Практика:
Введение в Keras

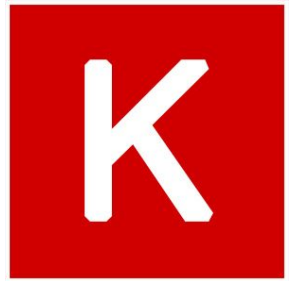
Grigory Sapunov  CTO / Intento

HSE / #3 / 18.09.2018

Введение в Keras

Keras

- Высокоуровневая библиотека для работы с нейросетями в Python.
- Очень удобна для экспериментирования и быстрого прототипирования.
- Может использовать TensorFlow, CNTK, mxnet или Theano как бэкенд.
- Использует CPU, GPU.
- Поддерживает CNN, RNN
- https://keras.io/
- https://github.com/fchollet/keras



Создание модели

```
from keras.models import Sequential

model = Sequential()
```

Sequential модель — это линейная последовательность слоёв.

Есть также более сложный функциональный API, позволяющий строить более сложные модели как модели со множеством выходов, DAG (directed acyclic graphs) или модели с разделяемыми слоями.

Создание модели

```
from keras.layers import Dense, Activation

model.add(Dense(units=64, input_dim=100))
model.add(Activation("relu"))
model.add(Dense(units=10))
model.add(Activation("softmax"))
```

Или проще

```
model = Sequential([
    Dense(32, input_dim=784),
    Activation('relu'),
    Dense(10),
    Activation('softmax'),
])
```

Создание модели

Есть также более сложный функциональный API, позволяющий строить более сложные модели как модели со множеством выходов, DAG (directed acyclic graphs) или модели с разделяемыми слоями.

```
from keras.layers import Input, Dense
from keras.models import Model

inputs = Input(shape=(784,)) # This returns a tensor

# a layer instance is callable on a tensor, and returns a tensor
x = Dense(64, activation='relu')(inputs)
x = Dense(64, activation='relu')(x)
predictions = Dense(10, activation='softmax')(x)

# This creates a model that includes the Input layer and three Dense layers
model = Model(inputs=inputs, outputs=predictions)
```

Создание модели

```
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 32)                25120
_________________________________________________________________
activation_1 (Activation)    (None, 32)                0
_________________________________________________________________
dense_2 (Dense)              (None, 10)                330
_________________________________________________________________
activation_2 (Activation)    (None, 10)                0
=================================================================
Total params: 25,450
Trainable params: 25,450
Non-trainable params: 0
_________________________________________________________________
```

Компиляция модели

Готовая модель компилируется:

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

Через компиляцию настраиваются параметры оптимизатора:

```
from keras.optimizers import SGD

model.compile(loss='categorical_crossentropy', optimizer=SGD(lr=0.01,
momentum=0.9, nesterov=True))
```

Обучение модели

Скомпилированная модель обучается:

```
model.fit(X_train, Y_train, nb_epoch=5, batch_size=32) # keras 1

model.fit(x_train, y_train, batch_size=32, epochs=5,
    verbose=1,
    validation_data=(x_test, y_test))
```

Обучение модели

Можно обучать отдельными батчами:

```
model.train_on_batch(X_batch, Y_batch)
```

Можно обучать на генераторе:

```
model.fit_generator(
    datagen.flow(x_train, y_train, batch_size=batch_size),
    steps_per_epoch=x_train.shape[0] // batch_size,
    epochs=epochs,
    validation_data=(x_test, y_test))
```

Использование модели

Обученную модель можно оценить:

```
score = model.evaluate(x_test, y_test, batch_size=batch_size, verbose=1)
```

Или использовать для предсказания:

```
classes = model.predict_classes(X_test, batch_size=32)
proba = model.predict_proba(X_test, batch_size=32)
```

Более подробная справка:

https://keras.io/getting-started/sequential-model-guide/

Jupyter notebook example