Tokenizing

In very general term, a token can be seen as a word. Every sentence consists of different words, from a technical point of view we call them tokens. However, that is not all there is to a sentence, there is punctuation, too, and maybe a number, an emoticon, a line break etc. All these are tokens, too. Wikipedia describes a token as "… a structure representating a lexeme that explicitly indicates its categorization for the purpose of parsing." Even though this approach is very technical, it also gives an insight into the problem we faced when tokenizing the SMS corpus. One of the basic question when creating a corpus is: *which entities might a linguistic researcher be looking for?* Splitting a sentence into tokens is one step that has to be performed when creating these entities.

A out-of-the-box tokenizer, as they exist in computational linguistics, identifies every punctuation as a token, because they are normally found between words and separated by spaces and are used to separate clauses. The combination ; -) would thus consist of three tokens, a semicolon, a dash and a round closing bracket. However, in SMS research, these three characters should not be considered as individual tokens of punctuation but as a single token that forms an emoticon. In this situation, three characters that would normally be considered as individual tokens have to be pulled together to form a unity. A similar situation can be found in ordinary French spelling, where *pomme de terre* ('apple from the soil', i.e. 'potato') should not be considered as three, but rather as one token. But we might also find the opposite problem, e.g. with clitic forms: from a syntactic point of view Swiss German, *hani* ('have I') should be interpreted as two tokens: *han* and *i* even though they are not separated by any character.

When tokenizing the SMS corpus, an ordinary tokenizer, as it is used in computational linguistics, was applied to the data with special rules, e.g. for emoticons. In a second step, the student helpers (while performing other tasks) checked all the tokens and marked them for correction where applicable. Please consider the following examples to illustrate the rules applied:

Language	automatic tokenization	corrected to	Translation
French	[jsuis]	[je][suis]	I am
French	[ajourd'][hui]	[ajourd'hui]	today
all	[n][8]	[n8]	night
Romansh	bn	[Buna][notg]	good night

Please consult Ruef/Ueberwasser (2013) in the bibliography for more information about the technical steps of the tokenization.