Computation and Performance Studies
Volume 1 · Number 1 · 2026

Faster on Which Platform? Heterogeneous Timing Gains in an Image-Quality Toolbox

Noah Crawford[1] · Samuel Lawson[2] · Benjamin Lewis[3]

[1] Romania · [2] Serbia · [3] Slovakia

Original publication: October 28, 2026 | Article 642

Abstract

The phrase GPU acceleration can conceal a very uneven computational tradeoff. In seventy-four paired timing records from a pinned public image-quality toolbox, the geometric CPU-to-GPU time ratio was 8.98, but four metrics were slower in the GPU column. The two platforms reversed 682 of 2,701 pairwise speed comparisons. Larger reported GPU memory allocation was associated with a larger timing ratio, without identifying why. These findings favor platform-specific feasibility checks over a single acceleration factor for an entire assessment pipeline.

Secondary analysis of published platform measurements. The reported timings were not collected on this computer and do not measure energy consumption or image-quality accuracy.

Faster on Which Platform? Heterogeneous Timing Gains in an Image-Quality Toolbox

> Secondary analysis of published platform measurements. The reported timings were not collected on this computer and do not measure energy consumption or image-quality accuracy.

Abstract

The phrase GPU acceleration can conceal a very uneven computational tradeoff. In seventy-four paired timing records from a pinned public image-quality toolbox, the geometric CPU-to-GPU time ratio was 8.98, but four metrics were slower in the GPU column. The two platforms reversed 682 of 2,701 pairwise speed comparisons. Larger reported GPU memory allocation was associated with a larger timing ratio, without identifying why. These findings favor platform-specific feasibility checks over a single acceleration factor for an entire assessment pipeline.

1. A deployment choice hidden inside a metric choice

An image-quality metric can be chosen because its scores align with human judgments, but the chosen score must still be computed under a particular resource constraint. A pipeline processing a small number of images on an existing workstation faces a different question from a service already equipped with a GPU. The availability of an accelerated implementation does not establish that acceleration benefits every scoring rule or that the fastest choices retain their order across execution platforms.

This study asks how uniform published CPU-to-GPU timing gains are across quality metrics and whether their speed rankings transfer between the two reported platform columns. We examine a pinned public toolbox report with eighty method rows, rather than fitting a quality predictor. The principal evidence consists of paired time ratios, exact changes in speed ordering, and illustrative feasibility screens. A regression then summarizes how the log ratio varies with the report's allocated-memory measure.

The multi-scale two-stream assessment architecture in Meng, 2026 motivates attention to the computational consequences of quality measurement. The present report contains no execution of that architecture, so it supplies no estimate of its latency, memory use, or relative performance. It instead addresses a practical prerequisite for comparing assessment systems: an efficiency claim should identify the hardware, measured boundary, candidate implementation, and information omitted from the timing record.

We examine two descriptive propositions. A strong overall timing gain can coexist with individual metrics that gain little or become slower. In addition, a platform change can alter the order in which metric implementations satisfy a time budget. The association with allocated memory is exploratory and has no prespecified architectural mechanism. Memory, operator mix, implementation choices, and platform properties can move together without making any one variable an identified cause.

2. Computational cost is a measured outcome

The Roofline framework Williams et al., 2009 explains why attainable computational performance depends on more than an algorithm's nominal operation count. The present table does not contain the quantities required to fit a Roofline model, but that systems perspective motivates retaining platform identity when reporting cost. Rigorous benchmarking practice Kalibera and Jones, 2013 and demonstrations of measurement sensitivity Mytkowicz et al., 2009 likewise caution against turning a single published environment into a universal speed multiplier.

The implementations use the PyTorch ecosystem Paszke et al., 2019, yet the framework name alone does not standardize input loading, synchronization, preprocessing, precision, or device residency. These details determine the boundary of a timing measurement. We therefore inspect the acquired timing helper as documentation, retain its limitations, and call the outcomes reported path-call wall times. No result is labeled pure kernel time, steady-state server throughput, or end-to-end user response latency.

Quality metrics also represent different computational objects. Perceptual feature distances Zhang et al., 2018, joint structure-texture comparisons Ding et al., 2020, and multi-scale transformers Ke et al., 2021 need not execute similar workloads. Neural image assessment Talebi and Milanfar, 2017, multidimensional attention Yang et al., 2022, and language-image scoring Wang et al., 2022 add further variation. Their underlying publications identify the methods; the numeric timing evidence comes from the captured toolbox report rather than from timing claims reconstructed from those papers.

Statistical approaches such as NIQE Mittal et al., 2013 and spatial-domain assessment Mittal et al., 2012 offer other score constructions, while feature similarity Lin Zhang et al., 2011 provides another comparison rule. A report spanning these approaches should not be interpreted as a randomized comparison between architectural families. Multiple rows may reuse components or vary only in training data and settings. Both the descriptive denominator and the covariance grouping must acknowledge that a list of identifiers is not a list of wholly independent scientific ideas.

3. Source scope and a caution about its current helper

The source is the efficiency CSV maintained by Chen and Mo, 2022, captured at repository commit d47b7ac72ed25388371d0b637076c125cdb95a3c. The acquisition manifest stores the CSV, README, timing helper, license, hashes, and source URLs. We compare columns named cpu_xe6150 and cuda_v100. Other CPU columns are retained in the source snapshot but do not enter this analysis. Their presence does not provide repeated measurements of the two selected platforms.

The README describes average timing on 1080 by 800 image inputs. The accompanying helper creates a metric outside the timed region, calls it using image paths, synchronizes CUDA around each call, and omits the first image's timing before averaging. The metric receives the same image path in both arguments. This documentary description is narrower than evaluating each metric on its normal scientific comparison task, and it does not establish how every historical row was generated.

A further source limitation is consequential. The current helper calls CUDA synchronization unconditionally, including its CPU branch, and does not document how the Apple CPU columns were produced. We did not run it to reproduce the report, repair it silently, or claim that it would generate every stored cell on its named device. Our empirical object remains the published table. A deployment-grade replication would need environment-specific scripts and repeated measurements beyond this captured evidence.

The memory helper divides allocated bytes by 1024 cubed. Although the CSV header labels the field GB, we use its operational binary unit, GiB. It is a peak allocated-GPU-memory measure, not the process's full reserved memory, total workstation RAM, or a direct count of model weights. Its numerical scale cannot be freely exchanged with a storage measurement, consistent with the importance of defining scales Stevens, 1946 and maintaining numeric conventions Goldberg, 1991.

4. Constructing paired measures and dependence groups

Seventy-four rows have positive values for both selected time columns and the memory field. The six excluded identifiers are MSSWD, DEEPDC, Q-Align, Q-Align 8-bit, Q-Align 4-bit, and Compare2Score, as spelled in the retained source files. They are omitted because a required paired field is missing, not because their measured GPU performance is poor. In particular, this design cannot estimate a CPU-to-GPU ratio for the excluded language-model-based scores.

For each retained identifier, the time ratio divides its CPU seconds by its GPU seconds. Values exceeding one indicate a shorter reported GPU time; values below one indicate the reverse. Model 1 fits an intercept to the natural log ratios, making its exponential an equal-record geometric mean. This transformation gives reciprocal relative changes a coherent direction and reduces the influence of extremely large ratios compared with averaging the untransformed gains.

Model 2 regresses log time ratio on log allocated GPU memory. Its slope is an observational association among captured implementations. The ratio includes GPU time in its denominator, and both variables may share dependence on implementation details. We do not interpret the fitted coefficient as an intervention that changes memory while holding all else fixed. Reporting correlation without agreement or causal interpretation remains a relevant measurement caution Bland and Altman, 1986.

Thirty-five declared groups combine related identifier spellings, including eight ARNIQA rows, seven TOPIQ rows, six LPIPS rows, and four each for several other families. The complete grouping is saved with the results. Family-clustered covariance uses a small-sample correction and a t reference with thirty-four degrees of freedom Cameron and Miller, 2015. Heteroskedasticity-robust reasoning White, 1980 underlies the need to avoid a common-variance assumption, but clustering does not remove dependencies shared across all metrics through the same benchmark conditions.

Figure 1 maps platform columns to the paired ratios and downstream descriptions. The regressions contain seventy-four records; the thirty-five groups determine the cluster reference distribution. The source has no per-run timing distribution, so these intervals characterize conditional cross-family heterogeneity under a working independence assumption. They are not confidence intervals for repeated measurements on a newly purchased CPU or GPU. All transformations were executed locally using array and scientific-computing tools Harris et al., 2020 and Virtanen et al., 2020.

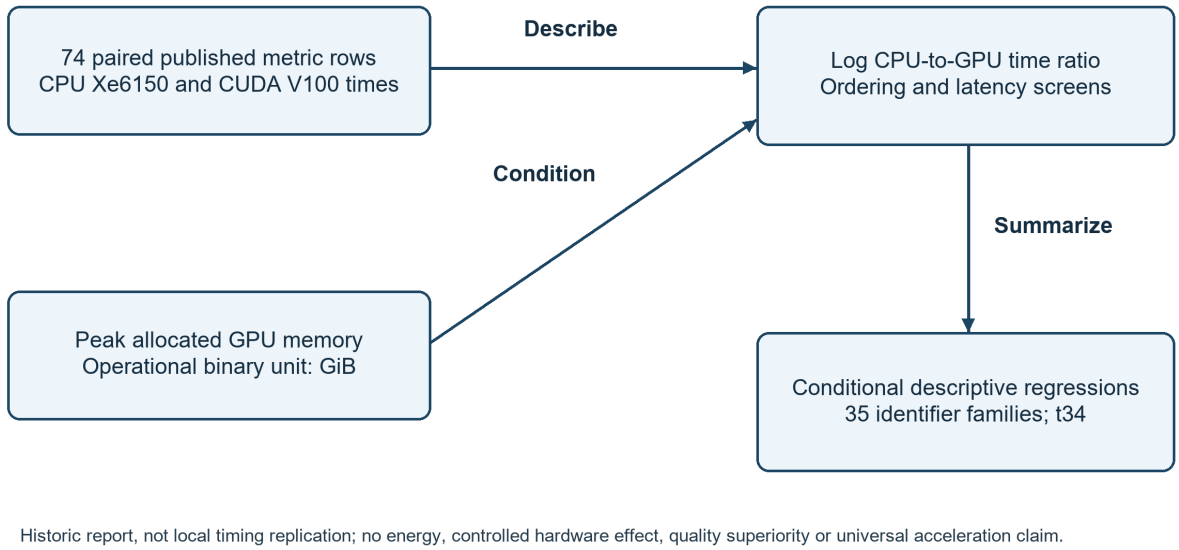


Figure 1. The same captured metric identifiers link two reported platform times, with allocated memory as a separate descriptive covariate and family groups defining conditional uncertainty.

5. A large central gain with meaningful exceptions

The estimated mean log time ratio is 2.195378, corresponding to a geometric ratio of 8.983392. Its conditional log-scale interval, 1.831339 to 2.559416, transforms to 6.242242 to 12.928260. These values summarize the retained record mixture. They should not be described as a nearly ninefold gain expected for any arbitrary future metric, because the underlying ratios range from 0.471541 to 185.329984.

Four entries have a ratio below one. PSNR reports 0.0416 CPU seconds and 0.0443 GPU seconds; PSNR-Y reports 0.0436 and 0.0461. CNNIQA reports 0.2038 and 0.4322, and CW-SSIM reports 0.8336 and 0.9904. We retain these exceptions without attributing them to launch overhead, preprocessing, or implementation defects that were not separately measured. At the opposite extreme, AHIQ has a reported ratio of 185.329984, with TReS variants also exceeding one hundred.

Table 1. Conditional OLS descriptions of published log timing ratios. *M1 contains an intercept; M2 includes log peak allocated GPU memory in GiB. Both retain the same seventy-four paired rows. Standard errors and intervals use thirty-five declared identifier families and t34; they do not estimate the variability of repeated hardware timings.*

Model	Term	Estimate (SE)	95% CI	N
M1	Mean outcome/contrast	2.1954 (0.1791)	[1.8313, 2.5594]	74
M1: fit	R-squared = 0.0000	-	-	74
M2	Intercept at 1 GiB	2.7215 (0.1531)	[2.4103, 3.0326]	74
M2	Log allocated GPU memory, GiB	0.7441 (0.0900)	[0.5612, 0.9270]	74
M2: fit	R-squared = 0.5217	-	-	74

Model labels: M1 = Mean log reported CPU-to-GPU timing ratio; M2 = Memory association with log reported timing ratio.

The log-memory coefficient is 0.744085, with conditional interval 0.561208 to 0.926963. Model 2 has R-squared 0.521749. Larger allocated-memory entries tend to have larger CPU-to-GPU timing ratios in

this table. That association is compatible with several computational explanations, but the report lacks the operator-level measurements needed to distinguish them. A high fitted share of variation is therefore descriptive, rather than evidence that increasing memory allocation improves acceleration.

6. Platform-specific speed order and feasibility

Among 2,701 unordered pairs of the seventy-four retained identifiers, 682 reverse their strict time ordering between the selected platform columns. Three pairs are tied on at least one platform at the recorded precision. The reversal fraction is 0.252499. This is an exact finite comparison of displayed values, and the pair count is not treated as 2,701 independent experimental observations. Every implementation contributes to many pairs.

The largest normalized absolute rank movement is 0.726027, using the seventy-three-position range as the denominator. This confirms that a time-based shortlist can change substantially even when almost all rows become faster. It also shows why an average acceleration ratio cannot replace the actual time column used in a decision. A method can gain strongly relative to its own CPU implementation while remaining slower than many other GPU implementations.

Illustrative time screens make the distinction visible. At 0.05 seconds per reported call, two CPU entries and twenty-one GPU entries pass; both CPU passers also pass on GPU. At 0.10 seconds, the counts are seven and forty-nine. At 0.25 seconds, twenty-one CPU entries and sixty-four GPU entries pass, with twenty satisfying both. At one second, forty-seven CPU entries pass and all seventy-four GPU entries pass. These cutoffs are declared examples, not real-time standards or service-level commitments.

None of these screens includes an accuracy requirement. Picture-quality assessment resources Ying et al., 2019, smartphone photography judgments Fang et al., 2020, and KonIQ-10k Hosu et al., 2019 define different evaluation settings, while TID2013 Ponomarenko et al., 2015 and PIPAL Gu et al., 2020 add other quality-comparison contexts. Their existence underscores the need to link the relevant scientific accuracy evidence separately. Passing a computational threshold alone does not establish that the selected score measures what a downstream application requires.

The reciprocal of a displayed call time is not automatically a sustainable service throughput. Concurrent requests, batching, shared memory, and overlapping transfers can change the relationship between latency and completed images per second. The captured rows do not report those operating conditions or a load curve. Keeping the screen in seconds per reported call therefore preserves the available measurement, while a throughput commitment would require additional observations under the intended request pattern and resource-sharing policy.

7. What this evidence can support operationally

The practical use of the report is to identify implementations for a local replication, not to issue a universal hardware recommendation. Before procurement or deployment, a researcher should repeat timings on the intended image sizes and workload, specify whether decoding and transfers are included, and inspect both central and tail latency. Those additional measurements were not performed here. Published time ratios can guide that exercise without being converted into guaranteed performance.

The extremes also deserve retained visibility rather than automatic deletion. Influence analysis Cook, 1977 provides a general reason to inspect observations that can shape fitted summaries, but this study does not claim a completed Cook-distance diagnostic or remove a row on that basis. Log transformation and the explicit ratio range keep both central tendency and extremes visible. Architectural attributions would require a separate experimental design with measured computational mechanisms.

Reusability depends on documentation as well as a convenient CSV. Dataset documentation principles Gebru et al., 2021 and the FAIR guidance Wilkinson et al., 2016 motivate retaining the exact report, source conditions, field definitions, and exclusions. Our local manifest distinguishes a stored public measurement from a newly executed benchmark. It also preserves the repository's actual license text; public availability does not establish unrestricted rights for every underlying dataset, image, model, or implementation.

The evidence does not measure energy efficiency. Seconds cannot be converted into joules without power information, and peak GPU allocation cannot stand in for power draw. It also does not compare a controlled GPU intervention with an otherwise identical CPU system. The named platform columns bundle device, software, and implementation conditions. These limits matter because a computational advantage stated without its measurement boundary can misdirect an otherwise well-designed quality-assessment study.

8. Conclusion

A strong geometric timing gain and widespread GPU feasibility coexist with four slower GPU entries and hundreds of reversed pairwise speed comparisons. The captured report therefore supports platform-specific computational screening, while leaving repeated-run uncertainty, accuracy, and energy outside its evidence. For quality-assessment research, efficiency should be reported as a measured relationship between an implementation, workload, and environment, with its exceptions retained alongside the central gain.

References

1. Paszke, Adam; Gross, Sam; Massa, Francisco; Lerer, Adam; Bradbury, James; Chanan, Gregory; Killeen, Trevor; Lin, Zeming; Gimelshein, Natalia; Antiga, Luca; Desmaison, Alban; Köpf, Andreas; Yang, Edward; DeVito, Zach; Raison, Martin; Tejani, Alykhan; Chilamkurthy, Sasank; Steiner, Benoit; Fang, Lu; Bai, Junjie; Chintala, Soumith. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. https://arxiv.org/abs/1912.01703.
2. Ying, Zhenqiang, et al. "From Patches to Pictures (PaQ-2-PiQ): Mapping the Perceptual Space of Picture Quality." *arXiv preprint arXiv:1912.10088*, 2019. https://arxiv.org/abs/1912.10088.
3. Gebru, Timnit, et al. "Datasheets for datasets" *Communications of the ACM*, 2021. https://doi.org/10.1145/3458723.
4. Meng, Zijie. (2026). Decoupling Semantics from Distortions: Multi-Scale Two-Stream Vision-Language Alignment for AI-Generated Image Quality Assessment. arXiv:2606.16799. https://arxiv.org/abs/2606.16799.
5. Cameron, A. Colin, and Douglas L. Miller. “A Practitioner’s Guide to Cluster-Robust Inference.” *Journal of Human Resources*, vol. 50, no. 2, 2015, pp. 317-372. https://doi.org/10.3368/jhr.50.2.317.
6. Stevens, S. S.. (1946). On the Theory of Scales of Measurement. Science. https://doi.org/10.1126/science.103.2684.677.
7. Cook, R. Dennis. (1977). Detection of Influential Observation in Linear Regression. Technometrics. https://doi.org/10.1080/00401706.1977.10489493.
8. Wang, Jianyi, et al. "Exploring CLIP for Assessing the Look and Feel of Images." *arXiv preprint arXiv:2207.12396*, 2022. https://arxiv.org/abs/2207.12396.
9. Wilkinson, Mark D.; Dumontier, Michel; Aalbersberg, IJsbrand Jan; Appleton, Gabrielle; Axton, Myles; Baak, Arie; Blomberg, Niklas; Boiten, Jan-Willem; da Silva Santos, Luiz Bonino; Bourne, Philip E.; Bouwman, Jildau; Brookes, Anthony J.; Clark, Tim; Crosas, Mercè; Dillo, Ingrid; Dumon, Olivier; Edmunds, Scott; Evelo, Chris T.; Finkers, Richard; Gonzalez-Beltran, Alejandra; Gray, Alasdair J.G.; Groth, Paul; Goble, Carole; Grethe, Jeffrey S.; Heringa, Jaap; ’t Hoen, Peter A.C; Hooft, Rob; Kuhn, Tobias; Kok, Ruben; Kok, Joost; Lusher, Scott J.; Martone, Maryann E.; Mons, Albert; Packer, Abel L.; Persson, Bengt; Rocca-Serra, Philippe; Roos, Marco; van Schaik, Rene; Sansone, Susanna-Assunta; Schultes, Erik; Sengstag, Thierry; Slater, Ted; Strawn, George; Swertz, Morris A.; Thompson, Mark; van der Lei, Johan; van Mulligen, Erik; Velterop, Jan; Waagmeester, Andra; Wittenburg, Peter; Wolstencroft, Katherine; Zhao, Jun; Mons, Barend. (2016). The FAIR Guiding Principles for scientific data management and stewardship. Scientific Data. https://doi.org/10.1038/sdata.2016.18.

10. Ponomarenko, Nikolay; Jin, Lina; Ieremeiev, Oleg; Lukin, Vladimir; Egiazarian, Karen; Astola, Jaakko; Vozel, Benoit; Chehdi, Kacem; Carli, Marco; Battisti, Federica; Jay Kuo, C.-C.. (2015). Image database TID2013: Peculiarities, results and perspectives. Signal Processing: Image Communication. https://doi.org/10.1016/j.image.2014.10.009.
11. Yang, Sidi, et al. "MANIQA: Multi-dimension Attention Network for No-Reference Image Quality Assessment." *arXiv preprint arXiv:2204.08958*, 2022. https://arxiv.org/abs/2204.08958.
12. Mittal, A., et al. "Making a “Completely Blind” Image Quality Analyzer" *IEEE Signal Processing Letters*, 2013. https://doi.org/10.1109/lsp.2012.2227726.
13. Ke, Junjie, et al. "MUSIQ: Multi-scale Image Quality Transformer" *arXiv preprint arXiv:2108.05997*, 2021. https://arxiv.org/abs/2108.05997.
14. Zhang, Richard, et al. "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric" *arXiv preprint arXiv:1801.03924*, 2018. https://arxiv.org/abs/1801.03924.
15. Lin Zhang, ; Lei Zhang, ; Xuanqin Mou, ; Zhang, D.. (2011). FSIM: A Feature Similarity Index for Image Quality Assessment. IEEE Transactions on Image Processing. https://doi.org/10.1109/tip.2011.2109730.
16. Fang, Yuming; Zhu, Hanwei; Zeng, Yan; Ma, Kede; Wang, Zhou. (2020). Perceptual Quality Assessment of Smartphone Photography. CVPR 2020, 3677-3686. https://openaccess.thecvf.com/content_CVPR_2020/html/Fang_Perceptual_Quality_Assessment_of_Smartphone_Photography_CVPR_2020_paper.html.
17. Bland, J. Martin, and Douglas G. Altman. "Statistical Methods for Assessing Agreement between Two Methods of Clinical Measurement." *The Lancet*, vol. 327, no. 8476, 1986, pp. 307-310. https://doi.org/10.1016/S0140-6736(86)90837-8.
18. Chen, Chaofeng; Mo, Jiadi. (2022). IQA-PyTorch: PyTorch Toolbox for Image Quality Assessment. Software repository. https://github.com/chaofengc/IQA-PyTorch/tree/d47b7ac72ed25388371d0b637076c125cdb95a3c.
19. Mytkowicz, Todd; Diwan, Amer; Hauswirth, Matthias; Sweeney, Peter F.. (2009). Producing wrong data without doing anything obviously wrong!. Proceedings of the 14th international conference on Architectural support for programming languages and operating systems. https://doi.org/10.1145/1508244.1508275.
20. White, Halbert. “A Heteroskedasticity-Consistent Covariance Matrix Estimator and a Direct Test for Heteroskedasticity.” *Econometrica*, vol. 48, no. 4, 1980, pp. 817-838. https://doi.org/10.2307/1912934.
21. Harris, Charles R.; Millman, K. Jarrod; van der Walt, Stéfan J.; Gommers, Ralf; Virtanen, Pauli; Cournapeau, David; Wieser, Eric; Taylor, Julian; Berg, Sebastian; Smith, Nathaniel J.; Kern, Robert; Picus, Matti; Hoyer, Stephan; van Kerkwijk, Marten H.; Brett, Matthew; Haldane, Allan; del Río, Jaime Fernández; Wiebe, Mark; Peterson, Pearu; Gérard-Marchant, Pierre; Sheppard, Kevin; Reddy, Tyler; Weckesser, Warren; Abbasi, Hameer; Gohlke, Christoph; Oliphant, Travis E.. (2020). Array programming with NumPy. Nature. https://doi.org/10.1038/s41586-020-2649-2.
22. Ding, Keyan, et al. "Image Quality Assessment: Unifying Structure and Texture Similarity." *arXiv preprint arXiv:2004.07728*, 2020. https://arxiv.org/abs/2004.07728.
23. Kalibera, Tomas; Jones, Richard. (2013). Rigorous benchmarking in reasonable time. Proceedings of the 2013 international symposium on memory management. https://doi.org/10.1145/2464157.2464160.
24. Goldberg, David. (1991). What every computer scientist should know about floating-point arithmetic. ACM Computing Surveys. https://doi.org/10.1145/103162.103163.
25. Gu, Jinjin; Cai, Haoming; Chen, Haoyu; Ye, Xiaoxing; Ren, Jimmy; Dong, Chao. (2020). PIPAL: a Large-Scale Image Quality Assessment Dataset for Perceptual Image Restoration. https://arxiv.org/abs/2007.12142.

26. Virtanen, Pauli; Gommers, Ralf; Oliphant, Travis E.; Haberland, Matt; Reddy, Tyler; Cournapeau, David; Burovski, Evgeni; Peterson, Pearu; Weckesser, Warren; Bright, Jonathan; van der Walt, Stéfan J.; Brett, Matthew; Wilson, Joshua; Millman, K. Jarrod; Mayorov, Nikolay; Nelson, Andrew R. J.; Jones, Eric; Kern, Robert; Larson, Eric; Carey, C J; Polat, İlhan; Feng, Yu; Moore, Eric W.; VanderPlas, Jake; Laxalde, Denis; Perktold, Josef; Cimrman, Robert; Henriksen, Ian; Quintero, E. A.; Harris, Charles R.; Archibald, Anne M.; Ribeiro, Antônio H.; Pedregosa, Fabian; van Mulbregt, Paul; SciPy 1.0 Contributors; Vijaykumar, Aditya; Bardelli, Alessandro Pietro; Rothberg, Alex; Hilboll, Andreas; Kloeckner, Andreas; Scopatz, Anthony; Lee, Antony; Rokem, Ariel; Woods, C. Nathan; Fulton, Chad; Masson, Charles; Häggström, Christian; Fitzgerald, Clark; Nicholson, David A.; Hagen, David R.; Pasechnik, Dmitrii V.; Olivetti, Emanuele; Martin, Eric; Wieser, Eric; Silva, Fabrice; Lenders, Felix; Wilhelm, Florian; Young, G.; Price, Gavin A.; Ingold, Gert-Ludwig; Allen, Gregory E.; Lee, Gregory R.; Audren, Hervé; Probst, Irvin; Dietrich, Jörg P.; Silterra, Jacob; Webber, James T; Slavič, Janko; Nothman, Joel; Buchner, Johannes; Kulick, Johannes; Schönberger, Johannes L.; de Miranda Cardoso, José Vinícius; Reimer, Joscha; Harrington, Joseph; Rodríguez, Juan Luis Cano; Nunez-Iglesias, Juan; Kuczynski, Justin; Tritz, Kevin; Thoma, Martin; Newville, Matthew; Kümmerer, Matthias; Bolingbroke, Maximilian; Tartre, Michael; Pak, Mikhail; Smith, Nathaniel J.; Nowaczyk, Nikolai; Shebanov, Nikolay; Pavlyk, Oleksandr; Brodtkorb, Per A.; Lee, Perry; McGibbon, Robert T.; Feldbauer, Roman; Lewis, Sam; Tygier, Sam; Sievert, Scott; Vigna, Sebastiano; Peterson, Stefan; More, Surhud; Pudlik, Tadeusz; Oshima, Takuya; Pingel, Thomas J.; Robitaille, Thomas P.; Spura, Thomas; Jones, Thouis R.; Cera, Tim; Leslie, Tim; Zito, Tiziano; Krauss, Tom; Upadhyay, Utkarsh; Halchenko, Yaroslav O.; Vázquez-Baeza, Yoshiki. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. Nature Methods. https://doi.org/10.1038/s41592-019-0686-2.

27. Mittal, A.; Moorthy, A. K.; Bovik, A. C.. (2012). No-Reference Image Quality Assessment in the Spatial Domain. IEEE Transactions on Image Processing. https://doi.org/10.1109/tip.2012.2214050.

28. Hosu, Vlad, et al. "KonIQ-10k: An ecologically valid database for deep learning of blind image quality assessment" *arXiv preprint arXiv:1910.06180*, 2019. https://arxiv.org/abs/1910.06180.

29. Williams, Samuel; Waterman, Andrew; Patterson, David. (2009). Roofline: an insightful visual performance model for multicore architectures. Communications of the ACM. https://doi.org/10.1145/1498765.1498785.

30. Talebi, Hossein and Peyman Milanfar. "NIMA: Neural Image Assessment" *arXiv preprint arXiv:1709.05424*, 2017. https://arxiv.org/abs/1709.05424.