

[< Back to Alex Krizhevsky's home page](#)

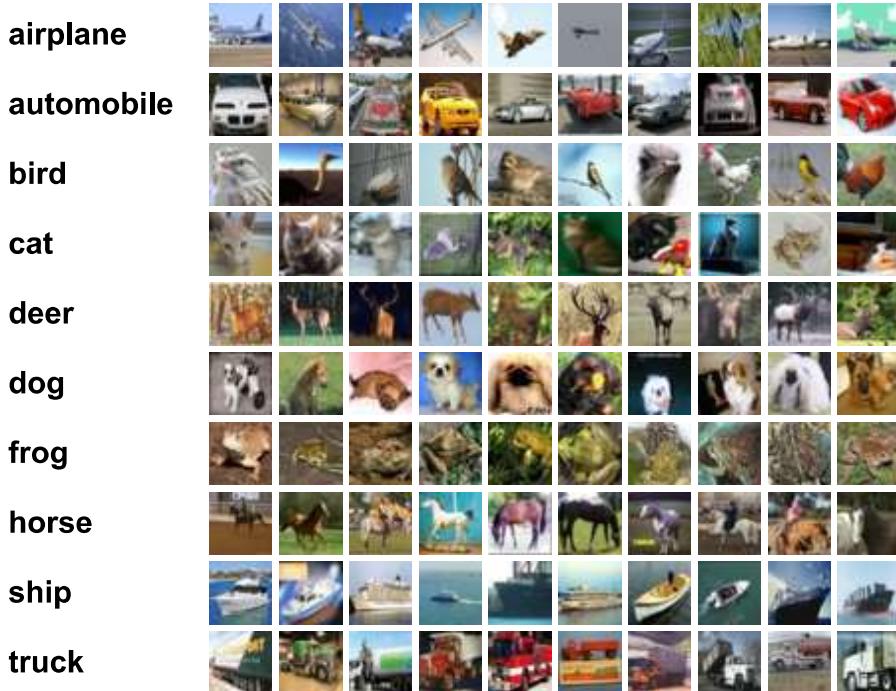
The CIFAR-10 and CIFAR-100 datasets are labeled subsets of the [80 million tiny images](#) dataset. CIFAR-10 and CIFAR-100 were created by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton.

## The CIFAR-10 dataset

The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images.

The dataset is divided into five training batches and one test batch, each with 10000 images. The test batch contains exactly 1000 randomly-selected images from each class. The training batches contain the remaining images in random order, but some training batches may contain more images from one class than another. Between them, the training batches contain exactly 5000 images from each class.

Here are the classes in the dataset, as well as 10 random images from each:



The classes are completely mutually exclusive. There is no overlap between automobiles and trucks. "Automobile" includes sedans, SUVs, things of that sort. "Truck" includes only big trucks. Neither includes pickup trucks.

## Download

If you're going to use this dataset, please cite the tech report at the bottom of this page.

### Version

	<b>Size</b>	<b>md5sum</b>
<a href="#">CIFAR-10 python version</a>	163 MB	c58f30108f718f92721af3b95e74349a
<a href="#">CIFAR-10 Matlab version</a>	175 MB	70270af85842c9e89bb428ec9976c926
<a href="#">CIFAR-10 binary version (suitable for C programs)</a>	162 MB	c32a1d4ab5d03f1284b67883e8d87530

## Baseline results

You can find some baseline replicable results on this dataset [on the project page for cuda-convnet](#). These results were obtained with a convolutional neural network. Briefly, they are 18% test error without data

augmentation and 11% with. Additionally, [Jasper Snoek](#) has a [new paper](#) in which he used Bayesian hyperparameter optimization to find nice settings of the weight decay and other hyperparameters, which allowed him to obtain a test error rate of 15% (without data augmentation) using the architecture of the net that got 18%.

## Other results

[Rodrigo Benenson](#) has collected results on CIFAR-10/100 and other datasets on his website; [click here](#) to view.

## Dataset layout

### Python / Matlab versions

I will describe the layout of the Python version of the dataset. The layout of the Matlab version is identical.

The archive contains the files `data_batch_1`, `data_batch_2`, ..., `data_batch_5`, as well as `test_batch`. Each of these files is a Python "pickled" object produced with `cPickle`. Here is a python2 routine which will open such a file and return a dictionary:

```
def unpickle(file):
    import cPickle
    with open(file, 'rb') as fo:
        dict = cPickle.load(fo)
    return dict
```

And a python3 version:

```
def unpickle(file):
    import pickle
    with open(file, 'rb') as fo:
        dict = pickle.load(fo, encoding='bytes')
    return dict
```

Loaded in this way, each of the batch files contains a dictionary with the following elements:

- **data** -- a 10000x3072 [numpy](#) array of `uint8s`. Each row of the array stores a 32x32 colour image. The first 1024 entries contain the red channel values, the next 1024 the green, and the final 1024 the blue. The image is stored in row-major order, so that the first 32 entries of the array are the red channel values of the first row of the image.
- **labels** -- a list of 10000 numbers in the range 0-9. The number at index *i* indicates the label of the *i*th image in the array **data**.

The dataset contains another file, called `batches.meta`. It too contains a Python dictionary object. It has the following entries:

- **label\_names** -- a 10-element list which gives meaningful names to the numeric labels in the **labels** array described above. For example, `label_names[0] == "airplane", label_names[1] == "automobile", etc.`

## Binary version

The binary version contains the files `data_batch_1.bin`, `data_batch_2.bin`, ..., `data_batch_5.bin`, as well as `test_batch.bin`. Each of these files is formatted as follows:

```
<1 x label><3072 x pixel>
...
```

<1 x label><3072 x pixel>

In other words, the first byte is the label of the first image, which is a number in the range 0-9. The next 3072 bytes are the values of the pixels of the image. The first 1024 bytes are the red channel values, the next 1024 the green, and the final 1024 the blue. The values are stored in row-major order, so the first 32 bytes are the red channel values of the first row of the image.

Each file contains 10000 such 3073-byte "rows" of images, although there is **nothing delimiting the rows**. Therefore each file should be exactly 30730000 bytes long.

There is another file, called `batches.meta.txt`. This is an ASCII file that maps numeric labels in the range 0-9 to meaningful class names. It is merely a list of the 10 class names, one per row. The class name on row  $i$  corresponds to numeric label  $i$ .

## The CIFAR-100 dataset

This dataset is just like the CIFAR-10, except it has 100 classes containing 600 images each. There are 500 training images and 100 testing images per class. The 100 classes in the CIFAR-100 are grouped into 20 superclasses. Each image comes with a "fine" label (the class to which it belongs) and a "coarse" label (the superclass to which it belongs).

Here is the list of classes in the CIFAR-100:

### Superclass

aquatic mammals  
fish  
flowers  
food containers  
fruit and vegetables  
household electrical devices  
household furniture  
insects  
large carnivores  
large man-made outdoor things  
large natural outdoor scenes  
large omnivores and herbivores  
medium-sized mammals  
non-insect invertebrates  
people  
reptiles  
small mammals  
trees  
vehicles 1  
vehicles 2

### Classes

beaver, dolphin, otter, seal, whale  
aquarium fish, flatfish, ray, shark, trout  
orchids, poppies, roses, sunflowers, tulips  
bottles, bowls, cans, cups, plates  
apples, mushrooms, oranges, pears, sweet peppers  
clock, computer keyboard, lamp, telephone, television  
bed, chair, couch, table, wardrobe  
bee, beetle, butterfly, caterpillar, cockroach  
bear, leopard, lion, tiger, wolf  
bridge, castle, house, road, skyscraper  
cloud, forest, mountain, plain, sea  
camel, cattle, chimpanzee, elephant, kangaroo  
fox, porcupine, possum, raccoon, skunk  
crab, lobster, snail, spider, worm  
baby, boy, girl, man, woman  
crocodile, dinosaur, lizard, snake, turtle  
hamster, mouse, rabbit, shrew, squirrel  
maple, oak, palm, pine, willow  
bicycle, bus, motorcycle, pickup truck, train  
lawn-mower, rocket, streetcar, tank, tractor

Yes, I know mushrooms aren't really fruit or vegetables, and bears aren't really carnivores.

## Download

### Version

[CIFAR-100 python version](#)

### Size

161 MB

### md5sum

eb9058c3a382ffc7106e4002c42a8d85

<a href="#">CIFAR-100 Matlab version</a>	175 MB	6a4bfa1dcd5c9453dda6bb54194911f4
<a href="#">CIFAR-100 binary version (suitable for C programs)</a>	161 MB	03b5dce01913d631647c71ecec9e9cb8

## Dataset layout

### Python / Matlab versions

The python and Matlab versions are identical in layout to the CIFAR-10, so I won't waste space describing them here.

### Binary version

The binary version of the CIFAR-100 is just like the binary version of the CIFAR-10, except that each image has two label bytes (coarse and fine) and 3072 pixel bytes, so the binary files look like this:

```
<1 x coarse label><1 x fine label><3072 x pixel>
...
<1 x coarse label><1 x fine label><3072 x pixel>
```

## Indices into the original 80 million tiny images dataset

Sivan Sabato was kind enough to provide [this file](#), which maps CIFAR-100 images to images in the 80 million tiny images dataset. Sivan Writes:

The file has 60000 rows, each row contains a single index into the tiny db, where the first image in the tiny db is indexed "1". "0" stands for an image that is not from the tiny db. The first 50000 lines correspond to the training set, and the last 10000 lines correspond to the test set.

## Reference

This tech report (Chapter 3) describes the dataset and the methodology followed when collecting it in much greater detail. Please cite it if you intend to use this dataset.

- [Learning Multiple Layers of Features from Tiny Images](#), Alex Krizhevsky, 2009.