MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITÉ ABDELHAMID BEN BADIS DE MOSTAGANEM
FACULTÉ DES SCIENCES EXACTES ET DE L'INFORMATIQUE
DÉPARTEMENT DE MATHÉMATIQUES ET INFORMATIQUE

MÉMOIRE

Master Académique

pour obtenir le diplôme de Master délivré par

Université de Mostaganem

Spécialité "Modélisation, Contrôle et Optimisation"

présenté et soutenu publiquement par

Sarah BENAYED

le 27 Juin 2021

les mèthodes de point intèieur pour la dètection faciale

Encadeur : **Abdessamad AMIR (UNIVERSITÉ DE MOSTAGANEM, ALGÉRIE)**

Jury

M.Ablaoui Houcine ,	MCB	Président (Université de Mostaganem, Algérie)
Mme.Djelloul Naima ,	MCB	Examinateur (École normale supérieur de Mostaganem, Algérie)

LABORATOIRE DE MATHÉMATIQUES PURES ET APPLIQUÉES
FACULTÉ DES SCIENCES EXACTES ET DE L'INFORMATIQUE (FSEI)
Chemin des Crêtes (Ex-INES), 27000 Mostaganem, Algérie

MASTER

Résumé

La détection faciale consiste à développer un code qui, dans une image arbitraire, peut détecter et localiser des visages humains. Il gagne en intérêt en tant que domaine de recherche important avec des applications importantes dans la surveillance de sécurité, l'interaction homme-machine et d'autres domaines de la vision par ordinateur et de l'apprentissage automatique. De nombreuses méthodes ont été développées avec plusieurs approches. La majorité des approches utilisées reposent sur deux tâches. La première est la tâche d'extraction de caractéristiques, elle concerne la description numérique précise de ce qui distingue les visages humains des autres objets. La deuxième tâche est généralement un algorithme de classification supervisé qui devra reconnaître avec une bonne précision un visage d'un non-visage.
L'objectif de notre travail est d'utiliser une méthode de point intérieur, afin de résoudre le programme quadratique dans les SVMs. On s'intéressera ensuite au comportement de cette méthode de classification dans le domaine de détection de visage dans une image.

abstract : Facial detection involves developing code that, in an arbitrary image, can detect and locate human faces. It is gaining interest as an important research area with important applications in security surveillance, human-machine interaction, and other areas of computer vision and machine learning. Many methods have been developed with several approaches. The majority of the approaches used are based on two tasks. The first is the feature extraction task, it concerns the precise digital description of what distinguishes human faces from other objects. The second task is usually a supervised classification algorithm which will have to recognize a face from a non-face with good precision.
The objective of our work is to use an interior point method, in order to solve the quadratic program in SVMs. We will then focus on the behavior of this classification method in the field of detection of faces in an image.

Remerciements

Tout d'abord je tiens à remercier Dieu le tout puissant de m'avoir donné la vie, la santé et la volonté d'entamer et de terminer ce mémoire.

J'adresse tous mes remerciements à toutes les personnes qui m'ont aidés dans la réalisation de ce mémoire, en premier lieu, je tiens à exprimer ma toute gratitude et ma reconnaissance à mon encadreur de mémoire, monsieur Amir Abdsamad, Professeur à l'université Abdelhamid Ibn Badis de Mostaganem, pour sa gentillesse , sa compétence et ses conseils qui m'ont été d'un grand secoure .

Je tiens à remercier sincèrement, Docteurs m.Ablaoui Houcine et mme.Djelloul Naima pour accepter de juger mon travail, c'est un grand honneur de vous voir siéger dans notre jury.

Aussi un grand merci à mes amies Bencheib Fadela et Boubekeur Marwa Amel pour toute l'aide qu'ils m'ont apportées

Mes remerciements les plus chaleureux vont à tous mes professeurs, enseignants et les étudiants de ma promotion et en particulier de ma spécialité Modélisation Contrôle et Optimisation pour leur présence dans les moments difficiles et les excellents moments que j'ai passé avec eux tout au long de cette année.

Je garde le meilleur pour la fin, ma famille qui a supporté toutes les difficultés morales et matérielles pour me soutenir tout au long de mes études . A mes très chers parents, que Dieu les garde et les bénisse. Tous les mots ne suffisent pas pour exprimer le respect et l'amour que j'avoue pour eux. A mon frère et mes sœurs, je leur souhaite tout le bonheur du monde.

A tous les mathématiciens qui pourront être aidés par ce travail, souvenez-vous qu'une période d'échec est un moment rêvé pour semer les graines du succès.

Dédicaces

Toutes les lettres ne sauraient trouver les mots qu'il faut...Tous les mots ne sauraient exprimer la gratitude, l'amour, le respect, la reconnaissance... Aussi, c'est tout simplement que

Je dédie ce modeste travail :

A celle qui m'a arrosé de tendresse et d'espoirs, à la source d'amour incessible, à la mère des sentiments fragiles qui ma bénie par ces prières... ma mère .

A mon support dans ma vie, qui m'a appris m'a supporté et ma dirigé vers la gloire... mon père.

pour leur amour inestimable, leurs sacrifices, leur confiance, leur soutien et toutes les valeurs qu'ils ont su m'inculquer.

je vous souhaite tout le bonheur du monde.

A mon cher frère Mansour et mes adorables sœurs (Khadija, Nadia et Souhila) ceux qui ont partagé avec moi tous les moments d'émotion lors de la réalisation de ce travail. Ils m'ont chaleureusement supporté et encouragé tout au long de mon parcours.

Je leurs souhaite une vie pleine du bonheur et de succès.

Et n'oublions pas mes chères tantes Houria et Rachida et ses magnifiques filles Chahinez et Khouira, ainsi que mon oncle Abd el rahman, que Dieu Tout-Puissant, les protège et leur donne santé et longue vie.

A mes chères amies qui m'ont toujours soutenu
Faten, Fadela, Marwa, Feriel, Hafsa
En souvenir de nos éclats de rire et des bons moments, en souvenir de tout ce qu'on à vécu ensemble, j'espére de tout mon cœur que notre amitié durera éternellement inshallah .

A tous ceux que j'aime et qui m'aiment .
Je dédie ce travail espérant avoir répondu à leurs souhaits de me voir réussir.

Table des matières

Table des figures

Introduction générale

La classification est un des problèmes de l'apprentissage automatique, définie comme le processus de reconnaissance et de compréhension des choses et des idées et de leur regroupement en catégories prédéfinies à l'aide d'un ensemble de données d'apprentissage pré-classifiées. La classification dans l'apprentissage automatique regroupe une grande variété d'algorithme de classification, parmi ces méthodes, on peut citer : la méthode réseaux de neurones [8], régression logistique [7], analyse discriminante linéaire, Random Forest... [3]. Dans ce travail, nous nous intéressons à une méthode de classification intitulée, Machines à vecteurs de support(SVMs) [6].

Les (SVMs) sont des modèles d'apprentissage supervisés destinés à résoudre les problèmes de classification et de régression, introduit pour la première fois en 1992 par Vladimir Vapinik [6]. L'idée est de trouver un hyperplan de marge optimale qui sépare parfaitement les données. Leur extension aux problèmes non linéaires est proposée par Boser [4] en introduisant les fonctions noyaux.

Le cœur des SVMs est un problème de programmation quadratique (QP) [22] (fonction objectif quadratique et contraintes linéaires). L'hyperplan de marge optimale est entièrement déterminé par ce problème QP. Les problèmes QP font partie des classes de problèmes d'optimisation très utilisés en pratique. Il existe plusieurs approches et méthodes de résolution pour les problèmes QP. On peut citer à titre d'exemples, la méthode Active-set [22], les méthodes de type gradient projeté, ainsi que les méthodes de point intérieur, qui font l'objet de ce travail.

En 1984, Karmarkar [13] a développé un algorithme de type point intérieur pour la programmation linéaire [9], cet algorithme a permis à la classe des problèmes de programmation linéaire, d'appartenir à la classe des problèmes à complexité polynomial. L'algorithme de Karmarkar a permis de résoudre des problèmes de programmation linéaire qui dépassaient les capacités de la méthode du simplexe. Contrairement à la méthode du simplexe, l'algorithme de Karmarkar atteint la solution optimale en traversant l'intérieur de la région admissible. La méthode peut être généralisée à la QP, en se basant sur la technique des fonctions barrières.

L'objectif de notre travail est d'utiliser une méthode de point intérieur (IPM)[1], afin de résoudre le programme quadratique dans les SVMs. On s'intéressera ensuite au comportement de cette méthode de classification dans le domaine de détection de visage dans une image.

Le manuscrit est organisé comme suit, le premier chapitre est une introduction très abrégée aux SVMs. Les méthodes IPM ont été introduites d'abord dans le cadre de la programmation linéaire au second chapitre, leur extension dans le cas quadratique a fait l'ob-

jet du troisième chapitre. Au quatrième chapitre, une application des SVMs avec un code IPM au domaine de détection de visage dans une image est présentée. Finalement, nous concluons ce manuscrit au dernier chapitre.

Chapitre 1

Les problèmes d'optimisation dans les SVMs

Les machines à vecteurs de support ou séparateur a vaste marge (SVMs) sont des modèles d'apprentissage supervisés avec des algorithmes d'apprentissage destinés à résoudre les problèmes de classification et de régression [6]. L'idée de base est de trouver un hyperplan de marge optimale qui sépare parfaitement les données tout en étant le plus éloigné possible de toutes les observations. Leur extension aux problèmes non linéaires est proposée par Boser [4] en introduisant les fonctions noyaux, pour traiter les cas des données non séparables. Les SVMs ont été utilisés dans différents domaines. Dans ce chapitre, nous nous concentrons sur les aspects théoriques de la méthode SVMs .

1 SVMs sur des données linéairement séparables

Étant donné un ensemble d'apprentissage de n données de la forme $(x_i, y_i)_{1\leq i\leq n} \in \mathrm{X}\times\mathrm{Y}$, où X est l'espace des données d'apprentissage souvent pris dans $\mathbb{R}^d$, $\mathrm{Y} = \{-1, +1\}$. L'appartenance d'une observation x_i à une classe ou à une autre est matérialisée par la valeur $y_i = 1$ ou $y_i = -1$.

L'objectif des SVMs dans le cas linéaire est de calculer un hyperplan qui sépare au mieux les échantillons des deux classes. Cet hyperplan donné par la fonction :

$$f(x) = \omega^{\mathrm{T}} x + b$$

Où ω [1] est un vecteur de $\mathbb{R}^d$ et $b \in \mathbb{R}$. La fonction f est une fonction de décision (appelée aussi la fonction classificatrice), on peut lui associer l'hyperplan séparateur :

$$\mathrm{H}_0 := \{x \in \mathbb{R}^d : \omega^{\mathrm{T}} x + b = 0\}$$

Il peut y avoir un nombre infini d'hyperplans séparateur (voir la figure (1.1)). Le principe des machines à vecteurs de support est de sélectionner le meilleur hyperplan, c'est à dire, celui qui sépare mieux les autres données que celles de l'ensemble d'aprentissage. Pour déterminer ce qui caractérise le meilleur hyperplan, introduisons le concept de marge.

1. ω est le vecteur normale à H_0 et b le bias ($b = 0$, si l'hyperplan passe par l'origine)

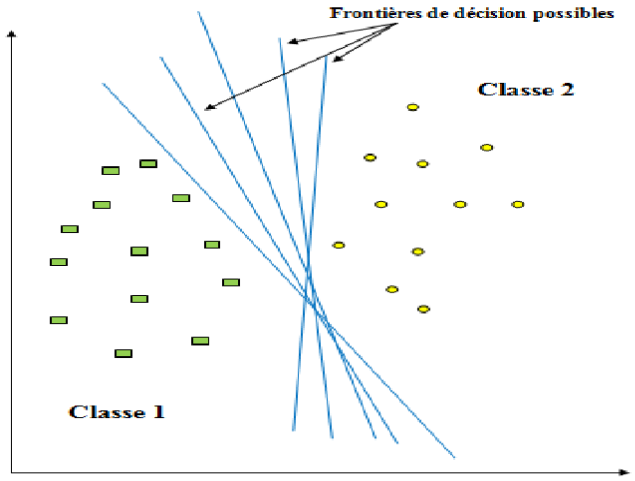

FIGURE 1.1 – Plusieurs hyperplans séparateurs

2 La marge d'un classifieur

Pour tout point dans l'éspace $\mathbb{R}^d$, la distance à l'hyperplan séparateur est donnée par :

$$d(x, \mathrm{H}_0)) = \frac{|\omega^{\mathrm{T}} x + b|}{\|\omega\|}$$

On définit les hyperplans H_1 *et* H_2 :

$$\mathrm{H}_1 := \{x \in \mathbb{R}^d : \omega^{\mathrm{T}} x + b = +1\}$$

$$\mathrm{H}_2 := \{x \in \mathbb{R}^d : \omega^{\mathrm{T}} x + b = -1\}$$

Ces deux hyperplans sont les hyperplans canonique et sont parallèle à H_0 (voir la figure (1.2)), la distance entre un point x situé dans H_1 où (H_2) à H_0 est donnée par :

$$x \in \mathrm{H}_1 \text{ où } x \in \mathrm{H}_2 : \quad d(x, \mathrm{H}_0)) = \frac{1}{\|\omega\|}$$

Les SVMs cherchent à maximiser la distance entre H_1 et H_2, cette distance appelée la Marge vaut $\frac{2}{\|w\|}$.

$$\begin{cases} x_i \ est \ dans \ la \ classe \ 1 \ si : & \omega^{\mathrm{T}} x_i + b \geq 1 \\ x_i \ est \ dans \ la \ classe \ -1 \ si : & \omega^{\mathrm{T}} x_i + b \leq -1 \end{cases} \tag{1.1}$$

Les deux inéquations de (1.1) peuvent être ressembler dans une seule équation c-à-d :

$$y_i(\omega^{\mathrm{T}} x_i + b \geq 1) \tag{1.2}$$

Le problème de détermination de ω et b revient à résoudre le problème de maximisation suivant :

$$\begin{cases} \max_{(\omega,b)} \frac{2}{\|w\|} \\ avec \quad y_i(\omega^{\mathrm{T}} x_i + b) \geq 1 \quad i = 1..n \end{cases} \tag{1.3}$$

Où a minimisé son inverse :

$$\begin{cases} \min_{(\omega,b)} \frac{1}{2}\|w\| \\ avec \quad y_i(\omega^T x_i + b) \geq 1 \qquad i = 1..n \end{cases} \tag{1.4}$$

Comme la fonction objective (1.4) est non différentiable, il est préférable de résoudre le problème équivalent suivant :

$$\begin{cases} \min_{(\omega,b)} \frac{1}{2}\|w\|^2 \\ avec \quad y_i(\omega^T x_i + b) \geq 1 \qquad i = 1..n \end{cases} \tag{1.5}$$

La constante $\frac{1}{2}$ est rajoutée pour simplifier les calculs.

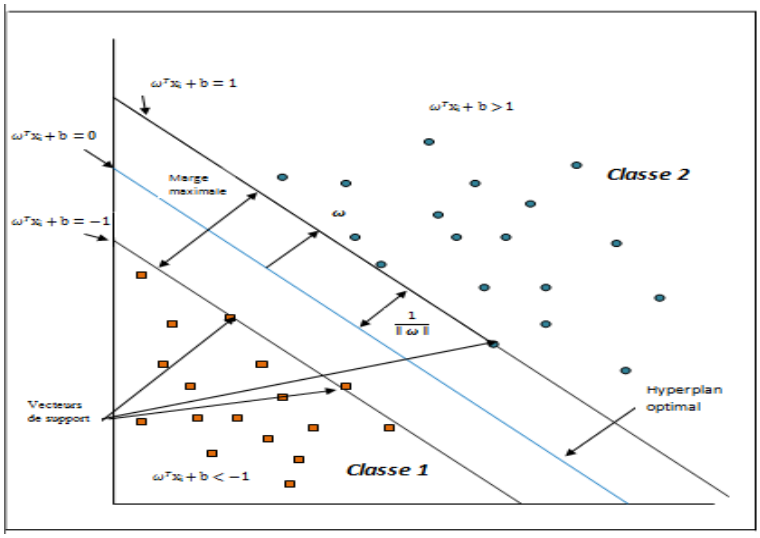

FIGURE 1.2 – Classification SVMs avec maximisation de la marge séparatrice.

3 SVMs non séparables

Nous considérons ici le cas des données bruités, cela ce traduit par des points mals classé (voir la figure (1.3)). Pour régler ce genre de problème, Vapnik [6] a introduit le concept de la marge souple « soft margin ». L'idée est toujours basée sur la recherche d'un hyperplan de marge optimale. Il est possible de suivre la même démarche adoptée dans le cas séparable au prix de l'introduction de variables d'écart ξ_i.

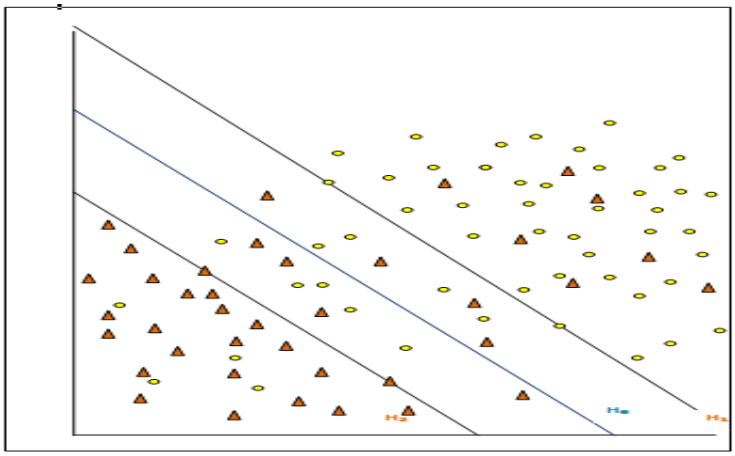

FIGURE 1.3 – Séparation de classes par les SVMs dans le cas de données non linéairement séparables

3.1 Marge poreuse (Souple)

L'idée de la marge souple est de modéliser les erreurs potentielles par des variables d'écart positives $\xi_i (i = 1,..,n)$ associées à chacune (x_i, y_i) ; $i \in [1;n]$. On a les deux cas suivants :

$$\begin{cases} bien\ \ classé\ \ :\ \ y_i(\omega^{\mathrm{T}} x_i + b \geq 1) \implies \xi_i = 0 \\ male\ \ classé\ \ :\ \ y_i(\omega^{\mathrm{T}} x_i + b < 1) \implies \xi_i = 1 - y_i(\omega^{\mathrm{T}} x_i + b) > 0 \end{cases}$$

Où les $\xi_i \in \mathbb{R}^+$ sont des réels positifs qui mesurent la violation des points mal classés, le problème d'optimisation devient :

$$\begin{cases} \min_{(\omega,b,\xi)} \ \ \frac{1}{2}\|w\|^2 + c\sum_{i=1}^{n} \xi_i \\ y_i(\omega^{\mathrm{T}} x_i + b) + \xi_i \geq 1 \ \ , i = 1,...n \\ \xi_i \geq 0 \ \ , i = 1,...,n \end{cases} \tag{1.6}$$

Où $c > 0$ est une constante qui représente la pénalité d'avoir des données mal classées. Lorsque c est très élevée, il y aura très peu de données mal classées, alors qu'il y en aura plus pour une valeur plus faible de cette constante. Le choix de c a une grande influence sur le modèle. En pratique, plusieurs modèles sont souvent construits, avec différentes valeurs de c, puis le meilleur est choisi.

3.2 Représentation duale

Notons par α et β les multiplicateurs de (KKT) pour les contraintes $y_i(\omega^{\mathrm{T}} x_i + b) + \xi_i \geq 1$ et $\xi_i \geq 0$, la fonction lagrangienne et les conditions de (KKT) peuvent être écrites comme suit :

$$\mathrm{L}(\omega, b, \alpha, \xi, \beta) = \tfrac{1}{2}\|w\|^2 + c\sum^n{}_{i=1}\xi_i - \sum_{i=1}^n \alpha_i (y_i(\omega^{\mathrm{T}} x_i b) + \xi - 1) - \sum_{i=1}^n \beta_i \xi_i$$

Les dérivées partielles de L par rapport à ω, ξ, b et β donnent : :

$$\begin{cases} \frac{\partial \mathrm{L}(\omega,b,\alpha,\xi,\beta)}{\partial \omega} = 0 \Longrightarrow \omega = \sum^n{}_{i=1}\alpha_i \; y_i \; x^i \\ \frac{\partial \mathrm{L}(\omega,b,\alpha,\xi,\beta)}{\partial b} = 0 \Longrightarrow \sum^n{}_{i=1}\alpha_i y_i = 0 \\ \frac{\partial \mathrm{L}(\omega,b,\alpha,\xi,\beta)}{\partial \xi} = 0 \Longrightarrow c = \alpha_i + \beta_i \,, \qquad i = 1, \ldots, n \\ \qquad\qquad \alpha_i((y_i(\omega^{\mathrm{T}} x_i b) + \xi - 1)) = 0 \,, \quad i = 1, \ldots, n \\ \qquad\qquad \beta_i \xi_i = 0 \,, \qquad i = 1, \ldots, n \end{cases} \tag{1.7}$$

Le problème initial d'optimisation se réduit à minimiser :

$$\begin{cases} \max -\frac{1}{2}\sum_{i=1}^n \sum_{j=1}^n \alpha_i y_i \alpha_j y_j (x^i)^{\mathrm{T}} x^j + \sum_{i=1}^n \alpha_i \\ \sum_{i=1}^n \alpha_i y_i = 0 \\ 0 \leq \alpha_i \leq c, \quad i = 1 \ldots n \end{cases} \tag{1.8}$$

4 SVM non linéaires

Pratiquement, dans le cas où les données ne sont pas linéairement classifiables, le processus de séparation est plus compliqué. Pour résoudre ce problème, le formalisme des SVMs propose d'envoyer les données dans un autre espace H de dimension supérieure à d à l'aide de la fonction non linéaire $\phi(x)$. La même procédure d'optimisation précédente est appliquée aux données prédites par $\phi(x)$ dans H.

On définit la fonction ϕ par :

$$\phi : \mathbb{R}^d \longrightarrow \mathrm{H}$$

$$x \longrightarrow \phi(x)$$

L'espace H est appelé espace des caractéristiques ou aussi espace transformé. Le principe revient donc à résoudre le problème dans l'espace H, en remplaçant $< x_i, x_j >$ par $< \Phi(x_i), \Phi(x_j) >$, $i, j = 1, \ldots, d$. On résoudre le problème :

$$\begin{cases} \max -\frac{1}{2}\sum_{i=1}^n \sum_{j=1}^n \alpha_i y_i \alpha_j y_j < \Phi(x^i), \Phi(x^j) > + \sum_{i=1}^n \alpha_i \\ \sum_{i=1}^n \alpha_i y_i = 0 \\ 0 \leq \alpha_i \leq c, \quad i = 1 \ldots n \end{cases} \tag{1.9}$$

L'hyperplan séparateur obtenu dans l'espace H est :

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i < \Phi(x_i), \Phi(x_j) >_{\mathrm{H}} + b$$

appelé hyperplan optimal généralisé.

La résolution des SVMs s'appuient sur le produit $< \Phi(x_i), \Phi(x_j) >$ dans un espace de plus grande dimension via la transformation $\phi : x \longrightarrow \phi(x)$, par la définition de fonction symétrique k le produit devient :

$$k : \mathbb{R}^d \times \mathbb{R}^d \longrightarrow \mathbb{R}$$

$$(x, x') \longrightarrow k(x, x') =< \Phi(x), \Phi(x') >_{\mathrm{H}}$$

k est appelée la fonction noyau, on peut donc reformuler le problème duale de la manière suivant :

$$\begin{cases} \max -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i y_i \alpha_j y_j k(x^i, x^j) + \sum_{i=1}^{n} \alpha_i \\ \sum_{i=1}^{n} \alpha_i y_i = 0 \\ 0 \leq \alpha_i \leq c, \quad i = 1 \ldots n \end{cases} \tag{1.10}$$

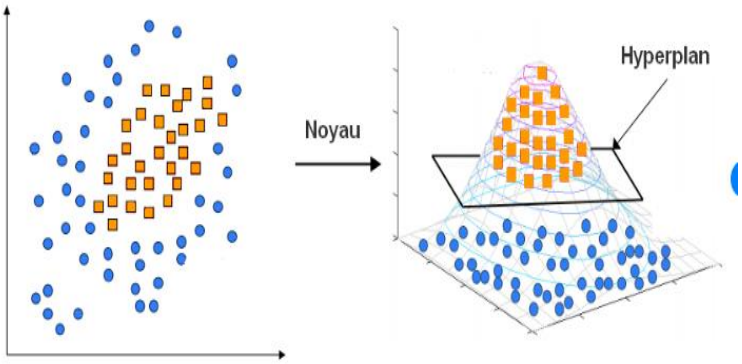

FIGURE 1.4 – Noyau appliqué aux SVMs dans le cas de données non linéairement séparables [2]

5 Conclusion

Ce chapitre est une introduction très abrégée, sur les SVMs, on peut renvoyer le lecteur intéressé au référence ([6],[16],[5]). Il apparaît clairement que le corps de SVMs est le programme quadratique (1.10). Dans ce mémoire on s'intéresse à la résolution par les méthodes de point intérieur. Il est préférable pour une bonne compréhension de cette méthode d'aborder d'abord le cas de la programmation linéaire .

Chapitre 2

La méthode de point intérieur pour la programmation linéaire

En 1984, Karmarkar [13] a publié un algorithme de type point intérieur pour la programmation linéaire, dans lequel il a annoncé qu'il avait développé un algorithme rapide qui génère des itérations situées dans la partie intérieur de l'ensemble réalisable (au lieu des limites, comme le font les méthodes simplexes), où il a ouvert de nouvelles voies passionnantes pour la recherche à la fois dans la programmation mathématique et la complexité arithmétique de nature polynomiale, il a confirmé qu'elle est plus efficace que le simplexe surtout pour des problèmes de grande taille. Depuis lors, il y a eu des recherches approfondies sur les méthodes de point intérieur, il donne comme résultat une grande variété d'algorithmes de ce type qui peuvent être classés en quatre catégories de méthodes : méthodes projectives, méthodes affines, méthodes du potentiel et méthodes de la trajectoire centrale. Dans ce chapitre, nous nous concentrons sur la méthode de trajectoire central, nous décrivons certaines idées de base derrière les méthodes du point intérieur primals-duals, y compris la relation avec la méthode de Newton et le concept du chemin central ([9],[22] et [10]).

1 Pourquoi la méthode de point intérieur ?

La méthode de point intérieur est considérée comme une nouvelle méthode de résolution pour les programmation linéaire et non linéaire, parmis leurs avantages :

1. La méthode de karmarkar [13] peut être implémentée de manière à être beaucoup plus rapide qu'aucun autre algorithme connu pour résoudre des problèmes de très grande taille, ce qui rend le temps de calcul de ces méthodes concurrentiel .

2. Ces méthodes s'exécutent en temps polynomiale (ce n'est pas le cas pour l'algorithme du simplexe, de nature exponentielle)[14], cela signifie que son temps d'exécution n'explose pas lorsque la taille du problème augmente .

3. Les idées derrière cette nouvelle méthode peuvent être utilisées dans des problèmes de programmation non linéaire [23].

2 Notions fondamentales

2.1 Couple primal-dual

Considérons le problème de programmation linéaire (P) sous la forme standard suivante :

$$(\mathrm{P}) \quad \begin{cases} \min z = p^{\mathrm{T}} x \\ \mathrm{A}x = b \\ x \geq 0 \quad x \in \mathbb{R}^{n\times 1} \end{cases}, \tag{2.1}$$

Où $\mathrm{A} \in \mathbb{R}^{m\times n}$, de rang plein, $b \in \mathbb{R}^{m\times 1}$ et $p, x \in \mathbb{R}^{n\times 1}$.

Pour tout problème (P), on définit son problème dual (D) [9] de la façon suivante :

$$(\mathrm{D}) \quad \begin{cases} \max b^{\mathrm{T}} y \\ \mathrm{A}^{\mathrm{T}} y \leq p \end{cases} \tag{2.2}$$

Ce qui est équivalent à :

$$(\mathrm{D}) \quad \begin{cases} \max b^{\mathrm{T}} y \\ \mathrm{A}^{\mathrm{T}} y + s = p \\ s \geq 0 \end{cases}, \tag{2.3}$$

Où $s \in \mathbb{R}^{n\times 1}$ désigne une variable d'écart .

Les conditions d'optimalité de (P) affirment que x est une solution de ce problème si et seulement s'il existe un couple $(y, s) \in \mathbb{R}^{m\times 1} \times \mathbb{R}^{n\times 1}$ tel que (x, y, s) vérifiant les conditions de (KKT)[22] suivantes :

$$(\mathrm{KKT}) \quad \begin{cases} \mathrm{A}x = b \\ \mathrm{A}^{\mathrm{T}} y + s = p \\ x_i s_i = 0 \quad i = 1...n \\ x \geq 0, s \geq 0 \end{cases} \tag{2.4}$$

Finalement, si nous définissons le vecteur, $e = (1, \ldots, 1)^{\mathrm{T}}$, nous pouvons reformuler les conditions d'optimalité de (KKT) sous une forme légèrement différente :

$$\begin{cases} \mathrm{A}x = b \\ \mathrm{A}^{\mathrm{T}} y + s = p \\ \mathrm{XS}e = 0 \\ x \geq 0, s \geq 0 \end{cases} \tag{2.5}$$

où X et S sont les matrices diagonales :

$$\mathrm{X} = diag(x_1, x_2, ..., x_n) = \begin{bmatrix} x_1 & & & & \\ & x_2 & & & \\ & & . & & \\ & & & . & \\ & & & & x_n \end{bmatrix}$$

$$\mathrm{S} = diag(s_1, s_2, ..., s_n) = \begin{bmatrix} s_1 & & & & \\ & s_2 & & & \\ & & . & & \\ & & & . & \\ & & & & s_n \end{bmatrix}$$

2.2 La méthode de Newton et la fonction barrière

La résolution des méthodes primales-duales est basée sur la méthode de Newton pour trouver les solutions $(x^\star, y^\star, s^\star)$ du système (2.5), cela se fait en modifiant les directions de recherche et les longueurs de pas de sort que les inégalités $(x, s) \geq 0$ seront satisfaites strictement à chaque itération.

Pour rappel, soit $\mathrm{F}_0 : \mathbb{R}^n \to \mathbb{R}^n$, on veut trouver $x \in \mathbb{R}^n$ solution de F_0 tel que : $\mathrm{F}_0(x) = 0$. La méthode de Newton consiste alors à passer de l'itéré x_k à l'itéré x_{k+1} en calculant : $x_{k+1} = x_k + \Delta x_k$. En pratique, on obtient la valeur de Δx_k par résolution du système d'équation linéaire :

$$\mathrm{J}_0(x_k)\Delta x_k = -\mathrm{F}_0(x_k)$$

De manière plus précisé, l'itéré de Newton est déterminé par :

$$\mathrm{J}_0(x, y, s)(\Delta x, \Delta y, \Delta s) = -\mathrm{F}_0(x, y, s)$$

Où $\mathrm{J}_0(x, y, s)$ est la matrice Jacobien de F_0, et la fonction $\mathrm{F}_0(x, y, s)$ est donnée par :

$$\mathrm{F}_0(x, y, s) = \begin{bmatrix} \mathrm{A}x - b \\ \mathrm{A}^{\mathrm{T}} y + s - \mathrm{P} \\ \mathrm{XS}e \end{bmatrix} = 0 \qquad (x, s) \geq 0 \tag{2.6}$$

Notons que les trois premières égalités du système (2.5) ne sont pas difficiles à résoudre par la méthode de Newton, mais le problème devient beaucoup plus difficile quand on ajoute l'exigence de non-négativité $(x, y) \geq 0$, qui donne les complications dans la conception et l'analyse des méthodes de point intérieur. Pour cela on utilise la fonction barrière.

Définition 1

Soit une fonction $\Phi : \mathbb{R} \longrightarrow \mathbb{R}$. On appelle Φ une fonction barrière si :

$$\lim_{x \longrightarrow 0} \Phi\ (x)\ = +\infty$$

Dans ces conditions, on peut remplacer le problème avec contraintes :

$$\begin{cases} \min f(x) \\ g_i(x) \geq 0 \quad pour\ \ i = 1,2,\ldots \end{cases}$$

Par le problème sans contraintes :

$$\begin{cases} \quad \min f(x) - \tau \sum_{i=1}^{n} \Phi(g_i(x)) \\ avec \qquad \tau \in \mathbb{R}^+ \end{cases} \tag{2.7}$$

Où τ est le paramètre de la fonction barrière (2.7), on peut choisir une des fonctions barrières les plus courantes, la fonction logarithme : $\phi(x) = -\ln(x)$

3 La méthode de point intérieur

3.1 La mise à l'échelle affine

Soit (x^k, y^k, s^k) l'itéré courant et $(\Delta x, \Delta y, \Delta s)$ le pas fournit par le système (2.8), l'approche affine primal-dual consiste à appliquer la méthode de Newton directement à la fonction F_0, en utilisant le pas α_k inférieur à 1 si cela est nécessaire pour maintenir la positivité des composantes de x et s. Plus précisément, à l'itération (x^k, y^k, s^k), satisfaisant $x^k > 0\ \ et\ \ s^k > 0$, on obtient une direction de Newton en résolvant le système :

$$\text{(NWT)} \qquad \begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{bmatrix} = - \begin{bmatrix} Ax^k - b \\ A^T y^k + s^k - P \\ X^k S^k e \end{bmatrix} \tag{2.8}$$

Nous pouvons écrire la dernière ligne de ce système comme suit :

$$s_i^k \Delta x_i^k\ +\ x_i^k \Delta s_i^k\ = -x_i^k s_i^k \qquad pour\ \ i = 1 \ldots n \tag{2.9}$$

Ensuite, nous définissons la nouvelle itération $(x^{k+1}, y^{k+1}, s^{k+1})$ par la formule :

$$(x^{k+1}, y^{k+1}, s^{k+1})\ =\ (x^k, y^k, s^k) + \alpha_k (\Delta x^k, \Delta y^k, \Delta s^k) \tag{2.10}$$

Le but est de définir une longueur de pas appropriée α_k pour maintenir la positivité des composantes x et s, c'est-à-dire :

$$\begin{cases} x_i^k + \alpha_k \Delta x_i^k > 0 \quad i = 1,2,\ldots,n \\ s_i^k + \alpha_k \Delta s_i^k > 0 \quad i = 1,2,\ldots,n \end{cases} \tag{2.11}$$

Il n'est pas difficile de voir que la plus grande valeur de α_k qui maintient ces conditions est donnée par la formule suivante :

$$\alpha_{\max} = \min\left(\min_{i|\Delta x^k{}_i<0} \frac{x_i^{\mathrm{K}}}{-\Delta x_i^k}, \min_{i|\Delta s^k{}_i<0} \frac{s_i^{\mathrm{K}}}{-\Delta s_i^k}\right) \tag{2.12}$$

La solution idéale serait de choisir la valeur de α_k qui évite que chaque x_i et s_i soit trop proche de zéro et qui donnera le meilleur itéré suivant. Il est préférable de choisir α_k comme suit :

$$\alpha_k = \min(1, \rho * \alpha_{\max}) \tag{2.13}$$

Où ρ est un coefficient fixe proche de 1 (la valeur typique est : 0.999), dans ce cas, α_k atteint sa longueur maximale tout en maintenant les conditions de positivité, et donc faire un grand progrès vers la solution.

Cette approche est appelée mise à l'échelle affine primal-dual, il nécessite souvent de nombreuses itérations pour arriver à la solution, les composantes x et s de l'étape Δx^k et Δs^k se déplacent très fortement vers la frontière des ensembles définis par $x \geq 0$ et $s \geq 0$. Par conséquent, la longueur de pas α_k doit être définie sur une petite valeur pour éviter de violer ces conditions.
Les approches les plus réussies utilisent des modifications de la direction de recherche qui ne se déplacent pas aussi brusquement vers la limite. C'est-à-dire que les composantes Δx^k et Δs^k sont réorientées de manière à permettre l'utilisation d'une longueur de pas α_k plus longue. Ces modifications sont définies en appliquant la méthode de Newton comme nous le verrons dans la section suivante.

3.2 La méthode de trajectoire centrale

Les méthodes de trajectoire centrale adoptent une approche moins gourmande pour satisfaire les conditions de complémentarité que les méthodes de mise à l'échelle affine, se caractérisent par un choix du paramètre τ différent de zéro. Plutôt que de viser à chaque itération de la méthode de Newton pour satisfaire ces conditions $XSe = 0$, ils utilisent des itérations de Newton pour viser les points qui réduits de leur valeur actuelle, ces algorithmes sont basés sur le concept de chemin central, que nous présentons maintenant.

Considérons le système d'équations :

$$\begin{cases} \mathrm{A}x = b \\ \mathrm{A}^{\mathrm{T}} y + s = \mathrm{P} \\ \mathrm{XS}e = \tau e \\ x > 0, s > 0 \end{cases} \tag{2.14}$$

Comme on le voit, ces conditions représentent une perturbation des conditions (KKT), dans lesquelles les produits $x_i s_i$ doivent tous prendre la même valeur positive τ au lieu d'être nuls. Pourvu qu'il y ait un vecteur triple (x, y, s) tel que x soit réalisable pour (P) avec $x > 0$ et (y, s) sont réalisables pour (D) avec $s > 0$, le système associe à chaque valeur strictement positive τ, une solution unique (x_τ, y_τ, s_τ).

Nous définissons le chemin central C comme l'ensemble des points satisfaisant pour tout $\tau > 0$:

$$C := \{(x(\tau), y(\tau), s(\tau)) \quad | \tau > 0\} \tag{2.15}$$

On appelle C le chemin central du problème primal-dual .

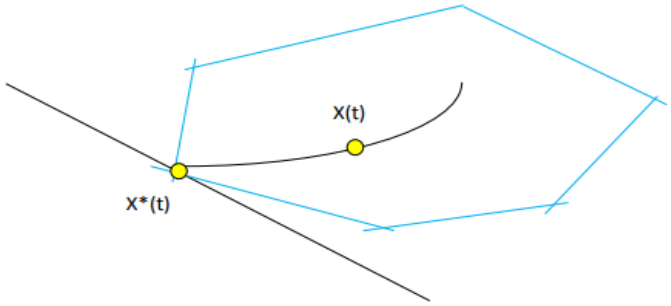

FIGURE 2.1 – Exemple de chemin central à deux dimensions

Les points de C forment un chemin par rapport auquel le saut de dualité décroît régulièrement au fur et à mesure que τ approche 0, notez que pour tous $(x(\tau), y(\tau), s(\tau)) \in C$, la mesure de dualité μ est égale à τ, c'est-à-dire :

$$\mu = \frac{X_\tau^T S_\tau}{n} = \tau. \tag{2.16}$$

Les algorithmes de trajectoire centrale restreignent les itérations à un voisinage du chemin central C et le suivent jusqu'à une solution du programme linéaire. En empêchant les itérés de s'approcher trop près de la limite non négative, ils garantissent qu'il est possible de faire un pas non trivial le long de chaque direction de recherche. De plus, en forçant la mesure de dualité μ à zéro lorsque $k \longrightarrow \infty$, nous nous assurons que les itérations (x^k, y^k, s^k) se rapprochent de plus en plus de la satisfaction des conditions (KKT). Les deux voisinages les plus intéressants de C sont celles du voisinage $N_2(\theta)$ et du voisinage $N_{-\infty}(\gamma)$ tel que :

$$N_2(\theta) = \{(x, y, s) \in J^0 \ | \ \|XSe - \mu e\|_2 < \theta\mu\}$$

Pour $\theta \in [0, 1]$, et :

$$N_{-\infty}(\gamma) = \{(x, y, s) \in J^0 \ | \ x_i s_i \geq \gamma\mu \quad , i = 1, \ldots, n\}$$

Pour $\theta \in [0, 1]$, (0,5 et 10^{-3} sont des valeurs typiques respectivement pour les paramètres γ et θ).

J^0 est l'ensemble des points strictement réalisables défini par :

$$J^0 = \{(x, y, s) | Ax = b, A^T y + s = p, (x, s) > 0\}$$

Chaque pas généré par une méthode de trajectoire central est un pas de Newton vers un point du chemin central C. Pour un $\tau \geq 0$ donné, on définit la modification suivante de la fonction F_0 en prenant les trois premières conditions du système :

$$F_\tau(x, y, s) = \begin{bmatrix} Ax - b \\ A^T y + s - P \\ XSe - \tau e \end{bmatrix} = 0 \tag{2.17}$$

Aussi, nous avons le jacobien de F_τ :

$$J_\tau = \begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \tag{2.18}$$

Les méthodes de trajectoire centrale génèrent chaque pas en appliquant la méthode de Newton à $F_{\tau_k}(x, y, s) = 0$, où : $\tau_k = \sigma_k \mu_k$, $\mu_k = \frac{x^{k^T} s^k}{n}$ est la mesure de dualité à l'itération courante (x^k, y^k, s^k) et σ_k est un paramètre(sa valeur entre (0,1)), notez que la solution $(x(\tau), y(\tau), s(\tau))$ de $F_{\tau_k}(x, y, s) = 0$ a une mesure de dualité τ_k, qui est plus petite que la mesure de dualité μ_k à l'itération k. En fixant $\sigma_k < 1$, nous visons un point qui n'est pas seulement sur le chemin central mais aussi plus avancé vers la solution que le point courant (x^k, y^k, s^k). Comme dans l'approche de mise à l'échelle affine, nous mettons à l'échelle la direction de Newton d'un pas α_k, choisie pour garantir que l'itération suivante $(x^{k+1}, y^{k+1}, s^{k+1})$ satisfait également les conditions de positivité $x^{k+1} > 0$ et $s^{k+1} > 0$.

Le paramètre σ_k est appelé paramètre de centrage. Lorsque σ_k est proche de 1, le pas de Newton pour $F_\tau(x, y, s)$ produit un point plus central que l'itération courante, mais il ne fait pas beaucoup de progrès vers la solution primal-dual. Par contre lorsque σ_k est proche de zéro, le pas de Newton se déplace plus agressivement vers l'ensemble de solutions, à la manière de pas de mise à l'échelle affine .

Les méthodes de trajectoire centrale ne prennent qu'un seul pas de Newton pour chaque valeur de τ_k à chaque itération, on réinitialise $\tau_k = \sigma_k \mu_k$ pour la valeur choisie de σ_k, ce qui produit une séquence de valeurs généralement décroissante pour τ_k. Dans un sens, les répétitions de Newton suivent une cible qui se déplace le long du chemin central vers la solution primal-dual.
Nous pouvons maintenant énoncer un cadre algorithmique simple pour une méthode de trajectoire centrale, en utilisant les paramètres décrits ci-dessus.

Algorithm 1 ***Algorithme de trajectoire centrale (Path-Following)***

Choisissez $\sigma_{\min}$ et $\sigma_{\max}$ tel que $0 < \sigma_{\min} < \sigma_{\max} < 1$;
Choisissez le point initial (x^0, y^0, s^0) avec $x^0 > 0$ et $s^0 > 0$;
pour $k = 0, 1, \ldots$.
Choisissez $\sigma_k \in [\sigma_{\min}, \sigma_{\max}]$;
Résoudre pour $(\Delta x^k, \Delta y^k, \Delta s^k)$:

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{bmatrix} = - \begin{bmatrix} Ax^k - b \\ A^T y^k + s^k - P \\ X^k S^k e - \sigma_k \mu_k e \end{bmatrix}$$

Où :

$$\mu_k = \frac{(x^k)^T s^k}{n}$$

Choisissez $\alpha_{\max}$ comme étant la plus grande valeur positive de α tel que :

$$(x^k, s^k) + \alpha(\Delta x^k, \Delta s^k) \geq 0$$

Mettre $\alpha_k = \min(1, \rho_k * \alpha \max)$ pour certains $\rho_k \in (0, 1)$
Mettre $(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k, y^k, s^k) + \alpha_k(\Delta x^k, \Delta y^k, \Delta s^k)$;

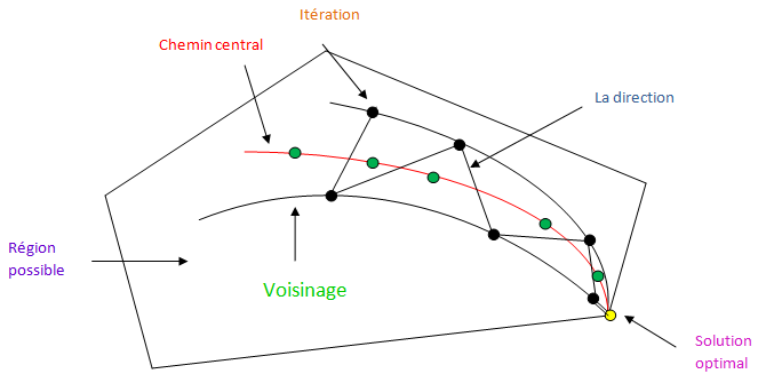

FIGURE 2.2 – Quelques itérés d'une méthode de trajectoire central

4 Complexité de l'algorithme

4.1 Taille de problème

On peut simplement mesurer la taille du problème par n (la dimension du vecteur à optimiser) et m (le nombre de contraintes). On peut supposer que les éléments A, b et c sont des entiers. Chaque entier k occupe un nombre de bits égal au plus petit entier supérieur à $log_2(k)+1$. Etant donné que A, b et c comportent respectivement $m \times n$, m et n entiers.
On définit la taille du problème par :

$$\mathrm{L} = \mathrm{L}_0 + (mn + m + n) \quad avec \quad \mathrm{L}_0 = \sum_{k \in \mathrm{A}, b, c} [\log_2 k] + 1$$

Où L_0 est la somme des tailles (en bits) des entiers de A, b et c et le terme $(mn+m+n)$ correspond au nombre de bits nécessaires pour séparer ces entiers [11].

4.2 Complexité

Sous quelques hypothèses supplémentaires et des conditions légèrement différentes sur le choix de α_k, on peut montrer que l'algorithme ci-dessus a une propriété de complexité polynomiale. Plus précisément, étant donné une petite tolérance ϵ, un point de départ (x^0, y^0, s^0) qui satisfait J^0 et étant proche du chemin central dans un certain sens, et une stratégie de sélection de α_k qui conserve toutes les itérations dans un certain voisinage du chemin central, la méthode converge vers un point (x, y, s) pour lequel $x^{\mathrm{T}} s \leq \epsilon$ dans $(n \log \frac{(x^0)^{\mathrm{T}} s^0}{\epsilon})$ itérations [9].

5 Conclusion

Dans ce chapitre, nous avons présenté la méthode du point intérieur pour la programmation linéaire. Nous avons défini le concept du chemin central et la méthode de Newton. Nous nous sommes également concentrés sur la méthode de trajectoire centrale. Les méthodes du point intérieur ne se sont pas limitées seulement à la programmation linéaire, mais ont été développées pour inclure la programmation quadratique, les concepts de base restent les mêmes. Nous vous présentons un algorithme de point intérieur dans le cas quadratique, vu que le problème qui nous intéresse est le programme quadratique pour les SVMs.

Chapitre 3

La méthode de point intérieur pour la programmation quadratique

Il existe plusieurs méthodes pour la résolution des problèmes quadratiques avec contraintes linéaires. On peut citer à titre d'exemples, la méthode Active-set [22] qui n'est rien d'autre que la généralisation de la méthode simplexe au cas quadratique, les méthodes de type gradient [1], comme la méthode du gradient projeté.
La méthode de point intérieur, introduite pour la programmation linéaire dans le chapitre précédent, peut également être généralisée aux cas quadratique convexe .

1 Position du problème

On considère le problème d'optimisation quadratique convexe sous contraintes linéaires suivant :

$$\text{(CQP)} \quad \begin{cases} \min f(x) = \frac{1}{2} x^{\mathrm{T}} \mathrm{Q} x + p^{\mathrm{T}} x \\ \quad \mathrm{A}x = b \\ \quad x \geq 0 \quad x \in \mathbb{R}^{n \times 1} \end{cases} \tag{3.1}$$

Où $\mathrm{Q} \in \mathbb{R}^{n \times n}$ est une matrice symétrique semi définie positive, $\mathrm{A} \in \mathbb{R}^{m \times n}$, de rang plein, $b \in \mathbb{R}^{m \times 1}$ et $p, x \in \mathbb{R}^{n \times 1}$.

On définit son problème dual (CQD)[1] :

$$\text{(CQD)} \quad \begin{cases} \max -\frac{1}{2} x^{\mathrm{T}} \mathrm{Q} x + b^{\mathrm{T}} y \\ \quad \mathrm{A}^{\mathrm{T}} y \leq \mathrm{Q}x + p \\ \quad y \in \mathbb{R}^{m \times 1} \end{cases} \tag{3.2}$$

ce qui est équivalent à :

$$\text{(CQD)} \quad \begin{cases} \max -\frac{1}{2} x^{\mathrm{T}} \mathrm{Q} x + b^{\mathrm{T}} y \\ \quad \mathrm{A}^{\mathrm{T}} y + s = \mathrm{Q}x + p \\ \quad s \geq 0, y \in \mathbb{R}^{m \times 1} \end{cases} , \tag{3.3}$$

où $s \in \mathbb{R}^{n\times 1}$ désigne une variable d'écart .

Les conditions d'optimalité de (KKT) [22] du problème présente comme suite :

$$(\text{KKT}) \quad \begin{cases} \mathrm{A}x = b \\ \mathrm{A}^{\mathrm{T}}y + s - \mathrm{Q}x = p \\ \mathrm{XS}e = 0 \\ x \geq 0, s \geq 0 \end{cases}, \tag{3.4}$$

où e est le vecteur défini par : $e = (1,\ldots,1)^{\mathrm{T}}$, et X , S sont les matrices diagonales : $\mathrm{X} = diag(x_1, x_2, \ldots, x_n), \mathrm{S} = diag(s_1, s_2, \ldots, s_n)$.
Étant donné un itéré courant (x, y, s) qui satisfait (3.4), nous pouvons définir la mesure de dualité μ comme :

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i s_i = \frac{\mathrm{X}^{\mathrm{T}}\mathrm{S}}{n} \tag{3.5}$$

2 Direction de Newton

Définissons maintenant l'application non linéaire suivante $\mathrm{F} : \mathbb{R}^{2n+m} \rightarrow \mathbb{R}^{2n+m}$

$$\mathrm{F}(x, y, s) = \begin{bmatrix} \mathrm{A}x - b \\ \mathrm{A}^{\mathrm{T}}y + s - \mathrm{Q}x - p \\ \mathrm{XS}e \end{bmatrix} = 0 \qquad (x, s) \geq 0 \tag{3.6}$$

Notez que nous avons le même principe déjà mentionné dans le chapitre 2, les méthodes de points intérieurs de type primal-dual calculent les solutions (x, y, s) du système (3.4) en appliquant la méthode de Newton à $\mathrm{F}(x, y, s) = 0$, à condition que les inégalités $x^k > 0$ et $s^k > 0$ restent satisfaites à chaque itération. Cependant la solution obtenue peut ne pas vérifier les condition $(x, s) \geq 0$.

3 La fonction barrière

Pour introduire une méthode de point intérieur pour résoudre les problèmes (CQP) et (CQD), on peut associer au problème (CQP), le problème paramétrisé par la fonction barrière (CQP_τ) En remplaçant toutes les contraintes non négatives dans le problème primal (CQP) nous obtenons le sous-problème barrière suivant :

$$(\mathrm{CQP}_\tau) \begin{cases} \min f_\tau(x) = \frac{1}{2}x^{\mathrm{T}}\mathrm{Q}x + p^{\mathrm{T}}x - \tau\sum_{i=1}^{n}\ln(x_i) \\ \mathrm{A}x = b \\ x > 0 \quad , x \in \mathbb{R}^{m\times 1} \quad , \tau > 0 \end{cases} \tag{3.7}$$

Où la fonction : $x \rightarrow -\tau\sum_{i=1}^{n}\ln(x_i)$ est définie par (2.7)

La solution $x(\tau)$ est caractérisé d'une manière unique par les conditions d'optimalité de (KKT) suivantes :

$$\begin{cases} Ax = b \\ Qx + p - \tau X^{-1} e - A^T y = 0 \end{cases} \tag{3.8}$$

Dans ce cas, On peut poser $S = \tau X^{-1}$ avec $S \in \mathbb{R}^n_+$, le système (KKT) devient :

$$(KKT_\tau) \quad \begin{cases} Ax = b \\ A^T y + S = Qx + P \\ XSe = \tau e \\ X > 0 \ , S > 0 \end{cases} \tag{3.9}$$

D'après la dernière équation du système (3.9), on a $x_i(\tau)s_i(\tau) = \tau$ pour tout $1 \leq i \leq n$, donc $x^t s = n\tau$: où $x^t s$ désigne le saut dualité de ce système.

4 La méthode de point intérieur de type primal-dual

4.1 La méthode de trajectoire centrale

Rappelons qu'un chemin central C est un ensemble des points réalisables stricts qui joue un rôle de guide vers la solution optimale dans les algorithmes de type primal-dual. Il est paramétré par un scalaire $\tau > 0$. En appliquant la méthode de Newton pour le système (KKT_τ), chaque point (x_τ, y_τ, s_τ) du chemin central C résout le système :

$$(NWT_\tau) \quad \begin{bmatrix} A & 0 & 0 \\ -Q & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} b - Ax \\ P + Qx - A^T y - S \\ \tau e - XSe \end{bmatrix} \tag{3.10}$$

Les produits $x_i s_i$ doivent tous prendre la même valeur positive τ au lieu d'être nuls, si C converge lorsque τ tend ver zéro, il doit converger vers la solution primale-duale du système (3.10).

Le principe de la méthode de trajectoire central reste toujours le même, nous résolvons le système d'équations non linéaire de (KKT_τ) à partir d'un point initial $(x_0, s_0) > 0$, $y \in \mathbb{R}^{m\times 1}$ avec la mesure de dualité μ qui est définie par :

$$\mu = \frac{x^T s}{n} = n\tau \tag{3.11}$$

.

En posant $\tau = \sigma\mu$, où σ [1] est un paramètre de centrage, nous calculons les directions de Newton $(\Delta x^k, \Delta y^k, \Delta s^k)$ en résolvant le système suivant :

$$\begin{bmatrix} A & 0 & 0 \\ -Q & A^T & I \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{bmatrix} = \begin{bmatrix} b - Ax^k \\ P + Qx^k - A^T y^k - S \\ \sigma_k \mu_k e - X^k S^k e \end{bmatrix} \tag{3.12}$$

1. le choix de σ dépend de la méthode considérée

Une fois $(\Delta x^k, \Delta y^k, \Delta s^k)$ calculé, nous définissons la nouvelle itération $(x^{k+1}, y^{k+1}, s^{k+1})$ par la formule :

$$(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k, y^k, s^k) + \alpha_k(\Delta x^k, \Delta y^k, \Delta s^k) \tag{3.13}$$

Où $\alpha_k > 0$ est le pas de déplacement qui doit être déterminé de telle manière que $(x^{k+1}, s^{k+1}) > 0$ restent satisfait a chaque itération [1].

4.2 La méthode de trajectoire centrale à prédiction-correction

Principe de La méthode

Parmis tous les algorithmes utilisés pour les méthodes de point intérieur, la méthode trajectoire central à prédiction-correction est l'un des algorithmes les plus populaires pour la programmation quadratique convexe, c'est d'allier l'algorithme utilisé dans la fonction Matlab 'quadprog' [15]. L'idée est de commencer par calculer une direction de recherche d'optimisation basée sur un terme du premier ordre (prédicteur). La taille de pas qui peut être prise dans cette direction est utilisée pour évaluer le degré de correction de centralité nécessaire. Ensuite, un terme correcteur est calculé : il contient à la fois un terme de centralité et un terme du second ordre.

En posant $\sigma = 0$, la méthode prédicteur-correcteur fonctionne en utilisant la méthode de Newton pour obtenir la direction de mise à l'échelle affine. Ceci est réalisé en résolvant le système d'équations linéaires suivant[12] :

$$(\mathrm{NWT}_{aff}) \quad \begin{bmatrix} \mathrm{A} & 0 & 0 \\ -\mathrm{Q} & \mathrm{A}^{\mathrm{T}} & \mathrm{I} \\ \mathrm{S} & 0 & \mathrm{X} \end{bmatrix} \begin{bmatrix} \Delta x^{aff} \\ \Delta y^{aff} \\ \Delta s^{aff} \end{bmatrix} = \begin{bmatrix} b - \mathrm{A}x \\ \mathrm{P} + \mathrm{Q}x - \mathrm{A}^{\mathrm{T}}y - \mathrm{S} \\ \sigma\mu e - \mathrm{XS}e \end{bmatrix} \tag{3.14}$$

Ensuite, nous calculons les longueurs de pas maximales autorisées le long de la direction de mise à l'échelle affine par la formule :

$$\begin{cases} \alpha_{aff}^{pri} = \min\left(1, \min_{i:\Delta x_i^{aff}<0} -\frac{x_i}{\Delta x_i^{aff}}\right) \\ \alpha_{aff}^{dual} = \min\left(1, \min_{i:\Delta s_i^{aff}<0} -\frac{s_i}{\Delta s_i^{aff}}\right) \end{cases} \tag{3.15}$$

En utilisant les définitions ci-dessus, nous calculons μ_{aff} conformément à la définition de μ par :

$$\mu_{aff} = (x + \alpha_{aff}^{pri}\Delta x^{aff})^{\mathrm{T}}(s + \alpha_{aff}^{dual}\Delta s^{aff})/n \tag{3.16}$$

Le paramètre de centrage σ est choisi selon l'heuristique [2] suivante :

$$\sigma = \left(\frac{\mu_{aff}}{\mu}\right)^3 \tag{3.17}$$

2. Il n'a pas de justification analytique forte, mais il semble bien fonctionner dans le pratique

Puis on définit le pas de correction qui vise à améliorer l'étape de mise à l'échelle affine comme suit :

$$(\mathrm{NWT}_{cor}) \quad \begin{bmatrix} \mathrm{A} & 0 & 0 \\ -\mathrm{Q} & \mathrm{A}^{\mathrm{T}} & \mathrm{I} \\ \mathrm{S} & 0 & \mathrm{X} \end{bmatrix} \begin{bmatrix} \Delta x^{cor} \\ \Delta y^{cor} \\ \Delta s^{cor} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -\Delta x^{aff} \Delta s^{aff} \end{bmatrix} \tag{3.18}$$

finalement , le pas total est obtenu en résolvant le système suivant :

$$(\mathrm{NWT}_{pre-cor}) \quad \begin{bmatrix} \mathrm{A} & 0 & 0 \\ -\mathrm{Q} & \mathrm{A}^{\mathrm{T}} & \mathrm{I} \\ \mathrm{S} & 0 & \mathrm{X} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} b - \mathrm{A}x \\ \mathrm{P} + \mathrm{Q}x - \mathrm{A}^{\mathrm{T}} y - \mathrm{S} \\ \sigma\mu e - \Delta x^{aff} \Delta s^{aff} - \mathrm{XS}e \end{bmatrix} \tag{3.19}$$

Nous avons rassemblé tous les étapes mentionnés ci-dessus pour un algorithme pratique réussi dans le tableau ci-dessous.

Algorithm 2 Algorithme de trajectoire centrale prédiction-correction

Début Algorithme
Choisissez (x^0, y^0, s^0) avec $(x^0, s^0) > 0$.
Mettre $\sigma \in [0,1]$.
pour $k = 0, 1, \ldots, n$
Mettre $(x, y, s) = (x_k, y_k, s_k)$ $et\ calculer$ $(\Delta x^{aff}, \Delta y^{aff}, \Delta s^{aff})$ avec $\sigma = 0$ par résoudre le système (3.14).
Calculer $\mu = \frac{x^{\mathrm{T}} s}{n}$.
Calculer $\tilde{\alpha}_{aff} = \max\{(x + \alpha_{aff}^{pri} \Delta x^{aff}), (s + \alpha_{aff}^{dual} \Delta s^{aff})\}$.
Calculer $\mu_{aff} = (x + \tilde{\alpha}_{aff} \Delta x^{aff})(s + \tilde{\alpha}_{aff} \Delta s^{aff})^{\mathrm{T}} / n$.
Mettre le paramètre de centrage $\sigma = \left(\frac{\mu_{aff}}{\mu}\right)^3$
calculer $(\Delta x, \Delta y, \Delta s)$ par résolution du système (3.19).
Sélectionnez α_k^{pri} et α_k^{dual}
Mettre $\tilde{\alpha} = \min(\alpha_k^{pri}, \alpha_k^{dual})$
Mettre $(x_{k+1}, y_{k+1}, s_{k+1}) = (x_k, y_k, s_k) + \tilde{\alpha}(\Delta x_k, \Delta x_k, \Delta s_k)$
$k = k + 1$
Fin pour

5 Conclusion

Dans ce chapitre, nous avons fait une présentation illustrée de la méthode de point intérieur pour la programmation quadratique, pour cela nous avons d'abord rappelé les notions fondamentales sur la programmation quadratique convexe , nous avons introduit les conditions d'optimalités de (KKT), et nous avons utilisé la méthode de Newton et le terme barrière pour définir de nouvelles directions de recherche, ensuite, nous avons annoncé la méthode du trajectoire central spécifiquement le principe de l'algorithme prédicteur correcteur . Nous allons appliqué cette méthode pour le problème quadratique SVMs et voir son impacte sur le problème détection de visage.

Chapitre 4

Application au problème de détection de visage dans une image

La détection d'un visage dans une image, consiste à développer un algorithme qui, peut détecter et localiser des visages humains dans cette image. Il suscite de plus en plus d'intérêt en tant que domaine de recherche important avec des applications importantes dans la sécurité, le contrôle d'accès, interaction homme-machine et autre domaine de la vision par ordinateur et de l'apprentissage automatique (Szeliski, 2011 [20] ; Zafeiriou et al., 2015 [24]). De nombreuses méthodes ont été développées avec plusieurs approches [18], malgré le succès de certains de ces systèmes dans des scénarios contraints, la tâche générale de détection faciale pose encore un certain nombre de défis en ce qui concerne les changements , d'expression faciale, pose et temps d'exécution.
La plupart des approches utilisées reposent sur deux tâches. Le premier est la tâche d'extraction de caractéristiques, il s'agit de la description numérique précise de ce qui distingue les visages humains des autres objets. La deuxième tâche est généralement un algorithme de classification supervisé qui devra reconnaître avec une bonne précision un visage d'un non-visage. Comme nous avons opté dans notre travail sur la classification par SVMs [6], en particulier nous avons appliqué une méthode de point intérieur (IPM) [1] pour résoudre le problème quadratique qui détermine le classifieur SVMs, ceci va nous permettre d'appliquer le code de O. Sakhi [18] avec une légère modification. Nous avons utilisé IPM, au lieu de l'algorithme SMO [17] (dans ce code). Bien que nous avons agi uniquement sur le volet classification, ceci ne va pas nous empêcher de dire un mot sur l'extraction de caractéristique utilise dans ([18]).

1 Extraction des caractéristiques

Afin de construire le classifieur, il est nécessaire d'avoir un ensemble de données constitué à deux classes (visage et non visage) (voire les figures ci-dessous).

FIGURE 4.1 – Visage

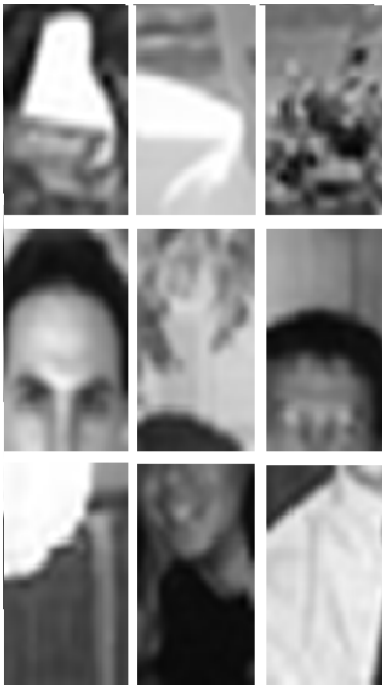

FIGURE 4.2 – Non visage

Extraire les caractéristiques dans une image, dans ce cadre, consiste à transformer une image (27×18) à un vecteur uni-colonne de $\mathbb{R}^{19440\times 1}$, en utilisant le Filtre de Gabor [19]. Dans un premier temps il est nécessaire de traiter l'image [21]. En effet la fonction Matlab 'adapthisteq' est utiliser pour améliorer l'image. Le produit de la transforme de Fourier entre l'image (visage) et les Filtre de Gabor est utilise au lieu de calculer le produit de convolution [21] qui est trop coûteux .

L'inverse de la transformée de Fourier est applique à ce produit pour rendre l'image dans son domaine spaciale. La taille de l'image résultante est (135×144), une vectorisation de ce vecteur utilisée par la fonction 'reshape' de Matlab, afin d'avoir une donnée de dimension $\mathbb{R}^{19440\times 1}$ (voire la figure(4.3)).

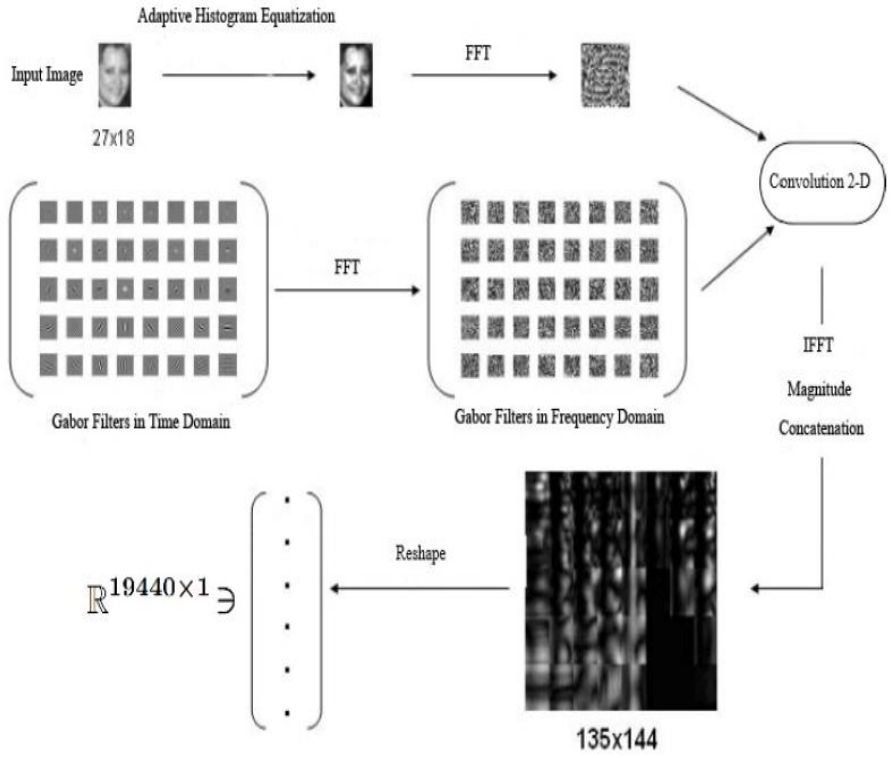

FIGURE 4.3 – Extraction des caractéristiques par le Filtre de Gabor

Finalement nous disposons d'une matrice des données $X \in \mathbb{R}^{19440\times 910}$, où 19440 est le nombre de caractéristiques de chaque donnée et 910 est le nombre de donnés visages et non visages (classification binaire).

2 Classification

Dans le code [18], l'auteur utilise la fonction 'svmtrain' de Matlab. La fonction 'svmtrain' de Matlab applique la méthode 'SMO' [17] par défaut pour résoudre le problème quadratique des SVMs [6]. Dans notre travail nous avons utilisé la méthode de point intérieur [1]. Ceci consiste simplement à introduire comme option la méthode de (IPM) dans la fonction 'svmtrain'.

$$"options = optimset('Algorithm','interior-point')"$$
$$"net = svmtrain(P','T','Kernel_Function','linear','options')";$$

Nous avons considéré, deux images test dans les données du code sakhi [18].

FIGURE 4.4 – Exemple 1

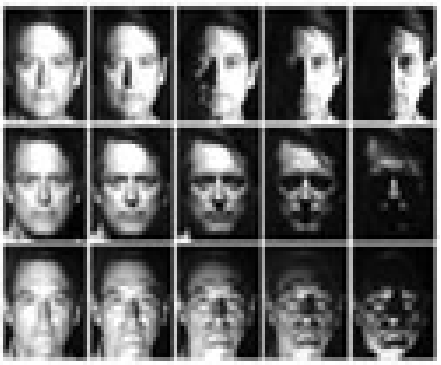

FIGURE 4.5 – Exemple 2

Le résultat est donné dans les figures ci-dessous .

FIGURE 4.6 – svmtrain-IPM

FIGURE 4.7 – svmtrain-IPM

FIGURE 4.8 – svmtrain-SMO

FIGURE 4.9 – svmtrain-SMO

Conclusion générale

La détection des visages est l'un des domaines les plus intrigants de la vision par ordinateur, qui représente une branche de l'intelligence artificielle, cela ne nous empêche pas de l'envisager d'un point de vue mathématiques.

Dans cette étude, nous utilisons la méthode du point intérieur [1] comme méthode pour résoudre le problème quadratique qui détermine le classifieur SVMs [6]. Nous avons étudier le comportement de cette méthode, où nous l'avons appliqué dans le domaine de la détection de visage, en utilisant le code de O. Sakhi [18], et nous l'avons comparée à une autre méthode d'optimisation quadratique appelée SMO [17], les résultats étaient presque identiques.

Ce travail nous a permis de constater, que l'optimisation est présente dans plusieurs algorithmes de classification supervisée en particulier, et en générale dans les algorithmes d'apprentissage automatique. Ceci nous donne comme perspective d'étudier d'autres méthodes de classification.

Bibliographie

[1] Antoniou, A., and Lu, W. S. (2007). Practical optimization : algorithms and engineering applications. Springer Science and Business Media. 4, 21, 24, 26, 28, 30

[2] Bonazza, P. (2019). Système de sécurité biométrique multimodal par imagerie, dédié au contrôle d'accès (Doctoral dissertation, Bourgogne Franche-Comté). 11

[3] Bonissone, P., Cadenas, J. M., Garrido, M. C., and Díaz-Valladares, R. A. (2010). A fuzzy random forest. International Journal of Approximate Reasoning, 51(7), 729-747. 4

[4] Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992, July). A training algorithm for optimal margin classifiers. In Proceedings of the fifth annual workshop on Computational learning theory (pp. 144-152). 4, 6

[5] Boswell, D. (2002). Introduction to support vector machines. Departement of Computer Science and Engineering University of California San Diego. 11

[6] Cortes, C., and Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273-297. 4, 6, 9, 11, 26, 28, 30

[7] Desjardins, J. (2005). L'analyse de régression logistique. Tutorial in quantitative methods for psychology, 1(1), 35-41. 4

[8] Dreyfus, G., Martinez, J. M., Samuelides, M., Gordon, M. B., Badran, F., Thiria, S., and Hérault, L. (2002). Réseaux de neurones (Vol. 39). Paris : Eyrolles. 4

[9] Ferris, M. C., Mangasarian, O. L., and Wright, S. J. (2007). Linear programming with MATLAB. Society for Industrial and Applied Mathematics. 4, 12, 13, 20

[10] Glavic, M. (2004). Interior point methods : A survey, short survey of applications to power systems, and research opportunities. University of Liege. 12

[11] GLINEUR, F. (1997). ETUDE DES METHODES DE POINT INTERIEUR APPLIQUEES A LA PROGRAMMATION LINEAIRE ET A LA PROGRAMMATION SEMIDEFINIE. 20

[12] Hungerländer, P. (2014). Algorithms for convex quadratic programming. arXiv preprint arXiv :1409.5222. 24

[13] Karmarkar, N. (1984, December). A new polynomial-time algorithm for linear programming. In Proceedings of the sixteenth annual ACM symposium on Theory of computing (pp. 302-311). 4, 12

[14] Klee, V., and Minty, G. J. (1972). How good is the simplex algorithm. Inequalities, 3(3), 159-175. 12

[15] Mehrotra, S. (1992). On the implementation of a primal-dual interior point method. SIAM Journal on optimization, 2(4), 575-601. 24

[16] Naima, D., « Optimisation Quadratique pour les Machines à Vecteurs de Support (SVMs)», Thèse de doctorat en Mathématiques Appliquées, sous la direction Abdessamad AMIR, Université de Mostaganem,le 03 juillet 2018. 11

[17] Platt, J. (1999) 'Sequential minimal optimization : a fast algorithm for training support vector machines', in Scholkopf, B. et al. (Eds.) : Advances in Kernel Methods : Support Vector Learning, pp.185–208, MIT Press, Cambridge, MA, USA. 26, 28, 30

[18] Sakhi, O. (2020) Face Detection using Support Vector Machine (SVM), MATLAB Central File Exchange [online] site web :(-face-detection-using-support-vector-machine-svm) 26, 28, 29, 30

[19] Shen, L., Bai, L., and Fairhurst, M. (2007). Gabor wavelets and general discriminant analysis for face identification and verification. Image and Vision Computing, 25(5), 553-563. 27

[20] Szeliski, R. (2011) Computer Vision : Algorithms and Applications, Springer, London 26

[21] T, MathWorks, Inc. MATLAB Image Processing Toolbox™ User's Guide [R2020a ed.] 2,755 282 67MB. English. Pages 1286. Year 2020. 27

[22] Wright, S., and Nocedal, J. (1999). Numerical optimization. Springer Science, 35(67-68), 7. 4, 12, 13, 21, 22

[23] Ye, Y., and Tse, E. (1989). An extension of Karmarkar's projective algorithm for convex quadratic programming. Mathematical programming, 44(1), 157-179. 12

[24] Zafeiriou, S., Zhang, C. and Zhang, Z. (2015) 'A survey on face detection in the wild : past, present and future', Computer Vision and Image Understanding, Vol. 138, No. C, pp.1–24. 26