Welcome to the Statistical Methods of Language Technologyb SoSe21 course

Dr. Seid Muhie Yimam

- Email: yimam@informatik.uni-hamburg.de (mailto:yimam@informatik.uni-hamburg.de)
- Office: Informatikum, F-415

Dr. Özge Alaçam

- Email: alacam@informatik.uni-hamburg.de (mailto:alacam@informatik.uni-hamburg.de)
- Office: Informatikum, F-435

Topic of this week;

In this first practice class, we are going to focus on two main topics, which will be useful to complete the assignment;

- POS HMM
- CRF

Deadline: 19/24 May

In class Exercises

Problem 5.1 POS HMM

a) Train a POS HMM with MLE estimation, using the annotated text:

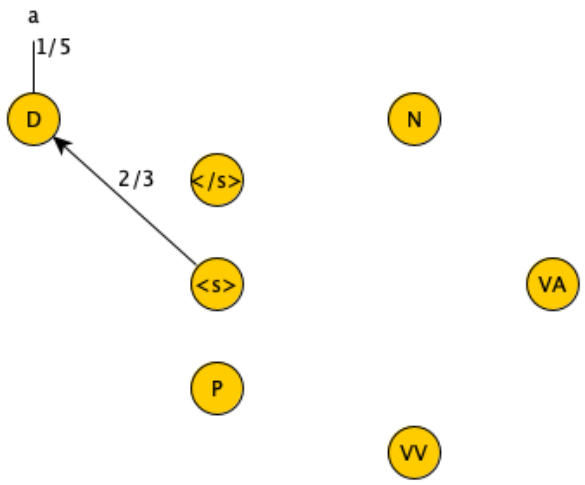
```
the/D cat/N can/VA fish/VV a/D fish/N

the/D fish/N is/VV in/P the/D fish/N can/N

workers/N can/VV the/D fish/N
```

Reminder: The MLE estimates can be obtained by

$$P(s_i|s_j) = \frac{C(s_i, s_j)}{C(s_j)} \qquad \text{and} \qquad P(w_k|s_i) = \frac{C(w_k, s_i)}{C(s_i)} \quad ,$$

where C is the count function.

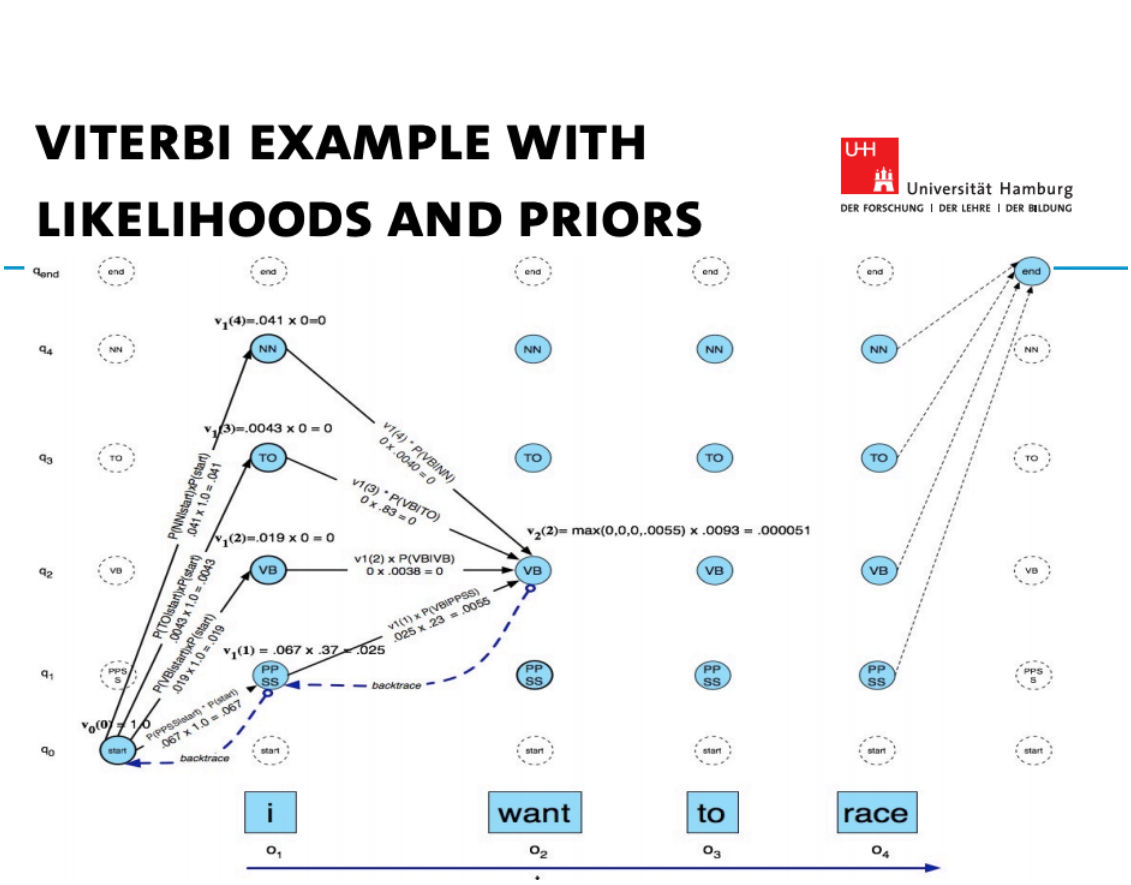
The following initial model shows the transition probability and the transition probability for one word.



b) Tag the following texts, using the HMM from a) and provide the probabilities of the sequences given the observation.

```
1) a cat can

2) a cat can can

------
</S>

P           P        P        P

VV          VV       VV       VV

VA          VA       VA       VA

N           N        N        N

D           D        D        D

<S>         a        cat      run
------
```

The following sketch from the lecture shows on how to compute the probabilities of the sequence.



c) What is the probability of the following sequence?

```
the workers can can the food in a can
```

Problem 5.2 CRF ++

Download and install **CRF++** from http://taku910.github.io/crfpp/ (http://taku910.github.io/crfpp/) (tested with version 0.58; Linux users might need to install g++).

Download the **PC5-data.tar.gz** file from Moodle and unpack it. For evaluation, you need to have access to Perl (Windows users can use Babun, http://babun.github.io/ (http://babun.github.io/)).

a) Let's have a look at the data. What do the columns mean? What is the task? What makes sense to check for evaluation, Token-level accuracy vs. chunk-level P/R/FB1? The description can be found at https://www.clips.uantwerpen.be/conll2000/chunking/ (https://www.clips.uantwerpen.be/conll2000/chunking/)

```
in      IN      B-PP
the     DT      B-NP
pound   NN      I-NP
is      VBZ     B-VP
widely  RB      I-VP
expected        VBN     I-VP
to      TO      I-VP
take    VB      I-VP
another DT      B-NP
sharp   JJ      I-NP
dive    NN      I-NP
if      IN      B-SBAR
trade   NN      B-NP
figures NNS     I-NP
for     IN      B-PP
September       NNP     B-NP
,       ,       O
due     JJ      B-ADJP
for     IN      B-PP
release NN      B-NP
tomorrow        NN      B-NP
,       ,       O
fail    VB      B-VP
to      TO      I-VP
show    VB      I-VP
a       DT      B-NP
substantial     JJ      I-NP
improvement     NN      I-NP
from    IN      B-PP
July    NNP     B-NP
and     CC      I-NP
August  NNP     I-NP
```

b) Let's train a CRF on the small training data that only is conditioned on the POS tag at the current position and on the neighbor (-1, +1) output tags (bigram option).

```
------ pc5.template ------
U11:%x[0,1]
B
```

command: `$ crf_learn -m 200 pc5.template train.small.data pc5b.small.model`

c) Now, let us apply it to the small validation data. Look at the output and evaluate it using the provided perl script.

```
$ crf_test -m pc5b.small.model vali.small.data  > vali.pc5b.output
```

```
$ perl conlleval.pl < vali.pc5b.output
```

The output should look like the following

```
accuracy:  89.76%; precision:  83.21%; recall:  81.32%; FB1:  82.26
             ADJP: precision:  53.33%; recall:  21.62%; FB1:  30.77
30
             ADVP: precision:  62.60%; recall:  60.74%; FB1:  61.65
131
            CONJP: precision:   0.00%; recall:   0.00%; FB1:   0.00
0
               NP: precision:  82.67%; recall:  80.37%; FB1:  81.50
2031
               PP: precision:  83.91%; recall:  96.80%; FB1:  89.89
901
              PRT: precision:   0.00%; recall:   0.00%; FB1:   0.00
0
             SBAR: precision:   0.00%; recall:   0.00%; FB1:   0.00
0
               VP: precision:  88.22%; recall:  88.22%; FB1:  88.22
815
```

d) Let's extend the model to use also the preceding (-1) and following (1) POS tag. What happens to the error rates while training? What happens in the evaluation?

```
------ pc5.template: ------
U11:%x[-1,1]
U21:%x[0,1]
U31:%x[1,1]
B
```

commands:

```
$ crf_learn -m 200 pc5.template  train.small.data pc5d.small.model
```

```
$ crf_test -m pc5d.small.model vali.small.data | perl conlleval.pl
```

e) Let's extend it for POS in positions -2 and 2. Explain error rates and performance measures.

```
------ pc5.template: ------
U11:%x[-2,1]
U21:%x[-1,1]
U31:%x[0,1]
U41:%x[1,1]
U51:%x[2,1]
B
```

commands:

```
$ crf_learn -m 200 pc5.template  train.small.data pc5e.small.model
```

```
$ crf_test -m pc5e.small.model vali.small.data | perl conlleval.pl
```

f) Add POS bigram features [-2,-1] and [1,2]. Error rates? Performance measures?

```
------ pc5.template: ------
U11:%x[-2,1]
U21:%x[-1,1]
U31:%x[0,1]
U41:%x[1,1]
U51:%x[2,1]
U62:%x[-2,1]/%x[-1,1]
U72:%x[1,1]/%x[2,1]
B
```

commands:

```
$ crf_learn -m 200 pc5.template  train.small.data pc5f.small.model
```

```
$ crf_test -m pc5f.small.model vali.small.data | perl conlleval.pl
```

g) Add the current word to the features from \textit{d}). Does it help?

```
------ pc5.template: ------
U00:\%x[0,0]
U11:\%x[-1,1]
U21:\%x[0,1]
U31:\%x[1,1]
B
```

commands:

```
$ crf_learn -m 200 pc5.template  train.small.data pc5g.small.model
```

```
$ crf_test -m pc5g.small.model vali.small.data | perl conlleval.pl
```

h) Experiment: Who can build the best model for the small data, and how well does it perform on the yet unseen test data (`test.small.data`)?

In [ ]:

In [ ]:

Good luck with your assignment :-)