TUTORIAL 2

Naquee Rizwan

IIT Kharagpur

September 10, 2024

Question 1: Eager-Arc Parsing

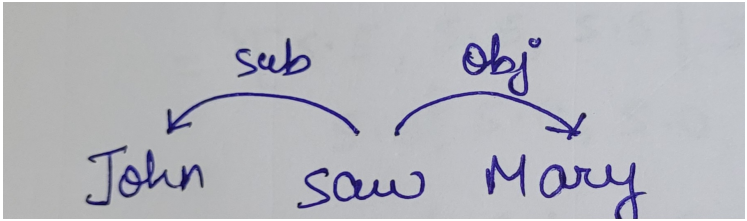
Consider the sentence, *'John saw Mary'*.

- Draw a dependency graph for this sentence.
- Assume that you are learning a classifier for the data-driven deterministic parsing and the above sentence is a gold-standard parse in your training data. You are also given that *John* and *Mary* are 'Nouns', while the POS tag of *saw* is 'Verb'. Assume that your features correspond to the following conditions:
  - The stack is empty.
  - Top of stack is Noun and Top of buffer is Verb.
  - Top of stack is Verb and Top of buffer is Noun.

  Initialize the weights of all your features to 5.0, except that in all of the above cases, you give a weight of 5.5 to Left-Arc. Define your feature vector and the initial weight vector.
- Use this gold standard parse during online learning and report the weights after completing one full iteration of Arc-Eager parsing over this sentence.

Hint 2: Eager-Arc Parsing (Dependency Graph)

Dependency graph as per the provided gold label / ORACLE data. This graph will be useful in our training procedure. Now, directly jump to the `LEARN` algorithm as present in lecture slides of classification based dependency parsing to start the training process.



Hint 2: Eager-Arc Parsing (Initial Configuration)

As per given in the question, assume the following as corresponding features:

- **C1:** The stack is empty.
- **C2:** Top of stack is Noun and Top of buffer is Verb.
- **C3:** Top of stack is Verb and Top of buffer is Noun.

Now, since this has turned out to be a classification problem where we need to figure out 1 of the 4 possible transitions, make a 12 dimensional feature vector as follows:

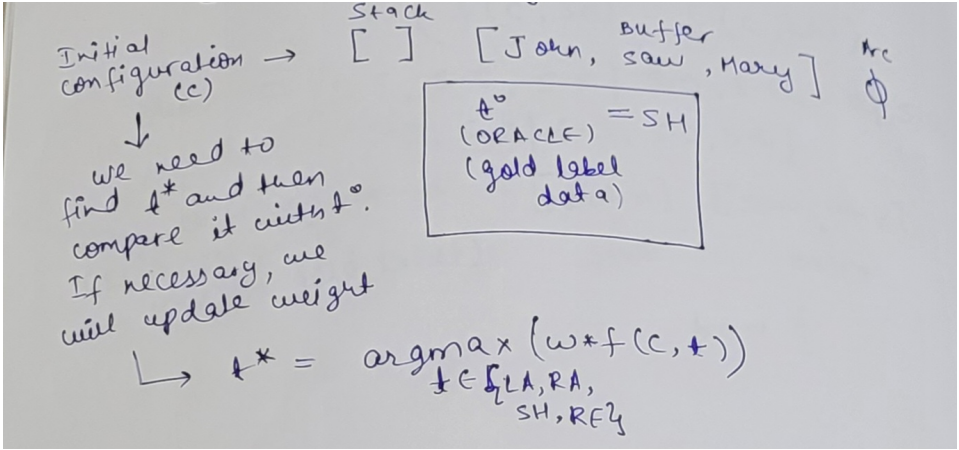
```
f(c,t):  [C1&&LA, C2&&LA, C3&&LA | C1&&RA,C2&&RA, C3&&RA | C1&&RE,
C2&&RE, C3&&RE | C1&&SH, C2&&SH,C3&&SH]
```

Here, **LA:** Left Arc, **RA:** Right Arc, **RE:** Reduce, **SH:** Shift and && is the binary `AND` operator.

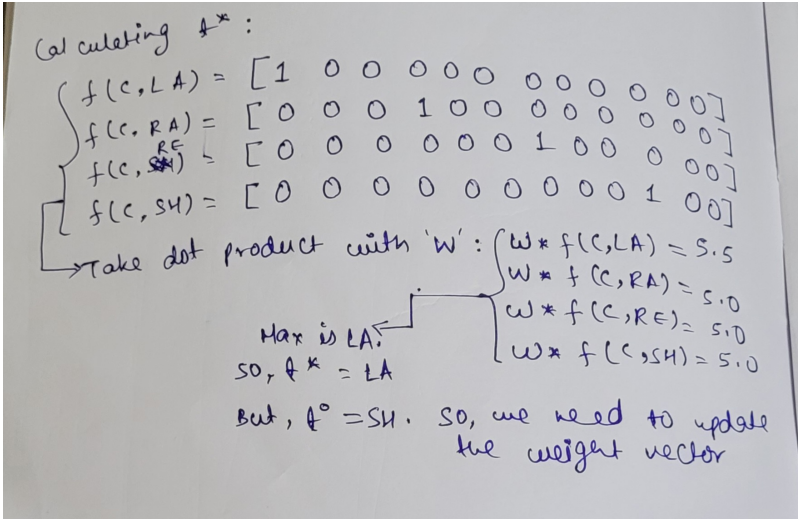
Also initialize the weight as per the question.

W: = [5.5, 5.5, 5.5 | 5.0, 5.0, 5.0 | 5.0, 5.0, 5.0 | 5.0, 5.0, 5.0]

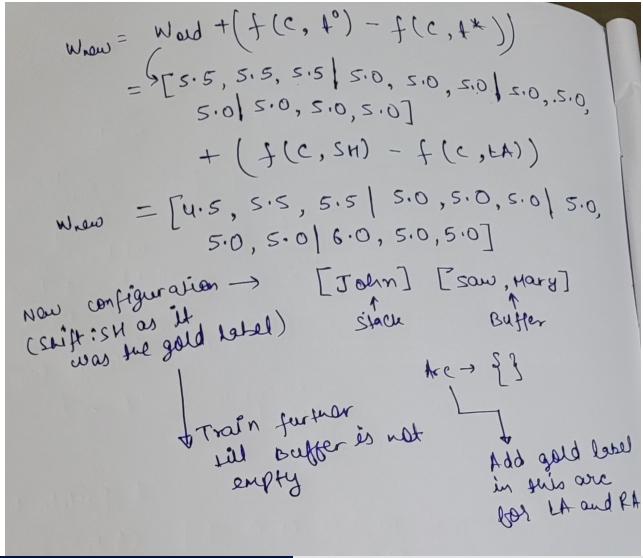
Solution 2: Eager-Arc Parsing (Initial Configuration)



Solution 2: Eager-Arc Parsing (Calculate t*)



Solution 2: Eager-Arc Parsing (Update Weights)



Question 2: Perplexity

Consider the following toy example:
Training data:

```
<s> I am Sam </s>
<s> Sam I am </s>
<s> Sam I like </s>
<s> Sam I do like </s>
<s> do I like Sam </s>
```

Assume that we use a bigram language model (no smoothing) based on the above training data.
Compute **(a)** the bigram probabilities and **(b)** perplexity of: `<s> I do like Sam`

Solution 2: Perplexity

Bigram probabilities:

$$P(\texttt{Sam}|\texttt{<s>}) = \frac{3}{5} \quad P(\texttt{I}|\texttt{<s>}) = \frac{1}{5}$$
$$P(\texttt{I}|\texttt{Sam}) = \frac{3}{5} \quad P(\texttt{</s>}|\texttt{Sam}) = \frac{2}{5}$$
$$P(\texttt{Sam}|\texttt{am}) = \frac{1}{2} \quad P(\texttt{</s>}|\texttt{am}) = \frac{1}{2}$$
$$P(\texttt{am}|\texttt{I}) = \frac{2}{5} \quad P(\texttt{like}|\texttt{I}) = \frac{2}{5} \quad P(\texttt{do}|\texttt{I}) = \frac{1}{5}$$
$$P(\texttt{Sam}|\texttt{like}) = \frac{1}{3} \quad P(\texttt{</s>}|\texttt{like}) = \frac{2}{3}$$
$$P(\texttt{like}|\texttt{do}) = \frac{1}{2} \quad P(\texttt{I}|\texttt{do}) = \frac{1}{2}$$

Perplexity:

The probability of this sequence is $\frac{1}{5} \cdot \frac{1}{5} \cdot \frac{1}{2} \cdot \frac{1}{3} = \frac{1}{150}$.

The perplexity is then $\sqrt[4]{150} = 3.5$

Question 3: POS Tagging

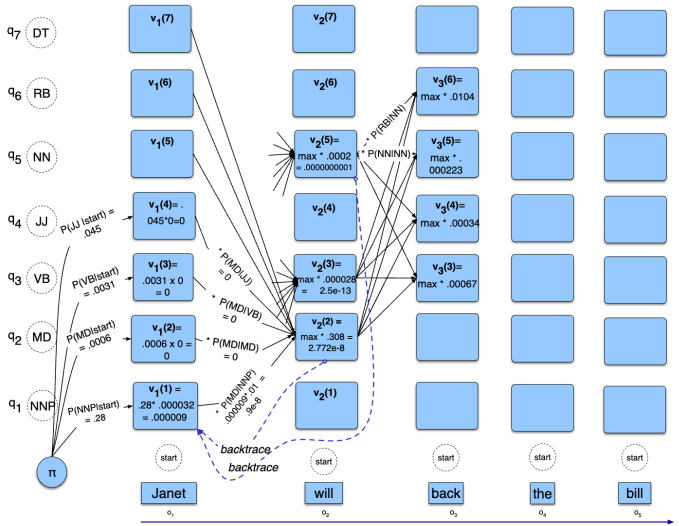
Suppose you want to use a HMM tagger to tag the phrase, *"Janet will back the hill"*, where we have the following probabilities:

	NNP	MD	VB	JJ	NN	RB	DT
<s>	0.2767	0.0006	0.0031	0.0453	0.0449	0.0510	0.2026
NNP	0.3777	0.0110	0.0009	0.0084	0.0584	0.0090	0.0025
MD	0.0008	0.0002	0.7968	0.0005	0.0008	0.1698	0.0041
VB	0.0322	0.0005	0.0050	0.0837	0.0615	0.0514	0.2231
JJ	0.0366	0.0004	0.0001	0.0733	0.4509	0.0036	0.0036
NN	0.0096	0.0176	0.0014	0.0086	0.1216	0.0177	0.0068
RB	0.0068	0.0102	0.1011	0.1012	0.0120	0.0728	0.0479
DT	0.1147	0.0021	0.0002	0.2157	0.4744	0.0102	0.0017

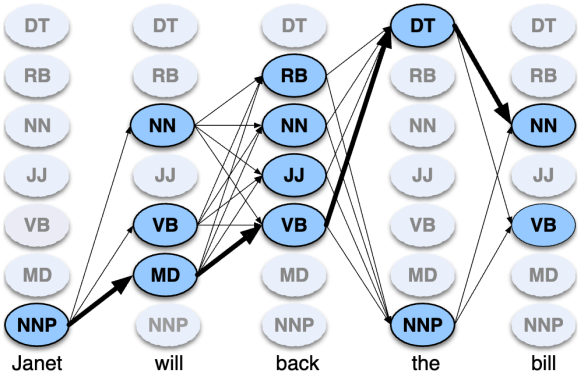
	Janet	will	back	the	bill
NNP	0.000032	0	0	0.000048	0
MD	0	0.308431	0	0	0
VB	0	0.000028	0.000672	0	0.000028
JJ	0	0	0.000340	0	0
NN	0	0.000200	0.000223	0	0.002337
RB	0	0	0.010446	0	0
DT	0	0	0	0.506099	0

Work out in details the steps of the Viterbi algorithm. You can use a Table to show the steps. Assume that all tags have the same probabilities to appear in the beginning of a sentence.

Hint 3: POS Tagging

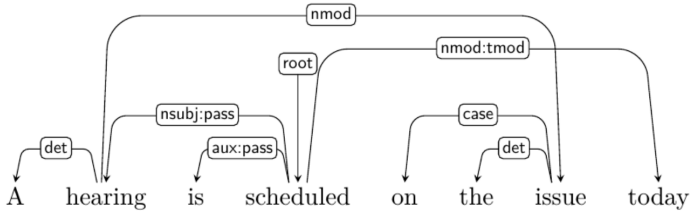


Solution 3: POS Tagging



Question 4: Dependency Parsing

a: Identify the necessary condition which is violated in this dependency graph:



b: Why do we remove the word from stack in Left Arc Operation? Why we don't remove the word in Right Arc Operation?

Hint 4: Dependency Parsing

Look through following properties of Dependency graph:

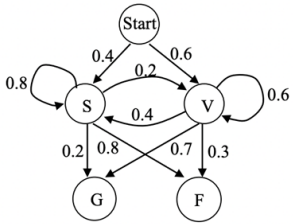
- `G is connected:` For every node i there is a node j such that i -> j or j -> i.
- `G is acyclic:` if i -> j then not j -> *i.
- `G obeys the single head constraint:` if i -> j then not k -> j, for any k , i.
- `G is projective:` if i -> j then j -> *k, for any k such that both j and k lie on the same side of i.

Hint 4: Dependency Parsing

a. Projective property is violated.
b. In Left Arc operation, the top word in stack has already completed its dependency arcs with remaining words in the stack. On top of this, forming a dependency with the remaining words of buffer in future *(which was hypothetically only possible through Right Arc operation, else it would have violated single head constraint)* will violate the projective property. Hence, it is necessary to remove it.
In Right Arc operation, the word after being moved from queue to stack is capable of creating dependency arcs with remaining items of buffer without any violation. That is why this word from buffer is pushed to stack and is not removed. The word which was at the top of stack initially, also is capable of creating dependency arcs with the same logic as above.

Question 5: HMM

Consider the HMM in the Figure that models a students activities every hour, where during each hour period the student is in one of two possible hidden states: Studying (S) or playing Video games (V). While doing each activity the student will exhibit an observable facial expression of either Grinning (G) or Frowning (F).



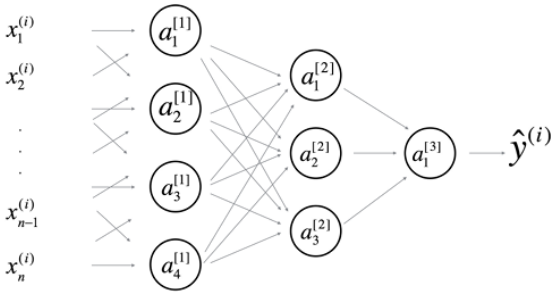
If in the second hour the student is Studying, what is the probability that in the fourth hour the student is playing Video games? That is, compute P(q4 = V|q2 = S). Show the steps of computation.

Solution 5: HMM

```
STEP 1:  P(q4=V | q2=S)
STEP 2:  P(q4=V, q3=V | q2=S) + P(q4=V, q3=S | q2=S)
STEP 3:  P(q4=V | q3=V, q2=S) * P(q3=V | q2=S) + P (q4=V | q3=S ,
q2=S) * P (q3=S | q2=S)
STEP 4:  P (q4=V | q3=V) * P(q3=V | q2=S) + P(q4=V | q3=S) * P(q3=S |
q2=S)
Result:  (.6) * (.2) + (.2) * (.8) = 0.28
```

Question 6: Forward Propagation

Consider the following 2-layer fully connected neural network. All activations are sigmoids and your optimizer is stochastic gradient descent. You initialize all the weights and biases to zero and forward propagate an input $\boldsymbol{x}$ in the network. What is the output?



Solution 6: Forward Propagation

`Hint:`
Carefully read the question!! All weights and bias = 0.

`Result:` 0.5

Question 7: Backward Propagation

Consider an example with two inputs (x1 , x2) and two outputs given as:

$$f_1(x_1, x_2) = x_1 x_2 + exp(x_1 x_2) - sin(x_2)$$

$$f_2(x_1, x_2) = (x_1 x_2 - sin(x_2))exp(x_1 x_2)$$

Provide the expression for $\frac{\partial f}{\partial x}$, with f and x defined for the outputs and inputs mentioned above. To solve this, please make use of the following primal variables:
`v1 = x1, v2 = x2, v3 = v1*v2, v4 = sin(v2), v5 = exp(v3), v6 = v3-v4, v7 = v5+v6, v8 = v5*v6`
Draw the forward propagation diagram with the help of these variables, and use back-propagation to compute $\frac{\partial f}{\partial x}$. Provide your final answer in terms of the primal variables before giving the final expression in terms of *x1* and *x2*.

Solution 7: Backward Propagation

- TODO