Training ML.NET on GPUs using Google Colab

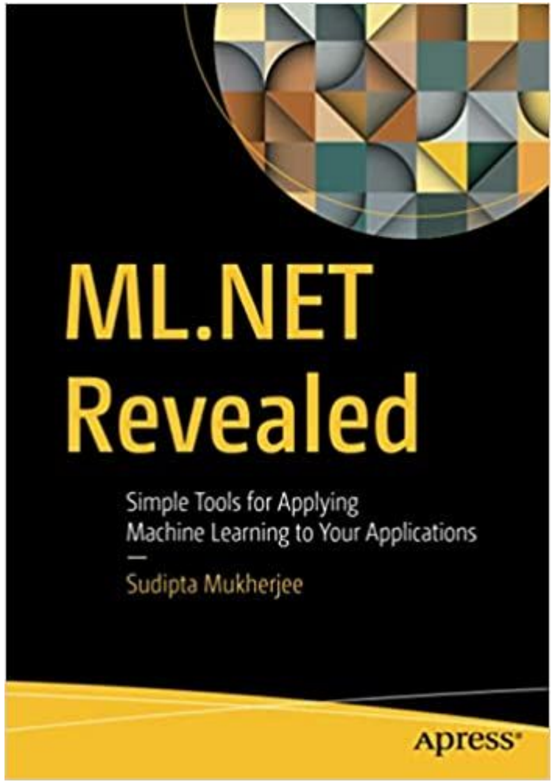
Brandon Atkinson

December 2021, Practical ML.NET User Group

Thank you Apress!

Apress has donated a digital copy of `**ML.NET Revealed**` by Sudipta Mukherjee to giveaway for this presentation.

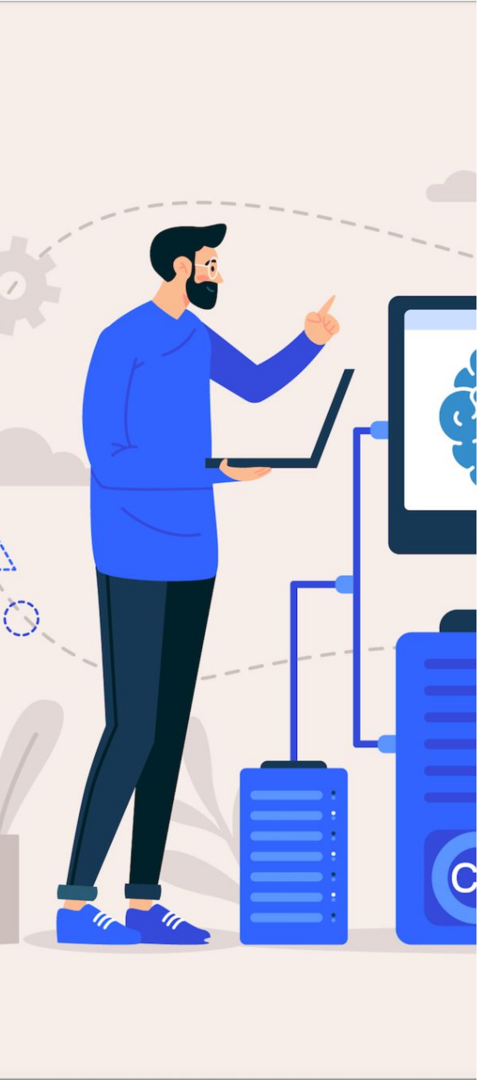
We'll draw a random winner at the end of the event.


ML.NET Revealed
Simple Tools for Applying Machine Learning to Your Applications
Sudipta Mukherjee
Apress

Brandon Atkinson

Accomplished technology leader with over 15 years of industry experience encompassing analysis, design, development, and implementation of enterprise-level solutions. My passion is building successful teams and people as well as enterprise architecture that can transform businesses and alleviate pain points.

https://www.linkedin.com/in/brandongatkinson/



Topics

- What is Google Colab and why would we use it
- Setting up Colab to run ML.NET with GPUs
- Getting our training code to Colab
- Training our model, predicting, and downloading
- Closing Thoughts
- QA

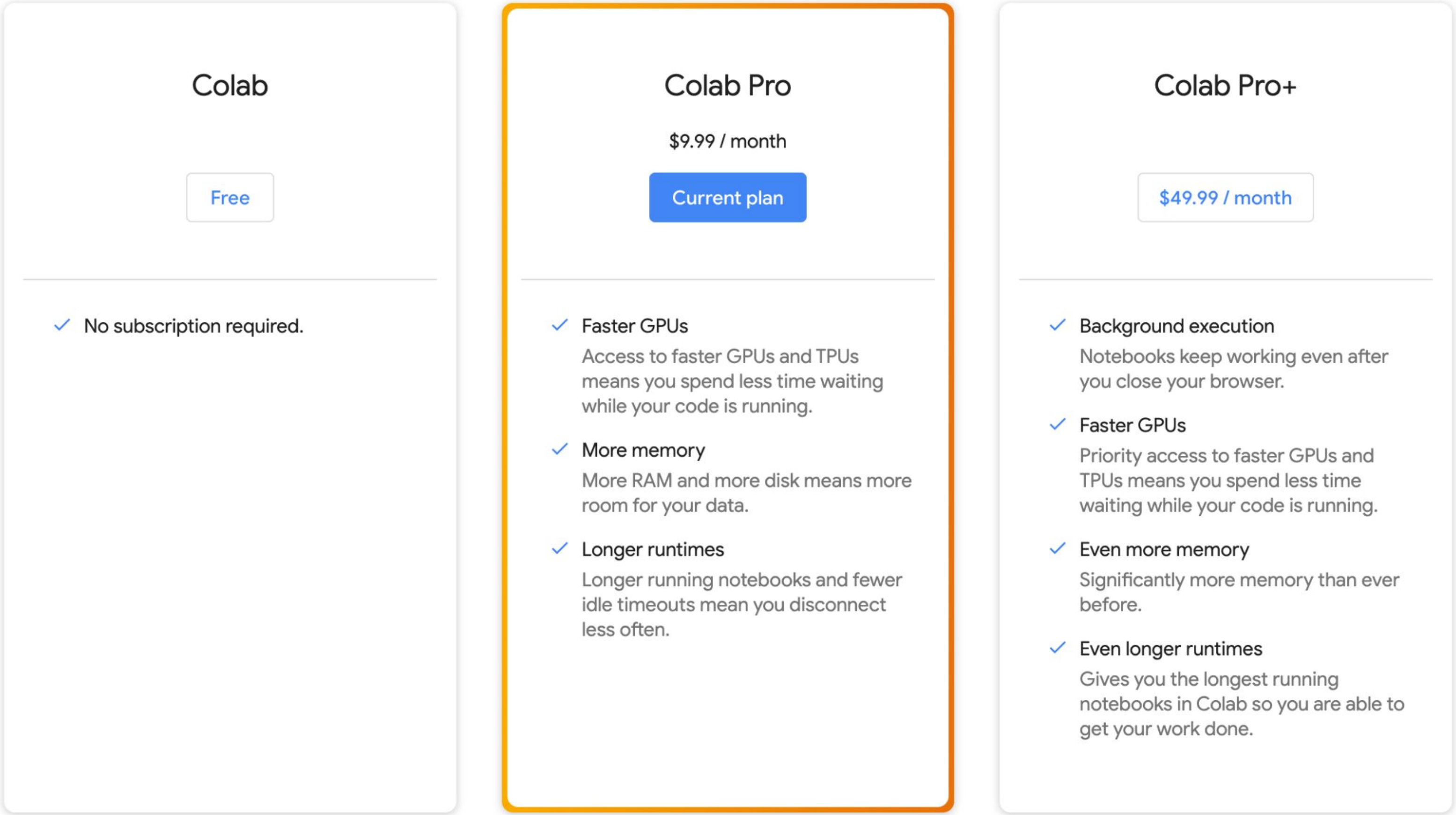
What is Google Colab and Why Would We Use It

What is Google Colab

Colaboratory, or “Colab” for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary Python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically, Colab is a hosted Jupyter notebook service that requires no setup to use, while providing free access to computing resources including GPUs.

- Not limited to Python!
- You get an Ubuntu environment, with some restrictions on what you can install.
- From their docs: “Colab focuses on supporting Python and its ecosystem of third-party tools.”
- Don’t expect any support from Google when running your ML.NET code :(

Various Flavors of Colab



Colab

Free

✓ No subscription required.

Colab Pro

$9.99 / month

Current plan

✓ Faster GPUs
Access to faster GPUs and TPUs means you spend less time waiting while your code is running.

✓ More memory
More RAM and more disk means more room for your data.

✓ Longer runtimes
Longer running notebooks and fewer idle timeouts mean you disconnect less often.

Colab Pro+

$49.99 / month

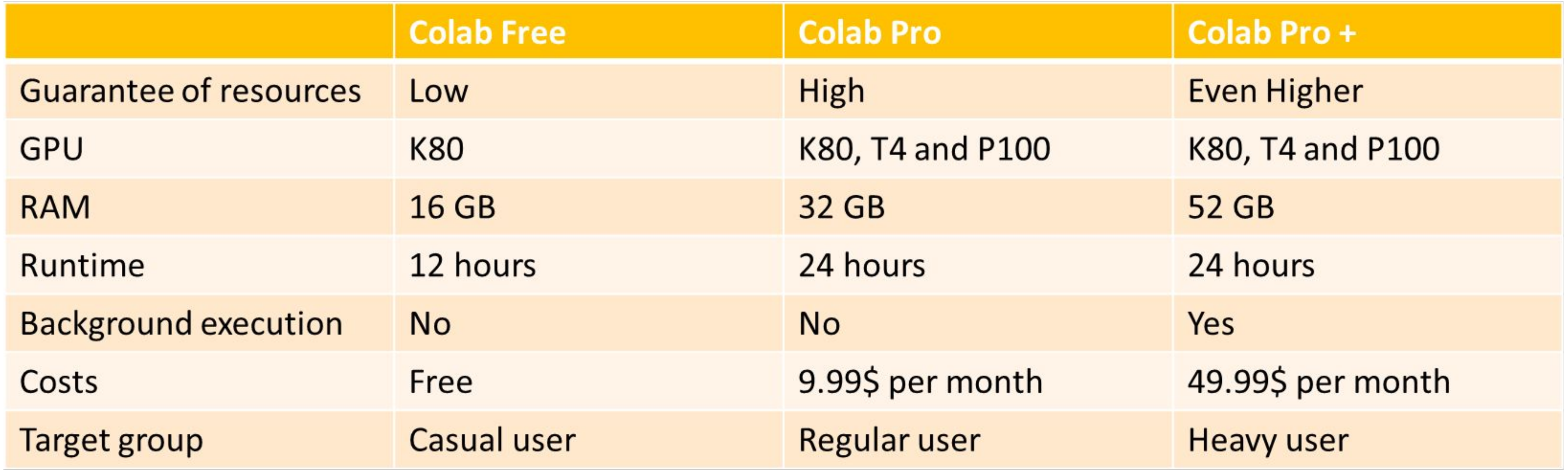
✓ Background execution
Notebooks keep working even after you close your browser.

✓ Faster GPUs
Priority access to faster GPUs and TPUs means you spend less time waiting while your code is running.

✓ Even more memory
Significantly more memory than ever before.

✓ Even longer runtimes
Gives you the longest running notebooks in Colab so you are able to get your work done.

Various Flavors of Colab

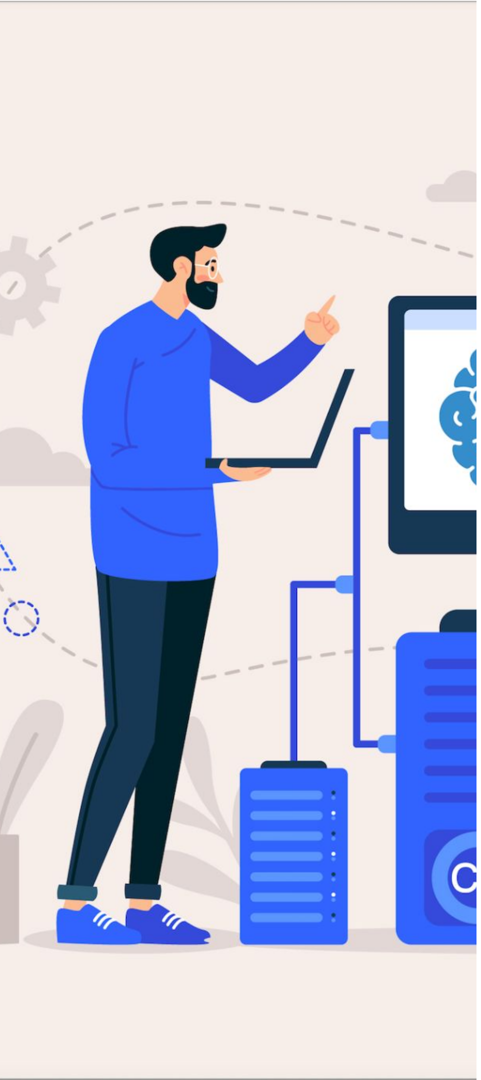


	Colab Free	Colab Pro	Colab Pro +
Guarantee of resources	Low	High	Even Higher
GPU	K80	K80, T4 and P100	K80, T4 and P100
RAM	16 GB	32 GB	52 GB
Runtime	12 hours	24 hours	24 hours
Background execution	No	No	Yes
Costs	Free	9.99$ per month	49.99$ per month
Target group	Casual user	Regular user	Heavy user

Why use Google Colab?

Colab is very inexpensive for what it is offering you. The storage and RAM available, even at the free tier is very impressive. In addition to that, there are several good reasons to look at Colab over some other options:

- **Azure** - Azure is an amazing product with A LOT of great offerings for machine learning. Azure works really well if you're already in the ecosystem, but for users who are using AWS or GCP, spinning up Azure for a single task may be overkill.
- **Kaggle** - Kaggle is pretty much built for Python machine learning. You can not install other frameworks there, like .NET or ML.NET.
- **AWS or GCP** - There are certainly options for running notebooks in these environments, but with any cloud provider, keep an eye on your costs. Running models for hours at a time will add up!
- **There are plenty of options out there** - What a time to be working in machine learning, there are so many places to work and train models!



Who should use Colab?

Running your ML workloads on Colab is definitely not for everyone!

- **Individuals / Students** - If you are looking to play around with ML.NET on GPUs for instance, and just learn, this is an excellent option.
- **Startups** - Given that you have a free tier, and Pro is fairly inexpensive for what you get, this may be a great starting point for start-ups depending on the needs.
- **MacBook Users (like me!)** - I really wanted to train on GPUs, but this was not an option for me, so Colab was a great alternative.
- **Larger Businesses, probably not** - If you have the money to run workloads in Azure, AWS, or GCP then you should definitely be doing that.

Setting up Colab to run ML.NET with GPUs

It’s fairly straightforward...

Getting ML.NET to run on Colab is very easy, getting it to run on GPUs took a little more effort and investigation. However, once all the pieces are in place, it’s very easy to replicate.

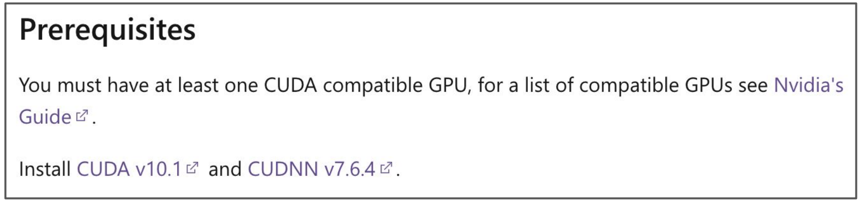
The following resources were used (links at the end):

- **ImageClassificationTrainer Class Docs** - Have great info on what is required for running on GPUs.
- **SciSharp/TensorFlow.NET Docs** - Fills in the gaps from the ImageClassificationTrainer docs.
- **Install TensorFlow for C** - Fills in the gaps from the SciSharp/TensorFlow.NET docs.
- **Hundreds of Google searches!**

ImageClassification Trainer Class Docs

A couple of very important notes from these docs:

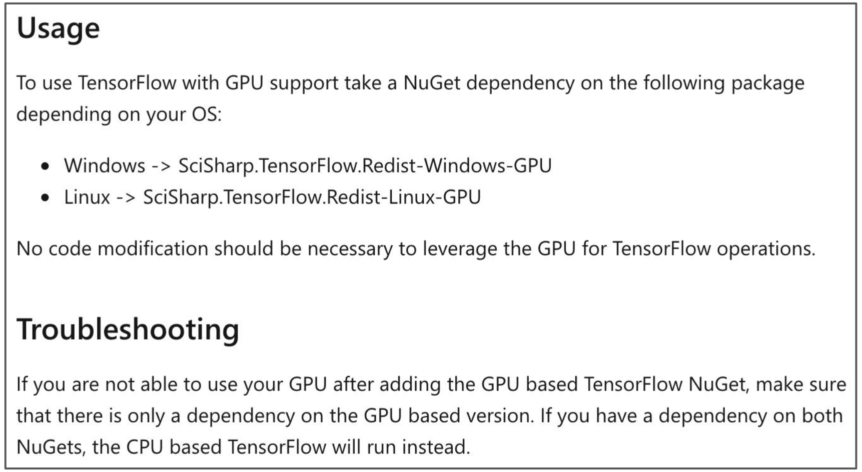
- **Prerequisites** - CUDA v10.1 and CUDNN v7.6.4 are listed as required. CUDA drivers are backwards compatible, so this is a minimum.
- **Usage** - Make sure you only reference the "GPU" SciSharp NuGet package. Having both the GPU and CPU packages will result in a downgrade to CPU.



Prerequisites

You must have at least one CUDA compatible GPU, for a list of compatible GPUs see Nvidia's Guide.

Install CUDA v10.1 and CUDNN v7.6.4.



Usage

To use TensorFlow with GPU support take a NuGet dependency on the following package depending on your OS:

- Windows -> SciSharp.TensorFlow.Redist-Windows-GPU
- Linux -> SciSharp.TensorFlow.Redist-Linux-GPU

No code modification should be necessary to leverage the GPU for TensorFlow operations.

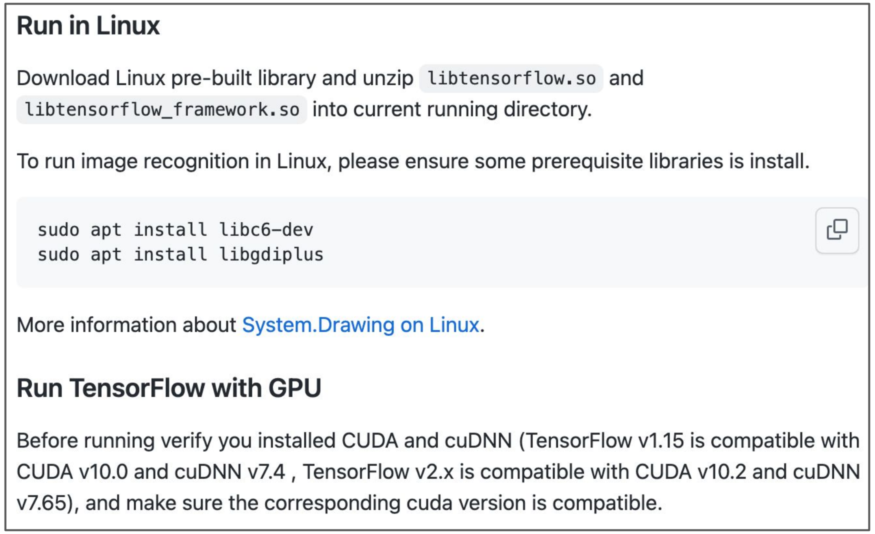
Troubleshooting

If you are not able to use your GPU after adding the GPU based TensorFlow NuGet, make sure that there is only a dependency on the GPU based version. If you have a dependency on both NuGets, the CPU based TensorFlow will run instead.

SciSharp/TensorFlow.NET Docs

A couple of very important notes from these docs:

- **Run in Linux** - The `libtensorflow` libraries listed here need to be downloaded from Tensorflow (next slide).
- **Run TensorFlow with GPU** - These docs make the same statement about the CUDA requirement.



Run in Linux

Download Linux pre-built library and unzip `libtensorflow.so` and `libtensorflow_framework.so` into current running directory.

To run image recognition in Linux, please ensure some prerequisite libraries is install.

```
sudo apt install libc6-dev
sudo apt install libgdiplus
```

More information about System.Drawing on Linux.

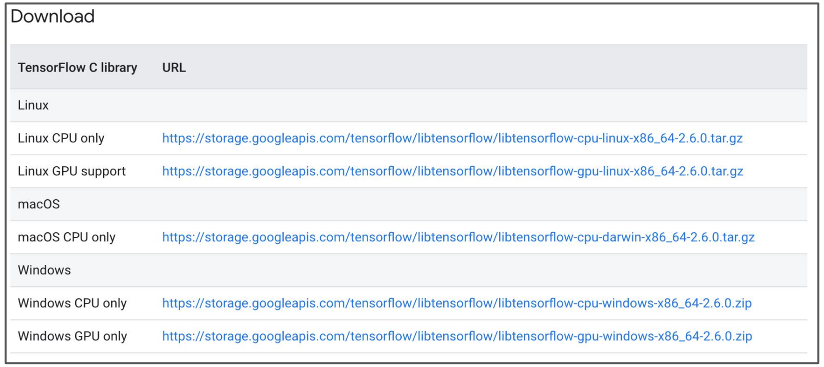
Run TensorFlow with GPU

Before running verify you installed CUDA and cuDNN (TensorFlow v1.15 is compatible with CUDA v10.0 and cuDNN v7.4 , TensorFlow v2.x is compatible with CUDA v10.2 and cuDNN v7.65), and make sure the corresponding cuda version is compatible.

Install TensorFlow for C

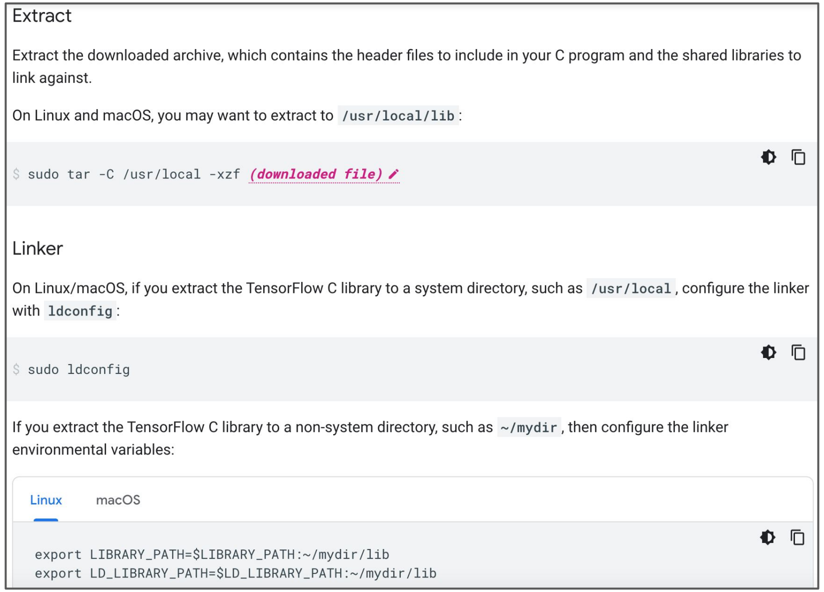
A couple of very important notes from these docs:

- **Download** - Provides URLs for all the library downloads.
- **Extract**- Good info on where to extract. I used “/usr/local” based on this info, as “/usr/local/lib” did not seem to work.
- **Linker** - VERY IMPORTANT! Running the “ldconfig” command and setting the environment variables are crucial.



Download

TensorFlow C library	URL
Linux	
Linux CPU only	https://storage.googleapis.com/tensorflow/libtensorflow/libtensorflow-cpu-linux-x86_64-2.6.0.tar.gz
Linux GPU support	https://storage.googleapis.com/tensorflow/libtensorflow/libtensorflow-gpu-linux-x86_64-2.6.0.tar.gz
macOS	
macOS CPU only	https://storage.googleapis.com/tensorflow/libtensorflow/libtensorflow-cpu-darwin-x86_64-2.6.0.tar.gz
Windows	
Windows CPU only	https://storage.googleapis.com/tensorflow/libtensorflow/libtensorflow-cpu-windows-x86_64-2.6.0.zip
Windows GPU only	https://storage.googleapis.com/tensorflow/libtensorflow/libtensorflow-gpu-windows-x86_64-2.6.0.zip



Extract

Extract the downloaded archive, which contains the header files to include in your C program and the shared libraries to link against.

On Linux and macOS, you may want to extract to `/usr/local/lib`:

```
$ sudo tar -C /usr/local -xzf (downloaded file)
```

Linker

On Linux/macOS, if you extract the TensorFlow C library to a system directory, such as `/usr/local`, configure the linker with `ldconfig`:

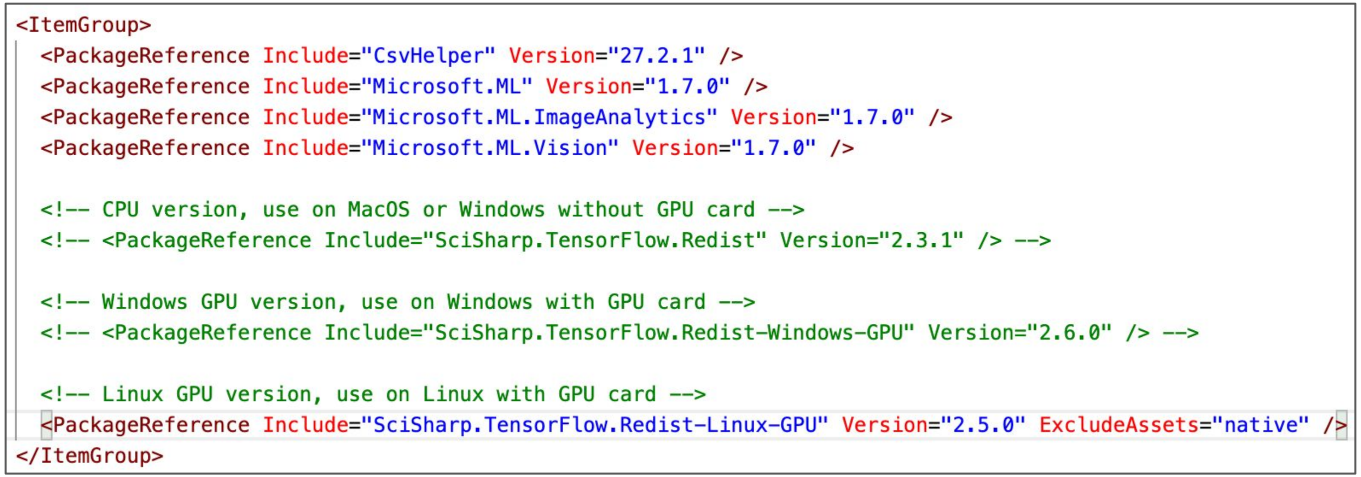
```
$ sudo ldconfig
```

If you extract the TensorFlow C library to a non-system directory, such as `~/mydir`, then configure the linker environmental variables:

Linux macOS

```
export LIBRARY_PATH=$LIBRARY_PATH:~/mydir/lib
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:~/mydir/lib
```

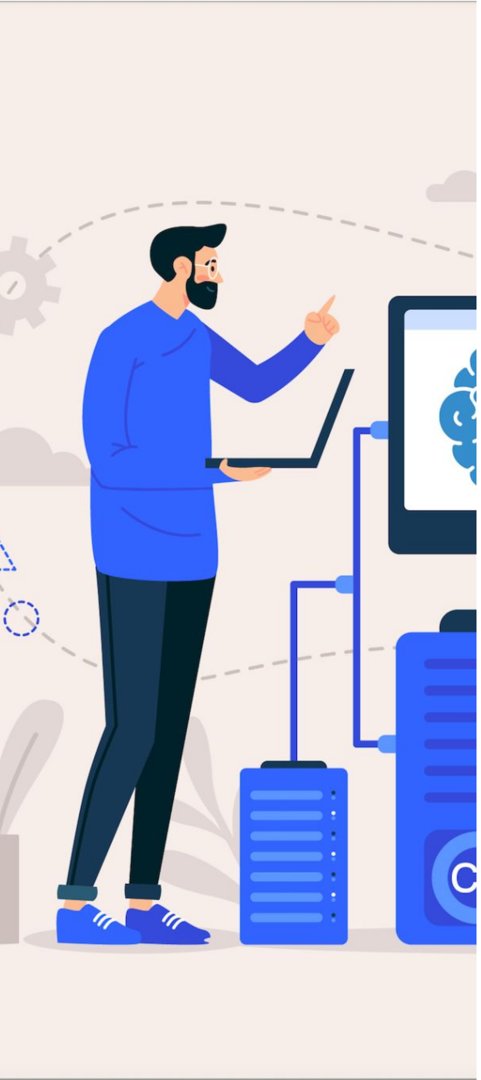
Project Setup



```xml
<ItemGroup>
  <PackageReference Include="CsvHelper" Version="27.2.1" />
  <PackageReference Include="Microsoft.ML" Version="1.7.0" />
  <PackageReference Include="Microsoft.ML.ImageAnalytics" Version="1.7.0" />
  <PackageReference Include="Microsoft.ML.Vision" Version="1.7.0" />

  <!-- CPU version, use on MacOS or Windows without GPU card -->
  <!-- <PackageReference Include="SciSharp.TensorFlow.Redist" Version="2.3.1" /> -->

  <!-- Windows GPU version, use on Windows with GPU card -->
  <!-- <PackageReference Include="SciSharp.TensorFlow.Redist-Windows-GPU" Version="2.6.0" /> -->

  <!-- Linux GPU version, use on Linux with GPU card -->
  <PackageReference Include="SciSharp.TensorFlow.Redist-Linux-GPU" Version="2.5.0" ExcludeAssets="native" />
</ItemGroup>
```

Project setup requires the correct package reference:

- **SciSharp.Tensorflow.Redist** - Our project file has all the versions of this package, but we only use the one specific to where we will run the code.
- **ExcludeAssets** - Setting this to “native” tells the compiler to not include the native tensorflow binaries, we will bring our own in Colab, which allows us more fine grain control over what is installed and where.



Demo

Closing Thoughts

Pros and cons of using Colab for ML.NET

Pros

- Ubuntu environment which allows for .NET
- Free and/or cheap to use
- Monthly fees are capped
- Lots of storage and RAM
- Setup is fairly straightforward
- Uses Drive which makes up/downloading files easy
- Can quickly get up and running

Cons

- Can be difficult to setup your first time
- Must use Pro+ for background processing
- Better suited for individuals / startups
- Not a dedicated to you environment, libraries installed could change without notice
- Single GPU sessions

Q&A

References

- **Google Colab** - https://research.google.com/colaboratory/
- **ImageClassificationTrainer Class Docs** - https://docs.microsoft.com/en-us/dotnet/api/microsoft.ml.vision.imageclassificationtrainer?view=ml-dotnet
- **SciSharp/TensorFlow.NET Docs** - https://github.com/SciSharp/TensorFlow.NET/tree/master/tensorflowlib
- **Install TensorFlow for C Docs** - https://www.tensorflow.org/install/lang_c
- **Code Example** - https://github.com/atkinsonbg/training-mlnet-using-google-colab-on-gpus
- **YouTube Presentation** - https://www.youtube.com/channel/UCyLZieUSKFGLBOibC7xB1zg/videos
- **GitHub Issue on ExcludeAssets** - https://github.com/dotnet/machinelearning/issues/871