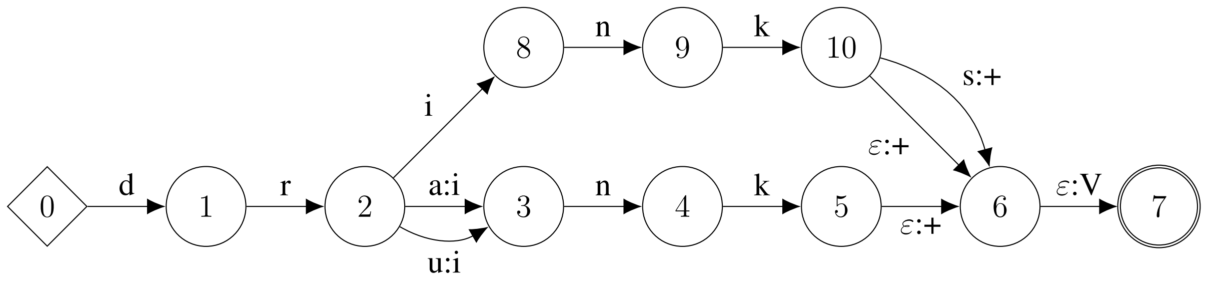
QUESTION I : Transducer [6 pt]

Consider the following (toy) transducer implementing (a very small fraction of) the inflectional morphology of English words:



① **[4 pt]** Provide the complete list of the (surface form, canonical representation) associations recognized by the provided transducer.

Answer: (drink, drink+V)

(drinks, drink+V)

(drank, drink+V)

(drunk, drink+V)

② **[2 pt]** What type of morphology does the provided transducer implement?

Answer: (concatenative) inflectional morphology

QUESTION II : Trees versus tags [19 pt]

Consider the (toy) SCFG describing English compound nouns (CN) and consisting of the following syntactic rules:

```
CN --> N       [p1]
CN --> N  CN   [p2]
CN --> CN N    [p3]
```

and of the following (excerpt of) lexical rules:

```
N --> satellite       [q1]
N --> communication   [q2]
N --> antenna         [q3]
```

① **[6 pt]** Indicate the parse tree(s) that can be associated by the provided grammar to the sequence "*satellite communication antenna*". To simplify the trees, you can use the notation w_1 (resp. w_2, w_3) to represent the word "*satellite*" (resp. "*communication*", "*antenna*").

Answer: 4 parse trees:

```
(CN (CN (CN (N w1)) (N w2)) (N w3))
(CN (CN (N w1) (CN (N w2))) (N w3))
(CN (N w1) (CN (CN (N w2)) (N w3)))
(CN (N w1) (CN (N w2) (CN (N w3))))
```

② **[5 pt]** Compute the probability P associated by the provided grammar to the sequence "*satellite communication antenna*". Justify your answer and express the requested probability as a formula using the coefficients of the above provided SCFG.

Answer:

$$\begin{aligned} P(w_1w_2w_3w_4) &= \sum_{\text{tree}} P(w_1w_2w_3w_4, \text{tree}) \\ &= \sum_{\text{tree deriving } w_1w_2w_3w_4} P(\text{tree}) \\ &= P(T_1) + P(T_2) + P(T_3) + P(T_4) \\ &= p_2 \cdot q_1 \cdot p_2 \cdot q_2 \cdot p_1 \cdot q_3 + p_3 \cdot p_2 \cdot q_1 \cdot p_1 \cdot q_2 \cdot q_3 \\ &\quad + p_2 \cdot q_1 \cdot p_3 \cdot p_1 \cdot q_2 \cdot q_3 + p_3 \cdot p_3 \cdot p_1 \cdot q_1 \cdot q_2 \cdot q_3 \\ &= p_1 \cdot (p_2 + p_3)^2 \cdot q_1 \cdot q_2 \cdot q_3 \end{aligned}$$

③ **[8 pt]** We now consider that the analysis of compound nouns is implemented in the form of a part-of-speech tagging task performed by an order-2 HMM with only two hidden states, `N` and `CN`, and such that:

$$P(\mathtt{N}|\mathtt{N}) = P(\mathtt{CN}|\mathtt{CN}) = 0$$

a) Indicate all the non-zero probability PoS taggings that can be associated by the provided HMM to the sequence "*satellite communication antenna*".
For each of them, derive the joint probability for the tagging and the provided sequence, as a formula using the parameters of the HMM model. Justify your answer. To simplify the derivation, you can use the notation w_1 (resp. w_2, w_3) to represent the word "*satellite*" (resp. "*communication*", "*antenna*").

Answer:

Notice that for an order-2 HMM, since $P(\mathtt{N}|\mathtt{N}) = \sum_x P(x\ \mathtt{N}|\mathtt{N}) \propto \sum_x P(x\ \mathtt{N}\ \mathtt{N})$, $P(\mathtt{N}|\mathtt{N}) = 0 \implies P(x\ \mathtt{N}\ \mathtt{N}) = 0 \implies P(\mathtt{N}|x\ \mathtt{N}) = 0$; similarly $P(x|\mathtt{N}\ \mathtt{N}) = 0$; and similarly as well for `CN`. Thus the order-2 HMM we are here refering to, indeed reduces to only two states: "`N CN`" and "`CN N`" which can unambiguously be named "`CN`" and "`N`", respectively.

Moreover $P(\mathtt{N}|\mathtt{CN}) = P(\mathtt{CN}|\mathtt{N}) = 1$.

2 possible taggings: (N CN N) and (CN N CN)

Joint probability:

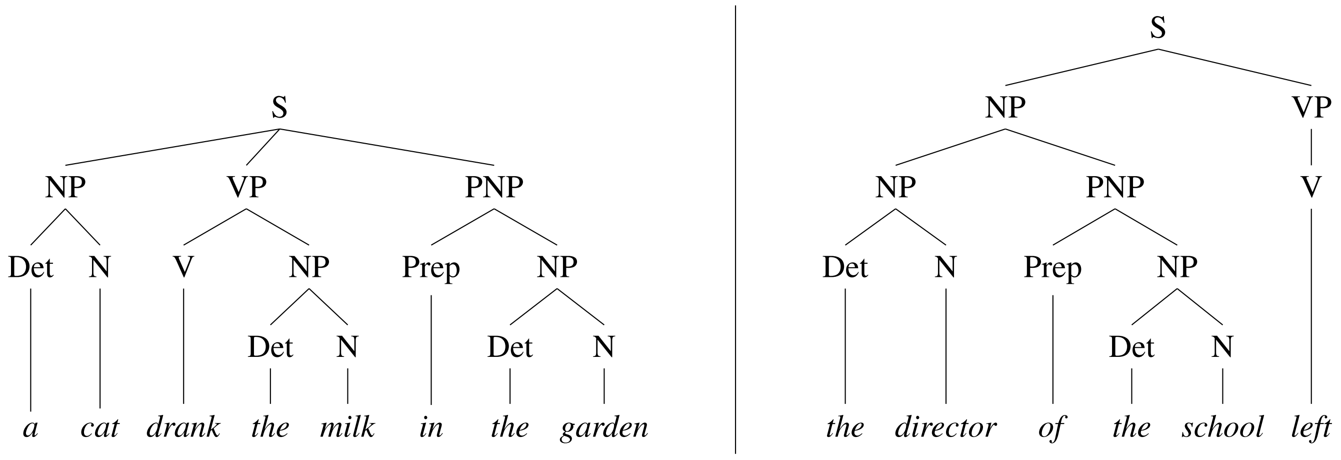
$$P(\mathtt{N\ CN\ N}, w_1w_2w_3) = P(w_1|\mathtt{N}) \cdot P(w_2|\mathtt{CN}) \cdot P(w_3|\mathtt{N}) \cdot P_{\text{init.}}(\mathtt{N})$$
$$P(\mathtt{CN\ N\ CN}, w_1w_2w_3) = P(w_1|\mathtt{CN}) \cdot P(w_2|\mathtt{N}) \cdot P(w_3|\mathtt{CN}) \cdot P_{\text{init.}}(\mathtt{CN})$$

b) Use the joint probabilities obtained in subquestion **a)** to derive the probability of the sequence "*satellite communication antenna*". Justify your answer.

Answer: The sum of the two above since $P(w_1w_2w_3) = \sum_{(t_1,t_2,t_3)} P(t_1t_2t_3, w_1w_2w_3)$ (and all other tagings having a 0 probability).

QUESTION III : [16 pt]

Consider the (toy) tree bank consisting of the following two trees:



① **[3 pt]** Out of this tree bank, extract all the CFG rules that have been used to generate the provided trees; first extract all the syntactic rules, then all the lexical ones, and, in each of these two sets, group one after the other the rules that share the same left-hand side non terminal.

Syntactic rules:

```
S --> NP VP
S --> NP VP PNP
VP --> V
VP --> V NP
```

```
PNP --> Prep NP
NP --> Det N
NP --> NP PNP
```

Lexical rules:

```
N --> cat
N --> milk
N --> garden
N --> director
N --> school
```

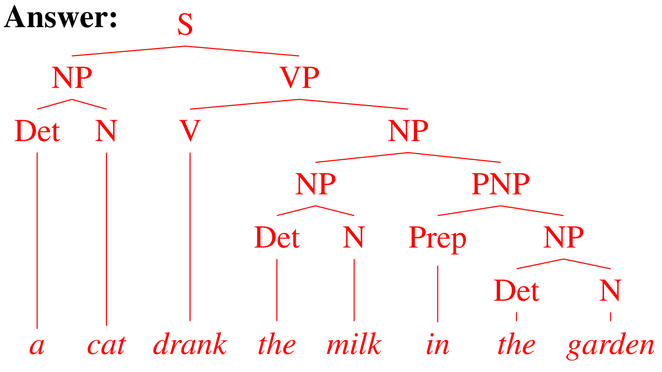
```
Det --> a
Det --> the

V --> drank
V --> left
```

```
Prep --> in
Prep --> of
```

② **[4 pt]** Draw hereafter the parse tree(s) different from the one provided in the tree bank that can be associated to the sequence "*a cat drank the milk in the garden*" by the CFG consisting of all the rules identified in subquestion ①.

Answer:



③ **[2 pt]** Indicate whether the grammar extracted in subquestion ① needs to be modified to be suitable for parsing with a CYK algorithm that operates on (non-extended) Chomsky Normal Form. If yes, indicate the grammar rules that should be removed and the new rules that should be added.

Answer: `S -> NP VP PNP` should be replaced by `S -> NP X` and `X -> VP PNP`

`VP -> V` should be replace by `VP -> drank` and `VP -> left`

④ **[2 pt]** Let us consider that the CFG extracted in subquestion ① has been converted into a SCFG by associating to each of its rules R a coefficient $p(R)$.

Indicate the constraints that must be verified by these coefficients. If necessary, use the notation "left(R)" to denote the left-hand side of the rule *R*.

Answer:

$$0 \leq p(R) \leq 1$$

$$\sum_{R':\text{left}(R')=\text{left}(R)} p(R') = 1$$

⑤ **[1 pt]** Given a rule R in a SCFG G, how many non-zero probability rules R' such that $\text{left}(R') = \text{left}(R)$ can be present in G if $p(R) = 1$?

Answer: 1

⑥ **[4 pt]** Provided that the grammar extracted in subquestion ① has first been extended into a SCFG as indicated in subquestion ④, and then modified as indicated in subquestion ③, what are the stochastic coefficients that should be associated with each of the new rules identified in subquestion ③ (if any)?
Express each of these coefficients, either in the form of a numerical value, or in the form of a formula using the coefficients of the original SCFG. Define your notations, if any.

Answer:

$$p(\texttt{S -> NP X}) = p(\texttt{S -> NP VP PNP})$$
$$p(\texttt{X -> VP PNP}) = 1$$
$$p(\texttt{VP -> drank}) = p(\texttt{V -> drank})$$
$$p(\texttt{VP -> left}) = p(\texttt{V -> left})$$

QUESTION IV : Fraud Detection [40 pt]

For your Master thesis, a financial company offers you to work on "*fraud detection using Natural Language Technology applied to client documents*".

① **[12 pt]** What they actually mean is to study the possible exploitation of some Deep-Learning (DL) method to achieve the proposed goal.

a) What are the main advantages and drawbacks of using DL over other methods for such a task?

Answer:

Advantages:

- seems to have better performances
- maybe less feature engineering

Drawback:

- requires lots of data
- requires heavy computational power
- hard and difficult to predict model design
- theoretical guarantees(/bounds) hard to optain

b) What are the 4 to 5 most important questions related to the proposed DL-based approach you should ask them to evaluate whether you have the means (or not) to perform such a study?

Answer: Here are 6 relevant questions:

- What type of fraud? What does it actually mean? $\longrightarrow$ more precisely define the task
- How could NLP help? Isn't it simply a usual data mining problem? $\longrightarrow$ better define input data
- Do you have evidences? (examples to learn from) $\longrightarrow$ better define data set/evaluation framework (what metric, how to assess these evidences?)
- How much data? enough data for DL? supervided?
- What computing power is available?
- What NLP competence existing in collaborators/supervisors?

c) Assuming that you received convincing answers to your questions, how would you design your DL NLP processing chain? Here we expect the main 3 to 4 *conceptual* modules, not the architectural details of the neural networks involved.

Answer:

(a) (if needed) document format transformation (e.g. PDF to text) – input representation

(b) word representation (word embeddings)

(c) sentence representation (sentence embeddings)

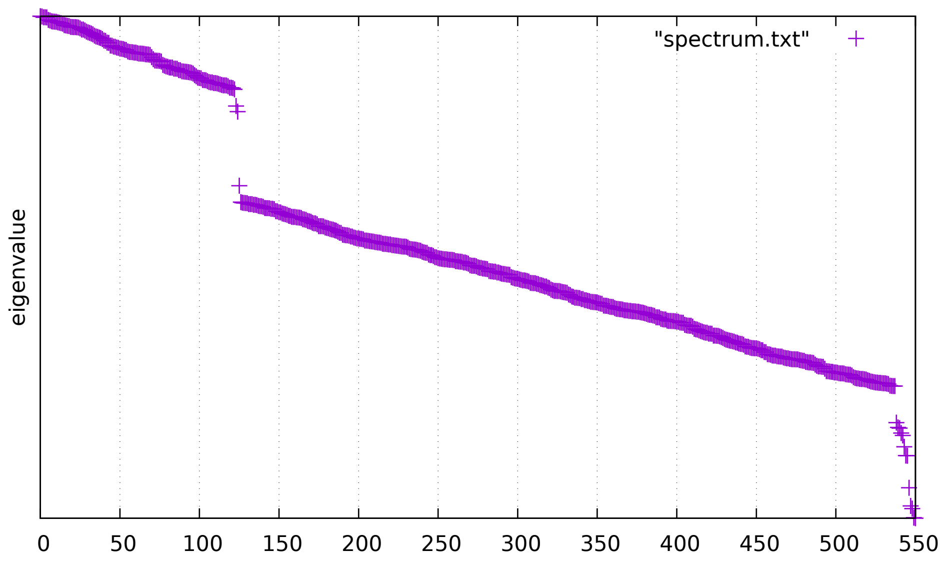
(d) classifier/prediction (softmax-like layer)

② **[2 pt]** What is a word embedding? What is it for?

Answer: It's a dense (= low-dimension; not sparse) numerical (= vector; computational) representation of a word to ease further vector-based semantical analysis of documents.

③ **[5 pt]** Some preliminary work has already been performed by a former intern who created tf-idf vectors based on an indexing set of 6'324 terms and reduced them to vectors of size 100 using PCA.

Reviewing his/her work and report, you found a graph related to the corresponding singular values. Here is a (rescaled) zoom on the first 550 left-most points in that graph:



a) What is the abscissa (x-value, horizontal axis) of the right-most point in the original complete graph (not reported here)?

Answer: 6'324

b) What do you think about the intern's methodology for selecting the dimension of the vector space? Would you have performed differently? If yes, how?

Answer: The general idea is good (reducing dimension keeping most data variance), however the concreate approach is not really sound as 100 seems like a random choice. $\simeq$ 125, or if compatible with other external constraints, $\simeq$ 540 are more appropriate since big gap in intertia.

[Reporting the percentage of total inertia would also help in such a context.]

continues on back ☞

④ **[5 pt]** Before considering more sophisticated methods, you wisely decide to start with a simple baseline, namely a Naive Bayes model.

a) For the targeted goal ("*fraud detection using Natural Language Technology applied to client documents*") and based on your answer to subquestion ③**b**, what is the input of the Naive Bayes module? What is the output? What are the parameters? What is needed for training such a model?

Answers:

input: "*document* vector" i.e. document PCA representation as done in subquestion ③**b**

output: most proable class (fraud/non-fraud); accepted answers: $P(\text{class}|\text{document}$ or $P(\text{class}, \text{document}$

parameters: $P(\text{class})$ for both classes and $P(\text{feature}|\text{class})$ for each "feature" (PCA dimension) resulting from subquestion ③**b**

needed for training: supervised (fraud/non-fraud) corpus of typical documents

b) Concretely, what probability should be computed as an output from the (very simple excerpt of) client document:

My salary is about 10'000 CHF and I don't pay any tax.

Answer:

$$P(\text{class}) \times \prod_{i=1}^{n} P(f_i|\text{class})$$

where n is either 125 or 540 from subquestion ③**b**, and f_i are the coordinates of the PCA representation of the above document (and "*class*" is either fraud or non-fraud; the other being 1 minus the former).

⑤ **[5 pt]** From your first analysis of the baseline results, you realize that single token features do not adequately capture dependencies that clearly appear at the syntactic level (for instance the one between "*don't*" and "*pay*" in the former example).

Using some syntactic parser, you are able to transform the former example sentence

My salary is about 10'000 CHF and I don't pay any tax.

into:

own_salary to_be about AMOUNT-10K-RANGE and I_PRON not_pay any tax

a) What further post-processing, if any, would you suggest on this output?

b) What probability would then be computed as the resulting output by the Naive Bayes model in such a case?

c) Compared to answer ④**b)**, what is the main fundamental reason why you can reasonably expect the results to be better?

Answers: **a)** We could keep on proceeding along the same lines (simplifying) and, for instance:

(a) remove some stop words like "to_be", "I_PRON", "and", etc.; maybe "about" (depends on next bullet, but "about" and "RANGE" really seem duplicated information);

(b) why is the curracy removed? add a curracy identifier;

(c) normalize further some expressions: normalized salary, normalized tax;

so as for instance go as far as having as final representation something like: "*salary_10k_CHF_range   tax_0*".

b) the same kind of formula as above except that now n is the number of remaining indexing tokens and f_i are those remaining indexing tokens

c) it increases features independence (key assumption).

⑥ **[11 pt]** To achieve the results described in the former subquestion, you actually make use of a probabilistic bottom-up chart parser.

In this question, we only consider the analysis of the following input sentence:

After refusing to sign a confession, former Nissan chairman Carlos Ghosn will defend himself in a Japanese court.

in which we more precisely focus on the subsequence:

former Nissan chairman Carlos Ghosn

a) Assuming that each punctuation sign is treated as a single token, in what unique cell (identified by its row and column indexes, starting from 1) would the non-terminals corresponding to the above subsequence be stored?

Answer: row= 5   **col=** 8

- Further assuming:
  - that only the two following SCFG rules can apply for filling the cell identified in **a)**:

    ```
    NP -> B   NP       [0.5]
    NP -> ADJ  NP  NP [0.4]
    ```
  - and that there are (among others):
    - 1 interpretation of `ADJ` in cell (row=1, col=8) with a probability of a;
    - 1 interpretation of `ADJ` in cell (row=1, col=17) with a probability of 3×10^{-3};
    - 1 interpretation of `NP` in cell (row=1, col=9) with a probability of $0.3b$;
    - 1 interpretation of `NP` in cell (row=1, col=10) with a probability of $0.3c$;
    - 1 interpretation of `NP` in cell (row=1, col=11) with a probability of $0.3d$;
    - 1 interpretation of `NP` in cell (row=1, col=12) with a probability of $0.3e$;
    - 1 interpretation of `B` in cell (row=2, col=8) with a probability of $0.2ab$;
    - 1 interpretation of `NP` in cell (row=2, col=9) with a probability of $0.15bc$;
    - 1 interpretation of `NP` in cell (row=2, col=11) with a probability of $0.15de$;
    - 1 interpretation of `NP` in cell (row=3, col=9) with a probability of $7.5 \times 10^{-2} \times bcd$;
    - 1 interpretation of `NP` in cell (row=3, col=10) with a probability of $7.5 \times 10^{-2} \times cde$;
    - 1 interpretation of `NP` in cell (row=4, col=9) with a probability of $3.75 \times 10^{-2} \times bcde$;
    - 2 interpretations of `NP` in cell (row=3, col=17) with a maximal probability of 4.82×10^{-4};
    - 2 interpretations of `NP` in cell (row=2, col=18) with a maximal probability of 2.17×10^{-3};

  then:

  b) How many times will the non-terminal `NP` appear in the cell corresponding to the considered subsequence ("*former Nissan chairman Carlos Ghosn*")?

  c) In total, how many interpretations are associated with this/these `NP` non-terminal(s)?

  d) What is the probability of the most probable interpretation(s)?

  e) How many of the corresponding interpretations are most probable?

  Notes:

  1. $0.15 \times 0.15 = 0.3 \times 7.5 \times 10^{-2}$.
  2. If it helps, you can make use of the chart at the bottom of next page (but then, please, provide clear answers outside of it).

Answers: **b)** It appear only *once* (otherwise the algo is exponential) but from multiple interpretations

c) 4 interpretations (1 from `B NP`, and 3 from `Adj NP NP`, one for each possible decomposition of "*Nissan chairman Carlos Ghosn*" into two `NP`s)

d) $0.4 \times 0.15 \times 0.15 \times abcde$ which is equal to $0.4 \times 0.3 \times 7.5 \times 10^{-2} \times abcde$
since this is bigger than the last one which is equal to $0.5 \times 0.2 \times 7.5 \times 10^{-2} \times abcde$

e) Three MPI: each of the 3 interpretations as `Adj NP NP`

After the introduction of `Y -> Adj NP` for CNF transformation, the table looks like this:

`NP`				
`Y`	`NP` (3.75×10^{-2})			
`Y`	`NP` (7.5×10^{-2})	`NP` (7.5×10^{-2})		
`Y` `B` (0.2)	`NP` (0.15)	`NP` (0.15)	`NP` (0.15)	
`Adj`	`NP` (0.3)	`NP` (0.3)	`NP` (0.3)	`NP` (0.3)

QUESTION V : Document excerpts retrieval [11 pt]

A document excerpt retrieval system, i.e. an information retrieval system retrieving document excerpts instead of full documents, is implemented as follows:

1. each document present in the available document collection is considered as the concatenation of consecutive non-overlapping windows of N consecutive words;
2. each of the word windows generated for any given document is represented using a standard vector space model;
3. the cosine is used to measure the similarity between two consecutive word windows;
4. for any given document, excerpt frontiers are placed:
   - at the beginning and the end of the document;
   - and between any two consecutive windows that have a similarity smaller than a predefined threshold S_{min};

   and the produced excerpts are any sequence of words between two excerpt frontiers;
5. the resulting collection of document excerpts is then processed by a standard vector space information retrieval system with document excerpts being treated as full documents.

① **[5 pt]** For each of the steps 2 and 5, indicate how the tf-idf scores should be computed. Clearly define all the notations you use (if any). The goal is to emphasize the difference between the two cases.

Answer:

- **in step 2:** tf.idf(t,w) = tf(t,w) $\times$ idf(t), where:
  - t : indexing term
  - w : a window
  - tf(t,w) = nb of occurrences of t in w
  - idf(t) = log(|W| / |W(t)|), where |W| is the total number of windows and |W(t)| the number of windows containing the term t.

  To define the windows to consider, there are 2 options:
  - either all the windows in all documents (global processing);
  - or only the windows in each of the documents (local processing)
- **in step 5:** tf.idf(t,e) = tf(t,e) $\times$ idf(t) where:
  - e : an excerpt
  - tf(t,e) = nb of occurrences of t in e
  - idf(t) = log(|E| / |E(t)|) where |E| is the total number of excerpts and |E(t)| the number of excerpts containing the term t.

② **[6 pt]** As, in the proposed approach, a document excerpt retrieval system is fully specified by the (N, S_{min}) parameter pair used to produce the excerpts, the quality of a given (N, S_{min}) parameter pair is measured by the performance of the corresponding excerpt retrieval system.

Indicate how the evaluation methodology defined for usual information retrieval systems may be used to search for a good performing (N, S_{min}) parameter pair.
Describe the various steps of the method you propose and indicate, for each of them, the potential difficulties you may be faced with.

Answer:

(a) choose a document collection, and a relevant set of queries

(b) for each (N, S_{min}) pair (e.g. through a grid search):

- build the corresponding collection of excerpts;
- build the referential, i.e. for each of the queries, ask human experts to identify the relevant excerpts;
- compute a performance metric (e.g. F-score, Precision, or Recall)

(c) Select the best performing (N, Smin) pair

Main issue: the referential must be built for each of the (N, S_{min}), which is very time consuming (and costly)

Other standard issues are also to be considered, e.g.:

- the choice of the performance metric is sensitive (and also often task dependent)
- documents and queries must be relevant for the task;

QUESTION VI : Wikipedia Lexicon [28 pt]

For some target application, you want to process Wikipedia pages in English. To do so, you are first concerned about the vocabulary used.
Here is a typical sentence[1] that you have to cope with:

> *The fossils of Parka decipiens seem small circular or irregular patches reaching a diameter of 0.5–7.5 centimeters (0.20–2.95 in), with a Y-like scar.[4]*

① **[10 pt]** For the following sequences of characters, explain how (or where) they would be represented in a typical NLP architecture or, for sequences you consider as unusual, explain how they should be specifically handled.

- *The*: stored in the lexicon ;
  discuss the capital letter , and maybe the sentence spliter
- *Parka decipiens*: OoV ; handled through NERs (regexp) or specialized lexicon ; discuss the fact that there are two tokens (compound) ;
- *patches*: stored in lexicon or morphological analysis (plural);
- *diameter*: stored in lexicon (morphological root) ;
- *(0.20–2.95 in),*: tokenization (parenthesis, comma): ; OoV ; NER(range, units):
- *Y-like*: OoV ; regexp or specialized lexion
- *scar.[4]*: tokenization : sentence spliter + ad-hoc refenrence handler ; *scar* itself: lexicon

Note: I don't understand why several students here focussed on IR-like applications.

② **[10 pt]** Assume that the following NLP software modules are already available:

1. a general English Part-of-Speech tagger operating on English sentences (one sentence at a time);
2. a specialized tagging software, based on regular expressions, which is able to identify physical measures; for instance to transform
   "*a 43 meters high building*"
   into
   "*a <metric-value value="43" units="m">43 meters</metric-value> high building*",
   or
   "*within a 4.5–5.3 µm range*"
   into
   "*within a <metric-range from="4.5" to="5.3" units="µm">4.5–5.3 µm</metric-range> range*";
3. a specialized tagging software for cross-references, able for instance to transform
   "*mouse.[1] The*"
   into
   "*mouse. <reference>[1]</reference> The*";

[1] inspired from real Wikipedia

4. a (general purpose) sentence spliter (adding end-of-sentence tags, for instance "*<EOS/>*").

a) In what order would you use the above software modules in your processing chain? Briefly explain your design.

b) What other pieces of software do you think you would need to improve Part-of-Speech tagging of Wikipedia pages containing sentences like the typical example sentence provided at the beginning of this question VI?

Explain what each new piece of software should do and where you would add it in the processing chain you defined in **a)**.

c) For illustrating **b)**, how would the example sentence:

> *The fossils of Parka decipiens seem small circular or irregular patches reaching a diameter of 0.5–7.5 centimeters (0.20–2.95 in), with a Y-like scar.[4]*

be represented once processed by your full processing chain?
Here, we don't expect precise values for the tags; if needed, you can simply denote them by `T1`, `T2`, etc. Also, feel free to remove all the information you think you should remove, but, if you do, briefly explain why you removed it.

Answers: **a)** Sentence splitting must come before PoS tagging; depending whether or not cross-reference tagging and sentence splitting perturbate each other, the order shall either be: 4–3 or 3–4 with maybe some glue/tag-cleanning code in between; in the most general case I would say: 3 + cross-reference tag removal (to be re-added later) + 4.

Then same story between 1 and 2: it seems more obvious here that one could perturbate the other, so running them in parallel + add some merging code would be the most appropriate design. But putting them in a raw without any adaptation would surely fail.

If 2 comes before 1, it could then be placed also before 3–4 (that would maybe even improve full stop handling for 4). The interference between references and physical units is unclear but could exist if, for instance, reference like "*[3–5]*" are present.

b) As explained above, certainly some glue code to remove perturbating information between modules, then re-add it later if needed by the application.

Also, as explained in ①, more specific NERs (e.g. to deal with "*Paka decipiens*" or "*Y-like*") would be usefull.

And finaly, depending on the application, further normalization (or even removal) of the tag information (content of tags) might be useful.

c) A typical result would be

the/`T1` *fossils*/`T2` *of*/`T3` *Parka decipiens*/`T4` *seem*/`T5` *small*/`T6` *circular*/`T7` *or*/`T8` *irregular*/`T9` *patches*/`T10` *reaching*/`T11` *a*/`T12` *diameter*/`T13` *of*/`T14` *METRIC_RANGE*/`T15` `(`/`T16` *METRIC_RANGE*/`T17` `)`/`T18` `,`/`T19` *with*/`T20` *a*/`T21` *Y-like*/`T22` *scar*/`T23` `.`/`T24` `</EOS>` *[4]*/`REFERENCE`

③ **[3 pt]** You realize that the corpus you received has actually been corrupted by Wikipedia excerpts from other languages. You plan to filter these out by using a language identification module using a 5-grams Markov model.

Within this 5-grams Markov model, what would be the probability of "*irregular*" to be English? Express this probability as a formula using the parameters of the model.

Answer:

$$\begin{aligned} P(irregular|\text{ENG}) &= P(irre|\text{ENG}) \cdot P(g|irre,\text{ENG}) \cdot P(u|rreg,\text{ENG}) \cdot P(l|regu,\text{ENG}) \\ &\quad \cdot P(r|gula,\text{ENG}) \\ &= P(irreg|\text{ENG}) \cdot P(u|rreg,\text{ENG}) \cdot P(l|regu,\text{ENG}) \cdot P(a|egul,\text{ENG}) \\ &\quad \cdot P(a|egul,\text{ENG}) \cdot P(r|gula,\text{ENG}) \\ &= P(irreg|\text{ENG}) \cdot \frac{P(rregu|\text{ENG})}{\sum_X P(rreg\text{X},\text{ENG})} \cdot \ldots \cdot \frac{P(gular|\text{ENG})}{\sum_X P(gula\text{X},\text{ENG})} \end{aligned}$$

④ **[5 pt]** Using K. Oflazer 1996 spelling error correction algorithm presented during the semester,

a) what are the values (a), (b) and (c) in the highlighted cells when computing the edit distance between "*desipiens*" and "*dispense*" using insertion, deletion, substitution and transposition?
Briefly justify your answer (e.g. by filling a few appropriate cells).

		d	i	s	p	e	n	s	e
	0	1	2	3	**4**				
d		0							
e		1			3				
s					2				
i				2	2				
p					**2**				
i					3				
e						3			
n							3		
s								3	**4**

b) will the search for corrections at distance 2 of "*desipiens*" continue once the prefix "*disp*" has been considered? Justify your answer.

Answer: Yes it will since the cut-off is 2 (min over the pink+grey area) and is equal to threshold. Search stops when *strictly* bigger than threshold.

Note: don't confuse actual distance and "cut-off edit distance"!