République Tunisienne
Ministère de l'Enseignement Supérieur
et de la recherche scientifique

Université de Gafsa
Institut Supérieur des Sciences
Appliquées et de la Technologie de Gafsa

Cycle de Formation en Mastère
Professionnelle dans la
Discipline Science de données

Mémoire de *MASTERE*
Science des données
N° d'ordre : 02−MSD

MEMOIRE

Présenté à

L'Institut Supérieur des Sciences Appliquées et de Technologie de Gafsa

(Département Informatique et télécommunication
En vue de l'obtention Diplôme en MPSD

MASTERE

Dans la discipline Sciences des données

Par
Raja Taleb

Reconnaissance des formes basées sur les expressions régulières

Soutenu devant les membres de jury composé de :

M.	**Saïd Taieb**	*Encadreur*
M.	**Moez Miraoui**	*Président*
Mme	**Hayfa Chorfi**	*Rapporteur*

A.U : 2022 – 2023

Dédicace

Je dédie ce travail à

À mes parents

Qui n'ont jamais cessé de formuler des prières à mon égard, de me soutenir et de m'épauler pour que je puisse atteindre mes objectifs.

À mes frères et mes sœurs,

Pour leurs soutiens moral et leurs conseils précieux tout au long de mes études

À mes chères amies,

Pour leurs aides et supports dans les moments difficiles

À tous ceux que j'aime

.....

✍Raja

Remerciements

Nos remerciements vont à l'encontre de notre encadrant qui nous a aidé à travers ses remarques pertinentes et ses qualités exceptionnelles.

Mr. **Saïd TAIEB**

Nous tenons également à exprimer notre gratitude aux **Membres de jury** qui ont eu l'amabilité de juger ce travail.

Nous remercions également tous **nos ami(e)s** pour leurs aides et on souhaite a tous le succès et la réussite dans leurs parcours de vie.

Table des matières

Liste des figures

Liste des tableaux

Liste des Abréviations

CNN: Convolutional **N**eural **N**etwork

ML: Machine **L**earning

RegEx: Regular **Ex**pressions

SVC:**S**upport **V**ector **C**lassification

SVM: **S**upport **V**ector**M**achine

SVR: **S**upport **V**ector**R**egression

Introduction générale

La reconnaissance de forme est un domaine de recherche passionnant dans le domaine de l'informatique et de la vision par ordinateur. Elle consiste à développer des algorithmes et des modèles capables d'identifier et de classifier des formes dans des images ou des données visuelles.

Le présent rapport propose une étude approfondie de la reconnaissance de forme et explore les différentes méthodes, algorithmes et techniques utilisés dans ce domaine. Ce rapport vise à fournir une compréhension approfondie de la reconnaissance de forme et à présenter les différentes méthodes et techniques utilisées dans ce domaine. Il offre également une perspective sur les applications potentielles de la reconnaissance de forme dans divers domaines.

Le rapport est structuré en plusieurs chapitres qui abordent divers aspects de la reconnaissance de forme.

Le premier chapitre, intitulé "Etat de l'art", offre une introduction générale à la reconnaissance de forme. Il présente les principes de base de la reconnaissance de forme, les étapes impliquées dans le processus de reconnaissance de forme, ainsi que la modélisation par expression régulière, qui est une méthode couramment utilisée dans ce domaine.

Le deuxième chapitre, intitulé "Algorithmes de reconnaissance de forme", explore en détail les différents algorithmes utilisés dans la reconnaissance de forme. Il examine les approches basées sur les caractéristiques, les modèles et les approches hybrides. De plus, il aborde l'évaluation des algorithmes de reconnaissance de forme et les domaines d'application de cette discipline.

Le troisième chapitre, intitulé "Algorithmes de Freeman pour la reconnaissance de forme", se concentre spécifiquement sur les algorithmes de Freeman, qui sont largement utilisés dans la reconnaissance de forme géométrique. Il présente le modèle de Freeman, l'extraction des caractéristiques et la classification des formes utilisant ces algorithmes.

Le quatrième chapitre, intitulé "Implémentation", décrit l'environnement de développement utilisé dans le cadre de ce projet, ainsi que les bibliothèques Python utilisées. Il présente également les modèles et la comparaison des résultats.

Enfin, le rapport se termine par une conclusion qui résume les principales conclusions de notre l'étude.

Chapitre 1 :

Etat de l’art

Chapitre1

Etat de l'art : Reconnaissance de forme

Introduction

Avant l'élaboration de tout projet, il est nécessaire de faire une présentation globale des notions théoriques. De ce fait, nous dédions ce premier chapitre à la présentation de l'état de l'art de la modélisation des formes usuelles et les expressions régulières comme étant un moyen puissant et flexible de représenter ces formes de manière concise et précise.

1. Reconnaissance de forme

1.1.Principe de reconnaissance de forme

Le domaine de reconnaissance de formes est un domaine de l'intelligence artificielle qui vise à développer des algorithmes pour identifier des modèles dans des données. Cela peut inclure des images, des vidéos, des signaux sonores, des textes, etc.

Le principe de reconnaissance de forme est une technique utilisée pour identifier des objets ou des formes dans des images, des vidéos, du texte ou d'autres types de données. Le but est de trouver des modèles dans les données qui correspondent à une forme ou un objet spécifique, même si ces formes sont légèrement modifiées ou altérées. La reconnaissance de forme est un processus complexe qui utilise des algorithmes de traitement d'image, de traitement de signal et de statistiques pour extraire les caractéristiques des données et les comparer à des modèles préétablis.

Parmi les outils de reconnaissance des formes plus récentes on peut citer ;

- **Réseaux de neurones profonds :** Les réseaux de neurones profonds sont actuellement les plus populaires dans le domaine de la reconnaissance de formes. Ils ont permis des avancées significatives dans la reconnaissance d'images, de la reconnaissance vocale et de la reconnaissance de texte. Les réseaux de neurones profonds sont capables d'apprendre à partir de grandes quantités de données et de produire des résultats précis.
- **Méthodes basées sur les caractéristiques :** Les méthodes basées sur les caractéristiques sont utilisées pour extraire des caractéristiques à partir de données brutes. Ces caractéristiques peuvent être utilisées pour entraîner des algorithmes de reconnaissance de formes. Les méthodes basées sur les caractéristiques comprennent

des techniques telles que la transformation de Fourier, la transformation en ondelettes et l'analyse en composantes principales.

- **Apprentissage par renforcement :** L'apprentissage par renforcement est une méthode d'apprentissage automatique dans laquelle un agent apprend à prendre des décisions en interagissant avec son environnement. Cette méthode est souvent utilisée pour la reconnaissance de formes dans les jeux, les robots et d'autres domaines où les décisions doivent être prises en temps réel.
- **Traitement du langage naturel :** Le traitement du langage naturel est un sous-domaine de la reconnaissance de formes qui vise à comprendre et à interpréter le langage humain. Cela peut inclure la reconnaissance de la parole, la traduction automatique, la reconnaissance d'entités nommées et l'analyse des sentiments.
- **Apprentissage non supervisé :** L'apprentissage non supervisé est une méthode d'apprentissage automatique qui permet à un algorithme de découvrir des modèles dans des données sans supervision préalable. Cette méthode est souvent utilisée pour la reconnaissance de formes dans des domaines tels que la biologie, la finance et la recherche scientifique.

En fait, la reconnaissance de formes est un domaine de recherche très actif et en constante évolution. Les avancées récentes dans les réseaux de neurones profonds, les méthodes basées sur les caractéristiques, l'apprentissage par renforcement, le traitement du langage naturel et l'apprentissage non supervisé ont permis de réaliser des progrès significatifs dans la reconnaissance de formes.

1.2.Reconnaissance d'une forme dans une image

La reconnaissance de forme dans une image est un processus complexe qui vise à identifier des formes spécifiques dans une image. Le but est de trouver des modèles dans l'image qui correspondent à une forme ou un objet spécifique, même si ces formes sont légèrement modifiées ou altérées.

La reconnaissance de forme dans une image est utilisée dans de nombreuses applications, telles que la reconnaissance de caractères pour la numérisation de documents, la détection de visages dans les images, la reconnaissance de plaques d'immatriculation sur les voitures, la détection de défauts dans les produits manufacturés, etc. Les algorithmes de reconnaissance de forme continuent d'évoluer et de s'améliorer avec les avancées technologiques, offrant ainsi des possibilités toujours plus grandes dans de nombreux domaines.

La reconnaissance d'une forme dans une image peut être réalisée à l'aide de différentes techniques. Voici quelques-unes des approches les plus courantes :

1. **Les méthodes basées sur les contours :** ces méthodes utilisent les contours de la forme recherchée pour la détecter dans l'image. Elles peuvent utiliser des opérateurs de détection de contours, tels que le filtre de Canny, pour extraire les contours de la forme.
2. **Les méthodes basées sur la classification :** ces méthodes utilisent des algorithmes de classification pour identifier la forme recherchée. Elles peuvent utiliser des techniques de reconnaissance de motifs, telles que les réseaux de neurones, pour classifier les régions d'intérêt de l'image.
3. **Les méthodes basées sur les caractéristiques :** ces méthodes extraient des caractéristiques des régions d'intérêt de l'image, telles que la forme, la taille, la texture et la couleur. Ces caractéristiques sont ensuite utilisées pour identifier la forme recherchée.
4. **Les méthodes basées sur les modèles :** ces méthodes utilisent des modèles pour représenter la forme recherchée. Elles peuvent utiliser des modèles statistiques, tels que les modèles de mélange gaussien, ou des modèles symboliques, tels que les expressions régulières, pour identifier la forme.

En général, la reconnaissance de forme dans une image est un problème difficile car les images peuvent contenir des variations d'échelle, de rotation, de translation et d'illumination. Il est souvent nécessaire de combiner plusieurs techniques pour obtenir une reconnaissance de forme précise et robuste.

1.3. Etapes de reconnaissance de forme

Le processus de reconnaissance de forme dans une image peut être divisé en plusieurs étapes. Voici les principales étapes:

1. **Acquisition de l'image :** Cette étape consiste à capturer une image de la forme à reconnaître, que ce soit à partir d'un appareil photo, d'un scanner ou d'autres dispositifs d'acquisition d'image.
2. **Prétraitement :** L'étape de prétraitement consiste à préparer l'image pour la reconnaissance, en éliminant les bruits, en améliorant la qualité de l'image et en augmentant le contraste.

3. **Segmentation** : La segmentation est l'étape qui consiste à identifier les régions d'intérêt dans l'image, c'est-à-dire les parties de l'image qui contiennent la forme à reconnaître.
4. **Extraction de caractéristiques** : Une fois que les régions d'intérêt ont été identifiées, l'étape suivante consiste à extraire les caractéristiques pertinentes de la forme, telles que la taille, la forme, la texture, la couleur, etc.
5. **Classification :** La classification est l'étape qui consiste à attribuer une classe ou une étiquette à la forme à reconnaître, en comparant les caractéristiques extraites avec celles des formes connues dans une base de données ou un modèle de référence.
6. **Évaluation:** Cette étape consiste à évaluer les performances du système de reconnaissance de forme, en mesurant la précision, la sensibilité, la spécificité et d'autres métriques de performance.

Chacune de ces étapes peut être plus ou moins complexe en fonction de la forme à reconnaître et de l'application visée.

2. Modélisation par expression régulière

2.1.Qu'est ce qu'une expression régulière

Une expression régulière est une chaîne de caractères qui définit un modèle correspondant à un format spécifique selon certaines règles. Elle est largement utilisée pour décrire des motifs tels que des dates, des numéros de facture ou des numéros de carte de crédit. En d'autres termes, elle permet de déterminer un format précis que doivent respecter les résultats attendus.

En effet, une expression régulière, également connue sous le nom de RegEx, est une séquence de caractères spéciale qui utilise un modèle de recherche pour identifier une ou plusieurs chaînes de caractères. Elle permet de détecter si un texte est présent ou non en le comparant à un modèle spécifique, et peut également diviser un modèle en plusieurs sous-modèles. En d'autres termes, elle offre une méthode puissante pour la recherche de motifs dans du texte.

Les expressions régulières peuvent être utilisées dans de nombreux cas comme par exemple ;

- Prétraitement de texte,
- Extraction de texte sur la base d'un modèle, recherche et remplacement de certains mots qui correspondent à un modèle, par exemple, recherche de tous les mots commençant par "e" et les remplacer par le mot "enfant",

- Correspondance de mots de passe,
- Validation des données.

2.2.Expressions régulières et automates finis

Les expressions régulières sont utilisées pour définir et manipuler des ensembles de séquences de caractères qui peuvent inclure des séquences spécifiques de caractères, des répétitions de séquences de caractères, ou des choix entre différentes séquences de caractères. En d'autres termes, elles permettent de décrire des motifs ou des schémas de caractères.

Une représentation alternative d'une expression régulière est un graphe connu sous le nom d'« automate fini ». C'est-à-dire une expression régulière peut être considérée comme un automate fini.

Un automate fini est une machine qui accepte ou rejette des chaînes de caractères en suivant un ensemble de règles prédéfinies. On peut considérer un automate comme une machine qui possède des états et des transitions qui permettent de passer d'un état à un autre en lisant des caractères. Il est caractérisé par un état initial (représenté par une petite flèche entrante) et des états finaux (représentés par des cercles doubles). Le comportement de l'automate est déterminé par l'ensemble des séquences de caractères qui le font passer de l'état initial à un état final.

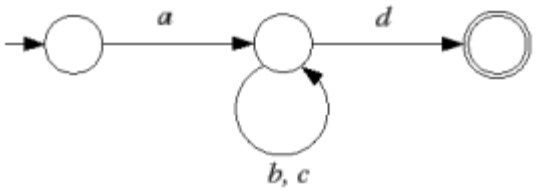


Figure 1. 1- *Exemple automate fini*

Les expressions régulières, quant à elles, sont des chaînes de caractères qui décrivent des motifs ou des règles pour les chaînes de caractères. Les expressions régulières peuvent être utilisées pour décrire les langages réguliers, qui peuvent être reconnus par des automates finis.

En d'autres termes, une expression régulière peut être considérée comme une description textuelle d'un automate fini. Lorsqu'une chaîne de caractères est soumise à une expression

régulière, elle est comparée au motif décrit par l'expression régulière. Si elle correspond au motif, elle est acceptée, sinon elle est rejetée.

Voici un exemple d'expression régulière qui peut être considérée comme un automate fini :

L'expression régulière (0|1)* décrit un langage régulier qui contient toutes les chaînes de caractères composées uniquement de 0 et de 1, y compris la chaîne vide. Cette expression régulière peut être considérée comme un automate fini déterministe à deux états.

Le premier état est l'état initial, qui accepte la chaîne vide et les chaînes qui commencent par 0 ou par 1. Le deuxième état est un état final qui accepte toutes les chaînes qui contiennent uniquement des 0 et des 1. Les transitions entre les états sont déterminées par les caractères de la chaîne.

Par exemple, la chaîne 010101 serait acceptée par cet automate fini, car elle contient uniquement des 0 et des 1. En revanche, la chaîne 23 serait rejetée, car elle ne contient pas de caractère 0 ou 1.

En résumé, l'expression régulière (0|1)* peut être considérée comme un automate fini déterministe qui reconnaît toutes les chaînes de caractères composées uniquement de 0 et de 1, y compris la chaîne vide.

Il est important de noter que toutes les expressions régulières ne sont pas équivalentes à des automates finis. Certaines expressions régulières peuvent décrire des langages qui ne peuvent pas être reconnus par des automates finis déterministes. Dans ces cas-là, il est nécessaire d'utiliser des automates finis non déterministes ou des expressions régulières plus complexes pour décrire le langage.

2.3.Modélisation par expression régulière

Les expressions régulières sont des outils de modélisation puissants pour représenter des motifs et des formes de texte. Voici quelques exemples de modélisation de formes par expression régulière :

- **Modélisation de nombres** : L'expression régulière \d+ correspondra à une ou plusieurs occurrences de chiffres. Par exemple, l'expression régulière \d{3}-\d{2}-

\d{4} correspondra à un numéro de sécurité sociale américain au format XXX-XX-XXXX.

- **Modélisation de chaînes de caractères** : L'expression régulière ^[A-Z][a-z]+ correspondra à une chaîne de caractères qui commence par une lettre majuscule suivie d'une ou plusieurs lettres minuscules. Par exemple, cette expression régulière correspondra à des noms propres tels que John, Mary, ou Robert.
- **Modélisation de motifs complexes** : L'expression régulière ([A-Za-z0-9._%+-]+)@([A-Za-z0-9.-]+.[A-Za-z]{2,}) correspondra à une adresse e-mail valide. Cette expression régulière est complexe, mais elle peut être utilisée pour valider les adresses e-mail saisies par l'utilisateur.
- **Modélisation d'un texte structuré :** L'expression régulière (<h[1-6]>.*?</h[1-6]>)+ peut être utilisée pour modéliser une structure HTML avec des balises de titre. Cette expression régulière correspond à une chaîne de caractères qui commence par une balise de titre HTML (h1 à h6), suivie de tout contenu textuel, puis se termine par la même balise de titre. La présence du + indique que cette séquence peut se répéter plusieurs fois dans le texte.

En somme, les expressions régulières peuvent être utilisées pour représenter une grande variété de formes, de motifs et de structures textuels en fonction de règles précises.

3. Reconnaisse des formes basée sur les expressions régulières

La reconnaissance de formes basée sur les expressions régulières est une technique courante utilisée dans de nombreux domaines, tels que la reconnaissance optique de caractères (OCR), la détection de motifs dans des images, la recherche de chaînes de caractères dans des textes, etc..

Elle consiste à utiliser des motifs définis par des expressions régulières pour identifier des formes spécifiques dans des données. Cette approche est particulièrement utile pour les formes régulières et répétitives, telles que les cercles, les carrés ou les triangles.

En fait, la recherche d'une forme dans une image en se basant sur les expressions régulières est une tâche complexe. Tout d'abord, il est important de segmenter l'image pour isoler les régions d'intérêt qui peuvent contenir la forme recherchée. Ensuite, un prétraitement peut être appliqué pour améliorer la qualité de l'image, tels que la réduction de bruit et l'amélioration du contraste.

Une fois que les régions d'intérêt sont identifiées, les expressions régulières peuvent être utilisées pour identifier la forme recherchée. Pour cela, une expression régulière doit être créée pour représenter la forme, en prenant en compte les caractéristiques de la forme telles que la taille, la forme et la couleur. Cette expression régulière peut ensuite être utilisée pour chercher la forme dans l'image, en utilisant des techniques de reconnaissance de texte ou de recherche de motifs.

C'est-à-dire, le processus de reconnaissance de formes basé sur les expressions régulières peut être divisé en plusieurs étapes :

1. Définir les expressions régulières correspondant aux formes à reconnaître.
2. Prétraiter les données en extrayant les informations pertinentes, telles que les coordonnées des points dans une image.
3. Appliquer les expressions régulières aux données pour identifier les formes correspondantes.
4. Valider les formes détectées en utilisant des techniques telles que la vérification de la cohérence des données.

Cependant, il est important de noter que la reconnaissance de formes basée sur les expressions régulières peut être limitée par la complexité de la forme recherchée et les variations possibles de la forme dans l'image. Dans certains cas, d'autres techniques de reconnaissance de formes telles que la reconnaissance de formes basée sur les contours ou la reconnaissance de formes basée sur les caractéristiques peuvent être plus efficaces.

4. Les algorithmes basés sur les caractéristiques

Les algorithmes basés sur les caractéristiques, également appelés méthodes d'extraction de caractéristiques, sont des techniques de reconnaissance de formes qui extraient des caractéristiques distinctives et significatives des objets à reconnaître. Ces caractéristiques peuvent être géométriques, statistiques ou fréquentielles et peuvent être utilisées pour former un modèle de l'objet. Les algorithmes basés sur les caractéristiques comparent ensuite ces modèles avec les caractéristiques des objets inconnus pour déterminer leur classe ou leur identité. Ces algorithmes sont souvent utilisés pour reconnaître des objets à partir d'images numériques ou de signaux, tels que des sons ou des vidéos.

4.1.Les caractéristiques de forme

Les caractéristiques de forme sont des propriétés mathématiques ou géométriques qui décrivent une forme ou un objet. Elles sont utilisées dans les algorithmes de reconnaissance de formes pour extraire les informations importantes d'une image ou d'un objet. Ces caractéristiques peuvent être de différents types, tels que des propriétés de contour, de texture, de couleur ou de géométrie. Les caractéristiques de forme sont souvent utilisées pour la reconnaissance d'objets dans des images, la classification de formes et la détection de défauts.

4.2.Les caractéristiques de texture

Les caractéristiques de texture sont des propriétés locales qui décrivent la variation de la couleur ou de la luminosité dans une région de l'image. Ces caractéristiques sont utilisées pour la reconnaissance de textures, qui consiste à identifier des motifs répétitifs ou des structures régulières dans une image. Les caractéristiques de texture peuvent être extraites à partir de différentes méthodes, telles que la transformée de Fourier, la transformée en ondelettes, ou encore des méthodes basées sur des modèles statistiques tels que les matrices de cooccurrence de niveaux de gris.

4.3.Les caractéristiques de couleur

Les caractéristiques de couleur sont des propriétés qui décrivent la distribution des couleurs dans une image. Ces caractéristiques sont utilisées pour la reconnaissance de couleurs, qui consiste à identifier les différentes couleurs présentes dans une image et leur proportion. Les caractéristiques de couleur peuvent être extraites à partir de différentes méthodes, telles que l'histogramme de couleur, qui compte le nombre de pixels ayant une certaine valeur de couleur, ou encore des méthodes basées sur des modèles de couleur, tels que l'espace de couleur RGB ou l'espace de couleur HSV (Teinte, Saturation, Valeur). Les caractéristiques de couleur sont utilisées dans de nombreuses applications, telles que la classification d'images, la recherche d'images et la détection d'objets.

5. Les algorithmes basés sur les modèles

Les algorithmes de reconnaissance de forme basés sur les modèles sont des méthodes qui comparent une image d'entrée avec des modèles préalablement enregistrés. Ces modèles peuvent être des images complètes ou des fragments d'images, et peuvent être créés

manuellement ou automatiquement. Les algorithmes basés sur les modèles sont souvent utilisés dans des applications où il est nécessaire de détecter des objets précis dans une image, comme la reconnaissance de visages ou la reconnaissance de plaques d'immatriculation.

Les algorithmes basés sur les modèles comparent l'image d'entrée avec les modèles en utilisant une mesure de similarité, telle que la corrélation croisée ou la distance euclidienne. Si la similarité est supérieure à un seuil de tolérance, l'algorithme reconnaît le modèle dans l'image. Les algorithmes basés sur les modèles sont souvent rapides et précis, mais leur performance peut être limitée par la qualité et la variabilité des modèles utilisés.

Parmi les méthodes de reconnaissance de forme basées sur les modèles, on peut citer par exemple les réseaux de neurones convolutifs (CNN), les machines à vecteurs de support (SVM). Ces méthodes peuvent être utilisées seules ou combinées pour améliorer la performance de la reconnaissance de forme.

Ces méthodes ont des avantages et des inconvénients en termes de précision, de temps de calcul et de complexité de mise en œuvre. Le choix de la méthode dépend de la tâche de reconnaissance de forme à effectuer, ainsi que des contraintes en termes de temps de calcul et de ressources informatiques disponibles.

5.1.Les modèles statistiques

Les modèles statistiques de reconnaissance de formes sont des algorithmes qui utilisent des modèles mathématiques pour décrire les caractéristiques des formes dans une image. Ces modèles sont souvent basés sur des distributions statistiques, telles que les distributions de probabilité, les modèles de Markov cachés ou encore les réseaux bayésiens. Les modèles statistiques permettent de représenter des formes complexes en utilisant un ensemble de paramètres qui peuvent être estimés à partir d'un ensemble d'images d'apprentissage. Les caractéristiques de forme, de texture et de couleur peuvent être utilisées comme entrée pour les modèles statistiques, qui peuvent ensuite être utilisés pour reconnaître des formes similaires dans de nouvelles images. Les modèles statistiques sont largement utilisés dans la reconnaissance de caractères manuscrits, la reconnaissance de visages, la reconnaissance de voix et dans d'autres applications de reconnaissance de formes.

5.2.Les modèles de réseaux de neurones

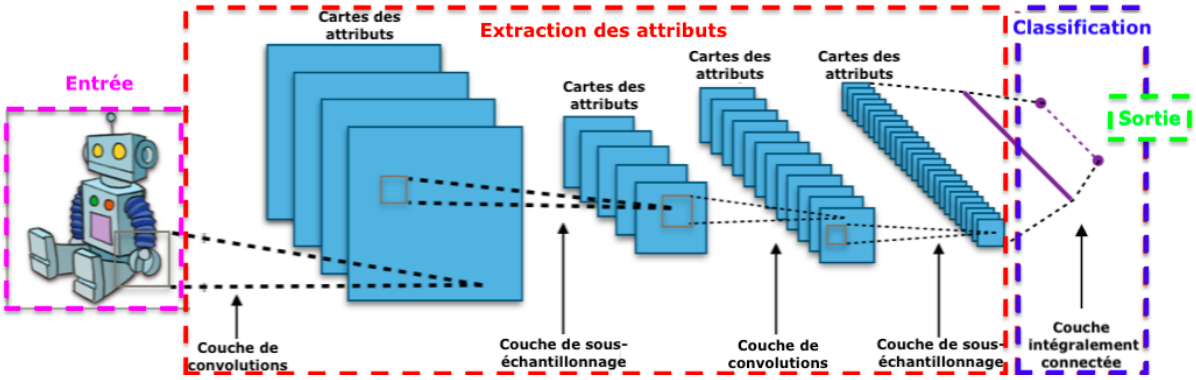


Figure 1. 2- *Exemples de modèles de CNN : Architecture standard d'un réseau à convolutions [23]*

5.3.Les modèles basés sur les arbres de décision

Les méthodes de reconnaissance de forme basées sur les arbres de décision sont des techniques d'apprentissage supervisé qui utilisent un arbre de décision pour classer les données en fonction de leurs caractéristiques. Les arbres de décision sont des structures de données en forme d'arbre, où chaque nœud représente une caractéristique ou une variable d'entrée, chaque branche représente une valeur possible pour cette caractéristique, et chaque feuille représente une classe de sortie.

Pour construire un arbre de décision, l'algorithme analyse les données d'entrée pour déterminer quelle caractéristique est la plus importante pour la classification, puis divise les données en sous-ensembles en fonction de cette caractéristique. Il répète ce processus pour chaque sous-ensemble jusqu'à ce que chaque feuille représente une classe de sortie distincte.

Une fois que l'arbre de décision est construit, il peut être utilisé pour classer de nouvelles données en suivant les branches de l'arbre en fonction de leurs caractéristiques. L'arbre attribue une classe de sortie à la feuille correspondante.

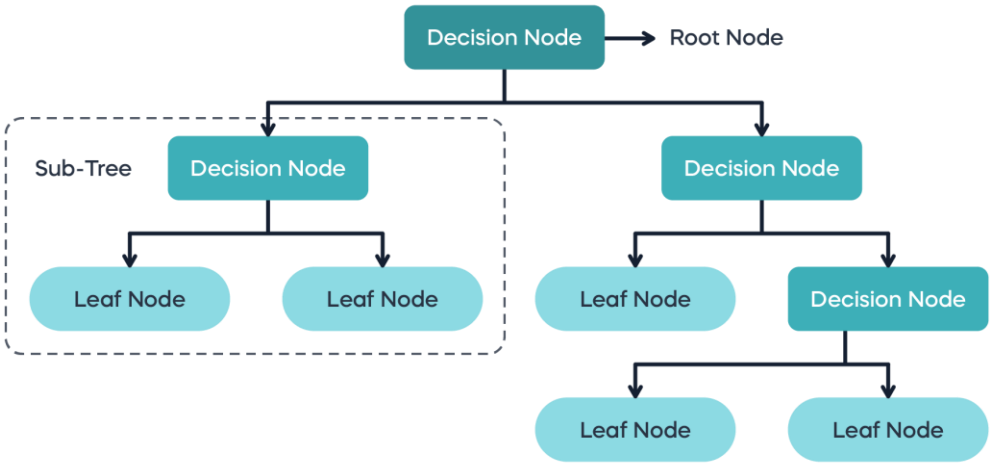


***Figure 1. 3**- Structure d'un arbre de décision [24]*

Les méthodes de reconnaissance de forme basées sur les arbres de décision sont utilisées dans de nombreux domaines, notamment la classification d'images, la reconnaissance de caractères, la reconnaissance de la parole, et la détection de fraudes.

5.4.Les modèles basés sur les graphes

Les modèles de reconnaissance de forme basés sur les graphes utilisent des graphes pour représenter les relations spatiales et structurelles entre les éléments d'une image. Les graphes peuvent être utilisés pour représenter les relations entre les pixels, les régions ou les objets dans une image. Les nœuds du graphe représentent les éléments d'intérêt, tels que les coins, les bords ou les régions, et les arcs du graphe représentent les relations entre ces éléments, telles que la connectivité ou la proximité spatiale. Les caractéristiques sont ensuite extraites à partir de la structure du graphe pour la reconnaissance de forme. Les modèles de reconnaissance de forme basés sur les graphes sont souvent utilisés dans des domaines tels que la vision par ordinateur, la reconnaissance de caractères manuscrits et la reconnaissance de formes géométriques.

« *Graph Matching* » est un exemple de modèle de reconnaissance de forme basé sur les graphes. Ce modèle consiste à représenter les formes comme des graphes, où les sommets représentent des points caractéristiques de la forme, et les arêtes représentent les relations spatiales entre ces points. [25]

Le but de la reconnaissance de forme est de trouver la correspondance entre deux graphes, c'est-à-dire de trouver quels sommets et arêtes du premier graphe correspondent à quels sommets et arêtes du second graphe. Cette correspondance permet de comparer les deux formes et de déterminer s'il y a une similarité entre elles.

6. Les approches hybrides

6.1.Principe de l'approche hybride

L'approche hybride en reconnaissance de formes consiste à combiner plusieurs techniques de reconnaissance de formes pour obtenir des résultats plus précis et fiables. Cette méthode est basée sur le principe que chaque technique a ses avantages et ses limites, et que la combinaison de plusieurs techniques peut compenser les faiblesses de chacune.

L'approche hybride peut être réalisée en utilisant différents algorithmes de fusion, tels que la fusion de décision, la fusion de score et la fusion de données. La fusion de décision consiste à combiner les décisions de plusieurs classificateurs pour prendre une décision finale. La fusion de score consiste à combiner les scores de plusieurs classificateurs pour obtenir un score global pour chaque classe. La fusion de données consiste à combiner les données d'entrée de plusieurs classificateurs pour obtenir une représentation plus complète de la forme.

L'approche hybride peut être utilisée dans de nombreux domaines, tels que la reconnaissance de caractères manuscrits, la reconnaissance faciale, la reconnaissance d'objets, la reconnaissance de gestes, etc.

6.2.Combinaison d'approches basées sur les caractéristiques et les modèles

La combinaison d'approches basées sur les caractéristiques et les modèles consiste à utiliser à la fois des caractéristiques extraites de l'image et des modèles de reconnaissance de forme pour améliorer la performance de la reconnaissance de forme. Cette approche hybride permet de tirer parti des avantages des deux approches, en utilisant les caractéristiques pour identifier les régions d'intérêt dans l'image, puis en utilisant les modèles pour classer ces régions en fonction de leur forme.

Cette approche peut être réalisée de différentes manières, par exemple :

- Extraction des caractéristiques à partir des modèles : Cette méthode consiste à extraire des caractéristiques à partir de modèles de forme, puis à les utiliser pour la reconnaissance de forme.
- Utilisation de caractéristiques et de modèles en parallèle : Cette méthode consiste à utiliser des caractéristiques et des modèles en parallèle pour la reconnaissance de forme. Les caractéristiques et les modèles peuvent être combinés à différents niveaux de traitement pour améliorer la performance.
- Utilisation de caractéristiques pour l'initialisation de modèles : Cette méthode consiste à utiliser des caractéristiques pour l'initialisation de modèles de forme, puis à affiner ces modèles en utilisant des données d'apprentissage.

En effet, une approche hybride pour la reconnaissance de chiffres manuscrits pourrait utiliser des caractéristiques telles que le nombre de boucles et la courbure des traits pour identifier les chiffres dans l'image, puis utiliser des modèles de réseaux de neurones pour classifier chaque chiffre en fonction de sa forme. Cette combinaison d'approches permettrait de réduire les erreurs de classification et d'améliorer la précision globale de la reconnaissance de forme.

6.3.Avantages et inconvénients des approches hybrides

Les approches hybrides ont plusieurs avantages, tels que :

- **Meilleure précision** : en combinant plusieurs méthodes, on peut améliorer la précision de la reconnaissance des formes.
- **Flexibilité** : les approches hybrides sont flexibles et peuvent être adaptées à différentes tâches de reconnaissance de formes.
- **Robustesse** : les approches hybrides sont plus robustes aux variations de données et de bruit, car elles intègrent plusieurs méthodes de traitement de l'information.

Cependant, les approches hybrides ont également des inconvénients, tels que :

- **Complexité :** la combinaison de plusieurs méthodes peut rendre l'approche plus complexe et difficile à mettre en œuvre.
- **Coût** : l'utilisation de plusieurs méthodes peut augmenter le coût de traitement des données et de calcul.
- La combinaison de plusieurs méthodes peut rendre l'interprétation des résultats plus difficile.

7. Evaluation des algorithmes de reconnaissance de formes

7.1.Les métriques d'évaluation

Les métriques d'évaluation sont des critères utilisés pour mesurer la performance des algorithmes de reconnaissance de formes. Elles permettent de quantifier la précision de la méthode utilisée pour reconnaître les formes et de comparer différentes approches.

Ces métriques peuvent être utilisées pour évaluer différents algorithmes de reconnaissance de formes et pour choisir la méthode la plus appropriée pour une application spécifique.

Les métriques les plus utilisées sont :

- **L'accuracy :** c'est la proportion de prédictions correctes par rapport au nombre total de prédictions.
- **Le rappel (recall) :** c'est la proportion de vrais positifs (objets correctement identifiés) par rapport au nombre total de vrais positifs et de faux négatifs (objets qui auraient dû être identifiés mais qui ne l'ont pas été).
- **La précision (precision) :** c'est la proportion de vrais positifs par rapport au nombre total de vrais positifs et de faux positifs (objets incorrectement identifiés).
- La mesure F1 : c'est la moyenne harmonique de la précision et du rappel. Elle permet de prendre en compte à la fois la précision et le rappel.
- **La courbe ROC (Receiver Operating Characteristic) :** elle permet de représenter la performance d'un algorithme de reconnaissance de forme en fonction du taux de faux positifs et du taux de vrais positifs.
- **La matrice de confusion :** elle permet de visualiser les performances d'un algorithme de reconnaissance de forme en comparant les prédictions avec les vraies étiquettes. Elle contient les vrais positifs, les faux positifs, les vrais négatifs et les faux négatifs.

7.2.Les techniques d'amélioration des performances

Il existe plusieurs techniques pour améliorer les performances des algorithmes de reconnaissance de formes. Chaque technique peut avoir ses avantages et ses inconvénients en fonction du type de données et de la tâche de reconnaissance de formes à effectuer. Il est donc important de sélectionner la technique la plus appropriée pour chaque situation.

Parmi ces techniques, ont peut citer ;

- **Prétraitement des données :** cette technique consiste à effectuer une série de transformations sur les données d'entrée avant de les utiliser dans l'algorithme de reconnaissance de formes. Les techniques de prétraitement peuvent inclure le filtrage, le redimensionnement, la normalisation, etc.
- **Sélection de caractéristiques :** cette technique consiste à sélectionner les caractéristiques les plus pertinentes pour la tâche de reconnaissance de formes. Cela permet de réduire la dimensionnalité des données et d'améliorer les performances de l'algorithme.
- **Utilisation des large Dataset:** l'utilisation d'un ensemble de données plus large et plus diversifié peut aider à améliorer la capacité de généralisation de l'algorithme.
- **…**

Conclusion

En conclusion, les algorithmes de reconnaissance de formes sont utilisés pour résoudre des problèmes tels que la classification, la segmentation et la détection d'objets dans les images. Il existe plusieurs approches pour la reconnaissance de formes, notamment celles basées sur les caractéristiques, les modèles, les graphes, ainsi que les approches hybrides qui combinent les deux. Chaque approche a ses avantages et ses inconvénients, et la sélection d'une méthode dépend des exigences de l'application et des caractéristiques des données. Les techniques d'évaluation des algorithmes de reconnaissance de formes sont importantes pour évaluer et comparer les performances des différentes méthodes. Enfin, la reconnaissance de formes a des applications pratiques dans de nombreux domaines, tels que la médecine, l'industrie, la surveillance, l'agriculture, la robotique, et bien d'autres encore.

Dans le chapitre suivant nous allons étudier l'algorithme de Freeman comme une méthode de reconnaissance de forme.

Chapitre 2 :

Algorithmes de Freeman pour la reconnaissance de formes

Chapitre2

Algorithmes de Freeman pour la reconnaissance de formes

Introduction

L'approche de Freeman est une méthode de reconnaissance de formes basée sur l'analyse de contours. Cette méthode consiste à décrire les contours d'une forme en utilisant une séquence de codes appelés codes de Freeman. Cette approche a été largement utilisée pour la reconnaissance de formes dans des domaines tels que la reconnaissance de caractères manuscrits, la reconnaissance de formes biologiques et la reconnaissance de formes industrielles.

Dans ce chapitre, nous allons explorer les principes de l'approche de Freeman et comment on peut l'utiliser pour la reconnaissance de formes.

1. Modèle de Freeman

1.1.Présentation du modèle de Freeman

L'algorithme de Freeman est utilisé pour la reconnaissance de formes à partir de contours d'images. Il se base sur le modèle de chaînes de Freeman pour représenter les contours sous forme de séquences de codes directionnels.

L'algorithme consiste en trois étapes principales :

1. **Prétraitement :** L'image est prétraitée pour obtenir un contour continu. Cela peut inclure des opérations telles que la binarisation, la détection de contours et le lissage.
2. **Extraction des codes directionnels :** Le contour continu est converti en une séquence de codes directionnels de Freeman. Chaque code directionnel correspond à une direction parmi les 8 directions de l'espace de Freeman (Nord, Nord-Est, Est, Sud-Est, Sud, Sud-Ouest, Ouest, Nord-Ouest).
3. **Comparaison :** Les séquences de codes directionnels sont comparées pour évaluer la similarité entre les contours. Différentes mesures de similarité peuvent être utilisées, telles que la distance de Hamming ou la distance de Hausdorff.

L'algorithme de Freeman peut être utilisé pour diverses tâches de reconnaissance de formes, telles que la reconnaissance de caractères manuscrits, la reconnaissance de formes géométriques, la reconnaissance d'objets, …

1.2.Description des directions de Freeman

Les directions de Freeman sont un ensemble de 8 directions discrètes utilisées pour décrire les contours d'une image. Chaque direction est représentée par un nombre entier allant de 0 à 7, correspondant à l'un des 8 mouvements possibles d'un pixel à son voisin dans le sens des aiguilles d'une montre ou dans le sens inverse.

Les directions de Freeman sont définies comme suit :

- Direction 0 : droite
- Direction 1: diagonale haut-droite
- Direction 2 : haut
- Direction 3 : diagonale haut-gauche
- Direction 4 : gauche
- Direction 5 : diagonale bas-gauche
- Direction 6 : bas
- Direction 7 : diagonale bas-droite

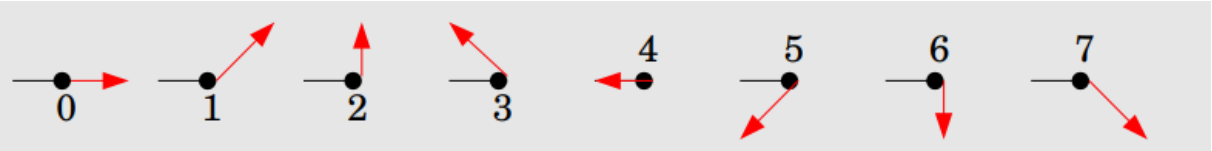


***Figure 2. 1**-Directions de Freeman*

Ces directions permettent de décrire de manière concise et précise les contours d'une image, en évitant les ambiguïtés liées à l'utilisation de courbes continues ou de segments de droite. Le modèle de Freeman utilise ces directions pour représenter les contours des objets dans une image et pour en extraire des caractéristiques géométriques.

2. Extraction des caractéristiques

2.1.Extraction des contours

L'extraction des contours est la première étape de l'algorithme de Freeman. Cette étape consiste à transformer l'image en une suite de codes de contour de Freeman qui décrivent la forme de l'objet en question. Les codes de contour de Freeman sont des séquences ordonnées de directions de mouvement décrivant la façon dont le contour de l'objet se déplace à travers l'image.

Le processus d'extraction des contours de Freeman commence par la recherche du point de départ du contour. Cela peut être effectué en parcourant la première ligne de l'image de gauche à droite jusqu'à ce que le premier point du contour soit trouvé.

L'extraction des contours se fait en parcourant l'image dans le sens des aiguilles d'une montre, en utilisant les directions de Freeman pour suivre les bords des objets. L'algorithme commence à partir d'un point de départ donné, puis suit les bords en utilisant les directions de Freeman pour passer d'un pixel à un autre. Chaque changement de direction est enregistré et stocké sous forme de séquence de directions de Freeman. Cette séquence de directions de Freeman représente le contour de l'objet dans l'image.

2.2.Codage des contours selon le modèle de Freeman

Une fois que les contours Freeman ont été extraits, différentes caractéristiques peuvent être calculées à partir de ces séquences de directions. Les caractéristiques les plus couramment utilisées dans l'algorithme de Freeman sont les suivantes :

- Le nombre total de contours Freeman : qui donne une mesure de la complexité de l'objet.
- La longueur du contour Freeman : qui donne une mesure de la taille de l'objet.
- Les angles de chaque direction de Freeman : qui donnent une mesure de la forme de l'objet.
- Les transitions de direction de Freeman : qui donnent une mesure de la régularité du contour.

À chaque étape, on examine les pixels voisins pour déterminer la direction de déplacement suivante, qui est codée par un chiffre de 0 à 7, représentant les huit directions possibles.

Chaque chiffre représente une direction de déplacement, et il est attribué à chaque pixel du contour en fonction de la direction dans laquelle il se déplace par rapport au pixel précédent. Le résultat est une séquence de chiffres de 0 à 7, représentant le chemin suivi par le contour.

3. Classification des formes

3.1.Classification basée sur la distance de Hausdorff

La classification basée sur la distance de Hausdorff est une autre approche possible pour la reconnaissance de formes à partir du modèle de Freeman. Cette méthode est basée sur la comparaison des contours de deux objets en utilisant la distance de Hausdorff.

La distance de Hausdorff est une mesure de la similarité entre deux ensembles de points. Pour deux ensembles de points A et B, la distance de Hausdorff est définie comme la plus grande distance entre un point de A et son point le plus proche dans B, ou vice versa. Cette distance mesure donc la similarité entre les formes des deux ensembles de points.

Dans le cas de la reconnaissance de formes à partir du modèle de Freeman, on peut calculer la distance de Hausdorff entre deux contours en considérant chaque contour comme un ensemble de points. On compare alors la forme des deux contours en calculant leur distance de Hausdorff.

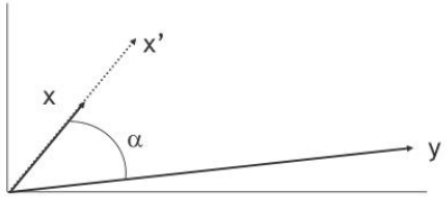
Une fois que la distance de Hausdorff a été calculée entre un contour inconnu et chaque contour de la base de données, on peut classer le contour inconnu en trouvant le contour de la base de données avec la distance de Hausdorff minimale.

La classification basée sur la distance de Hausdorff est une méthode robuste pour la reconnaissance de formes, car elle permet de prendre en compte les variations de forme entre les objets. Cependant, elle peut être plus coûteuse en termes de calcul que la classification basée sur la distance de Hamming.

3.2.Classification basée sur les angles de Freeman

La classification basée sur les angles de Freeman consiste à utiliser les angles de direction pour comparer les formes et effectuer une classification. Les angles de Freeman sont les angles correspondant à chaque direction de Freeman (0, 1, 2, 3, 4, 5, 6 et 7) dans le modèle de Freeman. Pour chaque forme, les angles de Freeman sont calculés et stockés dans un vecteur d'angles.

Ensuite, pour effectuer une classification, on calcule la distance angulaire entre les vecteurs d'angles de deux formes à l'aide de la formule [27]:



$$\text{distance angulaire} = a\cos(x, y)\ /(|x|\ |y|)$$

où x et y sont les vecteurs d'angles de deux formes à comparer, et |x| et |y| sont les normes des vecteurs.

Plus la distance angulaire entre deux formes est faible, plus elles sont similaires. On peut donc utiliser cette mesure pour classer les formes en fonction de leur similarité.

4. Reconnaissance de formes géométriques

Le modèle de Freeman est particulièrement adapté à la reconnaissance de formes géométriques, car il est basé sur la détection des contours des objets. Les formes géométriques sont généralement caractérisées par des arrangements particuliers de leurs segments ou bords. Par conséquent, l'approche de Freeman est bien adaptée pour extraire ces caractéristiques de contour et les utiliser pour reconnaître les formes géométriques.

Pour reconnaître les formes géométriques, il est d'abord nécessaire de définir les caractéristiques spécifiques qui les distinguent les unes des autres. Par exemple, un cercle peut être caractérisé par une courbe fermée avec un rayon constant, tandis qu'un carré peut être caractérisé par quatre côtés de longueur égale et quatre angles droits.

En utilisant l'approche de Freeman, ces caractéristiques géométriques peuvent être représentées sous forme de séquences de directions de Freeman, qui sont des codages compacts de la forme des contours. Une fois que ces séquences ont été extraites des images des formes géométriques, elles peuvent être comparées à des séquences de référence pour déterminer laquelle correspond le mieux à la forme en question.

Voici notre scénario pour la reconnaissance de forme géométrique en utilisant l'algorithme de Freeman :

1. Acquisition de l'image: Tout d'abord, une image contenant la forme géométrique à reconnaître doit être acquise à partir d'une caméra, d'un scanner ou d'image stockée.
2. Extraction du contour: Ensuite, le contour de la forme géométrique est extrait à l'aide de l'algorithme de Freeman. Le contour est représenté par une séquence de directions de Freeman, qui sont des vecteurs de longueur 1 indiquant la direction du prochain pixel de contour par rapport au pixel précédent.
3. Codage de la séquence: La séquence de directions de Freeman est ensuite codée pour former une signature unique de la forme géométrique. Le codage peut être effectué en utilisant un encodage de Gray pour éviter les erreurs de codage dues à des transitions abruptes de direction.

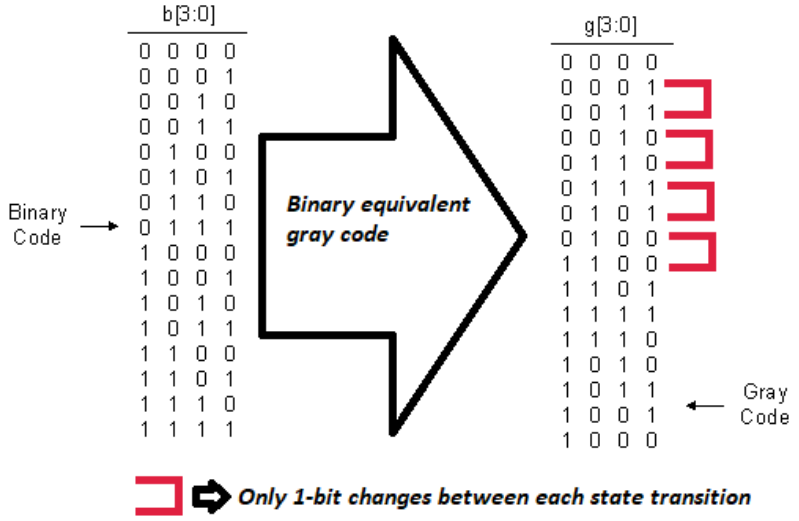


Figure 2. 2- *Codage binaire VS codage Gray [28]*

4. Stockage de la signature: La signature de la forme géométrique est stockée dans une base de données contenant les signatures de plusieurs formes géométriques connues.

5. Reconnaissance de la forme: Lorsqu'une nouvelle forme géométrique doit être reconnue, le contour est extrait et la séquence de directions de Freeman est codée en une signature. Cette signature est ensuite comparée à toutes les signatures dans la base de données à l'aide d'une mesure de distance appropriée. La forme géométrique dont la signature a la distance minimale par rapport à la signature de la forme inconnue est considérée comme la forme géométrique reconnue. Nous allons combiner la distance de Hamming et la distance de Hausdorff pour le calcul des distances et même le KNN dans une étape avancée.
6. Affichage des résultats: La forme géométrique reconnue est affichée à l'utilisateur, ainsi que la mesure de distance utilisée pour la reconnaissance.

5. Notre approche : reconnaissance de forme basé sur l'algorithme de Freeman

Pour la mise en place d'un modèle de reconnaissance de forme à base de Freeman, nous avons procéder en trois étapes :

1. Création/choix du Dataset : on est appelé à utiliser une collection de données de quelques centaines de formes.
2. Création de fichier CSV contenant les codes Freeman de l'ensemble des images dans le Dataset,
3. Entrainement et test de modèle

Notre approche est résumé dans le graphe suivante :

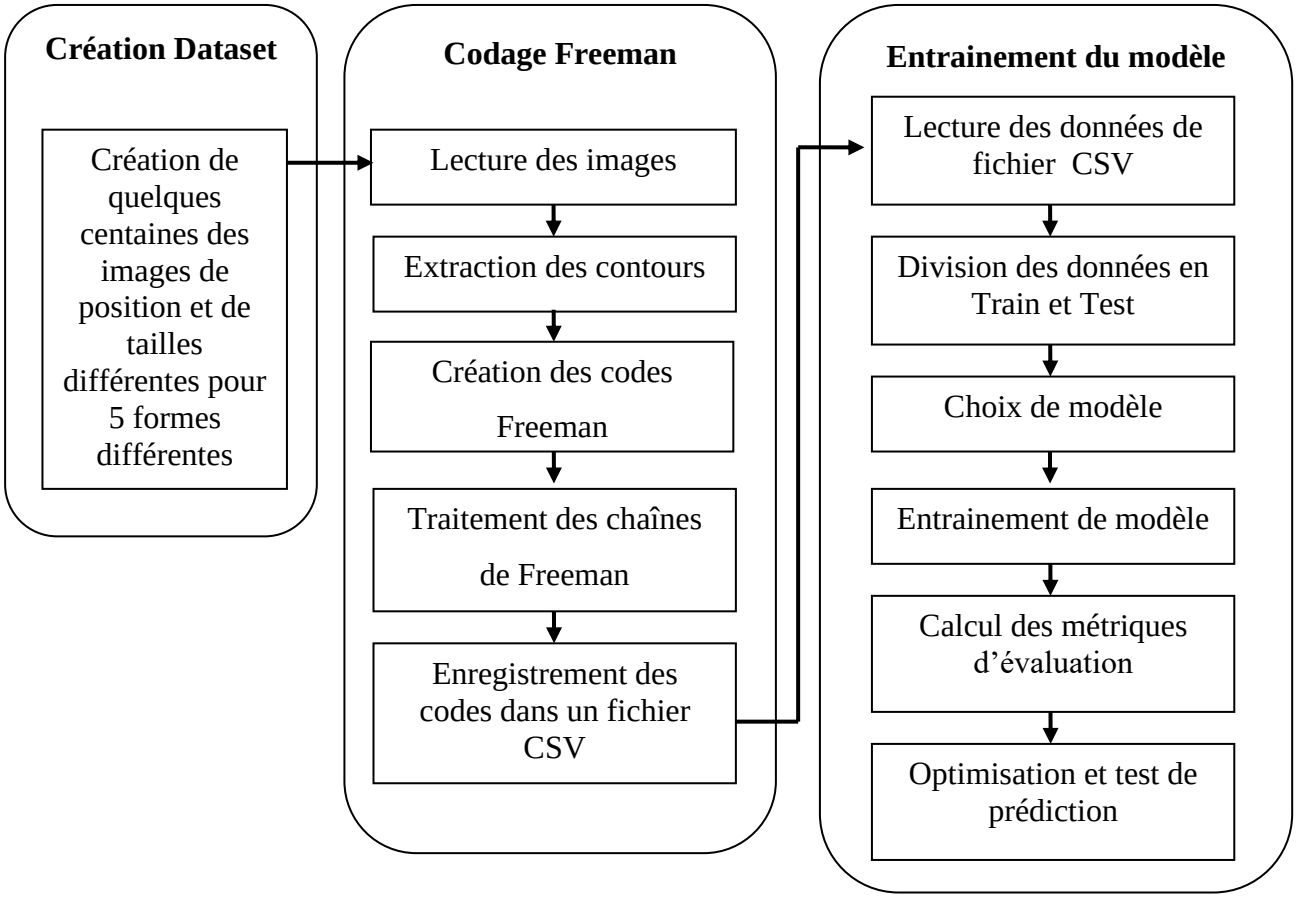


Figure 2. 3- *Organigramme de notre approche*

5.1.Prédiction de forme basée sur le KNN

Le modèle k-plus proches voisins (k-NN) est un algorithme d'apprentissage supervisé largement utilisé dans le domaine de l'apprentissage automatique. C'est une méthode simple et intuitive pour la classification et la régression.

L'idée fondamentale derrière le modèle k-NN est de classer ou prédire de nouvelles instances en se basant sur les exemples similaires dans l'ensemble de données d'entraînement. Le "k" dans k-NN représente le nombre de voisins les plus proches que l'algorithme prendra en compte pour prendre une décision.

L'idée fondamentale derrière le modèle k-NN est de classer ou prédire de nouvelles instances en se basant sur les exemples similaires dans l'ensemble de données d'entraînement. Le "k"

dans k-NN représente le nombre de voisins les plus proches que l'algorithme prendra en compte pour prendre une décision.

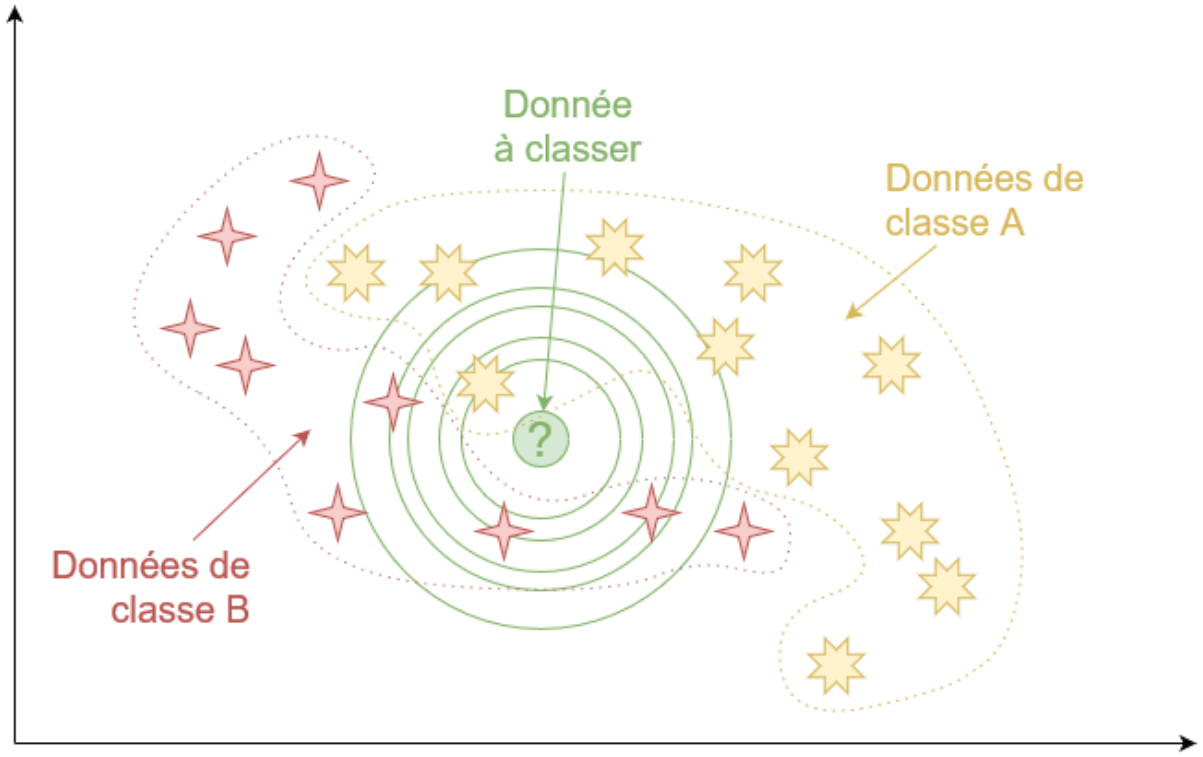


Figure 2. 4- Principe de KNN

L'intuition derrière l'algorithme des K plus proches voisins est l'une des plus simples de tous les algorithmes de Machine Learning supervisé.

Les étapes des classifications KNN sont les suivantes ;

- Étape 1 : Sélectionnez le nombre K de voisins
- Étape 2 : Calculez la distance du point non classifié aux autres points.

  La distance euclidienne: calcule la racine carrée de la somme des différences carrées entre les coordonnées de deux points.

$$D_e(x, y) = \sqrt{\sum_{j=1}^{n} (x_j - y_j)^2}$$

  Distance Manhattan: calcule la somme des valeurs absolues des différences entre les coordonnées de deux points.

$$D_m(x,y) = \sum_{i=1}^{k} \mid x_i - y_i \mid$$

Distance Hamming : la distance entre deux points donnés est la différence maximale entre leurs coordonnées sur une dimension.

$$D_h(x,y) = \sum_{i=1}^{k} \mid x_i - y_i \mid$$

Avec :

- $x = y \implies D = 0$
- $x \neq y \implies D = 1$

- Étape 3 : Prenez les K voisins les plus proches selon la distance calculée.
- Étape 4 : Parmi ces K voisins, comptez le nombre de points appartenant à chaque catégorie.
- Étape 5 : Attribuez le nouveau point à la catégorie les plus présents p a rmis ces K voisins.
- Étape 6 : Notre modèle est prêt :

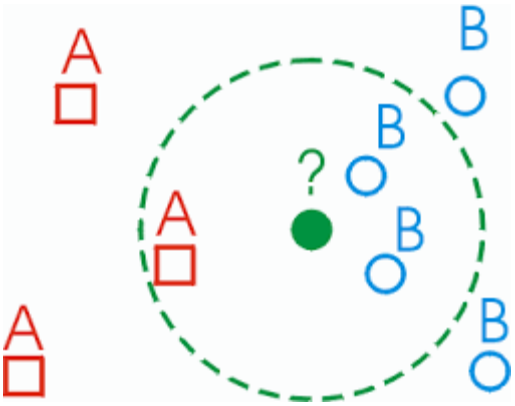


Figure 2. 5- *Indentification des proches voisins*

La prédiction de forme basée sur le modèle k-plus proches voisins (k-NN) consiste à utiliser cet algorithme pour classer une nouvelle forme en fonction de ses caractéristiques et des exemples de formes existantes dans l'ensemble de données d'entraînement.

En utilisant le modèle k-NN, il est possible de prédire la classe ou le type de formes inconnues en se basant sur les exemples existants et leurs caractéristiques. Cela peut être utile dans diverses applications, telles que la reconnaissance de caractères, la classification d'images ou la détection de motifs dans les données.

5.2.Prédiction de forme basée sur le SVM

Le l'ensemble de techniques destinées à résoudre des problèmes de discrimination (prédiction d'appartenance à des groupes prédéfinis) et de régression (analyse de la relation d'une variable par rapport à d'autres). Il s'agit des classifieurs binaires qui déterminent l'équation d'un hyperplan optimal qui sépare les données suivant leur classe.

Comme le montre la figure ci-dessous, leur principe est simple : ils ont pour but de séparer les données en classes à l'aide d'une frontière aussi, de façon que la distance entre les différents groupes de données et la frontière qui les sépare soit maximale. Cette distance est aussi appelée « marge » et les SVMs sont ainsi qualifiés de « séparateurs à vaste marge », les « vecteurs de support » étant les données les plus proches de la frontière.

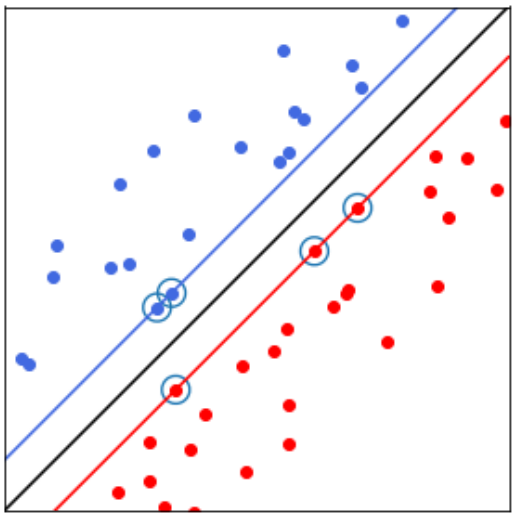


***Figure 2. 6**-Principe de SVM*

Le succès de la prédiction de forme basée sur les SVM dépend de la qualité des caractéristiques extraites, du choix du noyau et de la sélection optimale des paramètres du

modèle. Il est recommandé d'évaluer les performances du modèle en utilisant des mesures telles que la précision, le rappel et la F-mesure, ainsi que de comparer les résultats avec d'autres méthodes de classification pour déterminer l'efficacité du modèle SVM dans la prédiction de forme.

Les SVM offrent une approche puissante et largement utilisée pour la classification des formes, et elles peuvent être appliquées à divers domaines tels que la reconnaissance d'objets, la vision par ordinateur, la détection de motifs et bien d'autres applications liées à l'analyse d'images et à l'apprentissage automatique.

Conclusion

En conclusion, l'approche de Freeman pour la reconnaissance de formes géométriques est une méthode simple et efficace pour extraire les caractéristiques de contours et les utiliser pour la classification et la reconnaissance de formes. L'algorithme de Freeman est basé sur des directions de contour et permet l'extraction de caractéristiques invariantes à la rotation, à l'échelle et aux translations. Les méthodes de classification basées sur la distance et les angles de Freeman ont été largement utilisées pour la reconnaissance de formes géométriques. Les avantages de l'approche de Freeman comprennent sa simplicité, son efficacité et sa capacité à gérer des données bruyantes. Cependant, l'approche de Freeman peut également présenter des limites en termes de précision et de sensibilité aux variations de formes.

Des améliorations et des extensions ont été proposées pour cette approche, telles que l'utilisation de Gray Code pour éviter les erreurs de codage. En somme, l'approche de Freeman reste un outil précieux pour la reconnaissance de formes géométriques, en particulier pour les formes simples et régulières.

Chapitre 3 :

Implémentation

Chapitre3

Implémentation

Introduction

Dans ce chapitre nous présentons l'implémentation et l'évaluation de différents modèles pour la mise en place de code de Freeman pour la reconnaissance des formes usuelles. On va tout d'abord présenter les différentes bibliothèques utilisées. Par la suite, nous décrivons les outils utilisés et la collection de données. Enfin, on effectue une évaluation des résultats obtenus

1. Environnement de développement

1.1. Google Colab

Google Colaboratory, également connu sous le nom de "Google Colab", est un environnement de développement de machine learning basé sur le cloud et gratuit. Il offre aux utilisateurs la possibilité d'écrire et d'exécuter du code Python dans un environnement Jupyter Notebook, sans avoir besoin d'installer de logiciel sur leur ordinateur local.

Colab utilise des machines virtuelles hébergées par Google qui mettent à disposition des ressources de calcul gratuites pour des tâches de machine learning et d'apprentissage profond. De plus, les utilisateurs peuvent partager leurs notebooks Colab avec d'autres utilisateurs afin de collaborer en temps réel.

Figure 3. 1- *Logo Google Colab [31]*

1.2. Python

Python est un langage de programmation interprété, réputé pour sa facilité de lecture et d'écriture. Il est largement utilisé pour le développement d'applications dans divers domaines tels que la science des données, l'apprentissage automatique, la visualisation de données, la création de sites web, l'automatisation des tâches, …

Python est l'un des langages de programmation les plus populaires au monde en raison de sa simplicité et de sa polyvalence. Sa syntaxe concise et claire facilite la lecture et la maintenance du code. De plus, Python dispose d'une vaste collection de bibliothèques et de Frameworks qui simplifient le développement de projets.

Figure 3. 2- *Logo Google Python [32]*

2. Bibliothèques utilisées

1.3. Numpy

NumPy, qui est l'abréviation de "Numerical Python", est une bibliothèque open source en langage Python. Elle est largement utilisée dans la programmation scientifique en Python, notamment dans les domaines de la science des données, de l'ingénierie, des mathématiques, etc. NumPy est spécialement conçue pour effectuer des calculs logiques et mathématiques sur des tableaux et des matrices. Grâce à cette bibliothèque, ces opérations peuvent être réalisées de manière plus rapide et efficace par rapport aux listes Python traditionnelles.

Figure 3. 3- *Logo de Numpy [33]*

NumPy repose sur une combinaison de C et de Python, et utilise des tableaux de données multidimensionnels et homogènes appelés Ndarrays (ndimensional arrays). Les tableaux NumPy offrent plusieurs avantages par rapport aux listes Python classiques. Tout d'abord, ils occupent moins de mémoire et d'espace de stockage, ce qui constitue leur principal avantage. En effet, un tableau NumPy est généralement de taille plus petite qu'une liste Python (une liste peut atteindre 20 Mo, tandis qu'un tableau n'excède pas 4 Mo). De plus, ces tableaux sont faciles à accéder en lecture et en écriture, ce qui les rend pratiques à utiliser.

1.4. Pandas

Pandas, dont le nom est dérivé de "Panel Data", est une bibliothèque Python dédiée à la Data Science. Elle offre des fonctionnalités puissantes, flexibles et faciles à utiliser pour charger, aligner, manipuler et fusionner des données.

Figure 3. 4*- Logo Pandas[34]*

Pandas est largement utilisé pour le "Data Wrangling", c'est-à-dire les méthodes permettant de transformer les données non structurées en données exploitables. Elle est également compatible avec d'autres bibliothèques Python et excelle dans le traitement des données structurées sous forme de tableaux, de matrices ou de séries temporelles. Pandas permet d'importer et d'exporter des données dans différents formats tels que CSV ou JSON, et offre également des fonctionnalités de nettoyage des données.

Le fonctionnement de Pandas repose sur les "DataFrames", qui sont des tableaux de données en deux dimensions. Chaque colonne d'un DataFrame contient les valeurs d'une variable, et chaque ligne contient un ensemble de valeurs correspondantes pour chaque colonne. Les données stockées dans un DataFrame peuvent être des nombres ou des caractères.

1.5. Matplotlib

Matplotlib est une bibliothèque open source en Python qui permet de générer des visualisations de données. À l'origine, elle a été développée pour visualiser les signaux

électriques du cerveau de personnes épileptiques, mais au fil du temps, elle a été améliorée par de nombreux contributeurs de la communauté open source. Matplotlib est largement utilisée pour créer des graphiques et des diagrammes de haute qualité, offrant une alternative open source à MATLAB.

Figure 3. 5*- Logo matplotlib [35]*

Cette bibliothèque permet non seulement de créer des graphiques de base tels que des diagrammes à barres ou des histogrammes, mais également des figures plus complexes en trois dimensions. Grâce à ses nombreuses fonctionnalités et options de personnalisation, Matplotlib offre aux utilisateurs la possibilité de créer des visualisations variées et adaptées à leurs besoins spécifiques.

1.6. OpenCV

OpenCV (Open Source Computer Vision Library), est une bibliothèque open-source dédiée au traitement d'images et à la vision par ordinateur. Elle propose une large gamme d'algorithmes et de fonctionnalités pour le traitement et l'analyse d'images et de vidéos, ainsi que pour la reconnaissance et la détection de formes. Initialement écrite en C++, OpenCV dispose de bindings pour d'autres langages de programmation, notamment Python.

Figure 3. 6*- Logo OpenCV[36]*

OpenCV est largement utilisée dans le domaine de la recherche et de l'industrie pour diverses applications telles que la reconnaissance de caractères, la surveillance vidéo, la vision industrielle, la réalité augmentée, la robotique, et bien d'autres. Grâce à sa vaste collection d'outils et d'algorithmes, OpenCV facilite le développement d'applications de vision par ordinateur en fournissant des fonctionnalités avancées de traitement d'images et de vidéos.

1.7. Seaborn

Seaborn est une bibliothèque qui complète Matplotlib en remplaçant certains réglages par défaut et fonctions, tout en ajoutant de nouvelles fonctionnalités. Cette bibliothèque offre des performances similaires à Matplotlib, mais apporte une simplicité et des fonctionnalités uniques. Elle facilite l'exploration et la compréhension rapides des données.

Figure 3. 7- *Logo seaborn [37]*

Seaborn propose des fonctions spécifiques pour créer des graphiques utiles dans l'analyse statistique, tels que le "distplot" (graphique de distribution). De plus, elle fournit différents styles et palettes de couleurs par défaut, ce qui permet de créer des graphismes plus attrayants visuellement. Elle est donc un outil idéal pour la visualisation statistique des données.

L'un des avantages de Seaborn est sa syntaxe concise, qui simplifie la création des graphiques. De plus, elle offre des thèmes par défaut esthétiquement plaisants, ce qui permet de générer des visualisations attrayantes sans effort supplémentaire. En résumé, Seaborn est une bibliothèque puissante et pratique pour la visualisation et l'analyse statistique des données.

1.8. Scikit-learn

Scikit-Learn, construit à partir des bibliothèques NumPy, SciPy et Matplotlib, est une bibliothèque d'apprentissage automatique. Il fournit des outils simples et efficaces pour accomplir des tâches courantes en analyse de données telles que la classification, la régression, le regroupement, la réduction de la dimensionnalité et le prétraitement des données.

Figure 3. 8- *Logo scitkit-learn [38]*

3. Collection de données

3.1.Création la collection des données

Nous avons crée une dateSet contenant un ensemble des données. Nous avons créé 100 images de mêmes formes pour 5 formes différentes à savoir ; cercle, carré, rectangle, pentagone et triangle.

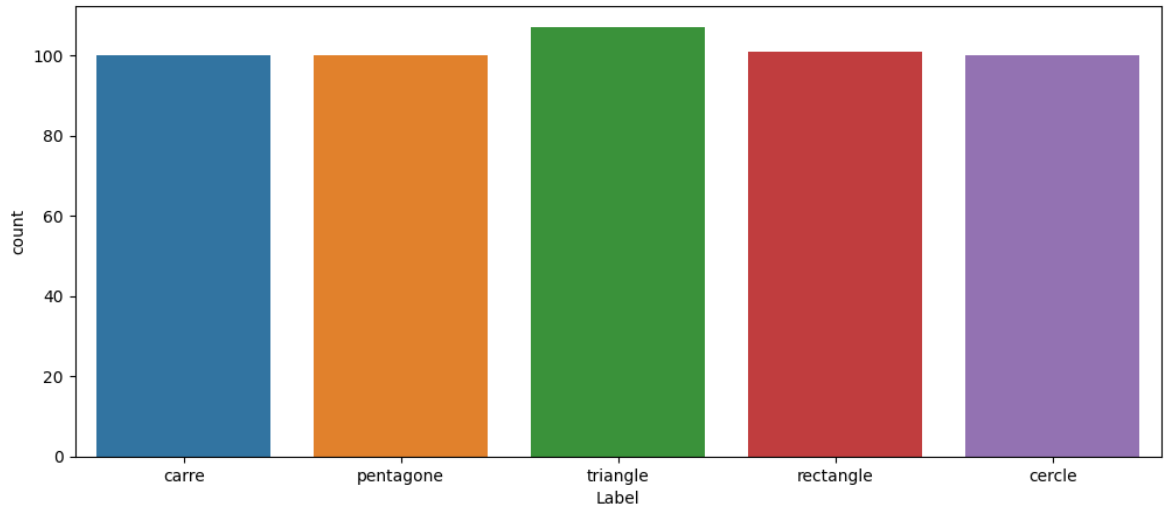


Figure 3. 9- *Répartition des formes dans le DataSET*

3.2. Préparation de fichier CSV

A ce stade nous appliqué le codage de Freeman pour chaque forme et nous avons enregistrer les chaînes obtenus dans un fichier CSV. Le code de codage est donné par la figure suivante ;

```
with open(filename, 'w', newline='') as csvfile:
    writer = csv.writer(csvfile)
    writer.writerow(['Chain', 'Label'])  # Écrire les en-têtes du fichier CSV

  # Parcourir chaque sous-dossier dans le dossier parent
    for class_folder in os.listdir(image_folder):
        class_folder_path = os.path.join(image_folder, class_folder)

        # Vérifier si l'élément dans le dossier parent est un dossier
        if os.path.isdir(class_folder_path):
            # Obtenir le label correspondant au nom du sous-dossier
            label = class_folder


            # Parcourir chaque image dans le sous-dossier
            for image_file in os.listdir(class_folder_path):
                image_path = os.path.join(class_folder_path, image_file)

                # Charger l'image, détecter les contours et obtenir les chaînes de Freeman
                image = cv2.imread(image_path)
                gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
                blurred = cv2.GaussianBlur(gray, (5, 5), 0)
                edged = cv2.Canny(blurred, 30, 150)
                contours, _ = cv2.findContours(edged, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

                # Obtenir la chaîne de Freeman pour chaque contour
                for contour in contours:
                    chain = freeman_chaincode(contour)
                    print("Chaîne de Freeman : ", chain)
                    # Écrire chaque chaîne de Freeman avec le label correspondant dans une ligne du fichier CSV
                    writer.writerow([chain, label])
```

Figure 3. 10- *Cadage Freeman*

3.3. Prétraitement des données

Après la collecte des données suit la phase de préparation des données, parfois appelée « prétraitement », est l'étape pendant laquelle les données brutes sont nettoyées et structurées en vue de l'étape suivante du traitement des données.

Pendant cette phase de préparation, nous avons calculer la longueur de la chaine la plus longue dans le fichier csv et nous avons ajouté des valeurs non significatives au chaîne ayant des valeurs inférieurs à cette valeur.

Par la suite nous avons divisé la collection de données en 2 parties (70 % pour la collection d'entrainement, 30 % pour la collection de test) et cela en utilisant la fonction proposée par Scikit-Learnsklearn.model_selection.train_test_split.

La figure suivante illustre la procédure de division de la collection de données ;

```
# Diviser les données en un ensemble de training et un ensemble de test.

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.30)
```

Figure 3. 11- *Division des données en trainement et test*

4. Création des modèles et présentation des résultats

5.3.Modèle 1

Le modèle 1 choisi est la régression linéaire. La figure suivante montre les résultats des calcules des mesures de performances sur la collection de test après l'entrainement du modèle 1.

```
Accuracy Score of Logistic Regression is: 0.5555555555555556
Confusion Matrix:
[[25  2  1  5  4]
 [ 1 19  0  0  1]
 [ 5  6 18  1  7]
 [17  4  1  6  1]
 [ 1  2  5  4 17]]
Classification Report:
              precision    recall  f1-score   support

       carre       0.51      0.68      0.58        37
      cercle       0.58      0.90      0.70        21
   pentagone       0.72      0.49      0.58        37
   rectangle       0.38      0.21      0.27        29
    triangle       0.57      0.59      0.58        29

    accuracy                           0.56       153
   macro avg       0.55      0.57      0.54       153
weighted avg       0.56      0.56      0.54       153
```

Figure 3. 12- *Calcul des performances du modèle 1*

5.4.Modèle 2

Le modèle 2 choisi est le modèle KNN. La figure suivante montre les résultats des calcules des mesures de performances sur la collection de test après l'entrainement du modèle 2.

```
Accuracy Score of KNN is : 0.5686274509803921
Confusion Matrix :
[[24  3  2  1  7]
 [ 1 19  0  0  1]
 [ 2 12 21  0  2]
 [17  7  1  3  1]
 [ 1  2  4  2 20]]
Classification Report :
              precision    recall  f1-score   support

       carre       0.53      0.65      0.59        37
      cercle       0.44      0.90      0.59        21
   pentagone       0.75      0.57      0.65        37
   rectangle       0.50      0.10      0.17        29
    triangle       0.65      0.69      0.67        29

    accuracy                           0.57       153
   macro avg       0.57      0.58      0.53       153
weighted avg       0.59      0.57      0.54       153
```

Figure 3. 13- *Calcul des performances du modèle 2*

5.5.Modèle 3

Le modèle 3 choisi est le SVM. La figure suivante montre les résultats des calcules des mesures de performances sur la collection de test après l'entrainement du modèle 3.

```
Accuracy Score of SVM is : 0.6078431372549019
Confusion Matrix :
[[21  2  5  2  7]
 [ 0 18  1  1  1]
 [ 3  8 23  2  1]
 [17  5  1  6  0]
 [ 0  1  2  1 25]]
Classification Report :
              precision    recall  f1-score   support

       carre       0.51      0.57      0.54        37
      cercle       0.53      0.86      0.65        21
   pentagone       0.72      0.62      0.67        37
   rectangle       0.50      0.21      0.29        29
    triangle       0.74      0.86      0.79        29

    accuracy                           0.61       153
   macro avg       0.60      0.62      0.59       153
weighted avg       0.60      0.61      0.59       153
```

Figure 3. 14- *Calcul des performances du modèle 3*

5.6.Modèle 4

Le modèle 2 choisi est l'arbre de décision. La figure suivante montre les résultats des calcules des mesures de performances sur la collection de test après l'entrainement du modèle 2.

```
Accuracy Score of Decision Tree is: 0.6535947712418301
Confusion Matrix:
[[26  0  3  3  5]
 [ 1 19  1  0  0]
 [ 3  3 25  1  5]
 [19  2  0  6  2]
 [ 0  1  3  1 24]]
Classification Report:
              precision    recall  f1-score   support

       carre       0.53      0.70      0.60        37
      cercle       0.76      0.90      0.83        21
   pentagone       0.78      0.68      0.72        37
   rectangle       0.55      0.21      0.30        29
    triangle       0.67      0.83      0.74        29

    accuracy                           0.65       153
   macro avg       0.66      0.66      0.64       153
weighted avg       0.65      0.65      0.63       153
```

Figure 3. 15- *Calcul des performances du modèle 4*

5.7.Modèle 5

Le modèle 5 choisi est la forêt aléatoire. La figure suivante montre les résultats des calcules des mesures de performances sur la collection de test après l'entrainement du modèle 5.

```
Accuracy Score of Random Forest is: 0.6862745098039216
Confusion Matrix:
[[26  2  3  1  5]
 [ 0 20  0  0  1]
 [ 2  6 26  1  2]
 [19  2  1  6  1]
 [ 0  0  1  1 27]]
Classification Report:
              precision    recall  f1-score   support

       carre       0.55      0.70      0.62        37
      cercle       0.67      0.95      0.78        21
   pentagone       0.84      0.70      0.76        37
   rectangle       0.67      0.21      0.32        29
    triangle       0.75      0.93      0.83        29

    accuracy                           0.69       153
   macro avg       0.70      0.70      0.66       153
weighted avg       0.70      0.69      0.66       153
```

***Figure 3. 16**-Calcul des performances du modèle 5*

5. Comparaison des résultats

La figure suivante montre un tableau récapitulatif des résultats des cinq modèles entrainés sur la même collection de données.

***Tableau 3. 1**-Comparaison des résultats des trois modèles*

	Accuracy
Modèle 1 : régression logistique	**0.55**
Modèle 2 : modèle KNN	**0.63**
Modèle 3 : modèle SVM	**0.60**
Modèle 4 : arbre de décision	**0.65**
Modèle 5 : forêt aléatoire	**0.68**

Conclusion

Dans ce dernier chapitre, nous avons présenté dans un premier temps, les différentes bibliothèques python utilisées dans le domaine de l'apprentissage automatique ainsi que l'utilité de chacune d'elles. Par la suite, nous avons créé les modèles proposés que nous

avons entrainé puis tester. Enfin, nous avons effectué une comparaison des résultats de ces trois modèles.

Conclusion générale

Le présent rapport a exploré en détail le domaine de la reconnaissance de forme, en mettant l'accent sur les algorithmes, les techniques et les méthodes utilisés dans ce domaine passionnant de l'informatique et de la vision par ordinateur.

Dans le premier chapitre, nous avons introduit les principes fondamentaux de la reconnaissance de forme et présenté les étapes clés du processus de reconnaissance. Nous avons également exploré la modélisation par expression régulière, une approche couramment utilisée pour la reconnaissance de forme, qui permet de représenter et de détecter des motifs dans les données visuelles.

Le deuxième chapitre a été consacré à l'étude des algorithmes de reconnaissance de forme. Nous avons examiné les approches basées sur les caractéristiques, qui utilisent des propriétés spécifiques des formes telles que la forme, la texture et la couleur. Nous avons également exploré les approches basées sur les modèles, qui utilisent des modèles statistiques, des réseaux de neurones, des arbres de décision et des graphes pour classifier les formes. De plus, nous avons examiné les approches hybrides qui combinent les deux approches. Enfin, nous avons discuté des métriques d'évaluation et des techniques d'amélioration des performances utilisées dans la reconnaissance de forme.

Le troisième chapitre s'est concentré sur les algorithmes de Freeman, qui sont couramment utilisés pour la reconnaissance de formes géométriques. Nous avons exploré le modèle de Freeman, qui utilise des directions spécifiques pour représenter les formes, ainsi que l'extraction des caractéristiques et la classification des formes utilisant ces algorithmes.

Le quatrième chapitre a abordé l'implémentation pratique de la reconnaissance de forme. Nous avons présenté l'environnement de développement utilisé, les bibliothèques Python essentielles telles que Numpy, Pandas, Matplotlib, OpenCV, Seaborn et Scikit-learn. Nous avons également discuté la création des modèles et de la comparaison des résultats obtenus.

En conclusion, ce rapport a permis d'approfondir nos connaissances sur la reconnaissance de forme en explorant différents aspects du domaine. Nous avons découvert les principes fondamentaux, les algorithmes et les techniques utilisées pour détecter et classifier les formes dans les données visuelles. Nous avons également pris conscience de l'importance de l'évaluation des algorithmes et des méthodes d'amélioration des performances pour obtenir des résultats précis et fiables.

Enfin, ce rapport a ouvert la voie à de nouvelles opportunités de recherche et a encouragé une exploration plus approfondie de la reconnaissance de forme. En combinant des approches basées sur les caractéristiques et les modèles, en utilisant des techniques avancées d'apprentissage automatique et en exploitant les bibliothèques et les outils disponibles, nous pouvons continuer à progresser dans ce domaine fascinant et exploiter tout son potentiel.

Webographie

[1] https://towardsdatascience.com/simplest-guide-for-regular-expressions-nlp-made-easy-bbb33cd694be [Consulté le 16/02/2023]

[2] https://www.analyticsvidhya.com/blog/2021/03/beginners-guide-to-regular-expressions-in-natural-language-processing/ [Consulté le 16/02/2023]

[3] https://aclanthology.org/D18-1224.pdf [Consulté le 16/02/2023]

[4] https://www.aurigait.com/blog/use-of-regex-in-ocr/ [Consulté le 16/02/2023]

[5] https://www-igm.univ-mlv.fr/~desar/Cours/automates/ch1.pdf [Consulté le 16/02/2023]

[6] http://pauillac.inria.fr/~remy/poly/compil/7/index.html [Consulté le 16/02/2023]

[7] https://damien.nouvels.net/cours/automates/4_ProprietesDesLanguagesReguliers.pdf [Consulté le 17/02/2023]

[8] https://members.loria.fr/KFort/files/fichiers_cours/ExpressionsRegulieres.pdf [Consulté le 17/02/2023]

[9] https://www.irif.fr/~carton/Enseignement/Complexite/ENS/Redaction/2006-2007/florian.praden.pdf [Consulté le 17/02/2023]

[10] https://www.emis.de/journals/BBMS/Bulletin/bul971/ziadi.pdf [Consulté le 17/02/2023]

[11] https://complex-systems-ai.com/theorie-des-langages/langages-reguliers-et-expressions-regulieres/ [Consulté le 17/02/2023]

[12] https://perso.liris.cnrs.fr/christine.solnon/langages.pdf [Consulté le 16/02/2023]

[13] https://infosetif.do.am/S4/TL/TL-SupportDeCours-MarcChemillier.pdf [Consulté le 17/02/2023]

[14] https://regenerativetoday.com/a-beginners-guide-to-match-any-pattern-using-regular-expressions-in-r/ [Consulté le 17/02/2023]

[15] https://medium.com/swlh/pattern-matching-with-regular-expressions-45442567b5d4 [Consulté le 17/02/2023]

[16] https://www.liquidweb.com/kb/what-are-regular-expressions/ [Consulté le 17/02/2023]

[17] https://openreview.net/pdf?id=EJKLVMB_9T [Consulté le 14/02/2023]

[18] https://arxiv.org/pdf/1908.03316.pdf [Consulté le 15/02/2023]

[19] https://theses.hal.science/tel-01802993/document [Consulté le 15/02/2023]

[20] https://geekyhumans.com/fr/detection-dobjet-en-python/ [Consulté le 15/02/2023]

[21] https://superdatabros.wordpress.com/2018/01/03/reconnaissance-de-formes-en-python-avec-keras-digits/ [Consulté le 14/02/2023]

[22] https://docstore.mik.ua/orelly/webprog/jscript/ch10_01.htm [Consulté le 14/02/2023]

[23] https://fr.wikipedia.org/wiki/R%C3%A9seau_neuronal_convolutif [Consulté le 28/04/2023]

[24] https://365datascience.com/tutorials/machine-learning-tutorials/decision-trees/ [Consulté le 28/04/2023]

[25] http://www.applis.univ-tours.fr/scd/EPU_DI/2017_PRD_Michaux_Kevin.pdf [Consulté le 28/04/2023]

[26] http://carolinepetitjean.free.fr/enseignements/ti/c6freeman.pdf [Consulté le 29/04/2023]

[27] https://www.slideserve.com/edison/synonymies-et-vecteurs-conceptuels [Consulté le 29/04/2023]

[28] https://www.engineersgarage.com/n-bit-gray-counter-using-vhdl/ [Consulté le 29/04/2023]

[29] http://e-biblio.univ-mosta.dz/bitstream/handle/123456789/9420/MINF70.pdf [Consulté le 13/05/2023]

[30] https://ruslanmv.com/assets/images/posts/2022-01-24-How-to-connect-Google-Colab-to-your-computer/colab.png [Consulté le 03/05/2023]

[31] https://python.developpez.com/tutoriels/debuter-avec-python-au-lycee/?page=comment-suivre-ce-tutoriel [Consulté le 03/05/2023]

[32] https://en.wikipedia.org/wiki/NumPy [Consulté le 04/05/2023]

[33] https://datascientest.com/wp-content/uploads/2022/01/illu_pandas-82-1024x562.png[Consulté le 16/04/2023]

[34] https://bioinfo-fr.net/wp-content/uploads/2017/11/logo_matplotlib.svg [Consulté le 16/04/2023]

[35] https://content.axopen.com/uploads/opencv_logo_62fb531c30.png [Consulté le 04/05/2023]

[36] https://seaborn.pydata.org/_static/logo-wide-lightbg.svg [Consulté le 16/04/2023]

[37] https://humancoders-formations.s3.amazonaws.com/uploads/course/logo/59/formation-machine-learning-avec-python.png [Consulté le 16/04/2023]

Résumé

Ce projet porte sur la reconnaissance de formes en utilisant des techniques d'apprentissage automatique en se basant sur l'algorithme de FreeMAN. L'objectif principal est de développer des modèles capables de prédire la classe ou la catégorie d'une forme donnée en fonction de ses caractéristiques..

Nous avons commencé par étudier l'état de l'art de la reconnaissance de forme. Par la suite, nous avons étudié les différents algorithmes de reconnaissance de formes, tels que les algorithmes basés sur les caractéristiques, les modèles statistiques, les réseaux de neurones, les arbres de décision et on s'est concentré spécifiquement sur les algorithmes de Freeman pour la reconnaissance de formes en se basant sur l'extraction des caractéristiques basée sur les contours et la classification des formes.

Finalement, nous avons implémenté les modèles KNN et SVM pour la reconnaissance des formes usuelles en utilisant une collection de données créés par nous même.

Abstract

This project focuses on pattern recognition using machine learning techniques based on the FreeMAN algorithm. The main objective is to develop models capable of predicting the class or category of a given shape based on its characteristics...

We began by studying the state of the art in shape recognition. Subsequently, we studied different pattern recognition algorithms, such as feature-based algorithms, statistical models, neural networks, decision trees and focused specifically on Freeman's algorithms for pattern recognition based on contour-based feature extraction and pattern classification.

Finally, we implemented KNN and SVM models for common pattern recognition using a self-created data collection.