classificateurs discriminants, tels que les classificateurs renforçateurs et les SVM, recherchent des paramètres pour séparer les échantillons positifs et négatifs. Les modèles à base de parties se sont révélés efficaces dans la détection et la reconnaissance d'objets. Le modèle basé sur la partie déformable est capable de détecter des objets avec quelques changements de position. Ce modèle est ensuite étendu aux classificateurs en cascade pour augmenter la vitesse

de calcul. Poselet dans [41] est capable de gérer la variation d'apparence des pièces. Comme les piétons ont une apparence variée, le mélange des parties a été utilisé dans de nombreuses approches. Les modèles de mélange forment les classificateurs à l'aide d'un regroupement supervisé ou non supervisé. Différemment, les classificateurs en cascade sont formés avec des échantillons de formation mal classés étape par étape. Les classeurs en cascade ont deux avantages : (1) ils peuvent fournir des hyperplans de classification linéaires par morceaux, et (2) ils aident à sauvegarder la charge de calcul de la détection d'objets de fenêtre glissante. Par conséquent, de nombreuses approches ont utilisé la cascade [42]. Alors que la structure en cascade a bien fonctionné dans de nombreux domaines, le seuillage strict pour chaque classificateur en cascade rejette beaucoup d'informations recueillies à chaque classificateur. Pour éviter de tels inconvénients, les approches récentes de détection ont utilisé la cascade molle [42,43], qui recueille les scores de classification extraits par chaque niveau de classificateurs puis combine les scores de classification pour la décision finale. Cependant, la cascade douce apprend encore les classificateurs étape par étape sans optimisation conjointe. Les classificateurs à différentes étapes ne peuvent pas coopérer les uns avec les autres dans la procédure de formation (apprentissage). Hinton et al. [44] ont prouvé que l'ajout d'une nouvelle couche, si elle est faite correctement, crée un modèle qui a une meilleure limite inférieure variationnelle sur la probabilité de log des données d'apprentissage. Cependant, le lien entre les modèles profonds et les classificateurs à plusieurs degrés est inconnu. Récemment, des modèles profonds ont été appliqués avec succès à la reconnaissance digitale manuscrite [44], la reconnaissance faciale [45,46,47]. Sermanet et al. [48] ont étudié les fonctionnalités multi-étapes apprises non supervisées avec un modèle profond. Cependant, ils n'ont pas ajouté de classificateur supplémentaire à chaque étape et les scores de classification n'ont pas été passés entre les stades en tant qu'information contextuelle. D'autres travaux ont été menés sur le Deep learning dans les domaines suivants : la détection d'objet [49], segmentation d'objets [50] et la compréhension de scène [51].

Les matrices de covariance ont récemment été un choix populaire suivi en raison de leurs puissantes propriétés en tant que descripteur local et de leurs faibles exigences de calcul. Dans leur travail en 2010 [52], « suivi des objets par analyse structurelle des tenseurs » ; ils soulignent les similitudes des matrices de covariance au tenseur de structure bien connu. Ils montrent que la version généralisée du tenseur de structure est un descripteur puissant et qu'il peut être calculé en temps constant en exploitant les propriétés des images intégrales. Pour mesurer les similitudes entre plusieurs tenseurs de structure, ils décrivent un schéma d'approximation qui permet la comparaison dans un espace euclidien. Une telle approche est

également beaucoup plus efficace que les distances de collecteurs Riemanniennes courantes et exigeantes en termes de calculs. L'évaluation expérimentale prouve l'applicabilité pour la tâche de suivi d'objet démontrant des performances améliorées par rapport au suivi de covariance. Les méthodes récentes considérées comme state-of-the-art dans le domaine du suivi visuel utilisent des matrices de covariance de fonctionnalités de bas niveau comme puissant descripteur local. L'utilisation de matrices de covariance comme descripteur a d'abord été proposée par Porikli et al. [42] à des fins de suivi, mais aussi des applications pour la détection des piétons et la classification des textures sont apparues. Le principal avantage de la matrice de covariance par rapport aux autres descripteurs locaux est sa fusion implicite de différents types d'entités et de modalités de bas niveau et sa faible dimensionnalité. En outre, les matrices de covariance peuvent être calculées de manière efficace sur la base d'images intégrales.

**Travaux antérieurs sur l'alignement :** L'alignement automatique du visage a été un domaine actif de la vision par ordinateur. Au cours des dernières décennies, le domaine a subi des changements importants tant sur le plan de la méthodologie que des conditions d'exploitation. Les premières œuvres peuvent généralement être classées en modèles de formes actives (ASM) (Cootes et Taylor 1992, 1993), Modèles d'apparence active (AAM) (Cootes, Edwards et Taylor 1998, Gross et al., 2005, Matthews et Baker, 2004) et Contraintes. Modèles locaux (CLM) (Saragih, Lucey et Cohn 2011, Sangineto 2013, Baltrusaitis, Robinson et Morency 2012, Yu et al., 2013). L'émergence des méthodes de régression en cascade (CRM) (Cao et al 2013, Yang et Patras 2013, Xiong et De La Torre 2013, 2015, Tzimiropoulos 2015; Zhu et al. 2015; Yang et al. 2015; Deng et al. 2016) a apporté un gain de performance significatif en termes de rapidité et de précision (Kazemi et Josephine 2014, Ren et al., 2014) .

## A.2 Travaux antérieurs sur la détection avec Deep Learning

L'expression faciale et la reconnaissance des émotions avec des méthodes d'apprentissage en profondeur ont été décrites dans [53]. En particulier, Tang [54] a rapporté un CNN profond appris conjointement avec une sortie de machine à vecteur de support linéaire (SVM). Sa méthode a obtenu la première place sur les données publiques (de validation) et privées sur le Défi FER-2013 [55]. Liu et al. [53] ont proposé un cadre de reconnaissance d'expressions faciales avec des contraintes 3DCNN et des parties d'actions déformables afin de localiser conjointement des parties d'action faciale et d'apprendre des représentations basées sur des parties pour la reconnaissance d'expression. En outre, Liu et al. [53] incluaient les modèles Caffe CNN pré-formés pour extraire les caractéristiques au niveau de l'image. Enfin, le travail de Kahou et al. [53] est probablement le plus lié à notre méthode proposée. Leur méthode a

respectivement formé un CNN pour la vidéo et un Deep Restricted Boltzmann Machines (RBM) pour l'audio. Les caractéristiques du "sac de bouche" sont également extraites pour améliorer encore les performances. Deux grands ensembles de données : le jeu de données Toronto Face et le jeu de données Google ont été combinés pour former le réseau CNN. L'ensemble de données Google se trouve être l'ensemble de données fourni à FER-2013 et donc la méthode [56] partage une partie de l'ensemble de formation avec [53]. Malgré cette coïncidence, la stratégie d'apprentissage proposée diffère de [53] de façon significative.

### A.2.1 Détection des points de repère faciaux (Facial landmark detection)

Les méthodes classiques de détection des points de repère faciaux peuvent être divisées en deux catégories, à savoir la méthode basée sur la régression et la méthode d'ajustement des modèles. Une méthode [62] basée sur la régression estime explicitement les emplacements de repères par régression à l'aide de fonctions d'image. Par exemple, Valstar et al. [57] prédire l'emplacement d'un repère à partir d'un patch d'image local avec une régression du vecteur de support. Cao et al. [58] et Burgos Artizzu et al. [59] emploient la régression de fougère en cascade avec des caractéristiques de différence de pixel. Un certain nombre d'études [60], [61], [62] utilisent la forêt de régression aléatoire pour émettre des votes pour l'emplacement du point de repère basé sur le patch image local avec des caractéristiques similaires à Haar. La plupart de ces méthodes redefinissent une estimation initiale de l'emplacement du point de repère de façon itérative, la première supposition / initialisation est donc critique. En revanche, le modèle profond prend les pixels bruts en entrée sans nécessiter d'initialisation du repère facial. Fait important, cette méthode diffère en ce qu'on exploite des tâches auxiliaires pour faciliter l'apprentissage de la détection des points de repère.

Une méthode d'ajustement de gabarit construit des gabarits de visage pour ajuster les images d'entrée. Le modèle basé sur les parties a été récemment utilisé pour l'ajustement frontal. Zhu et Ramanan [63] montrent que la détection des visages, la détection des points de repère faciaux et l'estimation de la pose peuvent être traitées conjointement. La méthode [64] diffère en ce sens que l'apprentissage n'est pas limité à des tâches spécifiques, c'est-à-dire que le RCCTD est facilement extensible pour être entraîné avec des tâches auxiliaires supplémentaires. Spécifiquement, en dehors de la pose, on montre d'autres attributs faciaux tels que le genre et l'expression qui peuvent être utiles pour apprendre un détecteur de point de repère robuste. Une autre différence par rapport à [45] est que l'on apprend la représentation des caractéristiques à partir de pixels bruts plutôt que des HOG prédéfinis comme descripteur de visage.

**A.2.1.1 La détection des repères par apprentissage en profondeur :** Les méthodes formulent généralement l'alignement du visage comme un problème de régression et utilisent plusieurs modèles profonds pour localiser les points de repère de manière grossière, comme le CNN en cascade de Sun et al. [65]. Le CNN en cascade nécessite une pré-partition des faces en différentes parties, chacune étant traitée par des CNN profonds séparés. Les sorties résultantes sont ensuite moyennées et canalisées pour séparer les couches en cascade afin de traiter individuellement chaque point de repère facial. De même, Zhang et al. [66] utilise des réseaux d'auto-codeurs successifs pour effectuer un alignement grossier-à-fin. Au lieu de cela, le modèle [56] ne nécessite ni pré-partition des faces ni des réseaux en cascade, ce qui conduit à une réduction drastique de la complexité du modèle, tout en obtenant une précision comparable ou même supérieure. Cela ouvre la possibilité d'une application dans un scénario contraint par calcul, tel que les systèmes embarqués. En outre, l'utilisation d'une tâche auxiliaire peut réduire le problème de sur impression du modèle profond car le minimum local pour différentes tâches peut être situé à différents endroits. Une autre différence importante est que cette méthode effectue automatiquement l'extraction de caractéristiques dans l'ensemble de l'image du visage, au lieu des régions locales artisanales.

**A.2.1.2 Apprendre plusieurs tâches dans un réseau de neurones :** L'apprentissage multi-tâches (MTL) est le processus d'apprentissage de plusieurs tâches simultanément dans le but de bénéfice mutuel. C'est une vieille idée dans l'apprentissage automatique. Caruana [67] fournit un bon aperçu en se concentrant sur le réseau de neurones. Le modèle profond est bien adapté pour l'apprentissage de tâches multiples, car il permet l'apprentissage de fonctionnalités communes et l'inférence multi-objectif. L'apprentissage conjoint de tâches multiples s'est également révélé efficace dans de nombreux problèmes de vision par ordinateur. Cependant, les modèles profonds existants ne sont pas adaptés pour résoudre le problème car ils supposent des difficultés d'apprentissage et des taux de convergence similaires pour toutes les tâches. Par exemple, dans le travail de [68], l'algorithme apprend simultanément un détecteur de posture humaine et plusieurs détecteurs de la partie du corps. Cet algorithme optimise plusieurs tâches directement sans apprendre la corrélation des tâches. En outre, il utilise des coefficients de tâches prédéfinis dans le processus d'apprentissage itératif. L'application de cette méthode à notre problème entraîne des difficultés dans la convergence de l'apprentissage. On peut pallier cette lacune en introduisant des coefficients de tâches dynamiques dans le modèle profond. Cette nouvelle formulation généralise l'idée d'un arrêt précoce. L'arrêt précoce du réseau neuronal peut remonter au travail de Caruana [67], mais il est heuristique et limité aux

perceptrons multicouches superficiels. Le schéma n'est pas non plus évolutif pour une grande quantité de tâches. Différent du travail de [69], qui apprend la tâche prioritaire pour gérer des tâches aberrantes, le coefficient de tâche dynamique dans l'approche [64] est basé sur l'erreur de formation et de validation, et vise à coordonner des tâches de différents taux de convergence. On montre que le coefficient de tâche dynamique est important pour l'apprentissage conjoint de plusieurs objectifs dans un réseau convolutionnel profond.

### A.2.2. Désembrouillage (Deblurring) et les structures CNN pour le traitement d'image

Dans cette section, nous examinons brièvement les méthodes de désembrouillage [37] d'image et les structures CNN récentes pour le traitement d'image. Après le travail fondateur de Fergus et al. [69] et Shan et al. [70], de nombreuses méthodes de deblurring ont été proposées tant pour la qualité de la restauration que pour l'adaptation à différentes situations. Les images a priori naturelles ont été conçues pour supprimer les artefacts et améliorer la qualité. Ils comprennent la variation totale (TV), les primeurs d'image clairsemées, le gradient à queues lourdes avant, l'antécédent hyper-laplacien, le gradient de norme  $l_0$ , etc... les méthodes suivent le cadre grossier à fin. Les exceptions incluent les méthodes de domaine fréquentiel, qui ne sont applicables qu'à des situations limitées. Le deblurring des images bénéficie également de l'avancée récente de CNN profond. Sun et al. [78] ont utilisé le réseau pour prédire la direction du flou. Schuler et al. [74] ont empilé plusieurs CNN d'une manière grossière à fine pour simuler l'optimisation itérative. Chakrabarti [75] a prédit le noyau de déconvolution dans le domaine fréquentiel. Ces méthodes suivent le cadre traditionnel avec plusieurs parties remplacées à la version CNN. Su et al. [76] ont utilisé un réseau encodeur-décodeur avec des connexions de saut pour apprendre le deblurring vidéo. Nah et al. [77] ont formé un réseau profond à plusieurs échelles pour restaurer progressivement des images nettes. Ces méthodes de bout en bout utilisent des informations multi-échelles via différentes structures.

### A.2.3 Reconnaissance d'activité d'individus

Une des premières approches [50] pour analyser les activités d'un groupe ou d'une foule a été proposée par [79]. Ils présentent le contexte de la foule en reconnaissant l'activité exercée par chaque individu dans le groupe. Traditionnellement, les modèles graphiques avec des caractéristiques contextuelles ont été déployés rigoureusement vers l'analyse de groupe. Cependant, ces modèles avec des caractéristiques artisanales ont été surclassés par de nouvelles architectures de réseaux neuronaux profonds tels que [80, 81, 82]. Pour la reconnaissance de l'activité de groupe, la plupart de ces réseaux neuronaux profonds sont inspirés par [83] qui utilisent une cascade de réseaux neuronaux récurrents à plusieurs niveaux pour la

reconnaissance d'activités de groupe. Dans cette approche, les humains sont détectés et suivis pour former des tracklets multi-personnes. Ces tracklets ainsi que leurs caractéristiques visuelles profondes sont transmis aux RNN de niveau inférieur. L'objectif de ces RNN de niveau inférieur est de comprendre et de modéliser les actions des individus. Les RNN de niveau supérieur dans l'architecture se concentrent plutôt sur la compréhension de l'activité de groupe. Les actions individuelles et les prédictions d'activité de groupe sont effectuées en utilisant softmax de manière anticipée. Cependant, chaque méthode aborde un problème très différent dans le même cadre. Shu et al. [82] utilisent une architecture hiérarchique similaire mais l'approche diffère des travaux précédents en proposant une approche basée sur l'énergie qui fonctionne beaucoup mieux si la quantité de données est faible. En outre, cette approche explore également l'interaction humaine, mais de manière holistique par des caractéristiques convolutives extraites des deux humains. On propose une approche commune pour détecter les humains et prédire leurs actions. Des graphiques spatio-temporels ont été utilisés en vision par ordinateur pour diverses applications telles que la prédiction des mouvements humains ou l'apprentissage des activités humaines et des affordances d'objets. Les graphes spatio-temporels représentent dans ces travaux des relations spatio-temporelles entre articulations ou articulations et objets dans une vidéo. Certaines méthodes utilisent des caractéristiques artisanales ainsi que des modèles graphiques, des champs aléatoires conditionnels ou des forêts aléatoires pour les applications susmentionnées. Récemment, des réseaux de neurones ont été déployés pour résoudre des problèmes de graphes spatio-temporels ; par exemple, des réseaux profonds suivis d'inférence en utilisant un modèle graphique probabiliste pour reconnaître les actions dans un groupe. L'approche [78] s'appuie sur le travail [84] où un ensemble de RNN couplés sont utilisés pour représenter les graphes spatio-temporels.

### Partie B Travaux récents sur l'apprentissage profond pour la détection

#### B.1 Apprentissage en profondeur contextuel en plusieurs étapes pour la détection des piétons

Les classificateurs en cascade ont été largement utilisés dans la détection des piétons et ont connu un grand succès. Ces classificateurs sont formés séquentiellement sans optimisation conjointe. Dans cet article [56], on propose un nouveau modèle profond qui peut former conjointement des classificateurs à plusieurs degrés à travers plusieurs étapes de rétro-propagation. Il conserve la carte de score produite par un classificateur dans une région locale et l'utilise comme information contextuelle pour soutenir la décision à l'étape suivante. Grâce à une conception spécifique de la stratégie de formation, cette architecture profonde est capable de simuler les classificateurs en cascade en extrayant des échantillons durs pour former le réseau étape par étape. Chaque classificateur manipule des échantillons à un niveau de difficulté différent. Une pré-formation non supervisée et une formation supervisée par étapes spécifiquement conçue sont utilisées pour régulariser le problème d'optimisation. L'analyse théorique et les résultats expérimentaux montrent que la stratégie de formation permet d'éviter le surdimensionnement. Les résultats expérimentaux sur trois ensembles de données (Caltech, ETH et TUD-Bruxelles) montrent que l'approche de la [56] surpasse les approches de pointe.

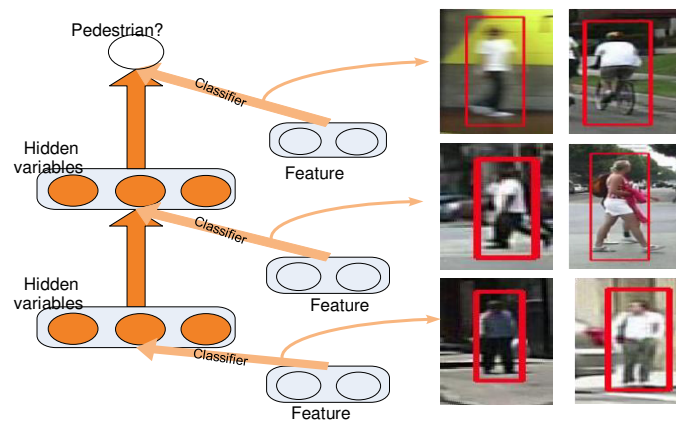
La contribution de ce travail est quadruple:

- Le groupe de classificateurs du modèle profond choisit des échantillons d'apprentissage étape par étape. L'entraînement est divisé en plusieurs étapes de rétro-propagation (BP). En raison de la conception de la procédure d'apprentissage, les gradients des paramètres de classification au stade actuel sont principalement influencés par les échantillons mal classifiés par les classificateurs aux étapes précédentes. A chaque étape BP, l'ensemble du modèle profond a été initialisé avec un bon point de départ appris à l'étape précédente et les classificateurs supplémentaires se concentrent sur les échantillons durs mal classifiés.
- Le groupe de classificateurs est optimisé conjointement. À chaque étape de la BP, les classificateurs des étapes précédentes travaillent conjointement avec le classificateur au stade actuel pour traiter les échantillons mal classifiés.
- Proposition d'une procédure de formation qui aide à éviter le surdimensionnement. Une pré-formation non supervisée et une formation supervisée par étapes spécialement conçue sont utilisées pour régulariser le problème d'optimisation. Il est différent de la BP standard, ce qui



optimise le réseau dans son ensemble. Avec la BP standard, un échantillon d'apprentissage facile peut influencer les classificateurs à n'importe quel stade, puisque ces étapes ne sont pas distinguées ou entraînées séparément. L'espace des paramètres de recherche est énorme et il est facile de le sur-ajuster.

- Les classificateurs en cascade existants ne transmettent qu'un seul score à l'étape suivante, tandis que le modèle profond conserve la carte de score dans une région locale et sert d'information contextuelle pour soutenir la décision à l'étape suivante.



*Fig. 2.1 Le modèle profond contextuel multi-étapes [23]*

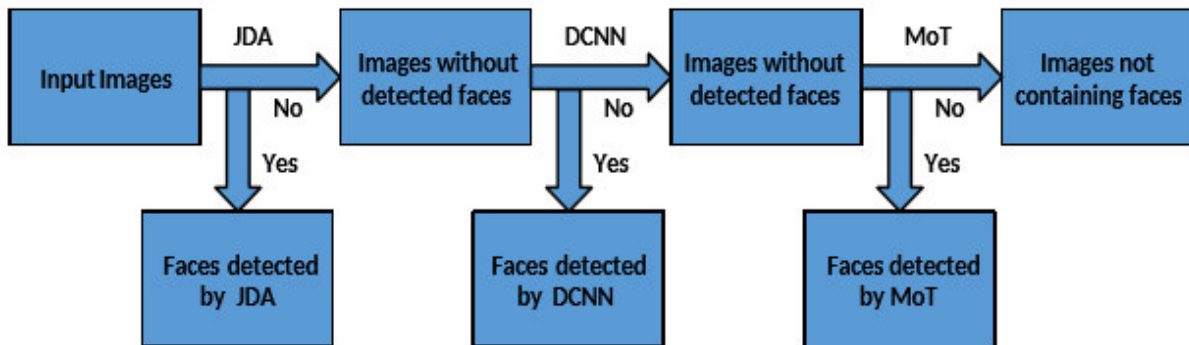
Cette architecture peut traiter des échantillons distribués complexes en utilisant plusieurs étapes de classificateurs. Dans chaque étape, un classificateur traite des échantillons à un niveau de difficulté différent.

Dans cet article, un nouveau modèle profond contextuel en plusieurs étapes est proposé ainsi que des stratégies de formation spécialement conçues pour la détection des piétons. Il simule les classificateurs en cascade. Les informations contextuelles provenant des pyramides des cartes de caractéristiques et des cartes de score se propagent à travers la cascade. Tous les classificateurs du modèle profond sont formés conjointement à travers plusieurs étapes de rétropropagation. Le surdimensionnement est évité grâce à la pré-formation non supervisée et à la formation supervisée multi-étapes.

## **B.2 Reconnaissance d'expression faciale statique basée sur l'image avec apprentissage en réseau profond multiple**

Les auteurs [55] se concentrent sur le sous-défi du jeu de données SFEW 2.0, où l'on cherche automatiquement à classer un ensemble d'images statiques en 7 émotions fondamentales. La méthode proposée contient un module de détection de visage basé sur l'ensemble des trois détecteurs de visage à la fine pointe de la technologie, suivi d'un module de classification avec

l'ensemble des réseaux de neurones convolutionnels profonds (CNN). Chaque modèle CNN est initialisé de manière aléatoire et pré-formé sur un jeu de données plus large fourni par le challenge de reconnaissance d'expression faciale (FER) 2013. Les modèles pré-formés sont ensuite finalisés sur l'ensemble d'apprentissage de SFEW 2.0. Pour combiner plusieurs modèles CNN, on présente deux schémas d'apprentissage des poids des réponses de réseau : en minimisant la perte de log de vraisemblance et en minimisant la perte de charnière (hinge loss).



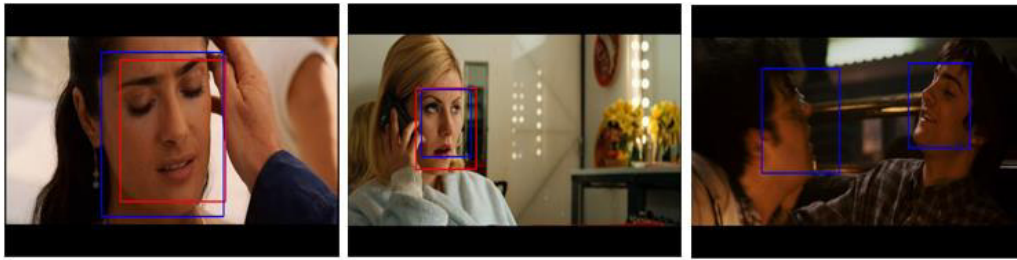
*Fig. 2. 2 Le schéma du système de détection de visage proposé sur SFEW 2.0 [60]*

JDA: the joint cascade detection and alignment; DCNN : Deep-CNN; MoT : Mixtures of Trees

La méthode proposée génère un résultat de pointe sur l'ensemble de données FER. Il atteint également 55,96% et 61,29% respectivement sur l'ensemble de validation et d'essai de SFEW 2.0, surpassant la base de référence de 35,96% et 39,13% avec des gains significatifs. Dans cet article, les auteurs se concentrent sur la tâche de la reconnaissance d'expression faciale statique basée sur l'image sur SFEW avec des CNN profonds. Les contributions principales peuvent être résumées comme suit: 1. Proposition d'une architecture CNN qui réalise d'excellentes performances de reconnaissance des émotions. 2. Proposition de perturbation des données et méthode de vote qui augmente considérablement les performances de reconnaissance de CNN. 3. Proposition de deux nouveaux cadres d'optimisation contraints pour apprendre automatiquement les poids d'ensemble de réseau en minimisant la perte de réponses de sortie de réseau.

Le prétraitement des visages s'avère être une étape cruciale pour une bonne performance de reconnaissance. Il aide à éliminer le bruit non-pertinent et unit tous les visages au même domaine. Le modèle de réseau profond est pré-formé sur FER, les faces détectées sur SFEW sont toutes redimensionnées à  $48 \times 48$  et sont transformées en niveaux de gris, ce qui est exactement la même chose que les données FER. Les images de visage provenant des ensembles de données SFEW et FER sont ensuite prétraitées avec une égalisation d'histogramme standard, suivie d'un ajustement linéaire du plan pour éliminer l'éclairage

déséquilibré. Enfin, les valeurs des pixels de l'image après ajustement du plan sont normalisées à une moyenne nulle et à un vecteur de variance unitaire.



*Fig. 2. 3 Exemples de détections de visage par JDA (rouge) et DCNN (bleu) [55].*

Dans cet article, une méthode de reconnaissance d'expression faciale est proposée. Elle est basée sur le modèle de réseau de neurones profond contextuel multi-étapes pour améliorer encore la performance. La méthode proposée atteint d'excellents résultats sur les ensembles de données FER et SFEW, indiquant le potentiel considérable de cette méthode de reconnaissance de l'expression faciale. Le cadre proposé atteint également la performance de pointe sur l'ensemble de données RPC.

### B.3 R-CNN rapide

Cet article [85] propose une méthode de réseau convolutif basé sur une région rapide (R-CNN rapide) pour la détection d'objets. Rapide R-CNN s'appuie sur des travaux antérieurs pour classer de manière efficace les propositions d'objets en utilisant des réseaux convolutionnels profonds. Comparé aux travaux précédents, Fast R-CNN utilise plusieurs innovations pour améliorer la vitesse d'entraînement et de test tout en augmentant la précision de la détection. Rapide R-CNN entraîne le réseau VGG16 très profond  $9 \times$  plus rapidement que R-CNN, est plus rapide de  $213 \times$  à l'heure du test, et atteint un plus haut mAP sur PASCAL VOC 2012. Comparé à SPPnet, Fast R-CNN entraîne VGG16  $3 \times$  plus vite, tests  $10 \times$  plus rapide, et est plus précis. R-CNN rapide est implémenté en Python et C ++ (en utilisant Caffe) et est disponible sous la licence MIT open-source sur <https://github.com/rbgirshick/fast-rcnn>. Cet article propose Fast R-CNN, une mise à jour propre et rapide de R-CNN et SPPnet. En particulier, les propositions d'objets clairsemés qui semblent améliorer la qualité du détecteur. Ce problème était trop coûteux (temps) pour le test dans le passé, mais devient pratique avec Fast R-CNN. Bien sûr, il peut exister des techniques encore non découvertes qui permettent aux toolbox de fonctionner aussi bien. Ces méthodes, si elles sont développées, peuvent aider à accélérer la détection d'objets.

### **B.4 Apprendre la représentation en profondeur pour l'alignement du visage avec les attributs auxiliaires**

Dans cette étude [64], les auteurs montrent que la détection de repères ou la tâche d'alignement des faces n'est pas un problème unique et indépendant. Au lieu de cela, sa robustesse peut être grandement améliorée avec des informations auxiliaires. Spécifiquement, ils optimisent conjointement la détection de repères avec la reconnaissance d'attributs faciaux hétérogènes mais subtilement corrélés, tels que les attributs de genre, d'expression et d'apparence. Ceci est non trivial puisque différentes tâches d'inférence d'attributs ont des difficultés d'apprentissage et des taux de convergence différents. Pour résoudre ce problème, un nouveau modèle profond contraint par tâches est formulé, qui non seulement apprend la corrélation inter-tâches, mais utilise également des coefficients de tâches dynamiques pour faciliter la convergence d'optimisation lors de l'apprentissage de multiples tâches complexes. Des évaluations approfondies montrent que l'apprentissage par contrainte de tâche proposé (i) surpasse les méthodes d'alignement de visage existantes, en particulier pour traiter les visages avec une occlusion sévère et une variation de pose, et (ii) réduit considérablement la complexité du modèle par rapport aux méthodes les plus récentes basées sur le modèle profond en cascade.

Les changements sur le visage sont souvent régis par les mêmes règles déterminées par la structure faciale intrinsèque. Par exemple, lorsqu'un enfant sourit, sa bouche est largement ouverte (la deuxième image de la figure 2.4 (a)). La découverte et l'exploitation efficaces d'un tel attribut facial intrinsèquement corrélé aideraient à détecter plus précisément les coins de la bouche. De plus, la distance inter-oculaire est plus petite dans les visages avec une rotation en lacet importante (la première image de la figure 2.4 (a)). De telles informations de pose peuvent être exploitées comme une source d'information supplémentaire pour contraindre l'espace de solution de l'estimation du point de repère. En effet, les espaces d'entrée et de solution d'alignement de face peuvent être efficacement divisés en fonction des attributs de visage auxiliaires. Dans une petite expérience, on fait la moyenne d'un ensemble d'images de visage selon différents attributs, comme le montre la figure 2.4 (b)), où les visages frontaux et souriants montrent les coins de la bouche, alors qu'il n'y a pas de détails spécifiques de l'ensemble des données. Compte tenu de la riche information auxiliaire, le traitement de la détection du point de repère facial isolé est contre-productif. Cette étude vise à étudier la possibilité d'optimiser la détection de repères faciaux (la tâche principale) en tirant parti des informations auxiliaires des tâches d'inférence d'attributs. Les tâches auxiliaires potentielles comprennent l'estimation de la posture de la tête, la classification par sexe, l'estimation de l'âge et la reconnaissance de

l'expression faciale ou l'inférence d'attributs faciaux. Étant donné les multiples tâches, le réseau convolutionnel profond semble être un choix de modèle viable puisqu'il permet l'apprentissage de caractéristiques communes et l'inférence multi-objective. Typiquement, on peut formuler une fonction de coût qui englobe toutes les tâches et utiliser la fonction de coût dans l'apprentissage de rétro-propagation du réseau. Ce schéma d'apprentissage multi-tâches classique est difficile. Il existe plusieurs raisons : Premièrement, les différentes tâches d'alignement du visage et d'inférence d'attributs sont intrinsèquement différentes dans les difficultés d'apprentissage. Par exemple, apprendre à identifier l'attribut «porter des lunettes» est plus facile que de déterminer si l'on sourit. Deuxièmement, nous avons rarement des tâches auxiliaires avec un nombre similaire de cas positifs / négatifs. Par exemple, la classification homme / femme bénéficie d'échantillons plus équilibrés que la reconnaissance d'expression faciale. En conséquence, différentes tâches ont une convergence différente.

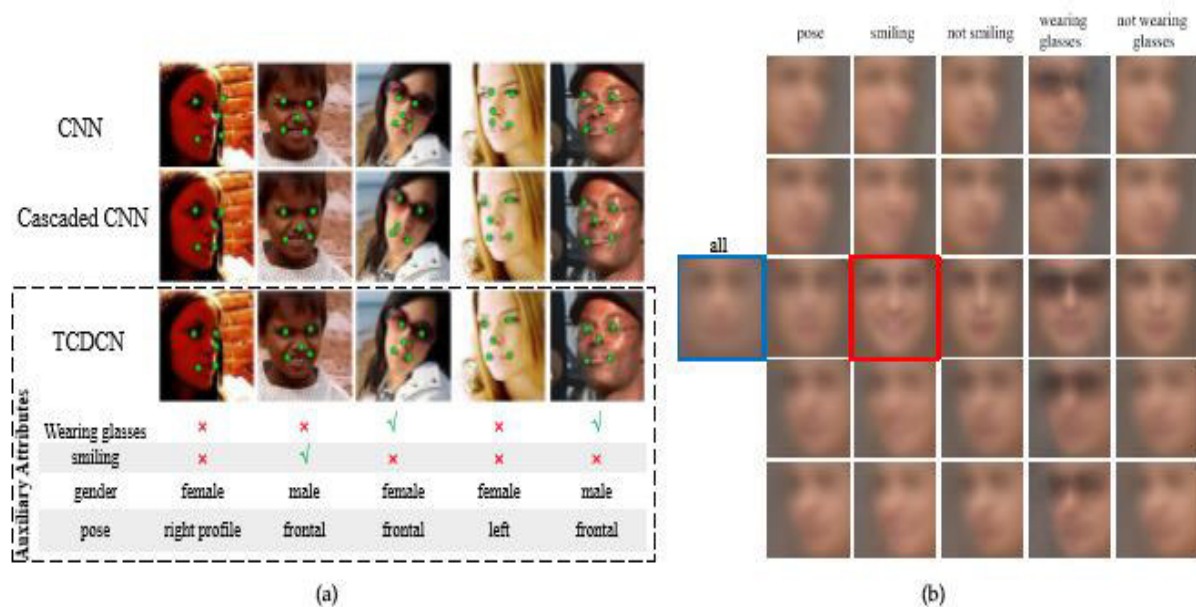


Fig. 2. 4 (a) Exemples de détection de repères faciaux par un seul CNN conventionnel, le CNN en cascade [65], et le réseau convolutif profond à contraintes (TCDCN).

(b) Images de visage moyennes avec différents attributs.

Une détection plus précise peut être obtenue en optimisant la tâche de détection conjointement avec les tâches connexes / auxiliaires. L'image en rectangle bleu est moyennée sur l'ensemble des visages d'entraînement, tandis que celle en rouge provient des visages souriants avec pose frontale. Il indique que l'espace d'entrée et de solution peut être efficacement divisé en sous-ensembles, qui sont dans des distributions différentes. Cela diminue la difficulté d'apprentissage.

Dans de nombreux cas, nous observons que l'apprentissage conjoint avec une tâche auxiliaire spécifique améliore la convergence de la détection des points de repère au début de la procédure d'entraînement, mais devient inefficace lorsque l'entraînement aux tâches auxiliaires rencontre des minima locaux ou une sur-adaptation. Poursuivre la formation avec toutes les tâches compromet la convergence du réseau, ce qui conduit à une mauvaise performance de détection des points de repère. Cette étude est la première tentative de démontrer que l'alignement des faces peut être optimisé conjointement avec l'inférence d'attributs auxiliaires hétérogènes mais subtilement corrélés. On montre que le signal de supervision des tâches auxiliaires peut être rétropropagé conjointement avec celui de l'alignement du visage pour apprendre les régularités sous-jacentes de la représentation faciale. Néanmoins, l'apprentissage est non-trivial en raison des différentes natures et des taux de convergence des différentes tâches. Notre contribution clé est un réseau de convolution profond (TCDCN) nouvellement proposé et contraint aux tâches, avec une nouvelle fonction objective pour relever les défis susmentionnés.

Au lieu d'apprendre la détection des repères faciaux isolément, nous avons montré que la détection de repères plus robustes peut être obtenue grâce à l'apprentissage conjoint avec des tâches auxiliaires hétérogènes mais subtilement corrélées, comme l'apparence, l'expression, la démographie et la pose de la tête. Le TCDCN proposé permet de rétro-propager les erreurs des tâches auxiliaires dans des couches cachées profondes afin de construire une représentation partagée pertinente pour la tâche principale. Nous montrons aussi qu'en apprenant un coefficient de tâche dynamique, nous pouvons utiliser les tâches auxiliaires d'une manière plus efficace. Grâce à l'apprentissage avec les attributs auxiliaires, le modèle proposé est plus robuste pour les visages avec des occlusions sévères et des variations de pose importantes par rapport aux méthodes existantes. Nous avons observé qu'un modèle profond n'a pas besoin d'être cascadié [65] pour atteindre la meilleure performance. Le CNN plus léger permet des performances en temps réel sans l'utilisation de GPU ou de techniques de calcul parallèle. Les travaux futurs exploreront l'apprentissage en profondeur avec des informations auxiliaires pour d'autres problèmes de vision.

### **B.5 Analyse de repères faciaux de force brute avec un classificateur de 140 000 voies**

Dans ce travail [92], une approche simple de l'alignement visuel est proposée, en mettant l'accent sur la tâche d'illustration de l'estimation du point de repère facial. Alors que la plupart des travaux antérieurs traitant cela comme un problème de régression. Il est formulé ici plutôt comme une tâche discrète de classification K-way, où un classificateur est formé pour retourner un des K alignements discrets. Un avantage crucial d'un classificateur est la possibilité de

rapporter un (softmax) distribution sur les alignements putatifs. On démontre que cette distribution est une représentation riche qui peut être marginalisée (pour générer des estimations d'incertitude sur des groupes de repères) et conditionnée (pour intégrer le contexte topdown, fourni par des contraintes temporelles dans un flux vidéo ou un utilisateur interactif). De telles capacités sont difficiles à intégrer dans des approches classiques basées sur la régression. On étudie la performance en fonction du nombre de classes  $K$ , y compris l'extrême «classe exemplaire» où  $K$  est égal au nombre d'exemples d'entraînement (140K dans notre cadre). Peut-être étonnamment, on montre que les classificateurs peuvent encore être appris dans ce cadre. Comparé aux travaux antérieurs en classification, notre  $K$  est d'une taille sans précédent, incluant de nombreuses classes «fines» qui sont très similaires. Ces problèmes sont abordés en utilisant une fonction de perte multi-étiquettes qui permet de partager de façon non uniforme des exemples d'apprentissage entre classes distinctes. Une analyse expérimentale complète de la méthode est effectuée sur des benchmarks standard, démontrant des résultats de pointe pour l'alignement du visage dans les vidéos. Bien qu'énormément puissants, ces réseaux réduisent le problème à un problème de régression non linéaire. Une telle approche peut souffrir lorsque le problème inverse est intrinsèquement conditionné, ce qui implique que plusieurs interprétations / solutions peuvent être également valables. Dans ce cas, il peut être plus naturel de prédire une distribution plutôt qu'une interprétation. Par exemple, lors de l'application de tels réseaux pour prédire l'emplacement des points de repère dans un visage fortement occlus, il est probablement utile de rapporter plusieurs possibilités.

**B.5.1 L'Approche :** Compte tenu de la motivation ci-dessus, on propose une approche simple mais quelque peu radicale de l'alignement : discrétiser toutes les prédictions possibles en  $K$  classes discrètes, et traiter le problème comme une classification  $K$ -way à grande échelle. Puisque les réseaux sont facilement formés pour reporter les distributions (softmax) sur les classes, cette approche produit des estimations d'incertitude sur les interprétations possibles d'une image. Il est important de noter qu'une telle approche nécessite également d'étendre les réseaux de classification à un nombre massif de classes (par centaines de milliers ou millions), ce qui pose plusieurs défis théoriques et pratiques.



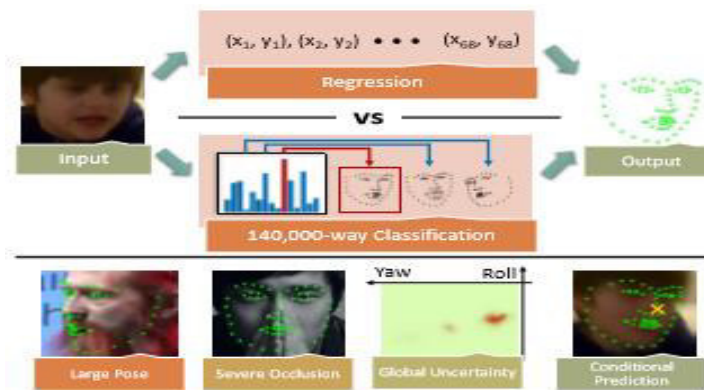


Fig. 2. 5 L'alignement des faces est un problème de régression, mais résolu par une classification à grande échelle [97].

Comme indiqué dans la rangée du bas, ce modèle est capable de gérer des occlusions sévères et de grandes variations de poses et de fournir une estimation de l'incertitude globale. De plus, une telle représentation d'incertitude peut être utilisée pour produire une prédiction conditionnelle dans une configuration interactive.

**B.5.2 Évaluation :** Le réseau de classification K-way pour la tâche d'alignement du visage dans les séquences vidéo est évalué, en se concentrant sur le récent benchmark de 300 vidéos dans la nature (300VW) (Shen et al., 2015, Chrysos et al., 2017). Pour explorer l'impact du grand K, on utilise un ensemble d'apprentissage de 140 000 images constitué d'images réelles et d'images synthétiques disponibles publiquement, obtenues en déformant le véritable ensemble d'apprentissage (Zhu et al., 2016). la précision de l'état de l'art en termes d'alignement grossier est démontrée, tel que mesurée par le nombre de cadres où les points de repère sont localisés dans une tolérance grossière. Ceci est quelque peu attendu car les sorties sont discrétisées par conception. Pour améliorer la précision pour les petites tolérances, on ajoute une étape de régression post-traitement qui produit des résultats de pointe à travers tous les seuils de tolérance.



Fig. 2. 6 Les contraintes de calcul du modèle au moment du test [92].

Orange représente toutes les couches avant la dernière couche entièrement connectée (extraction de caractéristiques) et le bleu représente la dernière couche entièrement connectée (classifieur).

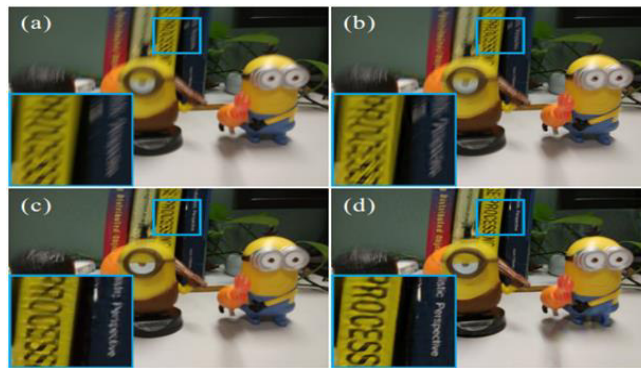


Avec un nombre croissant de classes, le temps d'exécution augmente à peine tandis que la consommation de mémoire augmente linéairement. Bien que l'alignement visuel soit naturellement considéré comme un problème de régression, il est reformulé comme une tâche de classification. Un avantage important est que les réseaux de classification softmax rapportent naturellement des distributions, qui peuvent être utilisées pour raisonner sur la confiance, l'incertitude et la condition sur des preuves externes (fournies par des contraintes contextuelles ou un utilisateur interactif). Malgré sa simplicité, une telle méthode est considérablement plus robuste que les travaux antérieurs, produisant une précision de pointe dans des scénarios difficiles. On se concentre sur la tâche d'illustration de l'alignement du point de repère facial, démontrant une performance robuste à travers une large variation de pose, des occlusions sévères et un éclairage extrême.

### **B.6 Réseau récurrent à l'échelle pour le désembrouillage d'image en profondeur**

Dans cet article [37], nous étudions cette stratégie et proposons un réseau Scale-Recurrent (SRN-DeblurNet) pour cette tâche de déconstruction. Comparé aux nombreuses approches récentes basées sur l'apprentissage [77], il présente une structure de réseau plus simple, un plus petit nombre de paramètres et est plus facile à former. La méthode est évaluée sur des ensembles de données de deblurring à grande échelle avec un mouvement complexe. Les résultats montrent que la méthode peut produire des résultats de meilleure qualité que l'état de l'art, à la fois quantitativement et qualitativement

Parmi eux, Nah et al. [77] ont atteint des résultats de pointe en utilisant un réseau de neurones convolutionnels multi-échelles (CNN). Leur méthode commence à partir d'une échelle très grossière de l'image floue, et récupère progressivement l'image latente à des résolutions plus élevées jusqu'à ce que la pleine résolution soit atteinte. Ce cadre suit le mécanisme multi-échelle dans les méthodes traditionnelles, où les pipelines grossiers à fins sont communs lors de la manipulation de grands noyaux de flou. Dans cet article, les auteurs explorent une structure de réseau plus efficace pour le deblurring d'image multi-échelle. Ils proposent le nouveau réseau récurrent à l'échelle (SRN, scale-recurrent network), qui traite et aborde deux problèmes importants et généraux dans les systèmes de débrouillage à base de CNN.



*Fig. 2. 7 Un exemple réel*

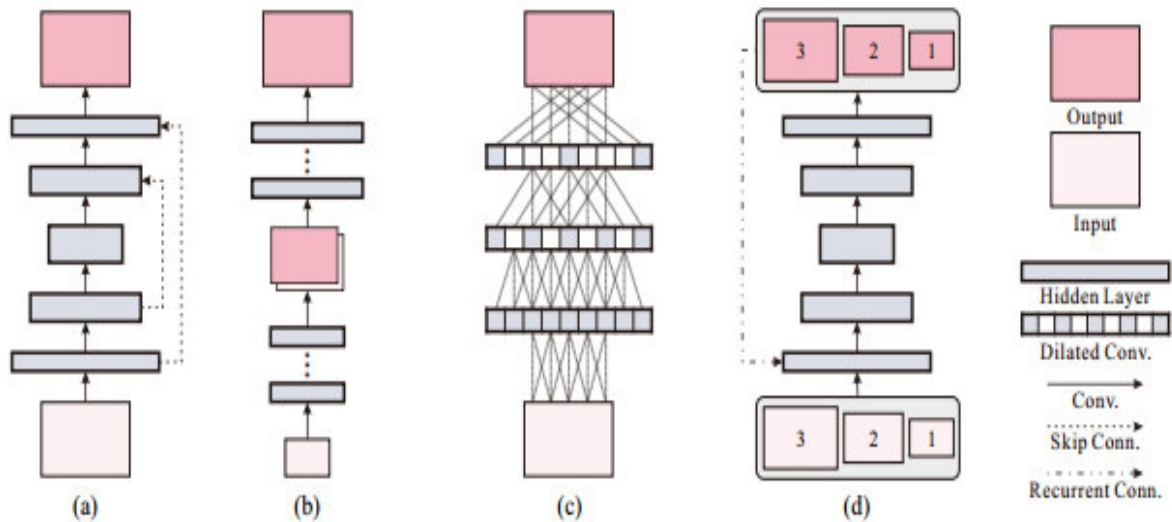
(a) image floue. (b) Résultat de Sun et al. [72]. (c) Résultat de Nah et al. [77]. (d) Résultat [37]

**Structure échelle-récurrent** : Dans ce travail, on propose de partager les pondérations de réseau à travers les échelles afin de réduire considérablement les difficultés d'entraînement et d'introduire des avantages évidents de stabilité. Les avantages sont doubles. Premièrement, il réduit considérablement le nombre de paramètres pouvant être entraînés. Même avec les mêmes données d'apprentissage, l'exploitation récurrente des poids partagés fonctionne de la même manière que l'utilisation multiple des données pour apprendre les paramètres, ce qui équivaut en réalité à une augmentation des données concernant les échelles. Deuxièmement, la structure proposée peut intégrer des modules récurrents, dont l'état caché capte implicitement des informations utiles et permet de restaurer les avantages à travers les échelles.

**Codeur-décodeur ResBlock Network** : Dans cet article, on montre que l'application directe d'une structure codeur-décodeur existante ne peut produire des résultats optimaux. Le réseau de décodeur de codeur ResBlock, au contraire, amplifie le mérite de diverses structures CNN et donne la faisabilité de la formation. Il produit également un très grand champ réceptif, ce qui est d'une importance vitale pour le déblayage à grande vitesse.

**CNNs pour traitement d'image** : Différent des tâches de classification, les réseaux pour le traitement d'images nécessitent une conception spéciale. Comme l'une des premières méthodes, SRCNN [49] a utilisé 3 couches de convolution plates (avec la même taille de carte de caractéristiques) pour la super-résolution. U-net [86] (comme le montre la figure 2.8 (a)), également appelé réseaux encodeur-décodeur [87], a permis d'améliorer considérablement la régression et est largement utilisé dans les travaux récents de FlowNet [88], déblocage vidéo, super-résolution vidéo, synthèse de trame, etc. CNN multi-échelle [77] et réseau de raffinement en cascade (CRN) [89] (figure 2.8 (b)) en affinant progressivement la production à partir d'une très petite échelle. Ils réussissent respectivement dans le débridage d'image et la synthèse d'image. La figure 2.8 (c) montre une structure différente [90] qui utilise des couches de

convolution dilatées avec des taux croissants, ce qui se rapproche de l'augmentation des tailles de noyau.



**Fig. 2. 8** Différents CNN pour le traitement d'image.

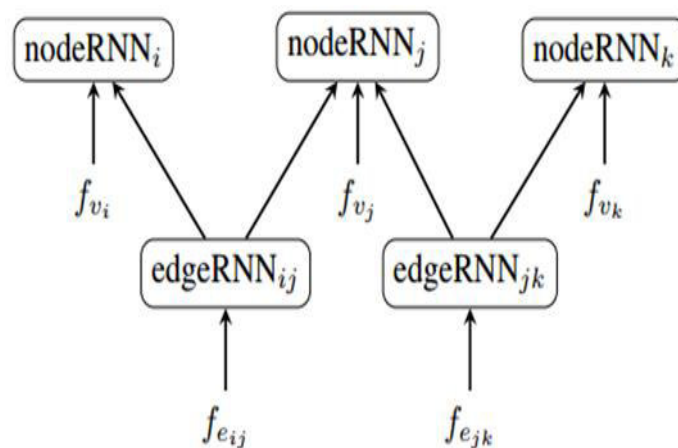
(a) réseau U-Net [86] ou codeur-décodeur [87]. (b) Réseau de raffinement à plusieurs échelles [77] ou en cascade [89]. (c) Réseau convolutif dilaté [90].(d) Notre réseau à échelle récurrente proposé (SRN) [68].

Dans cet article [68], on explique quelle est la structure de réseau appropriée pour utiliser le schéma "grossier à fin" dans le deblurring d'image. On a également proposé un réseau récurrent à l'échelle, ainsi qu'une structure ResBlocks codeur-décodeur dans chaque échelle. Cette nouvelle structure de réseau a moins de paramètres que les précédentes deblurring multi-échelle et est plus facile à former. Les résultats générés par la méthode sont de pointe, à la fois qualitativement et quantitativement. On pense que ce réseau récurrent peut être appliqué à d'autres tâches de traitement d'images.

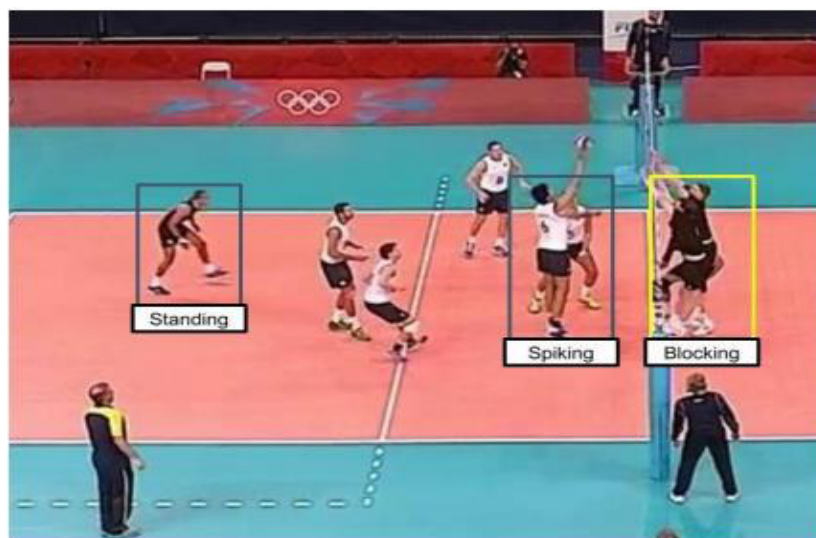
### B.7 Réseau Neuronal Récurrent Structurel (SRNN) pour l'Analyse d'Activité de Groupe

Un groupe de personnes peut être analysé à différents niveaux sémantiques tels que les actions individuelles, leurs interactions et l'activité de l'ensemble du groupe. Dans cet article [78], on propose un réseau neuronal récurrent structurel (SRNN) qui utilise une série de RNN interconnectés pour capturer conjointement les actions des individus, leurs interactions, ainsi que l'activité du groupe. Alors que les réseaux neuronaux récurrents structurels précédents supposaient que le nombre de nœuds et de bords est constant, les auteurs utilisent une couche de mise en commun de la grille pour tenir compte du fait que le nombre des individus dans un groupe peut varier. Deux variantes du réseau neuronal récurrent structurel sont évaluées sur le jeu de données Volleyball.

L'objectif principal de l'article est d'exploiter de telles interactions au sein d'un groupe pour améliorer la reconnaissance de l'activité de groupe ainsi que les actions individuelles. À cette fin, on s'appuie sur le réseau neuronal récurrent structurel (SRNN) [84] récemment proposé qui a la capacité unique de capturer des interactions sous la forme d'informations contextuelles à l'aide d'un ensemble de RNN interconnectés. Alors que dans [84], le nombre de noeuds et les arêtes sont constants et que le nombre de RNN est constant, l'approche est étendue pour gérer un nombre variable de noeuds et d'arêtes tel qu'il est requis pour analyser les activités de groupe.



*Fig. 2. 9 Réseau anticipé d'un RNN structurel (SRNN) formés en ce qui concerne les étiquettes de nœud [78].*



*Fig. 2. 10 Un cadre étiqueté comme activité de groupe «Spike gauche» et encadrés entourant chaque joueur d'équipe sont annotés dans l'ensemble de données avec des actions individuelles [78].*

Dans ce travail, deux variantes de réseaux neuronaux récurrents structuraux (SRNN) sont proposées pour reconnaître les actions des individus ainsi que l'activité de l'ensemble du groupe.

**Tableau 2. 1** (à gauche) Comparaison des approches SRNN proposées avec Hierarchy LSTM V3 utilisant Alexnet ou VGG 16 comme CNN [78]. (à droite) Comparaison des entités de périphérie à l'aide de SRNN-MaxNode.

Feature	Method	Group Activity Accuracy	Individual Action Recognition Accuracy
Alexnet	H. LSTM V3 - (1 group)	74.01%	75.96%
	SRNN-MaxNode - (1 group)	<b>74.39%</b>	<b>76.65%</b>
	SRNN-MaxEdge - (1 group)	68.39%	76.03%
VGG 16	H. LSTM V3 - (1 group)	70.34%	75.30%
	SRNN-MaxNode - (1 group)	71.20%	74.85%
	SRNN-MaxEdge - (1 group)	68.29%	75.96%
Alexnet	H. LSTM V3 - (2 groups)	83.12%	75.96%
	SRNN-MaxNode - (2 groups)	<b>83.47%</b>	<b>76.65%</b>
	SRNN-MaxEdge - (2 groups)	79.86%	76.03%
VGG 16	H. LSTM V3 - (2 groups)	81.34%	75.30%
	SRNN-MaxNode - (2 groups)	82.86%	74.85%
	SRNN-MaxEdge - (2 groups)	79.92%	75.96%

Edge feature	Group Activity Accuracy	Individual Action Recognition Accuracy
$f_{e_{ij}}^t$ (1 group)	74.39%	<b>76.65%</b>
$(f_{e_{ij}}^t, f_{v_i}^t, f_{v_j}^t)$ (1 group)	<b>74.48%</b>	75.89%
$f_{e_{ij}}^t$ (2 groups)	<b>83.47%</b>	<b>76.65%</b>
$(f_{e_{ij}}^t, f_{v_i}^t, f_{v_j}^t)$ (2 groups)	83.27%	75.89%

L'avantage de l'approche SRNN est qu'elle modélise explicitement les relations entre individus et que tous les RNN peuvent être entraînés ensemble en utilisant une seule fonction de perte. Les modèles ont été évalués sur le jeu de données Volleyball et montré que le modèle SRNN surpasse les LSTM (LSTM : Long Short-Term Memory) hiérarchiques.

### Conclusion

Dans ce chapitre, nous avons concentré sur les travaux antérieurs sur la détection et les travaux récents sur l'apprentissage profond pour la détection , Nous avons identifié les étapes de la technologie de détection visage en utilisant l'apprentissage en profondeur et nous avons également souligné le rôle des réseaux de neurones dans la détection du visage.

Dans le chapitre 4, nous examinerons l'identification des réseaux de neurones et leur rôle dans la performance de la détection visage

**Chapitre 3 Etude et Conception de la Détection**

### Introduction

La détection de visage est une des tâches visuelles que les humains peuvent accomplir sans effort. Cependant, en termes de vision par ordinateur, cette tâche n'est pas facile. Une formulation générale du problème peut être définie comme suit: étant donnée une image statique (still image) ou une vidéo (video image), on veut détecter un nombre inconnu (s'il existe) de visages. La solution à ce problème implique la segmentation, l'extraction, et la vérification des visages et probablement des traits faciaux à partir d'un arrière plan non contrôlé. Comme processeur visuel d'entrée, un système de détection de visage devrait également pouvoir réaliser la tâche indépendamment de l'illumination, de l'orientation, et de la distance de la camera. Ce chapitre vise à fournir un aperçu de la recherche contemporaine en détection de visage et d'une manière structurée. Les auteurs dans [94] ont conduit une étude détaillée sur la recherche en reconnaissance de visage. Dans leur étude, plusieurs aspects, y compris la segmentation et l'extraction des traits, liées à la reconnaissance de visage ont été passés en revue. Une des conclusions de [94] était que le problème de détection de visage a suscité étonnamment peu d'attention. Ceci a certainement changé au cours des dernières années comme on le montrera dans cet aperçu.

### 3.1 Méthodes Détection de visage

Dans cette partie du chapitre nous allons étudier puis concevoir différentes méthodes classiques de détection. A travers celà, nous choisirons la meilleure qui répond à nos besoins et nous la comparerons à la détection par le CNN. L'étude et la conception de cette dernière fera objet de notre quatrième chapitre. Nous avons consacré deux chapitres : trois et quatre pour trouver une approche optimale de détection qui répond à notre objectif. Les différentes méthodes étudiées et conçues sont les suivantes :

1. *Méthode de détection MSER features ( Maximally stable extremal regions features )*
2. *Méthode de Edge detection*
3. *Détection de visage à l'aide de réseaux neuronaux et de fonctionnalités de Gabor*
4. *Méthode de Viola-Jones*



**3.2 Méthode de détection MSER Features** (*Maximally Stable Extremal Regions Features* )

Dans la vision par ordinateur, les régions extrêmes extrêmement stables (MSER) sont utilisées comme méthode de détection des tâches sur les images. Cette technique a été proposée par Matas et al. [95] pour trouver des correspondances entre des éléments d'image à partir de deux images avec différents points de vue. Cette méthode d'extraction d'un nombre complet d'éléments d'image correspondants contribue à l'appariement à base large et a conduit à de meilleurs algorithmes de correspondance stéréo et de reconnaissance d'objet.

**3.2.1 Modèle Mathématique :** Les régions extrêmes extrêmement stables ont été définies dans [96] comme étant noires (ou blanches) les zones d'une image seuillée qui ne varient que de manière insignifiante sous le seuil changements. Formellement, une région binarisée  $Q(t)$  (où  $t$  indique son niveau de seuil) est considéré comme MSER si la fonction de taux de croissance  $q(t)$  atteint un minimum ;  $q(t)$  local définie par la dérivée de la zone de la région sur les valeurs de seuil :

$$q(t) = \frac{\frac{d}{dt} \|Q(t)\|}{\|Q(t)\|} \tag{Equ. 3.1}$$

En pratique, l'équation (Equ. 3.1) est substituée par l'une de ses approximations discrètes:

$$q(t_j) = \frac{\|Q(t_j) - Q(t_j - 1)\|}{\|Q(t_j)\|} \quad \text{ou} \quad q(t_j) = \frac{\|Q(t_j + 1) - Q(t_j - 1)\|}{\|Q(t_j)\|} \tag{Equ. 3.2}$$

où la différence  $t_j - t_{j-1}$  définit l'incrément de seuil  $\Delta t$ . Les formules ci-dessus s'appliquent à la fois aux MSER sombres et brisés (pour ce dernier, les images doivent être inversées). Un certain nombre d'autres paramètres est utilisé pour contrôler la stabilité de la détection MSER et pour réduire les effets d'emboîtement (provoqués par exemple par des flous).

**3.2.2 Avantages de MSER :** puisque les régions sont définies exclusivement par la fonction d'intensité dans la région et la frontière extérieure, cela conduit à de nombreuses caractéristiques clés des régions qui les rendent utiles. Sur une large gamme de seuils, la binarisation locale est stable dans certaines régions et possède les propriétés énumérées ci-dessous :

- *Invariance à la transformation affine des intensités d'image*
- *Covariance vers la contiguïté (continue) transformation  $T : D \rightarrow D$  sur le domaine de l'image*
- *Stabilité: seules les régions dont le support est presque identique sur une plage de seuils sont sélectionnées.*

- *Détection multi-échelle sans aucun lissage impliqué, à la fois fine et grande structure est détectée.*

Notez, cependant, que la détection des MSER dans une pyramide d'échelle améliore la répétabilité, et le nombre de correspondances à travers les changements d'échelle [97]. L'ensemble de toutes les régions extrémales peut être énuméré [98].

3.2.3 Comparaison avec d'autres détecteurs de région : Dans Mikolajczyk et al., [99] six détecteurs de région sont étudiés (Harris-affine, Hessian-affine, MSER, régions de bord, extrema d'intensité et régions saillantes). Un résumé des performances MSER par rapport aux cinq autres suit :

- **La densité de la région** - *en comparaison avec les autres MSER offre le plus de variété en détectant environ 2600 régions pour une scène de flou texturé et 230 pour une scène à lumière changée et la variété est généralement considérée comme bonne. MSER avait également une répétabilité de 92% pour ce test.*
- **Taille de la région** - *MSER avait tendance à détecter de nombreuses petites régions, par opposition à de grandes régions qui sont plus susceptibles d'être occluses ou de ne pas couvrir une partie plane de la scène. Bien que les grandes régions puissent être légèrement plus faciles à faire correspondre.*
- **Modification du point de vue** - *MSER surpasse les cinq autres détecteurs de région dans les images originales et celles avec des motifs de texture répétés.*
- **Changement d'échelle** - *Après le détecteur Hessian-affine, MSER arrive en deuxième position sous un changement d'échelle et une rotation dans le plan.*
- **Flou** - *MSER s'est avéré être le plus sensible à ce type de changement d'image, seul domaine dans lequel ce type de détection fait défaut.*

*Notez cependant que cette évaluation n'a pas utilisé la détection multi-résolution, qui a été montré pour améliorer la répétabilité sous le flou [97].*

- **Changement de lumière** - *MSER a montré le score de répétabilité le plus élevé pour ce type de scène, tous les autres ayant une bonne robustesse.*

*MSER a régulièrement obtenu le score le plus élevé à travers de nombreux tests, prouvant qu'il s'agissait d'un détecteur de région fiable [99].*

**3.2.4 Extensions et adaptations :** L'algorithme MSER a été adapté pour colorer les images, en remplaçant le seuillage de la fonction d'intensité par un regroupement agglomératif, basé sur des gradients de couleur [100].

L'algorithme MSER peut être utilisé pour détecter des régions basées sur la couleur plutôt que sur l'intensité. Ceci est fait par Chavez en créant une fonction d'intensité pour le rouge, le vert et le bleu dans l'espace couleur HSV. L'algorithme MSER est ensuite exécuté cinq fois; sur les intensités pseudo à trois couleurs, puis sur les intensités d'échelle de gris en utilisant les fonctions standard MSER + et MSER- [101].

L'algorithme MSER peut être utilisé pour suivre les objets de couleur, en effectuant une détection MSER sur la distance de Mahalanobis à une distribution de couleur [96].

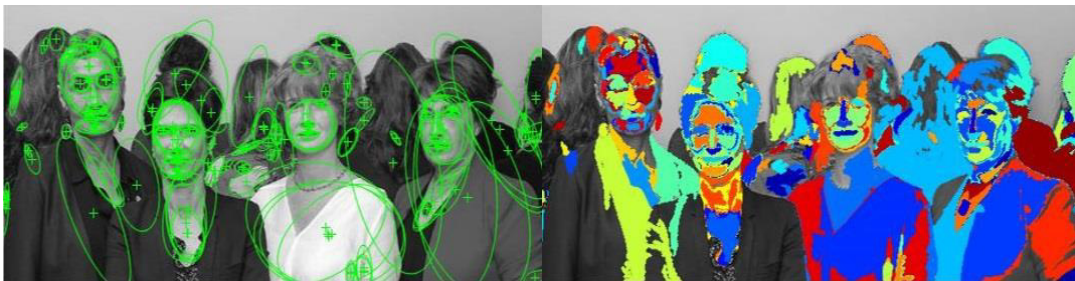
En détectant les MSER dans plusieurs résolutions, la robustesse au flou et le changement d'échelle peuvent être améliorés [97].

**3.2.5 Implémentation de la méthode de détection MSER et résultats :** L'algorithme original de Matas et al. [97] est :  $O(n \log(\log(n)))$  dans le nombre  $n$  de pixels. Il procède en triant d'abord les pixels par intensité. Cela prend  $On$  heure. Après le tri, les pixels sont marqués dans l'image et la liste des composants connectés croissants et fusionnés et leurs zones est conservée à l'aide de l'algorithme union-find. Cela prend  $O(n \log(\log(n)))$  temps. En pratique, ces étapes sont très rapides. Au cours de ce processus, la surface de chaque composant connecté en fonction de l'intensité est stockée en produisant une structure de données. Une fusion de deux composants est vue comme la fin de l'existence du plus petit composant et une insertion de tous les pixels du plus petit composant dans le plus grand. Dans les régions extrêmes, les variables «maximales stables» sont celles correspondant à des seuils où le changement de zone relatif en fonction du changement relatif de seuil est à un minimum local, ie les MSER sont les parties de l'image où la binarisation locale est stable sur une large gamme de seuils [96] [99]. L'arbre des composants est l'ensemble de tous les composants connectés des seuils de l'image, classés par inclusion. Des algorithmes efficaces (quasi-linéaires quelle que soit la gamme des poids) pour le calculer existent bel et bien [102]. Ainsi, cette structure offre un moyen facile de mettre en œuvre MSER [103].

Plus récemment, Nister et Stewenius ont proposé une méthode dans, [98] qui est aussi beaucoup plus rapide dans la pratique dont l'algorithme est similaire à celui de Ph. Salembier et al. [100]



*Fig. 3. 1 Résultats de MSER sur des exemples d'images une personne*



*Fig. 3. 2 Résultat de MSER sur image de plusieurs personnes*

**Discussion :**

Dans cet exemple, nous notons que ce détecteur se concentre uniquement sur les régions et à partir de là, nous concluons qu'il n'est pas efficace. Donc, nous ne pouvons pas l'utiliser pour détecter le visage.

3.3 Méthode Edge Detection

3.3.1 Méthode de détection des contours

D'abord, l'image de niveau de gris est considérée. Afin d'éviter l'apparition de doubles bords, de bords épais ou de tâches, nous nous intéressons à une nouvelle méthode pour détecter le bord d'une image. L'intensité du contour est normalisée dans  $[0, L - 1]$  et  $L$  est le nombre de niveaux de gris dans l'image numérique. Pour le masque  $3 \times 3$ , un pixel de bord appartient généralement à l'un des quatre bords de direction possibles voir (Fig. 3.3).

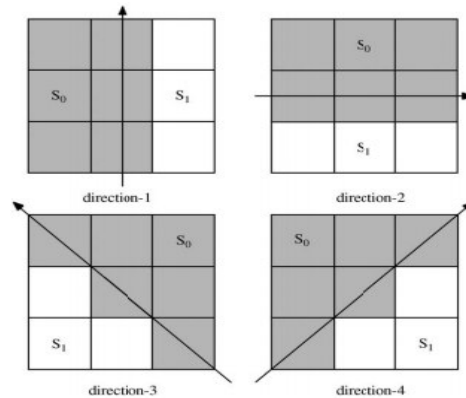


Fig. 3.3 Quatre directions de bord [109].

Dans le modèle de bord de direction -1, neuf pixels peuvent être divisés en deux ensembles,  $S_0$  et  $S_1$  comme  $s_0 = \{p_1, p_2, p_4, p_5, p_7, p_8\}$  et  $s_1 = \{p_3, p_6, p_9\}$  dans lequel cela n'a pas d'importance si  $S_0$  ou  $S_1$  a un niveau de gris plus élevé. De même,  $s_0 = \{p_1, p_2, p_3, p_4, p_5, p_6\}$  et  $s_1 = \{p_7, p_8, p_9\}$  pour le bord de direction -2,  $s_0 = \{p_1, p_2, p_3, p_5, p_6, p_9\}$  et  $s_1 = \{p_4, p_7, p_8\}$  pour le bord de direction -3. , et  $s_0 = \{p_1, p_2, p_3, p_4, p_5, p_7\}$  et  $s_1 = \{p_6, p_8, p_9\}$  pour le bord de la direction -4.

Le principe de la méthode étudiée est décrit comme suit. Considérons un masque  $3 \times 3$  dans une image locale dans laquelle n'importe quel type d'arêtes décrit ci-dessus peut exister. On sait que si la distance interset entre  $S_0$  et  $S_1$  dans le masque est grande et que les distances intra-série de  $S_0$  et  $S_1$  sont faibles, alors la compacité des ensembles  $S_0$  et  $S_1$  est élevée et l'intensité des bords est grande. Par conséquent, nous avons besoin d'une fonction objective qui contient les distances décrites ci-dessus afin d'estimer l'intensité du bord. Définissons une fonction objective correspondant à la direction du front-j comme (Equ. 3.3)

$$f_i = (L-1) \frac{N_f}{D_f} \tag{Equ. 3.3}$$

$$\text{Où} \quad N_f = \min\left(1, \frac{|m_0 - m_1|}{\omega_2}\right) \quad (\text{Equ. 3.4})$$

$$D_f = 1 + \frac{1}{15} \sum_{\substack{pm, pn \in S_0 \\ m > n, m \neq n}} \min\left(1, \frac{|p_m - p_n|}{\omega_2}\right) + \frac{1}{3} \sum_{\substack{pm, pn \in S_1 \\ m > n, m \neq n}} \min\left(1, \frac{|p_m - p_n|}{\omega_2}\right) \quad (\text{Equ. 3.5})$$

$\omega_1 = 90$ ,  $\omega_2 = 40$ ,  $m_0 = (1/6) \sum_{pi \in S_0} p_i$ , et  $m_1 = (1/3) \sum_{pi \in S_1} p_i$ . On note qu'il y a différentes valeurs de  $S_0$  et  $S_1$  pour différentes directions de  $j$ . Pour une image de niveau de gris de 8 bits,  $L$  est égal à 256. Le numérateur  $N_f$  de (Equ. 3.3) est le degré de la distance interset entre  $S_0$  et  $S_1$ . Il sera mis à 1 si la distance est supérieure à  $\omega_1$ . L'intensité du bord est suffisamment grande si la distance interset entre  $S_0$  et  $S_1$  est plus grande que le niveau de gris  $\omega_1$ . Si la distance interset est dans l'intervalle  $[0, \omega_1]$ , alors nous utilisons une fonction linéaire pour décrire  $N_f$ . Ici,  $\omega_1$  est défini sur 90, valeur sélectionnée parmi de nombreuses expériences. Le dénominateur  $D_f$  est la somme de deux degrés des distances intra-ensembles à l'intérieur de deux ensembles. Le premier terme 1 est utilisé pour empêcher que  $D_f$  soit nul. L'intensité du bord est faible si les distances intraset sont grandes et nous pouvons considérer que le degré de la distance intraset est grand alors que la distance intraset est plus grande que les niveaux de gris  $\omega_2$ . Ainsi,  $\omega_2$  est défini sur 40, valeur sélectionnée parmi de nombreuses expériences, et nous utilisons une fonction linéaire pour représenter le degré de la distance intraset si la distance intraset est comprise dans l'intervalle  $[0, \omega_2]$ .  $f_i$  est normalisé dans  $[0, L - 1]$  en raison de la multiplication  $[L - 1]$  dans (Equ. 3.3) Pour un masque  $3 \times 3$  dans l'image, nous calculons quatre valeurs de fonctions objectives,  $f_1, \dots, f_4$ , correspondant à quatre directions, respectivement. Nous pouvons obtenir l'intensité de bord  $E(x, y)$  à partir de (Equ. 3.6) et la direction de bord  $D(x, y)$  de (Equ. 3.7) pour le pixel central  $(x, y)$  le masque.

$$E(x, y) = \max(f_1, f_2, f_3, f_4). \quad (\text{Equ. 3.6})$$

$$D(x, y) = \text{Arg}(\max(f_1, f_2, f_3, f_4)). \quad (\text{Equ. 3.7})$$

Par conséquent, la carte de bord avec chaque pixel  $(x, y)$  remplacé par  $E(x, y)$  et la carte de direction avec chaque pixel remplacé par  $D(x, y)$  sont générées. En d'autres termes,  $E(x, y)$  et  $D(x, y)$  de tous les pixels sont obtenus. Si  $E(x, y) < T$ , où  $T$  est un seuil, alors l'intensité du bord est trop petite pour extraire un point de contour. Sous le cas  $E(x, y) \geq T$ , le NMS suivant

est appliqué aux deux cartes de bord et de direction pour extraire les points de bord. Ainsi, le pixel central est un point de bord lorsque l'un des quatre cas suivants se produit.

- (1)  $D(x, y) = 1$  and  $E(x, y) > E(x, y - 1)$  and  $E(x, y) > E(x, y + 1)$ .
- (2)  $D(x, y) = 2$  and  $E(x, y) > E(x - 1, y)$  and  $E(x, y) > E(x + 1, y)$ .
- (3)  $D(x, y) = 3$  and  $E(x, y) > E(x - 1, y + 1)$  and  $E(x, y) > E(x + 1, y - 1)$ .
- (4)  $D(x, y) = 4$  and  $E(x, y) > E(x - 1, y - 1)$  and  $E(x, y) > E(x + 1, y + 1)$ .

En excluant les quatre cas ci-dessus, le pixel central n'est pas un point de contour. L'analyse ci-dessus est résumée comme suit :

**Étape 1:** Calculer quatre valeurs de fonctions objectives  $f1, \dots, f4$  à partir de (Equ. 3.3.4.5) respectivement.

**Étape 2:** Générez la carte de bord et la carte de direction à partir des (Equ. 3.6) et (Equ. 3.7).

**Étape 3:** Appliquez NMS au bord et aux cartes de direction et extrayez les points de bord dans l'image.

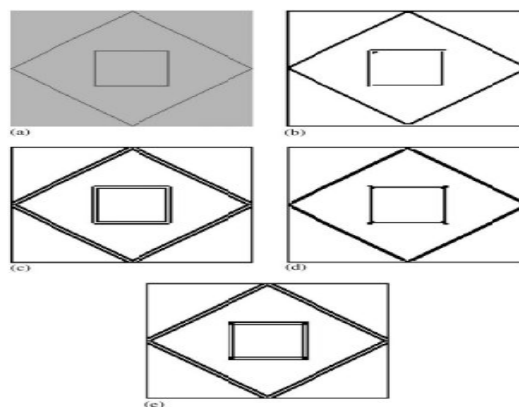


Fig. 3. 4 Comparaisons de détection de ligne utilisant différentes méthodes [104].

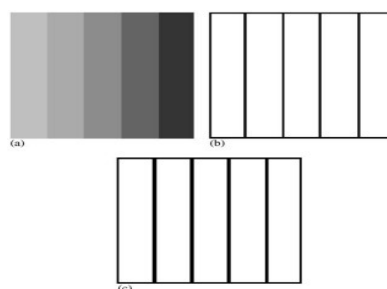


Fig. 3. 5 (a) image originale; (b) résultat de la méthode edge detection; (c) résultat d'autres méthodes [104].

Considérons le cas montré sur la figure (Fig. 3.5) , où  $f1$  ne doit jamais être 0 quelle que soit la ligne de direction avec une largeur de pixel passe à travers le masque. En appliquant NMS, seul le pixel dont l'intensité de bord est la plus grande le long de la direction perpendiculaire à sa

direction de bord peut être considéré comme un point de contour. Par conséquent, la largeur de presque n'importe quelle ligne de bord détectée dans l'image est seulement un pixel. Cela évite l'apparition de taches et les doubles bords provoqués par les points de bord voisins fermant le point de bord central dans le masque.

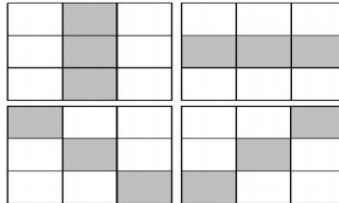


Fig. 3. 6 Les lignes d'une largeur de pixel traversent le masque en tant que ligne médiane [104].

### 3.3.2 Filtrage optimal

Les opérations décrites précédemment se traduisent par des filtres à appliquer à l'image. Canny a défini trois critères pour obtenir un filtre optimal pour la détection de contours :

1. *Bonne détection* : détecter un maximum de contours.
2. *Bonne localisation* : les points détectés doivent être les plus proches possibles du vrai contour.
3. *Réponse unique* : minimiser le nombre de contours détectés plusieurs fois.

Ces critères se traduisent par des conditions sur la réponse impulsionnelle du filtre et débouchent sur des détecteurs de contours très performants.

### 3.3.3 Dérivée première

3.3.3.1 *Méthode de base* : L'opérateur de Roberts décrit dans une image rectangulaire, ayant un contour rampe, les dérivées premières elles-mêmes sont d'un intérêt limité car la pente maximale a peu de chances de se trouver sur l'une des deux directions considérées. Ce qui importe c'est la longueur du vecteur gradient dont elles sont les composantes. Cette longueur se calcule en principe par le théorème de Pythagore au prix d'un calcul sur les réels et on l'accélère considérablement en utilisant une approximation entière :

$$|G| = |E - W| + |N - S|$$

Indépendamment de la précision du calcul, il faut transformer ce résultat en un filtre numérique. La notion physique de filtre correspond à la notion mathématique de convolution. Lorsqu'il s'agit de données numérisées comme dans le cas du traitement d'images, la relation entre les valeurs des pixels de sortie et celle des pixels d'entrée est décrite par un tableau de nombres appelé matrice de convolution.

3.3.3.2 *Filtre de Prewitt* : La matrice qui correspond au filtrage horizontal, faisant ressortir essentiellement les contours verticaux, selon l'opérateur de Prewitt, s'écrit  $h_x = [-1 \ 0 \ 1]$  tandis



que la matrice verticale  $h_y$  est sa transposée. Les deux convolutions avec le tableau de valeurs initiales créent deux tableaux  $G_x$  et  $G_y$  à l'origine du tableau  $G$  sur lequel on peut localiser les maximums.

**3.3.3.3 Filtre de Sobel :** On obtient un meilleur résultat en remplaçant le filtre rectangulaire par un filtre triangulaire.

**3.3.3.4 Filtre de Canny :** Le filtre de Sobel est apprécié pour sa simplicité et sa rapidité d'exécution. Ces qualités posent des problèmes lorsqu'il s'agit de traiter une image complexe. Le filtre de Canny a été bâti autour de l'algorithme de Sobel pour améliorer ses résultats.

D'une part, les filtres triangulaires utilisés par Sobel étant peu efficaces face à une image fortement bruitée, un filtre gaussien est utilisé. D'autre part, c'est là l'originalité de la méthode, elle permet d'éliminer des faux contours. En considérant non seulement l'intensité du gradient mais aussi sa direction, il est possible d'éliminer un pixel qui pointe vers deux pixels de valeur supérieure car ce n'est pas un maximum local. Il faut ensuite effectuer un *seuillage par hystérésis*. Pour cela on fixe deux seuils, un seuil haut  $s_h$  et un seuil bas  $s_b$ . On commence par sélectionner les points qui dépassent le seuil haut et on applique ensuite le seuil bas en ne conservant que les composantes connexes qui contiennent un point au-dessus de  $s_h$ . En d'autres termes à partir de chaque point au-dessus de  $s_h$  on « suit » un chemin constitué de points au-dessus de  $s_b$ , ce chemin est le contour recherché.[105]

### 3.3.4 Implémentation de la méthode edge detection et résultats

La détection de contour est très utile en traitement d'images, c'est par exemple une étape indispensable à la reconnaissance de formes. Ces algorithmes sont également utilisés en imagerie médicale, cartographie, etc.

#### 3.3.4.1 Application d'edge detection aux images couleurs

Cette méthodologie est applicable non seulement pour les images de niveau de gris, mais aussi pour les images en couleur. Dans une image en couleur, que le  $i^{\text{ème}}$  pixel soit exprimé par un modèle de couleur RVB tel que  $p_i = (r_i, g_i, b_i)$ . Le concept de distance est également appliqué au modèle de couleur avec l'expression vectorielle  $p_i = (r_i, g_i, b_i)$ . Par conséquent, la fonction objective (Equ. 3.3) est à nouveau utilisée, mais  $N_f$  dans l'équation. (Equ. 3.4) et  $D_f$  dans l'équation. (Equ. 3.5) sont remplacés par des Eqs.(Equ.3.8) et (Equ. 3.9), respectivement où  $m_0 = (1/6) \sum_{p_i \in S_0} p_i$  et  $m_1 = (1/3) \sum_{p_i \in S_1} p_i$ . Il convient de noter que  $p_m$  ou  $p_n$  dans Eq. (Equ. 3.9) est l'un des pixels de (Equ. 3.6) avec l'expression vectorielle. D'où les Eqs. (Equ. 3.8) et

(Equ. 3.9) devraient utiliser la valeur de la norme au lieu de la valeur absolue. Par conséquent, la procédure de détection de bord de la section 3 est également appliquée à l'image couleur.

$$N_f = \min\left(1, \frac{\|m_0 - m_1\|}{\omega_1}\right) \tag{Equ. 3.8}$$

$$D_f = 1 + \frac{1}{15} \sum_{\substack{pm, pn \in S_0 \\ m > n, m \neq n}} \min\left(1, \frac{\|p_m - p_n\|}{\omega_2}\right) + \frac{1}{3} \sum_{\substack{pm, pn \in S_1 \\ m > n, m \neq n}} \min\left(1, \frac{\|p_m - p_n\|}{\omega_2}\right) \tag{Equ. 3.9}$$

$p_1$	$p_2$	$p_3$
$p_4$	$p_5$	$p_6$
$p_7$	$p_8$	$p_9$

$(x-1, y-1)$	$(x-1, y)$	$(x-1, y+1)$
$(x, y-1)$	$(x, y)$	$(x, y+1)$
$(x+1, y-1)$	$(x+1, y)$	$(x+1, y+1)$

Fig. 3. 7 Le masque 3 × 3 [104].

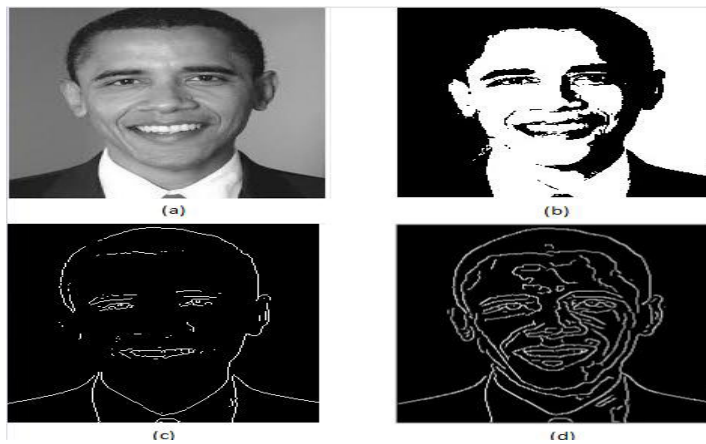
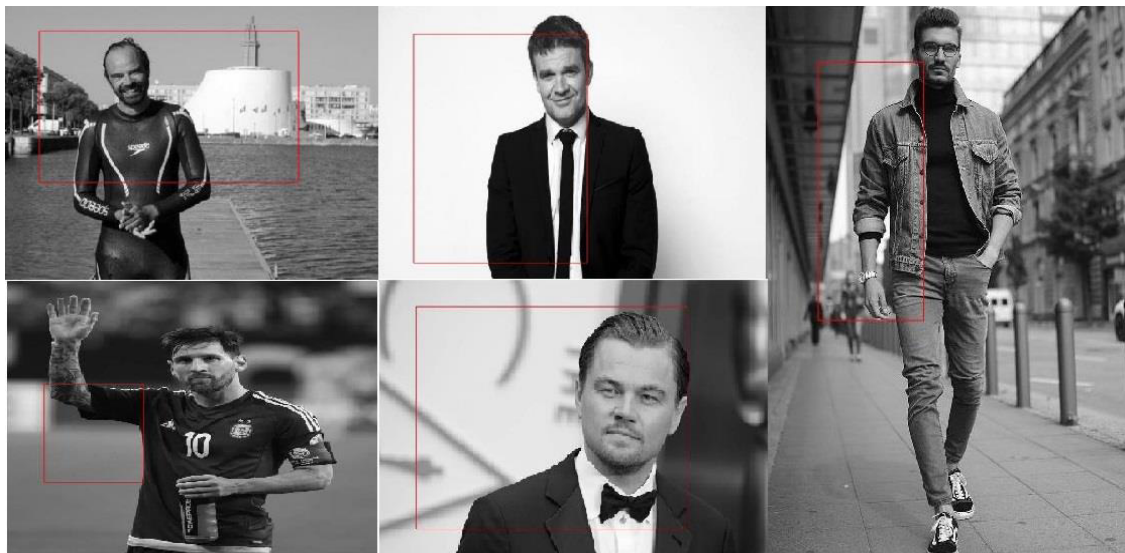


Fig. 3. 8 Détection de contours

(a) Image niveaux de gris, (b) image binaire, (c) détection de contours à l'aide de Prewitt (d) détection de contours à l'aide de Canny



*Fig. 3. 9 Images détectées*



*Fig. 3. 10 Images non détectées*

**Discussion :**

Nous remarquons que les détections par la méthode Edge Detection sont significativement efficaces, malgré les changements de caractéristique du visage et couleur de la peau. Elle reste inefficace dans certains cas de variation de pose , cas de mouvements (action) , différence de luminosité et changement de distance entre l’homme et la caméra.

3.4 Détection de visage à l'aide de réseaux neuronaux et de fonctionnalités Gabor

3.4.1 Principe du Filtre de Gabor

Les filtres de Gabor ont été appliqués à divers problèmes de reconnaissance d'image pour l'extraction de caractéristiques en raison de ses propriétés de localisation optimales dans le domaine spatial et fréquentiel. La forme fonctionnelle générale d'un filtre de Gabor 2D spécifié dans le domaine de la fréquence spatiale et spatiale est donnée par

$$g(x, y) = \exp \left[ -\left( \frac{x^2 + y^2}{2\sigma_{xy}^2} \right) \right] \cos(u_0 x + v_0 y), \quad \text{(Equ. 3.10)}$$

où  $\sigma_{xy}$  est l'écart type de l'enveloppe gaussienne qui caractérise l'étendue spatiale et la bande passante du filtre. Les paramètres  $(u_0, v_0)$  définissent la fréquence spatiale d'une onde plane sinusoïdale, qui peut également être exprimée en coordonnées polaires comme la fréquence radiale  $r_0$  et l'orientation  $\theta$ :

$$r_0^2 = u_0^2 + v_0^2, \quad \tan \theta = \frac{v_0}{u_0}. \quad \text{(Equ. 3.11)}$$

La propriété du filtre de Gabor est définie par la fréquence radiale  $r_0$ , l'orientation et la bande passante du filtre.

$$\theta_k = \frac{\pi(k-1)}{4}, \quad k = 1, 2, 3, 4.$$

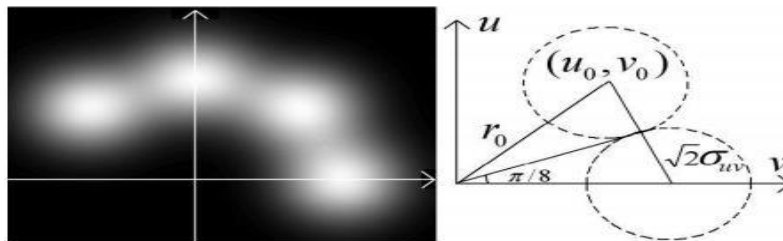


Fig. 3. 11 Les quatre filtres dans le domaine de la fréquence spatiale [106].

Les réponses en fréquence des quatre filtres sont représentées sur (Equ. 3.11). La bande de fréquence de chaque filtre est un gaussien centré sur  $(u_0, v_0)$ .

3.4.2 Convolution du filtre de Gabor et de l'image

La représentation de Gabor d'une image est la convolution de l'image avec le filtre de Gabor. Basé sur les représentations de Gabor, un vecteur de caractéristiques est formé. Soit  $I(x, y)$  une image, la convolution de l'image  $I(x, y)$  et un filtre de Gabor est défini comme:

$$O(x, y, r_0, \theta_k) = I(x, y) * g(x, y, r_0, \theta_k), \quad k = 1, \dots, 4 \quad \text{(Equ. 3.12)}$$

(Fig. 3.12) montre les images filtrées de Gabor d'un échantillon de visage. Les images filtrées sont visualisées en codant les valeurs de sortie du filtre de Gabor en niveaux de gris. Nous pouvons voir que les propriétés d'orientation de la figure sont bien représentées par les images filtrées de Gabor.



*Fig. 3. 12 Une image d'échantillon de visage et ses quatre images filtrées de Gabor [106].*

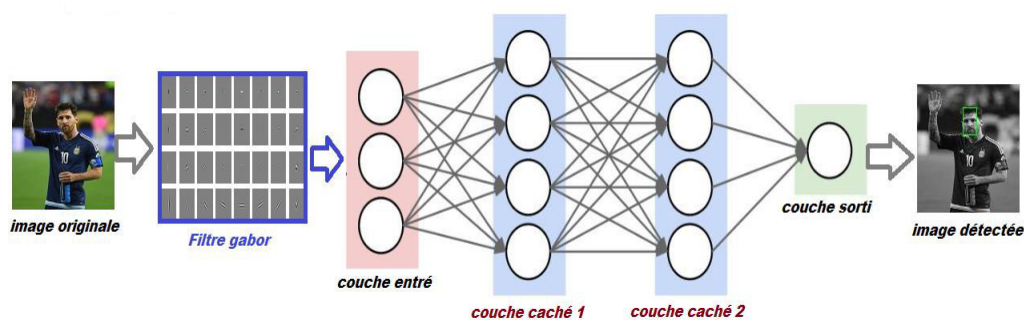
### 3.4.3 Réseaux neuronaux

Un réseau de neurones est un système de traitement de l'information qui a été développé sous la forme de généralisations de modèles mathématiques correspondant à la cognition humaine. Ils sont composés d'un grand nombre d'unités de traitement hautement interconnectées (neurones) qui travaillent ensemble pour effectuer une tâche spécifique. Selon Haykin [8], un réseau de neurones est un processeur massivement parallélisé qui a une prospérité naturelle pour stocker les connaissances expérimentales. Il ressemble au cerveau dans deux pôles:

- *La connaissance est acquise par le réseau à travers un processus d'apprentissage;*
- *Les forces de connexion interconnectées connues sous le nom de poids synaptiques sont utilisées pour stocker les connaissances;*
- *Chaque neurone a un état interne appelé sa fonction de seuil ou d'activation (ou fonction de transfert) utilisée pour classer les vecteurs. La classification neurale comprend généralement quatre étapes:*
  - *Pré-traitement, par exemple, correction atmosphérique, suppression de bruit, rationnement de bande, analyse de composante principale, etc.*
  - *Entraînement - sélection des caractéristiques particulières qui décrivent le mieux le motif;*
  - *Décision - choix de la méthode appropriée pour comparer les modèles d'image avec les modèles cibles;*
  - *Évaluation de l'exactitude de la classification.*

### 3.4.4 Implémentation de la méthode réseaux de neurones-Gabor (GNN)

Le système de détection de visage pour MATLAB est l'un des meilleurs programmes que l'on peut apprendre. La détection de visage est la première et dernière étape de tout système de reconnaissance faciale automatique. Sa fiabilité affecte grandement les performances et la convivialité de l'ensemble du système. Étant donné une image unique ou une image vidéo, un détecteur de visage idéal devrait avoir la capacité de localiser tous les visages présents dans cette image, indépendamment de leur position, de leurs gestes faciaux, de leurs variations d'échelle et de leur orientation. En outre, il doit être robuste contre les variations d'éclairage, de couleur de la peau ou de fond ...



*Fig. 3. 13 Schéma de principe du GNN*

Plusieurs indices peuvent faciliter le processus de détection. La couleur de la peau (pour détecter les visages dans des images et des vidéos colorées) est souvent utilisée. Le mouvement (pour détecter les visages dans la vidéo) est un autre indice bien connu qui peut être estimé en analysant plusieurs images vidéo d'affilée. Mais le type de détection le plus difficile est le détecteur de visage dans les images fixes au niveau du gris, dans lesquelles il n'y a pas de repère d'aucune sorte, comme la couleur ou le mouvement.

- *Première étape (extraction de fonctionnalité)* : on a besoin d'une fonction qui peut transformer une petite parcelle d'image en un vecteur (extraction de caractéristiques: Fonctionnalités de niveau de gris - Égalisation d'histogramme - Effets d'illumination - Normalisation). Cette fonction est la fonction d'extraction de caractéristiques et devrait extraire les fonctionnalités de manière judicieuse, suivie d'une normalisation. Dans le système de détection de visage pour MATLAB, les fonctionnalités de Gabor sont extraites du patch (Caractéristiques de Gabor et extraction de caractéristiques Gabor: concept, code et transformation rapide de Gabor) ;
- *Deuxième étape* : génération des données pour la formation

- *Troisième étape (formation du classificateur)* : Chaque système de détection a besoin d'un classificateur qui prend en considération le vecteur caractéristique et décide s'il s'agit de l'accord ou non. En cas de détection de visage, le classificateur recherche les visages. Les principaux problèmes sont de choisir le classificateur et de définir les paramètres de manière à obtenir des résultats raisonnables. Le système de détection de visage pour MATLAB utilise Neural Network comme classificateur (Réseaux neuronaux artificiels: Perceptron Neuron, Perceptron à couche unique, Réseau de feed-forward multi-couches, Formation de rétropropagation, Création de réseaux, Entrées et sorties, Initialisation, Tests et Simulation, Génération de kits de formation)
- *Quatrième étape (Numérisation de l'image)* : Une image est ensuite scannée à tous les emplacements possibles [et échelles] par une sous-fenêtre (patch). Chaque patch est envoyé à la fonction d'extraction de caractéristiques et le vecteur de sortie va au classificateur. Il existe des moyens de présélectionner les emplacements possibles et de déterminer l'emplacement des visages [106]. Autres éléments d'un système de détection de visages: présélection, algorithme de recherche et post-traitement

### 3.4.5 Résultats pour la détection par GNN

Quelques exemples d'images pour tester la structure proposée sont illustrées par les figures 3.14 et 3.15.



*Fig. 3. 14 Quelques exemples d'images pour tester la structure proposée*



*Fig. 3. 15 Exemples d'image de mauvaise détection de la pose*





Fig. 3. 16 Exemple d'image de mauvaise détection de la couleur de la peau



Fig. 3. 17 Exemple d'images de mauvaise détection avec l'algorithme proposé

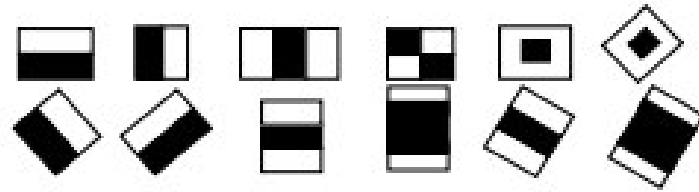
### 3.4.6 Discussion :

Cette méthode de détection du visage est une méthode efficace pour les visages frontaux avec de légères variantes. Mais, vu les résultats elle présente des limites dans certains cas, et nous nous référons à la localisation de l'image et aux difformités faciales, en particulier la différence de couleur de la peau, la pose, l'éloignement de la caméra, image bruitée . C'est ce qui affecte négativement la détection du visage de cette méthode, parceque comme nous le voyons sur les figures ci-dessus. Nous détectons pas les visages aussi dans les cas suivant : un groupe de personnes est variable en position, l'intensité de la lumière, présence du fond de l'image et l'éloignement de la caméra.

### 3.5 Méthode de Viola-Jones

La méthode Viola-Jones, proposée par P. Viola et M. Jones en 2001, est basée sur le traitement de l'image intégrale à l'aide d'une cascade de classificateurs «faibles», appelés caractéristiques de Haar [2].





*Fig. 3. 18 Les descripteurs de Haar [108].*

La méthode de Viola et Jones est une approche basée sur l'apparence[108], qui consiste à parcourir l'ensemble de l'image en calculant un certain nombre de caractéristiques dans des zones rectangulaires qui se chevauchent. Elle a la particularité d'utiliser des caractéristiques très simples mais très nombreuses. Une première innovation de la méthode est l'introduction des images intégrales, qui permettent le calcul rapide de ces caractéristiques. Une deuxième innovation importante est la sélection de ces caractéristiques par boosting, en interprétant les caractéristiques comme des classifieurs. Enfin, la méthode propose une architecture pour combiner les classifieurs boostés en un processus en cascade, ce qui apporte un net gain en temps de détection. La méthode, en tant que méthode d'apprentissage supervisé, est divisée en deux étapes : une étape d'apprentissage du classifieur basé sur un grand nombre d'exemples positifs (c'est-à-dire les objets d'intérêt, par exemple des visages) et d'exemples négatifs, et une phase de détection par application de ce classifieur à des images inconnues

La méthode Viola-Jones comprend les étapes suivantes [2]:

1. *Formation du classificateur en cascade en utilisant l'algorithme AdaBoost;*
2. *Fonctionnement en conditions réelles pour chaque trame: (1) Construire une image intégrée de la trame afin d'optimiser le travail; (2) Recherchez un objet dans l'image en utilisant la cascade.*

Ainsi, le paramètre principal qui existe dans l'algorithme de Viola-Jones est la cascade utilisée. Dans la mise en œuvre technique de la méthode Viola-Jones, des paramètres supplémentaires sont introduits [109]:

1. *L'indicateur de changement de la taille de la fenêtre pendant la transition d'une itération de l'exécution de l'algorithme à l'autre est  $s_f$ . La valeur minimale du paramètre est 1.01.*

2. Le nombre d'objets reconnus l'un à côté de l'autre, sur la base duquel on peut conclure qu'il y a un objet cible dans la zone donnée est min. La valeur minimale du paramètre est 1.

3. La taille initiale de la boîte de recherche est min. Ce paramètre ne varie pas dans le cadre de la recherche conduite, car la tâche nécessite de trouver tous les objets possibles, y compris l'objet minimum possible. La fenêtre de recherche minimale typique la taille est 20x20 pixels [2,110].

```

Algorithm :Viola-Jones Face Detection Algorithm
1: Input : original test image
2: Output : image with face indicators as rectangles
3: for i ← 1 to num of scales in pyramid of images do
4:     Downsample image to creat imagei
5:     Compute integral image, imageii
6:     for j ← 1 to num of shift steps of sub-window do
7:         for k ← 1 to num of stages in cascade classifier do
8:             for l ← 1 to num of filters of stage k do
9:                 Filter detection sub-window
10:                Accumulate filter outputs
11:            end for
12:            if accumulation fails per-stage threshold then
13:                Reject sub-window
14:                Break this k loop
15:            end if
16:        end for
17:        if sub-window passed all per-stage checks then
18:            Accept this sub-window as a face
19:        end if
20:    end for
21: end for

```

Fig. 3. 19 Contexte sur l'algorithm de Viola-Jones [111].

### 3.5.1 Caractéristiques

**3.5.1.1 Description** : Plutôt que de travailler directement sur les valeurs de pixels, et pour être à la fois plus efficace et plus rapide, Viola et Jones proposent d'utiliser des caractéristiques, c'est-à-dire une représentation synthétique et informative, calculée à partir des valeurs des pixels. Viola et Jones définissent des caractéristiques très simples, les caractéristiques pseudo-Haar [110], qui sont calculées par la différence des sommes de pixels de deux ou plusieurs zones rectangulaires adjacentes.

Pour calculer rapidement et efficacement ces caractéristiques sur une image, les auteurs proposent également une nouvelle méthode, qu'ils appellent « image intégrale ». C'est une

représentation sous la forme d'une image, de même taille que l'image d'origine, qui en chacun de ses points contient la somme des pixels situés au-dessus de lui et à sa gauche. Plus formellement, l'image intégrale  $ii$  au point  $(x, y)$  est définie à partir de l'image  $i$  par [110]

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y') \quad (\text{Equ. 3.13})$$

Grâce à cette représentation, une caractéristique formée de deux zones rectangulaires peut être calculée en seulement 6 accès à l'image intégrale, et donc en un temps constant quelle que soit la taille de la caractéristique

**3.5.1.2 Calcul :** Les caractéristiques sont calculées à toutes les positions et à toutes les échelles dans une fenêtre de détection de petite taille, typiquement de  $24 \times 24$  pixels [112] ou de  $20 \times 15$  pixels [112]. Un très grand nombre de caractéristiques par fenêtre est ainsi généré, Viola et Jones donnant l'exemple d'une fenêtre de taille  $24 \times 24$  qui génère environ 160 000 caractéristiques.

En phase de détection, l'ensemble de l'image est parcourue en déplaçant la fenêtre de détection d'un certain pas dans le sens horizontal et vertical (ce pas valant 1 pixel dans l'algorithme original [2]). Les changements d'échelles se font en modifiant successivement la taille de la fenêtre de détection. Viola et Jones utilisent un facteur multiplicatif de 1,25, jusqu'à ce que la fenêtre couvre la totalité de l'image.

Finalement, et afin d'être plus robuste aux variations d'illumination, les fenêtres sont normalisées par la variance [2].

La conséquence de ces choix techniques, notamment le recours aux images intégrales, est un gain notable en efficacité, les caractéristiques étant évaluées très rapidement quelle que soit la taille de la fenêtre.

### 3.5.2 Sélection de caractéristiques par boosting

Le deuxième élément clé de la méthode de Viola et Jones est l'utilisation d'une méthode de boosting afin de sélectionner les meilleures caractéristiques. Le boosting est un principe qui consiste à construire un classifieur « fort » à partir d'une combinaison pondérée de classifieurs « faibles », c'est-à-dire donnant en moyenne une réponse meilleure qu'un tirage aléatoire. Viola et Jones adaptent ce principe en assimilant une caractéristique à un classifieur faible, en construisant un classifieur faible qui n'utilise qu'une seule caractéristique. L'apprentissage du classifieur faible consiste alors à trouver la valeur seuil de la caractéristique qui permet de mieux séparer les exemples positifs des exemples négatifs. Le classifieur se réduit alors à un couple (caractéristique, seuil).

L'algorithme de boosting utilisé est en pratique une version modifiée d'AdaBoost, qui est utilisée à la fois pour la sélection et pour l'apprentissage d'un classifieur « fort ». Les classifieurs faibles utilisés sont souvent des arbres de décision. Un cas remarquable, fréquemment rencontré, est celui de l'arbre de profondeur 1, qui réduit l'opération de classification à un simple seuillage.

L'algorithme est de type itératif, à nombre d'itérations déterminé. À chaque itération, l'algorithme sélectionne une caractéristique, qui sera ajoutée à la liste des caractéristiques sélectionnées aux itérations précédentes, et le tout va contribuer à la construction du classifieur fort final. Cette sélection se fait en entraînant un classifieur faible pour toutes les caractéristiques et en sélectionnant celui avec l'erreur la plus faible sur l'ensemble d'apprentissage. L'algorithme tient également à jour une distribution de probabilité sur l'ensemble d'apprentissage, réévaluée à chaque itération en fonction des résultats de classification. En particulier, plus de poids est attribué aux exemples difficiles à classer, c'est-à-dire ceux dont l'erreur est élevée. Le classifieur « fort » final construit par AdaBoost est composé de la somme pondérée des classifieurs sélectionnés.

Plus formellement, on considère un ensemble de  $n$  images  $(x_1, \dots, x_n)$  et leurs étiquettes associées  $(y_1, \dots, y_n)$ , qui sont telles que  $y_i = 0$  si l'image  $x_i$  est un exemple négatif et  $y_i = 1$  si  $x_i$  est un exemple de l'objet à détecter. L'algorithme de boosting est constitué d'un nombre  $T$  d'itérations, et pour chaque itération  $t$  et chaque caractéristique  $i$ , on construit un classifieur faible  $h_j$ . Idéalement, le but est d'obtenir un classifieur  $h$  qui prédise exactement les étiquettes pour chaque échantillon, c'est-à-dire  $\forall i \in \{1 \dots n\}$ . En pratique, le classifieur n'est pas parfait et l'erreur engendrée par ce classifieur est donnée par :

$$\epsilon_j = \sum_{i=1}^n \omega_i |h_j(x_i) - y_i| \tag{Equ. 3.14}$$

les  $\omega_i$  étant les poids associés à chaque exemple et mis à jour à chaque itération en fonction de l'erreur obtenue à l'itération précédente. On sélectionne alors à l'itération  $t$  le classifieur  $h_t$  présentant l'erreur la plus faible :  $\epsilon_t = \min_j(\epsilon_j)$

Le classifieur fort final  $h(x)$  est construit par seuillage de la somme pondérée des classifieurs

faibles sélectionnés :

$$h(x) = \left\{ \begin{array}{l} 1_{si} \\ 0_{sin\ on} \end{array} \sum_{t=1}^T \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^T \alpha_t \right\} \tag{Equ. 3.15}$$

Les  $\alpha_t$  sont des coefficients calculés à partir de l'erreur  $\epsilon_t$ .

### 3.5.3 Cascade de classifieurs

La méthode de Viola et Jones est basée sur une approche par recherche exhaustive sur l'ensemble de l'image, qui teste la présence de l'objet dans une fenêtre à toutes les positions et à plusieurs échelles. Cette approche est cependant extrêmement coûteuse en calcul. L'une des idées-clés de la méthode pour réduire ce coût réside dans l'organisation de l'algorithme de détection en une cascade de classifieurs. Appliqués séquentiellement, ces classifieurs prennent une décision d'acceptation — la fenêtre contient l'objet et l'exemple est alors passé au classifieur suivant —, ou de rejet — la fenêtre ne contient pas l'objet et dans ce cas l'exemple est définitivement écarté —. L'idée est que l'immense majorité des fenêtres testées étant négatives (c.-à-d. ne contiennent pas l'objet), il est avantageux de pouvoir les rejeter avec le moins possible de calculs. Ici, les classifieurs les plus simples, donc les plus rapides, sont situés au début de la cascade, et rejettent très rapidement la grande majorité des exemples négatifs. Cette structure en cascade peut également s'interpréter comme un arbre de décision dégénéré, puisque chaque nœud ne comporte qu'une seule branche [2].

Le choix du nombre  $K$  d'étages est fixé par l'utilisateur ; dans leur méthode originale, Viola et Jones utilisent  $K = 32$  étages. L'utilisateur doit également spécifier le taux de détection minimal  $d_i$  et le taux de fausse alarme maximal  $f_i$  à atteindre pour l'étage  $i$ . Le taux

de détection de la cascade est alors donné par :  $D = \prod_{i=1}^K d_i$  et le taux de fausse alarme par :

$F = \prod_{i=1}^K f_i$  En pratique, les taux  $d_i$  et  $f_i$  sont les mêmes pour tous les étages. Indirectement,

ces taux déterminent également le nombre de caractéristiques utilisées par les classifieurs forts à chaque étage : les itérations d'Adaboost continuent jusqu'à ce que le taux de fausse alarme cible soit atteint. Des caractéristiques/classifieurs faibles sont ajoutés jusqu'à ce que les taux cibles soient atteints, avant de passer ensuite à l'étage suivant.

Plusieurs chercheurs font remarquer que cette idée de filtrer rapidement les exemples négatifs les plus simples n'est pas nouvelle. Elle existe dans d'autres méthodes sous forme d'heuristiques, comme la détection de la couleur chair ou une étape de pré-classification.

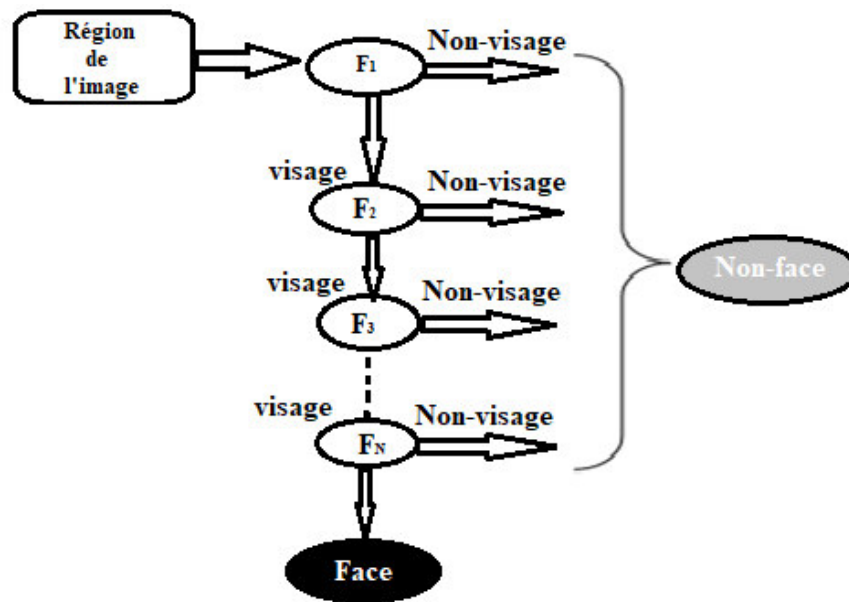


Fig. 3. 20 Cascade de classifieurs [2].

### 3.5.4 Étapes clés

**3.5.4.1 Apprentissage :** L'apprentissage est réalisé sur un très large ensemble d'images positives (c'est-à-dire contenant l'objet) et négatives (ne contenant pas l'objet). Plusieurs milliers d'exemples sont en général nécessaires. Cet apprentissage comprend :

- Le calcul des caractéristiques pseudo-Haar sur les exemples positifs et négatifs ;
- L'entraînement de la cascade : à chaque étage de la cascade, un classifieur fort est entraîné par AdaBoost. Il est construit par ajouts successifs de classifieurs faibles entraînés sur une seule caractéristique, jusqu'à l'obtention de performances conformes aux taux de détection et de fausse alarme souhaités pour l'étage.

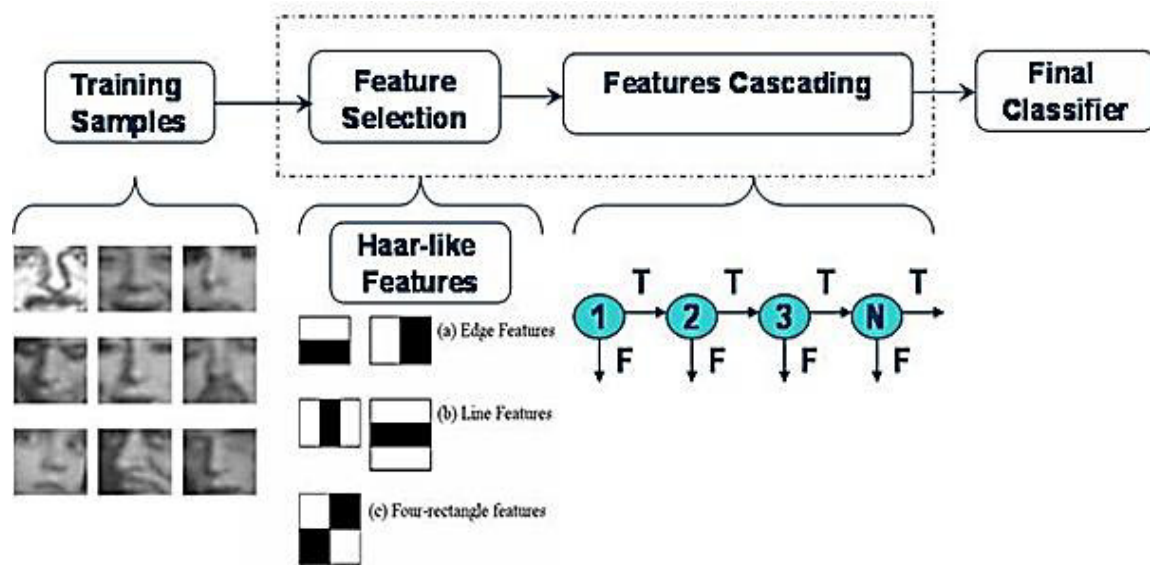


Fig. 3. 21 Algorithme d'apprentissage Adaboost [113].

**3.5.4.2 Détection :** La détection s'applique sur une image de test, dans laquelle on souhaite déceler la présence et la localisation d'un objet. En voici les étapes :

- parcours de l'ensemble de l'image à toutes les positions et échelles, avec une fenêtre de taille  $24 \times 24$  pixels, et application de la cascade à chaque sous-fenêtre, en commençant par le premier étage :
  - calcul des caractéristiques pseudo-Haar utilisées par le classifieur de l'étage courant,
  - puis calcul de la réponse du classifieur,
  - passage ensuite à l'étage supérieur si la réponse est positive, à la sous-fenêtre suivante sinon,
  - et enfin l'exemple est déclaré positif si tous les étages répondent positivement ;
- fusion des détections multiples : l'objet peut en effet générer plusieurs détections, à différentes positions et échelles ; cette dernière étape fusionne les détections qui se chevauchent pour ne retourner qu'un seul résultat.

### 3.5.5 Applications de la méthode de Viola Jones

La méthode de Viola et Jones a essentiellement été appliquée à la détection de visage et à la détection de personne, principalement en raison des nombreuses applications pratiques qu'offrent ces deux domaines, notamment en vidéosurveillance, en indexation d'images et de vidéo ou pour les interfaces homme-machine multimodales. Un exemple d'application grand public de la méthode est donné par les appareils photographiques numériques, où elle sert à

effectuer la mise au point automatique sur les visages. Combinée avec le standard JPEG 2000, la méthode peut également servir à compresser les visages avec un taux de compression plus faible que le reste de l'image, afin de préserver les détails des visages. Les constructeurs automobiles s'intéressent également à la méthode pour concevoir des systèmes de sécurité capables de détecter automatiquement les autres usagers de la route, en particulier les piétons. Des recherches ont également montré que l'efficacité de la méthode ne se limite pas au domaine visible, mais qu'elle s'étend également au domaine infrarouge. La méthode de Viola et Jones a également été utilisée pour détecter d'autres types d'objets, par exemple des mains, pour la commande gestuelle d'une interface homme-machine, des voitures dans des images satellites pour la création de systèmes d'information géographique débarrassés de toute présence visuelle d'automobiles, ou pour l'évaluation et le suivi du trafic routier. La méthode a également été évaluée pour la détection d'avions dans des images de basse résolution à partir d'une caméra embarquée dans un véhicule aérien, pour l'évitement de collisions. Des applications militaires existent aussi pour la détection de cibles (chars, avions) dans des images aériennes ou satellitaires

### 3.5.6 D'autres implémentations de Viola Jones

Il existe de nombreuses implémentations du détecteur de Viola et Jones, la plus utilisée étant celle en C++ présente dans la librairie de vision par ordinateur OpenCV, publiée sous licence BSD. Des implémentations ont été développées pour des environnements ou plates-formes spécifiques, notamment pour une exécution dans des navigateurs Web en utilisant le langage de script ActionScript du logiciel multimédia Flash. Des implémentations matérielles ont également été développées sur ASIC, FPGA et sur GPU. L'utilisation de l'architecture parallèle de ces derniers permet un net gain de temps de détection par rapport à l'implémentation OpenCV traditionnelle. Enfin, les implémentations les plus courantes sont celles rencontrées dans les appareils photographiques numériques pour la mise au point automatique par la détection de visage. Elles nécessitent des optimisations particulières pour faire face à la faible puissance de calcul de ce type de matériel.

### 3.5.7 Mécanisme d'évaluation de la qualité de la performance des algorithmes de recherche d'objets

Tous les paramètres pris en compte dans la mise en œuvre technique de la méthode Viola-Jones déterminent la qualité de la méthode [114]. Les principaux critères d'évaluation de la qualité de la performance est basée sur le calcul de l'exactitude, de l'exhaustivité et de la mesure



F globale pour chaque lancement de la méthode Viola-Jones [115] pour l'ensemble de test sélectionné, ainsi que le temps de traitement (T) pour chaque trame individuelle:

- Précision (Précision) - P est le rapport entre le nombre d'objets correctement récupérés et le nombre total d'objets récupérés.
- Complétude (Rappel) - R est le rapport entre le nombre d'objets correctement récupérés et le nombre total d'objets dans l'ensemble de test.
- F - mesure:

$$F = 2 P R / ( P + R ) \quad (\text{Equ. 3.16})$$

- Introduit pour accomplir la tâche par analogie avec [16] le critère supplémentaire

$$Q_1 = a \times ( 1 - R ) + b \times T \quad (\text{Equ. 3.17})$$

$$Q_2 = a \times ( 1 - F ) + b \times T \quad (\text{Equ. 3.18})$$

- Conformément à les paramètres sélectionnés sont  $a = 1$  et  $b = 1 / \max(T)$ .

Pour déterminer la qualité de l'algorithme, la base FDDB [116] est utilisée. Une caractéristique importante de cette base de données est qu'elle annote non seulement les visages, mais aussi les yeux sur les visages.

### 3.5.8 Partie Experimentale

**3.5.8.1 Description :** Le détecteur d'objets en cascade utilise l'algorithme de Viola-Jones pour détecter les visages, le nez, les yeux, la bouche ou le haut du corps. Vous pouvez également utiliser Training Image Labeler pour former un classificateur personnalisé à utiliser avec cet objet System. Pour plus de détails sur le fonctionnement de la fonction, voir Former un détecteur d'objets en cascade.

#### 3.5.8.2 Construction du détecteur

- Création d'un système objet qui détecte les objets en utilisant l'algorithme de Viola-Jones. La propriété ClassificationModèle contrôle le type d'objet à détecter. Par défaut, le détecteur est configuré pour détecter les visages.
- Création d'un système objet, configuré pour détecter les objets définis par le vecteur de caractères en entrée, MODEL. L'entrée MODEL décrit le type d'objet à détecter. Il existe plusieurs vecteurs de caractères MODEL valides, tels que 'FrontalFaceCART', 'UpperBody' et 'ProfileFace'. Voir la description de la propriété ClassificationModel pour une liste complète des modèles disponibles.
- Création d'un système objet et le configure pour utiliser le modèle de classification personnalisé spécifié avec l'entrée XMLFILE. Le fichier XMLFILE peut être créé en

utilisant la fonction `trainCascadeObjectDetector` ou la fonctionnalité de formation OpenCV (Open Source Computer Vision). Vous devez spécifier un chemin d'accès complet ou relatif au fichier XML, s'il ne se trouve pas sur le chemin MATLAB®.

- Configurer les propriétés de l'objet détecteur d'objet en cascade. Vous spécifiez ces propriétés en tant qu'un ou plusieurs arguments de paire nom-valeur. Les propriétés non spécifiées ont des valeurs par défaut.

**3.5.8.3 Pour détecter une fonctionnalité :** Définir et configurer le détecteur d'objets en cascade en utilisant le constructeur.

### 3.5.8.4 Propriétés

- `ClassificationModèle` - Modèle de classification en cascade formé
- `MinSize` - Taille du plus petit objet détectable
- `MaxSize` - Taille du plus grand objet détectable
- `ScaleFactor` - Mise à l'échelle pour la détection d'objets multi-échelles
- `MergeThreshold` - Seuil de détection
- `UseROI` - Utiliser la région d'intérêt `false` (par défaut) | `vrai`

### 3.5.9 Résultats de la détection par Viola Jones

Les résultats de détection de visage avec la méthode de Viola et Jones sont présentés par la figure 3.22 ci-dessous. Nous obtenons dans ce cas 17 visages sur 19 sont détectés



*Fig. 3. 22 Détection de visage avec la méthode de Viola et Jones.*

3.5.9.1 D'autres exemples de détection par Viola Jones

1) Détection des visages dans une image à l'aide du modèle de classification des faces frontales et en présence de variantes en milieux contrôlés et non contrôlés.

- *Créer un objet détecteur.*  
`faceDetector = vision.CascadeObjectDetector;`
- *Lire l'image d'entrée.*
- *Détecter les visages*
- *Annoter les visages détectés.*



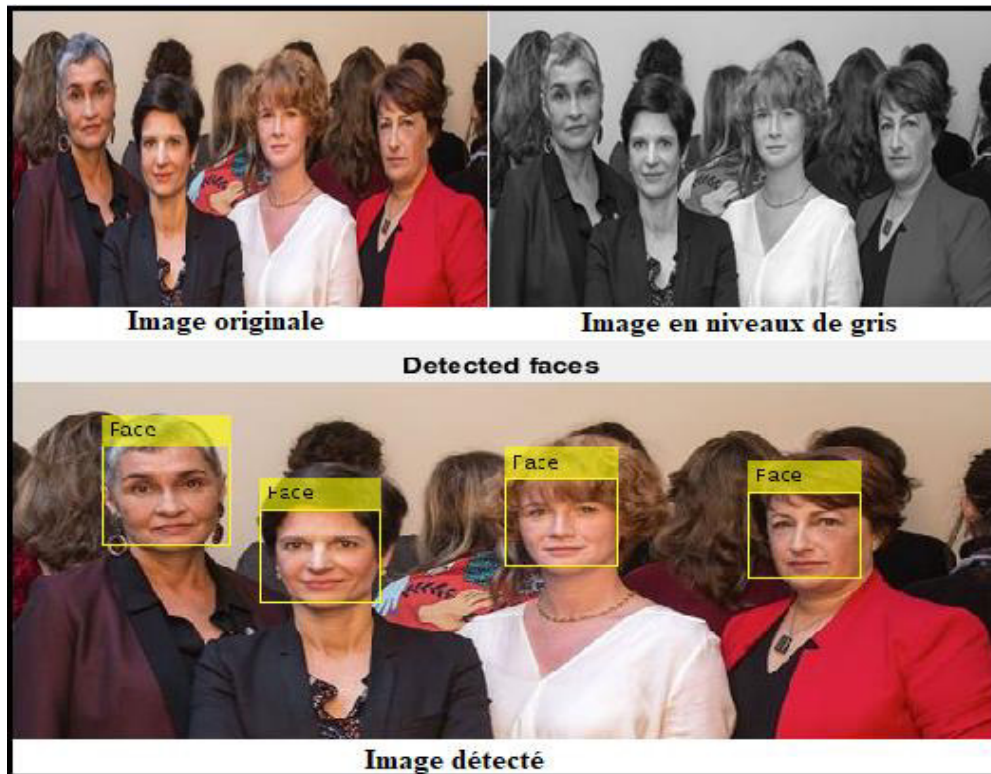
Fig. 3. 23 Exemple de bonne détection sur des images de plusieurs personnes





Fig. 3. 24 Exemple de mauvaise détection sur des images de plusieurs personnes

- 2) Détection Visage à partir de l'image et enregistrer le visage recadré dans un dossier
  - *Créer un objet détecteur.* `faceDetector = vision.CascadeObjectDetector;`
  - *Lire l'image d'entrée*
  - *Filtrer l'image avec filtre de Gabor*
  - *Détecter les visages*
  - *Annoter les visages détectés*
  - *Enregistrer le visage recadré*



*Fig. 3. 25 Images détectées*



*Fig. 3. 26 Images détectées et découpées*

### Discussion

Dans cette exemple nous remarquons que cette méthode de détection de visage présente des limites avec une gamme de situations pour cause de grande variation de pose (surtout de haut en bas) et occlusions.

### Conclusion

Ce chapitre présente une étude comparative de quatre méthodes de détection de visage concernant le taux de détection de visages simples et multiples dans une image en arrière-plan complexe. Les fonctionnalités de méthode Viola-Jones ont donné d'excellents résultats et obtenu un taux de détection plus élevé. Le détecteur de Viola Jones est donc retenu par notre étude. Les limites de Viola-Jones restent de mauvaises detections dans les cas : occlusions, rapprochement de visage, variation de pose. Notre objectif est la detection de visage en milieux

incontrôlés, pour optimiser la détection nous orientons notre étude vers la détection basée sur le Deep Learning . Dans le chapitre quatre nous nous intéressons à l'apprentissage profond en vue d'une meilleure détection

Chapitre 4 Etude et Implémentation du RCNN pour la Détection

**Introduction**

Des recherches récentes dans ce domaine se concentrent davantage sur le problème incontrôlé de détection de visage, où un certain nombre de facteurs tels que les changements de pose, les expressions exagérées et les illuminations extrêmes peuvent entraîner de grandes variations visuelles de l'apparence du visage. Nous rappelons que les difficultés de détection de visage proviennent principalement de deux aspects: 1) les grandes variations visuelles des visages humains dans les arrière-plans encombrés; 2) le grand espace de recherche des positions possibles du visage et des tailles de visage. La première exige que le détecteur de visage traite avec précision un problème de classification binaire alors que le dernier impose en outre une exigence d'efficacité temporelle. Depuis le travail fondateur de Viola et al. [2], la cascade boostée avec des fonctionnalités simples devient la conception la plus populaire et la plus efficace pour la détection de visage pratique. Le détecteur de visage original de Viola-Jones utilise la fonction de Haar qui est rapide à évaluer mais suffisamment discriminante pour les visages frontaux. Cependant, en raison de la nature simple de la fonction de Haar, elle est relativement faible dans l'environnement incontrôlé où les visages sont dans des poses variées, des expressions sous un éclairage inattendu. Un certain nombre d'améliorations au détecteur de visage Viola-Jones ont été proposées au cours de la dernière décennie [116]. La plupart d'entre eux suivent le cadre en cascade boosté avec des fonctionnalités plus avancées. La fonction avancée permet de construire un classificateur binaire plus précis au détriment du calcul supplémentaire. Cependant, le nombre d'étages en cascade requis pour atteindre la précision de détection similaire peut être réduit. Par conséquent, le calcul global peut rester identique ou même réduit en raison du nombre réduit d'étapes en cascade. Cette observation suggère qu'il est possible d'appliquer des fonctionnalités plus avancées dans une solution de détection de visage pratique, à condition que les fausses détections positives puissent être rejetées rapidement dans les premières étapes. Dans ce travail, nous proposons d'appliquer le Convolutional Neural Network (CNN) [117] pour faire face à la problématique de détection. Par rapport aux précédentes fonctionnalités, CNN peut apprendre automatiquement des fonctionnalités pour capturer des variations visuelles complexes en exploitant une grande quantité de données d'entraînement et sa phase de test peut être facilement parallélisée sur les cœurs GPU pour l'accélération.



**4.1 Pourquoi avons-nous besoin de Machine Learning?**

L'apprentissage automatique est nécessaire pour les tâches trop complexes pour que les humains puissent les coder directement. Certaines tâches sont si complexes qu'il est impossible, voire impossible, pour les humains d'élaborer toutes les nuances et de les coder explicitement. Au lieu de cela, nous fournissons une grande quantité de données à un algorithme d'apprentissage automatique et laissons l'algorithme s'en sortir en explorant ces données et en recherchant un modèle qui atteindra ce que les programmeurs ont défini.

**4.2 En savoir plus sur les réseaux de neurones convolutionnels**

Les réseaux de neurones convolutifs (ConvNets) sont des outils largement utilisés pour l'apprentissage en profondeur. Ils sont particulièrement adaptés aux images en tant qu'entrées, bien qu'ils soient également utilisés pour d'autres applications telles que le texte, les signaux et d'autres réponses continues. Ils diffèrent des autres types de réseaux de neurones de plusieurs façons. Les réseaux de neurones convolutionnels sont inspirés de la structure biologique d'un cortex visuel, qui contient des arrangements de cellules simples et complexes. Ces cellules s'activent en fonction des sous-régions d'un champ visuel. Ces sous-régions sont appelées champs réceptifs. Inspirés des résultats de cette étude, les neurones dans une couche convolutive se connectent aux sous-régions des couches avant cette couche au lieu d'être entièrement connectés comme dans d'autres types de réseaux neuronaux. Les neurones ne répondent pas aux zones situées en dehors de ces sous-régions de l'image. Ces sous-régions peuvent se chevaucher, d'où les neurones d'un ConvNet produisent des résultats spatialement corrélés, tandis que dans d'autres types de réseaux neuronaux, les neurones ne partagent aucune connexion et produisent des résultats indépendants. En outre, dans un réseau neuronal avec des neurones entièrement connectés, le nombre de paramètres (poids) peut augmenter rapidement à mesure que la taille de l'entrée augmente. Un réseau de neurones convolutif réduit le nombre de paramètres avec le nombre réduit de connexions, les poids partagés et le sous-échantillonnage. Un réseau ConvNet se compose de plusieurs couches, telles que des couches convolutives, des couches de mise en pool ou de regroupement maximal et des couches entièrement connectées.

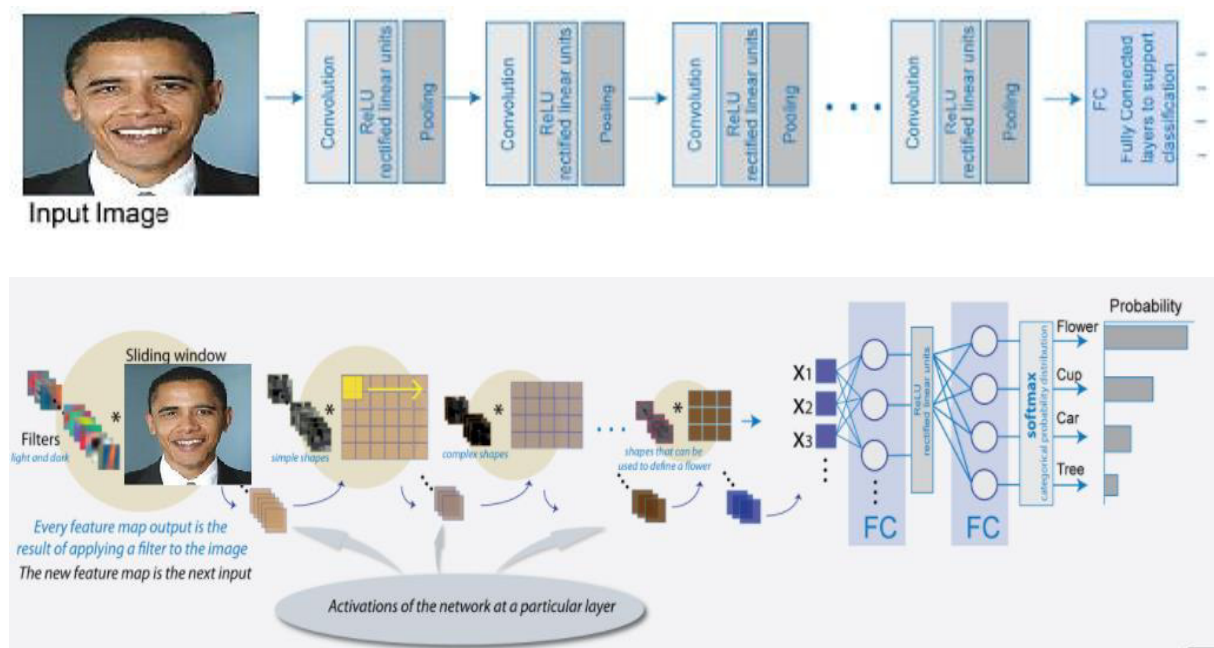


Fig. 4. 1 Schéma d'une architecture du ConvNet [118]

Les neurones dans chaque couche d'un ConvNet sont disposés de manière 3D, transformant une entrée 3D en une sortie 3D. Par exemple, pour une entrée d'image, la première couche (couche d'entrée) contient les images en tant qu'entrées 3D, les dimensions étant la hauteur, la largeur et les canaux de couleur de l'image. Les neurones de la première couche convolutionnelle se connectent aux régions de ces images et les transforment en une sortie 3D. Les unités cachées (neurones) de chaque couche apprennent des combinaisons non linéaires des entrées d'origine, ce que l'on appelle l'extraction de caractéristiques. Ces fonctions apprises, également appelées activations, d'une couche deviennent les entrées de la couche suivante. Enfin, les fonctions apprises deviennent les entrées du classificateur ou la fonction de régression à la fin du réseau. L'architecture d'un réseau ConvNet peut varier en fonction des types et des nombres de couches incluses. Les types et le nombre de couches incluses dépendent de l'application ou des données particulières. Par exemple, si vous avez des réponses catégoriques, vous devez avoir une fonction de classification et une couche de classification, alors que si votre réponse est continue, vous devez avoir une couche de régression à la fin du réseau. Un réseau plus petit avec seulement une ou deux couches de convolution peut être suffisant pour apprendre un petit nombre de données d'image en échelle de gris. D'un autre côté, pour des données plus complexes avec des millions d'images colorées, vous aurez peut-être besoin d'un réseau plus complexe avec plusieurs couches convolutives et entièrement connectées.

### 4.3 Réseau neuronal convolutif

En apprentissage automatique, un réseau de neurones convolutifs ou réseau de neurones à convolution (en anglais *CNN* ou *ConvNet* pour *Convolutional Neural Networks*) est un type de réseau de neurones artificiels acycliques (*feed-forward*), dans lequel le motif de connexion entre les neurones est inspiré par le cortex visuel des animaux. Les neurones de cette région du cerveau sont arrangés de sorte qu'ils correspondent à des régions qui se chevauchent lors du pavage du champ visuel. Leur fonctionnement est inspiré par les processus biologiques, ils consistent en un empilage multicouche de perceptrons, dont le but est de prétraiter de petites quantités d'informations. Les réseaux neuronaux convolutifs ont de larges applications dans la reconnaissance d'image et vidéo, les systèmes de recommandation et le traitement du langage naturel.

**4.3.1 Présentation :** Considérons l'analyse d'une image monochrome (en 2 dimensions, largeur et hauteur) ou en couleur (en 3 dimensions, en considérant l'image RVB avec 3 unités de profondeurs, dont la troisième correspond à l'empilement de 3 images selon chaque couleur, rouge, verte et bleue). Un réseau neuronal convolutif se compose de deux types de neurones artificiels, agencés en « couches » traitant successivement l'information :

- les neurones de traitement, qui traitent une portion limitée de l'image (appelée « champ réceptif ») au travers d'une fonction de convolution.
- les neurones de mise en commun des sorties dits de pooling (totale ou partielle).

Un traitement correctif non-linéaire et ponctuel peut être appliqué entre chaque couche pour améliorer la pertinence du résultat. L'ensemble des sorties d'une couche de traitement permet de reconstituer une image intermédiaire, qui servira de base à la couche suivante.



*Fig. 4. 2 Exemple d'image (gauche) et de son traitement par DeepDream (droite) [119].*

### 4.3.2 Traitement convolutif

#### 4.3.2.1 Traitement de profondeur 1 et sans chevauchement (facilitant la compréhension)

Dans le cadre de la reconnaissance d'image, cette dernière est « pavée », c'est-à-dire découpée en petites zones (appelées tuiles). Chaque tuile sera traitée individuellement par un neurone artificiel (qui effectue une opération de filtrage classique en associant un poids à chaque pixel de la tuile). Tous les neurones ont les mêmes paramètres de réglage. Le fait d'avoir le même traitement (mêmes paramètres), légèrement décalé pour chaque champ réceptif, s'appelle une convolution. Cette strate de neurones avec les mêmes paramètres est appelée « noyau de convolution ». Les pixels d'une tuile sont analysés globalement. Dans le cas d'une image en couleur, un pixel contient 3 entrées (rouge, vert et bleu), qui seront traitées globalement par chaque neurone. Donc l'image peut être considérée comme un volume, et notée par exemple  $30 \times 10 \times 3$  pour 30 pixels de largeur, 10 de hauteur et 3 de profondeur correspondant aux 3 canaux rouge, vert et bleu. De manière générale, on parlera de « volume d'entrée ».

**4.3.2.2 Généralisation (profondeur et chevauchement) :** Dans les faits, la zone analysée est légèrement plus grande que la tuile et est appelée « champ récepteur ». Les champs récepteurs se chevauchent donc, afin d'obtenir une meilleure représentation de l'image originale ainsi qu'une meilleure cohérence du traitement au fil des couches de traitement, le chevauchement est défini par le pas (décalage entre deux champs récepteurs adjacents). Un noyau de convolution va analyser une caractéristique de l'image d'entrée. Pour analyser plusieurs caractéristiques, on va empiler des strates de noyaux de convolution indépendants, chaque strate analysant une caractéristique de l'image. L'ensemble des strates ainsi empilées forme la « couche de traitement convolutif », qu'il faut voir en fait comme un volume (souvent appelé « volume de sortie »). Le nombre de strates de traitement s'appelle la profondeur de la couche de convolution (à ne pas confondre avec la profondeur d'un réseau de neurones convolutifs qui compte le nombre de couches de convolution). Une couche de convolution permet de traiter un volume d'entrée pour fournir un volume de sortie. On peut également assimiler le volume de sortie à une image intermédiaire. Pour formuler les choses différemment, dans un réseau de neurones convolutifs, chaque champ récepteur est traité par un perceptron monocouche. Et tous les perceptrons monocouche associés à l'ensemble des champs récepteurs sont paramétrés de manière identique.

**4.3.2.3 Caractéristiques et avantages :** Un avantage majeur des réseaux convolutifs est l'utilisation d'un poids unique associé aux signaux entrant dans tous les neurones d'un même noyau de convolution. Cette méthode réduit l'empreinte mémoire, améliore les performances et

permet une invariance du traitement par translation. C'est le principal avantage du réseau de neurones convolutifs par rapport au perceptron multicouche, qui, lui, considère chaque neurone indépendant et affecte donc un poids différent à chaque signal entrant. Lorsque le volume d'entrée varie dans le temps (vidéo ou son), il devient intéressant de rajouter un paramètre le long de l'échelle de temps dans le paramétrage des neurones. On parlera dans ce cas de réseau neuronal à retard temporel (TDNN).

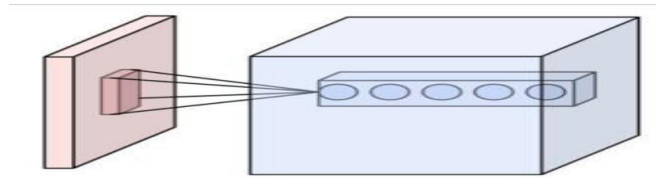
Comparés à d'autres algorithmes de classification d'image, les réseaux de neurones convolutifs utilisent relativement peu de pré-traitement. Cela signifie que le réseau est responsable de faire évoluer tout seul ses propres filtres (*apprentissage sans supervision*), ce qui n'est pas le cas d'autres algorithmes plus traditionnels. *L'absence de paramétrage initial et d'intervention humaine est un atout majeur des CNN.*

**4.3.2.4 Blocs de construction :** Une architecture de réseau de neurones convolutifs est formée par un empilement de couches de traitement :

- la couche de convolution (**CONV**) qui traite les données d'un champ réceptif ;
- la couche de pooling (**POOL**), qui permet de compresser l'information en réduisant la taille de l'image intermédiaire (souvent par sous-échantillonnage) ;
- la couche de correction (**ReLU**), souvent appelée par abus « ReLU » en référence à la fonction d'activation (Unité de rectification linéaire) ;
- la couche « entièrement connectée » (**FC**), qui est une couche de type perceptron ;
- la couche de perte (**LOSS**).

Vous pouvez définir les couches d'un réseau de neurones convolutionnels dans MATLAB® dans un format de tableau.

- **Couche d'entrée d'image :** La couche d'entrée d'image définit la taille des images d'entrée d'un réseau de neurones convolutif et contient les valeurs de pixels brutes des images. Vous pouvez ajouter une couche d'entrée à l'aide de la fonction *imageInputLayer*. Spécifiez la taille de l'image à l'aide de l'argument *inputSize*. La taille d'une image correspond à la hauteur, la largeur et le nombre de canaux de couleur de cette image. Par exemple, pour une image en niveaux de gris, le nombre de canaux est 1, et pour une image en couleur, il est 3. Cette couche peut également effectuer une normalisation des données en soustrayant l'image moyenne du jeu d'apprentissage de chaque image d'entrée.
- **Couche convolutionnelle :** c'est la couche qui traite les données d'un champ réceptif



**Fig. 4. 3** Ensemble de neurones (cercles) créant la profondeur d'une couche de convolution (bleu). Ils sont liés à un même champ réceptif (rouge)[120].

**4.3.2.5 Filtres et foulée :** Une couche convolutive est constituée de neurones qui se connectent aux sous-régions des images d'entrée ou aux sorties de la couche qui les précède. Une couche convolutionnelle apprend les caractéristiques localisées par ces régions lors de la numérisation d'une image. La taille de ces régions peut être spécifiée à l'aide de l'argument d'entrée *filterSize* lorsqu'on crée la couche à l'aide de la fonction *convolution2dLayer*. Pour chaque région, la fonction *trainNetwork* calcule un produit scalaire des poids et de l'entrée, puis ajoute un terme de biais. Un ensemble de poids appliqués à une région de l'image est appelé un filtre. Le filtre se déplace le long de l'image d'entrée verticalement et horizontalement, en répétant le même calcul pour chaque région, c'est-à-dire en convolvant l'entrée. La **taille du pas** avec lequel il se déplace s'appelle une **foulée**. On peut spécifier cette taille de pas avec l'argument de paire nom-valeur *Stride*. Ces régions locales auxquelles les neurones se connectent peuvent se chevaucher en fonction des valeurs *filterSize* et '*Stride*'.

Le nombre de poids utilisé pour un filtre est  $h * w * c$ , où  $h$  : est la hauteur, et  $w$  : est la largeur de la taille du filtre, et  $c$  : est le nombre de canaux dans l'entrée. Le nombre de filtres détermine le nombre de canaux dans la sortie d'une couche convolutionnelle. Le nombre de filtres est Indiqué à l'aide de l'argument *numFilters* de *convolution2dLayer*.

**4.3.2.6 Fonctionnalités :** Lorsqu'un filtre se déplace le long de l'entrée, il utilise le même ensemble de poids et de biais pour la convolution, formant une carte de caractéristiques. Par conséquent, le nombre de mappages d'entités qu'une couche convolutionnelle possède est égal au nombre de filtres (nombre de canaux). Chaque carte de caractéristiques a un ensemble différent de poids et un biais. Ainsi, le nombre total de paramètres dans une couche convolutionnelle est  $((h * w * c + 1) * \text{Nombre de filtres})$ , où 1 est pour le biais.

**4.3.2.7 Zéro remplissage :** On peut également appliquer un remplissage nul pour saisir les bordures d'image verticalement et horizontalement à l'aide de l'argument de la paire (nom,valeur) '*Remplissage*'. Le remplissage consiste essentiellement à ajouter des lignes ou des colonnes de zéros aux bordures d'une entrée d'image. Cela aide à contrôler la taille de sortie de la couche.

Taille de sortie : la hauteur et la largeur de sortie d'une couche convolutionnelle est (Taille d'entrée - Taille du filtre + 2\*Rembourrage) / Foulée + 1. Cette valeur doit être un entier pour que l'image entière soit entièrement couverte. Si la combinaison de ces paramètres ne permet pas de couvrir entièrement l'image, le logiciel ignore par défaut la partie restante de l'image le long des bords droit et inférieur de la convolution.

**4.3.2.8 Nombre de neurones :** Le produit de la hauteur et de la largeur de sortie donne le nombre total de neurones dans une carte de caractéristiques, par exemple la taille de la carte. Le nombre total de neurones (taille de sortie) dans une couche convolutive est alors égal à : (Taille de la carte \* Nombre de filtres). Par exemple, supposons que l'image d'entrée est une image couleur 32 x 32 x 3. Pour une couche convolutionnelle avec 8 filtres et une taille de filtre de 5 par 5, le nombre de poids par filtre est de  $5 * 5 * 3 = 75$ , et le nombre total de paramètres dans la couche est  $(75 + 1) * 8 = 608$ . Si la foulée est de 2 dans chaque direction et qu'il y a deux pixels de remplissage nul, alors chaque carte de caractéristiques est de 16 par 16. C'est parce que  $(32 - 5 + 2 * 2) / 2 + 1 = 16.5$ , et donc une partie du remplissage de zéro le plus à droite vers la droite et le bas de l'image est rejetée. Enfin, le nombre total de neurones dans la couche est  $16 * 16 * 8 = 2048$ .

**4.3.2.9 Paramètres d'apprentissage :** On peut également ajuster les taux d'apprentissage et les paramètres de régularisation pour cette couche en utilisant les arguments de la paire (nom,valeur) associés lors de la définition de la couche convolutionnelle. Si on choisit de ne pas les ajuster, *trainNetwork* utilise les paramètres d'entraînement globaux définis par la fonction *trainingOptions*. Un réseau neuronal convolutif peut consister en une ou plusieurs couches convolutionnelles. Le nombre de couches convolutives dépend de la quantité et de la complexité des données.

### 4.3.3 Couche de normalisation des lots

Des couches de normalisation par lots sont utilisées entre les couches convolutives et les non-linéarités telles que les couches ReLU pour accélérer l'apprentissage du réseau et réduire la sensibilité à l'initialisation du réseau. La couche normalise d'abord les activations de chaque canal en soustrayant la moyenne du mini-lot et en divisant par l'écart-type du mini-lot. Ensuite, la couche décale l'entrée d'un décalage  $\beta$  et la met à l'échelle par un facteur d'échelle  $\gamma$ .  $\beta$  et  $\gamma$  sont eux-mêmes des paramètres à apprendre qui sont mis à jour lors de l'apprentissage en réseau. Une couche de normalisation par lots est créé à l'aide de *batchNormalizationLayer*.

Les couches de normalisation par lots normalisent les activations et les gradients se propageant à travers un réseau de neurones, ce qui facilite l'apprentissage en réseau. Pour tirer pleinement parti de ce fait et essayer d'augmenter le taux d'apprentissage. Comme le problème d'optimisation est plus facile, les mises à jour des paramètres peuvent être plus importantes et le réseau peut apprendre plus rapidement. On peut également essayer de réduire la régularisation de  $L_2$  et régularisation de décrochage. Avec les couches de normalisation par lots, les activations d'une image spécifique ne sont pas déterministes, mais dépendent plutôt des images qui apparaissent dans le même mini-lot. Pour profiter pleinement de cet effet de régularisation, on mélange les données d'entraînement avant chaque période d'entraînement. Pour spécifier la fréquence à laquelle les données doivent être mélangées au cours de la formation, l'argument "*Shuffle*" prend une valeur de *formationOptions* est utilisé.

#### 4.3.4 ReLU Layer

Les couches de convolution et de normalisation par lots sont généralement suivies d'une fonction d'activation non linéaire telle qu'une unité linéaire rectifiée (ReLU), spécifiée par une couche ReLU. Une couche ReLU est réé à l'aide de la fonction *reluLayer*. Une couche ReLU effectue une opération de seuil sur chaque élément, où toute valeur d'entrée inférieure à zéro

est mise à zéro, c'est-à-dire

$$f(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

La couche ReLU ne change pas la taille de son entrée. Il existe des extensions de la couche standard ReLU qui effectuent des opérations légèrement différentes et peuvent améliorer les performances pour certaines applications. Une couche ReLU qui multiplie les valeurs d'entrée inférieures à zéro par un scalaire fixe, ce qui permet aux entrées négatives de «fuite» dans la sortie. On utilise la fonction *leakyReluLayer* pour créer une couche ReLU. Une couche ReLU écrêtée définit les entrées négatives sur zéro, mais définit également les valeurs d'entrée au-dessus d'un plafond de découpage égal à ce plafond de découpage. Cette coupure empêche la sortie de devenir trop grande. La fonction *clippedReluLayer* est utilisée pour créer une couche ReLU découpée.

#### 4.3.5 Couche de normalisation de canal (*normalisation de réponse locale*)

Cette couche effectue une normalisation de réponse locale par canal. Il suit généralement la couche d'activation ReLU. Cette couche est créé à l'aide de la fonction *crossChannelNormalizationLayer*. Cette couche remplace chaque élément par une valeur normalisée qu'il obtient en utilisant les éléments d'un certain nombre de canaux voisins



(éléments de la fenêtre de normalisation). C'est-à-dire que, pour chaque élément  $x$  de l'entrée, *trainNetwork* calcule une valeur normalisée  $x'$  en utilisant

$$x' = \frac{x}{\left(k + \frac{\alpha * ss}{windowChanelSize}\right)^\beta} \quad \text{(Equ. 4.1)}$$

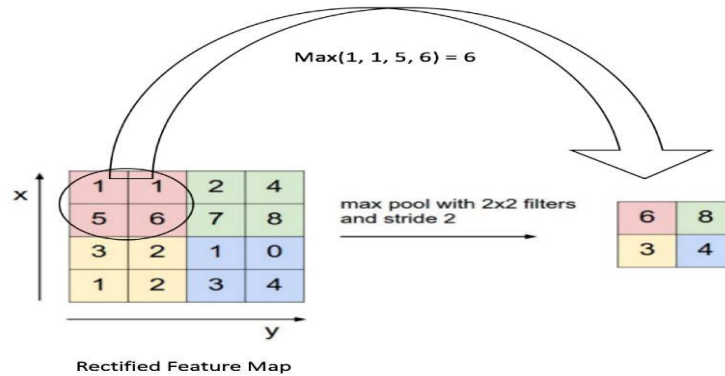
où  $K$ ,  $\alpha$  et  $\beta$  sont les hyperparamètres dans la normalisation, et  $ss$  : est la somme des carrés des éléments dans la fenêtre de normalisation [121]. On doit spécifier la taille de la fenêtre de normalisation à l'aide de l'argument *windowChannelSize* de la fonction *crossChannelNormalizationLayer*. On peut également spécifier les hyperparamètres à l'aide des arguments de paires nom-valeur Alpha, Bêta et K. La formule de normalisation précédente est légèrement différente de ce qui est présenté dans [121]. On peut obtenir la formule équivalente en multipliant la valeur *alpha* par *windowChannelSize*.

#### 4.3.6 Couches Max et Moyenne-Pooling

Les couches de regroupement maximum et moyen suivent les couches convolutionnelles pour le sous-échantillonnage, réduisant ainsi le nombre de connexions aux couches suivantes (généralement une couche entièrement connectée). Ils n'effectuent pas d'apprentissage eux-mêmes, mais réduisent le nombre de paramètres à apprendre dans les couches suivantes. Ils aident également à réduire le surapprentissage. On peut créer ces couches en utilisant les fonctions *maxPooling2dLayer* et *averagePooling2dLayer*. Une couche *max-pooling* renvoie les valeurs maximales des régions rectangulaires de son entrée. La taille des régions rectangulaires est déterminée par l'argument *poolSize* de *maxPoolingLayer*. Par exemple, si *poolSize* est égal à [121,122], la couche renvoie la valeur maximale dans les régions de hauteur 2 et de largeur 3. De même, la couche de mise en commun moyenne produit les valeurs moyennes des régions rectangulaires de son entrée. La taille des régions rectangulaires est déterminée par l'argument *poolSize* de *averagePoolingLayer*. Par exemple, si *poolSize* est [121,122], la couche renvoie la valeur moyenne des régions de hauteur 2 et de largeur 3. Les fonctions *maxPoolingLayer* et *averagepoolingLayer* parcourent l'entrée horizontalement et verticalement par pas que vous pouvez spécifier en utilisant argument de la paire (nom,valeur) de chaque fonction. Si *poolSize* est inférieur ou égal à *Stride*, les régions de regroupement ne se chevauchent pas.

Pour les régions non chevauchantes (*poolSize* et *Stride* sont égaux), si l'entrée de la couche de regroupement est n-par-n et que la taille de la région de regroupement est h-par-h, la couche de

regroupement échantillonne les régions par  $h$  [123]. c'est-à-dire que la sortie d'une couche de mise en commun maximale ou moyenne pour un canal d'une couche convolutionnelle est  $n / h$  par  $n / h$ . Pour les régions qui se chevauchent, la sortie d'une couche de regroupement est  $(Input\ Size - Pool\ Size + 2 * Padding) / Stride + 1$ .



**Fig. 4. 4** Max pooling avec un filtre  $2 \times 2$  et un pas de 2 [122]

#### 4.3.7 Couche entièrement connectée

Les couches de convolution (et d'échantillonnage inférieur) sont suivies par une ou plusieurs couches entièrement connectées. Une couche entièrement connectée est créé à l'aide de la fonction *fullyConnectedLayer*. Comme son nom l'indique, tous les neurones d'une couche entièrement connectée se connectent à tous les neurones de la couche précédente. Cette couche combine toutes les caractéristiques (informations locales) apprises par les couches précédentes à travers l'image pour identifier les motifs plus grands. Pour les problèmes de classification, la dernière couche entièrement connectée combine les fonctionnalités pour classer les images. C'est la raison pour laquelle l'argument *outputSize* de la dernière couche entièrement connectée du réseau est égal au nombre de classes de l'ensemble de données. Pour les problèmes de régression, la taille de sortie doit être égale au nombre de variables de réponse. On peut également ajuster le taux d'apprentissage et les paramètres de régularisation pour cette couche en utilisant les arguments de la paire (nom,valeur) associés lors de la création de la couche entièrement connectée. Si on choisit de ne pas les ajuster, alors *TrainNetwork* utilise les paramètres d'entraînement globaux définis par la fonction *trainingOptions*.

#### 4.3.8 Couches de sortie

**4.3.8.1 Softmax** : Pour les problèmes de classification, une couche *softmax* et ensuite une couche de classification doivent suivre la couche finale entièrement connectée. On peut créer ces couches à l'aide des fonctions *softmaxLayer* et *classificationLayer*, respectivement. La fonction d'activation de l'unité de sortie est la fonction *softmax*:

$$y_r(x) = \frac{\exp(a_r(x))}{\sum_{j=1}^k \exp(a_j(x))} \quad \text{où} \quad 0 \leq y_r \leq 1 \quad \text{et} \quad \sum_{j=1}^k y_{j=1} = 1 \quad \text{(Equ. 4.2)}$$

La fonction softmax est la fonction d'activation de l'unité de sortie après la dernière couche entièrement connectée pour les problèmes de classification multi-classes:

$$p(c_r | x, \theta) = \frac{p(x, \theta | c_r) p(c_r)}{\sum_{j=1}^k p(x, \theta | c_j) p(c_j)} = \frac{\exp(a_r(x, \theta))}{\sum_{j=1}^k \exp(a_j(x, \theta))} \quad \text{(Equ. 4.3)}$$

où  $0 \leq p(c_r | x, \theta) \leq 1$  et  $\sum_{j=1}^k p(c_j | x, \theta) = 1$  et  $\theta$  est le vecteur de paramètres

de plus  $a_r = \ln(p(x, \theta | c_r) p(c_r))$ ,  $p(x, \theta | c_r)$  est la probabilité conditionnelle de l'échantillon donné classe  $r$  et  $p(c_r)$  est la probabilité prioritaire de la classe. La fonction *softmax* est également connue sous le nom d'exponentielle normalisée et peut être considérée comme la généralisation multi-classe de la fonction sigmoïde logistique [124].

**4.3.8.2 couches de classification :** Une couche de sortie de classification doit suivre la couche softmax. Dans la couche de sortie de classification, *trainNetwork* prend les valeurs de la fonction *softmax* et affecte chaque entrée à l'une des  $k$  classes mutuellement exclusives en utilisant la fonction d'entropie croisée pour un schéma de codage 1-de- $k$  [124]:

$$E(\theta) = -\sum_{i=1}^n \sum_{j=1}^k t_{ij} \ln y_j(x_i, \theta) \quad \text{(Equ. 4.4)}$$

où  $t_{ij}$  est l'indicateur que le  $i$ th échantillon appartient à la  $j$ th classe,  $\theta$  est le vecteur de paramètres,  $y_j(x_i, \theta)$  est la sortie pour l'échantillon  $i$  qui, dans ce cas, est la valeur de la fonction *softmax*. Autrement dit, c'est la probabilité que le réseau associe l'entrée  $i$ th à la classe  $j$ .  $p(t_j = 1 | x_i)$

### 4.3.9 Couche de régression

On peut également utiliser *ConvNets* pour les problèmes de régression, où la variable cible (sortie) est continue. Dans de tels cas, une couche de sortie de régression doit suivre la couche finale entièrement connectée. On peut créer une couche de régression à l'aide de la fonction *regressionLayer*. La fonction de perte par défaut pour une couche de régression est l'erreur quadratique moyenne:

$$MSE = E(\theta) = \sum_{i=1}^n \frac{(t_i - y_i)^2}{n} \quad \text{(Equ. 4.5)}$$

Où  $t_i$  est la sortie cible, et  $y_i$  est la prédiction du réseau pour la variable de réponse correspondant à l'observation  $i$

#### 4.4 Architecture de modèles de CNN

La forme la plus commune d'une architecture de réseau de neurones convolutifs empile quelques couches **Conv-ReLU**, les suit avec des couches **Pool**, et répète ce schéma jusqu'à ce que l'entrée soit réduite dans un espace d'une taille suffisamment petite. À un moment, il est fréquent de placer des couches entièrement connectées (**FC**). La dernière couche entièrement connectée est reliée vers la sortie. Voici quelques architectures communes de réseau de neurones convolutifs qui suivent ce modèle :

- INPUT -> FC

Implémente un classifieur linéaire

- INPUT -> CONV -> RELU -> FC

- INPUT -> [CONV -> RELU -> POOL] \* 2 -> FC -> RELU -> FC

Ici, il y a une couche de CONV unique entre chaque couche POOL

- INPUT -> [CONV -> RELU -> CONV -> RELU -> POOL] \* 3 -> [FC -> RELU] \* 2 -> FC

Ici, il y a deux couches CONV empilées avant chaque couche POOL.

L'empilage des couches CONV avec de petits filtres de pooling (plutôt un grand filtre) permet un traitement plus puissant, avec moins de paramètres. Cependant, avec l'inconvénient de demander plus de puissance de calcul (pour contenir tous les résultats intermédiaires de la couche CONV). La figure 4.5 suivante illustre une architecture standard d'un réseau à convolution.

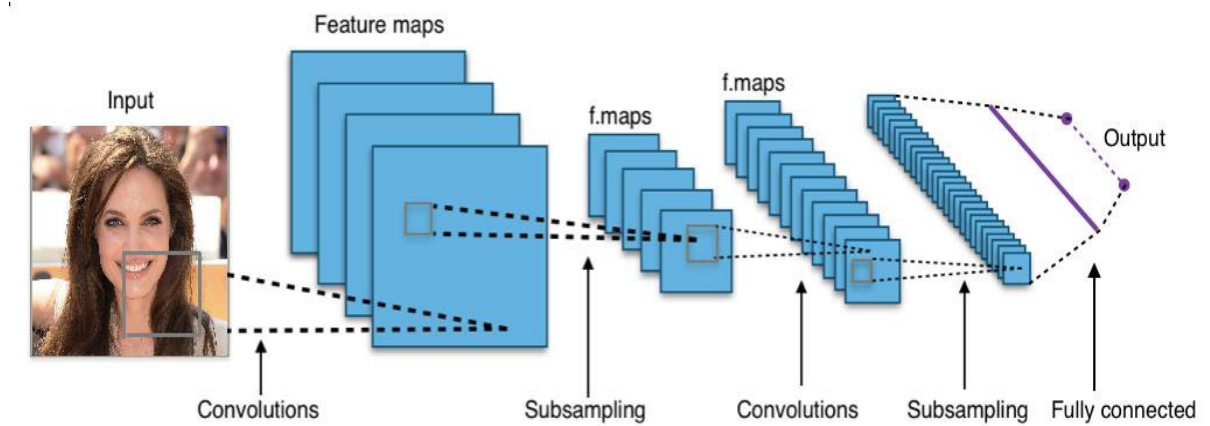


Fig. 4. 5 Architecture standard d'un réseau à convolutions.

#### 4.5 Configuration des paramètres et formation d'un réseau neuronal convolutif

Après avoir défini les couches du réseau neuronal comme décrit dans la section précédente pour spécifier les couches du réseau neuronal convolutif, l'étape suivante consiste à configurer les options de formation pour le réseau. Utilisons la fonction *trainingOptions* pour définir les paramètres d'entraînement globaux. Pour former un réseau, utilisons l'objet renvoyé par *trainingOptions* en tant qu'argument d'entrée de la fonction *trainNetwork*. Les couches avec des paramètres apprenants ont également des options pour ajuster les paramètres d'apprentissage.

##### 4.5.1 Spécification du solveur et du nombre maximum d'époques

La fonction *trainNetwork* peut utiliser différentes variantes de descente de gradient stochastique pour entraîner le réseau. On spécifie l'algorithme d'optimisation en utilisant l'argument *solverName* de *trainingOptions*. Pour minimiser la perte, ces algorithmes mettent à jour les paramètres du réseau en faisant de petits pas dans la direction du gradient négatif de la fonction de perte. Différents solveurs fonctionnent mieux pour différents problèmes. Les solveurs mettent à jour les paramètres en utilisant un sous-ensemble des données à chaque étape. Ce sous-ensemble est appelé un mini-lot. On peut spécifier la taille du mini-lot à l'aide de l'argument de la paire (nom,valeur) '*formationMinBatchSize*' de *formationOptions*. Chaque *mise à jour* de paramètre est appelée une *itération*. Un passage complet à travers l'ensemble des données est appelé *une époque*. le nombre maximal d'époques pour l'apprentissage est spécifié en utilisant l'argument de la paire (nom,valeur) '*maxEpochs*' de *formation Options*. La valeur par défaut est 30, mais on peut choisir un plus petit nombre d'époques pour les petits réseaux ou pour l'apprentissage de réglage fin et de transfert, où la majeure partie de l'apprentissage est déjà effectuée. Par défaut, le logiciel mélange les données une fois avant la formation. Ce paramètre peut être modifier en utilisant l'argument de la paire (nom,valeur) '*Shuffle*'.

##### 4.5.2 Spécification et modification du taux d'apprentissage

On peut spécifier le taux d'apprentissage global en utilisant l'argument *pair-valeur* '*initialLearnRate*' de *trainingOptions*. Par défaut, *trainNetwork* utilise cette valeur tout au long du processus de formation. le taux d'apprentissage peut être modifier à un certain nombre d'époques en multipliant le taux d'apprentissage par un facteur. Au lieu d'utiliser un petit taux d'apprentissage fixe tout au long du processus de formation, on peut choisir un taux d'apprentissage plus élevé au début de la formation et réduire progressivement cette valeur pendant l'optimisation. Cela peut raccourcir le temps d'entraînement, tout en permettant des pas

plus petits vers le minimum de perte à mesure que l'entraînement progresse. Pour réduire progressivement le taux d'apprentissage, on utilise l'argument de la paire (nom,valeur) "*LearnRateSchedule*", "*piecewise*". Une fois que cette option est choisie, *trainNetwork* multiplie le taux d'apprentissage initial par un facteur de 0,1 toutes les 10 époques. On peut spécifier le facteur permettant de réduire le taux d'apprentissage initial et le nombre d'époques en utilisant respectivement les arguments de paire nom-valeur '*LearnRateDropFactor*' et '*LearnRateDropPeriod*'.

#### 4.5.3 Spécification des données de validation

Pour effectuer une validation réseau au cours de la formation, les données de validation sont spécifiées à l'aide de l'argument paire de noms-valeurs '*ValidationData*' de *formationOptions*. Par défaut, *trainNetwork* valide le réseau toutes les 50 itérations en prédisant la réponse des données de validation et en calculant la perte de validation et la précision (erreur quadratique moyenne pour les réseaux de régression). On peut modifier la fréquence de validation à l'aide de l'argument de la paire (nom,valeur) '*ValidationFrequency*'. Si le réseau a des couches qui se comportent différemment pendant la prédiction que pendant l'entraînement (par exemple, les couches de décrochage), la précision de la validation peut être supérieure à la précision de l'entraînement (mini-lot). L'apprentissage en réseau s'arrête automatiquement lorsque la perte de validation cesse de s'améliorer. Par défaut, si la perte de validation est supérieure ou égale à la perte précédemment la plus faible cinq fois de suite, l'entraînement réseau s'arrête. Effectuer des validations à intervalles réguliers pendant l'entraînement aide à déterminer si le réseau sur-mesure les données d'entraînement. Un problème courant est que le réseau «mémorise» simplement les données d'apprentissage, plutôt que d'apprendre des caractéristiques générales qui lui permettent de faire des prédictions précises pour de nouvelles données. Pour vérifier si le réseau est en sur-adaptation, il faut comparer la perte et l'exactitude de la formation avec les mesures de validation correspondantes. Si la perte d'entraînement est significativement inférieure à la perte de validation, ou si la précision de l'entraînement est significativement supérieure à la précision de la validation, alors le réseau est sur-équipé.

Pour réduire le surajustement, on peut essayer d'ajouter une augmentation de données. Il faut utiliser un *augmentedImageDatastore* pour effectuer des transformations aléatoires sur les images d'entrée. Cela aide à empêcher le réseau de mémoriser la position exacte et l'orientation des objets. On peut également essayer de réduire la taille du réseau en réduisant le nombre de couches ou de filtres convolutifs, en augmentant la régularisation  $L_2$  à l'aide de l'argument de la paire (nom,valeur) '*L2Regularization*' ou en ajoutant des couches supprimées.

#### 4.5.4 Sélection d'une ressource matérielle

Si un GPU est disponible, alors *trainNetwork* l'utilise pour la formation, par défaut. Sinon, *trainNetwork* utilise un processeur. L'environnement d'exécution souhaité est également spécifier à l'aide de l'argument de la paire (nom,valeur) '*ExecutionEnvironment*'. On peut spécifier un seul processeur ('cpu'), un seul GPU ('gpu'), plusieurs GPU ('multi-gpu'), ou un pool parallèle local ou un cluster de calcul ('parallel'). Toutes les options autres que 'cpu' nécessitent Parallel Computing Toolbox™. La formation sur un GPU requiert un GPU compatible CUDA® avec une capacité de calcul de 3.0 ou supérieure.

#### 4.5.5 Enregistrer les réseaux de points de contrôle et reprendre la formation

Neural Network Toolbox™ permet de sauvegarder les réseaux sous forme de fichiers .mat après chaque époque pendant l'entraînement. Cette sauvegarde périodique est particulièrement utile lorsque on a un grand réseau ou un grand ensemble de données, et la formation prend beaucoup de temps. Si l'entraînement est interrompu pour une raison quelconque, on peut reprendre l'entraînement depuis le dernier réseau de points de contrôle enregistré. Si on souhaite que *trainNetwork* enregistre les réseaux de point de contrôle, on doit spécifier le nom du chemin en utilisant l'argument de la paire (nom,valeur) '*trainingpoint*' de '*CheckpointPath*'. Si le chemin que spécifié n'existe pas, alors *trainingOptions* renvoie une erreur. *trainNetwork* attribue automatiquement des noms uniques aux fichiers réseau de points de contrôle, par exemple, convnet\_checkpoint\_\_45\_\_2018\_05\_09\_\_17\_04\_33.mat. Dans cet exemple, 45 est le nombre d'itération, 2018\_05\_09 est la date et 17\_04\_33 est l'heure à laquelle *trainNetwork* enregistre le réseau. On peut charger un fichier réseau de point de contrôle en double-cliquant dessus ou en utilisant la commande load sur la ligne de commande. Par exemple:

```
charger convnet_checkpoint__45__2018_05_09__17_04_33.mat
```

On peut ensuite reprendre l'apprentissage en utilisant les couches du réseau comme argument d'entrée de *trainNetwork*. On doit spécifier manuellement les options de formation et les données d'entrée, car le réseau de points de contrôle ne contient pas ces informations.

#### 4.5.6 Configurer les paramètres dans les couches convolutives et entièrement connectées

On peut définir des paramètres d'apprentissage différents des valeurs globales spécifiées par *formationOptions* dans des couches avec des paramètres apprenants, tels que des couches convolutives et des couches entièrement connectées. Par exemple, pour ajuster le taux d'apprentissage pour les biais ou les poids, on peut spécifier une valeur pour les propriétés *BiasLearnRateFactor* ou *WeightLearnRateFactor* de la couche, respectivement. La fonction

*trainNetwork* multiplie le taux d'apprentissage spécifié en utilisant *trainingOptions* avec ces facteurs. De même, on peut également spécifier les facteurs de régularisation L2 pour les poids et les biais dans ces couches en spécifiant respectivement les propriétés *BiasL2Factor* et *WeightL2Factor*. *trainNetwork* multiplie ensuite les facteurs de régularisation L2 spécifié en utilisant *trainingOptions* avec ces facteurs.

Initialiser les poids dans les couches convolutives et entièrement connectées : Par défaut, les valeurs initiales des poids des couches convolutives et entièrement connectées sont générées aléatoirement à partir d'une distribution gaussienne de moyenne 0 et d'écart-type 0,01. Les biais initiaux sont par défaut égaux à 0. On peut modifier manuellement les poids initiaux et les biais après avoir créé les couches.

#### 4.5.7 Entraîner le réseau

Après avoir spécifié les couches du réseau et les paramètres d'entraînement, on peut entraîner le réseau en utilisant les données d'entraînement. Les données, les couches et les options d'apprentissage sont tous des arguments d'entrée de la fonction *trainNetwork*.

#### 4.6 Recadrage intelligent

Les données d'apprentissage peuvent être un tableau, une table ou un objet *ImageDatastore*. Si le pooling permet d'augmenter l'efficacité du traitement, il détruit le lien entre une image et son contenu (ex. le nez et le visage). Cette relation peut être pourtant très utile (notamment en reconnaissance faciale). En faisant déborder les tuiles de pooling les unes sur les autres, il est possible de définir une position pour un élément (ex. le nez est toujours au milieu du visage), mais ce débordement par translation empêche toute autre forme d'extrapolation (changement d'angle de vue, d'échelle...), contrairement à ce que le cerveau humain est capable de faire. À ce jour, cette limitation est contournée en modifiant légèrement les images d'entraînement (luminosité, angle, taille...), mais au prix d'un coûteux temps d'apprentissage. Il est cependant possible d'utiliser un pavage intelligent : l'image globale va être analysée et lorsqu'un élément va être identifié, il va être extrait de l'image (recadré), puis envoyé vers la couche suivante. C'est un fonctionnement utile pour reconnaître les individus d'une photo. Le premier étage de traitement va identifier les visages tandis que le second va essayer d'identifier la personne qui correspond à ce visage (cf. identification automatique dans les photos publiées sur Facebook). On peut comparer ce fonctionnement à une focalisation du regard. lorsqu'on veut reconnaître une personne qui passe, on va la regarder dans les yeux (recadrage). De plus il devient possible de prédire la présence d'un élément grâce à la vision



d'un ensemble de ses sous-parties (si je vois deux yeux et un nez, il y a de fortes chances pour que je sois face à un visage, même si je ne le vois pas complètement). Il est possible d'ajuster la taille et la position du recadrage (opérations linéaires) pour faciliter et généraliser le traitement.

#### **4.7 Détection de visage basée sur le réseau neuronal**

Au début de 1994, Vaillant et al. [125] ont appliqué des réseaux de neurones pour la détection de visage. Dans leur travail, ils ont proposé de former un réseau de neurones convolutionnels pour détecter la présence ou l'absence d'un visage dans une fenêtre d'image et de scanner l'image entière avec le réseau à tous les emplacements possibles. En 1996, Rowley et al. [126] ont présenté un réseau neuronal rétinaiement connecté pour la détection de face frontale verticale. La méthode a été étendue pour la détection de face invariante par rotation plus tard en 1998 [127] avec un réseau de "routeur" pour estimer l'orientation et appliquer le bon réseau de détecteurs. En 2002, Garcia et al. [128] ont développé un réseau de neurones pour détecter les visages humains semi-frontaux dans des images complexes; en 2005 Osadchy et al. [129] ont formé un réseau convolutif pour la détection simultanée du visage et l'estimation de la pose. On ne sait pas comment ces détecteurs fonctionnent dans les repères d'aujourd'hui avec des visages dans des environnements incontrôlés ? Néanmoins, compte tenu des récentes découvertes des CNN [130] pour la classification des images [131] et la détection des objets [132], il vaut la peine de revoir la détection des visages basée sur les réseaux neuronaux.

L'une des méthodes de détection récente basée sur CNN est le R-CNN de Girshick et al. [133] qui a atteint le résultat de l'état de la technique sur VOC 2012. R-CNN suit le paradigme de la «reconnaissance par les régions». Il génère des propositions de régions indépendantes de la catégorie et extrait les entités CNN des régions. Ensuite, il applique des classificateurs spécifiques à la classe pour reconnaître la catégorie d'objet des propositions. Comparée à la tâche générale de détection d'objet, la détection de visage non contrôlée présente différents défis qui rendent impossible l'application directe de la méthode R-CNN pour faire face à la détection. Par exemple, les méthodes de proposition d'objet générales peuvent ne pas être efficaces pour les faces en raison de visages de petite taille et de variations d'apparence complexes.

#### **4.8 Détection de visage dans des environnements non contrôlés**

Les systèmes de détection de visage non contrôlés précédents sont principalement basés sur des caractéristiques artisanales. Depuis le détecteur de face Viola Jones [1], un certain nombre de variantes sont proposées pour la détection de visage en temps réel. Récemment dans

la cascade boostée avec un cadre de fonctionnalités simples. Considérant le succès des CNN dans un certain nombre de tâches visuelles, y compris l'alignement du visage, le cadre est plus général en ce sens que l'on peut adopter une méthode d'alignement de visage basée sur CNN des fonctionnalités plus robustes pour les visages. Zhang et al. [135] et Park et al. [136] adoptent l'idée multirésolution dans la détection générale d'objets. Tout en partageant la technique similaire, notre méthode utilise CNN comme les classificateurs et combine les idées multirésolution et d'étalonnage pour la détection de visage. Par conséquent, on bénéficie des fonctionnalités puissantes apprises par le CNN pour mieux différencier les visages des arrière-plans très encombrés. Pendant ce temps, le détecteur est plusieurs fois plus rapide que les systèmes de détection basés sur un modèle et a une fréquence d'images comparable à la cascade boostée classique avec des caractéristiques simples. Partageant les avantages du CNN, le détecteur est facile à mettre en parallèle sur GPU pour une détection beaucoup plus rapide.

#### **4.9 Détection d'objet avec un R-CNN**

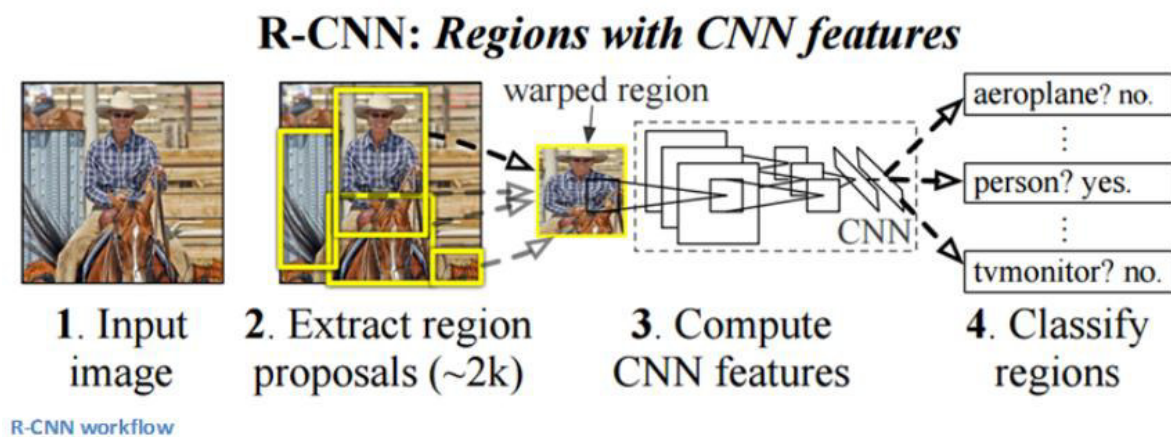
##### **4.9.1 Architecture du RCNN**

Notre système de détection d'objets est composé de trois modules. Le premier génère des propositions de régions indépendantes de la catégorie. Ces propositions définissent l'ensemble des détections de candidats disponibles pour notre détecteur. Le second module est un réseau convolutif qui extrait un vecteur de caractéristiques de longueur fixe de chaque région. Le troisième module est un ensemble de SVM linéaires spécifiques à la classe. Dans cette section, nous présentons nos décisions de conception pour chaque module, décrivons leur utilisation du temps de test, détaillons comment leurs paramètres sont appris, et montrons les résultats de la détection sur MATLAB2018.

Certains peuvent prétendre que l'avènement des R-CNN a eu plus d'impact que n'importe lequel des articles précédents sur les nouvelles architectures de réseau. Avec le premier article de R-CNN cité plus de 1600 fois, Ross Girshick et son groupe à UC Berkeley ont créé l'un des progrès les plus percutants dans la vision par ordinateur. Comme en témoignent leurs titres, R-CNN a travaillé pour rendre le modèle plus rapide et mieux adapté aux tâches de détection d'objets modernes.

Le but des R-CNN est de résoudre le problème de la détection d'objets. Étant donné une certaine image, nous voulons être en mesure de dessiner des boîtes englobantes sur tous les objets. Le processus peut être divisé en deux composantes générales, l'étape de proposition de région et l'étape de classification.

Les auteurs notent que toute méthode de proposition de la région agnostique de classe devrait s'adapter. La recherche sélective est utilisée en particulier pour RCNN. La recherche sélective effectue la fonction de générer 2000 régions différentes qui ont la probabilité la plus élevée de contenir un objet. Après avoir proposé un ensemble de propositions de région, ces propositions sont ensuite "déformées" en une taille d'image qui peut être introduite dans un CNN formé (AlexNet dans ce cas) qui extrait un vecteur de caractéristiques pour chaque région. Ce vecteur est ensuite utilisé comme entrée d'un ensemble de SVM linéaires formés pour chaque classe et générant une classification. Le vecteur est également alimenté dans un régresseur de boîte englobante pour obtenir les coordonnées les plus précises. La suppression non-maximale est ensuite utilisée pour supprimer les boîtes englobantes qui se chevauchent de manière significative.



*Fig. 4. 6 Architecture du RCNN [137]*

#### 4.9.2 Principe de fonctionnement du RCNN

Cet exemple montre comment former un détecteur d'objet R-CNN pour détecter Face. R-CNN est un cadre de détection d'objet, qui utilise un réseau de neurones convolutionnels (CNN) pour classer les régions d'image dans une image [137]. Au lieu de classer chaque région à l'aide d'une fenêtre glissante, le détecteur R-CNN ne traite que les régions susceptibles de contenir un objet. Cela réduit considérablement les coûts de calcul encourus lors de l'exécution d'un CNN.

Pour illustrer comment former un détecteur de visage R-CNN, cet exemple suit le flux de travail d'apprentissage de transfert couramment utilisé dans les applications d'apprentissage en profondeur. Dans l'apprentissage par transfert, un réseau formé sur une grande collection d'images, comme ImageNet [2], est utilisé comme point de départ pour résoudre une nouvelle tâche de classification ou de détection. L'avantage d'utiliser cette approche est que le réseau

pré-entraîné a déjà appris un riche ensemble de caractéristiques d'image qui sont applicables à un large éventail d'images. Cet apprentissage est transférable à la nouvelle tâche en affinant le réseau. Un réseau est affiné en faisant de petits ajustements aux poids de sorte que les représentations d'entités apprises pour la tâche d'origine soient légèrement ajustées pour supporter la nouvelle tâche.

L'avantage de l'apprentissage par transfert est que le nombre d'images requises pour l'entraînement et le temps d'entraînement sont réduits. Pour illustrer ces avantages, cet exemple forme un détecteur de visage à l'aide du flux de travail d'apprentissage de transfert. Tout d'abord, un CNN est pré-formé en utilisant l'ensemble de données CIFAR-10, qui contient 50 000 images d'entraînement. Ensuite, ce CNN pré-entraîné est affiné pour la détection de Face en utilisant seulement 60 images d'entraînement. Sans pré-entraîner le CNN, l'entraînement du détecteur de visage nécessiterait beaucoup plus d'images.

#### 4.9.3 Description de l'implémentation du RCNN

Pour mettre en œuvre le RCNN nous avons besoin principalement de la fonction :

*detector = trainRCNNObjectDetector (trainingData, network, options)* ; qui renvoie un détecteur d'objets basé sur RCNN (Régions avec Réseaux Neuronaux Convolutionnels). La fonction utilise l'apprentissage en profondeur pour former le détecteur pour la détection d'objets multiclassés. Nous spécifions nos données d'entraînement de terrain, le réseau et les options de formation. Le réseau peut être un réseau de séries prédéfini tel que AlexNet ou VGG16 pour la formation en utilisant l'apprentissage par transfert, ou nous pouvons former un réseau à partir de zéro en utilisant un tableau d'objets Layer avec des poids non initialisés.

Cette fonction nécessite que nous ayons Neural Network Toolbox™ et Statistics and Machine Learning Toolbox™. Il est recommandé d'utiliser Parallel Computing Toolbox™ avec un GPU NVIDIA® compatible CUDA® avec une capacité de calcul de 3.0 ou supérieure.

*detector = trainRCNNObjectDetector (\_\_\_,Name, Value)* renvoie un objet détecteur avec des propriétés d'entrée facultatives spécifiées par un ou plusieurs arguments : la paire (Name, Value).

*detector = trainRCNNObjectDetector (\_\_\_,'RegionProposalFcn',proposalFcn)* entraîne éventuellement un détecteur RCNN en utilisant une fonction de proposition de région personnalisée.

*[detector, info] = trainRCNNObjectDetector (\_\_\_)* renvoie en outre des informations sur les progrès de l'entraînement, tels que la perte et la précision de l'entraînement, pour chaque itération.

#### 4.9.3.1 Processus RCNN

##### a) Apprentissage du RCNN pour la détection de visage

- Charger les données d'apprentissage et les couches réseau ;
- Ajouter le répertoire d'image au chemin d'accès MATLAB ;
- Définir les options de formation réseau pour utiliser une taille de mini-lot de 32 afin de réduire l'utilisation de la mémoire GPU. Réduire la valeur *InitialLearningRate* pour réduire la vitesse à laquelle les paramètres réseau sont modifiés. Ceci est utile lors de la mise au point d'un réseau pré-formé et empêche le réseau de changer trop rapidement ;
- Entraîner le détecteur RCNN. La formation peut prendre quelques minutes pour être compléte.
- Formation d'un détecteur d'objets RCNN pour les classes de visage de personnes (objets) dans la base de données :
  - *Étape 1 sur 3*: Extraire des propositions de régions à partir de 32 images de formation, faire :
    - *Étape 2 sur 3*: Former un réseau de neurones pour classer les objets dans les données d'entraînement ;
- Formation en réseau terminée.
  - *Étape 3 sur 3*: Modèles de régression de boîte englobante pour chaque classe d'objets ... 100.00% ... fait.
- La formation **RCNN** est terminée.

##### b) Test du détecteur RCNN sur une image de test

- Afficher le résultat de détection le plus fort ;
- Supprimer le répertoire d'images du chemin.

##### c) Reprendre la formation d'un détecteur d'objets RCNN

- Reprendre la formation du détecteur d'objet RCNN en utilisant des données supplémentaires. Pour illustrer cette procédure, la moitié des données du terrain (BDD)

sera utilisée pour entraîner le détecteur. Ensuite, la formation est reprise en utilisant toutes les données ;

- Charger les données d'entraînement et initialiser les options de formation ;
- Entraîner le détecteur RCNN avec une partie de la base de données ;
- Obtenir les couches réseau formés du détecteur. Lorsque nous passons dans un tableau de couches réseau pour former *RCNNObjectDetector*. Elles sont utilisées telles quelles pour continuer la formation ;
- Reprendre l'entraînement en utilisant toutes les données d'entraînement.

#### d) Utiliser un réseau enregistré dans le détecteur d'objets RCNN

- Créer un détecteur d'objet RCNN et configurer-le pour utiliser un point de contrôle réseau enregistré. Un point de contrôle réseau est sauvegardé à chaque époque lors de la formation réseau lorsque le paramètre '*CheckpointPath*' de *trainingOptions* est défini. Les points de contrôle réseau sont utiles dans le cas où la session d'entraînement se termine de façon inattendue ;
- Charger les données d'apprentissage du visage ;
- Ajouter le chemin d'accès complet aux fichiers d'image ;
- Définir le '*Chemin du point de contrôle*' en utilisant la fonction *trainingOptions* ;
- Entraîner le détecteur d'objets RCNN avec quelques images ;
- Charger un point de contrôle du réseau enregistré ;
- Charger l'un des réseaux de points de contrôle ;  
*ans = SeriesNetwork* avec propriétés:  
*Calques: [15 × 1 nnet.cnn.layer.Layer]*
- Créer un nouveau détecteur d'objet RCNN et configurer-le pour utiliser le réseau enregistré ;
- Définir le réseau sur le point de contrôle réseau enregistré.

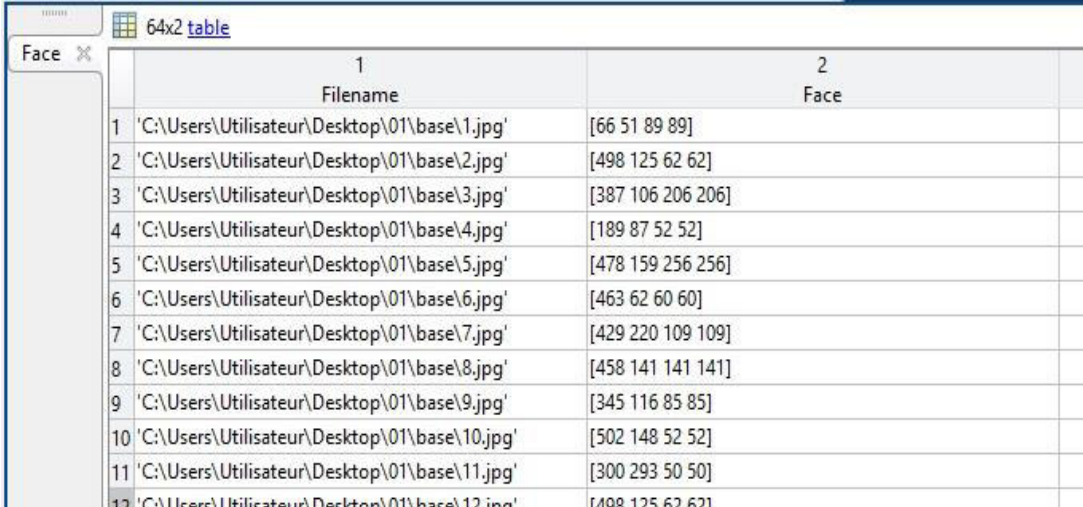
### 4.9.4 Expériences menées sur la détection par RCNN

#### 4.9.4.1 Définition d'arguments d'entrée

**Données de formation** - Images de vérité du terrain (base de données) étiquetées

Les images de vérité de terrain étiquetées, spécifiées sous la forme d'une table avec deux colonnes ou plus. La première colonne doit contenir le chemin d'accès et les noms de fichiers aux images en niveaux de gris ou en couleurs (RVB). Les colonnes restantes doivent

contenir des cadres de délimitation liés à l'image correspondante. Chaque colonne représente une classe d'objets unique, telle qu'un visage, un animal ou une fleur.



	1 Filename	2 Face
1	'C:\Users\Utilisateur\Desktop\01\base\1.jpg'	[66 51 89 89]
2	'C:\Users\Utilisateur\Desktop\01\base\2.jpg'	[498 125 62 62]
3	'C:\Users\Utilisateur\Desktop\01\base\3.jpg'	[387 106 206 206]
4	'C:\Users\Utilisateur\Desktop\01\base\4.jpg'	[189 87 52 52]
5	'C:\Users\Utilisateur\Desktop\01\base\5.jpg'	[478 159 256 256]
6	'C:\Users\Utilisateur\Desktop\01\base\6.jpg'	[463 62 60 60]
7	'C:\Users\Utilisateur\Desktop\01\base\7.jpg'	[429 220 109 109]
8	'C:\Users\Utilisateur\Desktop\01\base\8.jpg'	[458 141 141 141]
9	'C:\Users\Utilisateur\Desktop\01\base\9.jpg'	[345 116 85 85]
10	'C:\Users\Utilisateur\Desktop\01\base\10.jpg'	[502 148 52 52]
11	'C:\Users\Utilisateur\Desktop\01\base\11.jpg'	[300 293 50 50]
12	'C:\Users\Utilisateur\Desktop\01\base\12.jpg'	[498 125 62 62]

**Fig. 4. 7** Exemple de résultats de table de vérité du terrain au format [x, y, largeur, hauteur]

Chaque boîte englobante doit être au format [x, y, largeur, hauteur]. Le format spécifie l'emplacement du coin supérieur gauche et la taille de l'objet (visage) dans l'image correspondante. Le nom de la variable de table définit le nom de la classe d'objets. Pour créer la table de vérité du terrain, nous utilisons l'application *Image Labeler*. Les boîtes plus petites que 32 par 32 ne sont pas utilisées pour l'entraînement.

#### 4.9.4.2 options - Options de formation

##### a) Arguments de la paire (nom,valeur)

Des paires facultatives séparées par des virgules d'arguments (Name, Value) sont spécifiées. Le nom est le nom de l'argument et la valeur est la valeur correspondante. Le nom doit apparaître à l'intérieur des guillemets simples ("). On peut spécifier plusieurs arguments de paires (nom,valeur) dans n'importe quel ordre: Nom1, Valeur1, ..., NomN, ValeurN.

L'entrée, I, est une image définie dans la table groundTruth. La fonction doit renvoyer des cadres de délimitation rectangulaires dans un tableau M-by-4. Chaque rangée de bbox contient un vecteur à quatre éléments, [x, y, width, height], qui spécifie le coin supérieur gauche et la taille d'une boîte englobante en pixels. La fonction doit également renvoyer un score pour chaque boîte englobante dans un vecteur M-by-1. Les scores les plus élevés indiquent que la boîte englobante est plus susceptible de contenir un objet. Les scores sont utilisés pour sélectionner les régions les plus fortes, que vous pouvez spécifier dans NumStrongestRegions.

*b) Arguments de sortie*

- *detector* - Objet de détection d'objet basé sur RCNN

Le détecteur d'objets RCNN formé est renvoyé sous la forme d'un objet. On peut former un détecteur RCNN pour détecter plusieurs classes d'objets.

- *info* - Informations de formation structure

Les informations sur la formation sont renvoyées sous forme de structure avec des champs successifs. Chaque champ est un vecteur numérique avec un élément par itération d'entraînement. Les valeurs qui n'ont pas été calculées à une itération spécifique sont représentées par NaN.

*TrainingLoss* - Perte de formation à chaque itération. C'est la combinaison de la perte de classification et de régression utilisée pour former le réseau RCNN.

*TrainingAccuracy* - Précision du jeu d'entraînement à chaque itération

*BaseLearnRate* - Taux d'apprentissage à chaque itération

**4.9.4.3 Résultats pour la détection RCNN**

*1) Recherche de l'architecture de RCNN*

Nous menons deux expériences essentielles pour valider le RCNN pour la détection. Dans la première expérience nous avons testé le système, en utilisant une BDD composée de 5 personnes pour vérifier notre système de détection et ses performances à partir de 6 couches. la 2ème expérience nous utilisons la même BDD mais avec 15 couches.

```

layers = |
15x1 Layer array with layers:
1  ''  Image Input
2  ''  Convolution
3  ''  Batch Normalization
4  ''  ReLU
5  ''  Max Pooling
6  ''  Convolution
7  ''  Batch Normalization
8  ''  ReLU
9  ''  Max Pooling
10 ''  Convolution
11 ''  Batch Normalization
12 ''  ReLU
13 ''  Fully Connected
14 ''  Softmax
15 ''  Classification Output

layers =
6x1 Layer array with layers:
1  ''  Image Input
2  ''  Convolution
3  ''  ReLU
4  ''  Fully Connected
5  ''  Softmax
6  ''  Classification Output
    
```

(a) Architecture RCNN 6 couches

(b) Architecture RCNN 15 couches

**Fig. 4. 8 Architectures utilisées pour la détection RCNN**



Les résultats des deux expériences sont présentés sur la figure 4.9 ci-dessous :



(b) RCNN à six couches



(b) RCNN à quinze couches

**Fig. 4. 9** Résultats de détection par RCNN pour différentes architectures

**Discussion :** L'architecture du RCNN à quinze couches donne de meilleurs résultats de détection pour si bonne résultats

### 2) Configuration du RCNN

Suite à la séries d'expériences que nous avons mené, nous choisissons les paramètres suivants :

1	"	Image Input	32x32x3 images with 'zerocenter' normalization
2	"	Convolution	64 8x8 convolutions with stride [1 1] and padding 'same'
3	"	Batch Normalization	Batch normalization
4	"	ReLU	ReLU
5	"	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0 0 0]
6	"	Convolution	32 8x8 convolutions with stride [1 1] and padding 'same'
7	"	Batch Normalization	Batch normalization
8	"	ReLU	ReLU
9	"	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0 0 0]
10	"	Convolution	16 8x8 convolutions with stride [1 1] and padding 'same'
11	"	Batch Normalization	Batch normalization
12	"	ReLU	ReLU
13	"	Fully Connected	1 fully connected layer
14	"	Softmax	softmax
15	"	Classification Output	crossentropyex

### 3) Optimisation de RCNN

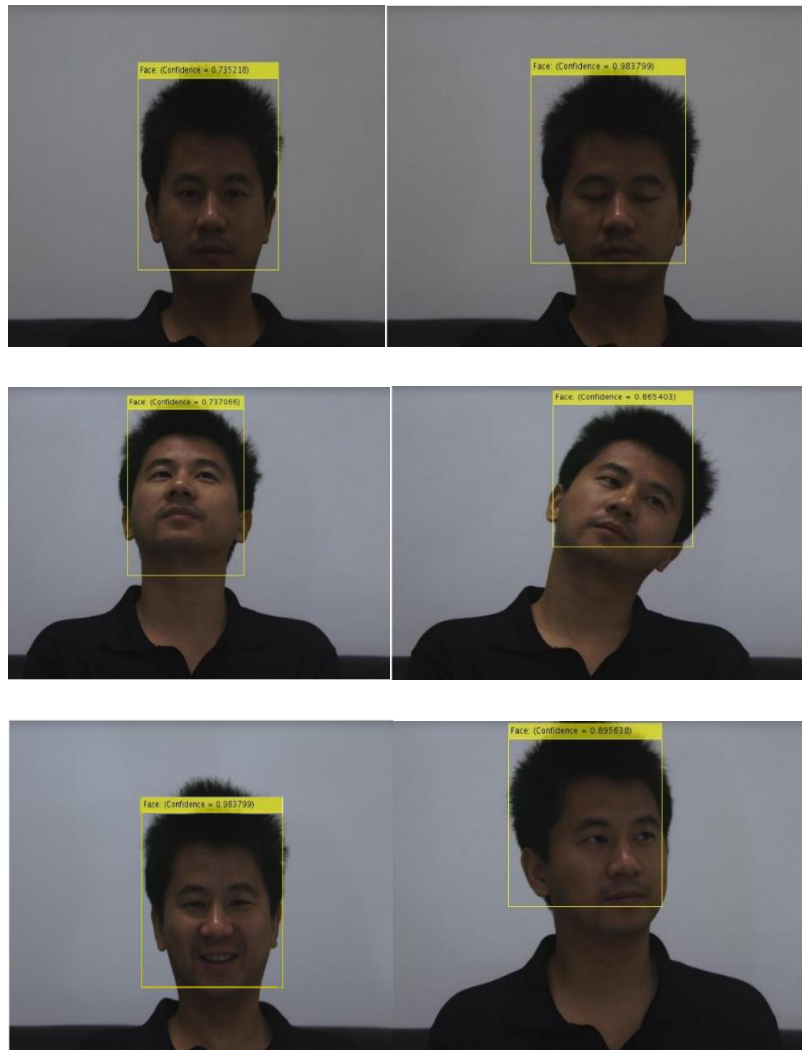
Nous retenons donc, l'architecture 15 couches avec les même paramètres pour la suite de nos expériences. Nous validons tout d'abord nos résultats sur des images acquises en milieu contrôlés en présence de différents variantes : expression , pose, illuminations, occlusions. Ensuite, la validation des résultats est effectuée en milieu incontrôlés en présence de différents variantes : expression , pose , action , couleur de la peau , occlusions....

#### 4) Expériences et résultats en milieux contrôlés BDD (2DCASIAV4)

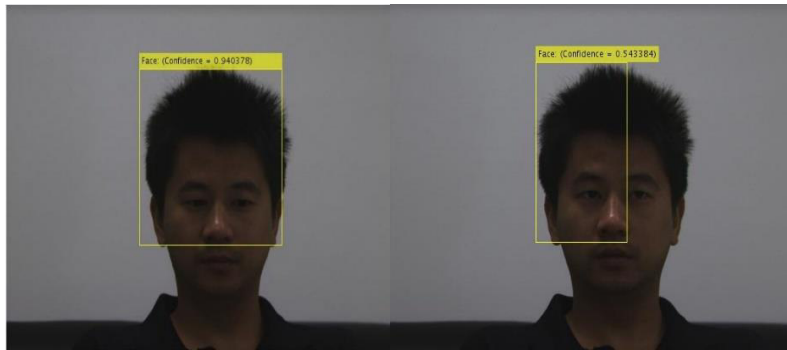
La base de données d'images de visage 2DCASIA version 4.0 (ou 2DCASIAV4) utilisée contient 123 personnes. Toutes les images de visage sont des fichiers BMP couleur de 16 bits et de résolution 640 \* 480. Les variations intra-classes typiques incluent illumination, pose, expression, lunettes, distance d'imagerie, etc. [[biometrics.idealtest.org/](http://biometrics.idealtest.org/)]

Nous avons utilisé 38 images pour un personne pour valider notre méthode de détection.

Dans les images de la figure 4.10 , nous présentons les bonnes détections par le RCNN en présence de pose et fermeture des yeux et expression.



*Fig. 4. 10 Résultats de bonnes détections RCNN*



*Fig. 4. 11 Résultats de mauvaises détections RCNN*

Nous avons validé le détecteur RCNN sur la totalité CASIA2DV4 c'est-à-dire sur un échantillon de 3000 images et les résultats sont très satisfaisants avec de très bonnes performances de détection avec un temps égale à 3.350 s ce qui est très appréciable.

### **Discussion**

Les mauvaises détections peuvent être liés au regard car dans l'image fermeture des yeux une performance de 98.37% obtenue. Ce résultat peut être très sollicité dans l'application par exemple de surveillance de somnolence des conducteurs. Les résultats les plus défavorables obtenus présentent des performances au pire des cas égale à 54.33%.

Nous pouvons affirmer que le détecteur est un très bon candidat pour le cas des BDD collectées en milieux contrôlés. Notre objectif, reste l'amélioration ou l'optimisation de la détection de visage en milieux incontrôlés. Pour cette raison, nous nous intéressons à l'application du détecteur RCNN étudié sur une base de données en milieux incontrôlés dans la deuxième série d'expériences.

#### ***4) Expériences et résultats en milieux incontrôlés***

Nous utilisons dans cette deuxième série d'expériences une BDD qui contient des images choisies sur internet présentant différentes variantes : expression , pose , action , couleur de peau , occlusions...

number des images 64, size d'image et extension n'a pas d'importance



Fig. 4. 12 échantillon d'images d'une BDD

Résultats obtenus dans le cas de détection de visage d'une personne sur l'image

La détection du visage par RCNN de la série d'images d'une personne se trouve sur la figure 4.13 ci-dessous :







Fig. 4. 13 Echantillon de détection de visage d'images d'une personne

**Discussion :**

L'occlusion (image de la femme) et orientation de la tête (image du Roi Mohamed VI) présentent de légères diminution de performances. Le regard de la personne peut avoir des effets sur la détection (cas des images : président Ordoghal et aussi de la dernière image en bas à droite). Le détecteur RCNN reste insensible aux expressions, inclinaison de la tête et à l'action ainsi que la position debout ou assise.

Malgré la taille réduite de la base de données usuelle, nous pouvons dire que le détecteur RCNN fonctionne très bien sur la majorité des images où l'on trouve une seule personne et cela dans quasiment toutes les images en milieux incontrôlés que nous avons testé.

a) Résultats obtenus dans le cas de détection de visage d'un groupe de personnes sur une image(en milieu incontrôlés)



Fig. 4. 14 Exemple de détection de visage d'un groupe de personnes sur une image (en milieu incontrôlés)



(a) Apprentissage du RCNN sur BDD en milieu incontrôlés





(b) Apprentissage du RCNN sur une personne en milieu contrôlé

(062\_xxx de BDD CASIA2DV4)

*Fig. 4. 15 Exemple de détection de visage sur des images en milieux incontrôlés*

### Discussion

Nous constatons que le détecteur RCNN sur cette deuxième série d'expérience détecte bien les visages en général, mais l'inconvénient c'est la détection d'autres régions existantes sur l'image, surtout dans le cas des images qui présentent un arrière plan encombré (joueurs avec tribune etc .. en arrière plan et image de la dame avec policier). Ceci peut être résolu par la création de tableau de vérité du terrain correspondants aux images d'application dans la phase d'apprentissage du réseau RCNN.

**Conclusion**

Dans ce chapitre 4, nous avons détaillé l'étude du réseau de neurones convolutif, puis nous avons expliqué les différentes étapes du processus de détection par RCNN. Nous constatons que le détecteur RCNN sur la deuxième série d'expérience détecte bien les visages en général, mais l'inconvénient c'est la détection d'autres régions existantes sur l'image, surtout dans le cas des images qui présentent un arrière plan encombré (joueurs avec tribune etc .. en arrière plan et image de la dame avec policier). Ceci peut être résolu par la création de tableau de vérité du terrain correspondants aux images d'application dans la phase d'apprentissage du réseau RCNN. Une analyse plus fine de nos données d'apprentissage fournit un premier élément de compréhension. Ainsi pour la variante de visage de la BDDCASIA2DV4 qui est la classe la mieux détectée, nous avons quelques 4127 images, soit au moins 4063 images de plus que la deuxième base de données en milieux incontrôlés. De même, avec seulement 64 photographies, il n'est pas surprenant que le réseau de neurone ne sache pas bien détecter le visage avec un taux de fausse détection égale à : 9. Donc, 9 personnes non détectées/64 personnes totales (dans la BDD) n'ont pas été correctement détectées. Dans ce cas la détection présente des limites à l'expression du visage et l'occlusion. Ceci est dû au mauvais entraînement du réseau. En effet l'une des particularités des réseaux de neurones est de nécessiter de très nombreuses données d'apprentissage. À titre de comparaison, la célèbre base de données ImageNet contient environ 3000 images pour chacune des classes.

Ce manque de données bloque notre réseau de deux manières différentes : en ne lui donnant pas assez d'information pour apprendre à différencier certaines classes; en l'amenant à se spécialiser sur les données d'apprentissage (over-fitting) et donc à être incapable de fournir une réponse correcte pour les données de tests.

Dans le cas de la BDD de petite taille, nous n'avons pas les données suffisantes pour réaliser l'application. Par contre nous en avons assez pour essayer d'identifier les visages de personnes en position frontale et quand il y'a qu'une seule personne dans l'image.

Sur ce problème-là, les scores obtenus sont bien meilleurs : ils atteignent les 87.147% de réussite pour un échantillon de 5 personnes et 88.571% pour 70 personnes en milieux incontrôlés. Par ailleurs le réseau atteint ce score maximal bien plus rapidement : seulement 4 ou 5 d'itérations avec un temps d'exécution de 91.497s pour l'apprentissage des 64 personnes



et 44.491s pour le test contre 10 itérations dans le cas de la BDD CASIA2DV4 avec un temps d'exécution de 40 mn pour l'apprentissage et 15mn de test pour la totalité de la base.

Par contre dans le cas d'une seule personne de la BDD CASIA2DV4 le temps est de 71.305 s (apprentissage par personne), 39.172s pour le test.

## Conclusion Générale

La reconnaissance d'objets est l'un des problèmes les plus difficiles dans la vision par ordinateur et important pour rendre les robots utiles dans les environnements domestiques. La nouvelle technologie de détection, telle que la Kinect, qui peut enregistrer des images RVB et de profondeur de haute qualité (RGB-D) est maintenant abordable et pourrait être combinée avec des systèmes de vision standard dans les robots domestiques. La modalité de profondeur fournit des informations supplémentaires utiles au problème complexe de la détection générale puisque les informations de profondeur sont invariantes à l'éclairage ou aux variations de couleur, fournit des repères géométriques et permet une meilleure séparation de l'arrière-plan. Les méthodes les plus récentes pour la reconnaissance d'objets avec des images RGB-D utilisent des fonctions conçues manuellement comme SIFT pour les images 2D, pour les nuages de points 3D ou des caractéristiques de couleur, de forme et de géométrie spécifiques. Seulement, toutes ces approches se basent sur un apprentissage supervisé. Le Deep Learning nous permet un apprentissage automatique et profond.

Les applications de vidéosurveillance intelligente ont pris beaucoup d'ampleurs. L'utilisation de ces systèmes dans la reconnaissance et la gestion des activités humaines est en extension. Les travaux initiés dans ce mémoire s'inscrivent dans le cadre de la mise en oeuvre d'un système de détection automatique pour la reconnaissance de visage en utilisant plusieurs algorithmes de détection. Ces travaux sont subdivisés en deux (02) grandes parties. La première partie est consacrée à l'étude et la conception de plusieurs méthodes de détection conventionnelles et la seconde présente l'étude et la conception d'une approche de détection basée sur l'apprentissage en profondeur. Dans chaque partie, nos recherches bibliographiques nous ont permis de dresser un état de l'art des approches et solutions proposées. Une fois présentée, nous avons analysé ces approches pour proposer notre contribution. Cette contribution a pour but d'améliorer les aspects de la détection en milieux incontrôlés.

Dans ce mémoire, une étude sur la détection par les outils classiques a été conçue. Les méthodes de détection les plus populaires et efficaces dans l'état de l'art ont été mises à l'épreuve pour la reconnaissance de visage. La méthode de détection de Viola Jones pour la reconnaissance de visage a été retenue comparativement à toutes les autres expérimentées. Cependant, des limites dans l'utilisation de Viola Jones concernant les arrière-plans, la pose et l'expression, nous pousse à nous orienter vers d'autres investigations pour optimiser les performances de la reconnaissance de visage.

La soumission de GoogleNet à ILSVRC 2014 utilise en fait  $12 \times$  moins de paramètres que l'architecture gagnante de Krizhevsky et al [138], il y a deux ans, tout en étant significativement plus précis. Les gains les plus importants dans la détection d'objets ne viennent pas de l'utilisation de réseaux profonds seuls ou de modèles plus grands, mais de la synergie des architectures profondes et de la vision informatique classique, comme l'algorithme R-CNN de Girshick et al [6].

Le réseau neuronal convolutif (CNN) a attiré une attention considérable dans le traitement d'image, la reconnaissance de formes, la vision par ordinateur, améliorant l'état des problèmes de classification. Particulièrement, bénéficiant des représentations discriminantes et efficaces extraites par les modèles CNN, la détection et la reconnaissance d'objets via CNN a connu un grand succès. De nombreuses techniques de détection d'objets basées sur CNN se concentrent sur l'apprentissage de représentations discriminantes et généralisées. Comme on le sait, l'extraction de caractéristiques efficaces joue un rôle crucial dans le processus FR. Le choix d'une représentation de visage appropriée peut rendre le traitement de face ultérieur non seulement réalisable par calcul, mais aussi robuste aux variations faciales intrinsèques et extrinsèques possibles. Une stratégie de patch dans les architectures CNN est introduite pour apprendre des fonctionnalités complémentaires et efficaces dans une approche de bout en bout. Une nouvelle couche réseau est proposée pour échantillonner uniformément plusieurs patches de l'image d'entrée dans le modèle CNN. Pour chaque patch, une branche de réseau convolutif est utilisée pour apprendre et extraire sa fonctionnalité.

En particulier, toutes les fonctionnalités sont d'abord normalisées par BatchNorm puis mises en cascade ensemble. Enfin, la concaténation des entités est en outre intégrée par une couche entièrement connectée sans réduction de dimensionnalité. Le compromis entre les caractéristiques globales et les caractéristiques locales peut être complété dans le cadre proposé en raison de la formation de bout en bout. Les contributions de notre travail sont résumées ci-dessous:

1. Trouver la méthode de détection la plus adaptée à la reconnaissance de visage parmi les quatre méthodes de détection que nous avons conçues
2. Conception d'une méthode de détection basée sur le RCNN pour la reconnaissance en milieux incontrôlés
3. Trouver la meilleure architecture du réseau adapté à l'application FR.
4. Des résultats comparables avec l'état de l'art sont obtenus avec moins de données d'entraînement.

Nous avons donc introduit un nouveau modèle basé sur une combinaison de région et réseaux neuronaux convolutifs. Contrairement aux modèles CNN précédents, nous corrigeons la structure du réseau CNN, lui permettant de combiner plusieurs vecteurs, utiliser plusieurs poids RNN et garder les paramètres initialisés aléatoirement. Cette architecture permet la parallélisation et les hautes vitesses, surpasse les CNN à deux couches et obtient des performances de pointe sans caractéristiques externes. Nous démontrons également l'applicabilité de l'apprentissage convolutionnel des caractéristiques au nouveau domaine des images de profondeur.

Nous avons démontré par l'état de l'art la performance de détection sur plusieurs jeux de données de visage de référence utilisant le CNN.

Les résultats expérimentaux suggèrent que l'efficacité provient du module du réseau de région CNN. Bien que le RCNN plus rapide est conçu pour un objet générique détection, il montre des performances impressionnantes de détection de visage lors de la rééducation sur une formation de détection de visage approprié ensemble. Il pourrait être possible d'améliorer encore ses performances en compte tenu des modèles spéciaux de visages humains. Pour une application industrielle les résultats obtenus restent à améliorer. Mais dans le cadre de notre formation ces deux exemples fournissent une excellente introduction aux réseaux de neurones convolutifs. Ils sont l'occasion d'implémenter de toutes pièces des réseaux de neurones, de charger et d'utiliser un réseau pré-entraîné, de se confronter au problème de l'overfitting et de constater l'impact négatif de classes rares (les classes pour lesquelles nous disposons de peu de données).

Enfin, le fait que les images soient de petites tailles et les données peu nombreuses, rendent réalisable une mise en pratique sur un PC de bureau et permet de conserver des temps d'exécution suffisamment courts pour laisser de la place aux expérimentations.

L'algorithme proposé utilise une approche d'apprentissage profond basée sur le réseau neuronal convolutif. Les résultats obtenus indiquent une très bonne précision globale pour l'étape de détection. Nous avons montré que notre méthode de travail sur la dernière carte de caractéristique permet de diminuer le temps d'exécution.

Nous pouvons dire que le RCNN possède cette capacité à générer automatiquement des variables pertinentes à partir de données non-structurées. Dans une approche traditionnelle, il nous aurait fallu effectuer un certain nombre de prétraitement sur nos images, afin que celles-ci soient aisément identifiables par notre modèle. Cependant, un tel gain sur la phase de prétraitement, rend l'apprentissage plus complexe et de ce fait plus long, en 40 mn sur notre jeu de données. Il convient donc de véritablement s'assurer de la pertinence d'un modèle de RCNN

vis-à-vis des autres types d'algorithmes, et ce afin de conserver une véritable plus-value sur l'investissement qu'il requiert.

### **Perspectives**

Dans les travaux futurs, nous souhaitons améliorer la méthode en utilisant un réseau de neurones convolutionnels 3D pour prendre en compte la dynamique temporelle dans notre modèle. En effet, le réseau convolutif est actuellement limité à gérer les entrées 2D qui nous amènent à traiter l'entrée vidéo seulement image par image. En revanche, le réseau convolutif 3D extrait des caractéristiques spatiales et temporelles en effectuant des convolutions 3D. Ainsi, l'information de mouvement de personne pourrait être encodée, ce qui permettrait de diminuer considérablement le temps de détection.

De plus, pour optimiser la détection et la localisation de la personne sur une vidéo, nous devons améliorer notre base d'image. Le visage est plus difficile à détecter et à localiser en raison de la nature de sa forme, de toutes ses variantes expressives et de sa texture. Nous pourrions améliorer cette détection en augmentant la taille du réseau ou en ajoutant des images de l'objet « visage » dans la base d'apprentissage.

Notre modèle ne détecte que les images où l'on a une seule personne, pour détecter d'autres personnes, nous devons augmenter notre base d'image avec d'autres personnes. En outre, nous prévoyons de comparer notre algorithme aux méthodes conventionnelles sur une plus grande variété des images: différentes variantes. Enfin, une approche intéressante serait de coupler notre réseau convolutionnel d'autres modèles avec une image infrarouge dans la bande LWIR afin de focaliser la recherche dans une zone déterminée de l'image. Ceci permettrait également de différencier le visage de l'arrière-plan.

## REFERENCES

- [1] P. Viola, M. Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features”, Conference On Computer Vision And Pattern Recognition 2001.
- [2] S. Yang, P. Luo, C. C. Loy, and X. Tang, “From facial parts responses to face detection: A deep learning approach,” Proceedings of the IEEE International Conference on Computer Vision, 2015: 3676-3684.
- [3] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, “A convolutional neural network cascade for face detection,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015: 5325- 5334.
- [4] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, “Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks,” IEEE Signal Processing Letters, 2016, 23(10): 1499-1503.
- [5] L. Huang, Y. Yang, Y. Deng, and Y. Yu, “Densebox: Unifying landmark localization with end to end object detection,” arXiv preprint arXiv:1509.04874, 2015, unpublished
- [6] M. Belahcene, “Identification et Authentification en Biométrie “ , Thèse de doctorat d’état, Université Mohamed KHIDER, BISKRA, Janvier 2013
- [7] Yang M H, Kriegman D, Ahuja N. “Detecting faces in images: A survey”. IEEE Trans Pattern Analysis and Machine Intelligence, 2002, 24(1):34-58.
- [8] C. Garcia, M. Delakis.”Convolutional Face Finder: A Neural Architecture for Fast and Robust Face Detection” IEEE Trans. PAMI, vol. 26, no. 11, Nov, 2004.
- [9] V. Vezhnevets, V. Sazonov, A. Andreeva. “A Survey on Pixel-Based Skin Color Detection Techniques”, Proc. Graphicon-2003 pp. 85-92, Moscow, Russia, September 2003.
- [10] W. Yang, A. Waibel Lu. “Skin-color modeling and adaptation”, ACCV98, 1998.
- [11] P. Peer, J. Kovac, F. Solina. “Human skin colour clustering for face detection”. In submitted to EUROCON 2003 – International Conference on Computer as a Tool, 2003.
- [12] D. Chai, K.N. Ngan. “Face segmentation using skin-color map in videophone applications”, IEEE Trans. Circuits Syst. Video Technol.9 (4) 1999.
- [13] [http://www.some.ecu. au/~sphung](http://www.some.ecu.au/~sphung).
- [14] C. Garcia, G. Tziritas. “Face detection using quantized skin color regions merging and wavelet packet analysis”, IEEE Trans. Multimedia 1 (3) 264–277, 1999.
- [15] G. Gomez, E. Morales, “Automatic feature construction and a simple rule induction algorithm for skin detection”, Proceedings of Workshop on Machine Learning in Computer

Vision, pp. 31–38,2002.

[16] M.H. Yang, N. Ahuja. “Detecting human faces in color images”, ICIP98, 1998.

[17] M. J. Jones and J. M. Rehg. “Statistical Color Models with Application to Skin Detection”. *Int. J. Computer Vision* 46(1), pp. 81-96, 2002.

[18] A. P. Dempster, N. M. Laird, D. B. Rubin. “Maximum Likelihood from Incomplete Data via the EM Algorithm *Journal of the Royal Statistical Society*”. Series B (Methodological), Vol. 39, No. 1, pp. 1-38,1977.

[19] M.H. Yang, N. Ahuja. “Gaussian Mixture model for human skin color and its application in image and video databases”, *Proceedings of SPIE: Conference on Storage and Retrieval for Image and Video Databases*, vol. 3656, pp. 458–466, 1999.

[20] H. Greenspan, J. Goldberger, I. Eshet. “Mixture model for facecolor modeling and segmentation”, *Pattern Recognition Lett.* 22 (14) 1525–1536, 2001.

[21] J.Y. Lee, S. I. Yoo. “An elliptical boundary model for skin color detection”. In *Proc. of the International Conference on Imaging Science, Systems, and Technology*, 2002.

[22] R.L. Hsu, M. Abdel-Mottaleb, A.K. Jain, “Face detection in color images”, *IEEE Trans. Pattern Anal. Machine Intell.* 24 (5) 696–706, 2002.

[23] B. Martinkauppi, M. Soriano, M. Pietikäinen, “Detection of skin color under changing illumination”: a comparative study, *12th International Conference on Image Analysis and Processing*, 2003.

[24] T. Kohonen. “Self-Organizing Map”, second ed., Springer, Berlin, 1997.

[25] D. Brown, I. Craw, J. Lewthwaite. “A SOM based approach to skin detection with application in real time systems”, *BMVC01*, 2001.

[26] M. J. Jones, and J. M. Rehg. “Statistical color models with application to skin detection”. In *Proc. of the CVPR '99*, vol. 1, 274–280,1999.

[27] P. Kakumanu, S. Makrogiannis, R. Bryll, S. Panchanathan, and N. Bourbakis. “Image chromatic adaptation using ANNs for skin color adaptation”, *Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence, ICTAI04*.

[28] P. Kakumanu. “A face detection and facial expression recognition method applicable to assistive technologies and biometrics”, *PhD Dissertaion*, CSE Department, Wright State University, 2006.

[29] K. Sobottka, I. Pitas, “A novel method for automatic face segmentation, facial feature extraction and tracking”, *Signal Process. Image Commun.* 12, 263–281,1998.

[30] K. Sobottka, I. Pitas. “Extraction of facial regions and features using color and shape information”. In: *Proc. 13th Internat. Conf. Pattern Recognition*, Vienna, Austria, pp. 421-425,

1996.

[31] J.C. Terrillon, M. David, and S. Akamatsu, "Automatic Detection of Human Faces in Natural Scene Images by Use of a Skin Color Model and Invariant Moments," Proc. Third Int'l Conf. Automatic Face and Gesture Recognition, pp. 112-117, 1998.

[32] J.C. Terrillon, M. David, and S. Akamatsu. "Detection of Human Faces in Complex Scene Images by Use of a Skin Color Model and Invariant Fourier-Mellin Moments", Proc. Int'l Conf. Pattern Recognition, pp. 1350-1355, 1998.

[33] R. Belaroussi, L. Prevost, M. Milgram. "Classifier combination for face localization in color images", M., ICIAP', 2005.

[34] C. Zhang and Z. Zhang. "A survey of recent advances in face detection". Technical Report MSR-TR-2010-66, 2010.

[35] Y. LeCun and Y. Bengio. "Convolutional networks for images, speech, and time series". *The handbook of brain theory and neural networks*, 1995

[36] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester, "Cascade object detection with deformable part models", Computer vision and pattern recognition (CVPR), 2010 IEEE conference on. IEEE, 2010: 2241-2248.

[37] J. Yan, Z. Lei, L. Wen, and S. Z. Li, "The fastest deformable part model for object detection," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014: 2497-2504.

[38] J. Yan, X. Zhang, Z. Lei, and S. Z. Li, "Real-time high performance deformable model for face detection in the wild", Biometrics (ICB), 2013 International Conference on. IEEE, 2013: 1-6.

[39] D. C. Ciresan, U. Meier, J. Masci, L. Maria. Gambardella, and J. Schmidhuber, "Flexible, high performance convolutional neural networks for image classification", IJCAI Proceedings-International Joint Conference on Artificial Intelligence, 2011, 22(1): 1237.

[40] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification", Advances in Neural Information Processing Systems, 2014: 1988-1996.

[41] L. Bourdev, S. Maji, T. Brox, and J. Malik. "Detecting people using mutually consistent poselet activations". In *ECCV*. 2010.

[42] P. Dollar, Z. Tu, P. Perona, and S. Belongie. "Integral channel features". In *BMVC*, 2009.



- [43] Y. Ding and J. Xiao. “Contextual boost for pedestrian detection”. In *CVPR*, 2012.
- [44] G. E. Hinton, S. Osindero, and Y.-W. Teh. “A fast learning algorithm for deep belief nets”. *Neural Computation*, 18:1527–1554, 2006.
- [45] S. Chopra, R. Hadsell, and Y. LeCun. “Learning a similarity metric discriminatively, with application to face verification”. In *CVPR*, 2005.
- [46] Y. Sun, X. Wang, and X. Tang. “Hybrid deep learning for computing face similarities”. In *ICCV*, 2013.
- [47] Z. Zhu, P. Luo, X. Wang, and X. Tang. “Deep learning identity preserving face space”. In *ICCV*, 2013.
- [48] P. Sermanet, K. Kavukcuoglu, S. Chintala, and Y. LeCun. “Pedestrian detection with unsupervised multi-stage feature learning”. In *CVPR*, 2013.
- [49] W. Ouyang and X. Wang. “Joint deep learning for pedestrian detection”. In *ICCV*, 2013.
- [50] P. Luo, X. Wang, and X. Tang. “Pedestrian parsing via deep decompositional network”. In *ICCV*, 2013.
- [51] M. Ranzato, J. Susskind, V. Mnih, and G. Hinton. “On deep generative models with applications to recognition”. In *CVPR*, 2011.
- [52] M. Donoser, S. Kluckner, H. Bischof, “Object Tracking by Structure Tensor Analysis”. 2010 International Conference on Pattern Recognition.
- [53] M. Liu, R. Wang, S. Li, S. Shan, Z. Huang, and X. Chen. “Combining multiple kernel methods on riemannian manifold for emotion recognition in the wild”. In Proceedings of the 16th International Conference on Multimodal Interaction, . ACM, 2014.
- [54] Y. Tang. “Deep learning using linear support vector machines”. *arXiv preprint arXiv:1306.0239*,
- [55] I. J. Goodfellow, D. Erhan, P. L. Carrier, A. Courville, M. Mirza, B. Hamner, W. Cukierski, Y. Tang, D. Thaler, D.-H. Lee, et al. “Challenges in representation learning: A report on three machine learning contests”. In *Neural Information Processing*, pages 117–124. Springer, 2013.
- [56] X. Zeng, W. Ouyang, X. Wang. “Multi-Stage Contextual Deep Learning for Pedestrian Detection”. In *CVPR*, 2013.
- [57] M. Valstar, B. Martinez, X. Binefa, and M. Pantic, “Facial point detection using boosted regression and graph models,” in IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 2729–2736.
- [58] X. Cao, Y. Wei, F. Wen, and J. Sun, “Face alignment by explicit shape regression”, in IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 2887–2894.

- [59] X. P. Burgos-Artizzu, P. Perona, and P. Dollar, “Robust face landmark estimation under occlusion”, in IEEE International Conference on Computer Vision, 2013, pp. 1513–1520.
- [60] H. Yang and I. Patras, “Sieving regression forest votes for facial feature detection in the wild”, in IEEE International Conference on Computer Vision, 2013, pp. 1936–1943.
- [65] S. Ren, X. Cao, Y. Wei, and J. Sun, “Face alignment at 3000 fps via regressing local binary features”, in IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1685 – 1692.
- [62] D. Chen, S. Ren, Y. Wei, X. Cao, and J. Sun, “Joint cascade face detection and alignment”, in European Conference on Computer Vision, 2014, pp. 109–122.
- [63] X. Zhu and D. Ramanan, “Face detection, pose estimation, and landmark localization in the wild”, in IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 2879–2886.
- [64] Z. Zhang, P. Luo, C. Change Loy, “Learning Deep Representation for Face Alignment with Auxiliary Attributes “, DOI 10.1109/TPAMI.2015.2469286.
- [65] Y. Sun, X. Wang, and X. Tang, “Deep convolutional network cascade for facial point detection”, in IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 3476–3483.
- [66] J. Zhang, S. Shan, M. Kan, and X. Chen, “Coarse-to-fine autoencoder networks (CFAN) for real-time face alignment”, in European Conference on Computer Vision, 2014, pp. 1–16.
- [67] R. Caruana, “Multitask learning”, *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997.
- [68] S. Li, Z.-Q. Liu, and A. Chan, “Heterogeneous multi-task learning for human pose estimation with deep convolutional neural network,” *International Journal of Computer Vision*, 2014.
- [69] Y. Liu, A. Wu, D. Guo, K.-T. Yao, and C. Raghavendra, “Weighted task regularization for multitask learning,” in *International Conference on Data Mining Workshops*, 2013, pp. 399–406.
- [70] X. Tao, H. Gao, Y. Wang, X. Shen, J. Wang, J. Jia, “Scale-recurrent Network for Deep Image Deblurring”, arXiv:1802.01770v1 [cs.CV] 6 Feb 2018.
- [71] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. “Removing camera shake from a single photograph”. In *ACM Trans. Graph.*, volume 25, pages 787–794. ACM, 2006.
- [72] Q. Shan, J. Jia, and A. Agarwala. “High-quality motion deblurring from a single image”. In *ACM Trans. Graph.*, volume 27, page 73. ACM, 2008.

- [73] J. Sun, W. Cao, Z. Xu, and J. Ponce. “Learning a convolutional neural network for non-uniform motion blur removal”. In *CVPR*, pages 769–777, 2015.
- [74] C. J. Schuler, M. Hirsch, S. Harmeling, and B. Scholkopf. “Learning to deblur”. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(7):1439–1451, 2016.
- [75] A. Chakrabarti. “A neural approach to blind motion deblurring”. In *ECCV*, pages 221–235. Springer, 2016.
- [76] S. Su, M. Delbracio, J. Wang, G. Sapiro, W. Heidrich, and O. Wang. “Deep video deblurring”. pages 1279–1288, 2017.
- [77] S. Nah, T. H. Kim, and K. M. Lee. “Deep multi-scale convolutional neural network for dynamic scene deblurring”. Pages 3883–3891, 2017.
- [78] S. Biswas, S. Biswas, “Structural Recurrent Neural Network (SRNN) for Group Activity Analysis “, arXiv:1802.02091v1 [cs.CV] 6 Feb 2018.
- [79] W. Choi, K. Shahid, and S. Savarese. “What are they doing?: Collective activity classification using spatio-temporal relationship among people”. In *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1282–1289. IEEE, 2009.
- [80] Z. Deng, A. Vahdat, H. Hu, and G. Mori. “Structure Inference Machines: Recurrent Neural Networks for Analyzing Relations in Group Activity Recognition”. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4772–4781, 2016.
- [81] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, F. Li, and S. Savarese. “Social LSTM: Human Trajectory Prediction in Crowded Spaces”. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 961–971, 2016.
- [82] T. Shu, S. Todorovic, and S. Zhu. “CERN: Confidence-Energy Recurrent Network for Group Activity Recognition”. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4255–4263, 2017.
- [83] M. S. Ibrahim, S. Muralidharan, Z. Deng, A. Vahdat, and G. Mori. “A Hierarchical Deep Temporal Model for Group Activity Recognition”. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.
- [84] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena. “StructuralRNN: Deep Learning on Spatio-Temporal Graphs”. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5308–5317, 2016.

- [85] R. Girshick, “FastR-CNN”, arXiv:1504.08083v2 [cs.CV] 27 Sep 2015.
- [86] O. Ronneberger, P. Fischer, and T. Brox. “U-net: Convolutional networks for biomedical image segmentation”. In *MICCAI*, pages 234–241. Springer, 2015.
- [87] X. Mao, C. Shen, and Y.-B. Yang. “Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections”. In *NIPS*, pages 2802–2810, 2016.
- [88] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox. “FlowNet: Learning optical flow with convolutional networks”. In *ICCV*, pages 2758–2766, 2015.
- [89] Q. Chen and V. Koltun. “Photographic image synthesis with cascaded refinement networks”. In *ICCV*. Springer, 2017.
- [90] Q. Chen, J. Xu, and V. Koltun. “Fast image processing with fully-convolutional networks”. In *ICCV*. Springer, 2017.
- [91] M. S. Ibrahim, S. Muralidharan, Z. Deng, A. Vahdat, and G. Mori. “Hierarchical Deep Temporal Models for Group Activity Recognition”. arXiv preprint arXiv:1607.02643, 2016.
- [92] M. Li, L. Jeni, D. Ramanan, “Brute-Force Facial Landmark Analysis With A 140,000-Way Classifier”. arXiv:1802.01777v1 [cs.CV] 6 Feb 2018xC
- [93] Z. Yu, C. Zhang, “Image based Static Facial Expression Recognition with Multiple Deep Network Learning”. *ICMI 2015*, November 9–13, 2015, Seattle, WA, USA. Références
- [94] R. Chellappa, C. L. Wilson, and S. Sirohey, 1995. “Human and machine recognition of faces: A survey”, *Proc. IEEE* 83, 5.
- [95] L. Najman and M. Couprie: “Building the component tree in quasi-linear time” Archived 2011-04-09 at the Wayback Machine.; *IEEE Transaction on Image Processing*, Volume 15, Numbers 11, 2006, pp 3531-3539
- [96] Matas, J., Chum, O., Urban, M., Pajdla, T.: “Robust wide baseline stereo from maximally stable extremal regions”. In: *Proc. British Machine Vision Conference*. (2002) 384–393
- [97] Forssen, P-E. and Lowe, D.G. “Shape Descriptors for Maximally Stable Extremal Regions” *ICCV*, 2007.

- [98] Nister, D. and Stewenius, H., “Linear Time Maximally Stable Extremal Regions”, ECCV, 2008.
- [99] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, T. Kadir and L. Van Gool: “A Comparison of Affine Region Detectors”; International Journal of Computer Vision, Volume 65, Numbers 1-2 / November, 2005, pp 43-72
- [100] Forssen, P-E. “Maximally Stable Colour Regions for Recognition and Matching”; Archived 2011-06-10 at the Wayback Machine., CVPR, 2007.
- [101] Chavez, Aaron; Gustafson, David (2011). “Color-Based Extensions to MSERs”. *ISVC 2011*: 358–366
- [102] Najman and M. Couprie: “Building the component tree in quasi-linear time” Archived 2011-04-09 at the Wayback Machine.; IEEE Transaction on Image Processing, Volume 15, Numbers 11 , 2006, pp 3531-3539
- [103] Donoser, M. and Bischof, H. “Efficient Maximally Stable Extremal Region (MSER) Tracking” CVPR, 2006.
- [104] Ch. Kanga, W. Wang, “A novel edge detection method based on the maximizing objective function” , <http://www.elsevier.com/locate/patcog> , 27 April 2006
- [105] [https://fr.wikipedia.org/wiki/D%C3%A9tection\\_de\\_contours](https://fr.wikipedia.org/wiki/D%C3%A9tection_de_contours)
- [106] L. Huang, A. Shimizu, and H. Kobatake, “Classification-Based Face Detection Using Gabor Filter Features”, IEEE International Conference on Automatic Face and Gesture Recognition (FGR’04) 0-7695-2122-3/04 \$ 20.00 © 2004 IEEE
- [107]: [https://fr.wikipedia.org/wiki/M%C3%A9thode\\_de\\_Viola\\_et\\_Jones#cite\\_noteSzeliski653-12](https://fr.wikipedia.org/wiki/M%C3%A9thode_de_Viola_et_Jones#cite_noteSzeliski653-12)
- [108] B.R. Bruce, J.M. Aitken, J. Petke, “Deep parameter optimisation for face detection using The Viola-Jones algorithm in OpenCV”, LNCS, pp. 238-243, 2016.
- [109] P. Viola, M.J. Jones, “Robust Real-Time Face Detection”, International Journal of Computer Vision, vol 57, issue 2, pp. 137-154, 2004.
- [110] Irgens P., Bader C., Le T., Saxena D., Ababei C. “An efficient and cost effective FPGA based implementation of the Viola-Jones face detection algorithm”, (2017) *HardwareX*, 1 , pp. 68-75.
- [111] P. Viola, M. Jones et D. Snow, “Detecting Pedestrians using Patterns of Motion and Appearance ”, *IJCV*, vol. 63, n° 2, 2005, p. 153-161.
- [112] M. FODDA. “DETECTION ET RECONNAISSANCE DE VISAGE”, 19 Octobre 2010
- [113] <http://utarevis.blogspot.com/2007/09/robust-real-time-face-detection-by-paul.html>

- [114] A.D. Egorov, A.N. Shtanko, P.E. Minin, “Selection of Viola-Jones Algorithm Parameters for Specific Conditions”. *Bulletin of the Lebedev Physics Institute*, Vol 42, No 8, pp 244-248, 2015.
- [115] D.M. Powers, “Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation”. 2011.
- [116] V. Jain, E. Learned-Miller, “FDDB: A Benchmark for Face Detection in Unconstrained Settings”, Technical Report UM-CS-2010-009, Dept. of Computer Science, University of Massachusetts, Amherst. 2010.
- [116] C. Zhang and Z. Zhang. “A survey of recent advances in face detection”. Technical Report MSR-TR-2010-66, 2010.
- [117] Y. LeCun and Y. Bengio. “Convolutional networks for images, speech, and time series”. *The handbook of brain theory and neural networks*, 1995.
- [118] <https://www.mathworks.com>
- [119] <http://erindickey.com/digital-art-history/dah-post-7-art-history-and-machine-learning>
- [120] [https://fr.wikipedia.org/wiki/R%C3%A9seau\\_neuronal\\_convolutif](https://fr.wikipedia.org/wiki/R%C3%A9seau_neuronal_convolutif)
- [121] Krizhevsky, A., I. Sutskever, and G. E. Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. *Advances in Neural Information Processing Systems*. Vol 25, 2012.
- [122] LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D., et al. “Handwritten Digit Recognition with a Back-propagation Network”. In *Advances of Neural Information Processing Systems*, 1990.
- [123] Nagi, J., F. Ducatelle, G. A. Di Caro, D. Ciresan, U. Meier, A. Giusti, F. Nagi, J. Schmidhuber, L. M. Gambardella. “Max-Pooling Convolutional Neural Networks for Vision-based Hand Gesture Recognition”. *IEEE International Conference on Signal and Image Processing Applications (ICSIPA2011)*, 2011.
- [124] Bishop, C. M. “Pattern Recognition and Machine Learning”. Springer, New York, NY, 2006.
- [125] R. Vaillant, C. Monrocq, and Y. “Le Cun. Original approach for the localisation of objects in images”. *IEE Proceedings Vision, Image and Signal Processing*, 1994.

- [126] H. Rowley, S. Baluja, and T. Kanade. “Neural network-based face detection”. In *Computer Vision and Pattern Recognition*, 1996
- [127] H. A. Rowley, S. Baluja, and T. Kanade. “Rotation invariant neural network-based face detection”. In *Computer Vision and Pattern Recognition*, 1998
- [128] C. Garcia and M. Delakis. “A neural architecture for fast and robust face detection”. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, 2002
- [129] M. Osadchy, Y. L. Cun, M. L. Miller, and P. Perona. “Synergistic face detection and pose estimation with energy-based model”. In *In Advances in Neural Information Processing Systems (NIPS)*, 2005.
- [130] Y. LeCun and Y. Bengio. “Convolutional networks for images, speech, and time series”. *The handbook of brain theory and neural networks*, 1995.
- [131] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. “ImageNet Large Scale Visual Recognition Challenge”, 2014.
- [132] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman. “The pascal visual object classes (voc) challenge”, 2009
- [133] R. Girshick, J. Donahue, T. Darrell, and J. Malik. “Rich feature hierarchies for accurate object detection and semantic segmentation”. *arXiv preprint arXiv:1311.2524*, 2013
- [134] D. Chen, S. Ren, Y. Wei, X. Cao, and J. Sun. “Joint cascade face detection and alignment”. In *Computer Vision–ECCV 2014*. 2014.
- [135] W. Zhang, G. Zelinsky, and D. Samaras. “Real-time accurate object detection using multiple resolutions”. In *Proc. IEEE International Conference on Computer Vision*, 2007.
- [136] D. Park, D. Ramanan, and C. Fowlkes. “Multiresolution models for object detection”. In *Computer Vision ECCV 2010*.
- [137]<https://adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html>
- [138][https://leonardoaraujosantos.gitbooks.io/artificialintelligence/content/object\\_localization\\_and\\_detection.html](https://leonardoaraujosantos.gitbooks.io/artificialintelligence/content/object_localization_and_detection.html)

