Réf: …/…….

AU: 2022/2023

Université de Sousse

Ecole nationale d'ingénieurs de Sousse

Rapport du projet semestriel:

Détection De Fausse Monnaie

Réalisé par:
Hichri Sinda

Encadré par :
Mme. Sayadi Fatma

Filière:
1ère année électronique industrielle

Remerciement

Nous remercions Dieu le tout-puissant de nous avoir donné la force et la patience suffisantes pour mener à bien ce travail.

Nous tenons à adresser nos sincères gratitudes à tous ceux qui nous ont soutenus et motivés tout au long du parcours.

Nous remercions également Madame Sayadi Fatma pour son temps précieux et ses conseils constructifs, lors des différents suivis, concernant les missions évoquées dans ce projet.

Nous ne pouvons conclure ces remerciements sans exprimer toutes nos reconnaissances à toute l'équipe pédagogique de l'Ecole Nationale d'Ingénieurs de Sousse, tous les intervenants professionnels responsables et les membres du jury pour avoir accepté d'évaluer notre travail.

Table des matières:

Table des Figures:

Abréviations:

SVM : Support Vector Machine (Machine à vecteurs de support)
GLCM : Gray-Level Co-Occurrence Matrix (Matrice de co-occurrence au niveau du gris)
DT : Dinars Tunisiens
RGB : Red, Green & Blue (Rouge, Vert et Bleu)
HSV : Hue, Saturation & Value (Teinte, Saturation et Valeur)

I. Contexte

La production de fausses monnaies augmente de jour en jour. Ceci affecte considérablement l'économie. Pour celà, on a proposé un système de traitement d'image qui a pour but de distinguer la différence entre les billets de banque authentiques et contrefaites.
Le logiciel MATLAB sera utilisé pour mettre en œuvre ce système de détection et extraire les caractéristiques des billets. Le résultat permettra de prédire si le billet est faux ou non.

II. Revue de littérature

Au cours des dernières années, de nombreuses recherches ont été menées dans le domaine de la détection de la fausse monnaie.

Une recherche faite à l'aide de la **vérification avec des machines à vecteurs de support (SVM)[1]** se base sur le prétraitement des images de la monnaie à l'aide d'une variété de techniques, et puis l'extraction de ses différentes caractéristiques en utilisant **la technique du motif binaire local** (un opérateur de texture simple qui étiquette les pixels d'une image en seuillant le voisinage de chaque pixel et considérant le résultat en tant qu'un nombre binaire).
Une fois les caractéristiques extraites, il est important de classer la monnaie à l'aide d'un classificateur efficace appelé machine à vecteur de support et, finalement, un prototype capable de distinguer la différence entre les fausses et les vraies monnaies sera créé.

Une technique d'extraction de caractéristiques de texture basée sur **la technique GLCM[2]** a été étudiée. La méthode proposée est basée sur les caractéristiques des monnaies de papier indiennes. Les caractéristiques statistiques de premier ordre et de second ordre sont extraites initialement de l'entrée. **Les vecteurs de caractéristiques efficaces** sont donnés à l'unité de classification SVM pour la classification. Cette méthode a produit une précision de classification de 95,8 %.

Dans le document de recherche **Fake currency detection using image processing[3],** les caractéristiques cachées dans l'image du billet sont mises en évidence à l'aide de la lumière ultraviolette. Maintenant le traitement de l'image est effectué sur l'image acquise en utilisant des concepts tels que la segmentation de l'image, l'information sur les bords et l'extraction des caractéristiques.

III. Présentation générale

3.1. Introduction

Les progrès technologiques ont ouvert la voie à la duplication des devises de manière à ce qu'elles ne soient pas facilement reconnaissables. Des imprimantes avancées et de nouveaux logiciels d'édition sont utilisés pour créer de fausses devises. Celles-ci peuvent être simplement cachées dans des paquets de devises authentiques, c'est ainsi qu'elles circulent généralement dans le marché.

Une méthode basée sur l'extraction de caractéristiques de texture et l'apprentissage automatique est proposée. L'extraction des caractéristiques dans la méthode proposée est basée sur les caractéristiques de la texture des monnaies tunisiennes en papier en utilisant la technique GLCM. Les caractéristiques statistiques de premier ordre et de second ordre sont extraites initialement de l'entrée. Les vecteurs de caractéristiques efficaces sont donnés à l'unité de classification SVM pour la classification. La sortie est affichée dans une boîte de messages. SVM est l'un des classificateurs qui sont utilisés efficacement pour diverses applications de traitement d'images telles que la segmentation, la compréhension de la scène, la classification, etc…

3.2. Problématique

Le monde actuel est basé, d'une part, sur des transactions financières digitales entre les différents segments institutionnels. D'autre part, l'échange direct d'argent reste désormais le moyen le plus populaire de transaction. Ceci conduit à l'émergence de nombreux types de fraude et à l'émission de fausses monnaies.

Pour celà, un contrôle rigoureux des devises par l'Etat et les autorités monétaires est indispensable.

Plusieurs études ont été faites dans le but d'atteindre cet objectif, mais de nombreux obstacles ont été rencontrés tels que l'absence d'une base de données variée, le bon choix du moyen de classification et du logiciel, quelle méthode à employer pour prédire la différence, etc…

3.3. Objectif

Dans le cadre de notre premier projet semestriel, nous allons employer le logiciel Matlab pour construire un système de détection de fausse monnaie en employant l'approche GLCM-SVM.

IV. Description du projet

4.1. Diagramme Bete à cornes

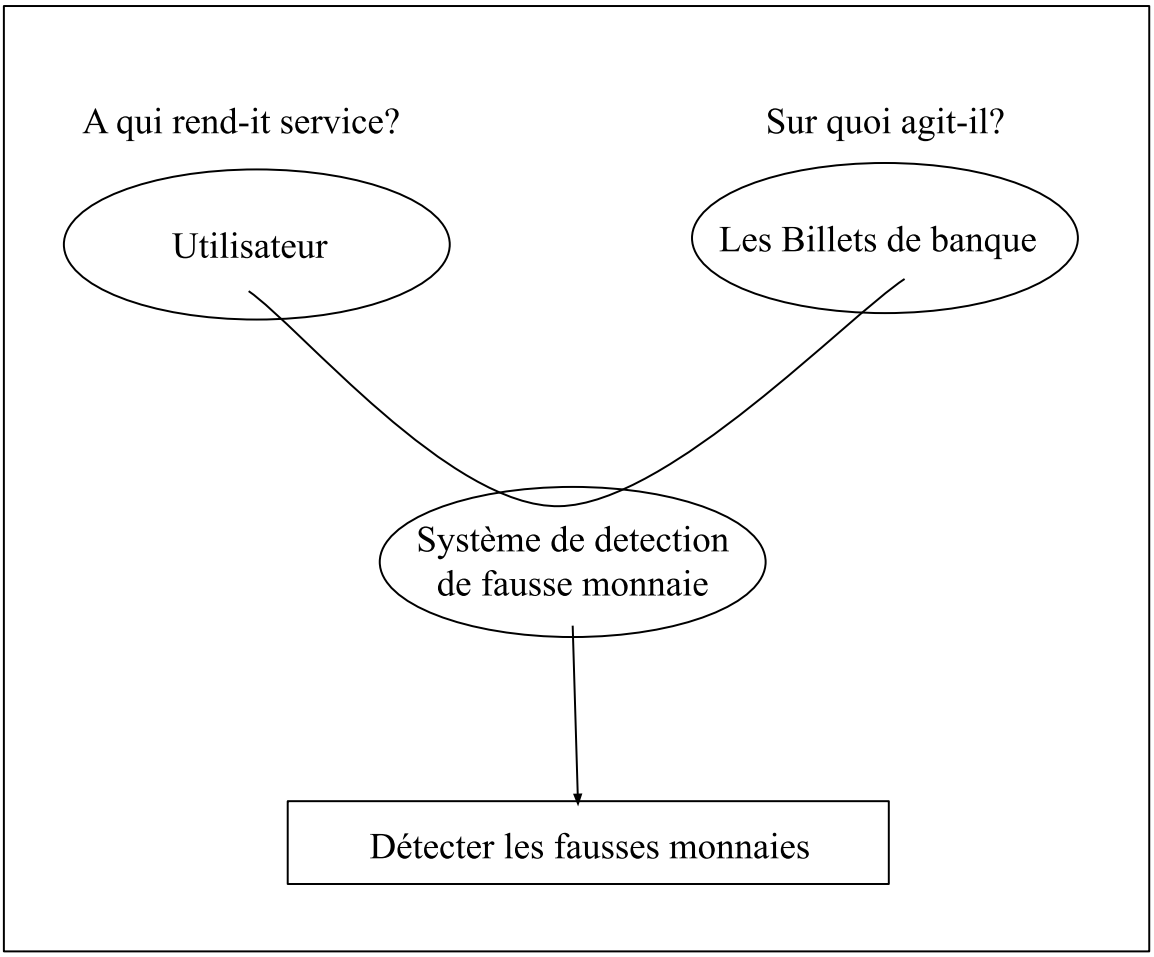
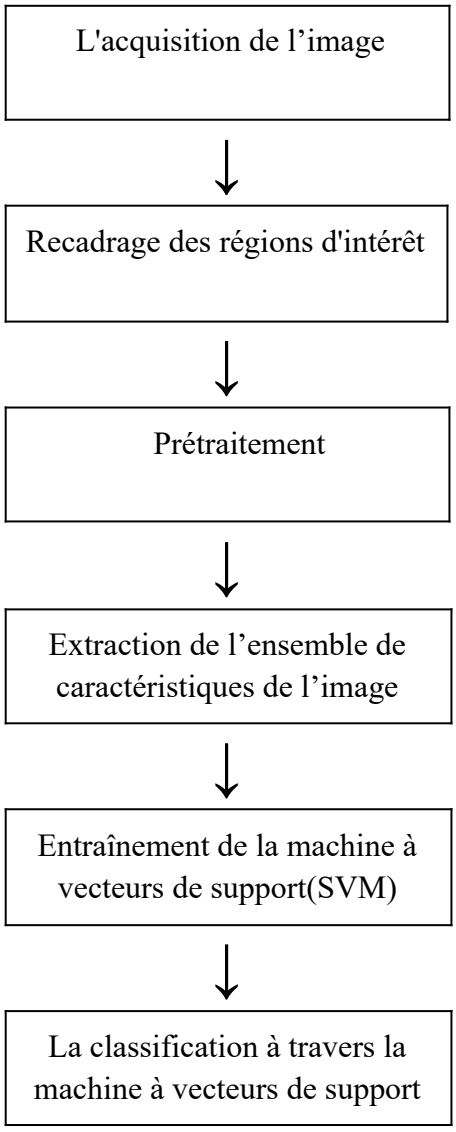


Figure 1 : Diagramme Bete à cornes

4.2. Méthodologie



4.2.1. L'acquisition de l’image:

L'acquisition d'images est un processus qui consiste à capturer des images de billets de banque afin de les prétraiter et d'en améliorer la qualité en vue de techniques de traitement ultérieures.

4.2.2. Recadrage des régions d'intérêt:

Cette étape permet de seuiller l'image RGB à l'aide de l'application de seuils de couleur et de masquer binairement l'image où la monnaie est présente. L'image segmentée est une image RGB composite.

4.2.3. Prétraitement:

L'image obtenue est maintenant convertie en image en niveaux de gris et redimensionnée en conséquence.

4.2.4. Extraction de l’ensemble de caractéristiques de l’image:

Les caractéristiques statistiques de premier ordre sont extraites des images recadrées selon les régions d'intérêt, puis converties en vecteurs de caractéristiques. Ces vecteurs de caractéristiques seront utilisés par le classificateur pour reconnaître les unités d'entrée et les unités de sortie cibles. La GLCM extrait ensuite les caractéristiques statistiques de second ordre dites aussi les caractéristiques de texture.

4.2.5. Entraînement de la machine à vecteurs de support (SVM):

Les vecteurs de caractéristiques extraits sont utilisés pour former le système.
Le processus de formation implique l'apprentissage des modèles de billets de banque et puis le résultat de cette formation sera le classificateur de vecteurs de support.

4.2.6. La classification à travers la machine à vecteurs de support:

Une fois que les vecteurs de caractéristiques des billets sont collectés, il est nécessaire de reconnaître le motif des billets en se basant sur ces caractéristiques extraites. Le résultat de ce modèle de classification permet de prédire si la monnaie est fausse ou non.

V. Fonctionnement et code

5.1. L'acquisition de l'image

Tout d'abord, un ensemble d'images du billet 10DT ont été prélevés d'une base de donnée tunisienne intitulée " Tunisian Currency [bills][4] " se trouvant sur Kaggle et puis manipulées à l'aide du logiciel Adobe Photoshop et l'éditeur des photos Meitu afin de mettre en valeur les différences qu'on peut distinguer entre les billets de monnaies réelles et fausses.

Ces images ont été manipulées afin de créer une base de données qui contient également un ensemble d'images des billets authentiques et d'autres qui sont contrefaites.
Cette étape de création de base de données était obligatoire faute de présence des bases de données contenant des fausses monnaies tunisiennes sur les moteurs de recherches.

Figure 2: Un vrai billet de 10 DT tunisien.

Figure 3: Un faux billet de 10 DT tunisien.

5.2. Recadrage des régions d'intérêt

Afin de cibler la région où le billet existe dans une image à traiter ou à tester, le passage par cette étape de recadrage est nécessaire afin d'augmenter la précision des résultats.
L'application de seuils de couleur (Color Thresholder App) déjà prédifinie dans la boite à outil de traitement d'image sur Matlab nous a permis de génerer automatiquement la fonction "createMask".
Cette fonction consiste à convertir l'image de RGB en HSV et en prélever ses valeurs caractéristiques minimales et maximales en teinte, saturation et valeur à partir de son histogramme de couleurs.

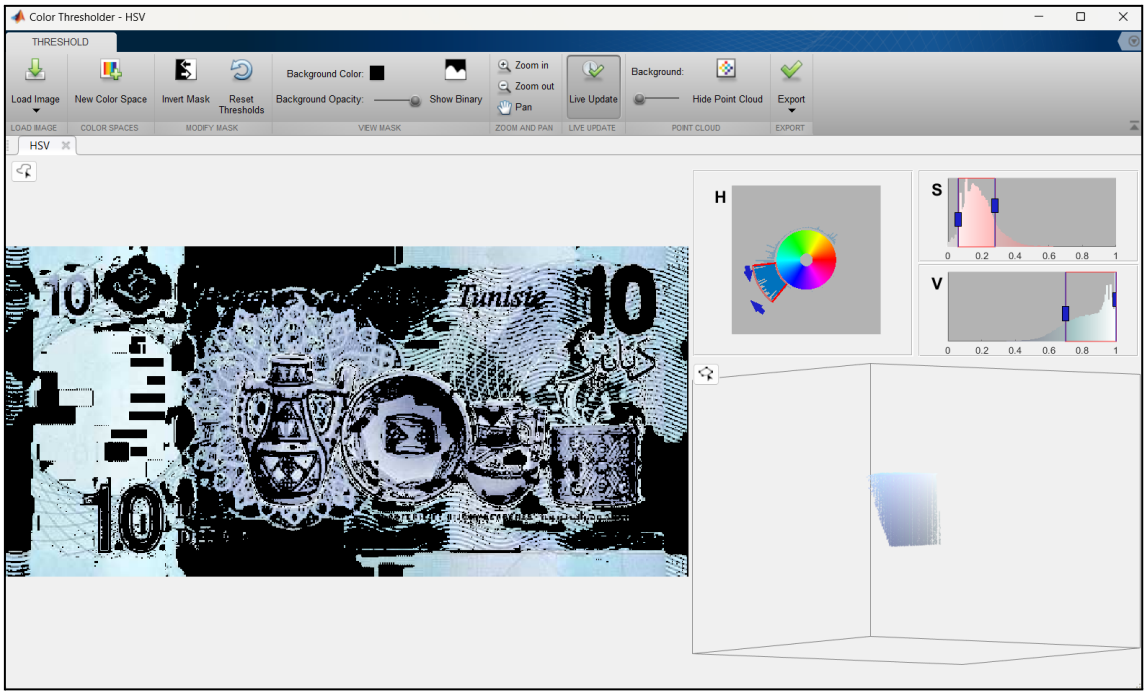


Figure 4: Histogramme des couleurs et sélection des valeurs min et max de (H,S,V).

```
function [BW,maskedRGBImage] = createMask(RGB)
% Convert RGB image to chosen color space
I = rgb2hsv(RGB);

% Define thresholds for channel 1 based on histogram settings
channel1Min = 0.529;
channel1Max = 0.633;

% Define thresholds for channel 2 based on histogram settings
channel2Min = 0.069;
channel2Max = 0.262;

% Define thresholds for channel 3 based on histogram settings
channel3Min = 0.710;
channel3Max = 1.000;
```

Figure 5: Définition des valeurs min et max de (H,S,V) dans la fonction createMask.

A partir des valeurs de teinte, saturation et valeur prélevées de l'histogramme des couleurs, un filtre, qui sera appliqué ultérieurement, est créé en utilisant ces valeurs.
Ce filtre consiste à attribuer la valeur “0” au régions de l’image dont le billet de monnaie n’existe pas.
Ceci servira à convertir toutes les pixels dont les valeurs valent “0” en noir afin de cacher les détails anodins.

```
% Create mask based on chosen histogram thresholds
sliderBW = (I(:,:,1) >= channel1Min ) & (I(:,:,1) <= channel1Max) & ...
    (I(:,:,2) >= channel2Min ) & (I(:,:,2) <= channel2Max) & ...
    (I(:,:,3) >= channel3Min ) & (I(:,:,3) <= channel3Max);
BW = sliderBW;

% Initialize output masked image based on input image.
maskedRGBImage = RGB;

% Set background pixels where BW is false to zero.
maskedRGBImage(repmat(~BW,[1 1 3])) = 0;

end
```

Figure 6: Création du filtre et masquage de l'arrière-plan de l'image à manipuler.

5.3. Prétraitement:

L'image ainsi obtenue de l'étape précédente est une image composite contenant des régions colorées et d'autres masqués en noir.

L'étape du prétraitement nous permet de convertir notre base de données d'images en niveaux de gris (après avoir appliqué le filtre) et de les redimensionner en conséquence. Ensuite chaque valeur de pixel sera représentée dans une matrice, dite GLCM.

Le passage par la conversion en gris est important afin d'améliorer la précision des caractéristiques de texture et les caractéristiques statistiques de premier et de second ordre de chaque image qui seront prélevées ultérieurement.

```
df=[]

for i = 1:20
    i
    B=imread(strcat(int2str(i),'.jpg'));
    I=imresize(B,0.5);

   %%%%%%%%color%%%%%%%%%%
  %%%%%[BW,maskedImage] = segmentImage(RGB);
   [BW,maskedRGBImage] = createMask(I);
            seg_img = maskedRGBImage;

            B = rgb2gray(seg_img);
    imshow(B)
            SE = strel('rectangle',[40 30]);
            img=imopen(B,SE);

            glcms = graycomatrix(img);
```

Figure 7: Redimensionnement, conversion en gris et représentation des valeurs de chaque pixel sous forme d'une matrice GLCM.

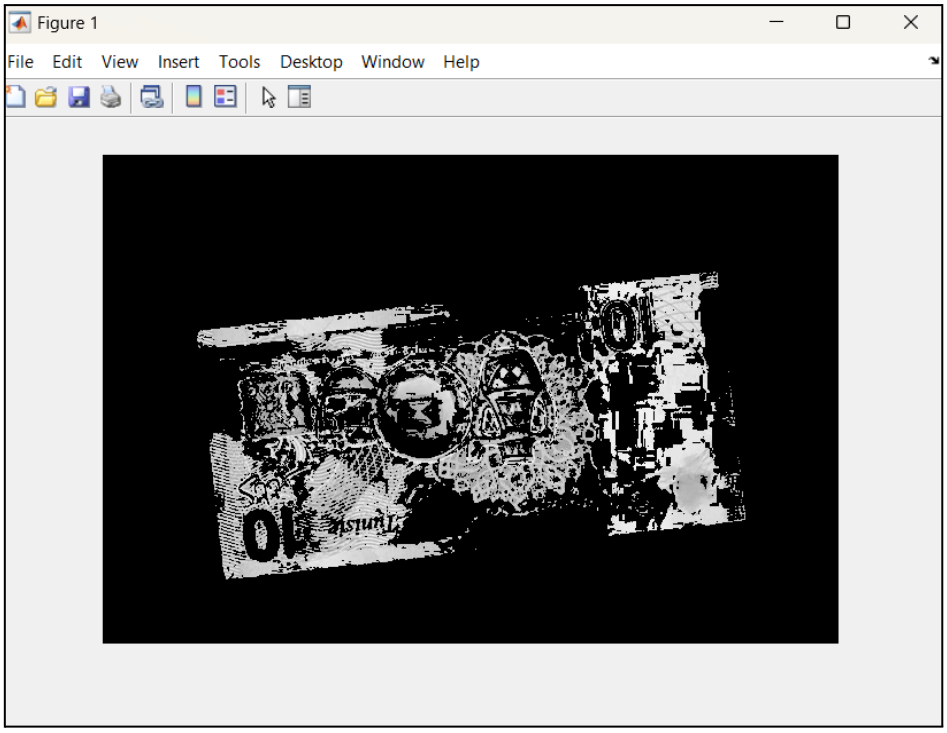


Figure 8: Exemple d'une image seuillée et convertie en gris.

5.4. Extraction de l'ensemble de caractéristiques de l'image:

Maintenant le calcul de l'ensemble des caractéristiques de texture (aussi appelés caractéristiques statistiques de second ordre) de chaque image est possible; une nouvelle matrice "stats" sera définie et contiendra les valeurs de ces caractéristiques qui sont sous forme de: contraste, corrélation, énergie et homogénéité caractérisant chaque image de la base de données.

```
26 -            stats = graycoprops(glcms,'Contrast Correlation Energy Homogeneity');
27
28 -            Contrast = stats.Contrast;
29 -            Energy = stats.Energy;
30 -            Homogeneity = stats.Homogeneity;
31 -            Mean = mean2(seg_img);
32 -            Standard_Deviation = std2(seg_img);
33 -            Entropy = entropy(seg_img);
34 -            RMS = mean2(rms(seg_img));
35 -            Variance = mean2(var(double(seg_img)));
36 -            a = sum(double(seg_img(:)));
37 -            Smoothness = 1-(1/(1+a));
```

Figure 9: Calcul des caractéristiques statistiques de chaque image et sauvegarde des valeur dans la matrice "stats"

Après avoir défini les valeurs des caractéristiques de texture de chaque photo, les caractéristiques statistiques de premier ordre, qui serviront d'améliorer la précision de la classification de l'image passée en tant que "input" ultérieurement, sont aussi calculées.
Ces caractéristiques sont sous forme de: moyenne, écart-type, entropie, moyenne quadratique, variance et finesse.

5.5 Entraînement de la machine à vecteurs de support(SVM):

Dans cette partie on va créer et entraîner un classificateur SVM qui sera entraîné par un ensemble d'images de la base de données chargées dans "TrainingSet" et leurs étiquettes de classe (labels) correspondantes dans "GroupTrain". Une fois le classificateur entraîné, il sera utilisé pour prédire le label d'une nouvelle image passée dans "TestSet".

La valeur "1" attribuée à une image de la base de données signifie que ce billet d'argent est réel. Par conséquent, la valeur "2" signifie que ce billet est contrefait.

```
107        %%%%%%%%%%%%%%%%%%%%%%%%trainning
108 -      TrainingSet=df;
109 -      GroupTrain={'1','1','1','1','1','1','1','1','1','1','2','2','2','2','2','2','2','2','2','2'};
110 -      TestSet=Testftr;
```

Figure 10: L'apprentissage du SVM à partir de la base de données des billets réels et contrefaits.

5.6. La classification à travers la machine à vecteurs de support:

Dans un premier temps, les caractéristiques de premier et de second ordre de l'image à tester sont prélevées.
Nous fournissons, tout d'abord, une image à tester et nous suivons les mêmes étapes qu'on a poursuivies lors du prélèvement des caractéristiques de chaque image de la base de données.

```
60     %%%%%%%%%%%%get test image %%%%%%%%%%
61 -                [fname,path]=uigetfile('.jpg','Provide currency for testing');
62 -   filename=strcat(path,fname);
63 -   B=imread(filename);
64 -   figure
65 -   I=imresize(B,0.5);
66 -   imshow(I);
67 -   title('Original Image');
```

Figure 11: La fourniture d'une image à tester.

```
69 -    [BW,maskedRGBImage] = createMask(I);
70 -                seg_img = maskedRGBImage;
71
72 -                B = rgb2gray(seg_img);
73 -                SE = strel('rectangle',[40 30]);
74 -                img=imopen(B,SE);
75
76 -                glcms = graycomatrix(img);
77
78 -                stats = graycoprops(glcms,'Contrast Correlation Energy Homogeneity');
79
80 -                Contrast = stats.Contrast;
81 -                Energy = stats.Energy;
82 -                Homogeneity = stats.Homogeneity;
83 -                Mean = mean2(seg_img);
84 -                Standard_Deviation = std2(seg_img);
85 -                Entropy = entropy(seg_img);
86 -                RMS = mean2(rms(seg_img));
87                  %Skewness = skewness(img)
88 -                Variance = mean2(var(double(seg_img)));
89 -                a = sum(double(seg_img(:)));
90 -                Smoothness = 1-(1/(1+a));
```

Figure 12: Seuillage, redimensionnement, conversion en gris et prélèvement des caractéristiques statistiques de l'image à tester.

Maintenant la partie la plus significative du travail est atteinte; C'est la création de la SVM:

```
111      %%%%%%%%%%SVM
112 -    Y=GroupTrain;
113 -    classes=unique(Y);
114 -    SVMModels=cell(length(classes),1);
115 -    rng(1);   %Reproductivity
116
117 -    for j=1:numel(classes)
118 -        idx=strcmp(Y',classes(j));
119 -        SVMModels{j}=fitcsvm(df,idx,'ClassNames',[false true],'Standardize',true,'KernelFunction','rbf','BoxConstraint',1)
120 -    end
121 -    xGrid=Testftr;
122 -    for j=1:numel(classes)
123 -    [~,score]=predict(SVMModels{j},xGrid)
124 -    Scores(:,j)=score(:,2);
125 -    end
```

Figure 13: La machine à vecteurs de support SVM.

Tout d'abord, on crée la variable 'Y' et on lui attribue la variable ‘GroupTrain’. ‘Y' est le vecteur contenant les étiquettes de classe de la base de données d'apprentissage.

Ensuite, on utilise la fonction ‘unique()’ afin d’extraire les étiquettes de classe uniques de 'Y' et les assigner à la variable 'classes'.
La variable ‘classes’ sera un vecteur contenant les deux valeurs ‘1’ et ‘2’ qui sont également les valeurs qu’on a assigné aux images authentiques et contrefaites de la base de données utilisée lors de l’étape de l’apprentissage.

Puis, on crée un tableau de cellules 'SVMModels' avec le même nombre d'éléments que le nombre de classes. Ce tableau sera divisé donc en deux colonnes.

La fonction 'rng(1)' est utilisée pour définir la tête du fil du générateur de nombres aléatoires, qui est une fonction mathématique qui génère une séquence de nombres aléatoires.
La tête du fil est une valeur de départ utilisée pour initialiser le générateur de nombres aléatoires. Lorsque la même valeur du tête du fil est utilisée, la même séquence de nombres aléatoires sera générée, ce qui est utile pour des objectifs de reproductibilité.
La tête du fil dans notre code est définie par la valeur ‘1’, cela signifie que tous les nombres aléatoires générés dans le code seront basés sur la valeur ‘1’, garantissant que la même séquence de nombres aléatoires sera générée chaque fois que le code est exécuté avec cette valeur de la tête de fil. Ceci est utile dans les situations où la reproductibilité est importante, dans notre cas; pour l’expérimentation.

Maintenant une boucle ‘for’ commence avec la variable 'j' initialisée à 1, et se poursuit jusqu'à ce que le nombre d'éléments dans 'classes' soit atteint.

Dans la boucle ‘for’, on a utilisé la fonction ‘strcmp()’ pour créer un index logique 'idx' qui indique quels éléments de 'Y' correspondent à la classe en cours de traitement (classes(j)).

Ensuite, la fonction ‘fitcsvm()’ est utilisée pour entraîner un SVM binaire, en utilisant les données 'df' (tableau contenant les caractéristiques des images de la base de données) et les étiquettes 'idx' comme entrées.
L'option 'ClassNames' est définie sur [false true], indiquant que la classe négative est celle qui n'est pas étiquetée avec la classe actuelle.
L'option 'Standardize' est définie à true, ce qui signifie que les données seront standardisées avant d'être utilisées dans la SVM.
L'option 'KernelFunction' est définie sur 'rbf', ce qui indique qu'un noyau de fonction de base radiale sera utilisé.
Et l'option 'BoxConstraint' est définie à ‘1’, ce qui est un paramètre d'optimisation.

Dès qu’on sort de la boucle précédente, une nouvelle boucle ‘for’ est utilisée pour prédire les étiquettes de classe de l'ensemble de test, qui est passé à la fonction comme 'xGrid'.

Dans cette boucle, la fonction ‘predict()’ est utilisée pour prédire les étiquettes de classe pour l'ensemble de test en utilisant le modèle SVM pour la classe actuelle. La fonction predict() renvoie également le score de la classe positive, qui est utilisé pour calculer le score de chaque classe.

Enfin, la variable 'Scores' est complétée par les scores de la classe actuelle, qui seront utilisés ultérieurement pour déterminer les étiquettes de classe finales pour l'ensemble de test.

Après avoir créé le modèle SVM, on peut déterminer les étiquettes de classe finale de l'ensemble de test en se basant sur les scores calculés par les modèles SVM.

```
128 -    [~,maxScore]=max(Scores,[],2)
129
130 -    result=maxScore;
131
132 -    if result == 1
133 -        msgbox('GENUINE')
134 -    elseif result == 2
135 -       msgbox('FAKE');
136 -    else
137 -        msgbox('NONE')
138 -    end
```

Figure 14: Détermination et affichage du résultat.

La fonction ‘max()’ a pour but de trouver la valeur maximale de chaque ligne de la variable 'Scores' qui contient également les scores de chaque classe pour toutes les instances de test, avec une seule colonne pour chaque classe.
La fonction ‘max()’ renvoie ensuite la valeur maximale de chaque ligne et l'affecte à la variable 'maxScore'.

La ligne suivante attribue la variable 'maxScore' à la variable 'result'.
La variable 'maxScore' contient l'index de la classe ayant le score le plus élevé pour chaque instance de test. Et comme l'index correspond aux étiquettes de classe, cette variable représente les étiquettes de classe prédites pour l'ensemble de test.

Les lignes de code suivantes utilisent les instructions 'if-elseif-else' pour vérifier la valeur de 'result' et afficher une boîte de message avec l'étiquette correspondante.

L'instruction 'if' vérifie si 'result' est égal à 1, ce qui signifie que le label de la classe est 'GENUINE'. Si c'est le cas, une boîte de message est affichée avec le texte 'GENUINE'.

L'instruction 'elseif' vérifie si le 'résultat' est égal à 2, ce qui signifie que l'étiquette de classe est 'FAKE'. Si c'est le cas, une boîte de message s'affiche avec le texte "FAKE".

Si aucune des conditions n'est vraie, l'instruction 'else' est exécutée et une boîte de message contenant le texte 'NONE' s'affiche.

En résumé, dans cette partie du code, on a pris les scores des étapes précédentes, trouvé la classe avec le score le plus élevé pour chaque instance de test, et attribué l'étiquette de classe correspondante à cette instance. Ensuite, on a affiché une boîte de message avec l'étiquette de classe correspondante.

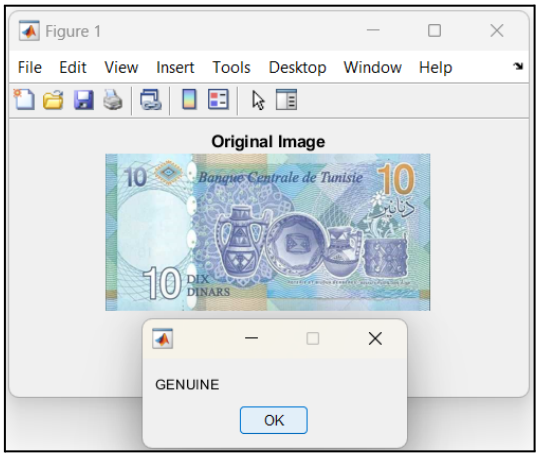


Figure 15: Boîte de message d'un billet authentique passé en test

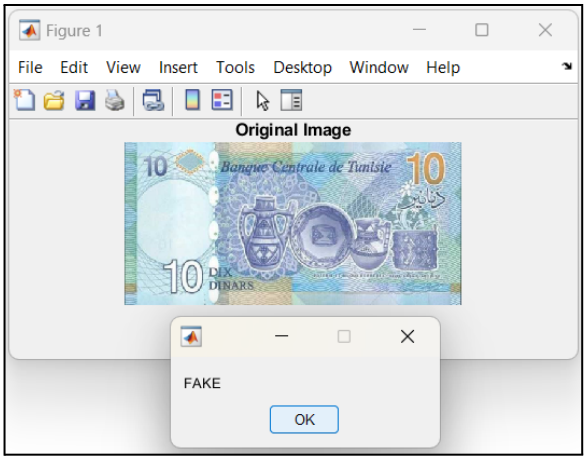


Figure 16: Boîte de message d'un billet contrefait passé en test

VI. Conclusion

La machine à vecteurs de support (SVM) est l'un des algorithmes d'apprentissage automatique supervisé les plus faciles à manipuler. L'idée de base de la SVM est de trouver un hyperplan (une ligne ou un plan) qui sépare les données en différentes classes. Cet hyperplan est choisi de manière à maximiser la marge, qui correspond à la distance entre l'hyperplan et les points de données les plus proches de chaque classe.

L'utilisation d'une base de données variée joue un rôle important dans l'augmentation de la précision du modèle SVM créé. Malheureusement, la pénurie d'une base de données des monnaies tunisiennes contrefaites a exigé une manipulation manuelle des images disponibles. Même si l'édition des images soit parfaite et ne soit pas détectable par l'œil humain, l'imitation identique d'un faux billet de banque ne sera jamais atteinte. Pour celà, bien que ce code fonctionne bien sur la base de données qu'on a créer, on ne peut jamais prédire si cela reste valide dans le cas d'une base de données réelle. Par conséquent, la nécessité de tester ce système en permanence demeure.

Et comme il est toujours possible de s'améliorer, la présence d'une base de données variées va nous permettre d'employer ce code pour la détection de tous types de devises tunisiennes et non seulement la pièce de 10 DT.

VII. Références Bibliographiques:

[1] Chin-Chen Chang, Tai-Xing Yu and Hsuan-Yen Yen, **Paper Currency Verification with Support Vector Machines** (2008 IEEE; Third International IEEE Conference on Signal Technologies and Internet-Based Systems.)
Disponible à: https://web.njit.edu/~usman/courses/cs732_spring15/04618864.pdf

[2] P. Mohanaiah, P. Sathyanarayana and L. GuruKumar, **Image Texture Feature Extraction Using GLCM Approach** (International Journal of Scientific and Research Publications, Volume 3, Issue 5, May 2013 1 ISSN 2250-3153)
Disponible à: https://www.ijsrp.org/research-paper-0513/ijsrp-p1750.pdf

[3] Tushar Agasti, Gajanan Burand, Pratik Wade and P Chitra, **Fake currency detection using image processing** (IOP Conf. Series: Materials Science and Engineering 263 (2017))
Disponible à: https://iopscience.iop.org/article/10.1088/1757-899X/263/5/052047

[4] Adam Miled, Mehri Chaima, **Tunisian Currency [Bills]** (October 2022) récupérée de Kaggle (CC0 1.0)
Disponible à: https://www.kaggle.com/datasets/adammiled/tunisian-currency