# Keras_bottleneck_features

December 15, 2017

```
In [ ]: import numpy as np
        import pandas as pd
        import keras
        from keras.applications.inception_v3 import preprocess_input
        from keras.models import Model
        from keras.layers import Dense, Dropout, Flatten, Input, Lambda, GlobalAveragePooling2D
        from keras.preprocessing import image

        import os
        from tqdm import tqdm
```

```
In [ ]: from keras.applications import xception
        from keras.applications import inception_v3, resnet50
```

```
In [ ]: data_dir = os.getcwd()
        df_train = pd.read_csv('labels.csv')
        df_test = pd.read_csv('sample_submission.csv')
```

```
In [ ]: df_train.head(10)
```

```
In [ ]: target_series = pd.Series(df_train['breed'])
        one_hot = pd.get_dummies(target_series, sparse=True)
```

```
In [ ]: one_hot_labels = np.asarray(one_hot)
```

```
In [ ]: def read_img(img_id, train_or_test, size):
            """Read and resize image.
            # Arguments
                img_id: string
                train_or_test: string 'train' or 'test'.
                size: resize the original image.
            # Returns
                Image as numpy array.
            """
            img = image.load_img(os.path.join(data_dir, train_or_test, '%s.jpg' % img_id), targe
            #img = image.img_to_array(img)
            return img
```

```
In [ ]: IM_SIZE = 299
In [ ]: x_train = np.zeros((len(df_train), IM_SIZE, IM_SIZE, 3), dtype=np.uint8)
        y_train = np.zeros((one_hot_labels.shape), dtype=np.uint8)
        for i, img_id in tqdm(enumerate(df_train['id'])):
            img = read_img(img_id, 'train', (IM_SIZE, IM_SIZE))
            x_train[i] = img
            y_train[i] = one_hot_labels[i]

        print('Train Images shape: {} size: {:,}'.format(x_train.shape, x_train.size))
```

```
In [ ]: print(y_train.shape, x_train.shape)
```

```
In [ ]: num_class = y_train.shape[1]
```

### 0.0.1 Extract Xception and Inception bottleneck features

```
In [ ]: def get_features(MODEL, data=x_train):
            cnn_model = MODEL(include_top=False, input_shape=(IM_SIZE, IM_SIZE, 3), weights='ima

            inputs = Input((IM_SIZE, IM_SIZE, 3))
            x = inputs
            x = Lambda(preprocess_input, name='preprocessing')(x)
            x = cnn_model(x)
            x = GlobalAveragePooling2D()(x)
            cnn_model = Model(inputs, x)

            features = cnn_model.predict(data, batch_size=64, verbose=1)
            return features
```

```
In [ ]: inception_features = get_features(inception_v3.InceptionV3, x_train)
        xception_features = get_features(xception.Xception, x_train)
```

```
In [ ]: resnet_features = get_features(resnet50.ResNet50, x_train)
```

```
In [ ]: features = np.concatenate([inception_features, xception_features, resnet_features], axis
```

### 0.0.2 Model training

```
In [ ]: sgd = keras.optimizers.SGD(lr=0.01, momentum=0.9, decay=1e-2)
        callbacks = [keras.callbacks.EarlyStopping(monitor='val_loss',
                                                    patience=10, verbose=1),
                     keras.callbacks.ReduceLROnPlateau(monitor='val_loss', factor=0.1,
                                                  patience=3, verbose=1)]
```

```
In [ ]: inputs = Input(features.shape[1:])
        x = inputs
        x = Dropout(0.5)(x)
        x = Dense(num_class, activation='softmax')(x)
        model = Model(inputs, x)
```

```
In [ ]: model.compile(loss='categorical_crossentropy', optimizer=sgd,
                      metrics=['accuracy'])
        model.summary()
```

```
In [ ]: model.fit(features, y_train, batch_size=128, epochs=150, validation_split=0.1, verbose=1
```

### 0.0.3 Testing

```
In [ ]: x_test = np.zeros((len(df_test), IM_SIZE, IM_SIZE, 3), dtype='float32')
        for i, img_id in tqdm(enumerate(df_test['id'])):
            img = read_img(img_id, 'test', (IM_SIZE, IM_SIZE))
            x_test[i] = img

        print('Test Images shape: {} size: {:,}'.format(x_test.shape, x_test.size))
```

```
In [ ]: test_x_features = get_features(xception.Xception, x_test)
        test_i_features = get_features(inception_v3.InceptionV3, x_test)
        test_resnet_features = get_features(resnet50.ResNet50, x_test)
        test_features = np.concatenate([test_i_features, test_x_features, test_resnet_features],
```

```
In [ ]: y_pred = model.predict(test_features, batch_size=128)
```

```
In [ ]: sub = pd.DataFrame(y_pred)
        col_names = one_hot.columns.values
        sub.columns = col_names

        sub.insert(0, 'id', df_test['id'])
        sub.head(10)
```

```
In [ ]: sub.to_csv('inc_exc_submission1.csv', index=False)
```

```
In [ ]: sub.tail()
```