Home

Keras: библиотека глубокого обучение на Python

Вы только что открыли для себя Keras.

Keras — это библиотека глубокого обучения, представляющая из себя высокоуровневый API, написанный на Python и способный работать поверх TensorFlow, Theano или CNTK. Он был разработан с расчетом на быстрое обучение. Способность переходить от гипотез к результаты с наименьшими временными затратами является ключом к проведению успешных исследований.

Используйте Keras, если Вам нужна библиотека глубокого обучения, которая:

- позволяет легко и быстро создавать прототипы (благодаря удобству, модульности и масштабированию);
- поддерживает как сверточные и рекуррентные сети, так и их комбинации;
- без проблем работает как на процессоре (CPU), так и на графическом процессоре (GPU).

Keras совместим с Python 2.7 — 3.6.

Многоуровневый бэкенд Keras и tf.keras.

В настоящее время мы рекомендуем пользователям Keras, которые используют многоуровневый бэкенд Keras с бэкендом TensorFlow, переключиться на tf.keras в TensorFlow 2.0. tf.keras имеет более качественную поддержку и интеграцию с функциями TensorFlow.

Keras 2.2.5 — последняя версия Keras, реализующая API 2.2. и поддерживая только TensorFlow 1 (а также Theano и CNTK).

Текущая версия Keras — Keras 2.3.0, в которой внесены существенные изменения в API и добавлена поддержка TensorFlow 2.0. Версия Keras 2.3.0 будет последней версией многоуровневого бэкенда Keras, который заменяется на tf.keras.

Ошибки многоуровневого бэкенда Keras будут исправляться только до апреля 2020 г. (в качестве небольших изменений версии).

Больше информации о планируемых изменениях Keras смотрите на: the Keras meeting notes.

Основные идеи.

- **Удобство для пользователя.** Keras — это API, разработанный для людей, а не для машин. Это ставит пользовательский опыт в основу всего. Keras использует передовые методы снижения когнитивной нагрузки: он предлагает согласованный и простой API, минимизирует количество действий пользователя, необходимых для решения распространенных задач, предоставляет четкую и действенную обратную связь в случае возникновения ошибок.
- **Модульность.** Под моделью понимается последовательность или граф автономных, полностью сконфигурированных модулей, которые могут быть подключены без каких-либо дополнительных ограничений. Например нейронные слои, функции ошибки, оптимизаторы, схемы инициализации, функции активации и схемы регуляризации — все это отдельные модули, которые можно комбинировать для создания модели.
- **Расширяемость.** Новые модули легко добавлять (так же как новые классы и функции), а существующую модули предоставляют достаточное количество подобных примеров. Возможность добавлять новые модули делает Keras максимально привлекательным средством для проведения передовых исследований.
- **Работа в Python.** Все модели написаны на Python, благодаря чему код компактен, легко читается и отлаживается, а также легко расширяется.

Первые шаги: 30 секунд до Keras.

Базовая структура данных библиотеки Keras — это модель, описывающая способ организации слоев. Простейшим типом модели является модель Sequential, представляющая собой линейный стек слоев. Для более сложных архитектур вы должны использовать функциональное программирование (Keras functional API), которое позволяет строить произвольные графы слоев.

Модель Sequential:

```
from keras.models import Sequential
model = Sequential()
```

Добавлять слои можно с помощью метода .add():

```
from keras.models import Dense
model.add(Dense(units=64, activation='relu', input_dim=100))
model.add(Dense(units=10, activation='softmax'))
```

После того, как Ваша модель будет собрана, вызовите метод .compile() для подготовки модели к процессу обучения:

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

В случае необходимости Вы можете самостоятельно настроить оптимизатор. Основной принцип Keras — сделать процесс максимально простым, при этом позволяя пользователю полностью его контролировать по мере необходимости.

```
model.compile(loss=keras.losses.categorical_crossentropy,

optimizer=keras.optimizers.SGD(lr=0.01,momentum=0.9,nesterov=Tr
```

Теперь Вы можете подавать пакеты данных для обучения:

```
# x_train и y_train это Numpy массивы
model.fit(x_train, y_train, epochs=5, batch_size=32)
```

Помимо этого, вы можете обучать вашу модель вручную:

```
model.train_on_batch(x_batch, y_batch)
```

Оценить качество обучения можно следующим образом:

```
loss_and_metrics = model.evaluate(x_test, y_test,
batch_size=128)
```

Или вызвать метод .predict() для тестового набора:

```
classes = model.predict(x_test, batch_size=128)
```

Построение систем вопрос-ответ, систем классификации изображений, нейронной машины Тьюринга происходит так же быстро, как и рассмотренные примеры. Идеи глубокого обучения довольно просты, так почему же их реализация должна быть сложной?

Более подробные обучающие материалы Вы можете найти по ссылкам:

- Sequential model: руководство
- Functional API: руководство

В папке репозитория Вы можете найти более продвинутые модели: вопрос-ответ в сетях с памятью, генерация текста с использованием LSTM и др.

Установка.

Перед установкой Keras, пожалуйста, установите один из его движков: TensorFlow, Theano или CNTK. Мы рекомендуем TensorFlow.

- Инструкция по установке TensorFlow.
- Инструкция по установке Theano.
- Инструкция по установке CNTK.

Вам также могут понадобиться следующие компоненты:

- cuDNN (требуется, если Вы планируете проводить расчеты с использованием GPU).
- HDF5 и h5py (требуется, если Вы планируете сохранять модели Keras на диск).

- graphviz и pydot (Используются утилитами визуализации для построения графов моделей).

Затем Вы можете установить Keras. Есть два способа это сделать:

- **Установка Keras из PyPI (рекомендуется):**

Примечание: приведенные ниже этапы установки предполагают, что Вы работает в ОС Linux или Mac. Если Вы работаете в Windows, то используйте указанные команды без ключевого слова sudo.

```
sudo pip install keras
```

Если вы используете виртуальную среду Python, то можете не использовать ключевое слово sudo:

```
pip install keras
```

- **Альтернативный вариант: установка Keras из репозитория GitHub:**

Во-первых, выполните clone Keras'a, используя команду git:

```
git clone https://github.com/keras-team/keras.git
```

Затем перейдите в папку, в которой находится библиотека Keras, и выполните команду установки:

```
cd keras
sudo python setup.py install
```

Поддержка.

Вы можете задавать вопросы и присоединиться к обсуждению процесса разработки:

- В группе Keras Google
- На канале Keras Slack. Используйте эту ссылку для запроса приглашения на канал.

Вы также можете публиковать свои отчеты об ошибках и свои вопросы (только) в GitHub. Обязательно сначала прочитайте наши рекомендации.

Почему выбрано имя: Keras?

Keras (κέρας) — означает рог на греческом языке. Это отсылка к литературному образу из древнегреческой и латинской литературы, впервые найденному в Одиссее, где духи сновидений (Ониры) поделены на две группы. Одни обманывают людей с ложными видениями, попадая на Землю через ворота из слоновой кости. Другие сообщают людям о грядущем будущем, попадая на Землю через ворота рога. Получается игра слов: κέρας (рог) / κραίνω (выполнить) и ἐλέφας (слоновая кость) / ἐλεφαίρομαι (обмануть).

Изначально Keras был разработан в рамках исследовательского проекта ONEIROS (открытая нейро-электронная интеллектуальная операционная система роботов (Open-ended Neuro-Electronic Intelligent Robot Operating System)).

“Странник, конечно бывают и темные сны, из которых

Смысла нельзя нам извлечь; и не всякий сбывается сон наш.

Создано двое ворот для вступления снам бестелесным

В мир наш: одни роговые, другие из кости слоновой;

Сны, проходящие к нам воротами из кости слоновой,

Лживы, несбыточны, верить никто из людей им не должен;

Те же, которые в мир роговыми воротами входят,Верны; сбываются все приносимые ими виденья”
Гомер, Одиссея 19, 562 (перевод В.А.Жуковского)