Harran Üniversitesi Yüksek Başarımlı Hesaplama Merkezi

TENSORFLOW & KERAS KULLANIM REHBERİ

Arş. Gör. Mehmet Umut SALUR
HPC BİLGİSAYAR ARAŞTIRMA GRUBU | HARRAN ÜNİVERSİTESİ

İçindekiler

1. Proje Başvurusu ----------2

2. Kurulum ve Çalışma Ortamının Hazırlanması ----------2

3. TensorFlow & Keras Kurulumu ----------4

3.1. Bir düğüm üzerinde bulunan GPU kartlarını kullanarak model eğitimi gerçekleştirmek ----------4

3.2. Bir düğüm üzerinde bulunan birden fazla GPU kartlarını birlikte kullanarak model eğitimi gerçekleştirmek ----------5

3.3. Birden fazla düğüm üzerindeki birden fazla GPU kartlarını kullanarak modelin eğitiminin gerçekleştirilmesi ----------6

3.4. HPC üzerinde bash script dosyası çalıştırma ----------7

1. Proje Başvurusu

HPC merkezi laboratuvarını kullanım için başvuru yapacak araştırmacıların projesi ile başvuru formunu doldurması ve başvuruda bulunması gerekmektedir.

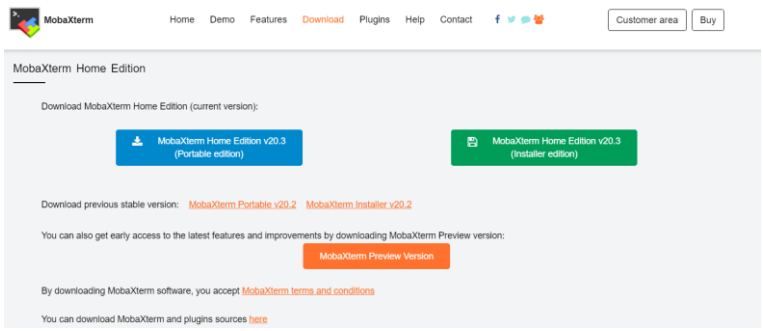
Akademik Hazırlık Programı, üniversiteler bünyesinde hali hazırda çalışan tüm Öğretim Elemanlarına, Lisans, Yüksek Lisans ve Doktora öğrencilerine araştırma projeleri başvurularına teşvik etmek amacı ile sağlanacak olan ücretsiz kaynak desteğini kapsamaktadır. Öğretim elemanları, Lisans, Yüksek Lisans, Doktora öğrencilerine ücretsiz 2.000 çekirdek-saat, 20 GB alan verilmektedir.

- http://hpc.harran.edu.tr/?page_id=326 adresinden proje başvurusu yapılmalıdır.
- HPC tarafından VPN bağlantı ayar dosyası, Kullanıcı Adı, Şifre, SSH bağlantı IP adresi bilgisi tarafınıza e-posta yoluyla iletilir.

2. Kurulum ve Çalışma Ortamının Hazırlanması

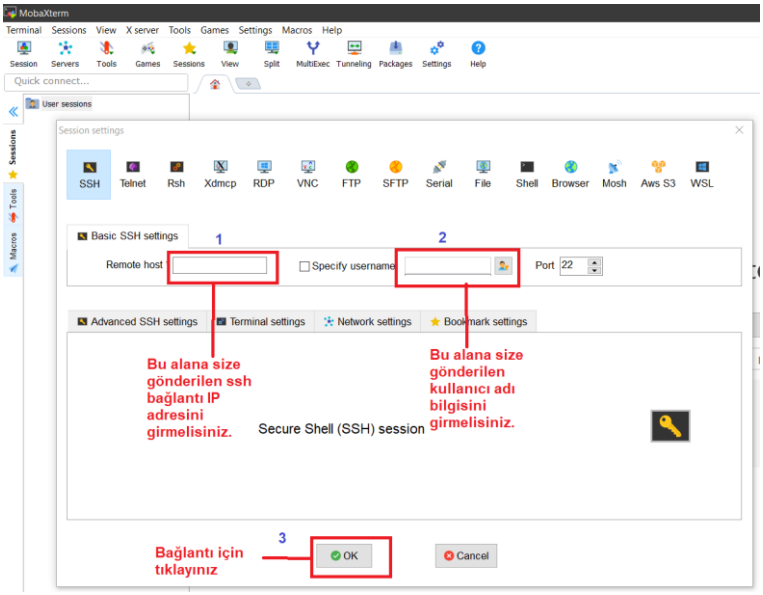
HPC hesap bilgileri kullanıcılara iletildikten sonra, kullanıcılar HPC kümesiyle bağlantı kurup, uygulama programlarını kullanabilirler. Bunun için HPC uzak bağlantısı ve çalışma ortamının hazırlanması ve çalışılacak uygulama programlarının yüklenmesi gerekmektedir. Windows ortamından HPC'ye bağlantı gerçekleştirmek için aşağıdaki adımlar takip edilebilir.

- Uzak sunucuya VPN üzerinden bağlantı gerçekleştirmek için Open VPN yazılımı kullanılabilir. Bunun için bu uygulamayı aşağıdaki adresten indiriniz ve kurunuz.
  Link: https://www.techspot.com/downloads/5182-openvpn.html
- Open VPN kurulumu gerçekleştirildikten sonra, e-posta ile size iletilen vpn yapılandırma dosyasını uygulamaya yükleyerek HPC ağına giriş yapabilirsiniz.
- HPC ağına bağlandıktan sonra, Windows ortamından HPC sistemindeki lunix işletim sistemi ile haberleşmek için SSH terminal uygulamasına ihtiyaç olmaktadır. Bunun için bu rehberde MobaXterm yazılımı üzerinden bağlantı gerçekleştirilecektir.
- MobaXterm yazılımını aşağıdaki adresten indirip, bilgisayarınıza kurmanız gerekmektedir.
  Link: https://mobaxterm.mobatek.net/download.html



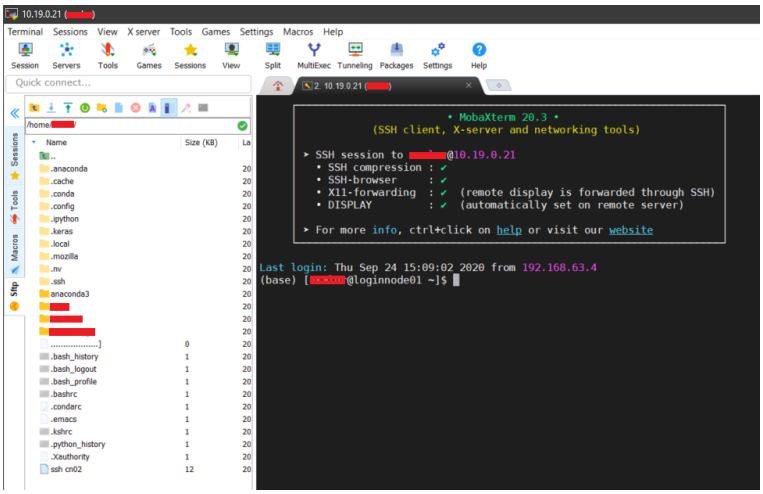
Şekil 1. MobaXterm indirme sayfası.

- MobaXtrem ile bağlantı sağlamak için gerekli bilgilerin nasıl doldurulmasıyla ilgili görsel Şekil 2'de verilmiştir.



Şekil 2. MobaXtrem kullanım arayüzü.

- MobaXtrem SSH bağlantısı gerçekleştirmek için MobaXtrem yazılımın son üst menüsünde "Sessions" butonuna tıklanır.
- Açılan pencerede Remote host ve Specify username bilgisi girilir ve "OK" butonuna tıklanarak bağlantı kurulur.
- Açılan terminal ekranında HPC tarafından iletilen kullanıcı giriş bilgilerindeki şifre bilgisi girilmelidir. Bu şifre MobaXterm tarafından ilk bağlantıda sorulmaktadır. Daha sonradan tekrar aynı kullanıcı olarak bağlantı yapmak istediğinizde şifre girmeniz gerekmeyecektir.



Şekil 3. MobaXterm SSH bağlantı ilk ekranı.

- Giriş yapıldıktan sonra terminalde "kullanıcıAdı@loginnode01" giriş düğümde olduğunuzu görürsünüz. Şekil 3'te MobaXterm il bağlantı ekranı görülmektedir. Burada sol tarafta size ait olan depolama alanını "home" dizini görebilirsiniz. Algoritmalar tarafından çalıştırılması istenen veriler, "upload" butonunu kullanılarak veya sürükle bırak yöntemiyle buraya eklenebilir.
- HPC üzerinde bulunan yüklü yazılımları listelemek ve kullanmak için aşağıdaki komut satırları kullanılabilir.

- ✓ **"`module load shared`"**: Paylaşılan modülleri kullanıcı alanına yükler.
- ✓ **"`module load spack`"**: Paylaşılan modülleri kullanıcı alanına yükler.
- ✓ "`module avail`": Modülleri listeler.
- ✓ **"`module load moduladiyaziniz`"**: Moduladi alanına yazılan modülün kullanımını sağlar.

3. TensorFlow & Keras Kurulumu

Burada ihtiyacınızı karşılayacak yazılım bulunmuyorsa eğer, kendi kullanıcı alanınıza istediğiniz açık kaynak kodlu yazılımı yükleyebilirsiniz. Örneğin biz TensorFlow ve Keras derin öğrenme yazılımlarını kullanmak için gerekli kurulumları gerçekleştireceğiz. TensorFlow ve Keras kurulumlarını Anaconda Python dağırımı üzerinden gerçekleştireceğiz. Kurulum için aşağıdaki adımlar takip edilmelidir.

- HPC kümesine kurulması için Anaconda'nın Linux sürümünün indirilmesi gerekmektedir. Bunun için anaconda.com/downloads adresinden Linux indirme link bilgisi kopyalamalıdır.
- Kurulması istenen bash (.sh file) kurulum linki kopyalanmalıdır.
- **"`wget`"** kullanılarak bash kurulum dosyası indirilmelidir. Bunun için "temp" isminde bir dizin oluşturulup, terminalden ilgili dizin açıldıktan sonra kurulum linki (https://repo.continuum.io/archive/Anaconda3<release>.sh) terminale yapıştırılır.
- Son olarak terminale **"`bash Anaconda3-5.2.0-Linux-x86_64.sh`"** komutu yazılarak Anaconda kurulumu gerçekleştirilir.
- Anaconda kurulumu yapıldıktan sonra terminale **"`conda install -c anaconda keras-gpu`"** komu yazılarak, Keras ve Tensorflow'un GPU sürümünün kurulumu gerçekleştirilmiş olunacaktır.

Terminalden **"`ssh gpu01`"** yazarak **"`loginnode01`"**den gpu01'e geçebilirsiniz. GPU node'dayken terminalden "python /home/kullaniciadi/dizinadi/sorucecode.py" komutuyla "KullanıcıAdı" altında "DizinAdi" altındaki "sorucecode.py" Python dosyasını çalıştırabilirsiniz. HPC düğümleri üzerinde arayüz olarak Anaconda veya Spyder IDE'sini çalıştırmanız mümkündür. Bunun için aşağıdaki işlemlerin takip edilmesi gerekmektedir.

- Anaconda'yı çalıştırmak için terminale **"`source ~/anaconda3/bin/activate root`"** komutu yazılarak çalıştırılmalıdır.
- Anaconda GUI'sinin açılması içinde **"`anaconda-navigator`"** komu çalıştırılmalıdır. Anaconda GUI'si ekranda açıldıktan sonra Spyder ve diğer uygulamalar çalıştırılabilir.

3.1. Bir düğüm üzerinde bulunan GPU kartlarını kullanarak model eğitimi gerçekleştirmek

GPU düğümlerinin biri üzerindeyken **"`python home/ali/uygulamalar/film_yorumlari_siniflandirma.py`"** komutuyla aşağıda içerdiği kodları verilen film_yorumlari_siniflandirma.py dosyasını çalıştırabilirsiniz. Şekil 4'te film yorumları sınıflandırma örneğinin kaynak kodları verilmiştir. Şekil 5'te ise modelin eğitimi sırasındaki görsel mevcuttur.

```
import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

max_features = 20000  # Film yorumlarından 2000 adetini sınıflandıracağız
maxlen = 200  # Her yorumun ilk 200 karakterini kullanılacaktır

inputs = keras.Input(shape=(None,), dtype="int32")
x = layers.Embedding(max_features, 128)(inputs)
x = layers.Bidirectional(layers.LSTM(64, return_sequences=True))(x)
x = layers.Bidirectional(layers.LSTM(64))(x)
outputs = layers.Dense(1, activation="sigmoid")(x)
model = keras.Model(inputs, outputs)
model.summary()
(x_train, y_train), (x_val, y_val) = keras.datasets.imdb.load_data(num_words=max_features )
print(len(x_train), "Training sequences")
print(len(x_val), "Validation sequences")
x_train = keras.preprocessing.sequence.pad_sequences(x_train, maxlen=maxlen)
x_val = keras.preprocessing.sequence.pad_sequences(x_val, maxlen=maxlen)
model.compile("adam", "binary_crossentropy", metrics=["accuracy"])
model.fit(x_train, y_train, batch_size=32, epochs=5, validation_data=(x_val, y_val))
```

Şekil 4. Film yorumları sınıflandırma kaynak kodları.

```
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, None)]            0
_________________________________________________________________
embedding (Embedding)        (None, None, 128)         2560000
_________________________________________________________________
bidirectional (Bidirectional (None, None, 128)         98816
_________________________________________________________________
bidirectional_1 (Bidirection (None, 128)               98816
_________________________________________________________________
dense (Dense)                (None, 1)                 129
=================================================================
Total params: 2,757,761
Trainable params: 2,757,761
Non-trainable params: 0
_________________________________________________________________
25000 Training sequences
25000 Validation sequences
Epoch 1/5
2020-09-24 14:40:20.517371: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dyna
mic library libcublas.so.10
2020-09-24 14:40:20.784205: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dyna
mic library libcudnn.so.7
782/782 [==============================] - 148s 189ms/step - loss: 0.3843 - accuracy: 0.8277 - val_loss: 0.3819 - va
l_accuracy: 0.8522
Epoch 2/5
782/782 [==============================] - 146s 187ms/step - loss: 0.2048 - accuracy: 0.9223 - val_loss: 0.3333 - va
l_accuracy: 0.8644
Epoch 3/5
782/782 [==============================] - ETA: 0s - loss: 0.1209 - accuracy: 0.9569
```

Şekil 5. Model eğitim aşaması görseli.

3.2. Bir düğüm üzerinde bulunan birden fazla GPU kartlarını birlikte kullanarak model eğitimi gerçekleştirmek

Bir düğüm üzerinde bulunan tüm GPU kartlarını kullanarak modellerin eğitimini gerçekleştirmek mümkündür. Bunun için Tensorflow'un sunduğu dağıtık hesaplama stratejisini kullanabilirsiniz. Uygulama programına "`strategy = tf.distribute.MirroredStrategy()`" satırını eklemeniz

durumunda modelinizin hesaplamaları dağıtık GPU'lar üzerinden gerçekleştirilecektir. Örnek uygulama kodları Şekil 6'da verilmiştir. Daha fazla bilgi için Keras[1] dokümantasyon sayfasını inceleyebilirsiniz.

```
import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
#strategy = tf.distribute.MirroredStrategy()
#print('Number of devices: {}'.format(strategy.num_replicas_in_sync))

max_features = 20000  # Only consider the top 20k words
maxlen = 200  # Only consider the first 200 words of each movie review
# Input for variable-length sequences of integers
with strategy.scope():
  inputs = keras.Input(shape=(None,), dtype="int32")
  # Embed each integer in a 128-dimensional vector
  x = layers.Embedding(max_features, 128)(inputs)
  # Add 2 bidirectional LSTMs
  x = layers.Bidirectional(layers.LSTM(64, return_sequences=True))(x)
  x = layers.Bidirectional(layers.LSTM(64))(x)
  # Add a classifier
  outputs = layers.Dense(1, activation="sigmoid")(x)
  model = keras.Model(inputs, outputs)
  model.summary()

  (x_train, y_train), (x_val, y_val) = keras.datasets.imdb.load_data(
    num_words=max_features )
  print(len(x_train), "Training sequences")
  print(len(x_val), "Validation sequences")
  x_train = keras.preprocessing.sequence.pad_sequences(x_train, maxlen=maxlen)
  x_val = keras.preprocessing.sequence.pad_sequences(x_val, maxlen=maxlen)
  model.compile("adam", "binary_crossentropy", metrics=["accuracy"])
  model.fit(x_train, y_train, batch_size=32, epochs=2, validation_data=(x_val, y_val))
```

Şekil 6. Bir düğüm üzerindeki tüm GPU'ları kullanarak film yorumlarını sınıflandırma

3.3. Birden fazla düğüm üzerindeki birden fazla GPU kartlarını kullanarak modelin eğitiminin gerçekleştirilmesi

Harran Üniversitesi HPC kümesinde dört adet GPU kart bulunduran düğüm bulunmaktadır. Modelinizi kümedeki birden fazla düğümde bulunan GPU kartları üzerinde eğitmek isterseniz, Slurm iş yönetim yazılım çatısını kullanmanız gerekmektedir. Bunun için aşağıdaki işlem adımlarını takip edebilirseniz. HPC'ye giriş yaptıktan sonra;

- "`modüle load slurm`" komutu ile slurm iş yönetim yazılım çatısını kendi oturumunuza yükleyiniz.
- Slurm'ın srun komutu ile çalıştırmak istediğimiz python dosyasını, kaç düğümdeki GPU kartlarını kullanmak istediğimizi belirterek çalıştırabilirsiniz. Örneğin; "`srun -N 2 -p gpu python dosyamiz.py`"
    - Srun Slrum komutudur.
    - "-N 2"; İki adet düğümdeki GPU kartlarını kullanmak istediğimiz belirtiyoruz. Buradaki 2 parametresi yerine 3 veya 4 yazılabilir.
    - "-p gpu"; gönderilen işin GPU kuyruğuna eklenmesini ifade ediyor.

[1] https://keras.io/guides/distributed_training/

3.4. HPC üzerinde bash script dosyası çalıştırma

Geliştirilen uygulama kodları HPC üzerinde bash script dosyası olarak çalıştırılabilir. Çalıştırılacak uygulamayı doğrudan terminalden çalıştırmak yerine bash script olarak yazıp, oturumu kapattıktan sonra da HPC üzerinde çalışmasını gerçekleştirmek mümkündür. Bunun için ilk olarak yazılan uygulamaya dair bir bash script (.sh dosyası (shell executables files)) oluşturulmalıdır. Bu bash dosyasının içine de terminal ekranındaki komut olduğu gibi yazılabilir. Şekil 7'de örnek bir .sh dosyasının içeriği görülmektedir. Bunun için de Slurm'ın `sbatch` komutunu kullanmak gerekmektedir. Şekil 8'de de örnek bir .sh dosyasını çalıştırma görseli bulunmaktadır. Bu şekilde geliştirilen uygulama terminalden çalıştırılmak yerine .sh dosyası olarak çalıştırmak mümkündür.

```
#!/bin/bash
srun -N 4 -p gpu python /home/username/Apps/uygulamamiz.py
```

Şekil 7. Örnek bir bash dosyasının içeriği.

```
(base) [      @gpu01 ~]$ sbatch /home/      /Apps/          .sh
Submitted batch job 871
```

Şekil 8. Bash dosyası çalıştırma.

`sbatch` komutunu kullanılarak HPC'ye gönderilen her iş için, Slurm tarafından bir iş ID'si atanmaktadır. Bu ID ile işin durumu kontrol edilebilir. Bunun için de Slrum'un `squeue` komutu kullanılmalıdır. Şekil 9'da gönderilen işin durumu görülmektedir.

```
(base) [      @gpu01 ~]$ squeue
             JOBID  PARTITION      USER  ST        TIME   NODES NODELIST(REASON)
               871       defq            R        5:39       1 cn01
(base) [      @gpu01 ~]$
```

Şekil 9. İş durumunu sorgulama.