Assignment 2

● Is now available.

● Due date: 23.59pm on Friday, November 4th.
Assignment 2

- Word Sense Disambiguation

- After A2, you will be familiar with:
  - NLTK, WordNet, Lesk algorithm, using Word2Vec/Bert word embeddings
NLTK Package

- WordNet.
- Tokenizer.
>>> from nltk.corpus import wordnet as wn

>>> wn.synsets('motorcar')

[Synset('car.n.01')]

>>> wn.synset('car.n.01').lemma_names()

['car', 'auto', 'automobile', 'machine', 'motorcar']

>>> wn.synsets('car')

[Synset('car.n.01'), Synset('car.n.02'), Synset('car.n.03'), Synset('car.n.04'), Synset('cable_car.n.01')]
Tokenizer

- Tokenize a string to split off punctuation other than periods.

- Input:

```python
s = """Good muffins cost $3.88\n in New York. Please buy me two of them.\n Thanks."""
```

- Output:

```python
['Good', 'muffins', 'cost', '$', '3.88', 'in', 'New', 'York', '.', 'Please', 'buy', 'me', 'two', 'of', 'them', '.', 'Thanks', '.']
```
Tokenizer

- "$3.88": ["$3.88"] or ["\$,3\", ".", \"88\"]

- sometimes: ["sometimes"] or ["some", "times"]
Stopwords

```python
>>> from nltk.corpus import stopwords
>>> stopwords.words('english')
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', 'your', 'yours',
 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', 'her', 'hers',
  
  ......
'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', 'should', 'now']
```
Lesk Algorithm

Algorithm 1: The simplified Lesk algorithm.

input: a word to disambiguate and the sentence in which it appears

best_sense ← most_frequent_sense word
best_score ← 0
context ← the bag of words in sentence

for each sense of word do
    signature ← the bag of words in the definition and examples of sense
    score ← Overlap(signature, context)
    if score > best_score then
        best_sense ← sense
        best_score ← score
    end
end

return best_sense
Score

- Count(overlap).
- Bag of words VS set of words.
- Vector representation.
  - Vector with counts.
  - Embedding.
- Vector similarity
  - Euclidean distance.
  - Cosine similarity.
  - Dot product.
Word2Vec

- Pretrained word vectors from large amount of text.

- Words are mapped to vectors of real numbers.

  \[ \text{word2vec}(\text{word: str}) \rightarrow \text{vector: np.array()} \]

- Each word only has one fixed vector representation, although it could have multiple senses.
Word2Vec
Word2Vec
# Contextual Word Embedding

<table>
<thead>
<tr>
<th>Source</th>
<th>Nearest Neighbors</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fixed</strong> play</td>
<td>playing, game, games, played, players, plays, player, Play, football, multiplayer</td>
</tr>
<tr>
<td>Chico Ruiz made a spectacular play on Alusik’s grounder {...}</td>
<td>Kieffer, the only junior in the group, was commended for his ability to hit in the clutch, as well as his all-round excellent play.</td>
</tr>
<tr>
<td><strong>Contextual</strong> Olivia De Havilland signed to do a Broadway play for Garson {...}</td>
<td>{...} they were actors who had been handed fat roles in a successful play, and had talent enough to fill the roles competently, with nice understatement.</td>
</tr>
</tbody>
</table>
BERT: pretrained language model
Bert: Next Sentence Prediction
BERT

- Feed model indices instead of strings.
- Input embedding layers + 12 hidden layers.
- Dimension.
- Realigning: BERT Tokenizer VS NLTK Tokenizer.
Do not use LOOP!

Consider matrix multiplication:

for (int i = 0; i < M; ++i)
    for (int j = 0; j < N; ++j)
        for (int k = 0; k < K; ++k)
            C[i][j] += A[i][k] * B[k][j];

Too slow!!!

```python
>>> mat1 = torch.randn(2, 3)
>>> mat2 = torch.randn(3, 3)
>>> torch.mm(mat1, mat2)
tensor([[ 0.4851,  0.5037, -0.3633],
         [-0.0760, -3.6705,  2.4784]])
```
Questions?