# Large Scale Keyword Extraction using a Finite State Backend

Miloš Jakubíček[1,2], Pavel Šmerk[1]

[1]Natural Language Processing Centre
Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
{jak,smerk}@fi.muni.cz

[2]Lexical Computing
Brighton, United Kingdom and Brno, Czech Republic
{milos.jakubicek}@sketchengine.co.uk

**Abstract.** We present a novel method for performing fast keyword extraction from large text corpora using a finite state backend. The FSA3 package has been adopted for this purposes. We outline the basic approach and present a comparison with previous hash-based method as used in Sketch Engine.

**Key words:** terminology extraction, keyword extraction, fsa, Sketch Engine

## 1 Introduction

In this paper we focus on the keyword (and terminology, as explained later) extraction task when solved using a system with a contrastive approach, such as the Sketch Engine corpus management system [1]. In this case, the input for this task consists of two arbitrary corpora: a focus corpus from which the keywords should be extracted, and a reference corpus that the term candidates from the focus corpus are contrasted with.

The keyword candidates come from different sources, but in the end the procedure always boils down to a very costly operation of matching all keyword candidates in the focus and reference corpus. While individual corpora are indexed in a database that assigns unique numeric identifiers to each string, hence intra-corpus processing operates on numbers and not strings, inter-corpus processing cannot take of this advantage and the any kind of pre-indexing (e.g. of particular corpus pairs) is not very flexible as systems like Sketch Engine deal with thousands of corpora, and the term extraction functionality is often used with user corpora built on-demand.

To speed up the process of string comparison, we present an approach that builds on intersecting two finite state automata. We show that this approach is more efficient both in space and time. We describe both the old method used in Sketch Engine and this new one and conclude by a comparison on a set of scenarios.

Aleš Horák, Pavel Rychlý, Adam Rambousek (Eds.): Proceedings of Recent Advances in Slavonic Natural Language Processing, RASLAN 2016, pp. 143–146, 2016. © Tribun EU 2016

## 2 Keyword Extraction in Sketch Engine

Sketch Engine contains a keyword and terminology extraction module [2] using a contrastive approach to find term candidates. Two corpora are given as input to the term extraction: a focus corpus consisting from texts in the target domain, and a (ideally very big) reference corpus which the focus corpus is compared to. Sketch Engine currently contains reference corpora for over 80 languages.

The elementary units for the extraction can be one of the following three:

1. **positional attributes** in the corpus, such as word forms, lemmas or part-of-speech tags,
2. **terms** as identified by the language-specific term grammars, e.g. noun phrases,
3. **collocation lists** represented by triples of *(headword, relation, collocate)* as derived from the word sketches.

In each case the system first extract all candidates from the focus corpus so as to be able to compare their relative frequencies (or other statistic) with the reference corpus.

### 2.1 Previous approach

The previous approach as used in Sketch Engine was based on a string-to-string comparison in the case of positional attributes, and comparison of pre-indexed fixed-length string hashes in the case of term and collocation lists. Especially the latter case suffered from a number of deficiencies:

- pre-indexing of string hashes was costly both in space and time. E.g. for a English corpus enTenTen12 [3] which has almost 13 billion words, the collocation list hashes occupies 2.2 GB (each hash being a 64bit binary). This is a problem especially with a cold disk cache when the whole file needs to be read into the memory for any comparisons.
- even the comparison of hashes took a long time (e.g. the comparison of the collocation list between the British National Corpus [4] and the enTenTen12 still took about 2 minutes with a cold disk cache, and about 13 seconds with hot disk cache.)

### 2.2 Finite-state based approach

To overcome the disadvantages described above we have designed a new method based on finite state automata (FSA). Instead of pre-indexing any hashes, for all the three source types we pre-index a minimal FSA containing all the strings. We use the FSA3 package[1] which can efficiently build a minimal FSA and provides a string-to-number and number-to-string mapping of each

[1] See `http://corpus.tools`.

stored string (where the numbering corresponds to enumeration of all strings sorted lexicographically).

Initial experiments with FSA-based string-to-number and number-to-string mappings were described in [5]. The FSA3 package is inspired by the tools for automata generation and minimization developed by Daciuk [6]. Alongside of the development described below, we have significantly improved compile-time performance of the FSA3 package which is now about 10 times faster compared to what was provided in [5] and has linear complexity with regard to the input data size. While the compile-time performance is not crucial for the keyword extraction task (where the compilation is performed only once per corpus) it is an important aspect for other tasks where automata need to be often recreated. A detailed report on all findings relevant to the automata compilation and minimization will be provided in a separate paper.

We have extended the FSA3 package by the intersect operation on two automata (denoted as $\mathrm{FSA}_1$ and $\mathrm{FSA}_2$), which can:

1. output all strings present in both $\mathrm{FSA}_1$ and $\mathrm{FSA}_2$ together with their respective numeric IDs in both automata (we call this an **intersect operation**).
2. output all strings present in $\mathrm{FSA}_1$ with matching numeric IDs or just the ID from $\mathrm{FSA}_1$ where the string was not present $\mathrm{FSA}_2$ (we call this a **left intersect operation)**.

The left intersect operation allows these automata to be directly exploited in the keyword extraction task so as to obtain a list of matching strings and IDs which can be used to retrieve pre-indexed frequencies from the individual corpora (where the string is present only in $\mathrm{FSA}_1$, the frequency in $\mathrm{FSA}_2$ is obviously zero).

## 3 Evaluation

We have conducted a number of comparisons of the hash-based and finite-state based approach using different usage scenarios and string sources. For the evaluation we used three corpora: the BNC (100 million words), the enTenTen12 (13 billion words) and enTenTen15 (30 billion words).

All results are summarized in Table 1.

The evaluation shows that the FSA-based approach is faster for all hot-cache scenarios. The slowdown for the cold-cache scenario was, after a detailed inspection, caused by the fact that the hash indices stored far less amount of data (ca 250 MB) because of filtering out items with frequency lower than 1 per billion words. Therefore, this comparison cannot be seen as representative.

## 4 Conclusions

In this paper we have presented a novel method for extracting keywords from very large (billion word) corpora that is based on finite-state machines. The

Table 1: Evaluation of hash-based and FSA-based keyword extraction

| **string source** | **corpus$_1$** | **corpus$_2$** | **FSA$_1$ size** | **FSA$_2$ size** | **page cache** | **time$_{prev}$** | **time$_{now}$** | **speedup** |
|---|---|---|---|---|---|---|---|---|
| lemma | BNC | enTenTen15 | 556k items 4 MB | 26,426k items 340 MB | cold | 80.2s | 51.4s | 1.56x |
| | | | | | hot | 6.3s | 0.7s | 9x |
| term list | Brown family | enTenTen12 | 320k items 4 MB | 164,189k items 2 GB | cold | 1m11s | 2m10.2s | 0.54x |
| | | | | | hot | 4s | 1.2s | 3.3x |

evaluation shows promising results so that the method is going to be adopted for the Sketch Engine corpus management system so as to be able to carry out practical results from a production environment.

**Acknowledgments.** The research leading to these results has received funding from the Norwegian Financial Mechanism 2009–2014 and the Ministry of Education, Youth and Sports under Project Contract no. MSMT-28477/2014 within the HaBiT Project 7F14047.

## References

1. Kilgarriff, A., Baisa, V., Bušta, J., Jakubíček, M., Kovář, V., Michelfeit, J., Rychlý, P., Suchomel, V.: The Sketch Engine: ten years on. Lexicography **1** (2014)
2. Kilgarriff, A., Jakubíček, M., Kovář, V., Rychlý, P., Suchomel, V.: Finding terms in corpora for many languages with the Sketch Engine. EACL 2014 (2014) 53
3. Jakubíček, M., Kilgarriff, A., Kovář, V., Rychlý, P., Suchomel, V.: The TenTen Corpus Family. International Conference on Corpus Linguistics, Lancaster (2013)
4. Leech, G.: 100 million words of English: the British National Corpus (BNC). Language Research **28**(1) (1992) 1–13
5. Jakubíček, M., Rychlý, P., Šmerk, P.: Fast construction of a word-number index for large data. RASLAN 2013 Recent Advances in Slavonic Natural Language Processing (2013) 63
6. Daciuk, J., Weiss, D.: Smaller representation of finite state automata. Theoretical Computer Science **450** (2012) 10–21