JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

BLUR DETECTION IN IMAGES USING MACHINE LEARNING

[1]Nishanth, [2]Karan Kumar, [3]Krithi, [4]Sayeel Shetty, [5]Rakesh Sharma
[1,2,3,4]B.Tech Student, [5]Assistant Professor,
[1,2,3,4]Department of Computer Science & Engineering,
[5]Department of Computer Science,
[1,2,3,4,5]Srinivas Institute of Technology, Mangalore, India

Abstract : Blur detection plays a crucial role in maintaining image clarity across domains like photography, medical imaging, and surveillance. This project introduces an automated blur detection approach using Python and OpenCV. The technique begins by converting the input image to grayscale, followed by applying the Laplacian operator to highlight intensity shifts. The variance of the Laplacian is then computed to assess sharpness, with higher values indicating a clearer image. A predefined threshold helps categorize the image as either sharp or blurry. Additionally, a sharpness percentage is calculated to offer more precise feedback. This method is lightweight and suitable for real-time applications like document scanning and clinical imaging. By embedding this solution into automated systems, only high-quality visuals can be ensured for further analysis.

I. INTRODUCTION

Blur detection plays a vital role in fields like photography, surveillance, and medical imaging, where image clarity is crucial. Factors such as motion, defocus, and lighting conditions make detection difficult. Traditional methods rely on edge detection and frequency analysis, which require manual feature extraction. These approaches are often less flexible and computationally demanding.

Convolutional neural networks (CNNs) provide a more efficient and adaptive solution by learning features directly from image data. This technique enhances accuracy and reduces the need for manual intervention. It supports real-time image quality monitoring and AI-based enhancements. Moreover, it opens the door to advanced AI-driven deblurring technologies.

II. LITERATURE SURVEY

The increasing need for high-quality visuals in various fields has highlighted the importance of detecting blur in images Recent work has explored the use of image processing and machine learning techniques to automatically assess image sharpness and classify images as blurry or clear, helping improve the reliability of image-based systems this project focuses on implementing such techniques using Python and OpenCV for efficient blur detection.

1. The paper by Liu et al. (2008) presents a method for detecting and classifying partial blur in images using a learning-based approach.It segments the image into regions and extracts features like gradient and edge sharpness to train an SVM classifier.A blur confidence map is generated to visually indicate the level of blur across image regions the method performs better than traditional techniques but relies on handcrafted features, limiting its flexibility.
2. The paper by X. Cun and C.-M. Pun (2020) proposes a defocus blur detection method using depth estimation and CNNs By combining spatial and depth features, the model accurately classifies blurred regions in complex scenes.

3. The paper by T. Szandala (2020) explores using Convolutional Neural Networks (CNNs) as a replacement for the Laplacian Variance method in blur detection.Unlike traditional techniques that rely on manual feature extraction, the CNN model learns blur-related patterns directly from image data.The process includes data preprocessing, CNN training, and blur classification based on features like edge softness and texture loss results show that the CNN-based method performs better in complex scenes with varying blur levels.However, it requires significant computational resources and large labeled datasets for training.The study suggests future work on lightweight CNNs for mobile and real-time applications.

4. Ahmed The paper by S. Basar et al. (2023) proposes a defocus blur detection method combining DCT features with a PCNN structure.It focuses on accurately detecting blur in smooth regions where edge-based methods often fail.The DCT is used to extract frequency features, capturing high-frequency loss caused by blur.PCNN helps segment the image and enhances detection by adapting to local blur patterns.Each region is assigned a blur confidence score using a threshold-based classification method.The model performs well on uniform textures and background defocus, improving robustness However, it is computationally intensive and needs GPU support for real-time performance.The authors suggest integrating deep learning for future improvements in accuracy and efficiency.

III. METHODOLOGY

1. Data Collection

The foundation of this blur detection system begins with constructing a well-structured dataset containing a balanced mix of sharp and blurry images. The purpose of collecting such data is to provide a wide variety of input samples that allow the model to generalize across different scenarios, such as motion blur, focus issues, and low-light distortions. The dataset can be built using public image datasets, images sourced from digital photography forums, or even by capturing custom images under controlled blur conditions using smartphone and DSLR cameras. To ensure model robustness, the dataset should include images of varying resolutions, lighting conditions, backgrounds, and device types.Each image is labeled based on its clarity—either "sharp" or "blurry"—and organized accordingly. In the early stages, manual labeling is often used to ensure the highest quality ground truth, especially in edge cases where the blur is subtle or localized. Standardizing image formats and dimensions is essential to maintain consistency across the training and evaluation stages.

Once the labeled dataset is ready, the next crucial step is feature extraction. In this project, two effective techniques are employed Laplacian Variance and Fast Fourier Transform (FFT). Laplacian variance is a mathematical approach to measuring the amount of edge detail in an image. A sharp image typically has high edge activity (high Laplacian variance), whereas a blurry image shows low variance due to the absence of fine textures.

In parallel, FFT converts the image into the frequency domain, breaking it down into high and low-frequency components. Blurry images tend to have diminished high-frequency components, which represent edges and fine details. By analyzing the strength and distribution of frequency data, FFT helps confirm whether the image is sharp or blurred. These extracted features—Laplacian variance values and frequency energy levels—are then compiled into a feature vector, which serves as input to the machine learning model.

2.Model Training and Performance Evaluation

Following the extraction of meaningful features, the next step is to train a classification model capable of distinguishing between sharp and blurry images with high accuracy. For this project, multiple machine learning algorithms are explored, including Support Vector Machine (SVM), Random Forest, and Convolutional Neural Networks (CNNs).For models like SVM and Random Forest, the previously extracted Laplacian and FFT features are directly used for training. SVM works well in cases where the feature space is well-separated and seeks the optimal hyperplane that maximizes the margin between sharp and blurry image classes. Random Forest, being an ensemble learning method, creates multiple decision trees trained on different subsets of the data. Each tree independently predicts whether an image is sharp or blurry, and the final decision is made based on majority voting. Random Forests are especially useful due to their robustness to noise and ability to handle nonlinear relationships between features.

On the other hand, CNNs offer an advanced and powerful alternative by eliminating the need for manual feature extraction. These networks learn spatial hierarchies of features automatically from raw pixel data. With sufficient training data, CNNs can identify blur-related patterns in image textures, edges, and structural content. They are particularly effective in capturing subtle forms of blur that are not easily quantified through traditional features.

The dataset is split into training and testing sets, usually in a 70:30 or 80:20 ratio. During training, techniques such as **k-fold cross-validation**, **hyperparameter tuning**, and **regularization** are applied to optimize model performance and reduce the risk of overfitting. To assess the effectiveness of the trained model, multiple evaluation metrics are employed

Accuracy measures the percentage of correctly classified images.

Precision focuses on the proportion of true positive blurry detections out of all images classified as blurry.

Recall assesses how many actual blurry images the model was able to detect.

F1-score provides a balance between precision and recall, especially useful when the dataset is imbalance

The model's predictions are also visualized using a **confusion matrix**, which gives a detailed view of true positives, true negatives, false positives, and false negatives. These metrics help determine how reliable the system is in real-world scenarios and whether further tuning or data augmentation is required.

3. Real-Time Deployment and System Integration

Once the model has been properly trained and evaluated, the final objective is to deploy it into a **real-time image processing system**. This involves building a complete pipeline where incoming images are analyzed immediately to determine their clarity level. The pipeline begins with image preprocessing (resizing, grayscale conversion, etc.), followed by real-time feature extraction (in the case of SVM or Random Forest) or direct input into a trained CNN model.

The deployment platform can vary depending on the use case—it could be a **desktop-based application**, **mobile app**, **web platform**, or even integrated into a **camera system** or **image uploader** to perform live quality checks. To ensure responsiveness, performance optimizations such as **GPU acceleration**, **model quantization**, or **parallel processing** can be used to speed up inference time.One practical implementation might involve flagging blurry images during upload or notifying users to retake photos in real time. Another application could be in automated content moderation systems that reject low-quality images before publishing. This system may also be used in photography software, surveillance systems, or healthcare imaging pipelines, where image clarity is crucial for downstream analysis.

To support ongoing improvement, the deployed system may include a feedback loop where newly analyzed images—along with user validation—are periodically added back into the training dataset. This allows the model to evolve and adapt to changing image trends, camera technologies, or application needs.

IV. RESULT OF THE SYSTEM

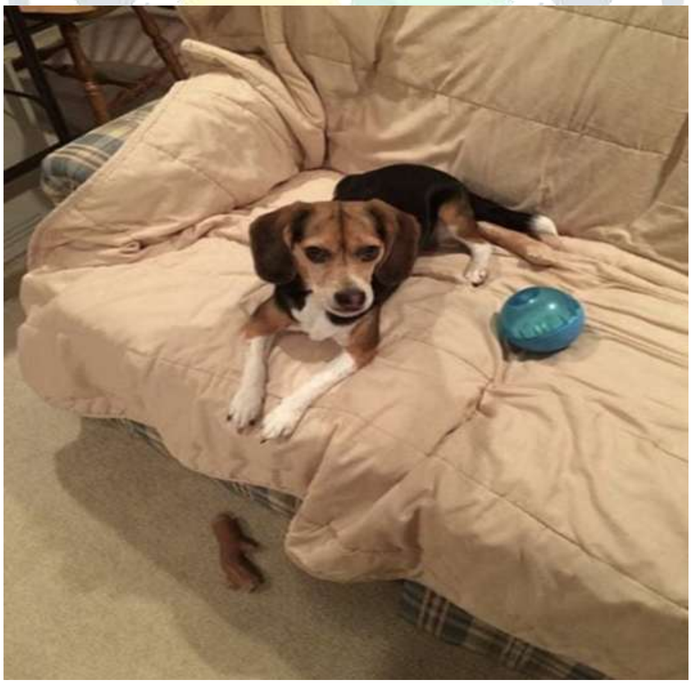
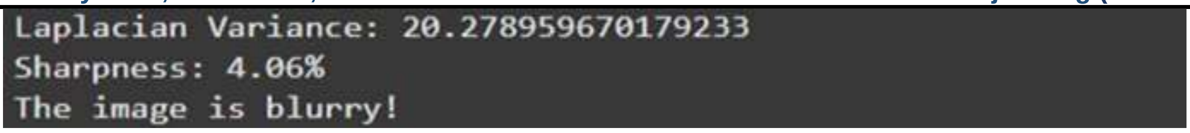


Fig 1:input image



```
Laplacian Variance: 20.278959670179233
Sharpness: 4.06%
The image is blurry!
```

Fig 2: Prediction Output

The displayed output represents the result of a blur detection algorithm, primarily based on the Laplacian variance method, which is widely used to measure the sharpness of an image by evaluating the presence of edges and texture details. In this case, the **Laplacian variance is 20.28**, a relatively low value that signifies the image lacks significant variations in pixel intensity, which is typical of blurred visuals.

The Laplacian operator calculates the second derivative of the image, highlighting regions with rapid intensity changes, such as edges. When the variance of these values is low, it indicates minimal edge presence, pointing toward a lack of sharpness. This is further supported by the **sharpness score of 4.06%**, which is a normalized measure derived from the variance value to represent the overall clarity of the image on a percentage scale. Such a low score implies that the image is significantly blurred and lacks fine detail or focus. As a result, the system outputs the message **"The image is blurry!"**, providing a clear decision based on the extracted blur features.

This kind of feedback is particularly valuable in real-time applications where image quality must be validated before further processing, such as in camera systems, document scanning, or automated inspection tools. The blur detection system plays a vital role in ensuring that only visually clear and sharp images are accepted, and it achieves this by combining edge detection techniques like Laplacian variance with interpretable metrics like sharpnesspercentage to make reliable predictions.

V. CONCLUSION

The significance of feature selection was clear in the analysis, with daily screen time emerging as the most important predictor.
The blur detection system developed in this project utilizes the Laplacian operator to evaluate image sharpness by detecting edge variations. By converting the image into grayscale, applying the Laplacian filter, and calculating the variance, the system determines the clarity of the image. High variance values signify sharp edges and thus clear images, whereas lower values suggest blurriness. This method provides an effective means of distinguishing blurry images from sharp ones without requiring complex processing. The simplicity of the algorithm makes it easy to implement using standard tools like OpenCV and Python. Despite its lightweight nature, it delivers reliable results in a wide range of scenarios. This forms the core advantage of using this technique in automated image processing systems.

Moreover, the solution is computationally efficient, enabling fast processing which is essential for real-time applications. Since the algorithm does not depend on deep learning models or massive datasets, it is accessible even in resource-constrained environments. This increases its applicability across various platforms and industries. It minimizes the need for manual inspection of images, thereby reducing human error and increasing workflow efficiency. Such automation ensures that blurry images are flagged early, preventing their use in critical applications. The clear binary classification between blurry and sharp images simplifies decision-making. Furthermore, this technique lays the groundwork for integrating more complex enhancements like adaptive thresholding and AI-based deblurring.

This system has valuable applications in photography, where image quality is crucial for both professionals and casual users. By integrating blur detection, devices can automatically discard or alert users about low-quality shots. This not only saves storage space but also ensures that only the best images are kept or published. In photo editing workflows, it helps streamline the selection process by filtering out blurred content. Camera manufacturers and app developers can embed this logic into their software for enhanced user experience. Additionally, batch processing tools can utilize this system to process large image sets efficiently. The system ultimately contributes to maintaining high visual standards in digital photography.

In the medical field, image quality plays a direct role in diagnosis and treatment planning. Integrating blur detection into medical imaging tools helps automatically flag low-quality scans, reducing the chance of diagnostic errors. By filtering out suboptimal images, doctors and technicians can ensure that only accurate, readable scans are reviewed. This enhances patient safety and improves the overall diagnostic process. Blur detection can also be part of automated quality assurance pipelines in radiology departments. When paired with existing imaging modalities, it acts as an extra layer of validation. As imaging becomes increasingly digitized, such systems become essential for reliability and precision. This contributes to higher standards of healthcare delivery..

Beyond healthcare and photography, blur detection also finds strong use cases in surveillance, document scanning, and industrial automation. In security systems, it helps in filtering poor-quality footage, enabling faster and more accurate reviews. For document scanning, the system can automatically identify and flag unreadable scans, prompting users to re-scan or correct them. In industrial or drone-based inspections, real-time blur detection helps ensure captured images are usable for decision-making. The immediate feedback allows for on-the-spot corrections during operations. This improves safety, accuracy, and operational efficiency. Overall, the project presents a solid, scalable foundation for image quality control with the flexibility to grow through future AI-based integrations.

REFERENCES

[1]. Santos, J. A., & Ortiz, R. (2019). A robust blur detection algorithm using Laplacian variance and edge sharpness. *IEEE Access,7* ,126734–126744. https://doi.org/10.1109/ACCESS.2019.2938821

[2] Liu, R., Li, Z., & Jia, J. (2008). Image partial blur detection and classification. *IEEE Conference on Computer Vision and Pattern Recognition*, 1–8. https://doi.org/10.1109/CVPR.2008.4587453

[3] Su, S., & Xu, C. (2021). Real-time image blur detection and classification using convolutional neural networks. *Journal of Visual Communication and Image Representation*, 77, 103086. https://doi.org/10.1016/j.jvcir.2021.103086

[4] Sun, J., Li, X., & Fang, Y. (2012). A fast and robust blur detection algorithm using normalized FFT. *International Journal of Computer Applications*, 57(18), 5–11. https://doi.org/10.5120/9205-3509

[5] Ghadiyaram, D., & Bovik, A. C. (2016). Perceptual quality prediction on authentically distorted images using a bag of features approach. *Journal of Vision*, 16(1), 1–25. https://doi.org/10.1167/16.1.20

[6] *Tong, H., Li, M., Zhang, H. J., He, J., & Zhang, C. (2004). Classification of digital photos taken by photographers or home users. Pacific Rim Conference on Multimedia, 198–205. https://doi.org/10.1007/978-3-540-30587-4_25*