Open for Innovation
KNIME

KNIME Integration von Deep Learning

Installationsanleitung

KNIME AG, Zürich, Schweiz

Version 5.7 (letzte Aktualisierung auf)

Inhaltsverzeichnis

Einleitung

Dieses Dokument beschreibt die Installation der KNIME Deep Learning Integrations. Folgender tiefe Lernbibliotheken wurden integriert:

- Die <a href="#page6" style="color: #ff6600; text-decoration: underline;">KNIME Integration von K basierend auf Keras tiefen Lernrahmen.
- Die <a href="#page8" style="color: #ff6600; text-decoration: underline;">KNIME TensorFlow 1 Int basierend auf TensorFlow 1 *
- Die <a href="#page10" style="color: #ff6600; text-decoration: underline;">KNIME TensorFlow 2 I basierend auf TensorFlow 2 *

* TensorFlow, die TensorFlow Logo und eine verwandt Markierun gen werd en Marken vo n Google Inc.

- Die <a href="#page10" style="color: #ff6600; text-decoration: underline;">KNIME ONNX Integrati basierend auf EINZELN.
- Die <a href="#page11" style="color: #ff6600; text-decoration: underline;">KNIME Tiefbau4j Integr basierend auf Tiefbau4j .

Die **KNIME Integration von Keras** , die **KNIME TensorFlow 1 Integration** , die **KNIME TensorFlow 2 Integration** und **KNIME ONNX Integration** abhängig von einer bestehenden Python-Installation, die bestimmte Python-Abhängigkeiten zu installieren erfordert. Um Python für die Deep Learning Integrations, folgen Sie den Anweisungen in der <a href="#page2" style="color: #ff6600; text-d Lernen Abschnitt.

Detaillierte Anleitungen zur manuellen Installation und Einrichtung der einzelnen KNIME Deep Lernintegrationen, siehe die entsprechenden Abschnitte unten.

Auf Apple-Computern mit den Apple Silicon Chips müssen Sie eine Umgebung mit spezifischen Paketen, um das KNIME Deep Learning zu nutzen Integration. Sie können Folgendes verwenden: Arbeitsablauf aus der KNIME-Gemeinschaft Hub zum Erstellen einer Conda-Umgebung mit allen erforderlichen Paketen

Python für KNIME einrichten Tiefen lernen

Dieser Abschnitt beschreibt, wie Python für die **KNIME Integration von Keras** , die **KNIME TensorFlow 1 Integration** , die **KNIME TensorFlow 2 Integration** und **KNIME ONNX Integration** .

Anaconda Setup

Ähnlich wie bei der KNIME Python Integration, KNIME Deep Learning verwendet Anaconda, um zu verwalten Python-Umgebungen (für eine kurze Einführung in Anaconda und wie es in KNIME verwendet wird, siehe und Abschnitt Wenn Sie bereits Anaconda installiert haben, z.B. für die KNIME Python Integration, skip diesen Schritt.

Erhalten Sie und installieren Sie die neueste Anaconda-Version (Anaconda ≥ 2019.03, conda ≥ ANHANG) aus Hier. . Auf der Anaconda Download-Seite können Sie zwischen Anaconda und Python 3.x wählen. oder Python 2.x, jedoch wirkt dies nur auf die Wurzel Conda Umwelt, die wir nicht nutzen werden (wie wir schaffen unsere eigenen). Daher können Sie entweder eine wählen (wenn Sie nicht sicher sind, wir Vorschlag für die Auswahl von Python 3).

Installation

Als nächstes, installieren Sie die gewünschten Deep-Learning-Integrationen mit der KNIME Analytics Plattform Seite aktualisieren In der KNIME Analytics Platform Datei → KNIME installieren Erweiterungen . Die Integrationen finden Sie unter KNIME Laborerweiterungen oder durch Eingabe von Deep Learning in die Suchfeld.

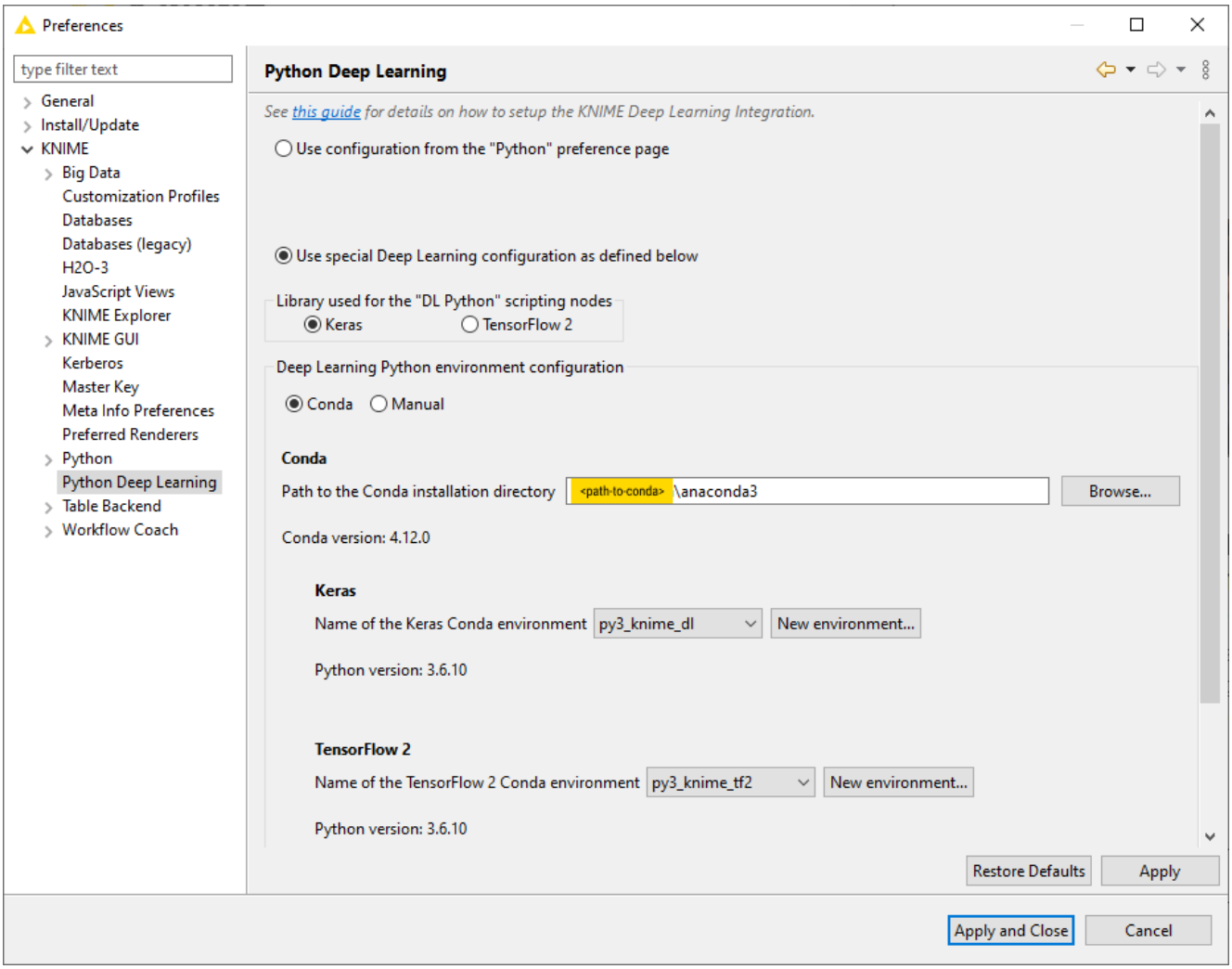
Konfigurieren Sie Python für KNIME Tiefen lernen

Nachdem die Integrationen installiert wurden, gehen Sie auf Python Deep Learning Preference Seite befindet sich in Datei → Vorlieben , dann wählen KNIME → Python Deep Learning aus der Liste auf der links. Ein Dialog öffnet Ihnen zwei Optionen zur Konfiguration des Python Deep Learning Umwelt:

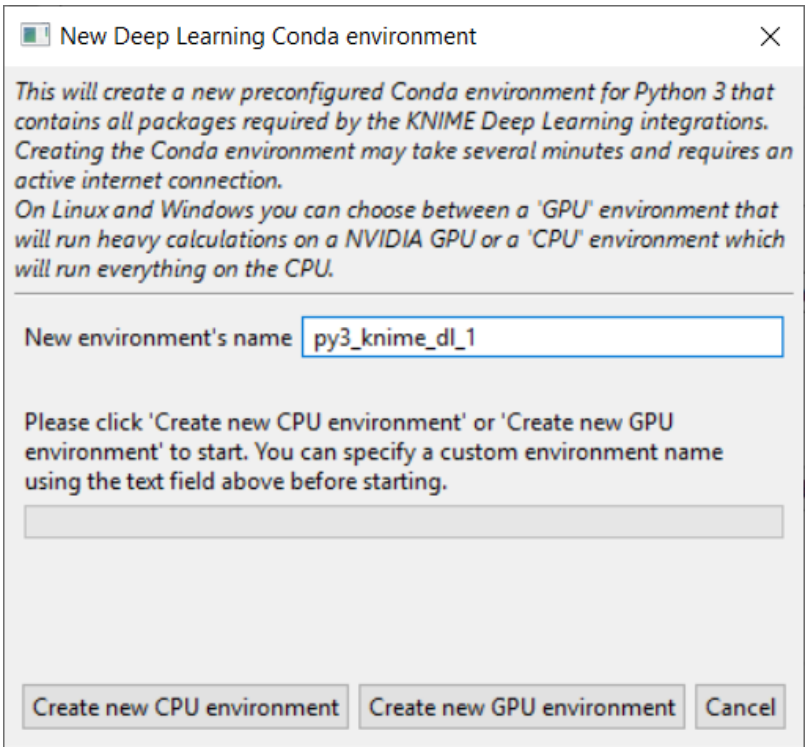
Option 1: Verwenden Sie spezielle Deep Learning-Konfiguration (empfohlen)

Durch Auswahl dieser Option, KNIME Deep Learning wird eine Python 3 Umgebung von die in der Python-Präferenz-Seite konfigurierte.

Ähnlich wie bei der Python Preference-Seite, können Sie entweder automatisch Python 3 erstellen Umgebungen, die alle erforderlichen Pakete enthalten (durch Auswahl der **Conda** Suboption oder Punkt zu Python Startskripte, die eine geeignete Umgebung aktivieren, die Sie manuell erstellt haben (von Auswahl der **Handbuch** Suboption). Wir empfehlen zu wählen **Conda** . Wenn Sie dies tun, der Dialog sollte wie der unten gezeigte Screenshot aussehen.



Geben Sie in diesem Dialog den Pfad zum Ordner mit Ihrem Anaconda Installation (Standard) Installationspfad wird dokumentiert Hier.) Sobald Sie einen gültigen Pfad eingegeben haben, installiert Conda wird die Version angezeigt und die KNIME Analytics Platform prüft automatisch alle verfügbaren Conda Umwelt. Unterhalb des Conda Versionsnummer, Sie können wählen, welche Conda für die KNIME Keras-Knoten und für die KNIME TensorFlow 2 Knoten durch die Auswahl aus den Combo-Boxen. Wenn Sie bereits einen Python Deep einrichten Lernumgebung mit allen notwendigen Abhängigkeiten für KNIME Deep Learning, Wählen Sie es einfach aus der Liste und Sie sind bereit zu gehen. Wenn Sie keine geeignete Umgebung haben verfügbar, klicken Sie auf Neue Umgebung... Knopf. Dies öffnet den folgenden Dialog:



Geben Sie zunächst einen Namen für die neue Umgebung. Als nächstes wählen Sie, ob Sie eine neue CPU erstellen möchten oder GPU-Umgebung und klicken Sie auf die entsprechende Schaltfläche. Dies schafft ein neues Conda Umwelt mit allen erforderlichen Python-Abhängigkeiten für Keras oder für TensorFlow 2.

Wählen Sie nur GPU, wenn Sie eine TensorFlow kompatible GPU zur Verfügung haben. Mehr Informationen zur Unterstützung von Keras GPU finden Sie . Mehr Informationen über TensorFlow 2 GPU-Unterstützung finden .

Abhängig von Ihrer Internetverbindung kann die Umwelterstellung eine während alle Pakete heruntergeladen und extrahiert werden müssen.

Sobald die Umgebung erfolgreich angelegt ist, schließt sich der Dialog und die neue Umgebung ist automatisch ausgewählt. Wenn alles gut funktioniert, wird die Python-Version nun unten gezeigt die Umweltauswahl.

Auswahl der Umgebung für die DL Python-Knoten

Über der Deep Learning Python Umgebungskonfiguration , Sie können wählen, welche Bibliothek für die "DL Python" Scripting-Knoten . wennKeras wird die Umgebung ausgewählt ausgewählt für die Keras-Knoten werden verwendet und die Keras Python-Bibliothek (und der TensorFlow 1 Python Bibliothek) wird verfügbar sein, wenn Sie "DL Python" Scripting-Knoten . wennTensorFlow 2 ist die für die TensorFlow 2 Knoten ausgewählte Umgebung ausgewählt wird und TensorFlow 2 Python Bibliothek wird verfügbar sein, wenn Sie "DL Python" Scripting-Knoten .

Option 2: Verwenden Sie Python-Konfiguration

Wenn Sie diese Option wählen, wird KNIME Deep Learning die gleiche Python 3 Umgebung verwenden, wie in der Python Voreinstellungsseite konfiguriert. Dies geht davon aus, dass diese Python 3 Umgebung bereits enthält alle erforderlichen Python Deep Learning Pakete (die installiert werden müssen) manuell). Eine Liste der benötigten Python Deep Learning Pakete für die Keras Integration ist

KNIME Keras I

Abschnitt. Diese Option ist für

KNIME Installation von Keras Integration

Dieser Abschnitt beschreibt die Installation der KNIME Integration von Keras .

Ähnlich wie bei der KNIME Python Integration nutzt die KNIME Keras Integration eine bestehende Python Installation, die neben KNIME installiert ist und hängt von bestimmten Python Pakete, die installiert werden müssen.

Sie müssen:

ANHANG Installieren Sie die KNIME Keras Integration in die KNIME Analytics Platform

2. Python einrichten. Wir **empfehlen** Sie haben Python mit dem Python Deep Learning eingerichtet Vorzugsseite. So folgen Sie den Anweisungen in der

<a href="#page2" style="color: #ff6600; te

Tiefen lernen

Abschnitt.

Es ist jedoch auch möglich, die Python-Installation manuell einzurichten, indem man

Man

.

Installation des KNIME Integration von Keras

Die KNIME Keras Integration Erweiterung kann über die KNIME Analytics Plattform Seite aktualisieren In der KNIME Analytics Platform Datei → KNIME installieren Erweiterungen . Die Integration finden Sie unter KNIME Laborerweiterungen oder durch Eingabe von Keras in die Suche Box.

Wenn Keras nicht als verfügbares Deep Learning Back-End angezeigt wird, können Sie um die KNIME Analytics Platform neu zu starten.

Nachdem Sie die KNIME Keras Integration installiert und Python aus dem Python Deep

Py

Sie sind bereit zu gehen, und Sie **nicht** muss durch den nächsten Abschnitt gelesen werden.

Manuelle Python Installation

Nur folgen Sie den Anweisungen in diesem Abschnitt, wenn Sie Python für das KNIME einrichten möchten

Keras Integration manuell. Wir empfehlen stattdessen Python mit dem Python Deep einzurichten

Vorzugsseite lernen. Um dies zu tun, gehen Sie bitte in die Lernen

.

ANHANG KNIME führt Keras in einer lokalen Python-Installation durch, die manuell eingerichtet werden muss. wenn

Sie haben bereits Ihre Python Installation eingerichtet, können Sie diesen Schritt überspringen. Wir **stark empfehlen** einrichten Conda Umgebung wie in unserer Python Installation beschrieben Guide. In diesem Führer können Sie entweder folgen Quickstart Abschnitt , oder folgen Sie der Vollständig Montagebereich wenn Sie einen detaillierten Durchgang haben möchten.

Die KNIME Keras Integration unterstützt nur Python 3, d.h. es ist notwendig, Python 3.

2. Zusätzlich zu den im vorherigen Schritt installierten Python-Paketen, der KNIME Keras

Die Integration hängt von weiteren Paketen ab. Wenn Sie Anaconda verwenden, können sie installiert mit einem einzigen Befehl. Die gewünschten Pakete sind H5py = 2.8 , Tensorflow... mkl = 1,12 (Tensorflow = 1.12 für GPU und Keras = 2.2.4 (Keras-gpu = 2.2.4 für GPU). Für die CPU-Version den Befehl:

```
conda install --name h5py=2.8 tensorflow-mkl=1.12 keras=2.2.4
```

und für die GPU-Version der Befehl:

```
conda install -Name h5py=2.8 tensorflow=1.12 keras-gpu=2.2.4
```

wenn ist der Name Ihrer conda-Umgebung (z. py3_knime)

Eine allgemeine Beschreibung zur Installation weiterer Python-Pakete mit Anaconda can werden gefundenHier. .

Die Pakete Keras und Keras-gpu sind nur mit neuerer conda verfügbar Versionen (Minimum conda Version: 4.3.30). Wenn Sie eine ältere Version haben, können Sie AktualisierungConda mit dem Befehl conda update conda .

Falls Sie aufgrund fehlender Keras-Abhängigkeiten Probleme innerhalb von KNIME haben oder wollen doppelt überprüfen, dass alles richtig eingerichtet wurde, hier ist eine Liste von Keras Abhängigkeiten (die sollten automatisch installiert sein):

- h5py (Version: 2.8)
- numpy (Version: 1.15)
- pyyaml (Version: 3.13)
- scipy (Version: 1.1)
- sechs (Mindestversion: 1.11)
- Tensorflow oder Tensorflow-gpu (Version: 1.12)

Wenn Sie Anaconda verwenden, können Sie überprüfen, ob diese Abhängigkeiten von
Laufen:

```
conda list -n
```

wenn ist der Name Ihrer conda-Umgebung.

Weitere Erweiterungen

Für die KNIME Deep Learning Integration die Erweiterung KNIME Bildbearbeitung - Tief
Erweiterung ist verfügbar. Es unterstützt Bilder von KNIME Bildbearbeitung .

GPU-Unterstützung

Keras ist in der Lage, Deep Learning Modelle mit einer kompatiblen NVIDIA® GPU über
TensorFlow. Die meisten der erforderlichen Abhängigkeiten für GPU (z.B. CUDA® und cuDNN) werden sein
automatisch installiert von Anaconda bei der Installation Conda Pakete Tensorflow = 1.12
und Keras-gpu = 2.2.4 . Die einzige zusätzliche Anforderung, die manuell installiert werden muss, ist
die neueste Version der NVIDIA® GPU Treiber .

Tensorflow 1.1.2. benötigt eine NVIDIA GPU-Karte mit CUDA Compute Capability 3.5 oder höher.
Siehe die Liste CUDA-fähige GPU-Karten zu überprüfen, ob Ihre GPU-Karte den Anforderungen entspricht.

KNIME TensorFlow 1 Integration Installation

Dieser Abschnitt beschreibt die Installation der KNIME TensorFlow 1 Integration für den Einsatz mit KNIME
Analyseplattform. Mit der KNIME TensorFlow 1 Integration kann man lesen und
TensorFlow 1 ausführen Gespeicherte Modellemit der TensorFlow 1 Java API, die unabhängig ist
von Python.

Installation

Die KNIME TensorFlow 1 Integration kann über die KNIME Analytics Plattform Seite aktualisieren In der KNIME Analytics Platform Datei → KNIME installieren Erweiterungen . Die Integration finden Sie unter KNIME Laborerweiterungen oder durch Eingabe von TensorFlow 1 in die Suchfeld.

Wenn TensorFlow 1 nicht als verfügbares Deep Learning Back-End angezeigt wird, werden Sie muss die KNIME Analytics Platform neu starten.

Erweiterte

TensorFlow 1 Gespeicherte Modelle können auch über Python ausgeführt werden. Wenn Sie TensorFlow verwenden möchten 1 Modelle in DL Python-Knoten mit benutzerdefinierten Python-Skripten, Sie benötigen eine Python-Installation neben KNIME. Diese Python-Installation hat die gleichen Anforderungen wie die KNIME Keras Integration. Um Python und die notwendigen Abhängigkeiten zu installieren, lesen Sie bitte die Python Installation Abschnitt .

GPU Unterstützung

Dieser Abschnitt beschreibt, wie die GPU-Unterstützung für die KNIME TensorFlow 1 Integration aufgebaut werden kann.

GPU-Unterstützung für die KNIME TensorFlow 1 Integration (die die TensorFlow 1 Java API) ist in der Regel unabhängig von der GPU-Unterstützung der KNIME Keras Integration (die Python verwendet). Daher müssen sie eingerichtet werden individuell.

Aufgrund der Einschränkungen von TensorFlow unterstützt die GPU die KNIME TensorFlow 1 Integration kann nicht auf Mac verwendet werden. Nur Linux und Windows werden unterstützt.

Die KNIME TensorFlow 1 Integration verwendet TensorFlow-Version 1.3.1 , die folgende NVIDIA® Software, die auf Ihrem System installiert werden soll:

- NVIDIA® GPU Treiber - CUDA® 10,0 erfordert 410.x oder höher.
- CUDA® Toolkit - TensorFlow (≥ 1.13.0) unterstützt CUDA® 10,0 . Detaillierte Installation Anleitungen finden Sie Hier. .

Stellen Sie sicher, CUDA zu installieren 10,0 . CUDA 10.1 wird **nicht** Arbeit.

- CunNN - (Version) ≥ ANHANG) - wählen Sie cuDNN V7.6.0 (Mai 20, 2019), für CUDA® 10,0 .
  Detaillierte Installationsanweisungen finden Sie Hier. .

KNIME TensorFlow 2 Integration Installation

Dieser Abschnitt beschreibt die Installation der KNIME TensorFlow 2 Integration für den Einsatz mit KNIME Analyseplattform. Mit der KNIME TensorFlow 2 Integration kann man lesen, modifizieren, Ausführung und Zug TensorFlow 2 Keras Modelle mit der TensorFlow 2 Python API.

Ähnlich wie bei der KNIME Python Integration nutzt die KNIME TensorFlow 2 Integration intern eine bestehende Python-Installation, die neben KNIME installiert ist und von bestimmten abhängig ist Python-Pakete, die installiert werden müssen. Bitte folgen Sie den Anweisungen in der Python für KNIME Tiefen lernen Abschnitt zum Setup Python mit TensorFlow 2 verwendet werden.

Installation

Die KNIME TensorFlow 2 Integration kann über die KNIME Analytics Plattform Seite aktualisieren In der KNIME Analytics Platform Datei → KNIME installieren Erweiterungen . Die Integration finden Sie unter KNIME Laborerweiterungen oder durch Eingabe von TensorFlow 2 in die Suchfeld.

Wenn TensorFlow 2 nicht als verfügbares Deep Learning Back-End angezeigt wird, werden Sie muss die KNIME Analytics Platform neu starten.

GPU Unterstützung

TensorFlow 2 kann Deep Learning Modelle mit einer kompatiblen NVIDIA® GPU beschleunigen. Die der erforderlichen Abhängigkeiten für GPU (z.B. CUDA® und cuDNN) werden automatisch installiert von Anaconda bei der Schaffung der Python-Umgebung Python Deep Learning Vorzugsseite. Die einzige zusätzliche Anforderung, die manuell installiert werden muss, ist die neueste Version der NVIDIA® GPU Treiber .

Tensorflow 2 benötigt eine NVIDIA GPU-Karte mit CUDA Compute Capability 3.5 oder höher. Vgl. die Liste der CUDA-fähige GPU-Karten zu überprüfen, ob Ihre GPU-Karte den Anforderungen entspricht.

KNIME ONNX Integration Installation

Dieser Abschnitt erklärt, wie die KNIME ONNX Integration mit KNIME installiert werden kann

Analyseplattform.

Installation

Der KNIME ONNX Integration kann über die KNIME Analytics Platform Update Seite . In der KNIME Analytics Platform Datei → KNIME installieren Erweiterungen . Die Integration kann finden unter KNIME Laborerweiterungen oder durch Eingabe von ONNX in das Suchfeld.

ONNX Abhängigkeiten

Wie die KNIME Keras Integration läuft die KNIME ONNX Integration mit dem KNIME Python Integration und hängt von zusätzlichen Python Paketen ab. Wir **empfehlen** um Python einzurichten mit der Python Deep Learning Präferenz Seite. Dies wird Python für alle KNIME Deep einrichten Integrationen auf einmal lernen, einschließlich aller ONNX Abhängigkeiten. Um dies zu tun, gehen Sie bitte in die <a href="#page2" style="color: #ff6600; text-decoration: underline;">Python für KNIME einrichten Tiefe Abschnitt.

Wenn Sie Python manuell einrichten möchten, hängt die KNIME ONNX Integration von folgenden Pip Python Pakete:

- *===1.4.1
- Aufnx-tf==1.2.1

> Die oben aufgeführten Pakete sind nicht über Conda . Sie können jedoch leicht in ein bestehendes eingebaut werden Conda Umwelt Pip . Um das zu tun, nur aktivieren Conda Umgebung, die Sie die Pakete hinzufügen und ausführen möchten eine pip install Befehl, z. pip install onnx==1.4.1 .

KNIME Deeplearning4j Installation

Dieser Abschnitt erklärt die Installation von KNIME Deeplearning4j Integration mit KNIME Analyseplattform.

Installation

Die KNIME Deeplearning4j Integration kann über die KNIME Analytics Plattform Seite aktualisieren In der KNIME Analytics Platform Datei → KNIME installieren Erweiterungen . Die Integration finden Sie unter KNIME Laborerweiterungen oder durch Eingabe von Deeplearning4j in die Suchfeld.

GPU Unterstützung

Die KNIME Deeplearning4j Integration unterstützt die Beschleunigung von Deep Learning-Modellen mit einem Kompatible NVIDIA® GPU. Für die GPU-Unterstützung, CUDA® Toolkit 8.0 muss auf Ihrem System. Detaillierte Installationsanweisungen finden Sie Hier. .

Bekannte Fragen

Ältere Versionen der KNIME Analytics Platform, die zusammen mit älteren Versionen des KNIME verwendet werden Deep Learning Integrations Abhängigkeiten, kann zu Fehlern führen. Diese Fehler können durch Aktualisierung der KNIME Analytics Platform und der KNIME Deep Learning Integrations auf die neuesten Versionen. Darüber hinaus, wenn Sie Python verwenden, stellen Sie bitte sicher, dass die empfohlenen Conda Paketversionen . Folgende Themen sind derzeit bekannt:

- Node fehlschlägt mit Fehler **AttributError: '[..]' Objekt hat kein Attribut „inbound_nodes '** an der Unterseite eines Python-Trackbacks im KNIME-Log (KNIME Nur 3.5.x).

  Keras Version 2.1.3 führte zu Bruchänderungen, die in der KNIME-Version angepasst wurden 3.6.0. Bitte aktualisieren Sie KNIME auf Version 3.6.0 oder Downgrade Keras auf Version 2.1.2 oder unten (Mindestversion: 2.0.7).

- Node fehlschlägt mit Fehler **UnicodeEncode Fehler: 'ascii' codec can „encode Zeichen [..] in Position [.]: Ordinal nicht im Bereich(128)** am Boden eines Python-Trackback im KNIME-Log.

  Dieser Fehler kann bei der Verwendung von Keras Version 2.1.2 auftreten, um ein Keras-Netzwerk zu laden, das war mit einer älteren Keras-Version gespeichert. Achten Sie darauf, Keras 2.1.2 in solchen Fällen nicht zu verwenden.

- Node scheitert bei beiden Fehlern **SystemError: unbekannt opcode** und eine Warnung **XXX lineno: [..], opcode: [..]** im KNIME-Log.

  Dies ist ein Python-verwandter Fehler, der beim Laden eines Keras-Netzwerks mit einem Lambda-Expression (z.B. innerhalb einer Lambda-Schicht), die mit einer anderen Python-Version. Stellen Sie sicher, dass Sie die gleiche Python-Version zum Speichern und Laden der gleiches Netzwerk.

- DL Keras Network Learner scheitert mit Fehler **AttributError: 'int' Objekt hat keine Attribut 'dtype' '** an der Unterseite eines Python-Trackbacks im KNIME-Log, wenn die Clip-Norm Option wird im Knoten-Dialog aktiviert und Keras (TensorFlow) ist das gewählte hintere Ende.

Dies ist ein TensorFlow-verwandter Fehler, der nur in sehr bestimmten Situationen auftritt. Versuchen Sie zu verwenden eine andere Keras zurück Ende, um um dieses Problem zu arbeiten.

- DL Python Network Executor scripting node gibt falsche Zahlenwerte aus, wenn mit Flatbuffers Serialisierungsbibliothek (KNIME 3.6.1 und älter).

  Flatbuffers in KNIME unterstützt momentan keine Daten von float32. Wir empfehlen Verwendung von Apache Arrow als Serialisierungsbibliothek. Diese Option kann in KNIME via Datei → Vorlieben → KNIME → Python → Serialisierungsbibliothek .

- TensorFlow 1 und 2 unter Windows: DL Python Scripting Nodes scheitert beim Speichern das Modell mit dem Fehler "Das System kann den angegebenen Pfad nicht finden."

  Dieser Fehler kann durch die Pfadlängengrenze von 260 Zeichen in Windows angezeigt werden. A Anleitung zur Deaktivierung dieser Grenze für fortgeschrittene Benutzer finden Hier. . Der Verwandte TensorFlow Problem ist zu finden Hier. .

- TensorFlow 2: Kann kein Modell aus dem TensorFlow Hub laden.

  TensorFlow Hub-Modelle enthalten nicht immer alle notwendigen Informationen, um sie in KNIME. Der Beispiel-Workflow "02 Use a TFHub Model" zeigt wie ein Tensor Strom Hubmodell kann mit einer einfachen Komponente verwendet werden.

- Installationstest für Python hinten ... Ich habe gewarnt. Bitte achten Sie darauf Die Python-Umgebung ist richtig eingerichtet und erwägt die Erhöhung der Timeout (derzeit 25000 ms) mit der VM-Option '-Dknime.dl.Installationstesttimeout=<neuer Timeout in ms>'.

  Bevor Sie die Python-Umgebung nutzen, überprüfen die Deep Learning-Knoten, ob alle benötigten Python Pakete können importiert werden. Wenn der Import nach dem Standard nicht beendet ist Zeit von 25 Sekunden gilt dies als ein Ausfall. Bitte stellen Sie sicher, dass Sie die Umgebung richtig eingerichtet wird, indem man Fehler auf dem "Python Deep Learning" überprüft

  <a href="#page2" style="color:#ff6600;text-decoration:none;">

) Auf einigen Systemen Import der benötigten Python Pakete (insbesondere Tensorflow) kann länger dauern als 25s. In diesem Fall kann durch die Einstellung der VM-Option -Dknime.dl.Installationstesttimeout= . Tun Sie dies, indem Sie die Linie zum Boden der knime.ini Datei.

- Netzwerke mit einer LSTM-Schicht arbeiten nicht an macOS

  Dieser Fehler wird wahrscheinlich durch Probleme mit der MKL-Bibliothek verursacht. Sie können den nomkl installieren Paket zu Ihnen conda Umgebung über conda install nomkl um MKL zu deinstallieren. Für mehr Details siehe https://docs.anaconda.com/mkl-optimizations/index.html#uninstalling-mkl .

KNIME AG
Talacker 50
8001 Zürich, Schweiz
www.knime.com
Info@knime.com