A Reasonably Language Independent, Heuristic Algorithm for the Marking of Names in Running Texts

Benny Brodda
Stockholm

0. Introduction

The identification of names and abbreviations in a raw text is an important subactivity in the "tokenization" process, i.e. the identification of the basic units of the text: paragraphs, sentences and words. Tokenization is in its turn an important subactivity in the "normalization" of texts, the totality of operations and preparations a text has to undergo before it is suitable for being added to a text corpus.

This specific paper deals indirectly with the sentence recognition problem . In most cases a configuration in a text of the type

(1) major_delim + word_interspace + up-case_letter_+ one or more low-case_letters;

indicates the end of one sentence and the beginning next, where a "major delim(iter)" stands for any of the characters '.', '!', '?', ..., and a "word interspace" a non-empty string of "white" characters (spaces and/or a linebreaks). There are, however, quite a few exceptions to this rule, and one important set of such exceptions involves combinations of abbreviations and names, e.g. *'...; cf. Brodda 1998* ' or *'As H.G. Wells writes in ....*' Neither *Brodda* nor *Wells* are here sentence openers. There are many other ways sentence breaks may be indicated (or not indicated) in a text, and there are other types of exceptions (or "tricky configurations") but the mentioned configuration in (1) certainly provides for the vast majority of cases.

When starting this project I set for myself a kind of specification: In principle I should be able to take a raw text of any kind and in any language, "throw" it through my algorithms and get the names in it marked. Now, it is never as simple as that. One always has to make at least some preliminary investigations, (finding out which character set is used, which word delimiters there are, if the words are hyphenated, if the texts contain printing codes, etc.) and some preliminary preprocessings (making something useful out of possible printing codes and/or remove them, fuse hyphenated words, perform a preliminary paragraph separation, etc.)

But still, in principle, if these preliminary preprocessings are done, I think I can do precisely this. I have made extensive test for Swedish, where my algorithms work quite satisfactory, and I have performed some preliminary tests on French, Italian and English, with comparable results so far. We have also tested it on Estonian, where the results were not that impressive (but still useful). I have not tested it on German, but that would probably be a waste of time; as long as they insist on spelling their nouns with capital initial letters my algorithms just won't work; Sorry, Germans.
As we will see, for Swedish I can reach a recall rate of at least 98% and a precision rate of about 95+%.

All computer programs used in the experiments reported here are written in the PCBeta programming system; cf. Brodda, 1998, forthcoming.

1. Preliminaries

In this paper I will mainly consider **prototypical** names, names that adhere to what I call the **Ux-format**, one upper case letter followed by one or more common letters. Furthermore, prototypical names should have a **(reasonably) stable orthography** in the sense that they always retain the Ux-format, and they only take some restricted types of "bending"; in Swedish and most other west European languages quite regularly the genitive, (in Swedish and in the other Nordic languages by just adding an *s* to the name: *Paul, Pauls*). Commercial names for physical objects tend to become used as common nouns and are correspondingly flexed: *Saab, Saaben, Saabar, Saabarna*. This type of flexion is a marginal problem, and I will not account for it here.

Of course there are other types of names than the prototypical ones. 'USA', for instance, represents one important type of names, which I will leave out here (they pose a minor problem). More troublesome are names of intellectual works, which names sometimes occur directly in the text without surrounding quotation marks (as they probably should have, if the proof reader had done his/her job properly). '*Dylan writes in The Times they are Achanging, that ...*'. Now, neither of the last three Ux-words have a stable orthography (in an English text). At least the word *The* would certainly appear in the text(s) also as *the*, and correspondingly for the word *Times* (maybe). For the word *Achanging* there is a lesser chance of finding the word *achanging* in the text(s); it would probably be accepted as a name according to my algorithms. Thus, what is accepted as a name may depend on the type of texts we are investigating and the total amounts of them.

The words *The*, *Times* and *Achanging* in the example above are examples of a type which I will call **formal names,** words adhering to the Ux-format and appearing in an **interior position**; at least one word (or correspondingly) away from either a paragraph beginning or the nearest major delim(iter) to its left. The intended meaning of "interior position" is, of course, "sentence interior position", but as I intend to use my name identification procedure(s) as an aid for sentence detection, I can not, at this point, assume that I know which these are. The "interior position"-concept, is, by and large, stronger than any reasonable meaning of "sentence interior". The addition "or correspondingly" above is to account for situations of the type '..., *cf., e.g., Brodda 1998*' where *Brodda* now is interior (',' should not occur in sentence beginnings), whereas
this word is not interior in the corresponding example in the Introduction.

Now a few words about "major delim(iter)s" and about "what a word is". If a sentence begins with '*10:15 this morning Benny Brodda presented his epoch-making theory on name identification*' then '*10:15* 'is, of course, a word; it is part of a topicalized time adverbial. If you have an itemized list of the type '10:15 *Benny Brodda on a name identification method*; 11:00 *Kimmo Koskenniemo on FS-parsing*; 11.45 ...(etc.)', then it is not at all clear whether '10:15' should be considered as part of the following sentence or not. (Personally I would say

not). Properly speaking, the time specifications here should normally be delimited by something (a full stop, a colon, or so), but the problem is what you do if they are not.

The characters usually assumed to belong to the set of "major delims" are '.', '!' and '?', but there are other characters which to a varying degree have similar functions, at least in certain situations. The most important of these are ':' (colon) and ';' (semicolon). Especially if these characters occur in configurations like (1), above, then the best procedure is to consider them to be major delims. In Swedish writing the ':' in a configuration of the type '... *xxx:* "*Uxx xxx* ... .' most certainly indicates that the Ux-word commences a sentence-like object; this is the ordinary way direct quotations are written. In the procedure described below I will to begin with take the conservative position that a quote-in character always is equivalent to a major delimiter, if the quoted stretch of text comprises more than one word. Dash, '—', finally, in ASCII-texts often occurring as an ordinary hyphen '-' surrounded by white characters, is again a character with somewhat unstable use as a sentence begin/end indicator; I assume that it is, if it occurs before an Ux-word.

2. The test texts

The big idea with my investigation is to (pre)process raw text, with no markings in it to begin with. In order to be able to evaluate my name identification algorithm, I needed, however, a text with the names already marked. Luckily, we had precisely that at my department, viz. a substantial - though preliminary - subset of the so called SUC-corpus (SUC = the Stockholm-Umeå Corpus; cf. Källgren: *www.ling.su.se/forskning/ SUC/distr&anv.htm*. The texts in SUC are SGML-marked (according to the Text Encoding Initiative conventions), meaning that they contain a lot of spurious information for my specific purposes, so I extracted from it the text itself together with (a transformed variant of) the name tags. I also kept the paragraph structure (essentially in the form of an empty line after each paragraph.) Here follows a small but rather typical sample of such a text. (The asterisks do not occur in the SUC-text itself. They are the results of my algos.)

```
aa03a 035 bådas framtida toppmöten kommer att minska .+ Med hänvisning
aa03a 036 till att relationen mellan [ USA ] och [ *Sovjetunionen ] nu
aa03a 037 undergått avgörande "kvalitativa" förbättringar beslöt sig
aa03a 038 [ president *George *Bush ] och *Sovjetledaren [ *Gorbatjov ]
aa03a 039 från och med nu att hålla regelbundna , sannolikt årliga ,
aa03a 040 ... +
```

Fig. 1: A text sample from one of the input texts.

"aa03a" in the line beginnings is the name of the corresponding (sub)text in SUC, and "035, 036, etc." are the (temporary) line numbers within that text I use during my experiments. The '+'-signs in the text above are what remain of the sentence-in and sentence-out tags in SUC; here they are strict sentence end markers. The '[' and ']' characters correspond to the name-in and name-out tags of SUC, respectively. My name marking programs just disregard the characters mentioned in this paragraph.

Altogether I have now a corpus of some 300,000+ words, but for the experiment reported here I have used a subset of some 77,000+ running words.

When evaluating my procedures I do, of course, take the name-in and name-out characters into account - they delimit a "name region" - and I can immediately see which words are "false hits", "true hits" and "misses". The last two concepts are defined relative to what I call **SUC-names**, Ux-words occurring inside a name region. **True hits** (or just hits) are the marked SUC-names and **misses** the non-marked ones. (NB! 'USA' in Fig. 1 is **not** a miss; cf. last para of this section). The **false hits** are the asterisked words outside any name region (e.g., '**Sovjetledaren*', "the Soviet leader"; in Swedish a compound like that may or may not be spelled with a capital first letter, but either way it should not be counted as a name. (The English counterpart is also a compound, but I think it is not entirely wrong to consider the word 'Soviet' in it to be a name.)

In the evaluations of my algorithm(s) I use the standard descriptive statistics **recall** and **precision** - originally emanating from the I/R-world - and (here) defined as:

(2) recall = (number_of_hits) / (number_of_SUC-names);
precision = (number_of_hits) / (number_of_hits + number_of_false_hits);

Both these statistics are commonly expressed in percent: 100% recall (here) = all the SUC-names were actually "recalled", i.e asterisked; 100% precision = all asterisked words are also SUC-names (= no disturbing "noise" in the output).

Before I proceed I must explain very carefully what my claims are. For the time being I only consider Ux-words, meaning that I here simply disregard words like "USA". (In the next warp I will try to catch such names, too). Words like *president* in *president George Bush* the SUC-people think is part of the name, but I will not even try to catch them; I simply think, that their view is wrong. Another thing that I have not tried to catch so far, is that name combinations like *George Bush* actually refer to one and the same object, the **constellation** must be considered as one name. At present they are counted as two separate names. I have done some preliminary testing for identifying such complex names, and for the moment the best - in terms of recall and precision - à priori procedure seems to consider **all** name meetings as being coreferential, but this is too early to say. Peculiarly enough, the presence of certain non-Ux-words seems to enhance such affinities: *Otto von Bismarck*, *Louis de Funès*, *Jan van der Velde*, *Gabriel de la Gardie*, etc. Such words or word combinations I call **name continuators**, and I have about 10 of them in a permanent lexicon, which is used in the procedures described in section 4.1 and 4.2. (In fact, this is the **only** lexicon I use, besides the temporary ones created during the processing; cf., e.g., next section).

The name marking is now performed in five distinct steps, two steps when looking at single words in isolation, two when immediate contexts are taken into account and one final left to right pass, marking such formal names which for some reasons have not been marked, so far.

3. Looking at names in isolation

The basic here idea is to extract all formal names in the text(s), make a lexicon of them, and then use that lexicon for marking (alledged) names occurring anywhere in the text, including their original positions.

3.1 Step 1: The marking of formal names

The very first step is to mark all formal names in the text; this is the basis for all later processings. The text I use here is a text called AA-AF.TXT, which comprises 77,000+ words of running text, and the result text is called AA-AFM0.TXT. When running the evaluation program on this text, I get the following basic statistics (Fig. 2).

```
LOG Output AA-AFM0.LOG   date:  28/2 1998
RuleFile:     NAMELOG.RUL
TextFile in:  AA-AFM0.TXT,  Lines in:  13764

Event statistics!
wds          77664          %%total # of wrds
nms           5575          %%total # of SUC-names (Ux-words only)
fse            254          %%false hits: only marked by rules
mss            589          %%misses: only SUC-marked
hts           4986          %%hits: both marked by rules and by SUC
rcl:          89.4          %%recall:     100*hts/nms
pre:          95.2          %%precision:  100*hts/(fse+hts)
```

Fig 2. The evaluation of text AA-AFM0.txt: formal names marked

Please, observe, that the recall rate as given in Fig. 2 is for the whole text. If only interior areas are considered - i.e. those words which Step 1 comprises - the recall rate is considerably higher, viz. about 98%.

Among the 254 false hits about 220 are compounds of the type **Sovjetledaren* (cf. sect. 2), and I will return to a comment about such "errors" in the final section.

3.2 Step 2: Apply the information obtained in Step 1 to non-interior Ux-words.

3.2.1 Step 2.1: "No morphology".

As a preliminary exercise we now extract all marked words in AA-AFM0.TXT, make a lexicon of them and mark those Ux-words in AA-AFM0.TXT that were not marked in step 1, i.e. essentially words in sentence beginnings, we obtain a remarkably high recall - viz. 98.1% - but at a cost of a rather bad precision 78.1%. If we only consider such words that this step actually comprises, i.e non-interior words, the result is a near catastrophy. We get a precision rate of only 29.1%, which means that more than 70% of all freshly marked words are non-names. Not very impressive.

The source of this disaster are to be found among "hits" in Step 1 of the following types:

```
ac04c 014 köper nu ut [ *Den norske Bank ] , [ Union ...
ae07d 024 ar ingen roll *Det är det som är det fina m...
```

In the first example *Den* in *Den norske bank* ("The Norwegian Bank") is (part of) a name, which so far is OK (i.e. in Step 1). In the second example, *Det* is marked because of a printing error in the SUC-text (or, raher, in the original raw text). *Det* here simply opens a new sentence, and there should have been a full stop before it. (As said, my test texts are derived from a preliminary version of the SUC-corpus. In fact, using Step 1 and then a search for false hits in interior positions is an excellent way of locating errors of the mentioned type, thus providing valuable up-date information for the final version.

Now, the words *Den* and *Det* are both ordinary words and, incidentally, also very common. Furthermore, they are typical "sentence openers", meaning there is a good chance to find them in sentence beginnings and spelled with a capital initial letter. Altogether there was a handfull of different words of that type marked in Step 1, and these few words together totally destroyed this simplistic approach.

This step is now modified in the following way. I compare the frequencies of all formal names obtained in Step 1 with the frequencies of the same words spelled with common letters. Then I keep only those formal names that are more common as formal names as they are as common words. Simple.

After step 2.1: recall = **95.8%** ; precision = **95.8.%.**

3.2.1 Step 2.2: Now taking morphology into account.

As said in Preliminaries I will assume that names can only take the genitive as flexion, which in Swedish is a simple: an -s added to the name (and no apostrophe): *Paul, Pauls*, and if the name already ends in an -s, then nothing is added: *Nils, Nils*.

The procedure is now: For every formal name found in the text, remove an -s if it ends in one, add an -s if it does not, and then treat the two forms as separate words as in Step 2.1, i.e. match the frequency of them as names against the frequency of them as common words, and knock those name forms out that had a lower frequency as their common counterparts. If the two forms of an alleged name both survive this sieve, they are again fused and treated according to a default rule, saying that a found item may or may not have an extra -s on it in order to be accepted. Thus *Nils* occurs in the lexicon only as *Nil,* but treated according to the default rule, (no *Nil* was found in the text, but who cares). *Svan*, which is a common Swedish family name survived this ordeal (a few *svan*, "swan", were not many enough), but *Svans* did not; a few *svans*, "tail", knocked that variant out. For the common Swedish first name *Hans* neither form survived: *hans,* "his", knocked *Hans* out, *han,* "he", knocked *Han* out; the latter was less catastrophic, because no *Han* actually appeared in the texts. In Step 3 and 4 we will se how to save *Hans*.

The algorithm in this final version of Step 2 was also modified in order to account names like *d'Aille* and *Prince's*. The solution here was simply to consider the ' ' to be a word delimiter in such contexts, meaning that no special arrangements needed for them. (I see no problem in using this simple "trick" to solve the genitive problem for English.) Anyhow, the final "scores" for Step 2 are now:

After Step 2: **recall = 96.0% ; precision = 95.7%.**

4. Looking at names in context

4.1 Step 3: "The property of being a name is contagious".

Names often come in "bursts", the earlier mentioned example *George Bush* is by no means rare. On the contrary, they abound: *Hans *Gustavsson*, *Svenska *Handelsbanken* ("The Swedish Business Bank"), *Nya *Zeeland*, **Paris All Stars*, *The International Tennis *Weekly*, **Perstorp *Plastic Systems*, **Sturkö *Design och Rökeri* and many others (The asterisks here are those put therein Step 2.).

Step 3 is now very simple, we apply what I call "asterisk propagation": An Ux-word in interior position receives an asterisk, if a neighbour of it has one, and this procedure is applied recursively. The "infection" may also spread over "name coninuators", (cf. sect. 2), and *och* is one of them. Through this mechanism all the unasterisked Ux-words in the examples above now become marked. The tennis journal through a repetitive asterisk left propagation, the Paris stars through a corresponding right asterisk propagation. The word *Rökeri*, "smoke house", receives its asterisk through a rightwards jump over *och*. 13 occurrances, for instance, of the name *Hans* were recovered through Step 3.

After Step 3: **recall = 97.0% ; precision = 95.7%.**

4.2 Step 4: "Once a forename, always a forename".

In this step we define an asterisked word occurring immediately to the left of another asterisked word as a **forename**, possibly with a word continuator in between. We start this step by extracting all forenames, make a lexicon of them, and apply this lexicon to the words in the sentence beginnings (well, in non-interior positions). A found word is then assumed to be a name, if the word after it is asterisked.

There were not many names caught in this process (five for this text, to be precise, including one *Hans*) and there were three false hits. In a larger sample of the SUC-texts, AA-ED.TXT, comprising somewhat more than 300,000 words, the corresponding figures were 57 hits and 30 false hits, so the increase in recall may not be that negligible.

After Step 4: **recall = 97.5% ; precision = 95.7%.**

In the next and final step we again look at interior words in isolation.

5. Step 5: "Formal names are names, after all"

In sect. 3.2.1 we briefly discussed the example

```
ac04c 014 köper nu ut [ *Den norske Bank ] , [ Union ...
```

The word *Den* in this example was asterisked in Step 1, but it lost its marking through the knock-out procedures of Step 2. This *Den* has then stayed unasterisked after that. But *Dendefines*, after all, at least the beginning of a name, and it is a SUC-name according to our definitions. How do we capture this observation?

In this final pass we once more scan the text from left to right and mark so far unasterisked formal names, now with somewhat slackened conditions compared to Step 1. There we excluded Ux-words occurring immediately after a quote-in sign. Now we accept such words unless this character is preceded by a colon (when the Ux-word opens a direct quotation and may or may not be a name). Ux-names now marked in this process are more often than not (the beginning of) either a name of an intellectual work or just an ordinary, complex name, as the one in the example discussed above.

After Step 5: **recall = 98.7%; precision = 95.2%.**

6. Conclusions

We see a slight decrease of precision rate in Steps 3 through 5. This decrease would most certainly be hard to verify statistically, but I think it is significant. The reason is that these steps (or rules) might be a little bit too heuristic, meaning that the number of new false hits will grow relatively faster than the number of new true hits. It might be possible, I think, to marginally refine these steps, but for my purposes the present figures are quite satisfactory.

Anyhow, I have run the same set of rules on the much larger text AA-AD.TXT, and found absolutely comparable results. Theoretically there should, though, exist some optimum text size for applying my algorithms, but my finding so far seem to indicate that the algorithms are very insensitive to text size.

Compounds of the **Sovjetleader* type (cf. sect. 2) represent the by far most common type of false hits, and they considerably contribute to the smaller figure of the precision rate. If you are actually trying to identify all (and only) the names in a text, such words will, of course, create trouble. But in the context of what I am trying to do - identify sentences - the question is, whether the acceptance of such words as names would be so bad, after all. I need to identify words with a stable Ux-spelling, and the mentioned compounds have exactly that.