**Face Recognition Accuracy Problems**

Adam Geitgey edited this page on Jun 12, 2018 · [4 revisions](https://github.com/ageitgey/face_recognition/wiki/Face-Recognition-Accuracy-Problems/_history)

**Pages 9**



* [Home](https://github.com/ageitgey/face_recognition/wiki)
* [Calculating Accuracy as a Percentage](https://github.com/ageitgey/face_recognition/wiki/Calculating-Accuracy-as-a-Percentage)
* [Common Errors](https://github.com/ageitgey/face_recognition/wiki/Common-Errors)
* [Face Detection Problems](https://github.com/ageitgey/face_recognition/wiki/Face-Detection-Problems)
* [Face Recognition Accuracy Problems](https://github.com/ageitgey/face_recognition/wiki/Face-Recognition-Accuracy-Problems)
  + [Question: I'm getting false positives. How can I make face comparisons more strict?](https://github.com/ageitgey/face_recognition/wiki/Face-Recognition-Accuracy-Problems#question-im-getting-false-positives-how-can-i-make-face-comparisons-more-strict)
  + [Question: What are some tricks I can use to improve face recognition accuracy?](https://github.com/ageitgey/face_recognition/wiki/Face-Recognition-Accuracy-Problems#question-what-are-some-tricks-i-can-use-to-improve-face-recognition-accuracy)
  + [Question: Face recognition works well with European individuals, but overall accuracy is lower with Asian individuals.](https://github.com/ageitgey/face_recognition/wiki/Face-Recognition-Accuracy-Problems#question-face-recognition-works-well-with-european-individuals-but-overall-accuracy-is-lower-with-asian-individuals)
  + [Question: Your application only uses one picture of each person to identify them. Can I use more than one picture of each person to make identification more accurate?](https://github.com/ageitgey/face_recognition/wiki/Face-Recognition-Accuracy-Problems#question-your-application-only-uses-one-picture-of-each-person-to-identify-them-can-i-use-more-than-one-picture-of-each-person-to-make-identification-more-accurate)
  + [Question: Can I re-train the face encoding model to make it more accurate for my images?](https://github.com/ageitgey/face_recognition/wiki/Face-Recognition-Accuracy-Problems#question-can-i-re-train-the-face-encoding-model-to-make-it-more-accurate-for-my-images)
* [How do I save face encodings to a file?](https://github.com/ageitgey/face_recognition/wiki/How-do-I-save-face-encodings-to-a-file%3F)
* [Known Face Image Datasets](https://github.com/ageitgey/face_recognition/wiki/Known-Face-Image-Datasets)
* [Loading corrupted or partially downloaded image files](https://github.com/ageitgey/face_recognition/wiki/Loading-corrupted-or-partially-downloaded-image-files)
* [Using Docker](https://github.com/ageitgey/face_recognition/wiki/Using-Docker)

**Clone this wiki locally**



[**Question: I'm getting false positives. How can I make face comparisons more strict?**](https://github.com/ageitgey/face_recognition/wiki/Face-Recognition-Accuracy-Problems#question-im-getting-false-positives-how-can-i-make-face-comparisons-more-strict)

Faces in this system are encoded and represented as 128-dimension points in space (called face vectors). By default, faces are considered a match if the euclidean distance between the face vectors is 0.6 or less.

You can control how strict the comparison is by passing a tolerance parameter to the compare\_faces() function:

results = face\_recognition.compare\_faces(known\_face\_encodings, face\_encoding\_to\_check, tolerance=0.5)

Using tolerance values lower than 0.6 will make the comparison more strict.

[**Question: What are some tricks I can use to improve face recognition accuracy?**](https://github.com/ageitgey/face_recognition/wiki/Face-Recognition-Accuracy-Problems#question-what-are-some-tricks-i-can-use-to-improve-face-recognition-accuracy)

First, you can use jittering when creating face encodings:

face\_encodings = face\_recognition.face\_encodings(face\_image, num\_jitters=100)

That will tell dlib to randomly distort your image 100 times (randomly zoomed, rotated, translated, flipped), take the encoding of each version of the image and return the average. This can give you more generic face encodings that work better in some cases with the tradeoff that the algorithm will run more slowly. Whether or not the increase in accuracy is worth the slower runtime depends on your use case.

Second, if you notice that same faces are not detected at all in your original images, you might try using upsampling when looking for faces in the images:

face\_locations = face\_recognition.face\_locations(face\_image, number\_of\_times\_to\_upsample=2)

face\_encodings = face\_recognition.face\_encodings(face\_image, known\_face\_locations=face\_locations, num\_jitters=100)

Passing number\_of\_times\_to\_upsample=2 means that the original image will be scaled up twice when looking for faces. This can help find smaller faces in the image that might otherwise be missed but it will cause everything to run much more slowly since the input image will now be much larger.

[**Question: Face recognition works well with European individuals, but overall accuracy is lower with Asian individuals.**](https://github.com/ageitgey/face_recognition/wiki/Face-Recognition-Accuracy-Problems#question-face-recognition-works-well-with-european-individuals-but-overall-accuracy-is-lower-with-asian-individuals)

This is a real problem. The face recognition model is only as good as the training data. Since the face recognition model was trained using public datasets built pictures scraped from websites (celebrities, etc), it only works as well as the source data. Those public datasets are not evenly distributed amongst all individuals from all countries.

I hope to improve this in the future, but it requires building a dataset of millions of pictures of millions of people from lots of different places. If you have access to this kind of data and are willing to share it for model training, please [email me](mailto:ageitgey@gmail.com).

[**Question: Your application only uses one picture of each person to identify them. Can I use more than one picture of each person to make identification more accurate?**](https://github.com/ageitgey/face_recognition/wiki/Face-Recognition-Accuracy-Problems#question-your-application-only-uses-one-picture-of-each-person-to-identify-them-can-i-use-more-than-one-picture-of-each-person-to-make-identification-more-accurate)

Sure! It just requires more work and the best way to do it depends on the kind of application you are building.

You can use the [face\_encodings()](http://face-recognition.readthedocs.io/en/latest/face_recognition.html" \l "face_recognition.api.face_encodings) function to get a representation of a single face image. The representation is an array with 128 floating point elements (i.e. a face vector). You can use these face vectors to build any kind of machine learning classification model you want.

Here's a working example using a KNN classifier to classify a new image based on multiple pictures of each known person: <https://github.com/ageitgey/face_recognition/blob/master/examples/face_recognition_knn.py>

[**Question: Can I re-train the face encoding model to make it more accurate for my images?**](https://github.com/ageitgey/face_recognition/wiki/Face-Recognition-Accuracy-Problems#question-can-i-re-train-the-face-encoding-model-to-make-it-more-accurate-for-my-images)

The default face encoding model was trained by @davisking on millions of images of faces grouped by individual. Re-training the model is not possible unless you have that volume of data. Adding a few thousand of your own images won't really help. Instead, try building a classifier on top of the current face encodings model [like this](https://github.com/ageitgey/face_recognition/blob/master/examples/face_recognition_knn.py).

So if you don't have tens of millions of images grouped by individual, you can't really retrain the model yourself. And if you do, [let me know](mailto:ageitgey@gmail.com) so we can combine training data! :)

If you do want to try to retrain the model yourself, you'll need to be familiar with C++ and compiling dlib.

The face encoding model is a ResNet. You can see the dlib C++ DL API definition of it here: <https://github.com/davisking/dlib/blob/master/tools/python/src/face_recognition.cpp#L98-L125>

You can see an example of how to train the network with dlib here: <http://dlib.net/dnn_metric_learning_on_images_ex.cpp.html>

But please note that you need to make sure the network definition that you train matches the one defined in face\_recognition.cpp above so that you'll be able to load your trained weights with the existing face recognition code.

If you do train your own model, then you might be better of just using the dlib python interface directly (or even the C++ api). The main utility of this library is to provide a simple solution that works "out of the box". If you are familiar enough with dlib to train your own model, then you probably don't need this.