

Embeddings for Medical Image Retrieval (EMIR)

By: Hernan Razo, Ali Colak, Rajpal Chowdhary

Problem Statement

- Companies need to process large amounts of images daily
- Dealing with .jpg, .png, etc... file types can be tedious and time consuming
- Companies must be wary of patient privacy laws

Solution

- Convert raw images into embedding representations using a neural network that still contain the original image information
- Embeddings can be processed for common tasks such as clustering, classification, and image retrieval
- Dimensionality reduction of embeddings enables faster and more memory efficient operations on image data

Implementation Details

- Gather example dataset from the internet
- Implement a ResNet18 neural network with final classification layers removed
- Store the embeddings created by the layers of the neural network
- Translate all embeddings into a lower dimensional latent space using principal component analysis
- Plot all points to retrieve clusters

Example Dataset



Replit Integration

Replit interface showing a Python script for image embeddings and a scatter plot visualization.

Files: main.py, data.json, gpu.py, scatter_plot.png, poetry.lock, pyproject.toml

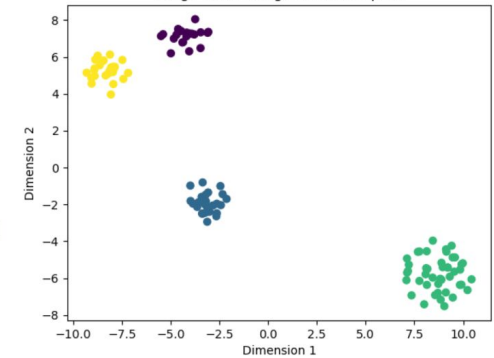
Tools: Ghostwriter, Deployments, Authentication, Chat, Code Search, Console, Database, Debugger, Docs

Code (main.py):

```
1 import os
2 import json
3 import numpy as np
4 import matplotlib.pyplot as plt
5 from sklearn.manifold import TSNE
6
7 def main():
8
9     # Read the JSON file
10     with open('data.json', 'r') as f:
11         data = json.load(f)
12
13     # Extract file paths, labels, and embeddings from the JSON data
14     file_paths = []
15     labels = []
16     embeddings = []
17
18     for file_path, value in data.items():
19         file_paths.append(file_path)
20         labels.append(value['label'])
21         embeddings.append(value['embedding'][0])
22
23     labels = np.array(labels)
24     embeddings = np.array(embeddings)
25
26     tsne = TSNE(n_components=2, random_state=42)
27     embeddings_tsne = tsne.fit_transform(embeddings)
28
29     # Create a scatter plot of the embeddings
30     plt.scatter(embeddings_tsne[:, 0], embeddings_tsne[:, 1], c=labels)
31     plt.title('Image Embeddings in Latent Space')
32     plt.show()
```

Output (Figure 1):

Image Embeddings in Latent Space



Console:

```
> Console x Shell x +
```

Final Latent Space Graph

