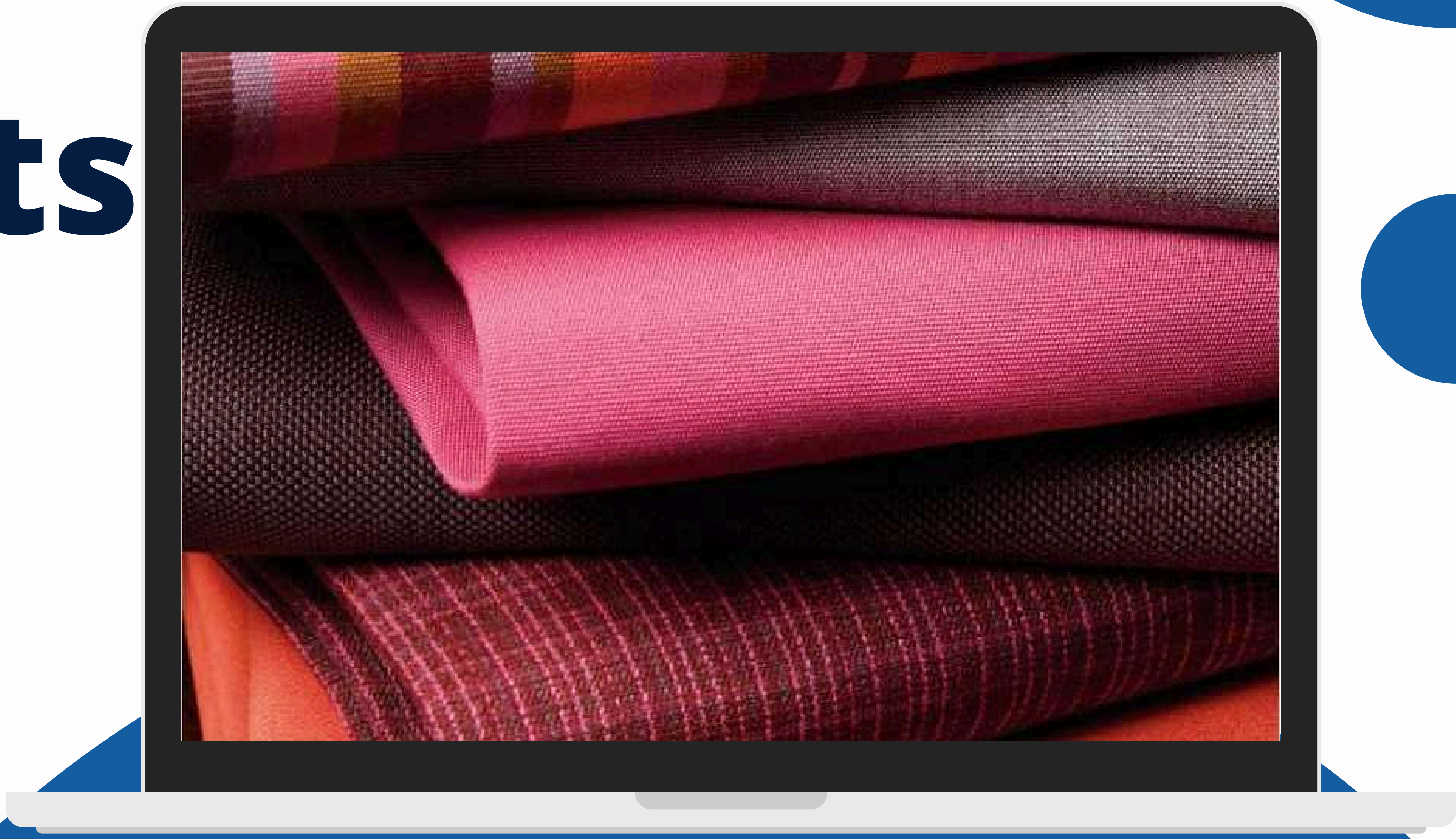
Smart Manufacturing Project Proposal:

Detecting Textile Defects using ResNet50



1093532 - 石聰

1103549 - 彭滄晏

1103558 - 林米克

Overview

BACKGROUND

Background

Industry 4.0 is here:

- Automation and data are revolutionizing manufacturing.
- Quality control remains critical for product standards.
- Efficient defect detection boosts productivity and reduces waste.

DOMAIN

Problem Domain:
Textile Manufacturing

In textile manufacturing, ensuring product quality is crucial, especially when detecting and managing fabric defects. Traditionally, this is a manual, labor-intensive process that relies heavily on human inspectors. These methods are often inconsistent due to fatigue and subjective judgment, leading to missed defects or misclassifications. Furthermore, the sheer variety of defects and complex fabric patterns make it even more challenging to maintain high accuracy. As demand for faster and more reliable quality control grows, a shift towards automated solutions has become essential.

To sum it up:

- **Manual Inspection:** Prone to human error, fatigue, and inconsistency.
- **Defect Variety:** Numerous types, including holes, stains, and irregular patterns.
- **Complex Fabrics:** Difficult for human eyes to detect subtle flaws.
- **What the Industry needs:** Faster, scalable, and highly accurate quality control.

TARGET

Target Problem

Automation of defect detection processes

Focus:

- accuracy
- generalization
- efficiency
- scalability

Challenge 01

Training the models :
Lack of availability of extensive labelled dataset

Challenge 02

High variety and complexity:
Subtlety of texture, weave patterns, color, size, and shape can give false positives without extensive datasets available.

Challenge 03

Adaptability and computational overhead:
Difficult to render the system effective and efficient (for real-time application) without continuous adjustments.

The benefits

Efficiency and cost reduction

Enhanced product quality

Scalability

LITERATURE

Existing literature

Several research efforts and solutions have been developed to address textile defect detection, evolving from traditional image processing methods to more advanced deep learning-based approaches

Traditional Image Processing

- Optimal Gabor Filter
- Wavelet Transform and PCA

Machine Learning

- Support Vector Machines (SVM) and k means clustering

Deep Learning

- Custom CNN
- Transfer Learning with pre-trained models: YOLO, VGGNet, ResNet etc.

SOLUTION

CNNs Save The Day

ResNet's residual blocks will allow for easy classification of our images. As it increases in depth its accuracy will be unaffected, allowing it to see the necessary features to identify the defects within the textiles.

01

Data Collection and Preprocessing:

- Label images
- Normalize values
- Reduce Dimensions

02

Extract Features:

- Extraction of edges, textures, shapes, etc
- Residual blocks increases efficiency
- Max pooling at the beginning
- Global average to end
- Feature map created

03

Model Training:

- Training, validation, and test subsets
  - 70/15/15
- Data augmentation (rotation and scaling)
- Batch Normalization
  - Improving generalization

04

Detect the Defects:

- Success of model will be based on the following scores:
  - Precision
  - Recall
  - F1

IMPLEMENTATION & RESULTS

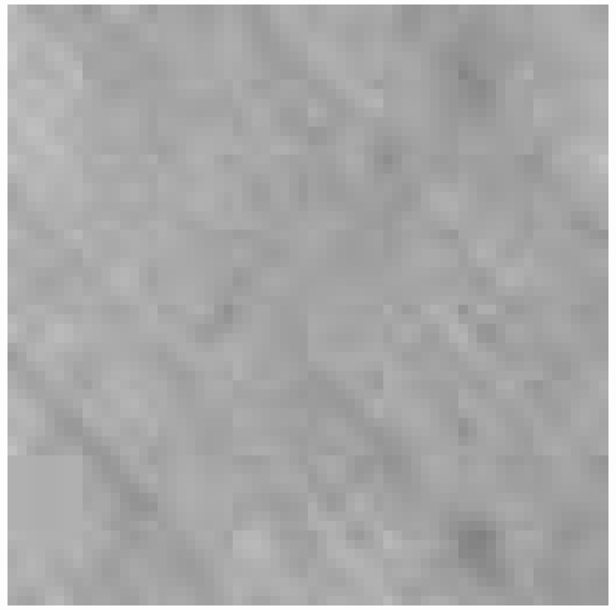
- **Used ResNet-18 pretrained**
- **Tried multiple loss functions and optimizers**
  - **Loss Functions:**
    - **Cross-Entropy**
    - **Focal Loss**
  - **Optimizers**
    - **Adam**
    - **SGD**
- **Used different combinations of loss functions and optimizers**
  - **Tried different combinations of adjusting hyperparameters**
- **Augmented images in multiple ways including:**
  - **Random flips and crops**

Training Results

Model	Accuracy	Precision	Recall	F1 Score
Adam-CrossE	.8358	.8398	.8358	.8363
Adam-Focal	.8718	.8726	.8718	.8720
SGD-CrossE	.7697	.7981	.7697	.7689
SGD-Focal	.8615	.8623	.8615	.8617

All models were unable to correctly predict images outside dataset

Sample Images



```
Image: 000_patch0-2.png, Predicted class: 1
Image: 000_patch0-0.png, Predicted class: 1
Image: 000_patch0-3.png, Predicted class: 1
Image: 000_patch0-1.png, Predicted class: 1
Image: 000_patch0-4.png, Predicted class: 1
Image: hole_patch1-1.png, Predicted class: 1
Image: damaged_1.jpg, Predicted class: 1
```

Conclusion

- **Models have good metrics but poor performance**
  - **Possibly due to overfitting**
- **Possible ways to improve models**
  - **Try different sample sizes as there could be over or undersampling**
  - **Apply dropout or weight decay techniques**
  - **Continue to test with different hyperparameters**
  - **Try using a different architecture**
  - **Revisit preprocessing (possible improper labeling or normalization)**

Short Demo

THANK YOU!