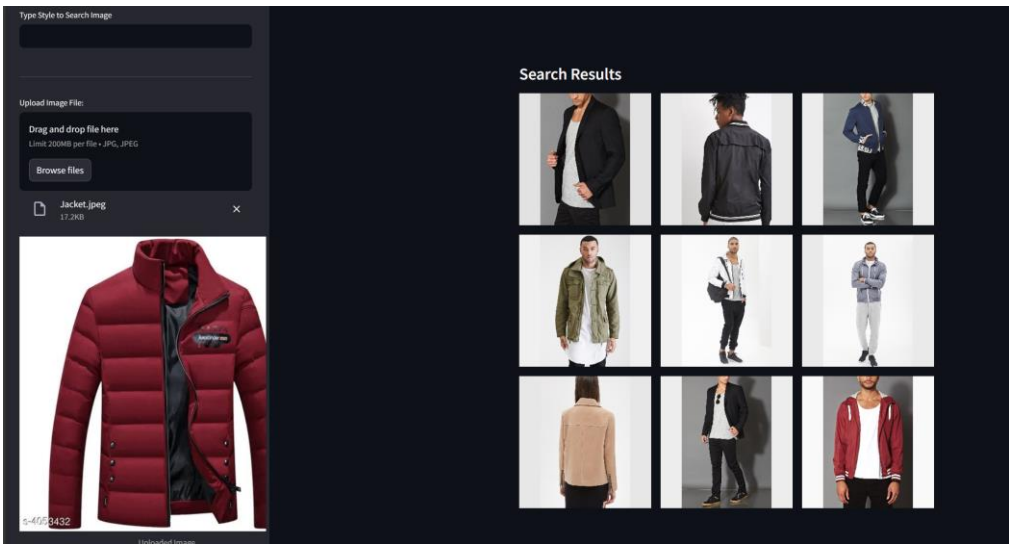
Solution Documentation: Clothing Item Similarity Search

1. Introduction

This document provides an in-depth guide to understanding the Clothing Item Similarity Search application, covering its functionality, architecture, setup process, and execution of training experiments.

2. Problem Statement

The Clothing Item Similarity Search application aims to address the challenge of finding similar clothing items based on uploaded images or typing text. Users often struggle to find clothing items that closely resemble those they have in mind. This application seeks to provide a solution by leveraging deep learning techniques to analyze patterns in images and retrieve similar items from a database.



3. Solution Overview

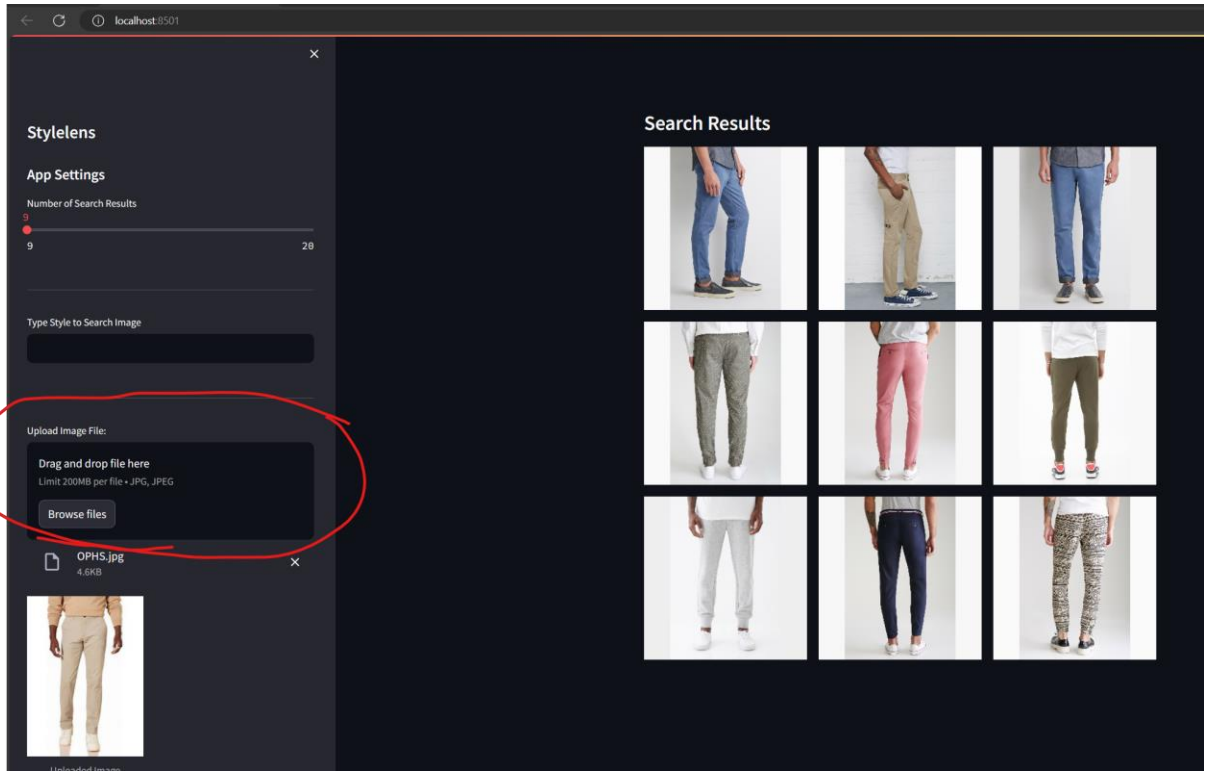
The application leverages sophisticated ResNet50 vision models trained on the ImageNet dataset to extract features from images and conduct comparisons with other images in its database. Specifically, ResNet50 undergoes fine-tuning using the

in-store DeepFashion dataset, optimizing the model for the precise task of clothing item similarity search. Furthermore, in addition to conducting similarity searches based on images, the application integrates text-based search functionality utilizing Sentence Transformer, specifically the all-MiniLM-L6-v2 model. Additionally, to streamline the search process, images undergo a process of hashing into high-dimensional binary codes using Locality Sensitive Hashing (LSH). This methodology enables efficient indexing and retrieval of visually similar images from the database.

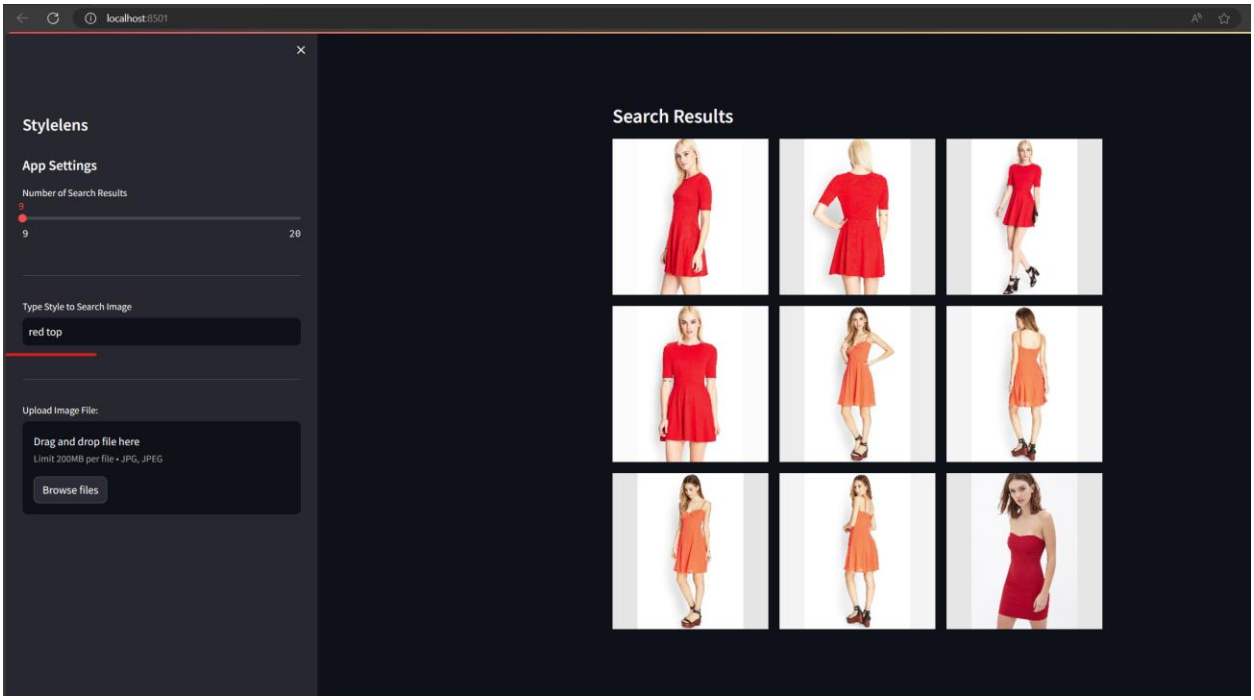
4. Key Features

Upload Image:

Users have the option to upload images of clothing items either by dragging and dropping them or by browsing images from a local folder through the application's user interface. Additionally, users can input text such as color or dress type to obtain similar results.



Text based search:



5. System Architecture

Frontend: The frontend is developed using Streamlit, providing a user-friendly interface for uploading images. Users can easily upload images of clothing items from their wardrobe using this interface.

Model: The similarity search model is based on ResNet50, a convolutional neural network architecture pre-trained on ImageNet. The model is fine-tuned by training only the last few layers, specifically the 5th layer, to adapt it to the clothing item similarity search task. By fine-tuning the model, it learns to extract more relevant features from clothing images, improving the accuracy of the similarity search.

Hyperparameters

num_epochs = 10

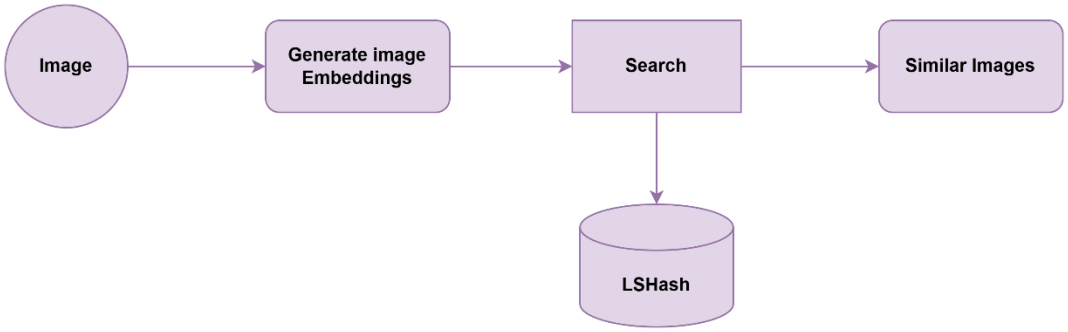
batch_size = 16

learning_rate = 0.001

checkpoint_interval = 10000

Image Embedding: To efficiently compare uploaded images with those in the database, Locality Sensitive Hashing (LSH) is employed for image hashing. LSH allows for approximate nearest neighbor search, enabling fast retrieval of similar images based on their hash codes. During query time, we start with an input image. We then utilize the same embedding to extract a feature vector and perform a search, also considering cosine similarity.

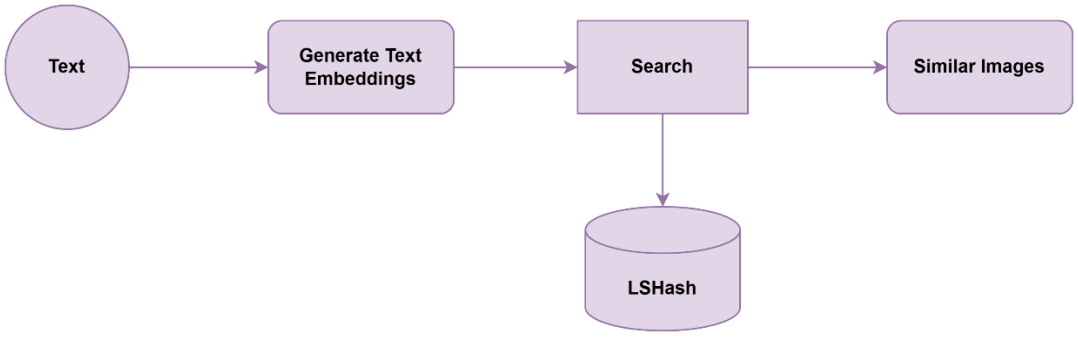
Image Search:



Text Embedding: The solution involves utilizing the Sentence Transformer model, specifically the all-MiniLM-L6-v2 variant, to generate text embeddings. This model takes input text and converts it into high-dimensional numerical vectors, capturing the semantic meaning of the text. These embeddings can then be used for various natural language processing tasks such as similarity search, classification, and clustering. The process involves feeding the text data into the Sentence Transformer model, which transforms it into embeddings that represent the underlying meaning of the text. These embeddings can then be

used to compare similarity between texts or to perform other downstream tasks.

Test Search:



6. Execution of Training Experiments

Data Preprocessing: Preprocess the DeepFashion dataset for training, including image resizing, normalization.

```
self.preprocess = transforms.Compose([
    transforms.Resize(224),
    #transforms.CenterCrop(224),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
])
```

Model Training: The model is fine-tuned by training only the last few layers, specifically the 5th layer, to adapt it to the clothing item similarity search task. By fine-tuning the model, it learns to extract more relevant features from clothing images, improving the accuracy of the similarity search.

```
# Load pre-trained ResNet50
resnet50 = torchvision.models.resnet50(pretrained=True)

# Freeze all layers except the 5th layer (fc layer)
for param in resnet50.parameters():
    param.requires_grad = False
for param in resnet50.layer4.parameters():
    param.requires_grad = True
```

Hyperparameters

num_epochs = 10

batch_size = 16

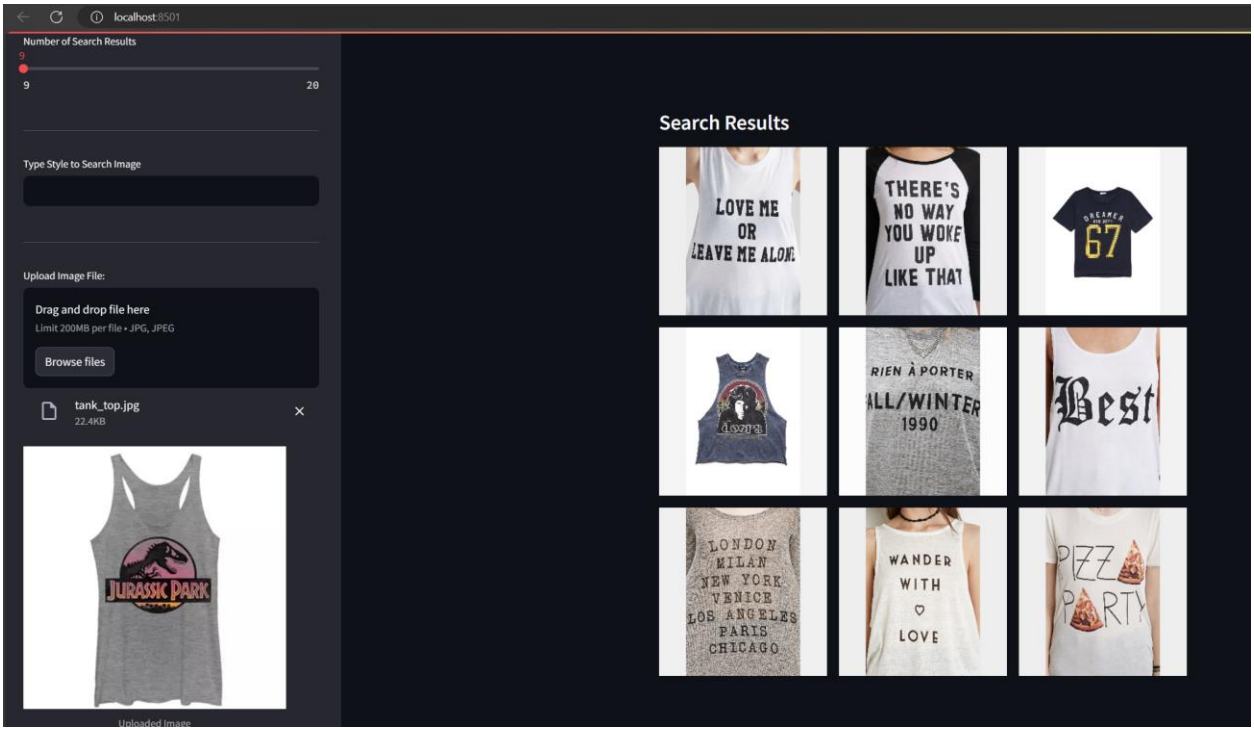
learning_rate = 0.001

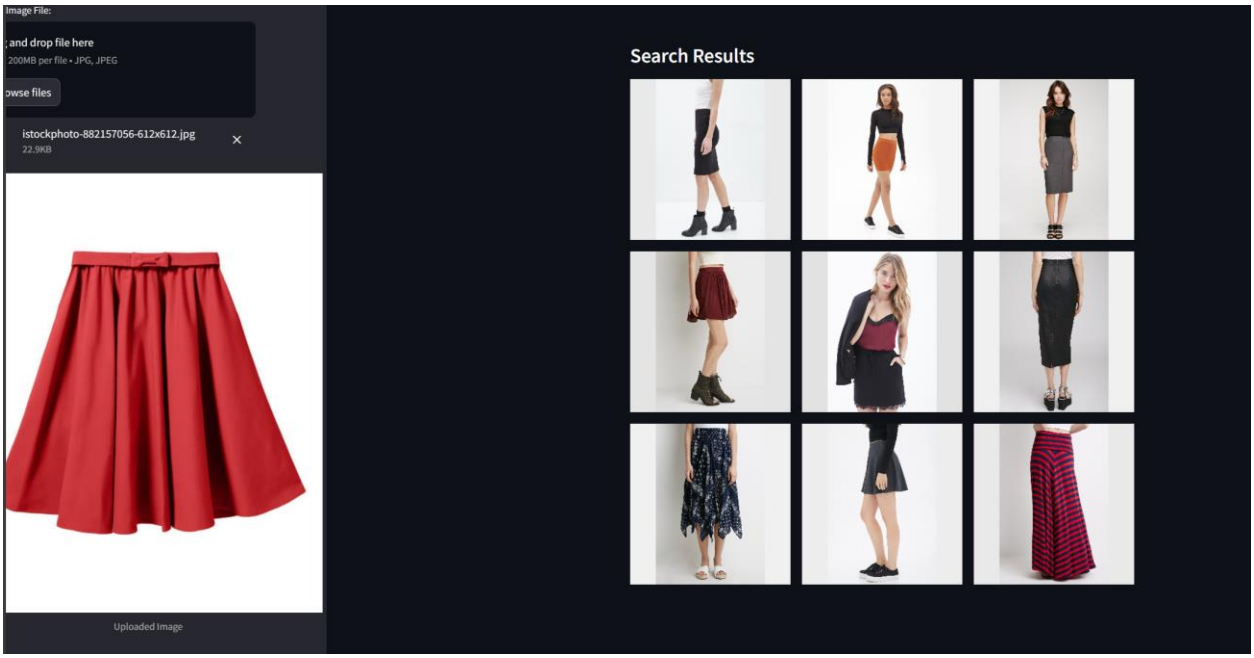
checkpoint_interval = 10000

Here is model training result: after 10 epochs model training loss is 0.0766

Evaluation: Evaluate the trained models using appropriate evaluation metrics such as accuracy. Additionally, cross-entropy loss is employed as a validation metric to assess the performance of the model during training and validation phases. During training, the model's validation loss is observed to be 1.5788, indicating room for improvement. While running more epochs could potentially decrease the validation loss, hardware limitations prevent further training. However, manual validation using the user interface yields promising results.

Example:





Optimization: In the optimization phase, we focus on refining our models and adjusting settings to improve performance. We use the Adam optimizer with a

learning rate of 0.001 to help fine-tune our models effectively. We carefully adjust parameters like batch size and regularization techniques. Through experimentation, we validate the chosen settings to ensure our models perform optimally.

7. Setup Process

To set up the application, follow the instructions outlined in the setup documentation. This involves installing necessary dependencies, configuring the environment, and deploying the application on your preferred platform. Additionally, detailed steps can be found in the readme.md document for further guidance.

8. Conclusion

The Clothing Item Similarity Search application offers a novel solution to the challenge of finding similar clothing items based on uploaded images. By leveraging deep learning techniques and the DeepFashion dataset, the application provides users with an intuitive and efficient way to discover clothing items that match their preferences.

9. Limitations:

- The accuracy of similarity search results may be affected by the quality of input images or textual descriptions.
- Currently, the application utilizes two separate embeddings. It may be beneficial to explore models such as CLIP, where both image and text embeddings can be stored together.

10. During implementation, I encountered several challenges, including:

- Data Quality: Ensuring the quality and diversity of the training data, especially in terms of image and textual descriptions, posed a significant challenge.

- Model Complexity: Managing the complexity of deep learning models like ResNet50 and Sentence Transformer required careful tuning of hyperparameters and optimization strategies.
- Computational Resources: Optimizing the utilization of computational resources for training and inference, especially when dealing with large datasets, was another challenge.

11. Future improvements could include:

- Enhanced Data Augmentation: Implementing more sophisticated data augmentation techniques to further diversify the training dataset and improve model generalization.
- Model Ensemble: Exploring ensemble methods to combine predictions from multiple models or variations of the same model for improved performance and robustness.
- User Feedback Integration: Incorporating mechanisms for user feedback to continuously improve the similarity search algorithm and enhance user experience.
- Semantic Understanding: Developing techniques for better semantic understanding of images and textual descriptions to provide more accurate and meaningful similarity search results.
- Scalability: Optimizing the system architecture and algorithms to handle larger datasets and scale to accommodate growing user demands efficiently.

12. References

DeepFashion Dataset: https://mmlab.ie.cuhk.edu.hk/projects/DeepFashion/InShopRetrieval.html

ResNet50 paper: https://arxiv.org/abs/1512.03385

Sentence Transformer: https://www.sbert.net/examples/applications/semantic-search/README.html.