Assignment 3 Q/A
CSE 538

1. apply() function

Question:

what is a good way to test the apply() function after coding? It seems it's only used in the tensorflow part of the code which we can't run until completing later portion

Answer:

In the genTrainExamples() function. After a sentence is parsed, if c.tree is equal to trees[i] then apply() is working properly.

Question:

I'm a bit confused with label part. Is the label 'L' and 'R' or is it the pos tag like 'num' and 'det'?

Answer:

transitions are something like:

```
L(root)
L(cc)
L(number)
L(ccomp)
L(possessive)
```

labels are:

```
root, cc, number, ccomp, etc, etc.
```

Question:

when applying the transition do we need to do checks, as in call canApply or hasOtherChild ? canApply I guess I can call anyways, so it isn't important. But hasOtherChild is the opposite of the single head condition, which checks that each node has a single head word. Not really sure how or where to use that fn.

Answer:

You don't have to do much check for apply(...) method. Get the two words from stack and do the actions accordingly to the transition.

Question:
I understand that we have to implement the algorithm mentioned in the lecture slides except for the action part. That is, we have to implement the condition part from slides and the action part from the paper.
The reason of my thinking is that I found "what" needs to be done on Left-arc, Right-arc, Shift, etc in Nivre paper but could not find anywhere "when" to perform these transitions.

Answer:
So, in the codes, the input of 'Apply' already has the transition(action) you should make and just do what the action should do.
The action you should make when you get into a specific configuration(the situation of stack, buffer and output arc) is the output(prediction) of the network. In training, of course you already have the ground truth and you know the actions. In test, you make the prediction of the action you should make and apply it and transfer to another configuration and then make another prediction and apply it

Question:
So basically, every time this function "apply" gets called, we will be given a transition which needs to be applied to the current configuration.
We check somehow what transition it is and then apply it to the configuration, according to the rules mentioned in the Nivre paper.

Please let me know if I am missing anything.

Answer:

Yes, that's it.

2. Order of applying Shift-Reduce Parsing

Question:
I know that we should use modules in Configuration.py to use in implementation of apply module. I wonder how should we find the order of the transitions? When we can both use shift and left-reduce which one should be used?

Answer:

Check the canApply method implementation. You have to do almost the same thing.

3. Feature Extraction

Question:

Should the output of getFeatures be structured the same as in the previous assignment, but using strings? A vector of named values (or list of values in the case of word-pair features)?
Additionally, what does the "wt" in s1.wt represent? Is there some operation to be performed on the word and tag of s1?

Answer:

You are required to implement the subheading 'Choice of Sw, St and Sl' under 3.1. of the Chen and Manning paper, using embeddings_array. The output of the method should be the input to the neural network. (embeddings with all features.)

Question:

in the feature array getFeatures returns, should I load all 18 words, then all 18 tags, then the 12 labels sequentially ?

Answer:

Yes

Question:

In DependencyParser.py, in function getFeatures(c), I'm not able to understand what is to be returned. In the paper, in Sw, there are 18 features to be added. Are those 18 features supposed to be actual words? Because c.getStack(0) is not returning the word but some index.
Or am I supposed to return the word embeddings?

Answer:

You should return an array with 18 features containing the ids for word/pos/label. Use the other modules in DependencyParser.py to convert the output of c.getStack(0) to word/index of the word.

Question:

In feature extraction, what should we do in cases where the stack has less than 3 elements or if the buffer has less than 3 elements. Do we simply ignore it and generate a feature vector of less than 48 elements?

Answer

One way to do this is using the existing APIs.
When you call getStack() with invalid stack position (i.e. stack is almost empty), then it will return Config.NONEXIST(which is -1) instead of the token index.

If you query the configuration to get the word of that token index, by calling getWord(), it will return Config.NULL(which is "NULL") since -1 is not a valid index.
Then, if you lookup your word dictionary to get its word ID, (getWordID), it will return whatever Word ID assigned to "NULL".

Question:
when we do the getLeftChild, getPOS and getLabel function calls, do we pass the token ids or the word ids?

Answer
You have to look into Configuration and DependencyTree classes to know what they do. You have to pass token index.

4. Helper function

Question:
Can we make extra helper functions in DependencyParser.py for modularity of code?

Answer1:

Please don't. If you need to, make inner functions.

```
def method_to_implement(...):

  def your_helper_func(...):
      ....

   your_helper_func()
```

5. Dealing with 3-D shape of train embeds

Question:
I am testing my forward pass and trying to wrap my head around the best way to handle the training embeddings. when we load them they would be of size (batch,tokens,embed) which doesn't work with multiplying into weight_input since that is only 2-D shape. So i am reshaping my embeddings to be of size (batch,tokens*embed) and this works well if you do matmul (embeddings, transpose(weights)) however I am getting a negative loss. Since working with the 3D shape of embeddings is hard for me to internalize can someone respond with if what I am doing is on the right path or if not a better way to think about it? I am not sure this is the error in my code to cause negative loss but the other aspects of my forward pass seem straight forward and not error prone

Answer:
I do believe that you are supposed to have inputs be size (batch, n_tokens*embedding_size). Just think of the feature matrix as a single vector and don't worry about its components. If you look carefully at the paper where it describes the cube activation function for the first time, it says the matrix sizes are (hidden_size, n_tokens*embedding_size) using different notation. Your negative loss must be coming from somewhere else, maybe your loss function.

6. Shape error

Question:
Any ideas as to what could be causing this error:

```
ValueError: Cannot feed value of shape (48,) for Tensor u'Placeholder_2:0', which
has shape '(10000, 48)'
```

on this line:

```
pred = sess.run(self.test_pred, feed_dict={self.test_inputs: feat})
```

during training at step 200? Also I start with a loss of -50?

Answer:
Your placeholder for test input should be (48,).
In terms of negative loss, there must be some errors in your program.

Question:
You mean that the shape for self.test_inputs should be [48]?

Answer:
Yes. For test, for each configuration, you generate corresponding features(18+18+12) and feed them to the training part. So it is 1*48 or (48,). And then look up for the corresponding embedding.

Question:
when declaring placeholders the dimensions of the train_inputs and test_inputs should be batch_size*(n_tokens*embedding_size) and 1*(n_tokens*embedding_size) correct? But what should be the dimensions for the train_labels?

Answer:

placeholder should be batch_size*n_token and 1*n_token. And then look up for embedding and reshape to batch_size*(n_tokens*embedding_size) and 1*(n_tokens*embedding_size). and now they are the input of the feed forward.
train_labels should be batch_size*numTrans

Question:

I am getting this error :
ValueError: Cannot feed value of shape (10000, 0) for Tensor u'Placeholder:0', which has shape '(480000,)'
at line :
_, loss_val = sess.run([self.app, self.loss], feed_dict=feed_dict)

placeholder for train_inputs is batch*n_tokens

Anyone facing this issue?

Answer:

It should be [batch, n_tokens] not [batch*n_tokens]

7. Reshaping tensors for forward pass

Question:

Are we allowed to reshape the embed tensor for forward pass or just the weights? I am having a problem with tf.matmul() since the ranks or dimensions of embed, test_embed and weight_inputs are all different and when I reshape weight_inputs for self.predictions, forward pass does not generalize to self.test_pred.

Answer:

In fact, you don't have to reshape the embed tensor nor weights.
Remember that matmul will allow you to matmul of rank 3 and rank 2, and you CAN change the order of weight and emb as long as you get what you have to get.

Question:

By order you mean by the position of the parameters? I've tried that and TF complains saying it must be of rank 2.

Answer:

Reshape embed to rank 2, [batch, -1]

8. right arc

Question:

for implementing right arc, we should add arc from a word in stack to the top of the buffer if top of buffer is a dependent of some word s in stack, how can we find the word s in stack? Is there any helper which tell us which words are dependent to which words?

Answer:

We are implementing arc-standard. So both words are from the stack.

9. forward_pass

Question:

Which part of the paper is to be implemented for forward_pass method?

Answer:

Map input layer to hidden layer and then map it to output layer. Basically the neural network part.

Question:

To go from input layer to hidden layer, we need 3 2-d matrices of size [hidden_size] [embedding_size*number of words or POS or labels respectively] and a bias of size [hidden_size] and initialize them randomly
To go from hidden to output layer, we need one matrix of size [number of transitions] [hidden_size] and initialize this randomly

I know we need to apply the simple cube activation function while going from input to hidden and then apply softmax to go from hidden to output

But am not quite getting the flow of how this is to be done so as to train our model?

Answer:

Think about what kind of predictions this neural network part of the system makes.
The output is the transitions, so we are training our model to predict the transitions.

10. Change the method parameters

Question:

Can we change the parameters which forward_pass() takes according to our best configuration?

Answer:

Comment out the non-default one with CLEAR comment of what it does.
Hence, comment out the forward_pass() for the best model, and leave the default one. In the header of the commented-out-but-best-model, write this is the best model version and etc etc.

11. format of function outputs/parameters

Question:

For example, could the output of getFeatures be either a single vector or a tuple of vectors? As long as our internal code handles it properly?

Or will you be testing it in such a way that forces our outputs/parameters to have a certain format? And if that is the case can you describe what the format should be for the parameters and outputs of our functions?

Answer:

As long as the subsequent codes can handle them, it will be okay.

12. Doubt regarding word and token IDS

Question:

I have a doubt regarding what we need to use in the apply, getFeatures functions. Are we supposed to use token ids for these or word ids? I'm currently performing the operations using token ids, but I think this might be wrong. Can anyone please clarify?

Answer:

Using token id to get the corresponding embedding, which is one inputs of the feed_forward function.

Question:

Aren't the word embedding based on the word ids? So shouldn't we be getting the word ids in the final features array?

Answer:

Yes. get the word id in feature generation part and in training you should by using word id to get corresponding embedding as the input of the neural network.

13. labelCount = [Config.NULL, rootLabel]

Question:
why don't we consider unknown while making labelDict like we did in posdict and worddict

Answer:
There is a 'NULL' in labels might have the same meaning with 'UNK' in wordDict and POSDict

14. Train_inputs

Question:
what will be our train inputs ? will it be a concatenation of xw, xt, xl? and we have to look up the embeddings of these and then pass it to the forward_pass? and then in the forward_pass we will have to separate the xw, xt, xl and then multiply them with the respective weights?

Answer:

inputs will be concatenation of features. The rest is correct, although, you wouldn't need to separate xw, xt and xl.

15. Error while training

Question:
Did anyone face this error?

ValueError: invalid literal for long() with base 10: 'ROOT'

Answer:

Not sure, but could be the case that you are not supposed to pass word in string but its id.

16. getLabelID

Question:

Is it OK or even necessary to make a change in getLabelID(), because the else block returns labelDict[Config,UNKNOWN] but this key is not set in genDicts? I changed the return to labelDict[Config.NULL] , not sure if that's correct.

Answer:

In fact, looking up Config.UNKNOWN for labelDict is incorrect since it is not added during the initialization.

BUT, if you use the related methods correctly, getLabelID() should not be called for unknown label.

The arguments of this method should be either known label strings or "NULL"(=Config.NULL)

17. 'UNK' key error

Question:

I am get UNK key error when I do getLabelID(c.getLabel(i)) i understand this is because getLabelID function which returning Config.UNKNOWN
can anyone hint at the mistake i am making or which part of the code is responsible for this.

Answer:

You might be calculating labels for all the 18 elements, which you should not.
You are supposed to calculate labels only for the 12 elements.
I think you should be fine if you fix this.

18. Error in train function

Question:

I didn't change the train() function also I didn't use any kind of changing the dimension of training or labels, but I get this error while it wants to evaluate the model on validation set (after 200 steps):
File "DependencyParser.py", line 194, in train
if pred[0, j] > optScore and parsing_system.canApply(c, parsing_system.transitions[j]):
IndexError: index 1 is out of bounds for axis 1 with size 1
anyone has any idea?

Answer:

check the shape of your prediction. It seems to be too small.

19. Constant accuracy and loss with two hidden layers

Question:

When I implement the two hidden layer code my loss suddenly gets a spike at 300 th step and after this the loss decreases and then remains constant for all other steps also the accuracy remains constant after this.
Is something wrong with the implementation or such decrease or increase in loss are okay considering overall the accuracy in last step was better from 100th step and loss less than the 100th step loss.

Answer:

Try decreasing learning rate. Your network may be diverging in initial steps.

20. Implementation of two hidden layers

Question:

To implement the model with two hidden layers, we will have to send an extra parameter while calling forward_pass as the weight for layer 2. So can we do changes in the code already provided by you?
If not should we send 2 times the dimension of weight input and then in forward_pass break it into two different weight variables for each of the layer?

Answer:

You can make another function if necessary.

Question:

It is ok to make changes in the given structure as if I make a new function, I will have to send my test_pred to the new function instead of self.forward_pass. So will that be ok?

Answer:

Basically, yes. If you make changes to carry out the experiments, and if it is necessary, then it is okay.
But remember that you are not changing the structure for good. Even for the experiments, you'd need to switch back and forth.

21. Multiple Hidden Layers

Question:

Can different hidden layers have different numbers of neurons?

Answer:

Yes

22. Combined hidden layers

Question:

For the cubic nonlinearity experiment, the assignment pdf says "Have three separate parallel hidden layers, one
for combining word embeddings, one for POS, and one for deps." For the remaining experiments, can we use a combined hidden layer for words, POS and deps?

Answer:

Yes

23. Cross entropy loss

Question:

I wanna know why we have -1 in labels and we still use cross entropy. The ground truth labels are not a valid probability distribution.

Answer:

So you have to handle it a little bit differently. In the paper, it says,

```
A slight variation is that we compute the softmax probabilities only among the
feasible transitions in practice.
```

Question:

It seems like to resolve this, the line 'label.append(-1.)' should be replaced with 'label.append(0.)' But if so then what was the point of that line in the first place?

Answer:

I didn't mean to replace the code.
When you compute the softmax/loss, you can use masks for cases with feasible choices (1,0) and correct choices(1), and use them while you compute the loss.

Answer:

label with -1 means this is not a valid transition, which means it is not a feasible transitions.
So when you calculate the softmax you should avoid those.

Question:
While computing the loss, are we supposed to ignore the predictions for non feasible transitions?

Answer:
basically, yes.

Question:
can we use tf.nn.softmax_cross_entropy_with_logits and tf.nn.l2_loss

Answer:
You can use them.

Question:
what should we pass as inputs to the functions? Should we pass the train_inputs to cross_entropy and the weights to the l2_loss?

Answer:
You can refer to tf's doc and the cited paper for how-to and what-to-pass.

24. Loss function explanation

Question:
Is there any doc/note that has a simplified explanation for the loss function calculation like the one we had for explaining the dependency parser model?

Answer:

The first term is just cross-entropy loss. While you could implement this yourself, tensorflow has a built in function that will compute the loss for you. If you give it a batch of predictions and a batch of labels, it will return the loss for each prediction, and you should take the average across those values.
As for the normalization term, you can compute ||theta|| by taking the sum of all the normalized values of the parameter matrices (embeddings, weights, biases, etc.) Again, tensorflow has functions to get the normalized value of a matrix.

25. Loss value remaining constant

Question:

My loss value is remaining constant at a 4.51 with default parameters. Any idea what can be the issue?

Answer:

Try changing your std deviation during initialization. A small number might affect multiplication. At least empirically that's what I realized.

26. Expected loss

Question:

Can you please tell what the expected loss would be at iteration 0 and at the end of all iterations so that we get some directions on the correctness of our implementation?

Answer:

Average loss at step 0 : 3.89940023422
Average loss at step 100 : 2.93363507867
Average loss at step 200 : 1.61602276564
Average loss at step 300 : 1.08881413758
Average loss at step 400 : 0.786221237183
Average loss at step 500 : 0.640354695916
Average loss at step 600 : 0.536588197351
Average loss at step 700 : 0.482249063849
Average loss at step 800 : 0.430445612073
Average loss at step 900 : 0.405791639388
Average loss at step 1000 : 0.373037184477

Question:

Is it loss or is it log loss? My training error is much higher than this! What may the problem be? For initialization I used random normal with std= 0.01 is that fine? When I change it the error changes a lot, even it becomes nan for some values. So I think the model is so sensitive to initial values. What is the best starting value?

Answer:

It's log loss. If you look up, there will be many docs on the initialization values. TF's default one seemed to work okay too.

27. Negative loss

Question:

I am getting negative loss for the training data from starting of the training with all the default configuration.
Is this expected or could be some code issue in my code?

Answer:

No it is not expected. Check your loss code first.

28. Expected Accuracy

Question:

Can you please tell how much accuracy is expected ?

Answer:

I don't have expected accuracy. Explore as much as you can.

29. Interpret Accuracy results

Question:

Can you please provide some explanation of the numbers printed by DependencyParser.py
Clarification on the meanings of UAS, UASnoPunc, LAS LASnoPunc, UEM, UEMnoPunc would be great!

Answer:

- UAS stands for unlabeled attachment scores, and
- LAS stands for labeled attachment scores
- noPunc means no punctuations since they were taken out when computing the accuracy.
- UEM stands for unlabeled sentence level exact match.
- ROOT means correct ROOT rate.

30. check overfitting

Question:

If my understanding is correct if you increase number of iterations you are going to get better accuracy but it is likely that your neural net has also learnt noise and performing better only on train data and won't perform that nice on the test data.
How to ensure I have not done that ?

Answer:

The dev set is there for that purpose. If you are overfitting, your performance on the training data would increase while the perf on the dev data would decrease.

31. Regularization of Bias Term

Question:

Usually in machine learning problems, we do not regularize the bias term. In this paper it is stated to regularize all of the parameters including the bias term. Should be regularize the bias term also according to the paper method or we should just follow what machine learning people do?

Answer:

Follow the paper. You are welcome to do experiments without the bias term.

32. DependencyParser.py Submission

Question:

Since we are experimenting with multiple models, I was wondering which version of DependencyParser.py should we submit ? The one resulting in the best validation score, or the one having the same model as described in the paper ?

Answer:

If you have one Dependencyparser.py, comment out non-default models.
Then, in the header of each commented out part, WRITE COMMENTS on
- what does the commented out block do
- which model it was used
- etc.

If you have multiple files, then name the non-default models with their keywords and LIST THEM in readme file with descriptions.

33. which 'result_test.conll' to submit

Question:

Do we need to submit the result_test.conll for the best model or with the model with default configuration?

Answer:

results_test.conll : generate from the best model
code: default/best/ and other codes should be there.

34. Changes in Config.py

Question:

Are we allowed to make changes in Config.py for parameters like learning rate or lambda values?

Answer:

You can change the values of the parameters as you wish.

Other Questions

If you didn't find your question here, you can ask it in Piazza, meet with TAs during office hours or email me: mbastan@cs.stony brook.edu

NOTES

- IF YOU ARE GOING TO USE YOUR OWN EMBEDDINGS:

You have to match "load_embeddings" method in *DependencyParser.py*
If you want to use your own embeddings, you'd need to re-train your model to meet the embedding size of 50, if you haven't done it already.
from:

```
def load_embeddings(filename, wordDict, posDict, labelDict):
dictionary, word_embeds = pickle.load(open(filename, 'rb'))
```

to:

```
def load_embeddings(filename, wordDict, posDict, labelDict):
dictionary, _ ,word_embeds = pickle.load(open(filename, 'rb'))
```

- This presentation of the main paper may be useful for understanding it:

https://www.youtube.com/watch?v=MLAcBv5dLEs

- You might want to increase the max_iter, to train the model further. 1001 was relatively small number. (I won't tell you what is the good iteration count, so don't ask.)

- For those of you who saved the training examples, please DO NOT include it. Probably you won't be able to, but, I am telling you just in case.

- Submitting README is optional. But having it could work in your favor since it could help to understand your code.