Building end-to-end QA model with reward based objective

TEAM REINFORCE (Prashant and Eti)

BiDAF Implementation(dynet)

213 lines (152 sloc) 7.71 KB Raw Blame History

```
#BIDAF Implementation
from __future__ import division
import numpy as np
import dynet as dy
from time import time
import json

class BiDAF():
    def __init__(self, pc, word_emb_dim, hidden_dim, load_model=False):
        self.word_emb_dim = word_emb_dim
        self.hidden_dim = hidden_dim

        #self.lookup_table = pc.add_lookup_parameters((100000, word_emb_dim)) # or u
        if not load_model:
            self.W_ss_ = pc.add_parameters((1, 3*hidden_dim))
            self.b_ss_ = pc.add_parameters((1))
            self.W_p1_ = pc.add_parameters((1, 5*hidden_dim))
            self.b_p1_ = pc.add_parameters((1))
            self.W_p2_ = pc.add_parameters((1, 5*hidden_dim))
            self.b_p2_ = pc.add_parameters((1))

            self.contextLSTM = dy.VanillaLSTMBuilder(1, word_emb_dim, hidden_dim, pc
            self.queryLSTM = dy.VanillaLSTMBuilder(1, word_emb_dim, hidden_dim, pc)
            self.modellingLSTM = dy.VanillaLSTMBuilder(1, 4 * hidden_dim, hidden_dim
            self.outputLSTM = dy.VanillaLSTMBuilder(1,hidden_dim,hidden_dim, pc)

        else:
            self.W_ss_, self.b_ss_, self.W_p1_, self.b_p1_, self.W_p2_, self.b_p2_,


    def similarity_score(self,h,u):
        concat = dy.concatenate([h,u,dy.cmult(h,u)],d=0)
        score = self.W_ss * concat + self.b_ss

        return score.scalar_value()

    def c2q_attention(self, sim_matrix, query_states):
        attention_vector = dy.softmax(sim_matrix)
        c2q = [dy.esum([b * dy.select_cols(attention_vector,[i])[j]  for j,b in enum

        return c2q

    def q2c_attention(self, sim_matrix, context_states):
        attention_vector = dy.softmax(dy.max_dim(sim_matrix,d=1))
        weighted_vectors = [b * a for a,b in zip(attention_vector, context_states)]

        return [dy.esum(weighted_vectors) for _ in range(self.T)]

    def similarity_matrix(self, context_states, query_states):
        sim_matrix = np.zeros((self.J,self.T))
        for i in range(len(query_states)):
            for j in range(len(context_states)):
                sim_matrix[i][j] = self.similarity_score(context_states[j],query_sta

        return dy.inputTensor(sim_matrix)

    def span_scores(self,combined_input1, combined_input2):
        s1 = [self.W_p1*combined_input1[i] + self.b_p1 for i in range(self.T)]
        s2 = [self.W_p2*combined_input2[i] + self.b_p2 for i in range(self.T)]
#           p2 = self.W_p2*dy.inputTensor(combined_input2) + self.b_p2

        p1 = np.zeros(self.T)
        p2 = np.zeros(self.T)

        for i in range(self.T):
            p1[i] = s1[i].scalar_value()
            p2[i] = s2[i].scalar_value()
        return p1, p2


    def complete_forward_pass(self, inputs):
```

Baseline Model Experiment Results

1. ***SQuAD***

	Accuracy(EM)(Ours)	Accuracy(EM)(Original)
With character embeddings / byte encodings		67.7
No Bidirectional LSTM + No char embeddings	32.4	-
Bidirectional LSTM + No char embeddings	48.0	65.0
Reducing Vocabulary size		

Error Analysis

Question	Predicted	Gold Answer	Reason for Error
Which articles of the Free Movement of Workers Regulation set out the primary provisions on equal treatment of workers?”	1 to 7	articles 1 to 7	-
What year did BSkyB acquire Sky Italia ?	2014	2014	Wrong span
when did French and Indian war ended ?	1754–1763	1763	Splitting on space

Things to do

1. Byte pair encoding(In process)
2. Removing irrelevant words from the vocabulary
3. Bi-Directional LSTM(✔)
4. Batching(✔)
   - Masking and Padding to make sentences and queries of equal length
   - Calculate loss based on normalized batch and mask it to remove losses corresponding to padded words
   - Sorting of sentences (to reduce variance in sentence lengths in a mini-batch)

Timeline

Date	Task
~~03/05 - 03/12~~	~~Implementation and training of own BiDAF model in dynet(delayed 1 week)~~
03/13 - 03/30	Running the model on SearchQA and improving upon the existing baseline model
04/01 - 04/15	Error Analysis and further improvement