Third-party deep-learning packages integration for TMVA

Stefan Wunsch

September 15, 2016

Who am I?

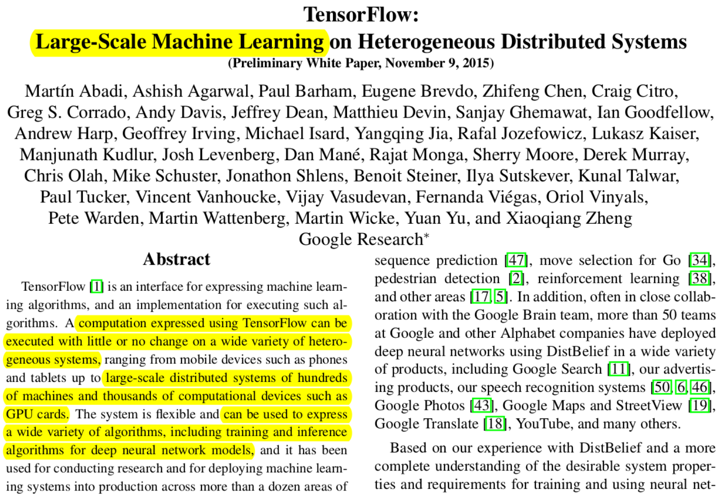
- Master student from KIT
- Just finished Summer Student (with CMS Muon POG about HLT and L1 trigger)
- Will be around for roughly one year, mainly working from Karlsruhe
- Working on deep-learning integration in TMVA

Karlsruher Institut für Technologie

CERN

Third-party deep-learning packages integration for TMVA

Why do we want third-party packages in TMVA?



TensorFlow:
Large-Scale Machine Learning on Heterogeneous Distributed Systems
(Preliminary White Paper, November 9, 2015)

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng
Google Research*

Abstract

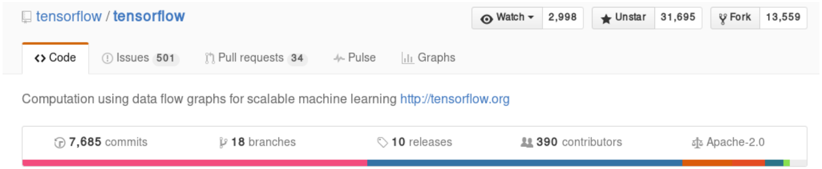
TensorFlow [1] is an interface for expressing machine learning algorithms, and an implementation for executing such algorithms. A computation expressed using TensorFlow can be executed with little or no change on a wide variety of heterogeneous systems, ranging from mobile devices such as phones and tablets up to large-scale distributed systems of hundreds of machines and thousands of computational devices such as GPU cards. The system is flexible and can be used to express a wide variety of algorithms, including training and inference algorithms for deep neural network models, and it has been used for conducting research and for deploying machine learning systems into production across more than a dozen areas of sequence prediction [47], move selection for Go [34], pedestrian detection [2], reinforcement learning [38], and other areas [17, 5]. In addition, often in close collaboration with the Google Brain team, more than 50 teams at Google and other Alphabet companies have deployed deep neural networks using DistBelief in a wide variety of products, including Google Search [11], our advertising products, our speech recognition systems [50, 6, 46], Google Photos [43], Google Maps and StreetView [19], Google Translate [18], YouTube, and many others.

Based on our experience with DistBelief and a more complete understanding of the desirable system properties and requirements for training and using neural net-

Tensorflow whitepaper, 2015

Why do we want third-party packages in TMVA? (2)

- **Huge community** which results in **stable software** (Theano, Tensorflow, . . .)
- Field of machine learning **changes rapidly**; hard to keep pace, e.g., with Google Deepmind
- **Staying up-to-date** without implementing functionality by yourself
- **High benefit with low maintenance**



Computation using data flow graphs for scalable machine learning http://tensorflow.org

7,685 commits 18 branches 10 releases 390 contributors Apache-2.0

Keras Implementation

- **Keras:** High-level API for Theano and Tensorflow
- **Done so far:** Binary classification is fully working

Why Keras?

- Tensorflow and Theano are low-level APIs, needs profound knowledge about deep learning to use
- Supports Theano and Tensorflow at once (leading deep-learning frameworks)

Details: Defining model

- **Configuration strings** are **not suitable** for complex models
- Model is defined in Python file, which is parsed to YAML model definition (Keras feature)

Example Python code (3-layer feed-forward network):

```
model = Sequential()
model.add(Dense(num_hidden_nodes, init='uniform',
    activation='relu', input_dim=num_input_nodes,
    W_regularizer=l2(1e-4)))
model.add(Dense(num_output_nodes, init='uniform',
    activation='softmax'))

with open('model.yml', 'w') as f:
    f.write(model.to_yaml())
```

That's all!

Details: Model definition

Model definition `model.yml` produced by Python file:

```
keras_version: 1.0.6
class_name: Sequential
config:
- class_name: Dense
  config:
    W_constraint: null
    W_regularizer: {l1: 0.0, l2: 0.01}
    activation: relu
    activity_regularizer: null
    b_constraint: null
    b_regularizer: null
...
```

You don't need to look at this during the workflow, though it's easily readable!

Details: Import model in TMVA

- Book method as usual with model definition `model.yml` as parameter

```
factory->BookMethod(dataloader,
    TMVA::Types::kPyKeras, "PyKeras",
    "H:!V:VarTransform=D,N:FilenameModel=model.yml:\
    Optimizer=SGD:NumEpochs=20:BatchSize=32:\
    LearnRate=0.01");
```

Acceleration, e.g., for Theano:

- **GPU:** Just run `export THEANO_FLAGS=device=gpu`, works out of the box if **CUDA** is installed
- **CPU:** Use `export THEANO_FLAGS='openmp=True'` to use **OpenMP** (done by default if more than one core is detected)

Coming features

- **DONE:** Training on pre-trained models (picking up weights from file)
- **DONE:** Storing only the weights with lowest validation error (and not simply the weights after a fix numer of epochs)
- **Regression**
- **Multi-class** classification
- Training with **sample weights** (directly supported by Keras)
- **Pre-training** methods, stacked training
- **Suggestions?**

Outlook

- On top follows application on analysis and **evaluation of systematics**
- **Further suggestions** for deliverables or comments on presented ideas are very welcome