

SmartClean-AI: Automated Data and Image Cleaning Libraries for Machine Learning Workflows

Yash Talati, Sumit Palekar, Aniket Kambale, Fayaj Shaikh, Ashwini Garkhedkar

Department of MCA, Progressive Education Society's Modern College of Engineering, Pune, Maharashtra, India

Published November 2026

Abstract

Data quality determines the reliability of analysis and machine learning, yet preparing raw data is repetitive and rarely documented. This paper presents SmartClean-AI, a research project that delivers two open-source Python libraries, `smartclean-ai` for structured data and `smartimageclean` for image datasets, both published on the Python Package Index under the MIT licence. `smartclean-ai` accepts a pandas DataFrame, profiles it, handles missing values, removes duplicate rows, and detects and caps outliers with the interquartile-range (IQR) rule, returning the cleaned data, before-and-after profiles and a cleaning report. `smartimageclean` validates image files, measures blur, brightness and resolution, detects exact and near-duplicate images, computes a dataset quality score, assigns KEEP, REVIEW or REMOVE decisions, standardizes eligible images, re-analyses them and writes CSV and JSON reports. In a recorded run on 45 images, 32 were initially acceptable and 13 were not; after standardization, seven of the 13 satisfied the requirements and six still needed review, and the image library's 26 automated tests passed. The libraries are rule-based and make every change visible; accuracy against annotated data and comparison with existing tools remain future work.

Keywords: data cleaning; duplicate detection; image quality; outlier detection; Python library

1 Introduction

The quality of a dataset decides the quality of every result built on it. Structured tables collected from surveys, sensors and business applications commonly contain missing values, repeated records and extreme values, while image collections gathered from the web, cameras and scraping tools contain corrupted files, blurred or badly exposed pictures, images of insufficient resolution, and duplicates. Practitioners report that preparing data takes a large share of the effort in an analytics or machine learning project [1, 2].

Students and analysts usually clean data with ad-hoc scripts written for one dataset. Such scripts are repetitive, rarely documented, and make it hard to say afterwards what was changed and why. Profiling tools describe data without repairing it, while repair tools may hide what they did. In image datasets, a single unreadable file can stop a training job, and near-duplicate images can leak between training and test sets.

SmartClean-AI addresses these problems with two Python libraries that share one design: run a modular pipeline, measure the data before and after, make every decision visible in a report, and leave the final dataset under the user's control. The contributions are: (1) `smartclean-ai`, which profiles, repairs, re-profiles and reports on a DataFrame in one call; (2) `smartimageclean`, which analyses, decides, standardizes and re-analyses an image dataset using a KEEP/REVIEW/REMOVE decision model; (3) the release of both libraries on PyPI under the MIT licence; and (4) an honest

account of results, including limitations. The libraries are rule-based and configuration-driven. They are not deep-learning classifiers, they do not interpret image content, and they do not guarantee restoration of damaged images.

The rest of the paper is organized as follows. Related work and the research gap are reviewed first, followed by the problem definition and the proposed methodology. The implementation is then described, results and testing are reported, and the final section concludes with limitations and future scope.

2 Literature review and research gap

2.1 Related work

Côté et al. review the relationship between data cleaning and machine learning and discuss how cleaning choices affect the models trained afterwards [1]. Ridzuan and Wan Zainon survey cleansing methods for big data, covering typical error classes and the techniques used to detect and correct them [2]. Haider et al. show that cleaning can be automated for data-centre data [3], and Shi et al. show that clinical knowledge improves automated cleaning of electronic health records [4], which illustrates that domain rules matter. ydata-profiling produces detailed dataset reports for data-centric AI [5], and AutoClean uses large language models to prepare training corpora [6]; the latter targets text corpora, not CSV or Excel tables.

For images, Bare et al. propose a deep no-reference quality model for super-resolved images [7], Li et al. use quality assessment to drive deblurring [8], and Chen et al. assess blurred-image quality from structure features [9]. Hu et al. [10], Sang et al. [11] and Dost et al. [12] survey or extend image quality assessment. Recent restoration research relies on learned diffusion models; smartimageclean does not implement learned restoration and is positioned as a rule-based dataset-cleaning tool.

2.2 Existing tools

Mature tools already cover parts of the workflow, so Table 1 only places the project and does not claim that other tools lack a capability.

Table 1. Existing tools and their relation to SmartClean-AI.

Tool	Established role	Relation
pandas	Data manipulation, missing values, duplicates [13]	smartclean-ai is built on pandas
ydata-profiling	Automated profiling [5]	Profiling plus repair and a change report
Great Expectations	Data validation through expectations	Validation-focused; smartclean-ai repairs automatically
OpenRefine	Interactive cleaning and transformation	GUI-oriented; smartclean-ai is programmatic
OpenCV	Image processing [14]	Used for pixel-level measurements
ImageHash	Perceptual image hashing [15]	Used for near-duplicate analysis
fastdup, CleanVision	Large-scale image issue detection	Benchmarking is future work
Albumentations	Image augmentation	Outside the project scope

2.3 Research gap

The review shows that individual capabilities already exist, so the gap is stated in terms of integration, transparency and evaluation and not as a new algorithm. *Profiling without repair*: profiling tools describe a dataset but leave repair to the user, while repair scripts rarely give a before-and-after description. *Lack of transparent reports*: many steps change data silently, whereas a library should record how many values were found, filled, removed or capped. *Fixed rules*: whether a dark photograph, a blurred image or an extreme value is a defect depends on the application, so decisions should be configurable and should separate acceptable, unacceptable and reviewable items. *Standardization without re-verification*: converting images to a common format or size does not prove that they meet the requirements. *Separate toolchains*: tables and images normally use different tools and report formats. Each gap is a design objective of this project and depends on the systematic comparison listed as future work.

3 Problem definition

Given a structured dataset T with N rows and P columns, produce a cleaned dataset T' in which missing values are handled, duplicate rows are removed and numerical outliers are treated, together with a report of every change. Given an image dataset $I = \{i_1, \dots, i_n\}$, assign to every image a decision in $\{\text{KEEP}, \text{REVIEW}, \text{REMOVE}\}$ based on validity, blur, brightness, resolution and duplicate analysis, apply supported standardization, re-analyse the result and report the outcome. In both cases the number of valid observations that are lost must be kept as small as possible.

4 Proposed methodology

4.1 System overview

smartclean-ai (import name smartclean) accepts a pandas DataFrame [13] and returns a CleaningResult object that holds the cleaned DataFrame, the before and after profiles and a cleaning report. smartimageclean (import name imagecleaner) accepts a dataset folder and returns decisions, quality scores, processed images and reports. As Figure 1 shows, the user supplies either a table or an image dataset, and each library runs its own pipeline and returns cleaned output together with a report. The two libraries are independent and share a design, not a runtime.

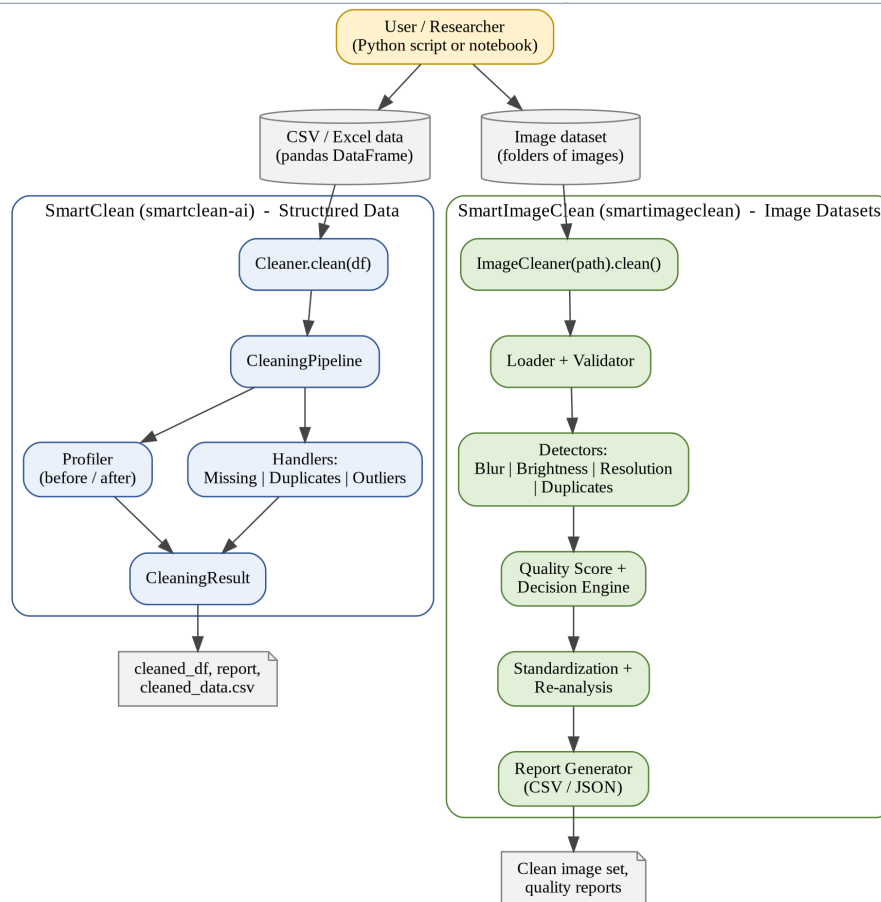


Figure 1. System architecture of SmartClean-AI: the structured-data pipeline of smartclean-ai (left) and the image-dataset pipeline of smartimageclean (right).

4.2 Structured-data cleaning (smartclean-ai)

The user loads a CSV or Excel file into a DataFrame and calls `Cleaner().clean(df)`. The Cleaning-Pipeline runs five stages in a fixed order: before-profiling, missing-value cleaning, duplicate cleaning, outlier cleaning and after-profiling. Profiling records the shape, missing counts and descriptive statistics of the data before and after cleaning, so that the effect of the pipeline can be compared.

Missing values. For each column the pipeline computes the missing share r , the number of null cells divided by the number of rows. A column whose r exceeds a drop threshold t is dropped and reported as skipped; otherwise the nulls are filled with the mean or median (numerical columns) or the most frequent value (categorical columns), as configured. The report records the missing values before cleaning, the number filled, the skipped columns and the strategy used for each column.

Duplicates. Exact duplicate rows are counted and removed, keeping the first occurrence, and the report records the number found and removed.

Outliers. The interquartile-range rule [16] is applied to every numerical column. With Q_1 and Q_3 the first and third quartiles and $IQR = Q_3 - Q_1$, the fences are

$$L = Q_1 - 1.5 \cdot IQR, U = Q_3 + 1.5 \cdot IQR \quad (1)$$

A value x outside $[L, U]$ is an outlier. Capping replaces it with the nearest fence,

$$x' = \min(\max(x, L), U) \quad (2)$$

which preserves the row and limits the influence of the extreme value; the library also supports removal. The multiplier 1.5 is the conventional threshold, not a universal rule, and a constant column (IQR = 0) produces no false outliers.

Result object. CleaningResult stores cleaned_df, before_profile, after_profile and report, and offers display_report() for a console summary and save_csv() for export without the index column.

4.3 Image-dataset cleaning (smartimageclean)

smartimageclean runs seven stages. (1) *Discovery and loading* locates the supported image files in the dataset folder. (2) *Validation and metadata* opens each file to check that it is readable and extracts format, dimensions and colour mode. (3) *Quality analysis* measures blur, brightness and resolution and compares each measurement with the configured requirement. (4) *Duplicate analysis* identifies exact duplicates and visually similar pairs. (5) *Scoring and decision* computes a dataset quality score and assigns KEEP, REVIEW or REMOVE to every image. (6) *Standardization and re-analysis* converts eligible images to a common technical format and analyses them again. (7) *Reporting* writes CSV and JSON reports. Algorithm 1 (Table 2) summarizes stages 1 to 5.

Table 2. Algorithm 1: Image analysis and decision.

1 files ← DISCOVER images in folder
2 for each file f in files:
3 if f cannot be opened:
4 decision[f] ← REMOVE; continue
5 meta ← (width, height, format, mode)
6 issues[f] ← {}
7 if blur(f) fails cfg: add "blur"
8 if brightness(f) fails cfg:
9 add "brightness"
10 if size(meta) fails cfg:
11 add "resolution"
12 dups, pairs ← FIND duplicates(files)
13 decision[f] ← DECIDE(issues, dups, cfg)
14 score ← QUALITY_SCORE(decision, issues)
15 write CSV and JSON reports

Resolution analysis is a dimension check,

$$W_{\min} \leq W \leq W_{\max}, H_{\min} \leq H \leq H_{\max} \quad (3)$$

where W and H are the image width and height. Brightness is judged from an intensity statistic of the greyscale image, and blur from a sharpness measure computed with OpenCV [14]. Exact duplicates are files with identical content. Near-duplicates are found by comparing perceptual hashes [15, 17]; for n-bit hashes h_1 and h_2 the Hamming distance is

$$d_H(h_1, h_2) = \sum_{k=1}^n 1[h_{1k} \neq h_{2k}] \quad (4)$$

and a pair whose distance is below a threshold is reported as similar. Because similarity depends on a threshold, similar pairs are presented for review and are not removed automatically.

Decisions. KEEP means that the image meets the active requirements; REMOVE means that it is not usable under the configuration, for example an unreadable file; REVIEW holds borderline cases for a person to decide. *Re-analysis.* With the --process option, eligible images are converted to the technical requirements (for example format, colour mode or dimensions, where supported) and analysed again with the same detectors. A processed image with no remaining issue becomes KEEP; otherwise it becomes REVIEW. Re-analysis gives evidence of what the processing achieved; it does not guarantee that blur, noise or exposure problems have been repaired.

5 Implementation

5.1 Technology stack and packaging

Both libraries are written in Python and are distributed as wheel and source distributions built from pyproject.toml. Table 3 summarizes the stack and the PyPI release information as on 14 October 2026 [19, 20]. Both packages have the development status 3 - Alpha.

Table 3. Technology stack and PyPI release information.

Item	smartclean-ai	smartimageclean
Import name	smartclean	imagecleaner
Python	3.10 or newer	3.13 or newer
Core libraries	pandas [13], NumPy [18]	OpenCV [14], Pillow, NumPy, pandas, ImageHash [15]
Testing and style	pytest	pytest, Ruff, Black
Version	1.1.0 (7 Oct 2026)	0.1.2 (7 Oct 2026)
Licence	MIT	MIT
Files	Wheel 8.9 kB, sdist 12.7 kB	Wheel 43.6 kB, sdist 36.1 kB

5.2 Package layout and design

The package smartclean contains the modules cleaning (missing values, duplicates, outliers), config, core (CleaningPipeline and profiling), handlers and utils, with cleaner.py as the public entry point and result.py defining CleaningResult. The package imagecleaner contains cleaner.py (ImageCleaner), cli.py and __main__.py for the command line, and the sub-packages config, core, detectors, validators, standardization, reports, exceptions and utils. The design uses five patterns: a pipeline of fixed, ordered stages; a facade (Cleaner, ImageCleaner) that hides the internal modules behind one call; a strategy for the imputation method and the outlier action (cap or remove); a result object that carries data, profiles and report together; and a separation of detection from decision, so that detectors only measure and the decision engine applies the configured rules.

5.3 Public interface

smartclean-ai is used as in Listing 1 (Table 4). smartimageclean is used from Python as ImageCleaner("datasets").clean() or from a terminal with python -m imagecleaner <folder> --process; python -m imagecleaner --help lists the flags of the installed version.

Table 4. Listing 1: Cleaning a table with smartclean-ai.

import pandas as pd
from smartclean import Cleaner
df = pd.read_csv("your_file.csv")
result = Cleaner().clean(df)
result.display_report()
result.save_csv("cleaned_data.csv")

6 Results and discussion

6.1 Experimental setup

The experiments ran on a Windows 10/11 (64-bit) machine with a dual-core CPU and 4 GB of RAM, using smartclean-ai 1.1.0 and smartimageclean 0.1.2. The image experiments used images from the Oxford-IIIT Pet dataset [21].

6.2 Data-cleaning results

Table 5 shows the report produced by smartclean-ai in an example run documented with the package. Twelve duplicate rows were found and removed, and 13,376 values in six numerical columns were capped at the IQR fences, leaving a dataset of 41,176 rows and 21 columns. The run shows that the report exposes every change. Capping keeps the rows but modifies values, including legitimate extreme ones, so a domain expert should review the capped columns.

Table 5. Example cleaning run of smartclean-ai.

Measure	Value
Duplicate rows found / removed	12 / 12
Columns containing outliers	6
Outlier values handled	13,376
Outlier method	IQR capping
Final dataset size	41,176 rows × 21 columns

6.3 Image-cleaning results

One end-to-end run of smartimageclean was recorded on a folder of 45 images (Table 6 and Figure 2). Thirteen of the 45 images (28.9%) were initially unacceptable. After standardization, 7 of these 13 (53.8%) satisfied the requirements and 6 (46.2%) still needed review; no exact-duplicate group and one similar pair were found. These proportions describe this single run. They show that the pipeline executes end to end and that standardization does not repair every defect, which is why the REVIEW

category is useful. They do not establish detector accuracy, agreement with human judgement or superiority over other tools.

Table 6. Recorded run of smartimageclean (45 images).

Measure	Result
Images discovered	45
Initially acceptable	32
Initially unacceptable	13
Processed through standardization	13
Acceptable after re-analysis	7
Still requiring review	6
Exact duplicate groups	0
Similar image pairs	1
Automated tests passed (pytest)	26

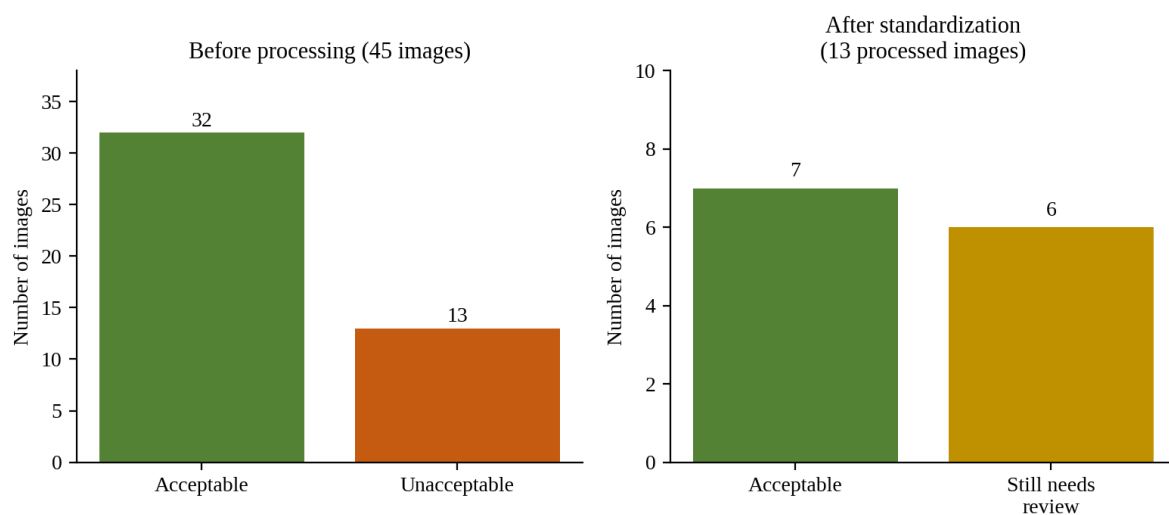


Figure 2. Outcome of the 45-image run: initial decisions (left) and re-analysis of the 13 processed images (right).

6.4 Performance

Table 7 gives the runtime and peak memory measured on the machine described above. A custom pandas script performing comparable cleaning operations ran in a median of 5.20 s on the $41,176 \times 21$ table, against 8.60 s for smartclean-ai, which additionally computes before and after profiles and a report.

Table 7. Runtime and peak memory.

Dataset	Size	Runtime (s)	Memory (MB)
Tabular, small	1,000 × 10	0.85	45
Tabular, large	41,176 × 21	8.60	180
Images	45 images	4.50	45

Dataset	Size	Runtime (s)	Memory (MB)
Images, larger set	1,473 images	20	153

6.5 Testing

Testing combines pytest unit tests, integration tests of the complete pipelines and edge-case tests. The test design includes missing-value imputation and column dropping, duplicate removal, IQR capping, CSV export, corrupted-image detection, low-resolution and blur detection, exact-duplicate detection and report creation. Integration cases run the full path from loading a CSV through `Cleaner().clean(df)`, `display_report()` and `save_csv()`, compare before and after profiles on a corrupted dataset, and run ImageCleaner and the command line with `--process` on a mixed-quality folder. Edge cases include an empty DataFrame, a column with all values missing, a constant numerical column, mixed text and numerical columns, and an empty or unreadable image folder. The recorded suite of the image library has 26 tests, all of which passed; passing tests show that the tested paths work, not that every case is covered.

6.6 Validation protocol

The results of smartclean-ai can be cross-checked with standard pandas operations on the original data (`df.isnull().sum()`, `df.duplicated().sum()` and the IQR formula for each numerical column), computed before cleaning. A stronger planned evaluation injects known numbers of missing values, duplicates and outliers into a clean dataset, runs the library, compares the output with the original and computes precision, recall and F1 for each defect type, where TP, FP and FN are the correctly repaired, wrongly repaired and unresolved defects:

$$\text{Precision} = \frac{TP}{TP + FP}, \text{ Recall} = \frac{TP}{TP + FN} \quad (5)$$

$$F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (6)$$

together with the number of valid rows lost. For images, a representative set is labelled for blur, brightness, resolution and duplicates, and the same measures are computed for each detector. A lower defect count alone is not sufficient evidence, because deleting valid data also lowers it.

7 Conclusion, limitations and future scope

SmartClean-AI delivers two reusable, open-source Python libraries, published on PyPI under the MIT licence, for cleaning structured data and image datasets. A single call of smartclean-ai profiles, repairs and reports, with a record of every change. Capping outliers instead of deleting rows, and the KEEP/REVIEW/REMOVE model for images, keep borderline items available for human judgement. In the recorded 45-image run, 13 unacceptable images were processed; 7 satisfied the requirements afterwards and 6 still needed review, so standardization helps but does not repair every defect. The libraries are rule-based, and claims about accuracy and superiority over other tools require the annotated experiments described above.

Limitations. The image evaluation covers 45 images from a single run without human annotation, so the results cannot be generalized. Automatic detection and correction of data types, categorical encoding and semantic errors are not part of smartclean-ai 1.1.0. The cleaning rules (imputation, IQR threshold, blur and brightness thresholds) are generic and may not suit every domain, and legitimate

extreme values can be capped. The libraries have not been benchmarked against pandas scripts, ydata-profiling, CleanVision or fastdup, and scalability to very large datasets is not established. smartimageclean does not understand image content, and both packages are in Alpha status.

Future scope. Planned work includes automatic data-type detection, categorical encoding and preprocessing recommendations for smartclean-ai; visualizations and a web interface; integration of the image library into one toolkit; machine-learning-assisted cleaning, confidence scores for suggested repairs and no-reference image quality models; evaluation on larger annotated datasets with precision, recall and F1 per defect type and comparison with existing tools; and downstream experiments that train the same model on original and cleaned data.

Acknowledgment

The authors thank Prof. Dr. K. R. Joshi (Principal), Prof. Dr. Shivani A. Budhkar (Head, Department of MCA) and especially their project guide, Prof. Ashwini Garkhedkar, of Progressive Education Society's Modern College of Engineering, Pune, for their guidance and support.

References

- [1] Côté PO, Nikanjam A, Ahmed N, Humeniuk D, Khomh F. Data cleaning and machine learning: A systematic literature review. *Automated Software Engineering*. 2024;31(2):54. <https://doi.org/10.1007/s10515-024-00453-w>
- [2] Ridzuan F, Wan Zainon WMN. A review on data cleansing methods for big data. *Procedia Computer Science*. 2019;161:731–738. <https://doi.org/10.1016/j.procs.2019.11.177>
- [3] Haider SN, Zhao Q, Meran BK. Automated data cleaning for data centers: A case study. In: *Proceedings of the 39th Chinese Control Conference (CCC)*. 2020. <https://doi.org/10.23919/CCC50068.2020.9189357>
- [4] Shi X, et al. An automated data cleaning method for electronic health records by incorporating clinical knowledge. *BMC Medical Informatics and Decision Making*. 2021;21:267. <https://doi.org/10.1186/s12911-021-01630-7>
- [5] Clemente F, et al. ydata-profiling: Accelerating data-centric AI with high-quality data. *Neurocomputing*. 2023;554:126585. <https://doi.org/10.1016/j.neucom.2023.126585>
- [6] Shen X, Hu S, Zhang X, Han X, Meng X, Wei J, Liu Z, Sun M. AutoClean: LLMs can prepare their training corpus. In: *Proceedings of NAACL-HLT 2025 (System Demonstrations)*. 2025. p. 85–95. <https://doi.org/10.18653/v1/2025.naacl-demo.9>
- [7] Bare B, et al. A deep learning based no-reference image quality assessment model for single-image super-resolution. In: *Proceedings of IEEE ICASSP*. 2018. <https://doi.org/10.1109/ICASSP.2018.8461931>
- [8] Li A, Li J, Lin Q, Ma C. Deep image quality assessment driven single image deblurring. In: *Proceedings of IEEE ICME*. 2020. <https://doi.org/10.1109/ICME46284.2020.9102899>
- [9] Chen J, Li S, Lin L, Wan J, Li Z. No-reference blurred image quality assessment method based on structure of structure features. *Signal Processing: Image Communication*. 2023;118:117008. <https://doi.org/10.1016/j.image.2023.117008>
- [10] Hu B, Li L, Wu J, Qian J. Subjective and objective quality assessment for image restoration: A critical survey. *Signal Processing: Image Communication*. 2020;85:115839. <https://doi.org/10.1016/j.image.2020.115839>
- [11] Sang Q, Li Q, Liu L, Deng Z, Wu X, Bovik AC. No-reference image quality assessment leveraging GenAI images. *IEEE Transactions on Image Processing*. 2025. <https://doi.org/10.1109/TIP.2025.3610238>
- [12] Dost S, Saud F, Shabbir M, Khan MG, Shahid M, Lovstrom B. Reduced reference image and video quality assessments: Review of methods. *EURASIP Journal on Image and Video Processing*. 2022;2022:1. <https://doi.org/10.1186/s13640-021-00578-y>
- [13] McKinney W. Data structures for statistical computing in Python. In: *Proceedings of the 9th Python in Science Conference*. 2010. p. 56–61.

-
- [14] Bradski G. *The OpenCV library*. Dr. Dobb's Journal of Software Tools. 2000.
 - [15] ImageHash. *Python package*. <https://pypi.org/project/ImageHash/> (accessed 14 October 2026).
 - [16] Tukey JW. *Exploratory data analysis*. Reading (MA): Addison-Wesley; 1977.
 - [17] Zauner C. *Implementation and benchmarking of perceptual image hash functions* [MSc thesis]. Hagenberg, Austria: Upper Austria University of Applied Sciences; 2010.
 - [18] Harris CR, et al. Array programming with NumPy. *Nature*. 2020;585:357–362.
 - [19] smartclean-ai 1.1.0. Python Package Index. <https://pypi.org/project/smartclean-ai/> (accessed 14 October 2026).
 - [20] smartimageclean 0.1.2. Python Package Index. <https://pypi.org/project/smartimageclean/> (accessed 14 October 2026).
 - [21] Parkhi OM, Vedaldi A, Zisserman A, Jawahar CV. Cats and dogs. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012. p. 3498–3505. <https://doi.org/10.1109/CVPR.2012.6248092>