

Project Name: DL Pytorch GPU House-Price-Prediction-Dataset

Table of Contents

Demo

Overview

Motivation

Technical Aspect

Installation

Run/How to Use/Steps

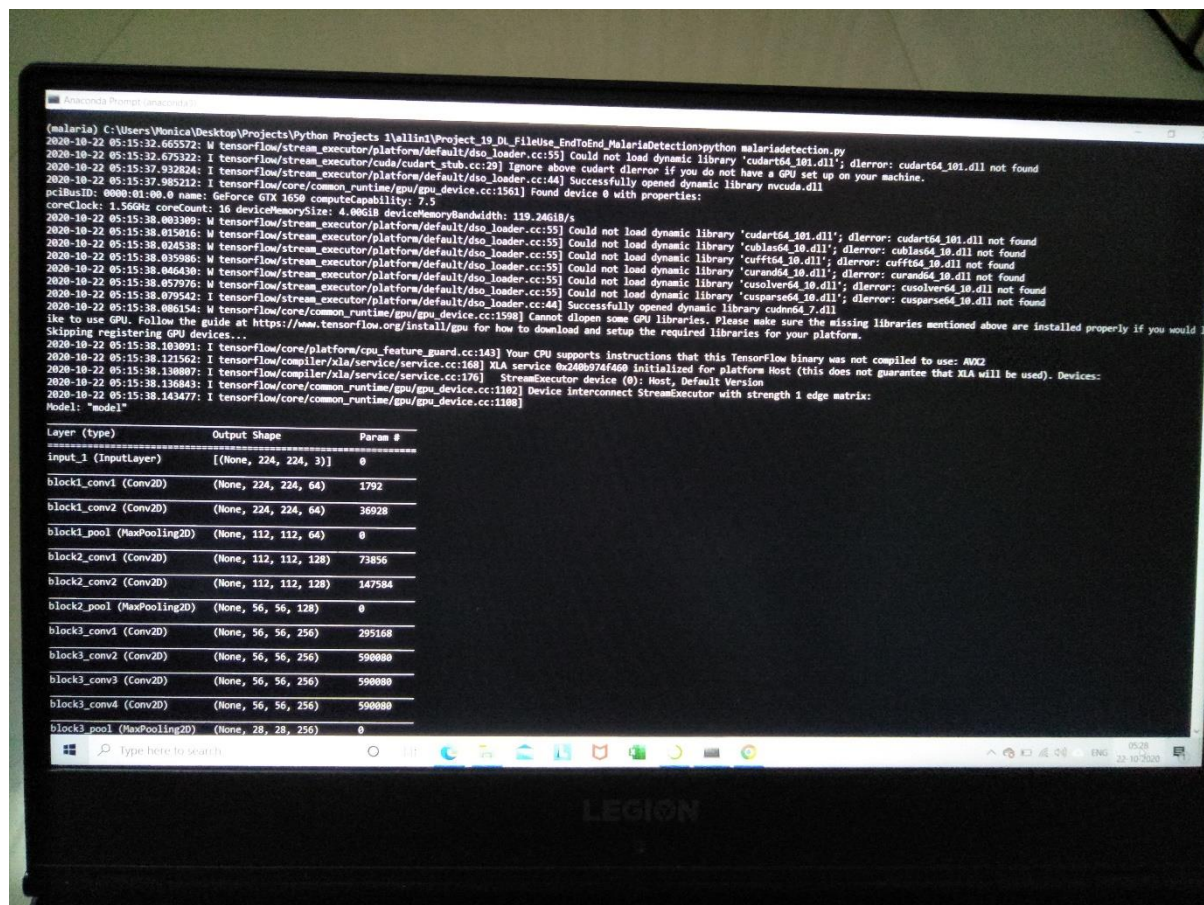
Directory Tree/Structure of Project

To Do/Future Scope

Technologies Used/System Requirements/Tech Stack

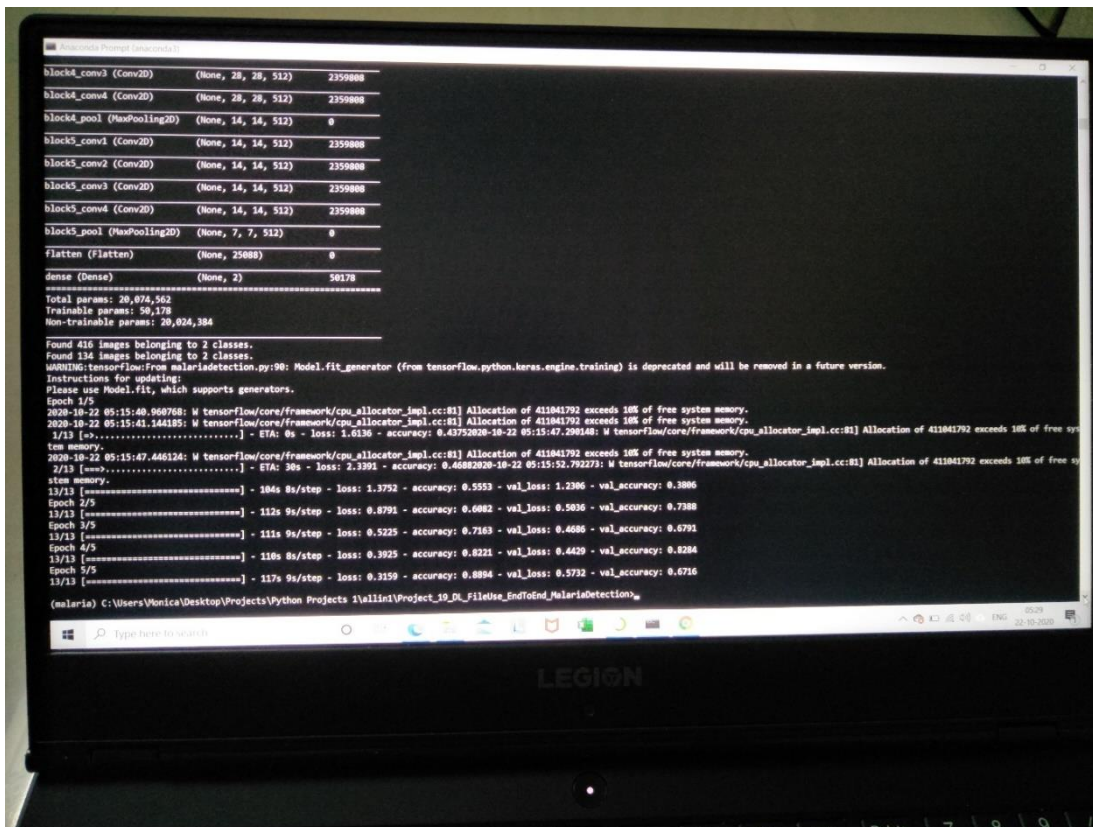
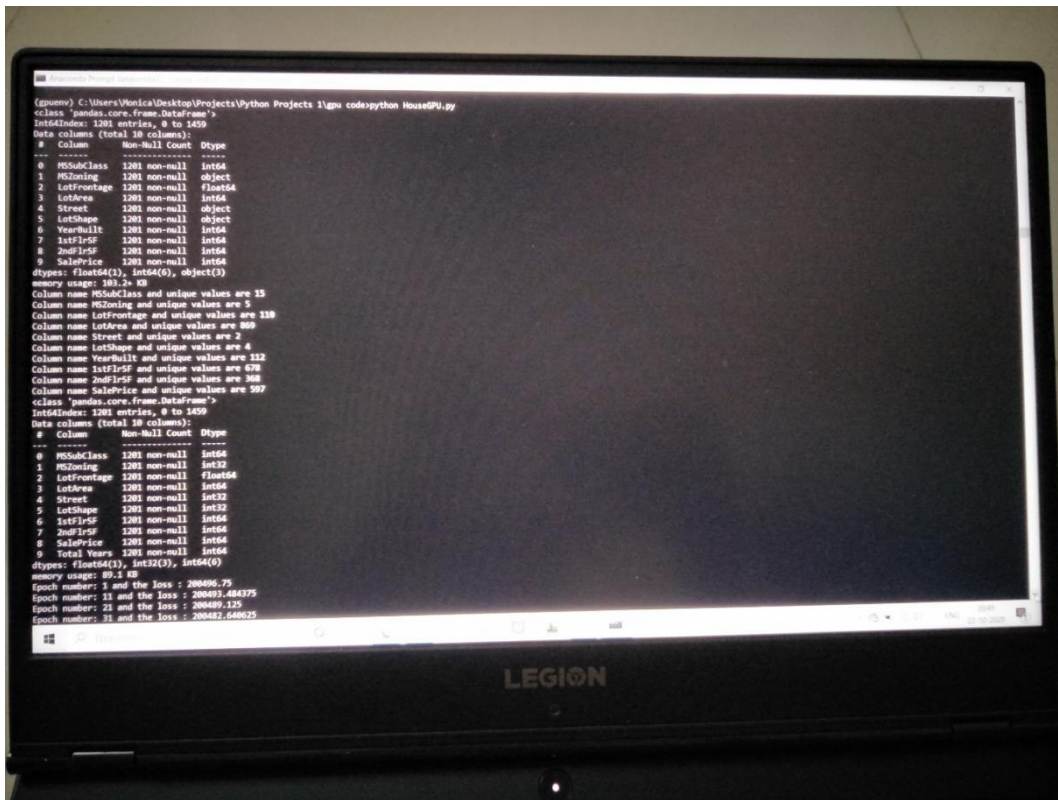
Credits

Demo



```
(malaria) C:\Users\Nurica\Desktop\Projects\Python Projects\1\allin1\Project_19_DL_FileUse_EndToEnd_MalariaDetection\python malariaDetection.py
2020-10-22 05:15:32.665372: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'cudart64_101.dll'; dlerror: cudart64_101.dll not found
2020-10-22 05:15:32.675322: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if you do not have a GPU set up on your machine.
2020-10-22 05:15:37.932824: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library nvcuda.dll
2020-10-22 05:15:37.985212: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1561] Found device 0 with properties:
pciBusID: 0000:01:00.0 Name: GeForce GTX 1650 computeCapability: 7.5
coreClock: 1.56GHz coreCount: 16 deviceMemorySize: 4.00GiB deviceMemoryBandwidth: 119.24GiB/s
2020-10-22 05:15:38.003389: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'cudart64_101.dll'; dlerror: cudart64_101.dll not found
2020-10-22 05:15:38.015016: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'cublas64_10.dll'; dlerror: cublas64_10.dll not found
2020-10-22 05:15:38.024538: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'cufft64_10.dll'; dlerror: cufft64_10.dll not found
2020-10-22 05:15:38.035986: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'curand64_10.dll'; dlerror: curand64_10.dll not found
2020-10-22 05:15:38.046430: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'cusolver64_10.dll'; dlerror: cusolver64_10.dll not found
2020-10-22 05:15:38.057976: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'cusparse64_10.dll'; dlerror: cusparse64_10.dll not found
2020-10-22 05:15:38.079542: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library cudnn64_7.dll
2020-10-22 05:15:38.086154: W tensorflow/core/common_runtime/gpu/gpu_device.cc:1598] Cannot dlopen some GPU libraries. Please make sure the missing libraries mentioned above are installed properly if you would
like to use GPU. Follow the guide at https://www.tensorflow.org/install/gpu for how to download and setup the required libraries for your platform.
Skipping registering GPU devices...
2020-10-22 05:15:38.103091: I tensorflow/core/platform/cpu_feature_guard.cc:143] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2
2020-10-22 05:15:38.121562: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x2408070f460 initialized for platform Host (this does not guarantee that XLA will be used). Devices:
2020-10-22 05:15:38.130807: I tensorflow/compiler/xla/service/service.cc:176] StreamExecutor device (0): Host, Default Version
2020-10-22 05:15:38.136843: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1182] Device interconnect StreamExecutor with strength 1 edge matrix:
2020-10-22 05:15:38.143477: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1188]
Model: "model"
```

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 224, 224, 3)]	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv4 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0

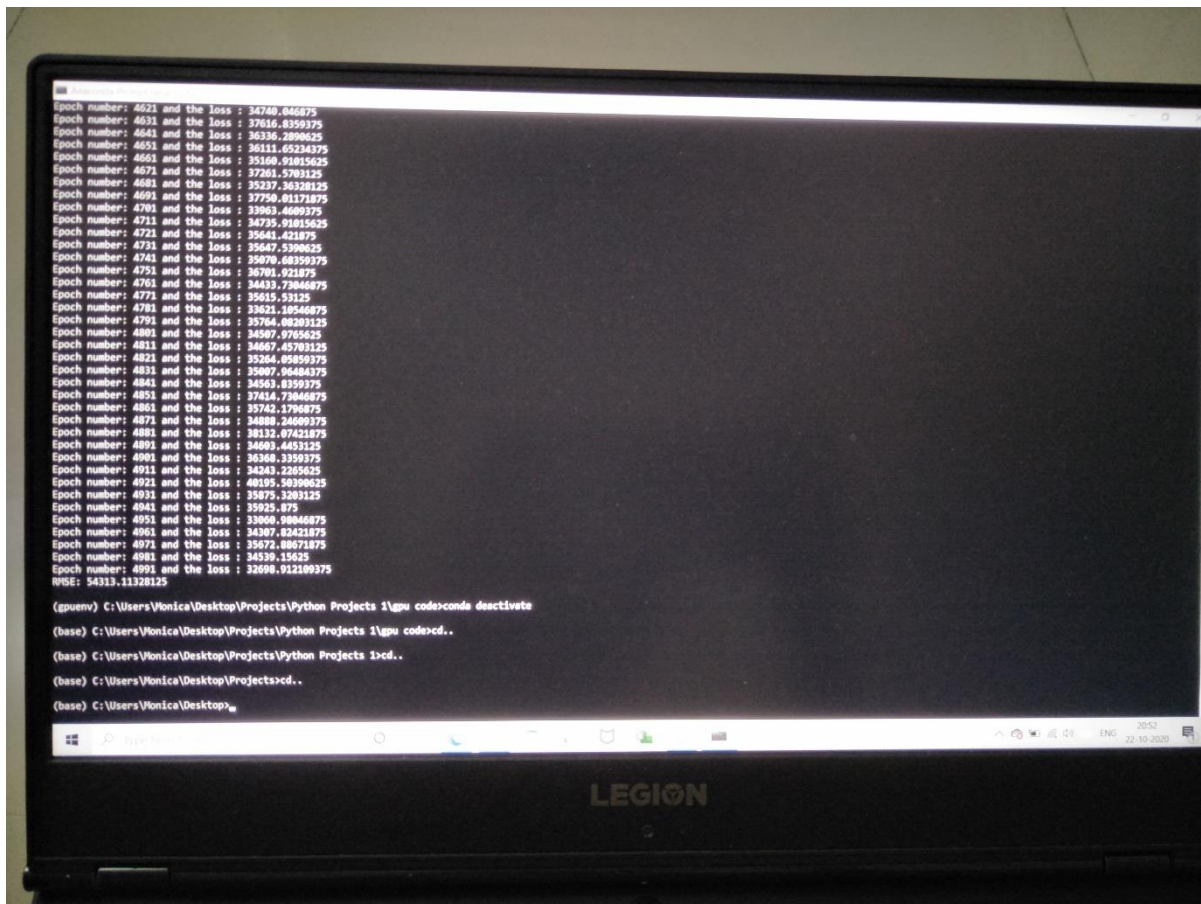
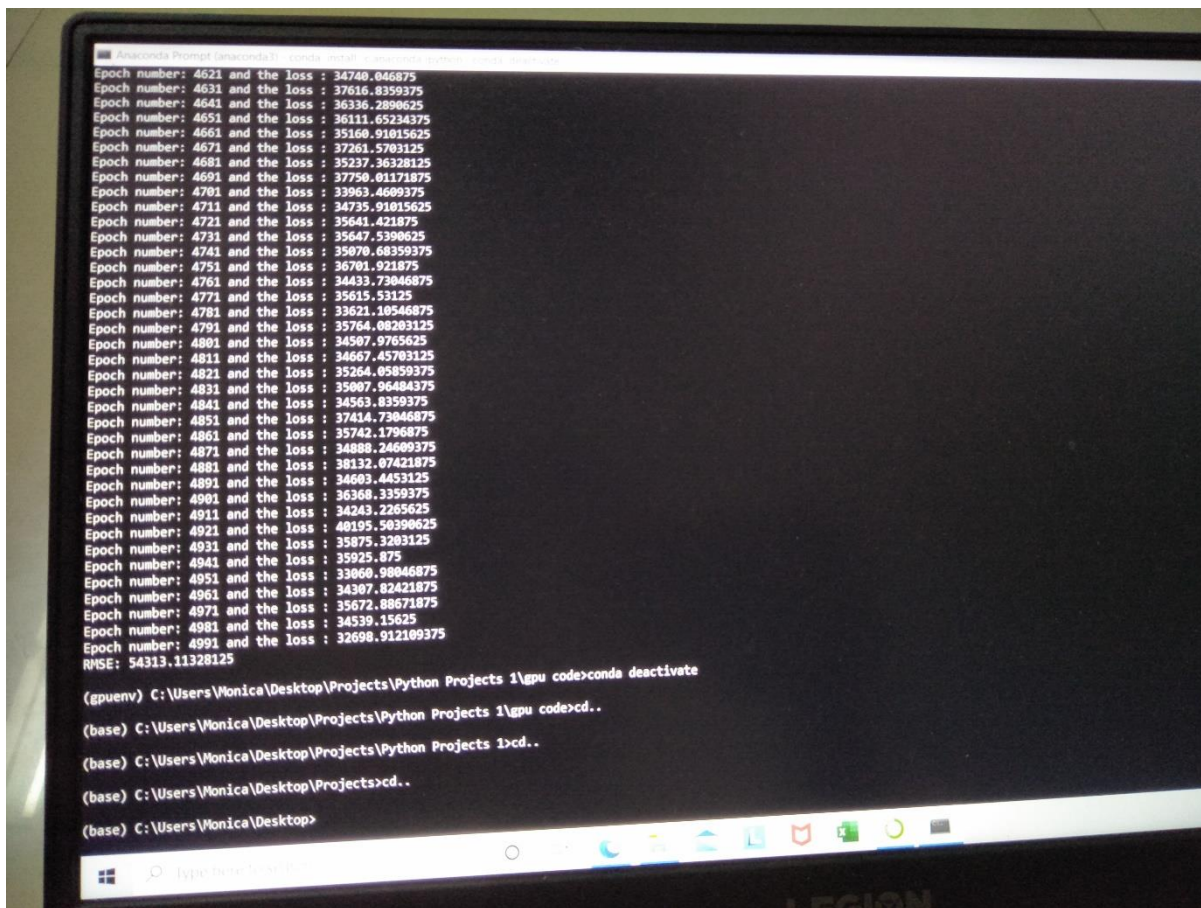


Anaconda Prompt (anaconda3)
 2020-10-22 05:15:38.143477: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1108]
 Model: "model"

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 224, 224, 3)]	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool1 (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool1 (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv4 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool1 (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv4 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool1 (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv4 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool1 (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0

Anaconda Prompt (anaconda3) - conda install -c anaconda python - conda deactivate

```
(gpuenv) C:\Users\Monica\Desktop\Projects\Python Projects 1\gpu code>python HouseGPU.py
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1201 entries, 0 to 1459
Data columns (total 10 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   MSSubClass  1201 non-null  int64
1   MSZoning    1201 non-null  object
2   LotFrontage 1201 non-null  float64
3   LotArea     1201 non-null  int64
4   Street      1201 non-null  object
5   LotShape    1201 non-null  object
6   YearBuilt   1201 non-null  int64
7   1stFlrSF    1201 non-null  int64
8   2ndFlrSF    1201 non-null  int64
9   SalePrice   1201 non-null  int64
dtypes: float64(1), int64(6), object(3)
memory usage: 103.2+ KB
Column name MSSubClass and unique values are 15
Column name MSZoning and unique values are 5
Column name LotFrontage and unique values are 110
Column name LotArea and unique values are 869
Column name Street and unique values are 2
Column name LotShape and unique values are 4
Column name YearBuilt and unique values are 112
Column name 1stFlrSF and unique values are 678
Column name 2ndFlrSF and unique values are 368
Column name SalePrice and unique values are 597
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1201 entries, 0 to 1459
Data columns (total 10 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   MSSubClass  1201 non-null  int64
1   MSZoning    1201 non-null  int32
2   LotFrontage 1201 non-null  float64
3   LotArea     1201 non-null  int64
4   Street      1201 non-null  int32
5   LotShape    1201 non-null  int32
6   1stFlrSF    1201 non-null  int64
7   2ndFlrSF    1201 non-null  int64
8   SalePrice   1201 non-null  int64
9   Total Years 1201 non-null  int64
dtypes: float64(1), int32(3), int64(6)
memory usage: 89.1 KB
Epoch number: 1 and the loss : 200496.75
Epoch number: 11 and the loss : 200493.484375
Epoch number: 21 and the loss : 200489.125
Epoch number: 31 and the loss : 200482.640625
```

Overview

This is diving into Deep Learning Project using GPU.

Deep learning is a subset of machine learning where artificial neural networks, algorithms inspired by the human brain, learn from large amounts of data. Deep learning allows machines to solve complex problems even when using a data set that is very diverse, unstructured and inter-connected.

This repository contains the code for Deep Learning using python's various libraries.

It used Numpy, Pandas, datetime, sklearn, Matplotlib and torch libraries.

These libraries help to perform individually one particular functionality.

Numpy is used for working with arrays. It stands for Numerical Python.

Pandas objects rely heavily on Numpy objects.

Matplotlib is a plotting library.

Sklearn has 100 to 200 models.

Datetime module supplies classes to work with date and time.

Pytorch is used for developing and training neural network based deep learning models.

The purpose of creating this repository is to gain insights into working with DL on GPU.

These python libraries raised knowledge in discovering these libraries with practical use of it.

It leads to growth in my DL repository.

This above few screenshots will help you to understand flow of output.

Motivation

The reason behind building this is, I wanted to start Deep Learning with something easy and that is when my mentor recommended me to start with Pytorch because it has simple to understand framework. And other reason is, is very simple to use, which also means that the learning curve for developers is relatively short. Other benefit is, If you're getting error in your code, you can start debugging by placing breakpoints using `pdb.set_trace()` at any appropriate line in your code. Then you can execute further computations and check the PyTorch Tensors or variables and nail down the root cause of the error. And hence less of other's guidance is needed if stuck. In addition to that, PyTorch has a very useful feature known as data parallelism. Using this feature, PyTorch can distribute computational work among multiple CPU or GPU cores and hence work faster. And PyTorch supports dynamic computational graphs, which means the network behavior can be changed programmatically at runtime. This facilitates more efficient model optimization and gives PyTorch a major advantage over other machine learning frameworks, which treat neural networks as static objects. With this dynamic approach, we can fully see each and every computation and know exactly what is going on. These are the points that assist me to create this project. Hence, I continue to gain knowledge while practicing the same and spread literary wings in tech-heaven.

Numpy contains a multi-dimensional array and matrix data structures. It works with the numerical data. Numpy is faster because it is densely packed in memory due to its homogeneous type. It also frees the memory faster.

Pandas module mainly works with the tabular data. It contains DataFrame and Series. Pandas is 18 to 20 times slower than Numpy. Pandas is seriously a game changer when it comes to cleaning, transforming, manipulating and analyzing data.

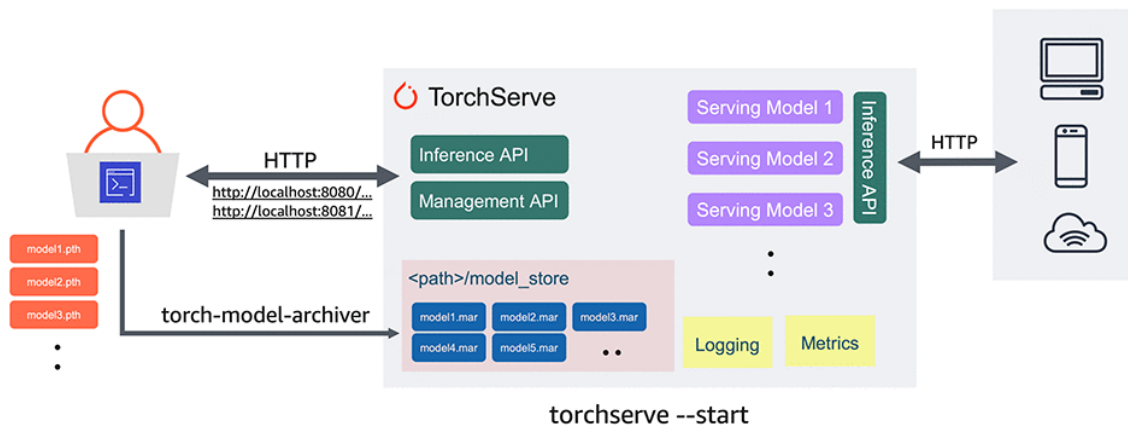
Matplotlib is used for EDA. Visualization of graphs helps to understand data in a better way than numbers in table format. Matplotlib is mainly deployed for basic plotting. It consists of bars, pies, lines, scatter plots and so on. Inline command display visualization inline within frontends like in Jupyter Notebook, directly below the code cell that produced it.

Sklearn is known as scikit learn. It provides many ML libraries and algorithms for it. It provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. Date and datetime are an object in Python, so when you manipulate them, you are actually manipulating objects and not string or timestamps. Working with dates and times is one of the biggest challenges in programming. Between dealing with time zones, daylight saving time, and different written date formats, it can be tough to keep track of which days and times you're referencing. Fortunately, the built-in Python datetime module can help you manage the complex nature of dates and times. For example, one great example of this irregularity is daylight saving time. In the United States and Canada, clocks are set forward by one hour on the second Sunday in March and set back by one hour on the first Sunday in November. However, this has only been the case since 2007. Prior to 2007, clocks were set forward on the first Sunday in April and set back on the last Sunday in October. Things get even more complicated when you consider time zones. Ideally, time zone boundaries would follow lines of longitude exactly.

PyTorch is as fast as TensorFlow, and potentially faster for Recurrent Neural Networks. PyTorch is heaven for researchers. PyTorch is a native Python package by design. PyTorch is built to be seamlessly integrated with Python and its popular libraries like NumPy.

Reason for selecting and working with pytorch is, it provides a complete end-to-end research framework which comes with the most common building blocks for carrying out every day deep learning research. It allows chaining of high-level neural network modules because it supports Keras-like API in its torch.

Pytorch Architecture:



Installation

Using intel core i5 9th generation with NVIDIA GFORCE GTX1650.

Windows 10 Environment Used.

Already Installed Anaconda Navigator for Python 3.x

The Code is written in Python 3.8.

If you don't have Python installed then please install Anaconda Navigator from its official site.

If you are using a lower version of Python you can upgrade using the pip package, ensuring you have the latest version of pip, *python -m pip install --upgrade pip and press Enter.*

Run/How to Use/Steps

Keep your internet connection on while running or accessing files and throughout too.

Follow this when you want to perform from scratch.

I have installed cuda Toolkit 10.0(2.1GB), cuDNN library ie cuda folder, VS Code Community 2017 with C++ libraries(6 to 7GB) and then created environment and pip install tensorflow-gpu==2.0.0 in order to use gpu.

After all installation, check in anaconda prompt and still if it gives error then switch off and on one time laptop then tried so worked all successfully.

Open Anaconda Prompt, Perform the following steps:

```
conda create -n gpuenv python=3.6
```

```
y
```

```
conda activate gpuenv
```

```
pip install tensorflow-gpu==2.0.0 [285.3MB]
```

```
python [to go in python shell]
```

```
import tensorflow
```

```
[see successfull message]
```

```
exit() [To come out from shell to prompt]
```

```
conda install pytorch torchvision cudatoolkit=10.0 -c pytorch [Type this in anaconda prompt gpuenv and not in shell]
```


nvidia-smi [Type this in prompt under gpuenv activate , it will show name of gpu device name]
python [to go in shell]
import torch
print(torch.cuda.is_available())
print(torch.cuda.current_device())
print(torch.cuda.get_device_name(0))
print(torch.cuda.memory_cached())
exit()
conda activate gpuenv
cd <PATH>
conda install -c anaconda ipython
run .py files.
You can also create requirement.txt file as, pip freeze > requirements.txt
run files.

Follow this when you want to just perform on local machine.

Download ZIP File.

Right-Click on ZIP file in download section and select Extract file option, which will unzip file.

Move unzip folder to desired folder/location be it D drive or desktop etc.

Open Anaconda Prompt, write cd <PATH> and press Enter.

eg: cd C:\Users\Monica\Desktop\Projects\Python Projects
1\19)GPU_Code\HousePricePrediction_with_pytorch_gpu

conda activate gpuenv

In Anconda Prompt, pip install -r requirements.txt to install all packages.

In Anconda Prompt, write <filename>.py and press Enter. That is,

In Anconda Prompt, write HousePriceGPU.py and press Enter.

This takes houseprice.csv file as input dataset and creates HousePrice.pt, HouseWeights.pt as output files in same working folder and then I created empty folder named Output_Files and transferres it there.

Please be careful with spellings or numbers while typing filename and easier is just copy filename and then run it to avoid any silly errors.

Note: cd <PATH>

[Go to Folder where file is. Select the path from top and right-click and select copy option and paste it next to cd one space <path> and press enter, then you can access all files of that folder] [cd means change directory]

Directory Tree/Structure of Project

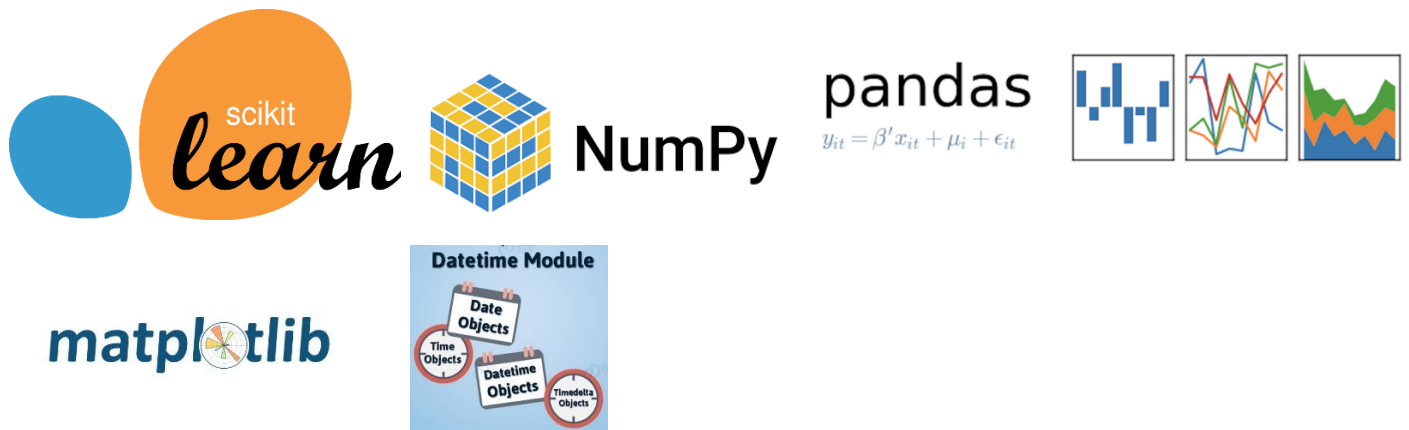
Folder: 19)GPU_Code > HousePricePrediction_with_pytorch_gpu

HousePriceGPU.py

To Do/Future Scope

Can try with image dataset.

Technologies Used/System Requirements/Tech Stack



Credits

Krish Naik Channel