Topic: Question Answering
Homework Assignment

CPSC 503 Computational Linguistics I

Authors: Sang-Wha Sien and Ramya Rao Basava

1 Short answer questions (Total: 20 points)

Question 1.1 (2 points)

List two applications of QA in the real world.

ANSWER: In any of the areas mentioned below:

- Information extraction from documents or web.
- Chat-bots: For example as customer service agents, Google assistants which recognize voice and answer questions.
- Search engines
- Summarization

Question 1.2 (2 points)

Give two differences between information retrieval and knowledge based methods for question answering.

ANSWER:

	IR	**KB**
Domain	More suitable for open domain corpora like the web.	Suitable only for restricted domains like databases.
Query formulation	No NLP related information is needed to obtain the query. Usually simpler methods like question reformulation or rephrasing are enough.	Semantic parsers are usually used to map the question to a query language.

Question 1.3 (2 points)

What is one advantage and disadvantage in combining multiple information sources for question answering?

ANSWER:

Advantage: Makes it possible to construct DeepQA systems like IBM Watson, which provide an extremely powerful solution for the question answering task.

Disadvantage: Lots of information is available, most of which could be redundant, for answering the question.

Question 1.4 (5 points)

What is the main characteristic of the BERT model that differentiates it from its previous work? Provide brief comparison with two relevant previous works.

ANSWER:

The BERT model pre-trains bidirectional representations from text, based on conditioning both left and right context in all layers of the transformer. It is the first model to incorporate both left and right context (bidirectional), unlike the previous methods. The previous methods for example ELMo [4] used separate LSTMs for including context from left-to-right and right-to-left, and then combined the resulting embeddings suitably for downstream tasks. Other methods like Open AI GPT [5] used only left-to-right unidirectional representations.

Question 1.5 (2 points)

Using the Quarc set of heuristic rules for WHO as an example, come up with a set of rules for WHEN. For Quarc, the scoring mechanism is as follows: clue (+3), good_clue (+4), confident (+6), slam_dunk (+20) and WordMatch(Q,S), which is the number of matches between question and answer candidate pairs. You can assume that there are semantic classes TIME, LOCATION, HUMAN, DATELINE. For other assumptions, please list before your answer. Explain your rationale.

ANSWER:
There are a variety of answers to this question and points should depend on the reasonableness of rule's rationale. Here is one possible answer:

(i) If contains(S, TIME)
Then Score(S) += good_clue + WordMatch(Q,S)

(ii) If contains(S, {yesterday, today, tomorrow})
Then Score(S) += good_clue + WordMatch(Q,S)

(iii) If contains(Q, the last) and contains(S,{first, last, since, ago})
Then Score(S) += slam_dunk

(iv) If contains(Q, {start, begin}) and contains(S, {start,begin,since})
Then Score(S) += slam_dunk

Rationale: If the sentence to check has any TIME format which can include months, years, days, or time of day, then the sentence should get the score of "good_clue" and whatever result from WordMatch. These are not "slam_dunks" because we have to be certain the sentence actually matches the question, hence the need for WordMatch. Any other sentence that matches the question in terms of time duration words is considered a "slam_dunk."

Question 1.6 (2 points)

For the following questions, craft their equivalent logical forms using lambda expressions. Then, describe how they can be broken down into smaller units and show their smaller logical forms as well. Finally, describe how they can be generalized to rules to process similar questions. Refer to the text's section on knowledge based methods and the following paper for more information:

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence (UAI'05). AUAI Press, Arlington, Virginia, USA, 658–666.

(i) *What is the least populous state bordering New York?*
ANSWER:
Logical form: $argmin(\lambda x.state(x) \wedge borders(x, new\ york) \wedge \lambda x.popsize(x))$
What states border New York? $\lambda x.state(x) \wedge borders(x, new\ york)$ which can be generalized to: What states border $--$? $\lambda x.state(x) \wedge borders(x, --)$
What is the least populous state? $argmin(\lambda x.state(x), \lambda x.popsize(x))$
Therefore the question is generalized to: What is the least populous state bordering New York? $\rightarrow$ $argmin(\lambda x.state(x) \wedge borders(x, --) \wedge \lambda x.size(x))$

(ii) *What is the biggest grossing film after 2010?*
ANSWER:
Logical form: $argmax(\lambda x.film(x) \wedge after(x, 2010) \wedge \lambda x.gross(x))$
What are the films after 2010? $\lambda x.film(x) \wedge after(x, 2010)$ which can be generalized to: What are the films after $--$? $\lambda x.film(x) \wedge after(x, --)$
What is the biggest grossing film? $argmax(\lambda x.film(x), \lambda x.gross(x))$
Therefore the question is generalized to:
What is the biggest grossing film after 2010? $\rightarrow$ $argmax(\lambda x.film(x) \wedge after(x, --) \wedge \lambda x.gross(x))$

Question 1.7 (5 points)

Name a real world problem where the application of QA can be useful. Do some research first to see it has not been done before. Explain why QA would be needed and how it can be applied. Limit to 100 words.

ANSWER:

There are a variety of answers to this question and points should depend on the reasonableness of the rationale and novelty of the idea. The point of this exercise is to give students an opportunity to scan a broad survey of QA research and possibly carve out a project in the research space. Here is one possible answer: Although there has been research done on QA systems where the questions and answers use a different language from the document [3], there has yet to be sufficient research on the different contexts that such QA translations may be needed. One idea could be that if I went on a trip where I did not know the language, I might want to understand what is written in the signage, public notices, or even menus. Having the ability to take pictures of the writing and ask questions about it would make it much easier to travel. This would involve computer vision much like VQA.

2 Paper summaries (Total: 20 points)

The summaries should be maximum one page length and include the following points:

- *Motivations for the work*
- *Proposed solution including strengths and weaknesses of the method*
- *How evaluation is carried out (evaluation metrics, experiments, etc.)*
- *Contributions of the work*

Write summaries for the papers listed below: (10 points each)

(i) *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding by Devlin et al. 2018 [2].*
(ii) *VQA: Visual Question Answering by Agrawal et al. 2015 [1].*

ANSWER:

Paper (i): BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding by Devlin et al. 2018 [2]

The paper presents BERT: Bidirectional Encoder Representations from Transformers which is a language representation model. This model pre-trains deep bidirectional representations from text in documents, by taking into consideration both left and right context in all layers. It provides state-of-the-art results and is one of the landmark papers in NLP.

Motivations:

Pre-training neural networks has been successfully used in machine learning to transfer learning, where the trained layers of the network are used for solving new or other problems. The pre-trained network is based on a large number of examples. In NLP, this can be done by pre-training word embeddings and using for other NLP tasks like next sentence prediction etc. Previously this was done using unidirectional approaches for text representations, which had context from either left to right, or right to left, or a combination of these methods. In this paper the motivation is to pre-train deep bidirectional representations by taking context in both left and right directions.

Proposed solution:

The proposed BERT method does pre-training using two unsupervised tasks. In the first task 15% of the word tokens are masked and then during the training, these masked words in the sentences are predicted. In the second task the model is trained to predict the next sentence, based on a given corpus. In the next part which is fine-tuning, BERT can be used for any of the downstream tasks like Question Answering, classification, etc. In this part all parameters are fine-tuned. The paper gives a state-of-the-art method for including context in word embeddings using deep bidirectional representations. On the other hand, one of the weakness might be that the masked word tokens, masked during training, are not seen in the fine-tuning phase.

Evaluation:

Experiments are conducted using standard benchmark datasets like GLUE, SQuAD v1.1, SQuAD v2.0. The method gives excellent results.

Contributions:

The major contributions of the paper are as follows:

- The importance and effectiveness of bidirectional pre-training for language representations is demonstrated.
- Validate that pre-trained representations are important to complex architectures for neural language models.
- This method advances the state-of-the-art in Neural Language Models, and is one of the milestone papers in this research area.

Paper (ii): VQA: Visual Question Answering by Agrawal et al. 2015 [1]

This paper presents Visual Question Answering (VQA) where given an image and associated natural language question related to the image, the method provides an answer in natural language. It combines the areas related to Computer Vision (CV), Natural Language Processing (NLP) and Knowledge Representation & Reasoning (KR). It has a lot of applications, like helping visually challenged people.

Motivations:

Previous methods which researched VQA usually had very restricted domain with small datasets. Answers to questions could be found from different object categories in the image like color, attributes, relationships between objects, etc. In contrast the proposed method emphasizes on open-ended and free form question answering (QA), similar to the way humans provide questions and answers from images. Given an image, any question can be asked for which the method provides an answer. It also takes into account diversity knowledge and reasoning unlike the previous methods. Apart from open-ended questions, multiple-choice questions are also considered.

Proposed solution:

The proposed method uses two channels to get the embeddings for the image and the question. One channel is for image processing and the other for NLP for the question. For the image part VGGNet framework is used, which is a very popular deep neural network for image processing. The NLP is done with LSTM neural model, either with one hidden layer or two. The resulting embeddings from both the channels are combined and passed through a softmax classifier, which selects the answer based on highest activation, from a set of output answers.

Evaluation:

Images from the Microsoft Common Objects in Context (MS COCO) dataset is used for training and validation. In addition, the authors create an abstract scenes dataset, which will explore higher level reasoning using VQA. A number of baseline methods are used to check the results. The proposed method with two hidden layer LSTM + VGGNet gives the best accuracy results compared to all the methods, both for open-ended questions and multiple-choice questions. Other analysis like, where scene features are more helpful, changing vocabulary size or filtering the dataset are also given to show how they influence the results. The evaluation metric used is accuracy and the authors do a robust job to extract and explain the measures taken for human comparison, but maybe more study needs to be performed to try other different metrics to get a better analysis specific to VQA.

Contributions:

The major contributions of the paper are as follows:

- The paper introduces the open-ended, free-form visual question answering task, where an image and natural language question is given as the input and a natural language answer is obtained as the output.
- A deep neural network based method which combines areas of CV and NLP is proposed, which gives highly accurate results compared to baseline or other methods.
- This is one of the first works which formally introduces VQA at a large scale.

3 Coding problem (Total: 10 points)

Please see the additional jupyter notebook provided (python code), for running the code for this problem. The pdf of the jupyter notebook with the output and answers is attached below (on the next page).

Note: The formatting is best when viewed in the jupyter notebook.

References

[1] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. VQA: Visual Question Answering. In *International Conference on Computer Vision (ICCV)*, 2015.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding, 2018.

[3] Ekaterina Loginova, Stalin Varanasi, and Günter Neumann. Towards multilingual neural question answering. In András Benczúr, Bernhard Thalheim, Tomáš Horváth, Silvia Chiusano, Tania Cerquitelli, Csaba Sidló, and Peter Z. Revesz, editors, *New Trends in Databases and Information Systems*, pages 274–285, Cham, 2018. Springer International Publishing.

[4] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations, 2018.

[5] Alec Radford. Improving language understanding by generative pre-training. 2018.

503 Project Homework Assignment

Problem III (10 points)

Overview

In this problem we are going to explore QA using python, with some basic embedding methods and k nearest neighbours (KNN) similarity measure. We are assuming that the question and the answer are semantically similar in some way.

Packages to install

nltk:
```
conda install nltk
```

gensim:
```
conda install -c conda-forge gensim
```

Input

The code below gives a passage and a query question. **First run the code below.**

In [1]:

```
import numpy as np
import string

import nltk
from nltk.corpus import stopwords
from nltk.tokenize import sent_tokenize, word_tokenize
```

In [2]:

```
# REFERENCE: From Wikipedia
# Below is a passage about Disneyland
passage = "Disneyland Park, originally Disneyland, is the first of two theme parks built a
t the Disneyland Resort in Anaheim, California, opened on July 17, 1955. It is the only th
eme park designed and built to completion under the direct supervision of Walt Disney. It
 was originally the only attraction on the property. Its official name was changed to Disn
eyland Park to distinguish it from the expanding complex in the 1990s. It was the first Di
sney theme park. Walt Disney came up with the concept of Disneyland after visiting various
amusement parks with his daughters in the 1930s and 1940s. He initially envisioned buildin
g a tourist attraction adjacent to his studios in Burbank to entertain fans who wished to
 visit; however, he soon realized that the proposed site was too small. After hiring a con
sultant to help him determine an appropriate site for his project, Disney bought a 160-acr
e (65 ha) site near Anaheim in 1953. Construction began in 1954 and the park was unveiled
 during a special televised press event on the ABC Television Network on July 17, 1955. Si
nce its opening, Disneyland has undergone expansions and major renovations, including the
 addition of New Orleans Square in 1966, Bear Country (now Critter Country) in 1972, Micke
y's Toontown in 1993, and Star Wars: Galaxy's Edge in 2019. Opened in 2001, Disney Califor
nia Adventure Park was built on the site of Disneyland's original parking lot. Disneyland
 has a larger cumulative attendance than any other theme park in the world, with 726 milli
on visits since it opened (as of December 2018). In 2018, the park had approximately 18.6
 million visits, making it the second most visited amusement park in the world that year,
 behind only Magic Kingdom, the very park it inspired."
print(passage)
```

```
Disneyland Park, originally Disneyland, is the first of two theme parks built
at the Disneyland Resort in Anaheim, California, opened on July 17, 1955. It
is the only theme park designed and built to completion under the direct supe
rvision of Walt Disney. It was originally the only attraction on the propert
y. Its official name was changed to Disneyland Park to distinguish it from th
e expanding complex in the 1990s. It was the first Disney theme park. Walt Di
sney came up with the concept of Disneyland after visiting various amusement
parks with his daughters in the 1930s and 1940s. He initially envisioned buil
ding a tourist attraction adjacent to his studios in Burbank to entertain fan
s who wished to visit; however, he soon realized that the proposed site was t
oo small. After hiring a consultant to help him determine an appropriate site
for his project, Disney bought a 160-acre (65 ha) site near Anaheim in 1953.
Construction began in 1954 and the park was unveiled during a special televis
ed press event on the ABC Television Network on July 17, 1955. Since its open
ing, Disneyland has undergone expansions and major renovations, including the
addition of New Orleans Square in 1966, Bear Country (now Critter Country) in
1972, Mickey's Toontown in 1993, and Star Wars: Galaxy's Edge in 2019. Opened
in 2001, Disney California Adventure Park was built on the site of Disneylan
d's original parking lot. Disneyland has a larger cumulative attendance than
any other theme park in the world, with 726 million visits since it opened (a
s of December 2018). In 2018, the park had approximately 18.6 million visits,
making it the second most visited amusement park in the world that year, behi
nd only Magic Kingdom, the very park it inspired.
```

In [3]:

```
# Getting the sentences from the passage
passage_sentences = sent_tokenize(passage)
passage_sentences
```

Out[3]:

```
['Disneyland Park, originally Disneyland, is the first of two theme parks bui
lt at the Disneyland Resort in Anaheim, California, opened on July 17, 195
5.',
 'It is the only theme park designed and built to completion under the direct
supervision of Walt Disney.',
 'It was originally the only attraction on the property.',
 'Its official name was changed to Disneyland Park to distinguish it from the
expanding complex in the 1990s.',
 'It was the first Disney theme park.',
 'Walt Disney came up with the concept of Disneyland after visiting various a
musement parks with his daughters in the 1930s and 1940s.',
 'He initially envisioned building a tourist attraction adjacent to his studi
os in Burbank to entertain fans who wished to visit; however, he soon realize
d that the proposed site was too small.',
 'After hiring a consultant to help him determine an appropriate site for his
project, Disney bought a 160-acre (65 ha) site near Anaheim in 1953.',
 'Construction began in 1954 and the park was unveiled during a special telev
ised press event on the ABC Television Network on July 17, 1955.',
 "Since its opening, Disneyland has undergone expansions and major renovation
s, including the addition of New Orleans Square in 1966, Bear Country (now Cr
itter Country) in 1972, Mickey's Toontown in 1993, and Star Wars: Galaxy's Ed
ge in 2019.",
 "Opened in 2001, Disney California Adventure Park was built on the site of D
isneyland's original parking lot.",
 'Disneyland has a larger cumulative attendance than any other theme park in
the world, with 726 million visits since it opened (as of December 2018).',
 'In 2018, the park had approximately 18.6 million visits, making it the seco
nd most visited amusement park in the world that year, behind only Magic King
dom, the very park it inspired.']
```

In [4]:

```
# Below is the query question for which we are trying to get the answer from the passage
question = "When was Disney California Adventure Park opened?"
```

Exercise 1: CountVectorizer and KNN

Note: This exercise is done for you as an example.

Using CountVectorizer which is based on word counts and KNN based on cosine similarity, to get the top 2 answers for the question. For more information regarding CountVectorizer see here (https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html).

In [5]:

```
from sklearn.feature_extraction.text import CountVectorizer
vec = CountVectorizer(min_df=5, max_df = 100, max_features=1000)
vec.fit(passage_sentences)
passage_fit = vec.transform(passage_sentences)
question_fit = vec.transform([question])

from sklearn.neighbors import NearestNeighbors
nn = NearestNeighbors(n_neighbors=2, metric='cosine') # for metric: Euclidean distance is
 the default
nn.fit(passage_fit);

dists, near_neigh = nn.kneighbors(question_fit)
print("Top 2 answers:")
for index in np.squeeze(near_neigh):
    print(passage_sentences[index])
```

```
Top 2 answers:
It was the first Disney theme park.
Opened in 2001, Disney California Adventure Park was built on the site of Dis
neyland's original parking lot.
```

Comment: The second neighbour is the correct answer, still the first one is not the right one!

Exercise 2: tf-idf and KNN (2 points)

Use TfidfVectorizer (https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html) instead of CountVectorizer.

- Use the default hyperparameters.
- Get the top 2 answers for the query question (similarly as done in Exercise 1).
- Breifly comment on the results.

BEGIN ANSWER

In [6]:

```
from sklearn.feature_extraction.text import TfidfVectorizer
vec = TfidfVectorizer()
vec.fit(passage_sentences)
passage_fit = vec.transform(passage_sentences)
question_fit = vec.transform([question])

from sklearn.neighbors import NearestNeighbors
nn = NearestNeighbors(n_neighbors=2, metric='cosine') # Euclidean distance is the default
nn.fit(passage_fit);

dists, near_neigh = nn.kneighbors(question_fit)
print("Top 2 answers:")
for index in np.squeeze(near_neigh):
    print(passage_sentences[index])
```

```
Top 2 answers:
Opened in 2001, Disney California Adventure Park was built on the site of Dis
neyland's original parking lot.
It was the first Disney theme park.
```

Comment: The first neighbour is the correct answer.

END ANSWER

Using word embeddings

Lets use word embeddings for the QA task:

- We are going to use fastText word embeddings.
- Go to this website here (https://fasttext.cc/docs/en/pretrained-vectors.html) and download 'English: bin+text'. Note that this is a large file ~9.64 GB, so it will take a while. Extract the zip file after downloading and place it in the same folder as your python notebook.
- Run the following code to load and test the pre-trained word embedding model.

Reference for fastText

- Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information.Transactions of the Association for Computational Linguistics, 5:135–146, 2017.
- License (https://creativecommons.org/licenses/by-sa/3.0/). No changes were made to any files used.

In [7]:

```
# Note: this step will take a while to run
from gensim.models import KeyedVectors
model = KeyedVectors.load_word2vec_format('wiki.en/wiki.en.vec')
```

In [8]:

```
print('Vocabulary size: ', len(model.vocab))
```

```
Vocabulary size:  2519370
```

In [9]:

```
print('Length of the embedding: ',model.vector_size)
```

```
Length of the embedding:  300
```

In [10]:

```
# Example: Finding similar words
wordsim = 'friend'
print(model.most_similar(wordsim))
```

```
[('friends', 0.7834614515304565), ('acquaintance', 0.7800023555755615), ('sch
oolmate', 0.7561026811599731), ('friend/acquaintance', 0.7354345321655273),
('reacquaintance', 0.7353211641311646), ('colleague', 0.7343807220458984),
('girlfriend', 0.7310606241226196), ('classmate', 0.7261524200439453), ('frie
nd–', 0.7244135141372681), ('friend–and', 0.7227555513381958)]
```

Exercise 3: Using pre-trained word embeddings and KNN (Total: 8 points)

We are going to construct embeddings for sentences, so that the question embedding and embedding for each sentence in the passage, can be compared through KNN similarity.

Part 1: (6 points)

- Write a function which takes a sentence as input and creates a "sentence embedding" by taking the average of the word embeddings in the sentence.
- The embeddings for the words should be constructed from the above fastText model.
- Make sure the input sentences are pre-processed to remove Stop words, punctuations, etc. as you see fit.

Part 2: (2 points)

- Next construct the sentence embeddings for the question and each sentence in the passage using the function you created in Part 1.
- Use KNN on these embeddings to find the top 2 answers for the question.
- Breifly comment on your results and compare to the result obtained from Exercise 2

BEGIN ANSWER

In [11]:

```
# Part 1
def sentence_embedding(sentence, modelfunc = model):

    emb_len = modelfunc.vector_size

    stopWords = list(set(stopwords.words('english')))
    punc = string.punctuation
    stopWords += list(punc)
    #stopWords.extend(['``',''', '`','"',"””"])

    sentence_cleaned = []
    sentence_token = word_tokenize(sentence)
    for token in sentence_token:
        token = token.lower()
        if token not in stopWords:
            sentence_cleaned.append(token)

    #print(sentence_cleaned)

    sentence_len = len(sentence_cleaned) #No. of words in the sentence
    average_embedding = np.zeros(emb_len, dtype='float64')

    for token in sentence_cleaned:
        if (sentence_len>0):
            try:
                temp = modelfunc[token]
            except:
                continue
            average_embedding = np.add(average_embedding,temp)
        else:
            continue
    if (sentence_len>0):
        average_embedding = np.divide(average_embedding, sentence_len)

    return average_embedding
```

In [12]:

```
# Part 2
question_avg_embed = sentence_embedding(question)
passage_avg_sentence_embeddings = [sentence_embedding(line) for line in passage_sentences]

from sklearn.neighbors import NearestNeighbors
nn = NearestNeighbors(n_neighbors=2, metric='cosine') # Euclidean distance is the default
nn.fit(passage_avg_sentence_embeddings);

dists, near_neigh = nn.kneighbors([question_avg_embed])
print("Top 2 answers:")
for index in np.squeeze(near_neigh):
    print(passage_sentences[index])
```

```
Top 2 answers:
Opened in 2001, Disney California Adventure Park was built on the site of Dis
neyland's original parking lot.
Disneyland Park, originally Disneyland, is the first of two theme parks built
at the Disneyland Resort in Anaheim, California, opened on July 17, 1955.
```

Comment: This method also gives the first neighbour as the correct answer. Compared to tf-idf in Excercise 2, the second neighbour chosen is also more related to the question, using word embeddings.

END ANSWER

In [ ]:

```

```