République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Alger 1 – BENYOUCEF Benkhedda

Faculté des Sciences
Département Informatique

Rapport

Module : Intelligence Artificielle

Réalisé par :

Moulay Abdellah Asma
Belabbas Rania

Section : M1 ISII groupe 3

Année universitaire : 2024/2025

Pour réaliser cela, on a utilisé Anaconda et Jupyter. Anaconda simplifie la gestion des bibliothèques et la création d'environnements isolés, tandis que Jupyter Notebook permet d'écrire et d'exécuter du code de manière interactive, avec une documentation intégrée et des visualisations en temps réel.

Lors de leurs installations, on a rencontré aucun problème, car nous disposions déjà de ce logiciel depuis l'année dernière.

1- Tout d'abord on a essayé le code que vous nous avez envoyer et on a rencontré cette erreur :

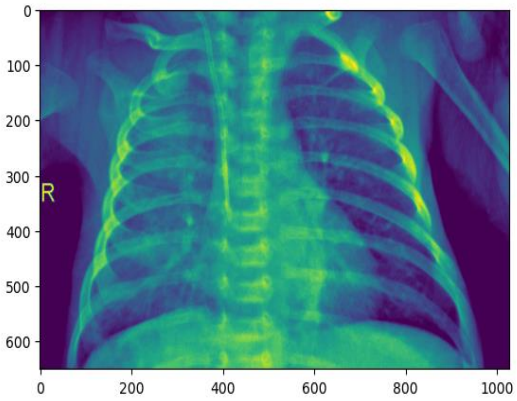
```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[2], line 18
     16 print("Dimensions de l'image : ",img.shape) # Dimensions x, y et nombre de couleurs
     17 print("Pixel 500 sur 200 : ",img[200,500])
---> 18 print("Composante rouge de ce dernier pixel :",img[200,500][0])
     19 # Tous le rouge de tous les pixels
     20 print("Rouge : ",img[:,:,0])

IndexError: invalid index to scalar variable.
```

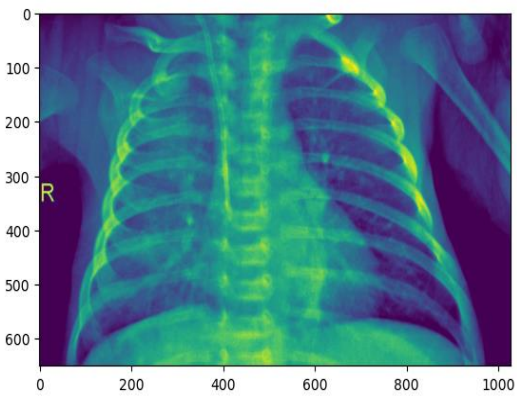
Explication : invalid index to scalar variable : se produit lorsque on essaye d'accéder à un index d'une variable qui n'est pas un tableau ou une liste, mais plutôt une valeur scalaire.

<u>Pour corriger cela :</u>

- Si l'image est en niveaux de gris : utilisez directement img [200, 500].
- Si l'image est colorée (RVB) : utilisez img [200, 500, 0] pour obtenir le composant rouge.

<u>Résultat :</u>





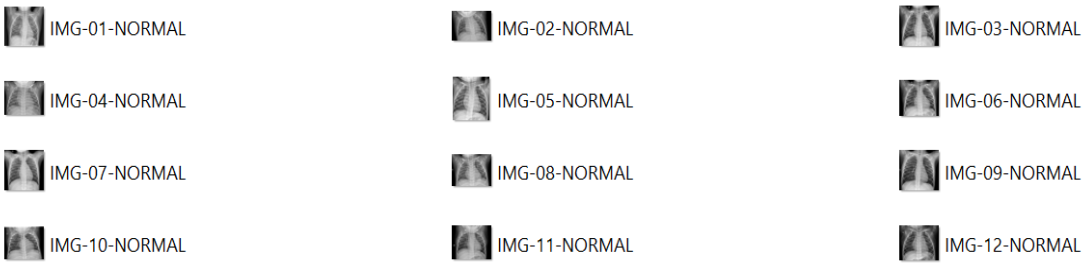
```
Dimensions de l'image :  (650, 1028)
Pixel 500 sur 200 :  148
Composante rouge de ce dernier pixel : 148
Rouge :  [[ 0  0  0 ... 38 45 49]
 [ 0  0  0 ... 37 43 47]
 [ 0  0  0 ... 36 42 46]
 ...
 [ 0  0  0 ...  0  0  0]
 [ 0  0  0 ...  0  0  0]
 [ 0  0  0 ...  0  0  0]]
```

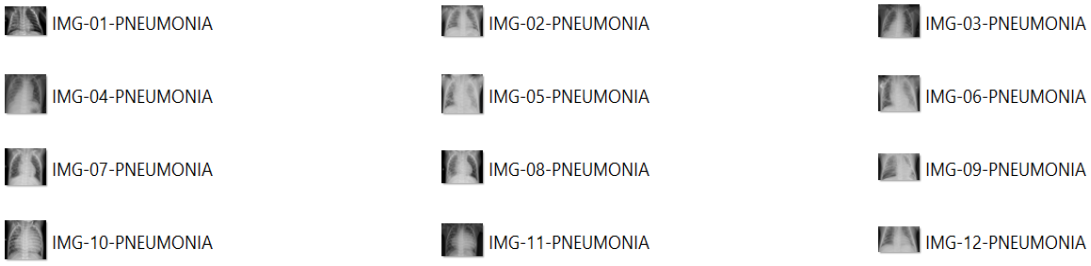
2- Renommage des images :

Nous avons commencé par cette étape avant d'entamer l'extraction des caractéristiques afin d'organiser les images et d'éviter des problèmes techniques tels que les doublons ou les incompatibilités. Cependant, ce n'est pas toujours nécessaire de le faire.

```python
import os
main_folder = 'C:\\Users\\HP\\TP-IA-G2\\Datasets'  # Update this to your main folder path

# Define the subfolder names
normal_folder = os.path.join(main_folder, 'normal')
pneumonia_folder = os.path.join(main_folder, 'pneumonia')

# Function to rename images in a folder
def rename_images(folder, prefix):
    for count, filename in enumerate(os.listdir(folder)):
        # Construct the old file path
        old_file_path = os.path.join(folder, filename)

        # Check if it's a file (to avoid renaming folders)
        if os.path.isfile(old_file_path):
            # Create a new filename with padded zeros
            new_filename = f"IMG-{count + 1:02d}-{prefix}.jpg"  # Change the extension if needed
            new_file_path = os.path.join(folder, new_filename)

            # Rename the file
            os.rename(old_file_path, new_file_path)
            print(f'Renamed: {old_file_path} to {new_file_path}')

# Rename images in both folders
rename_images(normal_folder, 'NORMAL')
rename_images(pneumonia_folder, 'PNEUMONIA')
```

Résultat :





Note : Il faut toujours changer le chemin d'accès avec le chemin où se trouve le dossier afin de modifier le nom des images.

3- Redimensionnement des images :

C'est une étape cruciale pour s'assurer que le traitement des données est efficace, cohérent et compatible avec algorithmes utilisés.

- D'abord on a vérifié si les images du dossier donné ont la même taille, par exemple avec le dossier PNEUMONIA on a trouvé ce résultat :
  (Nous avons rencontré une erreur lors de l'utilisation de cv2, car nous n'avions pas installé la bibliothèque OpenCV, qui permet de lire et de manipuler les images.)

Code :

```
import cv2
import os

# Chemin du dossier contenant les images
folder_path = "C:\\Users\\User\\OneDrive\\Documents\\M1 ISII\\TP IA\\Datasets\\PNEUMONIA"

# Initialiser une variable pour stocker la taille de la première image
initial_size = None
same_size = True

for filename in os.listdir(folder_path):
    img_path = os.path.join(folder_path, filename)
    img = cv2.imread(img_path)

    if img is not None:
        # Obtenir la taille de l'image (hauteur, largeur)
        img_size = img.shape[:2]  # (hauteur, largeur)

        if initial_size is None:
            initial_size = img_size  # Définir la taille de référence
        else:
            # Comparer la taille de l'image actuelle avec la taille de référence
            if img_size != initial_size:
                print(f"L'image {filename} a une taille différente : {img_size}")
                same_size = False
    else:
        print(f"Erreur lors du chargement de l'image : {filename}")

if same_size:
    print("Toutes les images ont la même taille :", initial_size)
else:
    print("Certaines images ont des tailles différentes.")
```

Résultat :

```
L'image IMG-002_PNEUMONIA.jpeg a une taille différente : (485, 856)
L'image IMG-003_PNEUMONIA.jpeg a une taille différente : (1310, 1646)
L'image IMG-004_PNEUMONIA.jpeg a une taille différente : (1152, 1264)
L'image IMG-005_PNEUMONIA.jpeg a une taille différente : (1365, 1736)
L'image IMG-006_PNEUMONIA.jpeg a une taille différente : (1392, 1666)
L'image IMG-007_PNEUMONIA.jpeg a une taille différente : (1535, 1762)
L'image IMG-008_PNEUMONIA.jpeg a une taille différente : (663, 981)
L'image IMG-009_PNEUMONIA.jpeg a une taille différente : (439, 712)
L'image IMG-010_PNEUMONIA.jpeg a une taille différente : (840, 1240)
L'image IMG-011_PNEUMONIA.jpeg a une taille différente : (1056, 1448)
L'image IMG-012_PNEUMONIA.jpeg a une taille différente : (1160, 1328)
L'image IMG-013_PNEUMONIA.jpeg a une taille différente : (1056, 1248)
L'image IMG-014_PNEUMONIA.jpeg a une taille différente : (952, 1200)
L'image IMG-015_PNEUMONIA.jpeg a une taille différente : (1345, 1654)
L'image IMG-016_PNEUMONIA.jpeg a une taille différente : (1265, 1736)
L'image IMG-017_PNEUMONIA.jpeg a une taille différente : (592, 1032)
L'image IMG-018_PNEUMONIA.jpeg a une taille différente : (1110, 1276)
L'image IMG-019_PNEUMONIA.jpeg a une taille différente : (840, 1168)
L'image IMG-020_PNEUMONIA.jpeg a une taille différente : (784, 1120)
L'image IMG-021_PNEUMONIA.jpeg a une taille différente : (776, 1160)
L'image IMG-022_PNEUMONIA.jpeg a une taille différente : (752, 928)
L'image IMG-023_PNEUMONIA.jpeg a une taille différente : (896, 1224)
L'image IMG-024_PNEUMONIA.jpeg a une taille différente : (588, 932)
L'image IMG-025_PNEUMONIA.jpeg a une taille différente : (697, 1033)
L'image IMG-026_PNEUMONIA.jpeg a une taille différente : (824, 1200)
L'image IMG-027_PNEUMONIA.jpeg a une taille différente : (1326, 1790)
L'image IMG-028_PNEUMONIA.jpeg a une taille différente : (696, 1256)
L'image IMG-029_PNEUMONIA.jpeg a une taille différente : (856, 1152)
L'image IMG-030_PNEUMONIA.jpeg a une taille différente : (728, 1248)
L'image IMG-031_PNEUMONIA.jpeg a une taille différente : (784, 1176)
Certaines images ont des tailles différentes.
```

- On a trouvé que certaines images ont des tailles différentes donc on a redimensionné toutes les images à la même taille pour faciliter notre travail :

```python
import cv2
import os

def resize_images_in_folder(input_folder, output_folder, size=(500, 500)):
    # Vérifier si le dossier de sortie existe, sinon le créer
    if not os.path.exists(output_folder):
        os.makedirs(output_folder)

    for filename in os.listdir(input_folder):
        if filename.endswith((".jpg", ".jpeg", ".png")):
            # Charger l'image
            img_path = os.path.join(input_folder, filename)
            image = cv2.imread(img_path)

            # Redimensionner l'image
            resized_image = cv2.resize(image, size)

            # Enregistrer l'image redimensionnée dans le dossier de sortie
            output_path = os.path.join(output_folder, filename)
            cv2.imwrite(output_path, resized_image)
            print(f"Image {filename} redimensionnée et enregistrée dans {output_path}")


input_folder = "C:\\Users\\User\\OneDrive\\Documents\\M1 ISII\\TP IA\\Datasets\\PNEUMONIA"
output_folder = "C:\\Users\\User\\OneDrive\\Documents\\M1 ISII\\TP IA\\Datasets\\PNEUMONIA"
resize_images_in_folder(input_folder, output_folder, size=(500, 500))
```

- On na revérifier avec le code précédent leurs tailles pour confirmer que cela a bien fonctionné :

```
Toutes les images ont la même taille : (500, 500)
```

4- <u>Les méthodes d'extraction d'images :</u>

Ce sont des techniques utilisées pour analyser, transformer des images et extraire des informations significatives ou distinctives d'une image. Chacune de ces méthodes à ses propres caractéristiques et applications :

1. **Gabor** : Un filtre linéaire dans de nombreuses applications de traitement d'images pour la détection des contours, l'analyse de texture, l'extraction de caractéristiques.
2. **DCT (Discrete Cosine Transform)** : Compression d'images comme dans le format JPEG
3. **PHOG (Pyramid Histogram of Oriented Gradient)** : Pour capturer les caractéristiques de forme et de contour.
4. **Transformée de Fourier** : Pour analyser les fréquences de l'image.

- **Gabor** et **PHOG** sont souvent utilisés pour extraire des caractéristiques liées aux textures et à la forme.
- **Fourier** et **DCT** sont idéaux pour analyser la fréquence, la compression, et le filtrage.

Note : Pour les utiliser en python il faut utiliser la bibliothèque OpenCV.

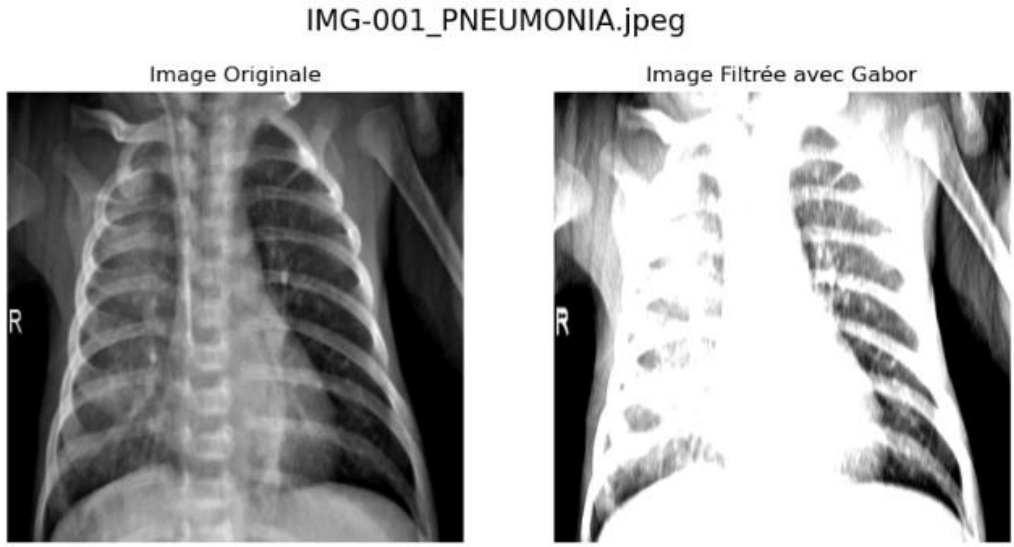
Gabor :

Ce code permet d'appliquer un filtre de Gabor pour extraire les caractéristiques de chaque image des deux dossiers NORMAL et PNEUMONIA. Il affiche également les images originales et filtrées pour comparaison, ainsi que le vecteur des caractéristiques correspondant :

(Avant cela, nous avons essayé d'appliquer l'algorithme de la méthode Gabor sur une seule image, puis sur l'ensemble du dossier)

```python
import cv2
import numpy as np
import os
import matplotlib.pyplot as plt

# Fonction pour créer plusieurs filtres de Gabor
def build_gabor_kernels():
    kernels = []
    ksize = 31  # Taille du noyau
    for theta in np.arange(0, np.pi, np.pi / 4):  # Différentes orientations
        for sigma in (1, 3):  # Différentes largeurs de filtre
            for lamda in np.arange(np.pi / 4, np.pi, np.pi / 4):  # Différentes fréquences
                kernel = cv2.getGaborKernel((ksize, ksize), sigma, theta, lamda, 0.5, 0, ktype=cv2.CV_32F)
                kernels.append(kernel)
    return kernels

# Fonction pour appliquer les filtres de Gabor et construire le vecteur de caractéristiques
def apply_gabor_filters(image, kernels, filename):
    features = []
    filtered_img_to_display = None  # Stocke une seule image filtrée pour l'affichage

    # Application de chaque filtre de Gabor
    for i, kernel in enumerate(kernels):
        filtered_img = cv2.filter2D(image, cv2.CV_8UC3, kernel)

        # Calcul de l'intensité moyenne pour chaque image filtrée (comme caractéristique)
        mean_val = filtered_img.mean()
        features.append(mean_val)

        # Enregistre la première image filtrée pour l'affichage
        if i == 0:
            filtered_img_to_display = filtered_img

    # Affichage de l'image originale et d'une image filtrée (la première)
    fig, axes = plt.subplots(1, 2, figsize=(10, 5))
    fig.suptitle(f" {filename}", fontsize=16)

    # Image originale
    axes[0].imshow(image, cmap='gray')
    axes[0].set_title("Image Originale")
    axes[0].axis('off')

    # Première image filtrée
    axes[1].imshow(filtered_img_to_display, cmap='gray')
    axes[1].set_title("Image Filtrée avec Gabor")
    axes[1].axis('off')

    plt.show()

    return features

# Chargement et traitement des images dans un dossier
def process_images_in_folder(folder_path):
    kernels = build_gabor_kernels()  # Création de plusieurs noyaux de Gabor
    for filename in os.listdir(folder_path):
        if filename.endswith((".jpg", ".jpeg", ".png")):
            # Charger l'image en niveaux de gris
            img_path = os.path.join(folder_path, filename)
            img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)

            # Extraction des caractéristiques Gabor et affichage de l'image filtrée
            features = apply_gabor_filters(img, kernels, filename)

            # Affichage du vecteur de caractéristiques complet
            print(f"Vecteur de caractéristiques pour {filename} :")
            print(features)
            print("\n" + "="*50 + "\n")

folder_path = "C:\\Users\\User\\OneDrive\\Documents\\M1 ISII\\TP IA\\Datasets\\PNEUMONIA"
process_images_in_folder(folder_path)
```

Résultat :

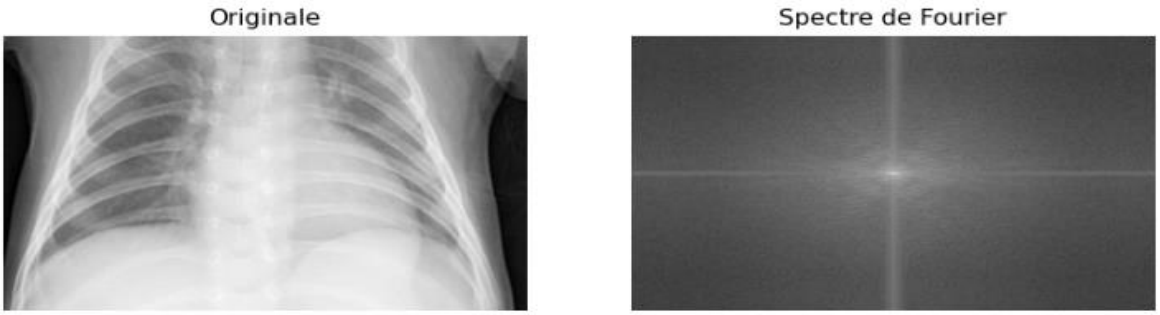



```
Vecteur de caractéristiques pour IMG-001_PNEUMONIA.jpeg :
[190.028636, 85.06904, 34.634376, 12.200924, 8.477016, 7.891268, 224.73116, 0.89894, 32.77742, 194.305984, 1.548296, 3.126956,
190.545112, 84.974812, 34.44172, 6.68802, 4.232444, 3.188256, 224.315292, 0.899804, 32.802832, 190.791304, 1.533176, 3.189196]
```

Transformation de fourrier :

```python
import cv2
import numpy as np
import os
import matplotlib.pyplot as plt

# Fonction pour calculer et afficher la transformée de Fourier
def apply_fourier_transform(image, filename):
    f = np.fft.fft2(image)
    fshift = np.fft.fftshift(f)
     magnitude_spectrum = 20 * np.log(np.abs(fshift) + 1)

    # Affichage de l'image originale et du spectre de Fourier
    fig, axes = plt.subplots(1, 2, figsize=(10, 5))
    fig.suptitle(f"Image originale et spectre de Fourier pour {filename}", fontsize=16)

    axes[0].imshow(image, cmap='gray')
    axes[0].set_title("Originale")
    axes[0].axis('off')
    axes[1].imshow(magnitude_spectrum, cmap='gray')
    axes[1].set_title("Spectre de Fourier")
    axes[1].axis('off')
    plt.show()

    # Extraction des caractéristiques valeurs de haute fréquence
    num_features = 20  # Nombre de caractéristiques à extraire
    rows, cols = fshift.shape
    crow, ccol = rows // 2, cols // 2
    features = np.abs(fshift[crow - 5:crow + 5, ccol - 5:ccol + 5]).flatten()
    features = np.sort(features)[-num_features:]
    return features

def process_images_in_folder(folder_path):
    for filename in os.listdir(folder_path):
        if filename.endswith((".jpg", ".jpeg", ".png")):
            img_path = os.path.join(folder_path, filename)
            img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)
            features = apply_fourier_transform(img, filename)

            print(f"Vecteur de caractéristiques pour {filename} :")
            print(features)
            print("\n" + "="*50 + "\n")

folder_path = "C:\\Users\\User\\OneDrive\\Documents\\M1 ISII\\TP IA\\Datasets\\PNEUMONIA"
process_images_in_folder(folder_path)
```

Résultat :




```
Vecteur de caractéristiques pour IMG-002_PNEUMONIA.jpeg :
[ 1552432.57502088  1567489.42097962  1616874.85929386  1616874.85929386
  1635663.47557813  1933856.09290903  1933856.09290903  2475917.82175835
  2475917.82175835  2674204.23108318  2674204.23108318  3312161.92466505
  3312161.92466505  4862207.08836378  4862207.08836378  6244980.89701461
  6244980.89701461  8930712.2196461   8930712.2196461   56990489.        ]
```

Ainsi nous avons commencé avec ses deux méthodes, ensuite on a combiné les quatre méthodes :

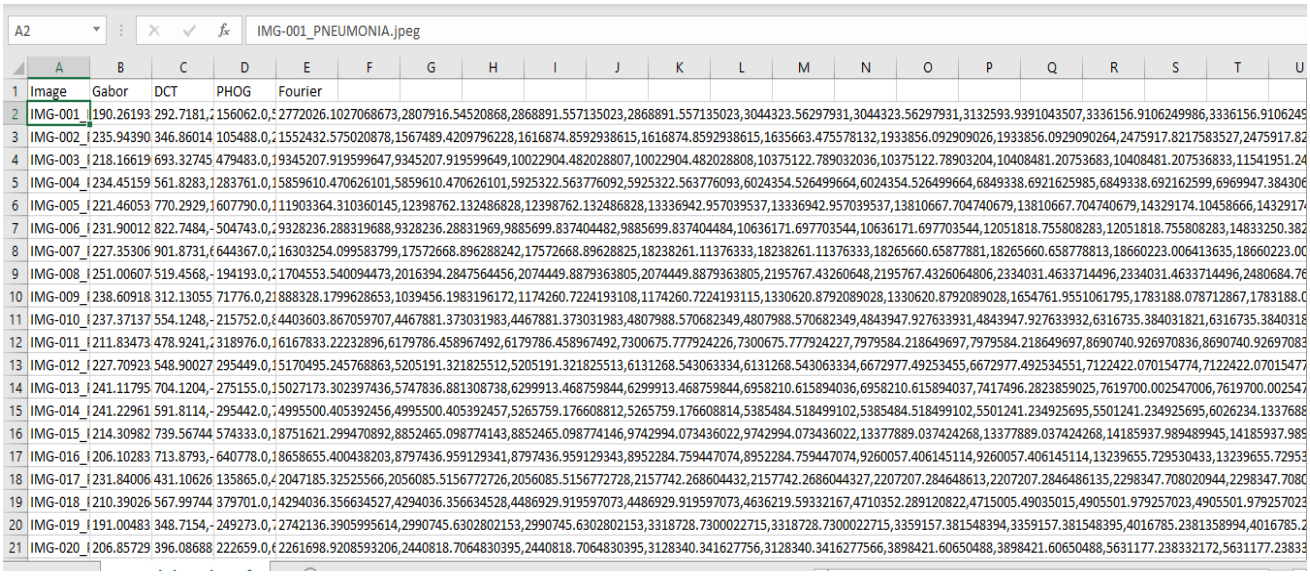
Ce code permet de charger les images à partir du dossier extraire les caractéristiques, visualiser les résultats et stocker les vecteurs dans un fichier csv

```python
import cv2
import numpy as np
import os
import pandas as pd
import matplotlib.pyplot as plt

# Extraction des caractéristiques Gabor
def gabor_features(image):
    kernels = []
    ksize = 31
    for theta in np.arange(0, np.pi, np.pi / 4):
        for sigma in (1, 3):
            for lamda in np.arange(np.pi / 4, np.pi, np.pi / 4):
                kernel = cv2.getGaborKernel((ksize, ksize), sigma, theta, lamda, 0.5, 0, ktype=cv2.CV_32F)
                kernels.append(kernel)
    features = []
    for i, kernel in enumerate(kernels):
        filtered_img = cv2.filter2D(image, cv2.CV_8UC3, kernel)
        features.append(filtered_img.mean())

    # Afficher l'image filtrée (premier filtre)
    if kernels:
        plt.imshow(cv2.filter2D(image, cv2.CV_8UC3, kernels[0]), cmap='gray')
        plt.title("Filtre Gabor (premier)")
        plt.axis('off')
        plt.show()
    return features

# Extraction des caractéristiques DCT
def dct_features(image):
    dct_image = cv2.dct(np.float32(image) / 255.0)
    dct_flat = dct_image.flatten()
    # Normaliser et visualiser le DCT
    dct_image_normalized = cv2.normalize(dct_image, None, 0, 255, cv2.NORM_MINMAX).astype(np.uint8)
    plt.imshow(dct_image_normalized, cmap='gray')
    plt.title("Coefficients DCT")
    plt.axis('off')
    plt.show()
    num_features = 20
```

```
    return dct_flat[:num_features]

# Extraction des caractéristiques PHOG
def phog_features(image, bins=8):
    gx = cv2.Sobel(image, cv2.CV_32F, 1, 0, ksize=1)
    gy = cv2.Sobel(image, cv2.CV_32F, 0, 1, ksize=1)

    # Calculer la magnitude et l'angle
    magnitude, angle = cv2.cartToPolar(gx, gy, angleInDegrees=True)

    # Afficher le gradient
    plt.imshow(magnitude, cmap='gray')
    plt.title("Gradient Magnitude")
    plt.axis('off')
    plt.show()

    hist = cv2.calcHist([angle], [0], None, [bins], [0, 360])
    hist = hist.flatten()
    return hist

# Extraction des caractéristiques de Fourier
def fourier_features(image):
    f = np.fft.fft2(image)
    fshift = np.fft.fftshift(f)
    magnitude_spectrum = 20 * np.log(np.abs(fshift) + 1)

    plt.imshow(magnitude_spectrum, cmap='gray')
    plt.title("Spectre de Fourier")
    plt.axis('off')
    plt.show()

    num_features = 20
    rows, cols = fshift.shape
    crow, ccol = rows // 2, cols // 2
    features = np.abs(fshift[crow - 5:crow + 5, ccol - 5:ccol + 5]).flatten()
    features = np.sort(features)[-num_features:]
    return features
```

```
def process_images_in_folder(folder_path, csv_filename):
    results = []
    for filename in os.listdir(folder_path):
        if filename.endswith((".jpg", ".jpeg", ".png")):
            img_path = os.path.join(folder_path, filename)
            img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)
            print(f"\nTraitement de l'image : {filename}")S
            # Gabor
            print("Caractéristiques Gabor :")
            gabor_feats = gabor_features(img)
            print(gabor_feats)

            # DCT
            print("Caractéristiques DCT :")
            dct_feats = dct_features(img)
            print(dct_feats)

            # PHOG
            print("Caractéristiques PHOG :")
            phog_feats = phog_features(img)
            print(phog_feats)

            # Fourier
            print("Caractéristiques Fourier :")
            fourier_feats = fourier_features(img)
            print(fourier_feats)

            # Sauvegarder les résultats
            results.append({
                'Image': filename,
                'Gabor': gabor_feats,
                'DCT': dct_feats,
                'PHOG': phog_feats,
                'Fourier': fourier_feats})

# Convertir les résultats en DataFrame et sauvegarder en CSV
df = pd.DataFrame(results)
df.to_csv(csv_filename, index=False)
```

Exemple du fichier csv :



A2 | IMG-001_PNEUMONIA.jpeg

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	Image	Gabor	DCT	PHOG	Fourier																
2	IMG-001_	190.26193	292.7181,2	156062.0,5	2772026.1027068673,2807916.54520868,2868891.557135023,2868891.557135023,3044323.56297931,3044323.56297931,3132593.9391043507,3336156.9106249986,3336156.910624																
3	IMG-002_I	235.94390	346.86014	105488.0,2	1552432.575020878,1567489.4209796228,1616874.8592938615,1616874.8592938615,1635663.475578132,1933856.092909026,1933856.0929090264,2475917.8217583527,2475917.82																
4	IMG-003_I	218.16619	693.32745	479483.0,1	9345207.919599647,9345207.919599649,10022904.482028807,10022904.482028808,10375122.789032036,10375122.78903204,10408481.20753683,10408481.207536833,11541951.24																
5	IMG-004_I	234.45159	561.8283,1	283761.0,1	5859610.470626101,5859610.470626101,5925322.563776092,5925322.563776093,6024354.526499664,6024354.526499664,6849338.6921625985,6849338.692162599,6969947.384306																
6	IMG-005_I	221.46053	770.2929,1	607790.0,1	11903364.310360145,12398762.132486828,12398762.132486828,13336942.957039537,13336942.957039537,13810667.704740679,13810667.704740679,14329174.10458666,1432917																
7	IMG-006_I	231.90012	822.7484,-	504743.0,2	9328236.288319688,9328236.28831969,9885699.837404482,9885699.837404484,10636171.697703544,10636171.697703544,12051818.755808283,12051818.755808283,14833250.382																
8	IMG-007_I	227.35306	901.8731,(	644367.0,2	16303254.099583799,17572668.896288242,17572668.89628825,18238261.11376333,18238261.11376333,18265660.65877881,18265660.658778813,18660223.006413635,18660223.00																
9	IMG-008_I	251.00607	519.4568,-	194193.0,2	1704553.540094473,2016394.2847564456,2074449.8879363805,2074449.8879363805,2195767.43260648,2195767.4326064806,2334031.4633714496,2334031.4633714496,2480684.76																
10	IMG-009_I	238.60918	312.13055	71776.0,21	888328.1799628653,1039456.1983196172,1174260.7224193108,1174260.7224193115,1330620.8792089028,1330620.8792089028,1654761.9551061795,1783188.078712867,1783188.0																
11	IMG-010_I	237.37137	554.1248,-	215752.0,8	4403603.867059707,4467881.373031983,4467881.373031983,4807988.570682349,4807988.570682349,4843947.927633931,4843947.927633932,6316735.384031821,6316735.3840318																
12	IMG-011_I	211.83473	478.9241,2	318976.0,1	6167833.22232896,6179786.458967492,6179786.458967492,7300675.777924226,7300675.777924227,7979584.218649697,7979584.218649697,8690740.926970836,8690740.92697083																
13	IMG-012_I	227.70923	548.90027	295449.0,1	5170495.245768863,5205191.321825512,5205191.321825513,6131268.543063334,6131268.543063334,6672977.49253455,6672977.492534551,7122422.070154774,7122422.07015477																
14	IMG-013_I	241.11795	704.1204,-	275155.0,1	5027173.302397436,5747836.881308738,6299913.468759844,6299913.468759844,6958210.615894036,6958210.615894037,7417496.2823859025,7619700.002547006,7619700.002547																
15	IMG-014_I	241.22961	591.8114,-	295442.0,1	4995500.405392456,4995500.405392457,5265759.176608812,5265759.176608814,5385484.518499102,5385484.518499102,5501241.234925695,5501241.234925695,6026234.1337688																
16	IMG-015_I	214.30982	739.56744	574333.0,1	8751621.299470892,8852465.098774143,8852465.098774146,9742994.073436022,9742994.073436022,13377889.037424268,13377889.037424268,14185937.989489945,14185937.989																
17	IMG-016_I	206.10283	713.8793,-	640778.0,1	8658655.400438203,8797436.959129341,8797436.959129343,8952284.759447074,8952284.759447074,9260057.406145114,9260057.406145114,13239655.729530433,13239655.72953																
18	IMG-017_I	231.84006	431.10626	135865.0,4	2047185.32525566,2056085.5156772726,2056085.5156772728,2157742.268604432,2157742.2686044327,2207207.284648613,2207207.2846486135,2298347.708020944,2298347.7080																
19	IMG-018_I	210.39026	567.99744	379701.0,1	4294036.356634527,4294036.356634528,4486929.919597073,4486929.919597073,4636219.59332167,4710352.289120822,4715005.49035015,4905501.979257023,4905501.979257023																
20	IMG-019_I	191.00483	348.7154,-	249273.0,2	2742136.3905995614,2990745.6302802153,2990745.6302802153,3318728.7300022715,3318728.7300022715,3359157.381548394,3359157.381548395,4016785.2381358994,4016785.2																
21	IMG-020_I	206.85729	396.08688	222659.0,6	2261698.9208593206,2440818.7064830395,2440818.7064830395,3128340.341627756,3128340.3416277566,3898421.60650488,3898421.60650488,5631177.238332172,5631177.23833																