Covariant Compositional Networks Library Manual

Truong Son Hy

December 5, 2019

Contents

1 Overview

Covariant Compositional Networks Library is an easy-to-use and efficient implementation of Covariant Compositional Networks (CCNs) with TensorFlow and PyTorch's APIs based on a shared common C++ core. Algorithms of CCNs are published in [Kondor et al., 2018] and [Hy et al., 2018]. The whole package can be found at:

`https://github.com/HyTruongSon/LibCCNs`

The original release of CCNs implementation was based on GraphFlow Deep Learning Framework at:

`https://github.com/HyTruongSon/GraphFlow`

However, the GraphFlow-based implementation has its own limitation of scaling up to learning on large-scale networks (e.g. citation graphs, or knowledge graphs), because we store each vertex's representation in a standalone tensor, thus there are a huge number of tensors during training and that also slows down the computation. Instead we concatenate all the representations in a single big tensor and do all the contractions at once. In the case of learning molecular properties, we have many different molecular graphs, and we need dynamic batching, the strategy would be to concatenate all the molecular graphs of a batch into a single graph (each connected component is a molecular graph) and concatenate all their vertex representations into a single big tensor. In practice, the single big tensor is stored as two-dimensional array in which the second index is for the channels.

2 C++ core

Both TensorFlow side and PyTorch side use the same C++ core for actual contraction computation. It is located in `cpp/` directory.

File	**Role**
`ccn1d_cpu.h`	• Actual C++ implementation of 5 contractions of CCN 1D: forward and backward. • Actual C++ implementation of tensor shrinking operation: forward and backward. • Actual C++ implementation of tensor normalization: foward and backward. • Helper functions for initialization of receptive fields.
`common.h`	Tensor indexing helper functions.

3 TensorFlow side

TensorFlow implementation of CCNs library using the C++ core (in sub-directory `ccn_lib/`), CCNs models, and training programs in Python for both learning on large-scale citation graphs (in sub-directory `large_graph/`) and learning molecular properties (in sub-directory `small_graphs/`) are included in folder `tensorflow/`.

Directory	**File**	**Role**
`ccn_lib/`	`ccn1d_grad.py`	Customized gradient computation of user-defined TensorFlow operators defined in `ccn1d_lib.cc`
`ccn_lib/`	`ccn1d_lib.cc`	User-defined TensorFlow operators in C++ using the C++ core
`ccn_lib/`	`compile.sh`	Script to compile the library
`ccn_lib/`	All other `*.py`	Gradient definition in Python

Directory	**File**	**Role**
`large_graph/`	`ccn1d_training_program.py`	Training program in Python
`large_graph/`	`ccn1d_training_program.sh`	Training script
`large_graph/`	`CCN1D.py`	CCN 1D model for large citation graph
`large_graph/`	`Dataset.py`	Dataset data structure of citation graph in Python
`large_graph/`	`Edge.py`	Edge of citation graph definition
`large_graph/`	`Vertex.py`	Vertex of citation graph definition

Directory	**File**	**Role**
`small_graphs/`	`Atom.py`	Atom of molecular graph definition
`small_graphs/`	`ccn1d_training_program.py`	Training program in Python
`small_graphs/`	`ccn1d_training_program.sh`	Training script
`small_graphs/`	`CCN1D.py`	CCN 1D model for molecular graphs
`small_graphs/`	`Dataset.py`	Dataset data structure of molecular graph in Python
`small_graphs/`	`Graph.py`	Graph as a concatenation of multiple molecular graphs
`small_graphs/`	`Molecule.py`	A single molecular graph

4 PyTorch side

PyTorch implementation of CCNs library using the C++ core (in sub-directory `ccn_lib/`), CCNs models, and training programs in Python for both learning on large-scale citation graphs (in sub-directory `large_graph/`) and learning molecular properties (in sub-directory `small_graphs/`) are included in folder `pytorch/`.

Directory	**File**	**Role**
`ccn_lib/`	`ccn1d_contractions.py`	Pythonic binding from PyTorch C++ into PyTorch operator of CCN 1D contractions
`ccn_lib/`	`ccn1d_lib.cpp`	Binding from the C++ core into PyTorch C++ API
`ccn_lib/`	`ccn1d_normalizing.py`	Pythonic binding from PyTorch C++ into PyTorch operator of normalizing tensor
`ccn_lib/`	`ccn1d_shrinking.py`	Pythonic binding from PyTorch C++ into PyTorch operator of shrinking a high-order tensor to a 1-dimenisonal tensor
`ccn_lib/`	`compile.sh`	Compilation script
`ccn_lib/`	`setup.py`	Setup

Directory	**File**	**Role**
`large_graph/`	`ccn1d_training_program.py`	Training program in Python
`large_graph/`	`ccn1d_training_program.sh`	Training script
`large_graph/`	`CCN1D.py`	PyTorch CCN 1D model
`large_graph/`	`Dataset.py`	Large-graph citation network data structure
`large_graph/`	`Edge.py`	Edge definition
`large_graph/`	`GCN.py`	Graph Convolution Neural Network model (this is not CCNs)
`large_graph/`	`gcn_training_program.py`	GCN's training program
`large_graph/`	`gcn_training_program.sh`	GCN's training script
`large_graph/`	`Vertex.py`	Vertex definition

Directory	**File**	**Role**
`small_graphs/`	`Atom.py`	Atom definition
`small_graphs/`	`ccn1d_training_program.py`	Training program in Python
`small_graphs/`	`ccn1d_training_program.sh`	Training script
`small_graphs/`	`CCN1D.py`	CCN 1D model in PyTorch
`small_graphs/`	`Dataset.py`	Small-graph molecules dataset structure
`small_graphs/`	`Graph.py`	Graph definition as concatenation of multiple molecules
`small_graphs/`	`Molecule.py`	Molecular graph definition

5 Node classification in large-scale networks

5.1 Datasets

5.2 Data pre-processing

5.3 Some results

6 Learning molecular properties

6.1 Datasets

6.2 Data pre-processing

6.3 Some results

References

[Hy et al., 2018] Hy, T. S., Trivedi, S., Pan, H., Anderson, B. M., and Kondor, R. (2018). Predicting molecular properties with covariant compositional networks. *The Journal of Chemical Physics*, 148(24):241745.

[Kondor et al., 2018] Kondor, R., Son, H. T., Pan, H., Anderson, B. M., and Trivedi, S. (2018). Covariant compositional networks for learning graphs. *CoRR*, abs/1801.02144.

List of Figures