CNN on CIFAR dataset

CIFAR-10 by DenseNet Implementation

Cifar-10 is a popular dataset available at https://www.cs.toronto.edu/~kriz/cifar.html We plan to solve this problem by the use of Densenet Architecture. An awesome way of solving this problem with help of Resnet is available at keras website https://keras.io/examples/cifar10_resnet/¶

But why not Transfer Learning?

As The weights trained on ResNet or DenseNet are for ImageNet which compromises of Images of Dimension 224x224 and the image dimensions in CIFAR-10 are of 32x32 that means we cannot upsample that much anyhow. So instead we will use the same architecture of DenseNet explained in https://arxiv.org/pdf/1608.06993.pdf and will try to get as much as Accuracy possible on the dataset. PS. We are using Google Colab for the training purpose.

In [1]:

```
from keras.preprocessing import image
from keras.utils import to_categorical
from keras.models import Sequential, Model
from keras.layers import Conv2D
from keras.layers import MaxPooling2D, Dropout, BatchNormalization, Activation, Concatenate
from keras.layers import Dense, Flatten, GlobalAveragePooling2D, Input, AveragePooling2D
from keras.preprocessing.image import ImageDataGenerator

from keras.optimizers import SGD, Adam
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from keras.datasets import cifar10
from sklearn.metrics import confusion_matrix

import matplotlib.pyplot as plt
import seaborn as sns
import cv2
import os
import tensorflow as tf
from keras import backend as k
```

```
Using TensorFlow backend.
```

The default version of TensorFlow in Colab will soon switch to TensorFlow 2.x.
We recommend you upgrade now or ensure your notebook will continue to use TensorFlow 1.x via the `%tensorflow_version 1.x` magic: more info.

In [0]:

```
# Allocate the memory as needed instead of preloading
config = tf.ConfigProto()
config.gpu_options.allow_growth = True
```

In [0]:

```
train_datagen = ImageDataGenerator(
        rotation_range=90,
        width_shift_range=0.2,
        height_shift_range=0.2,
        shear_range=0.2,
        zoom_range=0.2,
        horizontal_flip=True,
        fill_mode='nearest')
```

In [0]:

```
# Loading the CIFAR data from the keras dataset
num_classes = 10
(x_train, y_train), (x_test, y_test) = cifar10.load_data()
```

In [0]:

```
img_height, img_width, channel = x_train.shape[1],x_train.shape[2],x_train.shape[3]
```

In [0]:

```
# convert to one hot encoing
y_train = to_categorical(y_train, num_classes)
y_test = to_categorical(y_test, num_classes)
```

In [10]:

```
x_train.shape
```

Out[10]:

```
(50000, 32, 32, 3)
```

In [11]:

```
x_test.shape
```

Out[11]:

```
(10000, 32, 32, 3)
```

In [0]:

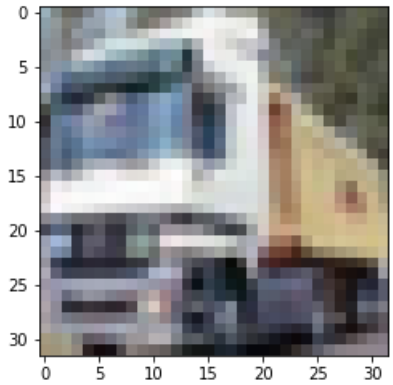
```
x_train = x_train / 255
x_test = x_test / 255
```

In [13]:

```
from matplotlib import pyplot as plt
plt.imshow(x_train[1])
```

Out[13]:

```
<matplotlib.image.AxesImage at 0x7f5e5e1497f0>
```



In [0]:

```
# Dense Block
compression = 0.5
def denseblock(input, num_filter = 12, dropout_rate = 0.2):
    global compression
    temp = input
    for _ in range(l):
        BatchNorm = BatchNormalization()(temp)
```

```
        relu = Activation('relu')(BatchNorm)
        Conv2D_3_3 = Conv2D(int(num_filter*compression), (3,3), use_bias=False ,padding='same')(rel
u)
        if dropout_rate>0:
            Conv2D_3_3 = Dropout(dropout_rate)(Conv2D_3_3)
        concat = Concatenate(axis=-1)([temp,Conv2D_3_3])

        temp = concat

    return temp
```

In [0]:

```
## transition Block
compression = 0.5
def transition(input, num_filter = 12, dropout_rate = 0.2):
    global compression
    BatchNorm = BatchNormalization()(input)
    relu = Activation('relu')(BatchNorm)
    Conv2D_BottleNeck = Conv2D(int(num_filter*compression), (1,1), use_bias=False ,padding='same')(
relu)
    if dropout_rate>0:
         Conv2D_BottleNeck = Dropout(dropout_rate)(Conv2D_BottleNeck)
    avg = AveragePooling2D(pool_size=(2,2))(Conv2D_BottleNeck)
    return avg
```

In [0]:

```
from keras import layers

#output layer
compression = 0.5
def output_layer(input):
    global compression
    BatchNorm = BatchNormalization()(input)
    relu = Activation('relu')(BatchNorm)
    cv = Conv2D(10, (1,1), use_bias=False ,padding='same')(relu)
    avg = AveragePooling2D(pool_size=(2,2))(cv)
    pooling = GlobalAveragePooling2D()(avg)
    output = Activation('softmax')(pooling)

    return output
```

In [0]:

```
# Hyperparameters
batch_size = 128
num_classes = 10
epochs = 35
l = 6
num_filter = 64
compression = 0.5
dropout_rate = 0
```

In [23]:

```
input = layers.Input(shape=(img_height, img_width, channel,))
First_Conv2D = layers.Conv2D(num_filter, (3,3), use_bias=False ,padding='same')(input)

First_Block = denseblock(First_Conv2D, num_filter, dropout_rate)
First_Transition = transition(First_Block, num_filter, dropout_rate)

Second_Block = denseblock(First_Transition, num_filter, dropout_rate)
Second_Transition = transition(Second_Block, num_filter, dropout_rate)

Third_Block = denseblock(Second_Transition, num_filter, dropout_rate)
Third_Transition = transition(Third_Block, num_filter, dropout_rate)

Last_Block = denseblock(Third_Transition,  num_filter, dropout_rate)
output = output_layer(Last_Block)
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
```

```
packages/keras/backend/tensorflow_backend.py:190: The name tf.get_default_session is deprecated. P
lease use tf.compat.v1.get_default_session instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:203: The name tf.Session is deprecated. Please use tf
.compat.v1.Session instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:207: The name tf.global_variables is deprecated. Plea
se use tf.compat.v1.global_variables instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:216: The name tf.is_variable_initialized is
deprecated. Please use tf.compat.v1.is_variable_initialized instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:223: The name tf.variables_initializer is deprecated.
Please use tf.compat.v1.variables_initializer instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:2041: The name tf.nn.fused_batch_norm is deprecated.
Please use tf.compat.v1.nn.fused_batch_norm instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:148: The name tf.placeholder_with_default is
deprecated. Please use tf.compat.v1.placeholder_with_default instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:4271: The name tf.nn.avg_pool is deprecated. Please u
se tf.nn.avg_pool2d instead.
```

In [24]:

```
model = Model(inputs=[input], outputs=[output])
model.summary()
```

```
Model: "model_1"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_2 (InputLayer)            (None, 32, 32, 3)    0
__________________________________________________________________________________________________
conv2d_2 (Conv2D)               (None, 32, 32, 64)   1728        input_2[0][0]
__________________________________________________________________________________________________
batch_normalization_1 (BatchNor (None, 32, 32, 64)   256         conv2d_2[0][0]
__________________________________________________________________________________________________
activation_1 (Activation)       (None, 32, 32, 64)   0           batch_normalization_1[0][0]
__________________________________________________________________________________________________
conv2d_3 (Conv2D)               (None, 32, 32, 32)   18432       activation_1[0][0]
__________________________________________________________________________________________________
concatenate_1 (Concatenate)     (None, 32, 32, 96)   0           conv2d_2[0][0]
                                                                 conv2d_3[0][0]
__________________________________________________________________________________________________
batch_normalization_2 (BatchNor (None, 32, 32, 96)   384         concatenate_1[0][0]
__________________________________________________________________________________________________
activation_2 (Activation)       (None, 32, 32, 96)   0           batch_normalization_2[0][0]
__________________________________________________________________________________________________
conv2d_4 (Conv2D)               (None, 32, 32, 32)   27648       activation_2[0][0]
__________________________________________________________________________________________________
concatenate_2 (Concatenate)     (None, 32, 32, 128)  0           concatenate_1[0][0]
                                                                 conv2d_4[0][0]
__________________________________________________________________________________________________
batch_normalization_3 (BatchNor (None, 32, 32, 128)  512         concatenate_2[0][0]
__________________________________________________________________________________________________
activation_3 (Activation)       (None, 32, 32, 128)  0           batch_normalization_3[0][0]
__________________________________________________________________________________________________
conv2d_5 (Conv2D)               (None, 32, 32, 32)   36864       activation_3[0][0]
__________________________________________________________________________________________________
concatenate_3 (Concatenate)     (None, 32, 32, 160)  0           concatenate_2[0][0]
                                                                 conv2d_5[0][0]
__________________________________________________________________________________________________
batch_normalization_4 (BatchNor (None, 32, 32, 160)  640         concatenate_3[0][0]
__________________________________________________________________________________________________
activation_4 (Activation)       (None, 32, 32, 160)  0           batch_normalization_4[0][0]
```

conv2d_6 (Conv2D)	(None, 32, 32, 32)	46080	activation_4[0][0]
concatenate_4 (Concatenate)	(None, 32, 32, 192)	0	concatenate_3[0][0] conv2d_6[0][0]
batch_normalization_5 (BatchNor	(None, 32, 32, 192)	768	concatenate_4[0][0]
activation_5 (Activation)	(None, 32, 32, 192)	0	batch_normalization_5[0][0]
conv2d_7 (Conv2D)	(None, 32, 32, 32)	55296	activation_5[0][0]
concatenate_5 (Concatenate)	(None, 32, 32, 224)	0	concatenate_4[0][0] conv2d_7[0][0]
batch_normalization_6 (BatchNor	(None, 32, 32, 224)	896	concatenate_5[0][0]
activation_6 (Activation)	(None, 32, 32, 224)	0	batch_normalization_6[0][0]
conv2d_8 (Conv2D)	(None, 32, 32, 32)	64512	activation_6[0][0]
concatenate_6 (Concatenate)	(None, 32, 32, 256)	0	concatenate_5[0][0] conv2d_8[0][0]
batch_normalization_7 (BatchNor	(None, 32, 32, 256)	1024	concatenate_6[0][0]
activation_7 (Activation)	(None, 32, 32, 256)	0	batch_normalization_7[0][0]
conv2d_9 (Conv2D)	(None, 32, 32, 32)	8192	activation_7[0][0]
average_pooling2d_1 (AveragePoo	(None, 16, 16, 32)	0	conv2d_9[0][0]
batch_normalization_8 (BatchNor	(None, 16, 16, 32)	128	average_pooling2d_1[0][0]
activation_8 (Activation)	(None, 16, 16, 32)	0	batch_normalization_8[0][0]
conv2d_10 (Conv2D)	(None, 16, 16, 32)	9216	activation_8[0][0]
concatenate_7 (Concatenate)	(None, 16, 16, 64)	0	average_pooling2d_1[0][0] conv2d_10[0][0]
batch_normalization_9 (BatchNor	(None, 16, 16, 64)	256	concatenate_7[0][0]
activation_9 (Activation)	(None, 16, 16, 64)	0	batch_normalization_9[0][0]
conv2d_11 (Conv2D)	(None, 16, 16, 32)	18432	activation_9[0][0]
concatenate_8 (Concatenate)	(None, 16, 16, 96)	0	concatenate_7[0][0] conv2d_11[0][0]
batch_normalization_10 (BatchNo	(None, 16, 16, 96)	384	concatenate_8[0][0]
activation_10 (Activation)	(None, 16, 16, 96)	0	batch_normalization_10[0][0]
conv2d_12 (Conv2D)	(None, 16, 16, 32)	27648	activation_10[0][0]
concatenate_9 (Concatenate)	(None, 16, 16, 128)	0	concatenate_8[0][0] conv2d_12[0][0]
batch_normalization_11 (BatchNo	(None, 16, 16, 128)	512	concatenate_9[0][0]
activation_11 (Activation)	(None, 16, 16, 128)	0	batch_normalization_11[0][0]
conv2d_13 (Conv2D)	(None, 16, 16, 32)	36864	activation_11[0][0]
concatenate_10 (Concatenate)	(None, 16, 16, 160)	0	concatenate_9[0][0] conv2d_13[0][0]
batch_normalization_12 (BatchNo	(None, 16, 16, 160)	640	concatenate_10[0][0]
activation_12 (Activation)	(None, 16, 16, 160)	0	batch_normalization_12[0][0]
conv2d_14 (Conv2D)	(None, 16, 16, 32)	46080	activation_12[0][0]
concatenate_11 (Concatenate)	(None, 16, 16, 192)	0	concatenate_10[0][0] conv2d_14[0][0]
batch_normalization_13 (BatchNo	(None, 16, 16, 192)	768	concatenate_11[0][0]

batch_normalization_13 (BatchNo	(None, 16, 16, 192)	768	concatenate_11[0][0]
activation_13 (Activation)	(None, 16, 16, 192)	0	batch_normalization_13[0][0]
conv2d_15 (Conv2D)	(None, 16, 16, 32)	55296	activation_13[0][0]
concatenate_12 (Concatenate)	(None, 16, 16, 224)	0	concatenate_11[0][0] conv2d_15[0][0]
batch_normalization_14 (BatchNo	(None, 16, 16, 224)	896	concatenate_12[0][0]
activation_14 (Activation)	(None, 16, 16, 224)	0	batch_normalization_14[0][0]
conv2d_16 (Conv2D)	(None, 16, 16, 32)	7168	activation_14[0][0]
average_pooling2d_2 (AveragePoo	(None, 8, 8, 32)	0	conv2d_16[0][0]
batch_normalization_15 (BatchNo	(None, 8, 8, 32)	128	average_pooling2d_2[0][0]
activation_15 (Activation)	(None, 8, 8, 32)	0	batch_normalization_15[0][0]
conv2d_17 (Conv2D)	(None, 8, 8, 32)	9216	activation_15[0][0]
concatenate_13 (Concatenate)	(None, 8, 8, 64)	0	average_pooling2d_2[0][0] conv2d_17[0][0]
batch_normalization_16 (BatchNo	(None, 8, 8, 64)	256	concatenate_13[0][0]
activation_16 (Activation)	(None, 8, 8, 64)	0	batch_normalization_16[0][0]
conv2d_18 (Conv2D)	(None, 8, 8, 32)	18432	activation_16[0][0]
concatenate_14 (Concatenate)	(None, 8, 8, 96)	0	concatenate_13[0][0] conv2d_18[0][0]
batch_normalization_17 (BatchNo	(None, 8, 8, 96)	384	concatenate_14[0][0]
activation_17 (Activation)	(None, 8, 8, 96)	0	batch_normalization_17[0][0]
conv2d_19 (Conv2D)	(None, 8, 8, 32)	27648	activation_17[0][0]
concatenate_15 (Concatenate)	(None, 8, 8, 128)	0	concatenate_14[0][0] conv2d_19[0][0]
batch_normalization_18 (BatchNo	(None, 8, 8, 128)	512	concatenate_15[0][0]
activation_18 (Activation)	(None, 8, 8, 128)	0	batch_normalization_18[0][0]
conv2d_20 (Conv2D)	(None, 8, 8, 32)	36864	activation_18[0][0]
concatenate_16 (Concatenate)	(None, 8, 8, 160)	0	concatenate_15[0][0] conv2d_20[0][0]
batch_normalization_19 (BatchNo	(None, 8, 8, 160)	640	concatenate_16[0][0]
activation_19 (Activation)	(None, 8, 8, 160)	0	batch_normalization_19[0][0]
conv2d_21 (Conv2D)	(None, 8, 8, 32)	46080	activation_19[0][0]
concatenate_17 (Concatenate)	(None, 8, 8, 192)	0	concatenate_16[0][0] conv2d_21[0][0]
batch_normalization_20 (BatchNo	(None, 8, 8, 192)	768	concatenate_17[0][0]
activation_20 (Activation)	(None, 8, 8, 192)	0	batch_normalization_20[0][0]
conv2d_22 (Conv2D)	(None, 8, 8, 32)	55296	activation_20[0][0]
concatenate_18 (Concatenate)	(None, 8, 8, 224)	0	concatenate_17[0][0] conv2d_22[0][0]
batch_normalization_21 (BatchNo	(None, 8, 8, 224)	896	concatenate_18[0][0]
activation_21 (Activation)	(None, 8, 8, 224)	0	batch_normalization_21[0][0]
conv2d_23 (Conv2D)	(None, 8, 8, 32)	7168	activation_21[0][0]
average_pooling2d_3 (AveragePoo	(None, 4, 4, 32)	0	conv2d_23[0][0]

batch_normalization_22 (BatchNo	(None, 4, 4, 32)	128	average_pooling2d_3[0][0]
activation_22 (Activation)	(None, 4, 4, 32)	0	batch_normalization_22[0][0]
conv2d_24 (Conv2D)	(None, 4, 4, 32)	9216	activation_22[0][0]
concatenate_19 (Concatenate)	(None, 4, 4, 64)	0	average_pooling2d_3[0][0] conv2d_24[0][0]
batch_normalization_23 (BatchNo	(None, 4, 4, 64)	256	concatenate_19[0][0]
activation_23 (Activation)	(None, 4, 4, 64)	0	batch_normalization_23[0][0]
conv2d_25 (Conv2D)	(None, 4, 4, 32)	18432	activation_23[0][0]
concatenate_20 (Concatenate)	(None, 4, 4, 96)	0	concatenate_19[0][0] conv2d_25[0][0]
batch_normalization_24 (BatchNo	(None, 4, 4, 96)	384	concatenate_20[0][0]
activation_24 (Activation)	(None, 4, 4, 96)	0	batch_normalization_24[0][0]
conv2d_26 (Conv2D)	(None, 4, 4, 32)	27648	activation_24[0][0]
concatenate_21 (Concatenate)	(None, 4, 4, 128)	0	concatenate_20[0][0] conv2d_26[0][0]
batch_normalization_25 (BatchNo	(None, 4, 4, 128)	512	concatenate_21[0][0]
activation_25 (Activation)	(None, 4, 4, 128)	0	batch_normalization_25[0][0]
conv2d_27 (Conv2D)	(None, 4, 4, 32)	36864	activation_25[0][0]
concatenate_22 (Concatenate)	(None, 4, 4, 160)	0	concatenate_21[0][0] conv2d_27[0][0]
batch_normalization_26 (BatchNo	(None, 4, 4, 160)	640	concatenate_22[0][0]
activation_26 (Activation)	(None, 4, 4, 160)	0	batch_normalization_26[0][0]
conv2d_28 (Conv2D)	(None, 4, 4, 32)	46080	activation_26[0][0]
concatenate_23 (Concatenate)	(None, 4, 4, 192)	0	concatenate_22[0][0] conv2d_28[0][0]
batch_normalization_27 (BatchNo	(None, 4, 4, 192)	768	concatenate_23[0][0]
activation_27 (Activation)	(None, 4, 4, 192)	0	batch_normalization_27[0][0]
conv2d_29 (Conv2D)	(None, 4, 4, 32)	55296	activation_27[0][0]
concatenate_24 (Concatenate)	(None, 4, 4, 224)	0	concatenate_23[0][0] conv2d_29[0][0]
batch_normalization_28 (BatchNo	(None, 4, 4, 224)	896	concatenate_24[0][0]
activation_28 (Activation)	(None, 4, 4, 224)	0	batch_normalization_28[0][0]
conv2d_30 (Conv2D)	(None, 4, 4, 10)	2240	activation_28[0][0]
average_pooling2d_4 (AveragePoo	(None, 2, 2, 10)	0	conv2d_30[0][0]
global_average_pooling2d_1 (Glo	(None, 10)	0	average_pooling2d_4[0][0]
activation_29 (Activation)	(None, 10)	0	global_average_pooling2d_1[0][0]

Total params: 871,168
Trainable params: 863,552
Non-trainable params: 7,616

In [0]:

```
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
```

```
x_test = x_test.astype('float32')
```

In [0]:

```
mean = x_train.mean(0)
std = x_train.std(0)
```

In [0]:

```
def preprocess_data(dataset):
    dataset -= mean
    dataset /= std
    return dataset
```

In [0]:

```
x_train = preprocess_data(x_train)
x_test = preprocess_data(x_test)
```

In [0]:

```
# Data augementation
from keras.preprocessing.image import ImageDataGenerator
datagen_train = ImageDataGenerator(
    width_shift_range=0.125,
    height_shift_range=0.125,
    horizontal_flip=True,
)

datagen_train.fit(x_train)
```

In [0]:

```
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping, TensorBoard,
ReduceLROnPlateau

checkpoint_3 = ModelCheckpoint("model_dense.h5",monitor="val_acc",mode="max",save_best_only = True
,verbose=1)
NAME = 'model_dense'
tensorboard2 =
TensorBoard(log_dir='logss\{}'.format(NAME),update_freq='epoch',batch_size=batch_size)
callbacks2 = [tensorboard2,checkpoint_3]
```

In [0]:

```
# determine Loss function and Optimizer
model.compile(loss='categorical_crossentropy',
              optimizer=Adam(),
              metrics=['accuracy'])
```

In [43]:

```
#https://machinelearningmastery.com/check-point-deep-learning-models-keras/
from keras.callbacks import ModelCheckpoint

history = model.fit_generator(datagen_train.flow(x_train, y_train,
batch_size=batch_size),steps_per_epoch=(len(x_train)/batch_size)*5,
    epochs=epochs,
    verbose = 1,
    validation_data=(x_test, y_test),
)
```

```
Epoch 1/35
1954/1953 [==============================] - 348s 178ms/step - loss: 0.8636 - acc: 0.6934 - val_lo
ss: 0.7326 - val_acc: 0.7601
Epoch 2/35
1954/1953 [==============================] - 330s 169ms/step - loss: 0.4543 - acc: 0.8426 - val_lo
ss: 0.5687 - val_acc: 0.8151
Epoch 3/35
1954/1953 [==============================] - 331s 169ms/step - loss: 0.3417 - acc: 0.8810 - val lo
```

```
ss: 0.6181 - val_acc: 0.8186
Epoch 4/35
1954/1953 [==============================] - 332s 170ms/step - loss: 0.2757 - acc: 0.9040 - val_lo
ss: 0.4066 - val_acc: 0.8688
Epoch 5/35
1954/1953 [==============================] - 331s 169ms/step - loss: 0.2287 - acc: 0.9198 - val_lo
ss: 0.4511 - val_acc: 0.8612
Epoch 6/35
1954/1953 [==============================] - 331s 170ms/step - loss: 0.1922 - acc: 0.9328 - val_lo
ss: 0.4016 - val_acc: 0.8775
Epoch 7/35
1954/1953 [==============================] - 331s 169ms/step - loss: 0.1650 - acc: 0.9419 - val_lo
ss: 0.5034 - val_acc: 0.8633
Epoch 8/35
1954/1953 [==============================] - 331s 169ms/step - loss: 0.1428 - acc: 0.9495 - val_lo
ss: 0.5385 - val_acc: 0.8616
Epoch 9/35
1954/1953 [==============================] - 331s 169ms/step - loss: 0.1231 - acc: 0.9560 - val_lo
ss: 0.4800 - val_acc: 0.8756
Epoch 10/35
1954/1953 [==============================] - 330s 169ms/step - loss: 0.1099 - acc: 0.9605 - val_lo
ss: 0.4703 - val_acc: 0.8782
Epoch 11/35
1954/1953 [==============================] - 330s 169ms/step - loss: 0.0964 - acc: 0.9655 - val_lo
ss: 0.4458 - val_acc: 0.8803
Epoch 12/35
1954/1953 [==============================] - 330s 169ms/step - loss: 0.0879 - acc: 0.9683 - val_lo
ss: 0.4594 - val_acc: 0.8882
Epoch 13/35
1954/1953 [==============================] - 330s 169ms/step - loss: 0.0792 - acc: 0.9717 - val_lo
ss: 0.4088 - val_acc: 0.8968
Epoch 14/35
1954/1953 [==============================] - 331s 169ms/step - loss: 0.0712 - acc: 0.9743 - val_lo
ss: 0.5100 - val_acc: 0.8776
Epoch 15/35
1954/1953 [==============================] - 331s 170ms/step - loss: 0.0664 - acc: 0.9763 - val_lo
ss: 0.5088 - val_acc: 0.8854
Epoch 16/35
1954/1953 [==============================] - 330s 169ms/step - loss: 0.0619 - acc: 0.9781 - val_lo
ss: 0.4823 - val_acc: 0.8903
Epoch 17/35
1954/1953 [==============================] - 331s 169ms/step - loss: 0.0566 - acc: 0.9796 - val_lo
ss: 0.4534 - val_acc: 0.8979
Epoch 18/35
1954/1953 [==============================] - 330s 169ms/step - loss: 0.0532 - acc: 0.9813 - val_lo
ss: 0.5110 - val_acc: 0.8904
Epoch 19/35
1954/1953 [==============================] - 331s 169ms/step - loss: 0.0501 - acc: 0.9822 - val_lo
ss: 0.4118 - val_acc: 0.9032
Epoch 20/35
1954/1953 [==============================] - 331s 169ms/step - loss: 0.0474 - acc: 0.9832 - val_lo
ss: 0.4715 - val_acc: 0.8966
Epoch 21/35
1954/1953 [==============================] - 330s 169ms/step - loss: 0.0452 - acc: 0.9840 - val_lo
ss: 0.5049 - val_acc: 0.8910
Epoch 22/35
1954/1953 [==============================] - 331s 169ms/step - loss: 0.0432 - acc: 0.9847 - val_lo
ss: 0.5037 - val_acc: 0.8963
Epoch 23/35
1954/1953 [==============================] - 331s 169ms/step - loss: 0.0390 - acc: 0.9864 - val_lo
ss: 0.5166 - val_acc: 0.8996
Epoch 24/35
1954/1953 [==============================] - 331s 169ms/step - loss: 0.0381 - acc: 0.9867 - val_lo
ss: 0.4750 - val_acc: 0.9045
Epoch 25/35
1954/1953 [==============================] - 332s 170ms/step - loss: 0.0369 - acc: 0.9870 - val_lo
ss: 0.4700 - val_acc: 0.9066
Epoch 26/35
1954/1953 [==============================] - 332s 170ms/step - loss: 0.0338 - acc: 0.9880 - val_lo
ss: 0.5411 - val_acc: 0.8958
Epoch 27/35
1954/1953 [==============================] - 332s 170ms/step - loss: 0.0335 - acc: 0.9881 - val_lo
ss: 0.4692 - val_acc: 0.9073
Epoch 28/35
1954/1953 [==============================] - 332s 170ms/step - loss: 0.0320 - acc: 0.9886 - val_lo
ss: 0.4441 - val_acc: 0.9066
Epoch 29/35
```

```
1954/1953 [==============================] - 332s 170ms/step - loss: 0.0309 - acc: 0.9893 - val_lo
ss: 0.5578 - val_acc: 0.8950
Epoch 30/35
1954/1953 [==============================] - 332s 170ms/step - loss: 0.0307 - acc: 0.9892 - val_lo
ss: 0.5403 - val_acc: 0.8983
Epoch 31/35
1954/1953 [==============================] - 331s 170ms/step - loss: 0.0282 - acc: 0.9903 - val_lo
ss: 0.4933 - val_acc: 0.9034
Epoch 32/35
1954/1953 [==============================] - 331s 170ms/step - loss: 0.0276 - acc: 0.9902 - val_lo
ss: 0.4794 - val_acc: 0.9102
Epoch 33/35
1954/1953 [==============================] - 331s 169ms/step - loss: 0.0278 - acc: 0.9904 - val_lo
ss: 0.6699 - val_acc: 0.8815
Epoch 34/35
1954/1953 [==============================] - 331s 169ms/step - loss: 0.0255 - acc: 0.9910 - val_lo
ss: 0.4926 - val_acc: 0.9079
Epoch 35/35
1954/1953 [==============================] - 331s 170ms/step - loss: 0.0253 - acc: 0.9912 - val_lo
ss: 0.5071 - val_acc: 0.9066
```

In [45]:

```
# Test the model
score = model.evaluate(x_test, y_test, verbose=1)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

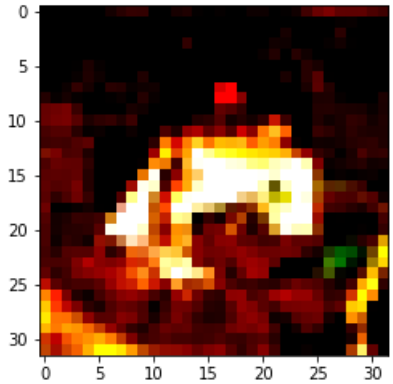
```
10000/10000 [==============================] - 5s 465us/step
Test loss: 0.5070862675895914
Test accuracy: 0.9066
```

In [47]:

```
img_to_visualize = x_train[0]
plt.imshow(img_to_visualize)

img_to_visualize = np.expand_dims(img_to_visualize, axis=0)
```

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for
integers).
```



In [0]:

```
```