**Lehrstuhl für Informatik 6**
**Rheinisch–Westfälische Technische Hochschule Aachen**
**Prof. Dr.–Ing. H. Ney**

**June 9, 2015**
**A. Guta, M. Nuhn**

**7. Exercise Sheet** **Statistical Methods in Natural Language Processing**

---

The solutions to the problems may be submitted until **Tuesday, June 16, 2015** before the exercise lesson. Either upload a digital version to the L2P, or bring your written/printed solution to the exercise lesson. Condition for obtaining the **Leistungsnachweis** (*Schein*) "Statistical Methods in Natural Language Processing" is the successful solution of 50% of the problems and the presentation of the solution of at least two problems in the exercise lessons.
The solutions to the problems can be submitted in groups of up to **two** students.
For programming exercises:

- The implementation has to be done in C++, but you can use additional standard Unix tools (tr, sed, awk...) for the preprocessing.
- Implementations have to be uploaded to L2P as well as printed out and submitted before the exercise lesson.
- Upload the programs as a `.tgz` or a `.zip` compressed directory to L2P. It must include a `Makefile` and must compile on Linux using simply the `make` command.
- Include a *short* description of the main data structures and algorithms used in your solution sheet.

1. [4 Points] For the symbol based Bayes decision rule for the spelling correction problem, i.e.

$$\hat{c}_n = \arg\max_c \{\Pr_n(c|x_1^N)\}$$

with the marginal probability distribution

$$\Pr_n(c|x_1^N) = \sum_{c_1^N : c_n = c} \Pr(c_1^N|x_1^N)$$

design an efficient algorithm for computing this sum. You may assume a bigram language model.

2. [6$^+$ Points] In this exercise you will implement a (2-gram based) spelling corrector and study its results. Provide as many results as you consider adequate (tables/graphs, ...).

   a) Implement a method to calculate the perplexity of a given test sequence $c_1^N$. The perplexity of a sequence $c_1^N$ is defined as:

$$PPL = p(c_1^N)^{-\frac{1}{N}}.$$

   You can either use a given software library (SRILM[1] or KenLM[2]) to read the file `en.lm.2gram.gz` (see also Exercise 5) or use the provided simplified language model `simple_lm.hh`[3]. Make sure that you make use of the sentence begin/end contexts (using the `<s>` and `</s>` tokens).

   Which one is the sentence in the `test_data` file with the lowest perplexity? What is the sentence with the highest perplexity? What are the values of these perplexities? Make sure that you properly convert the scores (log probabilities) into real probabilities when calculating the PPL.

[1] http://www.speech.sri.com/projects/srilm/
[2] https://kheafield.com/code/kenlm/
[3] See comments for more details

b) Given a sequence $c_1^N$ generate a random sequence $x_1^N$ with probability

$$p(x_1^N|c_1^N) = \prod_{n=1}^{N} p_\lambda(x_n|c_n) \quad \text{with} \quad p_\lambda(x|c) = \begin{cases} \lambda & \text{if } x = c \\ \frac{1-\lambda}{|X|-1} & \text{if } x \neq c \end{cases}$$

What are reasonable values for $\lambda$ given the fact that the probability of $p(x|c)$ with $x = c$ should be greater that $p(x|c)$ with $x \neq c$?

c) Given two sequences $c_1^N$ and $\tilde{c}_1^N$ implement a method to calculate the accuracy $\frac{1}{N}\sum_{n=1}^{N} \delta(c_n, \tilde{c}_n)$. Make sure that your implementation returns accuracy 1.0 when $c_1^N = \tilde{c}_1^N$.

d) Implement an efficient algorithm for finding $\arg\max_{c_1^N} \left\{p(c_1^N, x_1^N)\right\}$ with the model

$$p(c_1^N, x_1^N) = \prod_{n=1}^{N} p_\lambda(x_n|c_n)p(c_n|c_{n-1})$$

Reuse the LM, $\lambda$-model and evaluation code from the previous parts. For $\lambda \in \{0.1, 0.2, \ldots, 0.8, 0.9\}$ run the following experiments:

- for all sequences $c_1^N$ in `test_data` add noise with $\lambda$ to obtain $x_1^N$.
- run minimum string error rate spelling correction on $x_1^N$ to obtain $\tilde{c}_1^N$.
- report the error rate reduction from $x_1^N$ to $\tilde{c}_1^N$.

For which values of $\lambda$ is a reduction in error rate possible?