How can I measure the similarity between two images? [closed]

Asked 16 years, 4 months ago Modified 5 years, 10 months ago

Viewed 119k times

104

Closed. This question is seeking recommendations for software libraries, tutorials, tools, books, or other off-site resources. It does not meet Stack Overflow guidelines. It is not currently accepting answers.

We don't allow questions seeking recommendations for software libraries, tutorials, tools, books, or other off-site resources. You can edit the question so it can be answered with facts and citations.

Closed 6 years ago.

Improve this question

I would like to compare a screenshot of one application (could be a Web page) with a previously taken screenshot to determine whether the application is displaying itself correctly. I don't want an exact match comparison, because the aspect could be slightly different (in the case of a Web app, depending on the browser, some element could be at a slightly different location). It should give a measure of how similar are the screenshots.

Is there a library / tool that already does that? How would you implement it?

algorithm language-agnostic image image-processing

Share

Improve this question

Follow

edited Aug 26, 2009 at 0:49

Luke Quinane

16.6k ● 13 ● 72 ● 89

asked Aug 25, 2008 at 12:51

Antoine Aubry

12.4k ● 10 ● 46 ● 76

1 There are some good answers in this other similar question: stackoverflow.com/questions/75891/… – blak Jul 2, 2012 at 20:16

2 And more here: stackoverflow.com/questions/189943/… – Anoyz Jun 9, 2015 at 11:35

1 Time to update answers in light of recent advances in Machine Learning and more specifically "Deep Learning". – jldupont Jul 21, 2016 at 14:19

My lab needed to solve this problem too, and used the workflow outlined here: douglasduhaime.com/posts/… – duhaime Mar 28, 2018 at 12:18

1 Check out project sewar : pypi.org/project/sewar – mustafa can nacak Feb 10, 2023 at 10:32

17 Answers

Sorted by: Highest score (default)

77

This depends entirely on how smart you want the algorithm to be.

For instance, here are some issues:

- cropped images vs. an uncropped image
- images with a text added vs. another without
- mirrored images

The easiest and simplest *algorithm* I've seen for this is just to do the following steps to each image:

1. scale to something small, like 64x64 or 32x32, disregard aspect ratio, use a combining scaling algorithm instead of nearest pixel
2. scale the color ranges so that the darkest is black and lightest is white
3. rotate and flip the image so that the lighest color is top left, and then top-right is next darker, bottom-left is next darker (as far as possible of course)

Edit A *combining scaling algorithm* is one that when scaling 10 pixels down to one will do it using a function that takes the color of all those 10 pixels and combines them into one. Can be done with algorithms like averaging, mean-value, or more complex ones like bicubic splines.

Then calculate the mean distance pixel-by-pixel between the two images.

To look up a possible match in a database, store the pixel colors as individual columns in the database, index a bunch of them (but not all, unless you use a very small image), and do a query that uses a range for each pixel value, ie. every image where the pixel in the small image is between -5 and +5 of the image you want to look up.

This is easy to implement, and fairly fast to run, but of course won't handle most advanced differences. For that you need much more advanced algorithms.

Share Improve this answer Follow

edited Oct 15, 2008 at 14:58

answered Aug 25, 2008 at 13:53

Lasse V. Karlsen

391k ● 106 ● 646 ● 844

14 What is a "combining scaling algorithm"? – Gregg Lind Oct 8, 2008 at 13:54

Is the "combining scaling algorithm" similar to TensorFlow's Pooling? – The Singularity Mar 13, 2021 at 7:11

34

The 'classic' way of measuring this is to break the image up into some canonical number of sections (say a 10x10 grid) and then computing a histogram of RGB values inside of each cell and compare corresponding histograms. This type of algorithm is preferred because of

both its simplicity and it's invariance to scaling and (small!) translation.

Share Improve this answer

Follow

answered Aug 25, 2008 at 19:18

Louis Brandy

19.8k ● 3 ● 39 ● 29

7 Isn't this similar to doing a single histogram for the whole image, but with the added drawbacks of not being resiliant to mirror and rotate? – dodgy_coder May 17, 2012 at 4:46

1 2 histograms from 2 halves of image will have better matching precision than 1 histogram of a whole. Though it has drawbacks you mentioned, it depends on what problem you are solving. – psycho brm Oct 22, 2012 at 11:00

27

Use a normalised colour histogram. (Read the section on applications here), they are commonly used in image retrieval/matching systems and are a standard way of matching images that is very reliable, relatively fast and very easy to implement.

Essentially a colour histogram will capture the colour distribution of the image. This can then be compared with another image to see if the colour distributions match.

This type of matching is pretty resiliant to scaling (once the histogram is normalised), and rotation/shifting/movement etc.

Avoid pixel-by-pixel comparisons as if the image is rotated/shifted slightly it may lead to a large difference

being reported.

Histograms would be straightforward to generate yourself (assuming you can get access to pixel values), but if you don't feel like it, the OpenCV library is a great resource for doing this kind of stuff. Here is a powerpoint presentation that shows you how to create a histogram using OpenCV.

Share Improve this answer Follow

edited Aug 27, 2008 at 16:53

answered Aug 27, 2008 at 16:41

Lehane

48.6k ● 14 ● 55 ● 54

14

Don't video encoding algorithms like MPEG compute the difference between each frame of a video so they can just encode the delta? You might look into how video encoding algorithms compute those frame differences.

Look at this open source image search application http://www.semanticmetadata.net/lire/. It describes several image similarity algorighms, three of which are from the MPEG-7 standard: ScalableColor, ColorLayout, EdgeHistogram and Auto Color Correlogram.

Share Improve this answer Follow

answered Sep 16, 2008 at 20:29

Mark B

200k ● 27 ● 327 ● 321

1 This would not answer the question here. The question is not about pixel per pixel comparison. – user172163 Dec 16, 2015 at 19:13

2 @Kousha True, but still an interesting direction for thinking. – meaning-matters May 17, 2018 at 17:10

14

You could use a pure mathematical approach of `O(n^2)`, but it will be useful only if you are certain that there's no offset or something like that. (Although that if you have a few objects with homogeneous coloring it will still work pretty well.)

Anyway, the idea is the compute the normalized dot-product of the two matrices. `C = sum(Pij*Qij)^2/(sum(Pij^2)*sum(Qij^2))`.

This formula is actually the "cosine" of the angle between the matrices (wierd). The bigger the similarity (lets say `Pij=Qij`), C will be 1, and if they're completely different, lets say for every `i,j Qij = 1` (avoiding zero-division), `Pij = 255`, then for size `nxn`, the bigger `n` will be, the closer to zero we'll get. (By rough calculation: `C=1/n^2`).

Share Improve this answer

Follow

edited Nov 16, 2012 at 11:02

SchmitzIT

9,525 ● 10 ● 70 ● 95

answered Jul 23, 2011 at 14:55

Shachar

141 ● 1 ● 2

8

You'll need pattern recognition for that. To determine small differences between two images, Hopfield nets work fairly well and are quite easy to implement. I don't know any available implementations, though.

Share Improve this answer

Follow

answered Aug 25, 2008 at 13:00

Konrad Rudolph

545k ● 139 ● 956 ● 1.2k

7

A ruby solution can be found here

From the readme:

Phashion is a Ruby wrapper around the pHash library, "perceptual hash", which detects duplicate and near duplicate multimedia files

Share Improve this answer

Follow

edited Oct 10, 2012 at 13:06

Janak Nirmal

22.7k ● 18 ● 65 ● 101

answered Sep 22, 2011 at 20:51

edk750

1,110 ● 10 ● 11

6

How to measure similarity between two images entirely depends on what you would like to measure, for example: contrast, brightness, modality, noise... and then choose the best suitable similarity measure there is for you. You can choose from **MAD** (mean absolute difference), **MSD** (mean squared difference) which are good for measuring brightness...there is also available **CR** (correlation coefficient) which is good in representing correlation between two images. You could also choose from histogram based similarity measures like **SDH** (standard deviation of difference image histogram) or multimodality

similarity measures like **MI** (mutual information) or **NMI** (normalized mutual information).

Because this similarity measures cost much in time, it is advised to scale images down before applying these measures on them.

Share Improve this answer

Follow

edited Feb 24, 2019 at 21:55

Shmil The Cat

4,670 ● 2 ● 30 ● 38

answered Jan 31, 2016 at 10:41

Gregor Simončič

73 ● 1 ● 7

4

I wonder (and I'm really just throwing the idea out there to be shot down) if something could be derived by subtracting one image from the other, and then compressing the resulting image as a jpeg of gif, and taking the file size as a measure of similarity.

If you had two identical images, you'd get a white box, which would compress really well. The more the images differed, the more complex it would be to represent, and hence the less compressible.

Probably not an ideal test, and probably much slower than necessary, but it might work as a quick and dirty implementation.

Share Improve this answer

edited Aug 25, 2008 at 13:10

Follow

answered Aug 25, 2008 at 13:04

Matt Sheppard

118k ● 46 ● 113 ● 134

Think about rotating 90 degrees; images are still similar.
– meaning-matters May 17, 2018 at 17:13

3

You might look at the code for the open source tool findimagedupes, though it appears to have been written in perl, so I can't say how easy it will be to parse...

Reading the findimagedupes page that I liked, I see that there is a C++ implementation of the same algorithm. Presumably this will be easier to understand.

And it appears you can also use gqview.

Share Improve this answer

Follow

edited May 6, 2017 at 0:52

answered Aug 25, 2008 at 13:09

dmckee --- ex-moderator kitten

101k ● 25 ● 146 ● 235

2

Well, not to answer your question directly, but I have seen this happen. Microsoft recently launched a tool called PhotoSynth which does something very similar to

determine overlapping areas in a large number of pictures (which could be of different aspect ratios).

I wonder if they have any available libraries or code snippets on their blog.

Share Improve this answer

Follow

answered Aug 25, 2008 at 12:57

Vaibhav

11.4k ● 11 ● 53 ● 71

1 This tech. has been discontinued. – Joseph Rosson Apr 11, 2017 at 20:17

2

to expand on Vaibhav's note, hugin is an open-source 'autostitcher' which should have some insight on the problem.

Share Improve this answer

Follow

answered Aug 25, 2008 at 13:16

hometoast

11.8k ● 5 ● 42 ● 58

2

There's software for content-based image retrieval, which does (partially) what you need. All references and explanations are linked from the project site and there's also a short text book (Kindle): LIRE

Share Improve this answer

Follow

answered Apr 19, 2013 at 14:51

Mathias

324 ● 1 ● 5

1

You can use Siamese Network to see if the two images are similar or dissimilar following this tutorial. This tutorial cluster the similar images whereas you can use `L2` distance to measure the similarity of two images.

Share Improve this answer

Follow

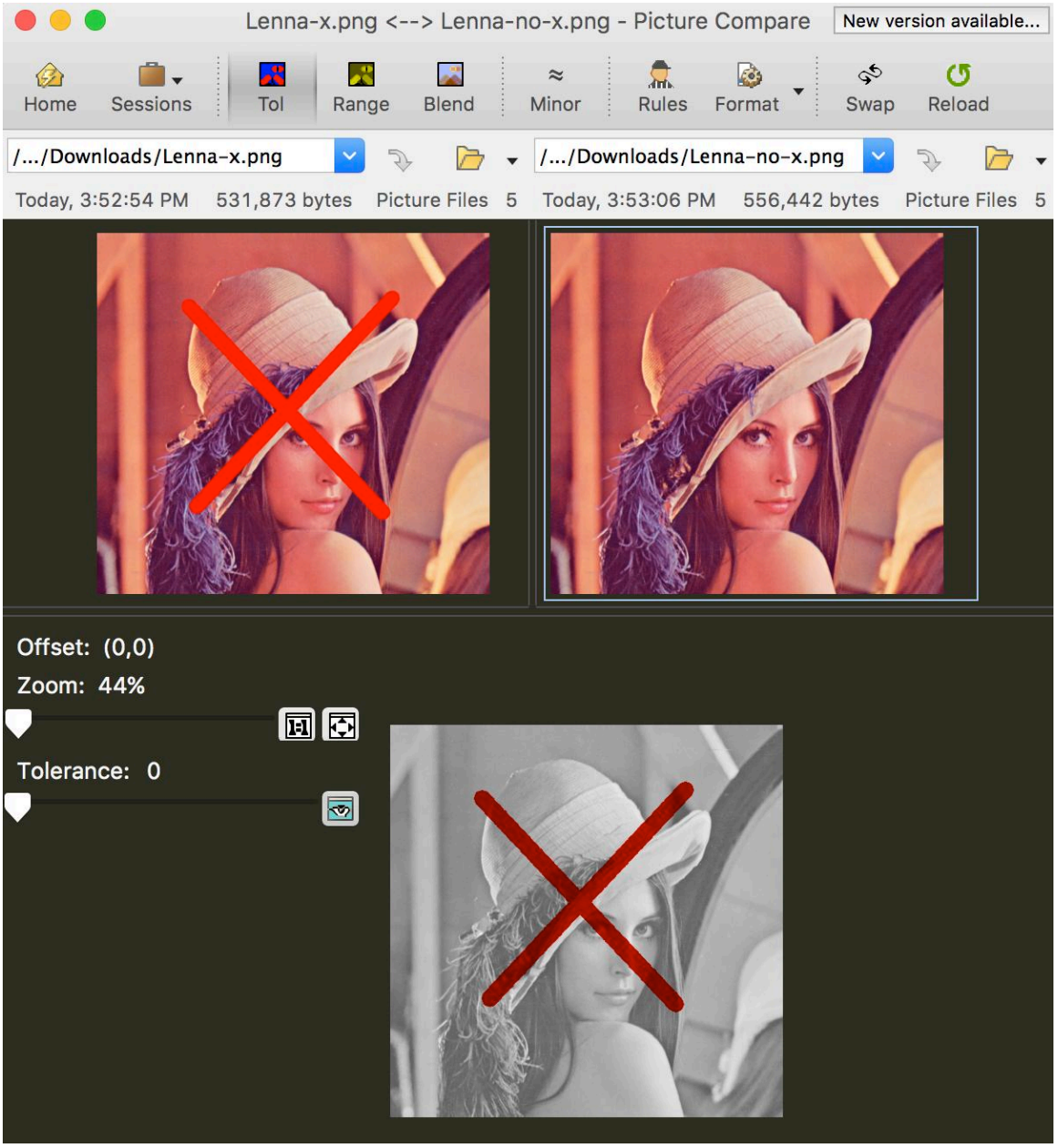
answered Feb 27, 2017 at 8:53

cpwah

141 ● 1 ● 3 ● 13

1

Beyond Compare has pixel-by-pixel comparison for images, e.g.,



Share Improve this answer

Follow

answered Apr 26, 2017 at 19:57

emallove

1,517 ● 1 ● 17 ● 23

@xilpex, the OP asks: *Is there a library / tool that already does that*? My answer includes a link to such a library / tool. – emallove Jul 27, 2020 at 23:34

If this is something you will be doing on an occasional basis and doesn't need automating, you can do it in an

0

image editor that supports layers, such as Photoshop or Paint Shop Pro (probably GIMP or Paint.Net too, but I'm not sure about those). Open both screen shots, and put one as a layer on top of the other. Change the layer blending mode to Difference, and everything that's the same between the two will become black. You can move the top layer around to minimize any alignment differences.

Share Improve this answer

Follow

answered Oct 15, 2008 at 15:15

Mark Ransom

308k ● 44 ● 416 ● 647

Another tool that makes this type of diffing very simple is kaleidoscopeapp.com – Michael Osofsky Dec 19, 2018 at 21:51

-1

Well a really base-level method to use could go through every pixel colour and compare it with the corresponding pixel colour on the second image - but that's a probably a very **very** slow solution.

Share Improve this answer

Follow

answered Aug 25, 2008 at 13:19

Ross

47k ● 39 ● 123 ● 173