COMP4650/COMP6490 - DOCUMENT ANALYSIS
Assignment NLP Report

Zhao, Longfei
u5976992

October 13, 2017

Q2. Kneser-Ney Smoothing

$count(am) = 3$
$count(am, Sam) = 2$
$|\{x : count(am, x) > 0\}| = 2$
Assume $N(w_i) = |\{x : count(x, w_i) > 0\}|$ and $d = 0.75$. Hence,
$N(\langle s \rangle) = 0$, $N(\langle /s \rangle) = 2$, $N(I) = 2$, $N(am) = 1$, $N(Sam) = 3$, $N(do) = 1$, $N(not) = 1$
$N(like) = 1$, $N(green) = 1$, $N(apples) = 1$, $N(and) = 1$
$\Rightarrow \sum_{w_i} N(w_i) = 14$
$\Rightarrow \lambda(am) = \frac{d}{count(am)} |\{x : count(am, x) > 0\}| = \frac{0.75}{3} * 2 = 0.5$
$\Rightarrow P_{kn}(Sam) = \frac{N(Sam)}{\sum_{w_i} N(w_i)} = \frac{3}{14}$
$\Rightarrow P_{kn}(Sam|am) = \frac{\max(count(am,Sam)-d,0)}{count(am)} + \lambda(am) P_{kn}(Sam) = \frac{1.25}{3} + 0.5 * \frac{3}{14} = 0.5238$

Q3. Context-Free Grammars

$PRP\$ \rightarrow my \mid his \mid her \mid its$
$\cancel{PNP \rightarrow nounEndWithS'}$ $\quad PNP \rightarrow nounEndWithS' \mid nounEndWith'S$
$Nominal \rightarrow PNP$
$\cancel{Det\ Nominal \rightarrow Det\ Noun}$ $\quad Nominal \rightarrow Det\ Nominal$
$Nominal \rightarrow PRP\$\ Nominal$
$Nominal \rightarrow Nominal\ Noun$
$Nominal \rightarrow Noun$

Q4. Word Embeddings

We can consider an unseen word as it's subwords or character n-grams. We could train a ngram model which takes letters as tokens(Bojanowski, Grave, Joulin, & Mikolov, 2016). Therefore, we will get the frequence and word embeddings of all "syllables". Then, an unseen word can be splitted properly to a set of syllables. Hence, we use easily combine those syllables to get the word embedding.

Q5. Transition-based Dependency Parsing

Denote $\langle v_i | S, v_j | I, A \rangle$

The reason why *Left-Arc*(LA) needs to remove the topmost element from the stack is that avoid creating a cycle in the graph. For example, if we keep v_i in S, there is a chance that adding an arc $v_i \rightarrow v_j$ to A in the later operation. However $v_j \rightarrow v_i$ is alread in A. Therefore, there is a cycle (Nivre, 2003).

The reason for *Right-Arc*(RA) is also to prevent to create a cycle. v_j should be reduced before v_i, otherwise arc linking these nodes might be added (Nivre, 2003).

The space complexity is also $O(n)$, the reason is as follow. For *Reduce*(R) and $Shift(S)$, they won't increase the space. For LA and RA, the space will increase 1. It can be easily seen that $T_{RA} + T_S = n, T_{LA} + T_R \leq n$ (T_i means the time of i operation). Hence, $T_{RA} + T_{LA} \leq 2n$ and the initial data space complexity is $O(n)$. As a result, the space complexity is $O(n)$.

References

[1] Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. *Enriching word vectors with subword information.* 2016

[2] Nivre, J. *An efficient algorithm for projective dependency parsing.* 2003