README:

This is a seq2seq LSTM neural network based on the one from Google (https://github.com/tensorflow/nmt) which is used here for morphology tagging.
Unlike the neural network from Google, this one can handle also an hybrid input, where the network trains not only on the tokens, but combines these with a representation of their characters, essentially extracting a hybrid representation for the input with both word and subword information.

Training:

There are many flags for training the model which mainly control the input files and the hyperparameters of the model such as the `number of training steps or learning rate. For a full list of all flags please refer to file nmt_hybrid/nmt.py
Here are some of the more important flags:

Input files:
--train-prefix: The path to train files, for example if --train-prefix=/nmt/trn then the source train file is /nmt/trn.from and the target train file is /nmt/trn.to
--dev-prefix: As --train-prefix but for the development/validation dataset
--test-prefix: As --train-prefix but for the test dataset
--vocab-prefix: The path to the vocabulary files, for example if --vocab-prefix=/nmt/vocab then the source and target vocabulary files will be vocab.from and vocab.to respectively
--src-suffix: The suffix for source, default is .from
--tgt-suffix: The suffix for target, default is .to
--out-dir: Stores the output files

Hyperparameters:
--num_train_steps: The number of training steps, default is 20000
--learning_rate: The training learning rate, recommended 0.001 for adam and 1.0 for sgd as optimizers.
--num_layers: How many layers should the network have, default is 2, i.e. 2 en
--optimizer: can be either sgd or adam, adam is recommended
--decay_scheme: Schemes to decay the learning rate. Default is none. Options are
    *luong234: after 2/3 num train steps, we start halving the learning rate
     for 4 times before finishing.
    *luong5: after 1/2 num train steps, we start halving the learning rate
     for 5 times before finishing.
    *luong10: after 1/2 num train steps, we start halving the learning rate
     for 10 times before finishing.
--attention: Whether to add an attention layer, default is none. Here are the options:
    *luong
    *scaled_luong

*bahdanau
*normed_bahdanau

--num_units: Size of network, specifically size of a cell in the network.

Hyperparameter for nmt_hybrid:
--num_units_char: Size of cells of the characters network in the hybrid model.
In this case the size of the network, would be dependent on both this flag and the --num_units one.
The character vocabulary will be created automatically from the train data if needed

Inference:

For inference one needs the hparams file including the flags's values used while training. Moreover are the ckpt's model files needed. The best ones with the corresponding hparams files are to be found in the best_accuracy directory in the output directory (defined by --out) of the trained model.

Flags needed for inference:
--inference_input_file: Input file to infer labels for, each sentence in a separate line.
--inference_output_file: Output of the inference, i.e. the inferred labels.
--ckpt: The ckpt file of the trained model. For example
--ckpt=./nmt/best_accuracy=translate.ckpt-20000

To run the model for either training or inference the command is: *python3 -m nmt.nmt [flags]*
This command should be run from within the nmt_hybrid directory.

Please notice that if an hparams file already exist in the output directory, the flags in it will be regarded before the flags you incorporate into the command itself. I.e. if you want to give new flag values in the command which are different than the one in hparams, you should delete the file hparams in the output directory.