Here are some tips for you to build you method on our framework.

1. About the model

The name of our model is “Multi_gen”, and you can find it in “model.py”.

The generators are build in Line69-Line77 of “model.py”. For each generator, it gets an input of [batch_size, sequence_len, hidden_dim] and output [batch_size, sequence_len, 2]. Yes, we denote the mask of a token as a 2-dimensional vector. [batch_size, sequence_len, 2] is then fed into the gumbel-softmax layer (before a layernorm and a dropout). So, the mask of a token is [0,1] or [1,0]. And [0,1] means the token is selected as part of the rationale. So the rationale “z”=[batch_size, sequence_len, 2].

2. About the training steps

During training, we mainly use the “forward” function, which is in Line114 of “model.py”. “gen_logits” is the generators’ output, and it’s size is [number_generators, batch_size, sequence_len, 2], which denotes the probability of each token to be selected as part of a rationale. z_list=[number_generators, batch_size, sequence_len, 2] is the list of the token masks for all the generators. cls_logits_list is the predictor’s output for the prediction.

Then we should go to the “train_multi_gen” function in “train_util.py”. The rationales from different generators get their respective losses and we sum them up.

3. If you're still having trouble reproducing the code, we can set up a video conference to communicate.