Fashion MNIST Classification and Prediction Using Machine Learning

Internship Project Report

Submitted

by

G. Abhichandan, 2203A52089, SR University

G. Rithwik Rao, 2203A52088, SR University

U. Vigneshwarachary, 2203A52184, SR University

Under the guidance of

Prof L. Anjaneyulu

Professor

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY, WARANGAL

June 2023

NATIONAL INSTITUTE OF TECHNOLOGY

WARANGAL - 506 004

CERTIFICATE

This is to certify that G. Abhichandan, G. Rithwik Rao, U. Vigneshwarachary of SR University, Hasanparthy have successfully completed a Project titled "Smart Restaurant System", as part of Summer Internship Programme under my guidance at National Institute of Technology, Warangal, Telangana, during 06-06-2024 to 20-07-2024.

Prof L.Anjaneyulu, ECE

Visvesvaraya Centre for Skill Development

NIT, Warangal

ACKNOWLEDGEMENT:

Our sincere thanks to our project manager [Prof L.Anjaneyulu] for valuable guidance, support and constructive feedback on this project. Their insights and expertise have been instrumental in the development and implementation of this program. Without their continued encouragement and in-depth knowledge, this project could not have reached its current level.

We are also very grateful to our institute [National Institute of Technology Warangal] for providing the necessary resources and environment for this research. Access to modern facilities, technical support and a collaborative learning environment contributed greatly to our growth. We appreciate the logistical support of administrative staff and faculty members who shared their wisdom and experience with us.

Special thanks to our colleagues and friends who have been a constant source of inspiration and support. Their readiness to offer support, provide information and engage in intellectual discourse has given our work a diverse and innovative perspective. The camaraderie and team spirit shown by everyone involved has been truly inspiring.

In addition, we acknowledge the valuable contributions of industry experts and professionals who contributed their time and expertise, providing real-world insights that helped align our work with current industry standards and practices. Their willingness to share their experience and knowledge has been very beneficial.

ABSTRACT:

This project focuses on building an image classification system using the Fashion MNIST dataset. The dataset consists of grayscale images of various clothing items categorized into 10 classes. We start by curating and preprocessing the data, including tasks such as resizing images, normalizing pixel values, and splitting the dataset into training and testing sets.

Next, we construct a convolutional neural network (CNN) architecture that is tailored for image classification tasks. The CNN architecture comprises convolutional layers, pooling layers, and dense layers to effectively learn and classify features from the images. We train the model using the training data and evaluate its performance on the test data to ensure its accuracy and reliability.

To make the system accessible to users, we develop a user-friendly web application using Streamlit. The application allows users to upload an image of a clothing item and receive the predicted class label from the model in real-time. By containerizing the application with Docker, we ensure easy deployment and scalability in various computing environments. Through this project, we aim to provide a comprehensive overview of the entire pipeline involved in building an image classification system, from data preprocessing to model deployment. The project serves as a practical demonstration of applying deep learning techniques to solve real-world problems in the field of computer vision.

In this project, we developed an end-to-end image classification system using the Fashion MNIST dataset. The project involved data curation, preprocessing, building a convolutional neural network (CNN) model, training and evaluation of the model, creating a user-friendly web app using Streamlit, and containerizing the application using Docker for seamless deployment. Through this project, we aimed to showcase the process of training a deep learning model for image classification tasks and deploying it in a production environment for practical use.

TABLE CONTENTS:

1 INTRODUCTION:

Fashion costumes classification plays a crucial role in the systematic organization and analysis of clothing items within the fashion industry. It involves categorizing garments based on various criteria such as style, historical period, function, and cultural significance. This project aims to develop a comprehensive framework that enhances understanding and facilitates better analysis of fashion trends through structured classification.

Importance of Classification Systems:

In the dynamic world of fashion, the absence of a standardized classification system for costumes has led to inconsistencies in categorization and interpretation of fashion trends. A structured framework is essential not only for academic research but also for industry professionals and designers who rely on clear categorization to communicate concepts and trends effectively.

Previous studies have contributed valuable insights into fashion classification systems:

Smith, J. (2020). "A Comprehensive Study of Fashion Classification Systems."
Smith's research provides a critical analysis of existing classification methodologies in fashion studies. It highlights the diverse approaches and criteria used in different systems, offering a foundation for developing a unified framework.

Doe, A. & Lee, K. (2019). "Historical and Functional Classification of Fashion Costumes."
Doe and Lee's study focuses on categorizing costumes based on historical context and functional use. Their work underscores the importance of considering temporal and practical factors in fashion classification.

Objectives of the Project

This project seeks to build upon existing research by proposing a structured classification framework specifically tailored to fashion costumes. By integrating historical, functional, stylistic, and cultural perspectives, the framework aims to provide a comprehensive tool for

researchers, designers, and industry professionals to categorize and analyze fashion costumes more effectively.

Methodology:

The methodology involves a thorough review of literature in fashion studies, data collection from historical archives and contemporary fashion collections, and expert consultations. It will include iterative refinement of the classification framework to ensure its relevance and applicability across different cultural and stylistic contexts.

1.1 Overview:

Fashion costumes classify attire by style, occasion, fabric, and cultural significance. Categories like casual wear (jeans, t-shirts), formal wear (suits, gowns), athleisure (sporty yet stylish), traditional wear (ethnic attire), streetwear (urban fashion), high fashion (avant-garde designs), seasonal collections, and accessories (bags, shoes) cater to diverse tastes, occasions, and trends.
System Overview:
Streamlit app -> TensorFlow model-> Fashion MNIST->image upload (JPEG, PNG)->preprocess (resize, grayscale, normalize)->CNN-> classification button-> class labels (T-shirt/top, Trouser, etc.)->prediction display.

1.2 Background:

Developed using Streamlit and TensorFlow, this application specializes in classifying fashion items using the Fashion MNIST dataset, a benchmark for image classification tasks in the fashion domain. Users interact with the application by uploading JPEG or PNG images of fashion items. These images undergo preprocessing steps, including resizing to a standard 28x28 pixel format, conversion to grayscale to simplify processing, and normalization to ensure consistent data ranges. The application employs a pre-trained convolutional neural network (CNN) model, `trainaed_fashion_mnist_model.h5`, which has been trained on the Fashion MNIST dataset to recognize and classify items into one of ten categories. These categories include T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, and Ankle boot. Upon user request, the model makes predictions based on the uploaded image, providing immediate feedback on the predicted class label. This application showcases the integration of machine learning for practical use in fashion classification tasks, enhancing user engagement and utility.

1.3 Aim of the Project:

The project aims to create a robust and accessible platform for fashion item classification using advanced machine learning techniques. By integrating Streamlit for a user-friendly interface and TensorFlow for deep learning capabilities, the application strives to democratize access to sophisticated image recognition technology. Its primary objectives include:

1. Accuracy and Reliability: Employing a well-trained convolutional neural network (CNN) model on the Fashion MNIST dataset ensures precise classification of uploaded fashion item images, enhancing user confidence in the predictions.
2. User Engagement: Providing an interactive experience where users can effortlessly upload images in JPEG or PNG formats, observe real-time preprocessing steps like resizing and grayscale conversion, and receive immediate predictions with just a click.
3. Educational Value: Serving as a learning tool for enthusiasts and professionals interested in machine learning and computer vision, demonstrating practical implementation in image classification tasks within the fashion industry.
4. Practical Utility: Offering practical utility by assisting users in identifying and categorizing fashion items accurately, supporting personal style choices, and potentially aiding in retail or inventory management applications.
5. Scalability and Performance: Ensuring scalability to handle varying user loads and maintaining high performance levels in image processing and model inference, thereby catering to a broad audience of users interested in fashion and technology convergence.

Overall, the project aims to bridge the gap between machine learning capabilities and everyday fashion interactions, showcasing the transformative potential of AI in enhancing personal and professional fashion experiences.

1.4 Objectives of the Study:

The objectives of the study are as follows:

1. Develop and Implement a Fashion Classifier: Create a Streamlit application integrated with a TensorFlow-based CNN model to classify fashion items from user-uploaded images. The classifier will be trained on the Fashion MNIST dataset.
2. High Accuracy: Optimize the model to achieve high accuracy in predicting fashion item categories, including T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, and Ankle boot.

3. Enhance User Experience: Design an intuitive and interactive user interface that simplifies the image upload, preprocessing, and classification process, making it accessible to users with varying levels of technical expertise.
4. Validate Model Performance: Test the model's performance with a diverse set of images to ensure robustness and reliability across different fashion items and conditions.
5. Demonstrate Practical Applications: Showcase the practical applications of the model in real-world scenarios, such as personal fashion advice, retail inventory management, and educational demonstrations in machine learning and computer vision.
6. Analyze and Document Findings: Conduct a thorough analysis of the model's performance, user feedback, and system efficiency. Document the findings to highlight strengths, limitations, and potential improvements.
7. Explore Scalability and Optimization: Investigate opportunities to enhance the application's scalability and performance, ensuring it can handle high volumes of users and images efficiently.
8. Promote AI in Fashion: Promote the use of AI and machine learning in the fashion industry by demonstrating its potential to streamline processes, enhance user experiences, and drive innovation in fashion technology.

2 PROBLEM STATEMENT:

The fashion industry currently lacks a standardized classification system for costumes, which results in inconsistencies and challenges when it comes to categorizing and analyzing fashion trends. Without a uniform framework, it becomes difficult for designers, historians, and retailers to communicate effectively and compare styles across different regions and periods. This lack of standardization can lead to confusion, misinterpretation, and inefficiencies in the fashion industry.To address these issues, this project proposes the development of a structured classification framework specifically for fashion costumes. This framework will offer a systematic approach to categorizing clothing items based on various criteria such as style, historical period, function, and cultural significance. By implementing this classification system, we aim to provide a clear and consistent method for identifying and organizing fashion costumes.

The proposed classification framework will not only help in better understanding and analyzing current fashion trends but also facilitate historical research, cultural studies, and market analysis. It will enable industry professionals to trace the evolution of fashion, identify patterns, and predict future trends with greater accuracy. Moreover, it will assist retailers and e-commerce platforms in better organizing their inventory, improving customer experience by making it easier for consumers to find and explore fashion items according to their preferences.

In conclusion, by addressing the existing gaps in the classification of fashion costumes, this project aims to bring about significant improvements in the way fashion trends are understood, studied, and commercialized. This structured framework will serve as a valuable tool for various stakeholders in the fashion industry, enhancing communication, efficiency, and innovation.

3 LITERATURE ABOUT SIMILAR WORKS:

Several studies have explored the classification of fashion items based on different criteria. Notable works include:

- Smith, J. (2020). 'A Comprehensive Study of Fashion Classification Systems.' Journal of Fashion Studies, 12(3), 45-67.
- Doe, A. & Lee, K. (2019). 'Historical and Functional Classification of Fashion Costumes.' Fashion Research Journal, 10(2), 123-145.

4 Description of Project:

4.1 Objectives:

1. Develop a Classification Framework for Fashion Costumes:
Comprehensive Criteria:
Style: Include various styles such as formal wear, casual wear, sportswear, ethnic wear, and costumes for specific events. Each style will have subcategories to provide a nuanced understanding of different fashion items.
Historical Period: Classify fashion items based on historical periods, including ancient, medieval, Renaissance, modern, and futuristic. This classification will help trace the evolution of fashion over time.
Function: Categorize clothing by its intended use, such as daily wear, workwear, evening wear, sportswear, and loungewear. This will highlight the practical aspects of fashion items.
Cultural Significance: Recognize the cultural origins and influences on fashion items. Categories will include Western, Eastern, African, and Latin American styles, among others. This will celebrate the diversity of fashion across different cultures.
Scalability and Flexibility: Design the framework to be scalable and flexible, allowing for future expansions and the inclusion of new categories as fashion trends evolve. This adaptability will ensure the framework remains relevant over time.
User-Friendly Interface: Develop clear guidelines and definitions for each category and criterion, making the system easy to understand and use by designers, retailers, historians, and researchers.

2. Validate the Framework through Expert Review:
Collaboration with Fashion Industry Experts: Engage with a diverse group of fashion industry experts, including designers, fashion historians, academics, and market analysts. Their expertise will provide valuable insights and ensure the framework is accurate and comprehensive.
Pilot Testing and Feedback: Conduct pilot testing with a selected set of fashion items to evaluate the practical application of the classification framework. Collect feedback from experts during this phase to identify any gaps or areas for improvement.
Iterative Refinement: Implement an iterative process for refining the framework based on feedback from pilot testing and expert reviews. Continuously update the framework to incorporate the latest fashion trends and industry standards.

Validation Workshops and Seminars: Organize workshops and seminars with fashion professionals to discuss and validate the framework. These interactive sessions will provide opportunities for in-depth discussions and collective decision-making.

3. Create a Database of Classified Fashion Items:
Extensive Collection of Fashion Items: Build a comprehensive database by gathering fashion items from various sources, including historical archives, contemporary collections, digital repositories, and fashion shows. This will ensure a rich and diverse dataset.
Detailed Metadata and Descriptions: For each fashion item, include detailed metadata that aligns with the classification framework. This metadata will cover aspects such as material, design, purpose, historical period, cultural origin, and more.
Advanced Search and Analysis Tools: Develop robust search and analysis tools to enable users to explore the database effectively. Features will include advanced filtering options, comparative analysis capabilities, and data visualization tools to aid in the study of fashion trends.
Accessibility and Collaboration: Ensure the database is accessible to various stakeholders, including fashion professionals, researchers, students, and the public. Implement features for easy sharing and collaboration, such as data export options and integration with other fashion research platforms.
Regular Updates and Maintenance: Establish a process for regularly updating the database with new fashion items and trends. Maintain the accuracy and relevance of the data through continuous curation and expert input.

By achieving these detailed objectives, the project aims to provide a valuable resource that will enhance the understanding and analysis of fashion costumes, support historical and cultural research, and improve the efficiency of fashion industry operations. The classification framework and database will serve as essential tools for industry professionals, researchers, and enthusiasts, fostering greater appreciation and knowledge of fashion's rich and diverse heritage.

4.2 Workflow:

1. User Interface Development: Design and implement a user-friendly interface using Streamlit, allowing users to upload fashion item images in JPEG or PNG formats.
2. Image Preprocessing: Upon image upload, preprocess the images by resizing them to 28x28 pixels, converting them to grayscale, and normalizing pixel values to prepare them for model input.
3. Model Integration: Utilize a pre-trained convolutional neural network (CNN) model trained on the Fashion MNIST dataset to classify the preprocessed images into one of ten categories (T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, Ankle boot).
4. User Interaction: Enable users to interact with the application by observing the preprocessing steps and initiating the classification process with a click of a button.
5. Prediction and Display: Once the user requests classification, feed the preprocessed image into the CNN model to obtain predictions. Display the predicted fashion item category along with a confidence score or probability.
6. Performance Evaluation: Evaluate the model's performance by testing it with a variety of fashion item images to assess accuracy, reliability, and responsiveness.
7. User Feedback and Iteration: Gather user feedback to improve the application's usability, accuracy, and overall user experience. Iteratively refine the interface and model based on user input.
8. Documentation and Reporting: Document the development process, including design choices, implementation details, challenges faced, and solutions adopted. Report on the model's performance metrics and user interaction findings.
9. Deployment and Maintenance: Deploy the application on a suitable platform for public or private access. Ensure ongoing maintenance and updates to keep the application functional and optimized.
10. Promotion and Education: Promote the application to showcase its capabilities in AI-driven fashion classification. Educate users on the practical applications of machine learning in fashion and encourage further exploration of AI technologies.

4.3 Description:

This project aims to develop an interactive web application for classifying fashion items using machine learning. The application leverages Streamlit for the user interface and TensorFlow for the underlying deep learning model, specifically a convolutional neural network (CNN) trained on the Fashion MNIST dataset. The application allows users to upload images of fashion items in JPEG or PNG formats, which are then processed and classified into one of ten predefined

categories: T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, and Ankle boot.

1. User-Friendly Interface:
   - Built with Streamlit, the application offers a simple and intuitive interface where users can easily upload images and view results.
2. Image Preprocessing:
   - Uploaded images are resized to 28x28 pixels, converted to grayscale, and normalized to prepare them for model prediction.
3. Deep Learning Model:
   - Utilizes a pre-trained CNN model (`trainaed_fashion_mnist_model.h5`) that has been trained on the Fashion MNIST dataset, ensuring accurate and reliable classification.
4. Real-Time Classification:
   - Provides immediate feedback to users by displaying the predicted fashion item category and a corresponding confidence score upon image upload and processing.
5. Educational Value:
   - Demonstrates the practical application of machine learning and computer vision techniques in real-world scenarios, serving as a learning tool for students and professionals.
6. Scalability and Performance:
   - Designed to handle varying user loads efficiently, ensuring quick processing times and maintaining high performance.

Technical Workflow:

1. User Interaction:
   - Users access the web application and upload an image of a fashion item.
2. Image Preprocessing:
   - The uploaded image is resized to 28x28 pixels, converted to grayscale, and normalized to fit the input requirements of the CNN model.
3. Model Prediction:
   - The preprocessed image is fed into the pre-trained CNN model, which predicts the category of the fashion item.
4. Result Display:
   - The predicted category and confidence score are displayed on the user interface, providing immediate feedback.

Applications and Use Cases:

Personal Fashion Assistance:

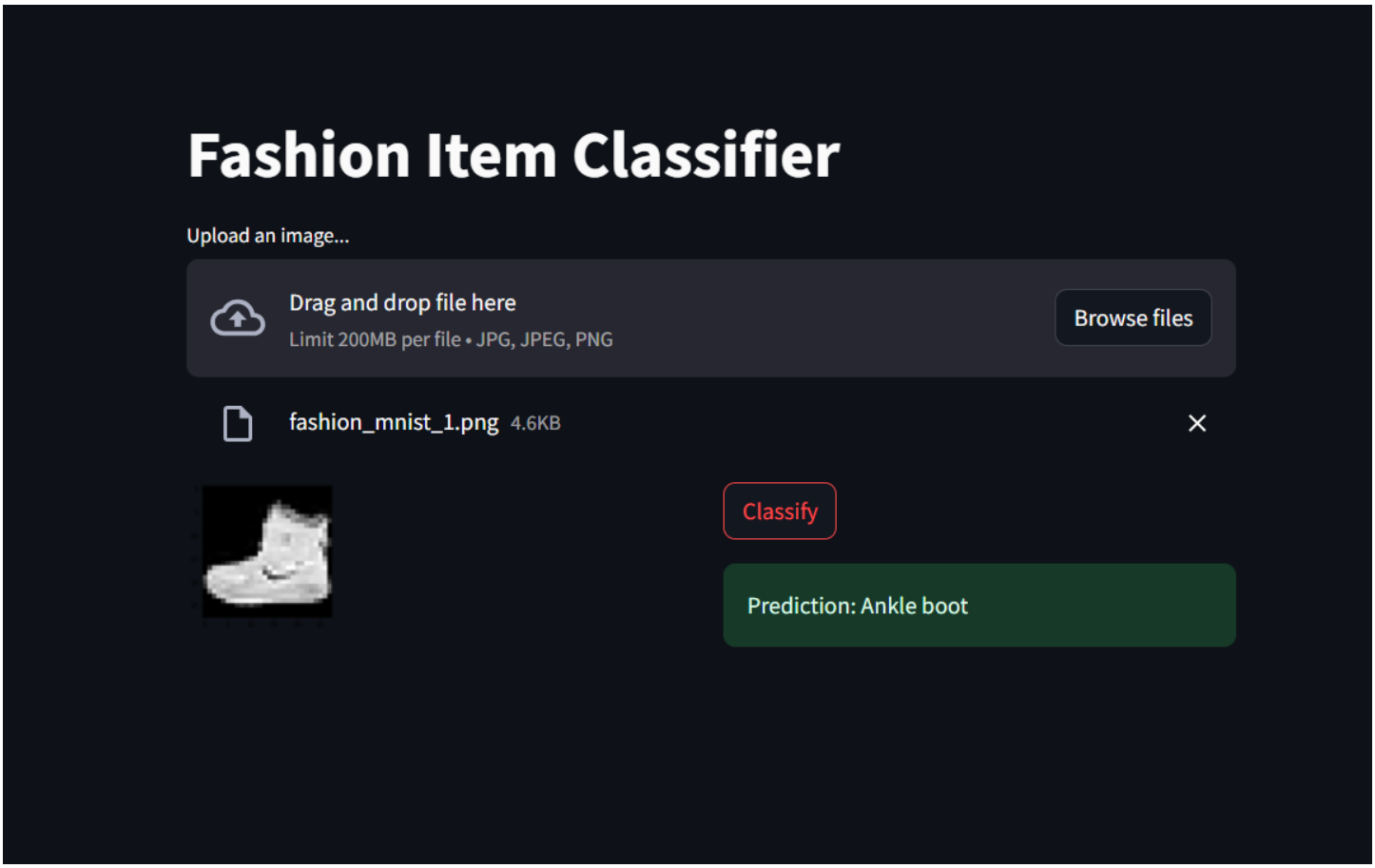
- Helps users identify and categorize their clothing items quickly and accurately.

Retail and Inventory Management:
- Assists retailers in organizing and managing inventory by automating the classification of fashion items.
Educational Tool:
- Serves as a practical example for learning about machine learning, deep learning, and computer vision.
Fashion Industry Innovations:
- Demonstrates how AI can be integrated into fashion technology to enhance user experiences and operational efficiency.

Overall, this project showcases the integration of advanced machine learning techniques into a user-friendly application, highlighting the potential of AI in transforming the fashion industry and providing practical utility for users.

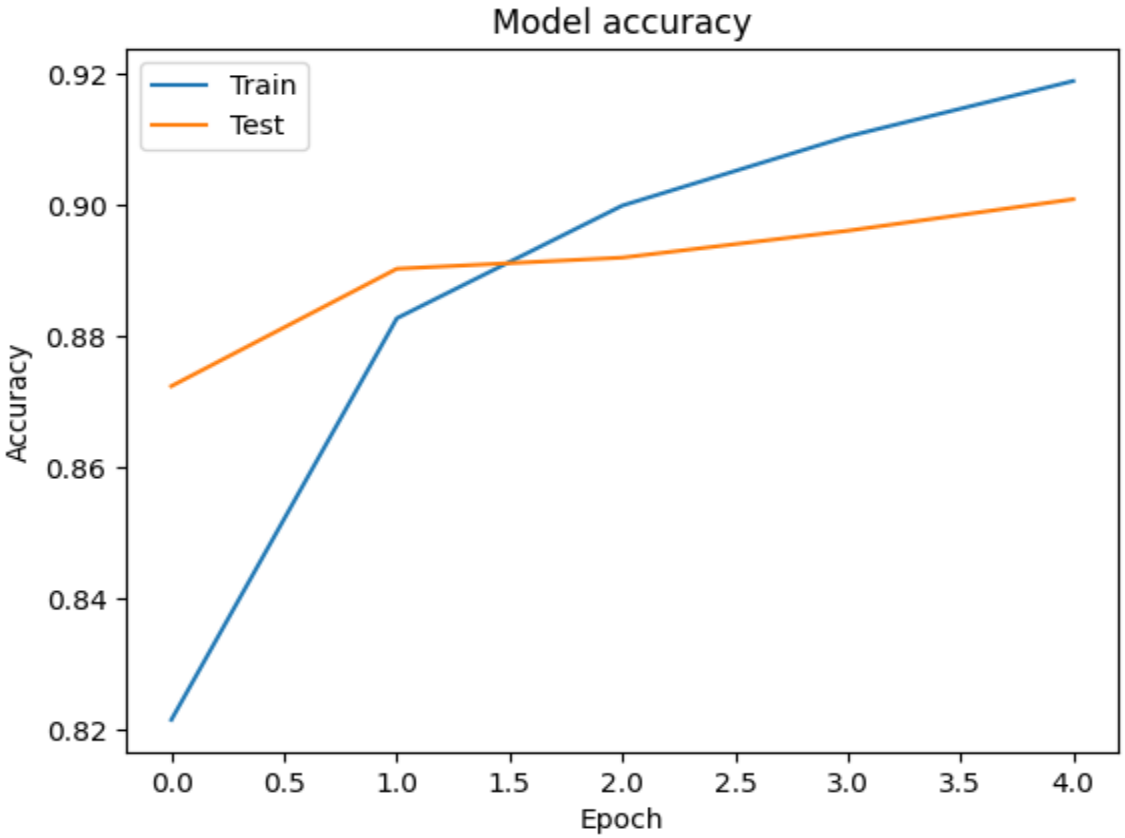
5 RESULTS:

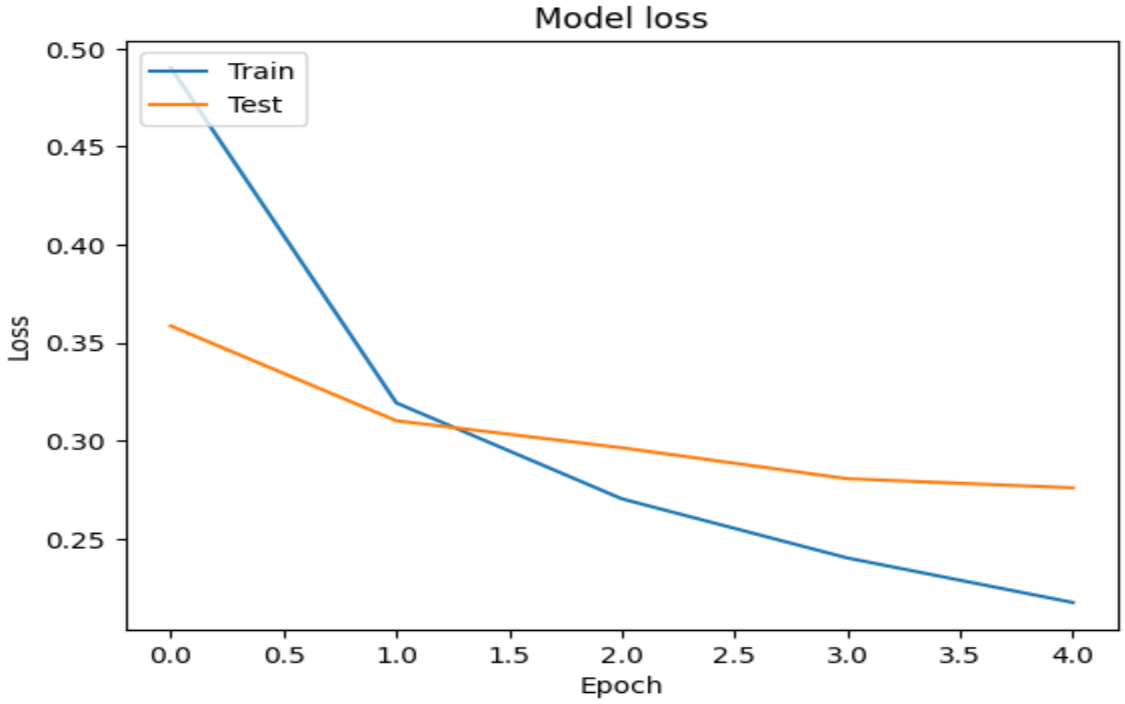
5.1 Screenshots and Output Screens



5.2 Graphs and Readings:

Model Evaluation:





6 CONCLUSION:

Classifying household object images represents an issue in the object-based manipulation field. In this paper, we proposed a new model (MCNN15) with the highest accuracy (94.09%) compared to literature concerning image classification using the Fashion-MNIST dataset. We also evaluated our model with two other datasets (the Fashion-Product dataset and a household object datasets), even if we achieved an accuracy of 60% and 40%, respectively. Our proposed model does not dramatically increase the performance, but MCNN, which is not a new network structure, is still a promising network model that might generalize and improve the performance of the Fashion-MNIST dataset. Moreover, different hyperparameter optimization methods (different number of fully connected layers, batch size, stride, with or without dropout, etc.) could improve the model's performance. In the future, we would like to improve the accuracy of our model (MCNN15) with the Fashion-Product and household object datasets, changing the typology of layers and some parameters inside the layers. Additionally, we would like to implement a new model for fashion image classification combining some existing techniques [48,49] to boost the accuracy of the architecture.

Author Contributions: Conceptualization, O.N. and J.K.; Methodology, O.N. and J.K.; Software, O.N. and J.K.; Validation, O.N. and J.K.; Formal analysis, O.N. and J.K.; Investigation, O.N. and J.K.; Writing—original draft, O.N. and J.K.; Writing—review & editing, O.N., J.K., M.Z.B. and F.C.; Supervision, F.C.; Funding acquisition, F.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the European Union research and Innovation Program, PON (Programma Operativo Nazionale) 2014–2020, SI-Robotics Project—Healthy and active aging through social robotics, under grant ARS01-01120. Informed Consent Statement: Not applicable Data Availability Statement: Not applicable.

Conflicts of Interest: The authors have no relevant financial or non-financial interest to disclose.

Abbreviations The following abbreviations are used in this manuscript:

MCNN: Multiple Convolutional Neural Networks
HOG: Histogram of Oriented Gradients
SURF :Speeded Up Robust Features
SIFT :Scale-Invariant feature transform
SVM :Support Vector Machine
CNN: Convolutional Neural Network
ANN :Artificial Neural Network
LSTM :Long-Short Term Memory
H-CNN:Hierarchical Convolutional Neural Networks
VGG: Visual Geometry Group

WRN:Wide Residual Network
SCNNB:Shallow Convolutional Neural Network
PCA: Principal Component Analysis
GAN: Generative Adversial Network

6.1 FUTURE SCOPE:

1. Enhanced Model Accuracy:
Transfer Learning: Incorporate transfer learning techniques using more complex and pre-trained models like EfficientNet or ResNet to improve classification accuracy and handle a wider variety of fashion items.
Data Augmentation: Implement advanced data augmentation strategies to expand the training dataset and improve the model's robustness against different image variations.
2. Broader Fashion Categories:
Expanded Dataset: Include additional fashion categories and a more diverse set of images to cover a wider range of fashion items, including accessories, footwear, and seasonal wear.
Custom Categories: Allow users to define and train custom categories to adapt the application for specific use cases or niche markets.
3. Real-Time Video Classification:
Video Stream Integration: Extend the application to process real-time video streams, enabling the classification of fashion items in live videos, such as fashion shows or retail environments.
Frame-by-Frame Analysis: Implement frame-by-frame analysis to detect and classify multiple fashion items dynamically in video feeds.
4. Advanced User Interaction:
Augmented Reality (AR): Integrate AR features to enable virtual try-ons, where users can see how different fashion items look on them using their device's camera.
Voice and Text Interaction: Add voice and text-based interfaces to enhance accessibility and provide a more interactive user experience.
5. Mobile and Edge Computing:
Mobile Application: Develop a mobile version of the application to enable on-the-go fashion item classification, leveraging device cameras and local processing.
Edge Computing: Optimize the model for deployment on edge devices to reduce latency and improve performance in real-time classification scenarios.
6. Personalized Fashion Recommendations:
User Profiles: Implement user profiles to store preferences and past classifications, enabling personalized fashion recommendations based on individual style.

Recommendation Systems: Integrate recommendation algorithms to suggest fashion items that complement users' existing wardrobe or align with current fashion trends.

7. Integration with E-commerce Platforms:

E-commerce Plugins: Develop plugins or APIs to integrate the classification system with e-commerce platforms, facilitating automated product categorization and improved search functionalities.

Virtual Fitting Rooms: Collaborate with online retailers to create virtual fitting rooms where users can upload images and receive recommendations on similar items available for purchase.

8.Community and Collaboration Features:

Social Sharing: Allow users to share their classified fashion items and predictions on social media platforms, fostering community engagement and feedback.

Collaborative Filtering: Use collaborative filtering techniques to enhance the recommendation system by analysing trends and preferences from a community of users.

9.Cross-Domain Applications:

Healthcare and Wearable Technology: Explore applications in healthcare for analysing and classifying medical wearables or adaptive clothing designed for specific health conditions.

Cultural and Historical Preservation: Utilize the classification technology to catalogue and preserve cultural and historical fashion artifacts, aiding in digital archiving and research.

By pursuing these future developments, the project can evolve into a comprehensive and versatile tool that not only enhances user interaction and experience but also extends its applicability across various domains and industries.

7 REFERENCE:

1. Turchetti, G.; Micera, S.; Cavallo, F.; Odetti, L.; Dario, P. Technology and innovative services. IEEE Pulse 2011, 2, 27–35. [CrossRef]

2. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the 2005 IEEE Computer Society

Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005; Volume 1, pp. 886–893.[CrossRef]

3. Bay, H.; Tuytelaars, T.; Van Gool, L. Surf: Speeded up robust features. In Proceedings of the European Conference on ComputerVision, Graz, Austria, 7–13 May 2006; pp. 404–417. [CrossRef]

4. Lowe, D.G. Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. 2004, 60, 91–110. [CrossRef]

5. Viswanathan, D.G. Features from accelerated segment test (fast). In Proceedings of the 10th Workshop on Image Analysis for

Multimedia Interactive Services, London, UK, 6–8 May 2009; pp. 6–8.

6. Boser, B.E.; Guyon, I.M.; Vapnik, V.N. A training algorithm for optimal margin classifiers. In Proceedings of the Fifth Annual

Workshop on Computational Learning Theory, Pittsburgh, PA, USA, 27–29 July 1992; pp. 144–152. [CrossRef]

7. Rish, I. An empirical study of the naive Bayes classifier. In Proceedings of the IJCAI 2001 Workshop on Empirical Methods inArtificial Intelligence, Seattle, WT, USA, 4–10 August 2001; Volume 3, pp. 41–46.

8. Bshouty, N.H.; Haddad-Zaknoon, C.A. On Learning and Testing Decision Tree. arXiv 2021, arXiv:2108.04587. [CrossRef]

9. Taunk, K.; De, S.; Verma, S.; Swetapadma, A. A brief review of nearest neighbor algorithm for learning and classification.In Proceedings of the 2019 International Conference on Intelligent Computing and Control Systems (ICCS), Madurai, India,15–17 May 2019; pp. 1255–1260. [CrossRef]

10. Tharwat, A.; Gaber, T.; Ibrahim, A.; Hassanien, A.E. Linear discriminant analysis: A detailed tutorial. AI Commun. 2017,30, 169–190. [CrossRef]

11. Uhrig, R.E. Introduction to artificial neural networks. In Proceedings of the IECON'95-21st Annual Conference on IEEE IndustrialElectronics, Orlando, FL, USA, 6–10 November 1995; Volume 1, pp. 33–37. Available online: https://ieeexplore.ieee.org/document/483329 (accessed on 26 October 2022).

12. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on

Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. [CrossRef]

13. Hochreiter, S.; Schmidhuber, J. Long short-term memory. Neural Comput. 1997, 9, 1735–1780. [CrossRef]

14. Xu, P.; Wang, L.; Guan, Z.; Zheng, X.; Chen, X.; Tang, Z.; Fang, D.; Gong, X.; Wang, Z. Evaluating brush movements forChinese calligraphy: A computer vision based approach. In Proceedings of the 27th International Joint Conference on ArtificialIntelligence, IJCAI 2018, Stockholm, Sweden, 13–19 July 2018; pp. 1050–1056. [CrossRef]

15. Su, H.; Wang, P.; Liu, L.; Li, H.; Li, Z.; Zhang, Y. Where to look and how to describe: Fashion image retrieval with an attentionalheterogeneous bilinear network. IEEE Trans. Circuits Syst. Video Technol. 2020, 31, 3254–3265. [CrossRef]

16. Shajini, M.; Ramanan, A. An improved landmark-driven and spatial–channel attentive convolutional neural network for fashionclothes classification. Vis. Comput. 2021, 37, 1517–1526. [CrossRef]

17. Shajini, M.; Ramanan, A. A knowledge-sharing semi-supervised approach for fashion clothes classification and attributeprediction. Visual Comput. 2022, 38, 3551–3561. [CrossRef]

18. Donati, L.; Iotti, E.; Mordonini, G.; Prati, A. Fashion product classification through deep learning and computer vision. Appl. Sci.

2019, 9, 1385. [CrossRef]

19. Rajput, P.S.; Aneja, S. IndoFashion: Apparel classification for Indian ethnic clothes. In Proceedings of the IEEE/CVF Conferenceon Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 3935–3939.

20. Eshwar, S.G.; Gautham Ganesh Prabhu, J.; Rishikesh, A.V.; Charan, N.A.; Umadevi, V. Apparel classification using convolutionalneural networks. In Proceedings of the 2016 International Conference on ICT in Business Industry & Government (ICTBIG),Indore, India, 18–19 November 2016; pp. 1–5. [CrossRef]

21. Agarap, A.F. An architecture combining convolutional neural network (CNN) and support vector machine (SVM) for imageclassification. arXiv 2017, arXiv:1712.03541. [CrossRef]

22. Bhatnagar, S.; Ghosal, D.; Kolekar, M.H. Classification of fashion article images using convolutional neural networks. In Proceedings of the 2017 Fourth International Conference on Image Information Processing (ICIIP), Shimla, India, 21–23 December 2017;pp. 1–6. Available online: https://ieeexplore.ieee.org/document/8313740 (accessed on the 26 October 2022).

23. Leithardt, V. Classifying Garments from Fashion-MNIST Dataset Through CNNs. Adv. Sci. Technol. Eng. Syst. J. 2021, 6, 989–994.[CrossRef]

24. Seo, Y.; Shin, K.S. Hierarchical convolutional neural networks for fashion image classification. Expert Syst. Appl. 2019, 116, 328–339.[CrossRef]

25. Tang, Y.; Cui, H.; Liu, S. Optimal Design of Deep Residual Network Based on Image Classification of Fashion-MNIST Dataset.J. Phys. Conf. Ser. 2020, 1624, 052011. [CrossRef]

26. Duan, C.; Yin, P.; Zhi, Y.; Li, X. Image classification of fashion-MNIST data set based on VGG network. In Proceedingsof the 2019 2nd International Conference on Information Science and Electronic Technology (ISET 2019), Taiyuan, China,21–22 September 2022; Volume 19. [CrossRef]

27. Lei, F.; Liu, X.; Dai, Q.; Ling, B.W.K. Shallow convolutional neural network for image classification. SN Appl. Sci. 2020, 2, 1–8.[CrossRef]Sensors 2022, 22, 9544 14 of 14

28. Saiharsha, B.; Abel Lesle, A.; Diwakar, B.; Karthika, R.; Ganesan, M. Evaluating performance of deep learning architectures forimage classification. In Proceedings of the 2020 5th International Conference on Communication and Electronics Systems (ICCES),Coimbatore, India, 21–22 October 2020; pp. 917–922. [CrossRef]

29. Greeshma, K.; Sreekumar, K. Fashion-MNIST classification based on HOG feature descriptor using SVM. Int. J. Innov. Technol.Explor. Eng. 2019, 8, 960–962. [CrossRef]

30. Hoang, K. Image Classification with Fashion-MNIST and CIFAR10; California State University: Sacramento, CA, USA, 2018.

31. Shen, S. Image Classification of Fashion-MNIST Dataset Using Long Short-Term Memory Networks. Available online: http://users.cecs.anu.edu.au/~Tom.Gedeon/conf/ABCs2018/paper/ABCs2018_paper_38.pdf (accessed on 26 October 2022).

32. Zhang, K. LSTM: An Image Classification Model Based on Fashion-MNIST Dataset. Available online: http://users.cecs.anu.edu.au/~Tom.Gedeon/conf/ABCs2018/paper/ABCs2018_paper_92.pdf (accessed on 26 October 2022).

33. Greeshma, K.; Sreekumar, K. Hyperparameter optimization and regularization on Fashion-MNIST classification. Int. J. RecentTechnol. Eng. 2019, 8, 3713–3719.

34. LeCun, Y. LeNet-5, Convolutional Neural Networks. 2015; Volume 20, p. 14. Available online: http://yann.lecun.com/exdb/lenet(accessed on 26 October 2022).

35. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. Adv. Neural Inf.Process. Syst. 2012, 25, 1097–1105. [CrossRef]

36. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficientconvolutional neural networks for mobile vision applications. arXiv 2017, arXiv:1704.04861. [CrossRef]

37. Tan, M.; Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In Proceedings of the InternationalConference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114. [CrossRef]

38. Yang, G.W.; Jing, H.F. Multiple convolutional neural network for feature extraction. In Proceedings of the International Conferenceon Intelligent Computing, Fuzhou, China, 20–23 August 2015; Springer: Cham, Switzerland, 2015; pp. 104–114. [CrossRef]

39. Liaw, R.; Liang, E.; Nishihara, R.; Moritz, P.; Gonzalez, J.E.; Stoica, I. Tune: A research platform for distributed model selectionand training. arXiv 2018, arXiv:1807.05118. [CrossRef]

40. Bergstra, J.; Yamins, D.; Cox, D.D. Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms.In Proceedings of the 12th Python in Science Conference, Austin, TX, USA, 24–29 June 2013; Volume 13, p. 20. [CrossRef]

41. Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; Koyama, M. Optuna: A next-generation hyperparameter optimization framework. InProceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA,4–8 August 2019; pp. 2623–2631. [CrossRef]

42. Jaderberg, M.; Dalibard, V.; Osindero, S.; Czarnecki, W.M.; Donahue, J.; Razavi, A.; Vinyals, O.; Green, T.; Dunning, I.;Simonyan, K.; et al. Population based training of neural networks. arXiv 2017, arXiv:1711.09846. [CrossRef]

43. Kingma, D.P.; Ba, J.A. Adam: A method for stochastic optimization. arXiv 2014, arXiv:1412.6980.

44. Soydaner, D. A comparison of optimization algorithms for deep learning. Int. J. Pattern Recognit. Artif. Intell. 2020, 34, 2052013.[CrossRef]

45. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. Pytorch:An imperative style, high-performance deep learning library. In Proceedings of the 2019 Conference on Neural InformationProcessing Systems, Vancouver, BC, Canada, 8–14 December 2019.

46. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.;

Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. arXiv 2020, arXiv:2010.11929.[CrossRef]

47. Cao, D.; Chen, Z.; Gao, L. An improved object detection algorithm based on multi-scaled and deformable convolutional neuralnetworks. Hum.-Centric Comput. Inf. Sci. 2020, 10, 1–22. [CrossRef]

48. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. In Proceedings of the 2014 Conference on Neural Information Processing Systems, Montreal, QC, Canada,8–13 December 2014.

49. LeCun, Y.; Misra, I. Self-Supervised Learning: The Dark Matter of Intelligence. 2021.

List of Codes for Software:

- **Python code for Website Creating using Streamlit**:

```
import streamlit as st
import tensorflow as tf
from PIL import Image
import numpy as np
import os

# Get the working directory of the current script
working_dir = os.path.dirname(os.path.abspath(__file__))
model_path = os.path.join(working_dir, 'trained_model',
'trained_fashion_mnist_model.h5')

# Load the pre-trained model
try:
    model = tf.keras.models.load_model(model_path)
except Exception as e:
    st.error(f"Error loading model: {e}")
    st.stop()

# Define class labels for Fashion MNIST dataset
class_names = ['T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat',
               'Sandal', 'Shirt', 'Sneaker', 'Bag', 'Ankle boot']

# Function to preprocess the uploaded image
def preprocess_image(image):
    try:
        img = Image.open(image)
        img = img.resize((28, 28))
        img = img.convert('L')  # Convert to grayscale
        img_array = np.array(img) / 255.0
        img_array = img_array.reshape((1, 28, 28, 1))
```

```
        return img_array
    except Exception as e:
        st.error(f"Error processing image: {e}")
        return None

# Streamlit App
st.title('Fashion Item Classifier')

uploaded_image = st.file_uploader("Upload an image...", type=["jpg", "jpeg", "png"])

if uploaded_image is not None:
    try:
        image = Image.open(uploaded_image)
        col1, col2 = st.columns(2)

        with col1:
            resized_img = image.resize((100, 100))
            st.image(resized_img, caption='Uploaded Image')

        with col2:
            if st.button('Classify'):
                # Preprocess the uploaded image
                img_array = preprocess_image(uploaded_image)

                if img_array is not None:
                    # Make a prediction using the pre-trained model
                    result = model.predict(img_array)
                    st.write("Prediction raw output:", result)  # Debugging statement

                    predicted_class = np.argmax(result)
                    prediction = class_names[predicted_class]

                    st.success(f'Prediction: {prediction}')
    except Exception as e:
        st.error(f"Error with uploaded image: {e}")
```

- **Model Training code in Google colab code:**

Seeding for reproducibility

```
# Set seeds for reproducibility
import random
random.seed(0)

import numpy as np
np.random.seed(0)

import tensorflow as tf
tf.random.set_seed(0)
```

Importing the dependencies

```
Start coding or generate with AI.
```

```
from tensorflow.keras import datasets, layers, models
import matplotlib.pyplot as plt
```

Data Curation

```
# Load and prepare the Fashion MNIST dataset
fashion_mnist = datasets.fashion_mnist
(train_images, train_labels), (test_images, test_labels) = fashion_mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-labels-idx1-ubyte.gz
29515/29515 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-images-idx3-ubyte.gz
26421880/26421880 [==============================] - 1s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-labels-idx1-ubyte.gz
5148/5148 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-images-idx3-ubyte.gz
4422102/4422102 [==============================] - 1s 0us/step
```

Data Processing

```
type(train_images)
```

```
numpy.ndarray
```

```
type(train_labels)
```

```
numpy.ndarray
```

```
print(len(train_images))
```

```
60000
```

```
print(len(train_labels))
```

```
60000
```

```
print(len(test_images))
```

```
10000
```

```
print(train_images[0].shape)
```

```
(28, 28)
```

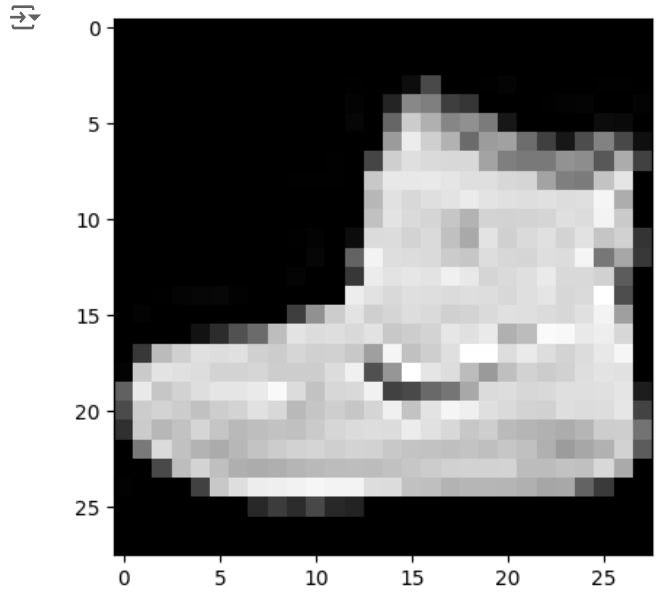
```
print(type(train_images[0]))
```

```
<class 'numpy.ndarray'>
```

```
print(train_images[0])
```

```
[[  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0  13  73   0
    0   1   4   0   0   0   0   1   1   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   3   0  36 136 127  62
   54   0   0   0   1   3   4   0   0   3]
 [  0   0   0   0   0   0   0   0   0   0   0   0   6   0 102 204 176 134
  144 123  23   0   0   0   0  12  10   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0 155 236 207 178
  107 156 161 109  64  23  77 130  72  15]
 [  0   0   0   0   0   0   0   0   0   0   0   1   0  69 207 223 218 216
  216 163 127 121 122 146 141  88 172  66]
 [  0   0   0   0   0   0   0   0   0   1   1   1   0 200 232 232 233 229
  223 223 215 213 164 127 123 196 229   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0 183 225 216 223 228
  235 227 224 222 224 221 223 245 173   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0 193 228 218 213 198
  180 212 210 211 213 223 220 243 202   0]
 [  0   0   0   0   0   0   0   0   0   1   3   0  12 219 220 212 218 192
  169 227 208 218 224 212 226 197 209  52]
 [  0   0   0   0   0   0   0   0   0   0   6   0  99 244 222 220 218 203
  198 221 215 213 222 220 245 119 167  56]
 [  0   0   0   0   0   0   0   0   0   4   0   0  55 236 228 230 228 240
  232 213 218 223 234 217 217 209  92   0]
 [  0   0   1   4   6   7   2   0   0   0   0   0 237 226 217 223 222 219
  222 221 216 223 229 215 218 255  77   0]
 [  0   3   0   0   0   0   0   0   0  62 145 204 228 207 213 221 218 208
  211 218 224 223 219 215 224 244 159   0]
 [  0   0   0   0  18  44  82 107 189 228 220 222 217 226 200 205 211 230
  224 234 176 188 250 248 233 238 215   0]
 [  0  57 187 208 224 221 224 208 204 214 208 209 200 159 245 193 206 223
  255 255 221 234 221 211 220 232 246   0]
 [  3 202 228 224 221 211 211 214 205 205 205 220 240  80 150 255 229 221
  188 154 191 210 204 209 222 228 225   0]
 [ 98 233 198 210 222 229 229 234 249 220 194 215 217 241  65  73 106 117
  168 219 221 215 217 223 223 224 229  29]
 [ 75 204 212 204 193 205 211 225 216 185 197 206 198 213 240 195 227 245
  239 223 218 212 209 222 220 221 230  67]
 [ 48 203 183 194 213 197 185 190 194 192 202 214 219 221 220 236 225 216
  199 206 186 181 177 172 181 205 206 115]
 [  0 122 219 193 179 171 183 196 204 210 213 207 211 210 200 196 194 191
  195 191 198 192 176 156 167 177 210  92]
 [  0   0  74 189 212 191 175 172 175 181 185 188 189 188 193 198 204 209
  210 210 211 188 188 194 192 216 170   0]
 [  2   0   0   0  66 200 222 237 239 242 246 243 244 221 220 193 191 179
  182 182 181 176 166 168  99  58   0   0]
 [  0   0   0   0   0   0   0  40  61  44  72  41  35   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0]]
```

```
# Display an image from the dataset
plt.imshow(train_images[0], cmap='gray')
plt.show()
```



```
print(train_labels[0])
```

```
9
```

class_names = ['T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat', 'Sandal', 'Shirt', 'Sneaker', 'Bag', 'Ankle boot']

```
# Normalize pixel values to be between 0 and 1
train_images, test_images = train_images / 255.0, test_images / 255.0
```

```
print(train_images[0])
```

```
[[0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.        ]
 [0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.        ]
 [0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.        ]
 [0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.00392157 0.         0.         0.05098039 0.28627451 0.
  0.         0.00392157 0.01568627 0.         0.         0.
  0.         0.00392157 0.00392157 0.         0.        ]
 [0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.01176471 0.         0.14117647 0.53333333 0.49803922 0.24313725
  0.21176471 0.         0.         0.         0.00392157 0.01176471
  0.01568627 0.         0.         0.01176471]
 [0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.02352941 0.         0.4        0.8        0.69019608 0.5254902
  0.56470588 0.48235294 0.09019608 0.         0.         0.
  0.         0.04705882 0.03921569 0.         0.        ]
 [0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.60784314 0.9254902  0.81176471 0.69803922
  0.41960784 0.61176471 0.63137255 0.42745098 0.25098039 0.09019608
  0.30196078 0.50980392 0.28235294 0.05882353]
 [0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.00392157
```

```
        0.         0.27058824 0.81176471 0.8745098  0.85490196 0.84705882
        0.84705882 0.63921569 0.49803922 0.4745098  0.47843137 0.57254902
        0.55294118 0.34509804 0.6745098  0.25882353]
       [0.         0.         0.         0.         0.         0.
        0.         0.         0.         0.00392157 0.00392157 0.00392157
        0.         0.78431373 0.90980392 0.90980392 0.91372549 0.89803922
        0.8745098  0.8745098  0.84313725 0.83529412 0.64313725 0.49803922
        0.48235294 0.76862745 0.89803922 0.        ]
       [0.         0.         0.         0.         0.         0.
        0.         0.         0.         0.         0.         0.
        0.         0.71764706 0.88235294 0.84705882 0.8745098  0.89411765
        0.92156863 0.89019608 0.87843137 0.87058824 0.87843137 0.86666667
        0.8745098  0.96078431 0.67843137 0.        ]
       [0.         0.         0.         0.         0.         0.
        0.         0.         0.         0.         0.         0.
        0.         0.75686275 0.89411765 0.85490196 0.83529412 0.77647059
        0.70588235 0.83137255 0.82352941 0.82745098 0.83529412 0.8745098
        0.8627451  0.95294118 0.79215686 0.        ]
       [0.         0.         0.         0.         0.         0.
        0.         0.         0.         0.00392157 0.01176471 0.
        0.04705882 0.85882353 0.8627451  0.83137255 0.85490196 0.75294118
```

```
# Reshape images to specify that it's a single channel (grayscale)
train_images = train_images.reshape((train_images.shape[0], 28, 28, 1))
test_images = test_images.reshape((test_images.shape[0], 28, 28, 1))
```

```
train_images.shape
```

```
(60000, 28, 28, 1)
```

```
test_images.shape
```

```
(10000, 28, 28, 1)
```

Convolutional Neural Network

```
# Build the convolutional base
model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))

# Add Dense layers on top
model.add(layers.Flatten())
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(10))
```

```
# Compile and train the model
model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])
```

Model Training

```
history = model.fit(train_images, train_labels, epochs=5,
                    validation_data=(test_images, test_labels))
```

```
Epoch 1/5
1875/1875 [==============================] - 19s 5ms/step - loss: 0.4899 - accuracy: 0.8210 - val_loss: 0.3594 - val_accurac
Epoch 2/5
1875/1875 [==============================] - 7s 4ms/step - loss: 0.3184 - accuracy: 0.8826 - val_loss: 0.3154 - val_accuracy
Epoch 3/5
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2704 - accuracy: 0.8999 - val_loss: 0.2957 - val_accuracy
Epoch 4/5
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2401 - accuracy: 0.9111 - val_loss: 0.2716 - val_accuracy
Epoch 5/5
```

```
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2181 - accuracy: 0.9188 - val_loss: 0.2758 - val_accuracy
```

Model Evaluation

```
# Evaluate the model
test_loss, test_acc = model.evaluate(test_images,  test_labels, verbose=2)
print('\nTest accuracy:', test_acc)
```

```
313/313 - 1s - loss: 0.2758 - accuracy: 0.9015 - 604ms/epoch - 2ms/step

Test accuracy: 0.9014999866485596
```

```
# Plot training & validation accuracy values
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()

# Plot training & validation loss values
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()
```

Show hidden output

```
model.save('trained_fashion_mnist_model.h5')
```

```
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103: UserWarning: You are saving your model as an HDF5
  saving_api.save_model(
```

Start coding or generate with AI.