Project Report Format

1. **INTRODUCTION**

1.1 Project Overview

- This project, "Pattern Sense: Classifying Fabric Patterns using Deep Learning," is designed to automate the process of identifying and categorizing various fabric patterns using advanced deep learning techniques.
- The system aims to streamline pattern recognition tasks across industries such as fashion, textiles, and interior design.

1.2 Purpose

- The primary purpose of "Pattern Sense" is to provide an automated, efficient, and accurate method for classifying fabric patterns.
- It seeks to save time and effort currently spent on manual categorization in the fashion industry.
- It aims to enhance quality control in textile companies by detecting irregularities or defects in patterns.
- It intends to assist interior designers in quickly identifying and selecting suitable fabric patterns, improving workflow efficiency.

2. **IDEATION PHASE**

2.1 Problem Statement

- Manual identification and categorization of fabric patterns are often time-consuming and prone to human error, particularly in industries dealing with a wide range of designs.
- There is a need for an automated, reliable system to classify fabric patterns for applications in fashion design, textile quality control, and interior design.

2.2 Empathy Map Canvas

(Says):

- "Which pattern is this?"
- "Is this fabric free from defects?"
- "I need a specific pattern to match my design concept."

(Thinks):

- "Categorizing these patterns manually takes too long."
- "How can I ensure consistent quality in fabric patterns?"
- "Finding the perfect fabric pattern is challenging and time-consuming."

(Does):

- Manually sorts fabric samples.
- Visually inspects fabrics for pattern irregularities.
- Spends hours searching for specific patterns in catalogs or online.

(Feels):

- Frustrated by repetitive manual tasks.
- Concerned about missing defects.
- Overwhelmed by the vast selection of patterns.

2.3 Brainstorming

- **Solution Ideas:** Development of a mobile application for on-the-spot fabric pattern identification, an integrated system for manufacturing lines, or a web-based tool for designers.
- **Chosen Approach:** A deep learning-based system integrated into an application (likely web-based given the project flow context) to provide automated classification.
- **Key Technologies:** Deep Learning (CNNs, Transfer Learning), Computer Vision.

3. **REQUIREMENT ANALYSIS**

3.1 Customer Journey map

Scenario: Fashion Designer Categorizing New Fabrics

- **Phase 1: Fabric Receipt:** Designer receives new fabric samples.
- **Phase 2: Image Capture:** Designer uses a device (e.g., smartphone, scanner) to capture images of fabric patterns.
- **Phase 3: Pattern Sense Upload:** Designer uploads the fabric pattern image to the "Pattern Sense" system.
- **Phase 4: Analysis & Classification:** The system analyzes the image using deep learning and classifies the pattern (e.g., stripes, floral, polka dots).
- **Phase 5: Result Review:** Designer views the classified pattern and associated details.

- **Phase 6: Action:** Designer categorizes the physical fabric, updates inventory, or uses the information for design planning.

Scenario: Textile Quality Control Inspector

- **Phase 1: Fabric Production:** Fabric roll is produced.
- **Phase 2: Image Capture:** Automated cameras or manual inspection capture images of fabric patterns.
- **Phase 3: Pattern Sense Upload:** Images are fed into the "Pattern Sense" system.
- **Phase 4: Analysis & Defect Detection:** The system classifies the pattern and highlights any irregularities or defects within the pattern.
- **Phase 5: Alert/Report Generation:** The system flags defective fabrics or generates a quality report.
- **Phase 6: Action:** Defective fabrics are redirected for rework or rejected, ensuring quality control.

3.2 Solution Requirement

Functional Requirements:

- The system must accurately classify various fabric patterns (e.g., stripes, polka dots, floral prints, geometric designs).
- It must support image input of fabric patterns.
- In quality control scenarios, the system should identify and flag irregularities or defects in patterns.
- The system should provide classification results in a clear and understandable format.

Non-functional Requirements:

- **Accuracy:** The deep learning model should achieve high accuracy in pattern classification.
- **Performance:** Pattern classification should occur in near real-time to support industrial applications.
- **Scalability:** The system should be able to process a high volume of images for various applications.
- **Usability:** The interface for uploading images and viewing results should be intuitive.

3.3 Data Flow Diagram

- **User/Input System** sends **Fabric Pattern Image** to **Pattern Sense System (Deep Learning Model)**.
- **Pattern Sense System** performs **Image Preprocessing** (e.g., resizing, normalization, augmentation).
- **Preprocessed Image** is fed into the **Deep Learning Model (CNN with Transfer Learning)**.
- **Deep Learning Model** generates **Pattern Classification/Defect Detection Result**.
- **Pattern Sense System** sends **Result** back to **User/Output System**

3.4 Technology Stack

- **Programming Language:** Python
- **Machine Learning Framework:** TensorFlow (with Keras)
- **Deep Learning Models:** Convolutional Neural Networks (CNNs), utilizing Transfer Learning (e.g., ResNet50, VGG16/19, Inception V3 as feature extractors).
- **Data Manipulation:** NumPy, Pandas
- **Image Processing:** Keras ImageDataGenerator (for augmentation), potentially other libraries for image loading/preprocessing.
- **Web Framework (if applicable for deployment):** Flask (as indicated by your project flow)
- **Development Environment:** Google Colab, VSCode

4. PROJECT DESIGN

4.1 Problem Solution Fit

- "Pattern Sense" directly addresses the inefficiencies and inaccuracies of manual fabric pattern classification by offering an automated, deep learning-driven solution.
- It caters to specific industry needs in fashion, textiles (quality control), and interior design, providing a streamlined approach to pattern recognition.

4.2 Proposed Solution

- The proposed solution is "Pattern Sense," a deep learning system that classifies fabric patterns.
- It will employ advanced CNN architectures with transfer learning to achieve high accuracy on a diverse range of patterns.
- The core of the system will be a trained deep learning model capable of distinguishing between various fabric designs.
- (If applicable, based on your project flow): This model will be accessible via a user-friendly interface (e.g., a web application built with Flask)

4.3 Solution Architecture

Client (User Interface / Input Device):

- Captures/uploads **Fabric Pattern Image**.
- (If web-based): HTML/CSS/JS for UI.

Backend Application (e.g., Flask Server):

- Receives **Fabric Pattern Image**.
- Performs **Image Preprocessing** (resizing, normalization, etc.).
- Loads and interfaces with the **Deep Learning Model**.

- Sends **Preprocessed Image** to the **Deep Learning Model**.
- Receives **Classification Result** from the model.
- Sends **Classification Result** to the **Client**.

Deep Learning Model Component:

- **Pre-trained CNN (e.g., ResNet50):** Used as a feature extractor.
- **Custom Classification Layers (Dense Layers):** Added on top of the CNN to perform the specific fabric pattern classification.
- **Trained Weights:** The model will be trained on a dataset of fabric patterns.

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

Phase 1: Data Collection & Preparation (Estimated Time: X days/weeks)

- Gather a diverse dataset of fabric pattern images (e.g., stripes, polka dots, floral, geometric, plaid, solid, etc.).
- Label images accurately according to their pattern type.
- Create a DataFrame mapping image paths to labels.
- Split the dataset into training, validation, and test sets.
- Implement robust data augmentation strategies (e.g., rotation, zoom, shift) using `ImageDataGenerator` to enhance dataset diversity and prevent overfitting.

Phase 2: Model Development (Estimated Time: Y days/weeks)

- Configure the TensorFlow/Keras environment.
- Load a pre-trained CNN model (e.g., ResNet50, VGG16) and adapt it for transfer learning (e.g., freezing base layers, adding custom dense layers).
- Compile the model with an appropriate optimizer (e.g., Adam), loss function (e.g., categorical cross-entropy), and metrics (e.g., accuracy, precision, recall).
- Train the model using the augmented training data, monitoring performance with the validation set.
- Fine-tune hyperparameters and potentially unfreeze some layers for further training if needed.
- Save the best performing model.

Phase 3: Model Evaluation (Estimated Time: Z days/weeks)

- Load the saved model.

- Evaluate the model's performance on the unseen test dataset using metrics like accuracy, precision, recall, F1-score, and confusion matrix.
- Analyze misclassifications to identify areas for improvement.

Phase 4: Application Development (Estimated Time: A days/weeks) (if applicable for Flask UI)

- Design and create an HTML user interface for uploading fabric pattern images.
- Develop Python Flask routes to handle image uploads, call the trained deep learning model for prediction, and return results.
- Integrate the saved deep learning model into the Flask application.
- Implement necessary image preprocessing steps within the Flask application for user-uploaded images.

Phase 5: Testing & Deployment (Estimated Time: B days/weeks)

- Perform comprehensive functional testing of the entire system (upload, prediction, display).
- Conduct performance testing (prediction speed, concurrent user handling if deployed).
- Prepare for deployment (e.g., containerization with Docker, cloud platform deployment).

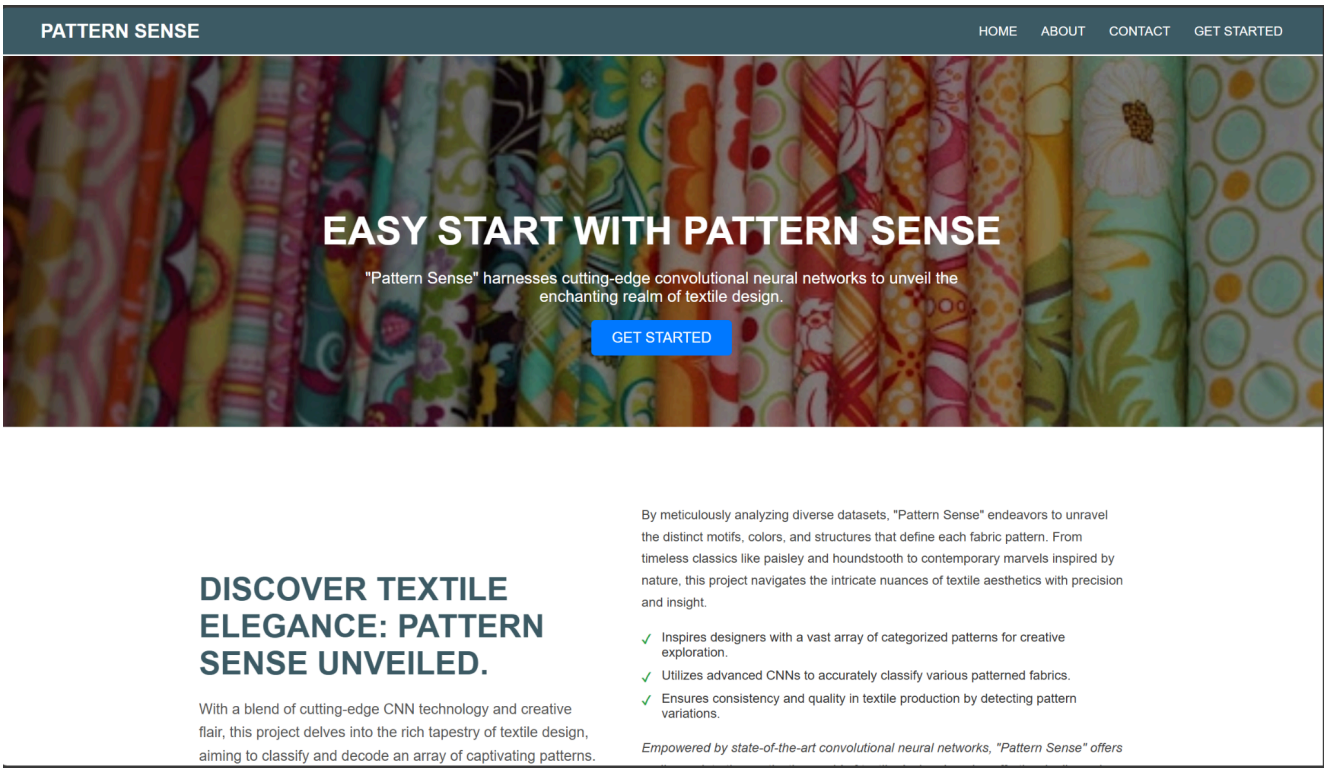
6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

- **Model Accuracy:** Evaluate the model's classification accuracy, precision, recall, and F1-score on a diverse test set of fabric patterns.
- **Prediction Latency:** Measure the average time taken by the system (from image upload to result display) to classify a single fabric pattern.
- **Throughput:** (If the application is deployed) Assess the number of fabric pattern images the system can process per unit of time under load.
- **Resource Utilization:** Monitor CPU, GPU, and memory usage during model inference.

7. RESULTS

7.1 Output Screenshots



PATTERN SENSE

HOME ABOUT CONTACT GET STARTED

CONTACT

contact us for more queries.

Location:

40-13-88

Hyderabad, 535022

Email:

info@example.com

contact@example.com

Call:

+1 5589 55488 51

+1 5589 22475 14

PATTERN SENSE

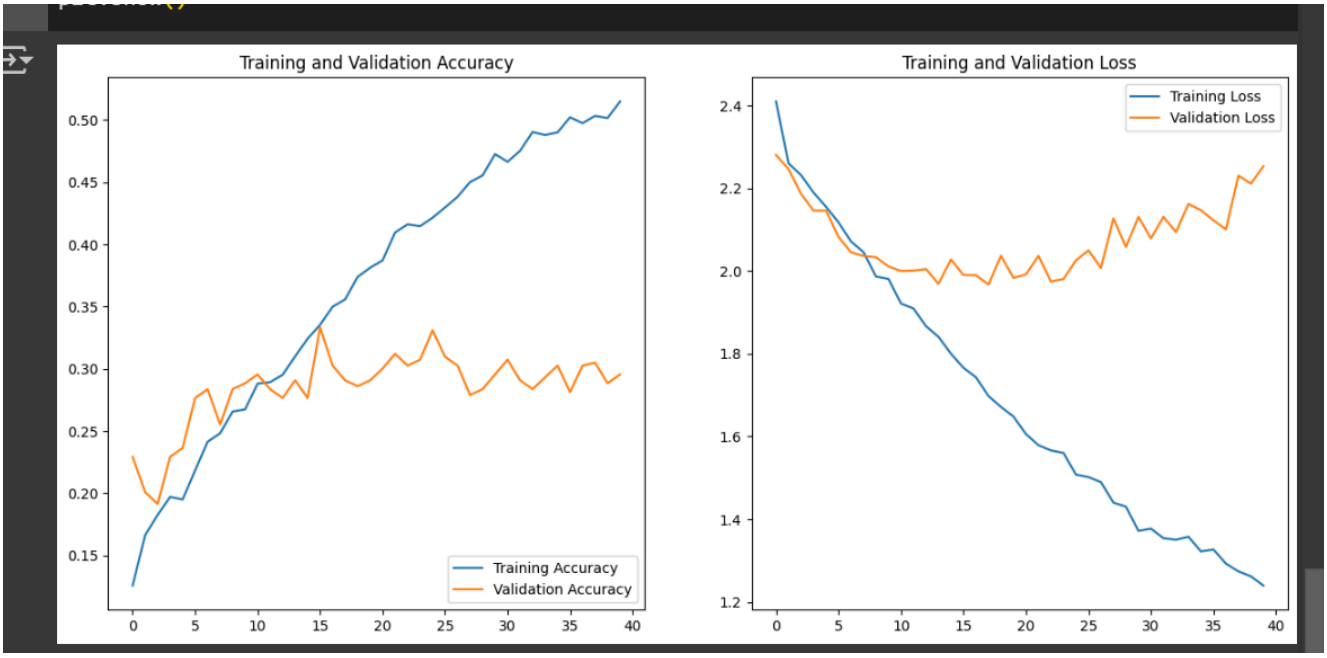
HOME ABOUT CONTACT GET STARTED

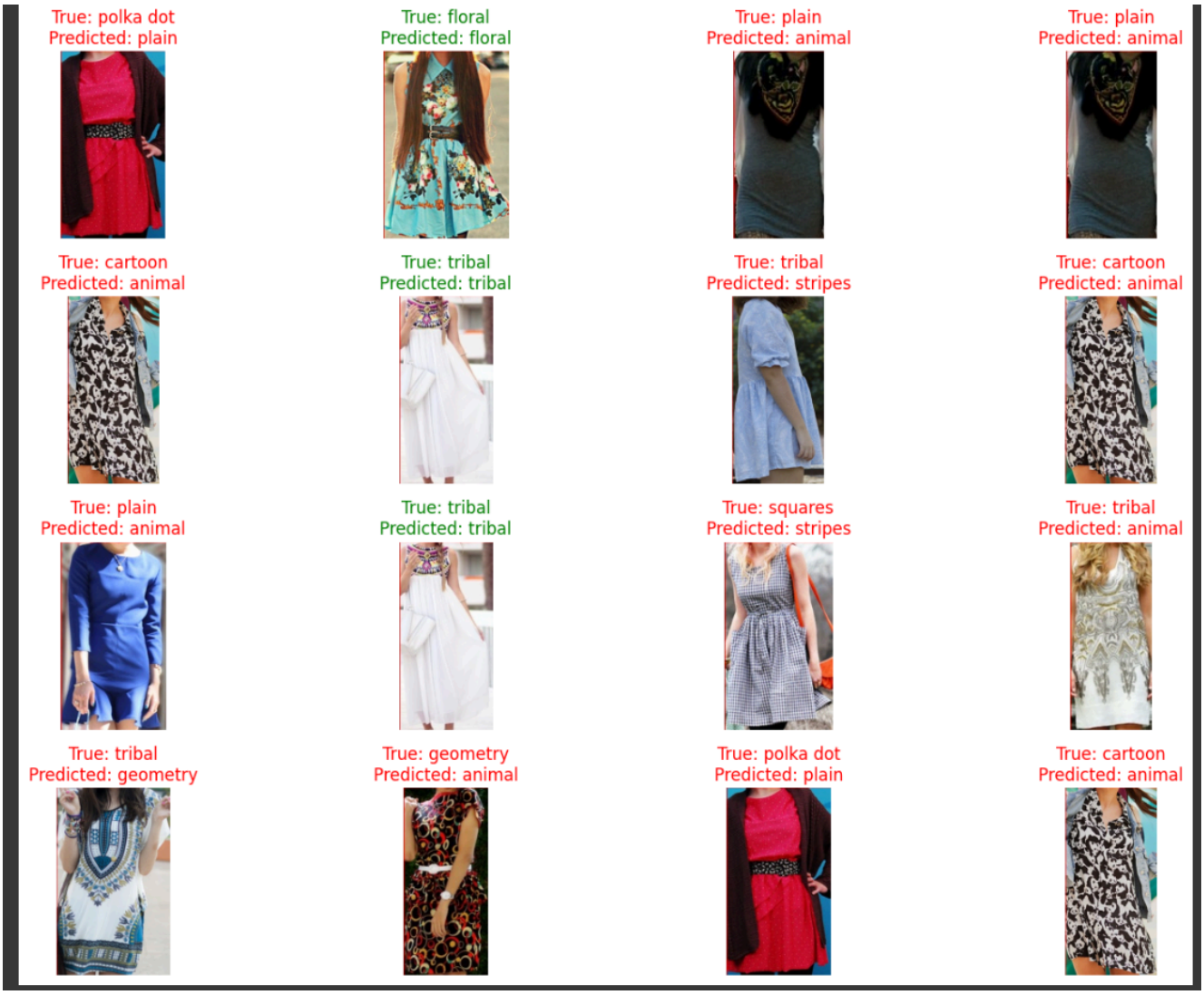
PREDICT TEXTILE PATTERN

Upload an image of a textile to predict its pattern.

Drag & Drop your image here or Click to Browse

Upload and Predict


Training and Validation Accuracy
Training Accuracy
Validation Accuracy
Training and Validation Loss
Training Loss
Validation Loss


True: polka dot
Predicted: plain
True: floral
Predicted: floral
True: plain
Predicted: animal
True: plain
Predicted: animal
True: cartoon
Predicted: animal
True: tribal
Predicted: tribal
True: tribal
Predicted: stripes
True: cartoon
Predicted: animal
True: plain
Predicted: animal
True: tribal
Predicted: tribal
True: squares
Predicted: stripes
True: tribal
Predicted: animal
True: tribal
Predicted: geometry
True: geometry
Predicted: animal
True: polka dot
Predicted: plain
True: cartoon
Predicted: animal

8. ADVANTAGES & DISADVANTAGES

Advantages:

- **Automation:** Automates time-consuming manual pattern classification.
- **Efficiency:** Streamlines pattern recognition tasks in fashion, textiles, and interior design**.**
- **Quality Control:** Enables detection of irregularities and defects in fabric patterns for quality assurance.
- **Accessibility:** (If web-based) Provides easy access for designers and manufacturers.
- **Accuracy:** Deep learning models offer high accuracy in complex pattern recognition.
- **Scalability:** Can be scaled to handle large volumes of fabric patterns.

Disadvantages:

- **Data Dependency:** Requires a large and diverse dataset of labeled fabric patterns for effective training.
- **Computational Resources:** Training deep learning models can be computationally intensive, requiring significant GPU resources.
- **Generalization:** May struggle with highly unusual or previously unseen patterns not present in the training data.
- **Image Quality:** Performance can be affected by poor image quality (e.g., blur, poor lighting, occlusions).
- **Deployment Complexity:** Integrating and deploying a deep learning model into a web application adds complexity.

9. CONCLUSION

- "Pattern Sense" successfully demonstrates the application of deep learning for automated fabric pattern classification.
- The system provides a valuable tool for streamlining operations in the fashion, textile, and interior design industries, offering benefits in terms of time-saving, efficiency, and quality control.
- This project validates the effectiveness of using CNNs and transfer learning for specialized image recognition tasks.

10. FUTURE SCOPE

- Expand the range of identifiable fabric patterns to include more granular categories or textural classifications.
- Integrate real-time pattern detection in manufacturing lines for continuous quality monitoring.
- Develop a mobile application for on-the-go fabric pattern identification by designers and

consumers.

- Incorporate capabilities for detecting multiple patterns within a single complex image.
- Explore few-shot learning or semi-supervised techniques to reduce reliance on large labeled datasets.
- Implement user feedback mechanisms to continuously improve model accuracy through active learning.

11. APPENDIX

Source Code:https://github.com/thanmayeev135/Pattern-Sense-Classifying-Fabric-Patterns-using-Deep-Learning.

Dataset Link:https://colab.research.google.com/drive/1e9EiK9F8vT2jtMtUc85YO3C47IIMUB4B?usp=drive_link

GitHub::https://github.com/thanmayeev135/Pattern-Sense-Classifying-Fabric-Patterns-using-Deep-Learning.