compimg Documentation

Release 0.2.0

Author

Apr 15, 2019

Contents:

CHAPTER 1

compimg

PyPI PyPI - Python Version PyPI - Wheel

Branches:master: CircleCIdevelop: CircleCI

1.1 Introduction

For full documentation visit documentation site.

Image similarity metrics are often used in image quality assessment for performance evaluation of image restoration and reconstruction algorithms. They require two images:

- test image (image of interest)
- reference image (image we compare against)

Such metrics produce numerical value.

Such methods are are widely called full/reduced-reference methods for assessing image quality.

`compimg` package is all about calculating similarity between images. It provides image similarity metrics (PSNR, SSIM etc.) that are widely used to asses image quality.

```python
import numpy as np
from compimg.similarity import SSIM
some_grayscale_image = np.ones((20,20), dtype=np.uint8)
identical_image = np.ones((20,20), dtype=np.uint8)
result = SSIM().compare(some_grayscale_image, identical_image)
assert result == 1.0
```

1.2 Features

- common metrics for calculating similarity of one image to another

- only `numpy` as a dependency

1.3 Installation

`compimg` is available on PyPI. You can install it using pip:`pip install compimg`

1.4 Note

Keep in mind that metrics are not aware of what kind of image you are passing. If metric relies on intensity values and you have YCbCr image you should pass only the first channel to the computing routine.

1.5 Help

If you have any problems or questions please post an issue.

CHAPTER 2

compimg package

2.1 How to use

Here is the simple example of how one can compare one image to another.

```
>>> import numpy as np
>>> from compimg.similarity import MSE
>>> img = np.ones((20,20), dtype = np.uint8)
>>> reference = np.ones((20,20), dtype = np.uint8)
>>> MSE().compare(img, img)
0.0
```

All metrics implement single interface so it is easy to use multiple of them for example you could run:

```
>>> import numpy as np
>>> from compimg.similarity import MSE, PSNR, SSIM
>>> for metric in [MSE(), PSNR(), SSIM()]:
...     img = np.ones((20,20), dtype = np.uint8)
...     reference = np.zeros((20,20), dtype = np.uint8)
...     value = round(metric.compare(img, reference), 2)
...     print(f"{metric.__class__.__name__} = {value}")
MSE = 1.0
PSNR = 48.13
SSIM = 0.87
```

compimg implicitly converts image to intermediate type (float64) to avoid overflow/underflow when doing calculation. Its advised to leave this type as is, albeit it is possible to change it. For example you could sacrafice precision to improve processing speed by changing it to float32 or even float16.

```
>>> import numpy as np
>>> import compimg
>>> import compimg.similarity
>>> compimg.config.intermediate_type = np.dtype(np.float32)
>>> # code that uses similarity metrics
```

2.2 Submodules

2.3 compimg.exceptions module

compimg exceptions module

exception `compimg.exceptions.`**`DifferentDTypesError`**(*dtype1: numpy.dtype*, *dtype2: numpy.dtype*)

Bases: `Exception`

exception `compimg.exceptions.`**`DifferentShapesError`**(*shape1: Sequence[int], shape2: Sequence[int]*)

Bases: `Exception`

exception `compimg.exceptions.`**`KernelBiggerThanImageError`**(*kernel_shape: Sequence[int], image_shape: Sequence[int]*)

Bases: `Exception`

exception `compimg.exceptions.`**`KernelShapeNotOddError`**(*kernel_shape: Sequence[int]*)

Bases: `Exception`

exception `compimg.exceptions.`**`NegativePadAmountError`**(*amount*)

Bases: `Exception`

2.4 compimg.similarity module

Module with routines for computing similarity between images

class `compimg.similarity.`**`GSSIM`**(*k1: float = 0.01*, *k2: float = 0.03*)

Bases: *`compimg.similarity.SimilarityMetric`*

Gradient-Based Structural similarity index according to the paper "GRADIENT-BASED STRUCTURAL SIMILARITY FOR IMAGE QUALITY ASSESSMENT" by Chen et al.

`compare`(*image: numpy.ndarray*, *reference: numpy.ndarray*) → float

Performs comparison.

Parameters

- **`image`** – Image that is being compared.
- **`reference`** – Image that we compare to.

Returns Numerical result of the comparison.

class `compimg.similarity.`**`MAE`**

Bases: *`compimg.similarity.SimilarityMetric`*

Mean absolute error.

`compare`(*image: numpy.ndarray*, *reference: numpy.ndarray*) → float

Performs comparison.

Parameters

- **`image`** – Image that is being compared.
- **`reference`** – Image that we compare to.

Returns Numerical result of the comparison.

class compimg.similarity.**MSE**

Bases: *compimg.similarity.SimilarityMetric*

Mean squared error.

compare(*image: numpy.ndarray*, *reference: numpy.ndarray*) → float

Performs comparison.

Parameters

- **image** – Image that is being compared.
- **reference** – Image that we compare to.

Returns Numerical result of the comparison.

class compimg.similarity.**PSNR**

Bases: *compimg.similarity.SimilarityMetric*

Peak signal-to-noise ratio according to https://en.wikipedia.org/wiki/Peak_signal-to-noise_ratio.

compare(*image: numpy.ndarray*, *reference: numpy.ndarray*) → float

Performs comparison.

Parameters

- **image** – Image that is being compared.
- **reference** – Image that we compare to.

Returns Numerical result of the comparison.

class compimg.similarity.**RMSE**

Bases: *compimg.similarity.SimilarityMetric*

Root mean squared error.

compare(*image: numpy.ndarray*, *reference: numpy.ndarray*) → float

Performs comparison.

Parameters

- **image** – Image that is being compared.
- **reference** – Image that we compare to.

Returns Numerical result of the comparison.

class compimg.similarity.**SSIM**(*k1: float = 0.01*, *k2: float = 0.03*)

Bases: *compimg.similarity.SimilarityMetric*

Structural similarity index according to the paper from 2004 “Image Quality Assessment: From Error Visibility to Structural Similarity” by Wang et al.

compare(*image: numpy.ndarray*, *reference: numpy.ndarray*) → float

Performs comparison.

Parameters

- **image** – Image that is being compared.
- **reference** – Image that we compare to.

Returns Numerical result of the comparison.

class compimg.similarity.**SimilarityMetric**

Bases: abc.ABC

Abstract class for all similarity metrics.

compare(*image: numpy.ndarray*, *reference: numpy.ndarray*) → float
Performs comparison.

Parameters

- **image** – Image that is being compared.
- **reference** – Image that we compare to.

Returns Numerical result of the comparison.

2.5 compimg.windows module

Module with SlidingWindow interface and its implementations.

class compimg.windows.**IdentitySlidingWindow**(*shape: Tuple[int, int], stride: Tuple[int, int]*)
Bases: *compimg.windows.SlidingWindow*

Slides through the image without making any changes.

slide(*image: numpy.ndarray*) → Generator[numpy.ndarray, None, None]
Using some windows slides over image returning its changed/unchanged fragments.

Parameters **image** – Image to slide over.

Returns Generator that returns views returned by window.

class compimg.windows.**SlidingWindow**
Bases: abc.ABC

slide(*image: numpy.ndarray*) → Generator[numpy.ndarray, None, None]
Using some windows slides over image returning its changed/unchanged fragments.

Parameters **image** – Image to slide over.

Returns Generator that returns views returned by window.

2.6 compimg.pads module

This module defines means to apply padding to images.

class compimg.pads.**ConstantPad**(*value: numbers.Number*, *amount: int*)
Bases: *compimg.pads.Pad*

Adds rows/columns of chosen value at the edges of an image.

apply(*image: numpy.ndarray*) → numpy.ndarray
Pads given image.

Parameters **image** – Image to pad.

Returns Padded image.

class compimg.pads.**EdgePad**(*amount: int*)
Bases: *compimg.pads.Pad*

Replicates neighbouring pixels at edges.

apply(*image: numpy.ndarray*) → numpy.ndarray
Pads given image.

Parameters **`image`** – Image to pad.

Returns Padded image.

class `compimg.pads.`**`FromFunctionPad`**(*function: Callable[[numpy.ndarray], numpy.ndarray]*)

Bases: `compimg.pads.Pad`

`apply`(*image: numpy.ndarray*) → numpy.ndarray

Pads given image.

Parameters **`image`** – Image to pad.

Returns Padded image.

class `compimg.pads.`**`NoPad`**

Bases: `compimg.pads.Pad`

Helper class when one has to pass Pad object but does not want apply any padding.

`apply`(*image: numpy.ndarray*) → numpy.ndarray

Pads given image.

Parameters **`image`** – Image to pad.

Returns Padded image.

class `compimg.pads.`**`Pad`**

Bases: `abc.ABC`

When performing convolution one needs to decide what to do filter is near border(s). Instances implementing this class address that problem.

`apply`(*image: numpy.ndarray*) → numpy.ndarray

Pads given image.

Parameters **`image`** – Image to pad.

Returns Padded image.

2.7 compimg.kernels module

Image processing using kernels.

`compimg.kernels.`**`convolve`**(*image: numpy.ndarray*, *kernel: numpy.ndarray*) → numpy.ndarray

Performs the convolution using provided kernel.

> **Attention:** In case when image has multiple channels and provided kernel has only one, the kernel values get replicated along every channel.

Parameters

- **`image`** – Image on which to perform a convolution.
- **`kernel`** – Kernel to be used.

Returns Convolved image.

Raises

- *`KernelBiggerThanImageError`* – When kernel does not fit into image.
- *`KernelShapeNotOddError`* – When kernel does not is of even shape.

`compimg.kernels.`**`make_guassian_kernel`**(*shape: Tuple[int, int], sigma: float*)

Produces two-dimensional Gaussian kernel according to https://en.wikipedia.org/wiki/Gaussian_function.

Parameters

- **`shape`** – Shape of the kernel.
- **`sigma`** – Sigma to use in the formula.

Returns Gaussian kernel.

CHAPTER 3

License

Apache License

Version 2.0, January 2004

http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

   “License” shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

   “Licensor” shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

   “Legal Entity” shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, “control” means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

   “You” (or “Your”) shall mean an individual or Legal Entity exercising permissions granted by this License.

   “Source” form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

   “Object” form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

   “Work” shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

   “Derivative Works” shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

   (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

   (b) You must cause any modified files to carry prominent notices stating that You changed the files; and

   (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

   (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

> To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

> http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either

CHAPTER 4

CHANGELOG

4.1 compimg 0.2.0

- Added `GSSIM` metric
- Added `RMSE` metric
- Added 'MAE' metric
- Added `compimg.pads` module which provides easy way to apply padding to an image (used in *SSIM implementations)
- Added `compimg.kernels` module which makes possible applying kernel to an image (used within *SSIM implementations)
- More and better exceptions
- Moved `compimg.similarity.intermediate_type` to `compimg.config.intermediate_dtype`
- Fixed `SSIM` metric (now implementation follows steps from the one provided by authors)

4.2 compimg 0.1.1

This release fixes some small documentation errors, readme typos and adds some badges to the README file. There are no actual code changes.

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index