Image Blur Detection Project Report

1. Introduction

Image blur detection is an important task in digital image processing and computer vision. Blurred images can negatively affect image analysis, object recognition, and visual quality. The goal of this project is to automatically detect whether an image is **sharp** or **blurred** using classical image processing features and a machine learning model.

---

2. Dataset Description

The dataset used in this project is the **Blur Dataset** obtained from Kaggle.

Dataset structure:

- sharp
- motion_blurred
- defocused_blurred

Each category contains **350 images**, resulting in a total of **1050 images**.

For classification purposes:

- `sharp` → label **0**
- `motion_blurred` and `defocused_blurred` → label **1**

---

3. Preprocessing

- Images were loaded from the dataset directory.
- All images were converted to **grayscale**.
- No resizing was required since the dataset images were already standardized.

---

4. Feature Extraction

Four handcrafted features were extracted from each image:

1. **Laplacian Variance**
   Measures image sharpness by detecting high-frequency edges.

2. **Gradient Variance**
   Measures the variance of gradient magnitudes in the image.

3. **Sobel Magnitude**
   Computes the average edge strength using Sobel operators.

4. **Tenengrad**
   Measures image focus by summing squared Sobel gradients.

These features were stored in a DataFrame and later saved as a CSV file: `blur_features_results.csv`

---

5. Feature Analysis

Statistical analysis showed clear differences between sharp and blurred images:

- Sharp images had higher edge-related feature values.
- Defocused blur significantly reduced high-frequency information.
- Motion blur showed intermediate behavior.

Feature distribution plots confirmed that **Laplacian variance** and **Tenengrad** are highly discriminative features.

---

6. Classification Model

A **Multilayer Perceptron (MLP)** classifier was used.

Model configuration:

- Hidden layer: 20 neurons
- Activation function: default (ReLU)
- Max iterations: 500

Preprocessing:

- Features were standardized using `StandardScaler`.
- Data split: 80% training, 20% testing.

---

7. Results

Overall Performance:

- **Accuracy:** 89%

Confusion Matrix:

- Sharp images: Some misclassified as blurred
- Blurred images: Detected with high accuracy

Classification Report:

- Precision (Sharp): 0.87
- Recall (Sharp): 0.79

- Precision (Blur): 0.90
- Recall (Blur): 0.94

Per-Blur-Type Evaluation:

- **Sharp:** Accuracy ≈ 80.9%
- **Motion Blur:** Accuracy ≈ 88.3%
- **Defocused Blur:** Accuracy ≈ 99.4%

---

8. Discussion

The model performed very well in detecting defocused blur, as it strongly reduces edge information. Motion blur was also detected effectively. Some sharp images were misclassified due to low contrast or lack of texture.

Handcrafted features proved effective, especially Tenengrad and Laplacian variance.

Comparison Between Laplacian Variance and Tenengrad

Both **Laplacian Variance** and **Tenengrad** are edge-based focus measures, but they capture image sharpness in slightly different ways:

- **Laplacian Variance** focuses on second-order intensity changes. It is very sensitive to fine details and high-frequency edges, which makes it effective for detecting defocused blur. However, it can be affected by image noise.

- **Tenengrad** relies on first-order gradients computed using Sobel operators. It measures the overall strength of edges in the image and is more stable in the presence of noise. Tenengrad performed consistently well across sharp, motion-blurred, and defocused images.

In this project, Tenengrad showed better robustness, while Laplacian Variance provided strong discrimination for defocused blur. Combining both features improved the overall classification performance.

Lessons Learned

- Classical image processing features can still achieve strong performance when carefully selected.
- Feature scaling is crucial for neural network models such as MLP.
- Defocused blur is easier to detect than motion blur due to the loss of high-frequency information.
- Simpler models with good features can outperform complex models on well-structured datasets.

These lessons highlight the importance of understanding both the data and the chosen features before moving to more complex deep learning approaches.

Why MLP Was Chosen?

The Multilayer Perceptron (MLP) classifier was chosen for this project due to its ability to model non-linear relationships between input features.

The extracted blur features are numerical and relatively low-dimensional, making MLP a suitable and efficient choice. Compared to more complex deep learning models, MLP requires less computational power and is easier to train and interpret, which makes it ideal for academic projects.

Additionally, MLP demonstrated strong performance when combined with feature standardization, achieving high accuracy

---

9. Limitations and Future Work

- Handcrafted features may fail in complex real-world images.
- Future work could involve:
- Convolutional Neural Networks (CNNs)
- Larger and more diverse datasets
- Additional blur-related features

---

10. Conclusion

This project demonstrated that classical image processing techniques combined with machine learning can effectively detect image blur. The achieved results confirm that handcrafted features remain a strong baseline solution for blur detection tasks.

End of Report