**parser\_transitions.py**

* Creates the basic dependency parsing machinery
  + Independent of neural net implementation
  + All it assumes is some kind of classifier to predict the next transition for a partial parse. Does not need to be a neural network classifiier
* Defines PartialParse class
  + init(): initialize a partial parse object (stack, buffer and dependency lists)
  + parse\_step(): apply a single transition (S/LA/RA) to a single partial parse object
  + parse(): apply a sequence of transitions to a partial parse object
* Defines minbatch\_parse function

**parser\_utils.py**

* Creates Parser class
  + "Contains everything needed for transition-based dependency parsing except for the model"
  + extract\_features() returns a set of features based on the present state of the stack, buffer and dependencies for a partially parsed sentence
  + Does a bunch of stuff that I don’t understand

**parser\_model.py**

* Defines a PyTorch neural network for dependency parsing

**run.py**

* Trains the network on training data
* Train() function
  + Defines loss function (cross entropy)
  + Defines optimizer (Adam)
  + Runs several (default 10) training epochs using the neural dependency parser defined in **parser\_model.py**
* Train\_for\_epoch() function