CLASSIFICATION OF FABRIC PATTERNS USING DEEP LEARNING

1. INTRODUCTION

1.1 Project Overview

The project titled "Classification of Fabric Patterns using Deep Learning" aims to leverage computer vision and deep learning technologies to identify and classify different types of fabric patterns. This includes recognizing patterns such as floral, striped, polka dots, geometric, plain, and more through image analysis. By automating this process, the system can enhance efficiency and accuracy in applications like textile quality control, e-commerce cataloging, and fabric inventory management.

1.2 Purpose

The purpose of the project is to build an intelligent system that can recognize various fabric patterns with high accuracy and speed. This automation not only reduces the manual effort required in the textile industry but also ensures consistent and scalable classification. The system can serve designers, manufacturers, and online retailers by providing instant pattern classification and enabling better organization and recommendation of fabric products.

2. IDEATION PHASE

2.1 Problem Statement

The textile industry currently relies heavily on manual classification of fabric patterns, which is not only time-consuming but also susceptible to human error. With the increasing diversity and volume of fabric designs, there is a pressing need for an automated solution that can handle large-scale classification efficiently and accurately.

2.2 Empathy Map Canvas

- User: Textile industry professionals, designers, retailers
- Says: "I need a quick way to identify fabric patterns."
- Thinks: "This classification task takes too much of my time."
- Does: Manually checks and tags fabric samples
- Feels: Frustrated with repetitive tasks and inconsistency in results
- Gains: Time-saving, reliable pattern classification
- Pains: Errors due to visual fatigue, lack of standard tagging

2.3 Brainstorming

During brainstorming sessions, the team identified potential technologies and solutions, including:
- Use of Convolutional Neural Networks (CNNs) for image classification
- Mobile app interface for real-time photo input
- Data augmentation for robust training
- Feedback loop for model improvement
- Integration with inventory and product management systems

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

The user journey starts from uploading a fabric image using a web or mobile interface. The image is then analyzed by a trained CNN model, which returns the pattern prediction along with a confidence score. The user can view, save, or export the result and optionally provide feedback. Historical results are accessible through a dashboard.

3.2 Solution Requirement

- Functional Requirements: Upload image, classify pattern, display result, export, view history, admin panel
- Non-functional Requirements: High usability, security, reliability, performance, availability, scalability

3.3 Data Flow Diagram

- Context-Level DFD shows the interaction between the user and system components like upload module, preprocessing, classification engine, and result storage.
- Level-1 DFD further details data flow between system processes and databases.

3.4 Technology Stack

- Frontend: HTML, CSS, JavaScript, React
- Backend: Python, Flask
- Machine Learning: TensorFlow, Keras (CNN model)
- Database: MongoDB (local + cloud via MongoDB Atlas)
- Hosting: Heroku / Firebase / AWS

4. PROJECT DESIGN

4.1 Problem-Solution Fit

The problem of manually classifying fabric patterns finds a fitting solution through the use of deep learning models. These models can recognize intricate visual patterns with high precision, significantly reducing time and effort required for manual classification.

4.2 Proposed Solution

The system enables users to upload fabric images via a user-friendly interface. The backend processes the image using a trained CNN model and classifies the pattern. Results are presented in a readable format and can be saved or exported. The admin can manage users, view analytics, and update the model.

4.3 Solution Architecture

The architecture consists of:
- Presentation Layer: Web/mobile UI
- Application Layer: Flask API, image processing pipeline
- ML Model Layer: CNN-based classification
- Data Layer: MongoDB, cloud storage
- Infrastructure: Hosted on cloud (Heroku/Firebase)

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

The project was planned over 4 sprints:
- Sprint 1: Requirement gathering, dataset collection, preprocessing
- Sprint 2: Model training and initial testing
- Sprint 3: Backend and frontend integration
- Sprint 4: Final testing, documentation, and deployment

Each sprint lasted two weeks with specific deliverables and review checkpoints.

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

Model was tested for:
- Accuracy: Training – 95.4%, Validation – 91.7%
- Tuning Result: Validation improved to 93.2% with learning rate optimization and dropout
- Execution Time: Classification within 2 seconds

Performance tests confirmed the robustness and efficiency of the model under various scenarios.

7. RESULTS

7.1 Output Screenshots

```
fig, axes = plt.subplots(nrows=4, ncols=4, figsize=(15, 10))
random_index = np.random.randint(0, len(test_gen), 16)

for i, ax in enumerate(axes.ravel()):
    img_path = test_df['path'].iloc[random_index[i]]

    ax.imshow(load_img(img_path))
    ax.axis('off')

    if test_df['label'].iloc[random_index[i]] == pred[random_index[i]]:
        color = "green"
    else:
        color = "red"

    ax.set_title(f"True: {test_df['label'].iloc[random_index[i]]}\nPredicted: {pred[random_index[i]]}", color=color)

plt.tight_layout()
plt.show()
```

8. ADVANTAGES & DISADVANTAGES

Advantages

- High classification accuracy
- Quick inference
- Scalable and modular design
- User-friendly interface

Disadvantages

- Requires internet for cloud classification
- Performance drops on low-resolution images
- Dataset-dependent accuracy

9. CONCLUSION

The project successfully demonstrates the feasibility of using deep learning for fabric pattern classification. It simplifies and accelerates the workflow for textile professionals. The implemented system is accurate, efficient, and ready for further deployment or enhancement based on industry feedback.

10. FUTURE SCOPE

- Multilingual support for broader user base
- Integration with inventory systems
- Expansion to texture and fabric quality classification
- Offline model inference capability
- Real-time camera-based classification on mobile apps

11. APPENDIX

- Source Code: Project description and chatgpt
- Dataset: https://www.kaggle.com/datasets/nguyngiabol/dress-pattern-dataset