SNLP 2016
Exercise 5

Submission date: 3.05.2016, 23:59

1) (3 points) Given the following statistics:

count, $N(w,h)$	count of Counts, $n_{N(w,h)}$
1	5000
2	1600
3	800
4	500
5	300

Table 1: These are all the bigrams with the history "beer"

count, $N(w,h)$	bigram
1	beer drinker
2	beer lover
3	beer glass

a) What are the discounted counts under GoodTuring discounting for the three given bigrams?
Solution:
$N(beer, drinker) = 2 * \frac{1600}{5000} = 0.64$
$N(beer, lover) = 3 * \frac{800}{1600} = 1.5$
$N(beer, glass) = 4 * \frac{500}{800} = 2.5$

b) The amounts from discounting counts are given to a back-off unigram model. Using such a back-off model, what are the probabilities for the following bigrams?
(i) $p(drinker|beer)$
(ii) $p(glass|beer)$
(iii) $p(mug|beer)$
Note: $p(mug) = 0.01$, $p(drinker) = 0.01$, $p(glass) = 0.015$. State any assumptions that you make.
Solution:
$\frac{N(beer,drinker}{N(beer)} = \frac{0.64}{6} = 0.11$
$\frac{N(beer,lover}{N(beer)} = \frac{1.5}{6} = 0.25$
$\frac{N(beer,glass}{N(beer)} = \frac{2.5}{6} = 0.42$
$\sum_w \frac{N(w,h)}{N(h)} = 0.78$
$\alpha(h) = 1 - 0.78 = 0.22$
$P(drinker|beer) = 0.11 + 0.22 * 0.01 = 0.1122$
$P(glass|beer) = 0.42 + 0.22 * 0.015 = 0.4233$
$P(mug|beer) = 0.22 * 0.01 = 0.0022$

b) (7 points) In this exercise you will complete the code of a back-off language model. You can find the the starter code in the provided materials: *ex5.py*

a) (1 point) Fill in the missing lines to return for each history the R value as described on slides 21 and 22.

b) (3 points) Complete discounting_model.__items__() in order to return the smoothed probability. Use the same discounting parameter d to smooth the bigram and unigram distributions.

$$P(w_i|w_{i-1}) = \begin{cases} \frac{N(w,h)-d}{N(h)} + \alpha(h)P(w) & \text{if } N(w,h) > 0 \\ \alpha(h)P(w) & \text{else} \end{cases} \tag{1}$$

$$P(w) = \begin{cases} \frac{N(w)-d}{N} + \alpha\frac{1}{V} & \text{if } N(w) > 0 \\ \alpha\frac{1}{V} & \text{else} \end{cases} \tag{2}$$

If the history h of the bigram (w, h) is not found in the training corpus return the estimate of the unigram language model $P(w)$.[1]

c) (3 points) In the lecture you learned about leaving-one out cross validation. This would be infeasible for such a big dataset. Istead of leaving only one word out for cross validation, you will implement a function where you always leave out a different fraction of the data for validation. This method is called K-fold cross validation, where K stands for the number of fractions.
Split the data into K parts. Use each of the K parts once for validation and merge the rest for training. Calculate perplexity on the validation set. Take the average of the K perplexity values and return it. You will use this value to optimize the discounting parameter d.
You will find the starter code in the function kfold_crossvalidation().
The code will automatically plot the cross-validation perplexity for different d values. have a look at the plot. What would be the optimal value for d?

1 Bonus

- 1) (2 points) Show that:

$$\frac{N(w,h)+\epsilon}{N(h)+\epsilon V} = \mu\frac{N(w,h)}{N(h)} + (1-\mu)\frac{1}{V}\ , \tag{3}$$

Solution: $\mu = \frac{N(h)}{N(h)+V\lambda}$

- 2) (1 point) Can you think of a better back-off distribution for the unigram model than the zero gram model?
Solution: any subword level LM

[1] A language model is defined for a specific history. In case a history is unseen an arbitrary other language model can be used. This is called a fall back language model.