Large Scale Calculations for Creating Polish Language Semantic and Syntactic Models

Abstract

Bartosz Ziółko, Dawid Skurzok, Jan Wicijowski
Department of Electronics
AGH University of Science and Technology
Al. Mickiewicza 30, 30-059, Kraków, Poland
{bziolko,wici}@agh.edu.pl

Introduction to our research

We use Cyfronet high performance computers to process linguistic data in aim of constructing the Polish language models. The results will be applied to a large vocabulary automatic speech recognition system. Natural language processing always faces problems of data sparsity. The quality of language models depends strongly on the amount of text corpora available during the training. This is the cause for a trade-off of quality and time spent on calculations. The high performance computers serve by obtaining the linguistic rules from the huge amount of texts written in Polish.

Word statistics can be applied in order to provide probabilities of appearance of particular words in Polish. It is a difficult task to create such a model, especially for Polish, as the number of possible words (considering rich morphology) is over a million. To provide a statistical model describing regularities between these words, enormous amount of text has to be analysed. An additional difficulty is that such texts may contains numerous typographical errors, foreign words, html tags and so on. Their presence in the corpus increases the amount of basic units drastically.

Our usage of Cyfronet resources

Our calculations were conducted using C++ and Python software prepared by us. We store collected statistics in SQLite databases because there are no available database servers on the clusters we used.

Table 1. Analysed text corpora with their sizes, perplexity.

Corpus	MBytes	Mwords	Basic forms	Perplexity
Rzeczpospolita journal	879	104	832 732	8 918
Wikipedia	754	97	2 084 524	16 436
Literature	490	68	610 174	9 031
Combination of above	2123	267	?	9 199
Transcripts	325	32	183 363	4 374
Literature 2	6500	949	?	?

We started using Mars computer. Our database became too large to be kept on nodes. The temporary hard disk resources are not provided for long enough that we could use them. It would result in loosing data if our processes are longer than 48 hours, what happens frequently. Any communications with external resources is a bottleneck for time efficiency so we avoid it.

Because of the storage problems we moved our database and calculations to Baribal computer. Its resources were good enough for us, but the regular shut downs and malfunctions caused inability to conduct calculations and finish some parts of work between the shut downs.

Then we started to use PL-grid were both resources and continuity of work allowed us to progress with the project. We were also very positively surprised with easiness of opening an account on PL-grid.

Our data and methods

N-grams collection was build on SQLite database. A text file or a part of a file is loaded to 1 MB buffer. Words to build n-grams are taken one after another from the buffer. While loading words from the buffer, every character is checked if it is an ASCII or UTF-8 letter. If it is a character representing an end of a word (like space, tabulator, end of line or dot), the length of the word is stored for later use.

Table 2. The number of different *n*-grams in the analysed corpora.

Corpus	1-grams	2-grams	3-grams
Rzeczpospolita journal	1 275 475	26 390 703	62 440 894
Wikipedia	2 623 358	31 139 080	61 865 543
Literature	1 151 043	23 830 490	50 794 854
Combination of above	3 161 748	59 590 565	143 502 429
Transcripts	381 166	6 848 729	16 283 781
Literature 2	7 032 022	159 614 194	439 272 946

Text is arranged in a sequence of three words, then saved as 1, 2 and 3-grams. Only the first word is saved as 1-gram, the first and the second as 2-gram and the whole sequence is saved as 3-gram. After that, the second word is moved at the beginning and the third one is moved to the second position. The new third word is loaded from a buffer, and

again the sequence is saved. 2 and 3-grams are stored as one string with each word separated by a space. The troubles with the access to files on Mars decreases the time efficiency largely. Every time while adding or updating some n-grams, a database engine reads and writes some data from a file or cache. To improve speed of input/output operations, we have increased cache size, compiled SQL statements before using them and disabled synchronisation between data in memory and on a hard drive. This is why every word is read only once from a corpus. The process of checking if the new word already exists in the statistics is conducted on the database cache rather than on the file for the same reason.

Table 3. Analysed text corpora with their sizes, perplexity and number of n-grams.

Corpus	single 1-grams	%	1-grams with errors	%
Rzeczpospolita journal	560 549	44	26 786	2
Wikipedia	1 426 958	54	108 338	4
Literature	467 376	41	75 204	6.5
Combination of above	1 645 474	52	187 382	5.9
Transcripts	147 440	39	1 373	0.4
Literature 2	4 119 667	58	601 420	8

It was checked that applying multithread algorithm did not improve the efficiency. The problems of the access to files and the necessity of using calculation power on controlling threads are possible reasons. Instead of multithreading, big corpora were split to smaller parts and processed separately, at the same time. Then results were joined together.

We faced another problem which was caused by format of special Polish characters like ó, ł, ę, ż. The same letter is kept in different formats using different bytes. Gżegżółka software was used to change text files from one standard into another and to unify them into UTF-8. However, some parts of files contained unexpected values which looked like they belong to a different standard. It is one of the reasons why we did not use any of the ready solutions for English. The percentage of 1-grams with the unrecognised symbols is from 6.5% in the literature corpus to 0.4% in the transcript corpus. All punctuations were removed using stream editor SED. STL library was replaced by our own function to manage strings in aim of improving speed of basic string operations.

The algorithm was created in a way that lengths of words are calculated once. Every symbol is checked once if it is a recognised ASCII letter or a special UTF-8 symbol.

The resources of Cyfronet were also used for developing another part of our system. The word hypotheses suggested by the aforementioned part are combined to form sentences hypotheses. Consequently, these sentences are valued by the similarity to the sequences gathered in another corpus. This level of the system is targeted to extract and find the semantic information in the texts. In Polish language, which is non-positional and highly inflective, the information is carried by the words of the whole sentences, which can be freely mixed with no change in meaning. As the N-gram model is inadequate to this task, we use an utility called the bag-of-words model.

The bag-of-words model is a type of a vector linguistic space, in which the vectors correspond to documents of a corpus and the vector coefficients are the word frequencies of subsequent words. In our work the text coprus of the bag-of-words model is Polish Wikipedia, due to its rich vocabulary and semantic contents. Four versions of the matrix are created, each for different level of document span: article, section, paragraph and sentence. The sentence hypothesis is then matched to the corpus using vector product.

We use the computers of Cyfronet to extract the documents into a hierarchical sqlite3 database. This is achieved by the means of third-party wikimedia markup parser. Also some resource-intensive post-processing calculations were conducted on the computers, especially due to memory constraints.

Table 4. The most popular 1-grams in the analysed Polish language corpora (r.p. - reflexive pronoun) presented with the number of times they occured in the analysed corpora and a percentage regarding to the whole text.

Polish	English	occureances	%
.	.	17 245 057	6.066
w	in	8 937 331	3.143
i	and	5 117 887	1.800
na	on, at	4 261 092	1.499
z	with	3 951 484	1.390
się	r.p.	3 904 232	1.373
do	to, till	2 873 022	1.010
nie	no, not	2 863 107	1.007
to	it, this	1 768 183	0.622
że	that	1 656 240	0.583
jest	is	1 489 305	0.524
o	about, at	1 412 878	0.497
a	and	1 386 398	0.488
1	1	1 076 986	0.379
od	from, since	1 019 478	0.359
po	after	946 097	0.333
przez	through	883 156	0.311
2	2	882 907	0.310
0	0	865 825	0.304

Table 5. The most popular 2-grams in the analysed Polish language corpora (r.p. - reflexive pronoun)

Polish	English	occur.	%
się w	r.p. in	330 439	0.1237
się na	r.p. {on, at}	260 365	0.0975
w tym	in this	224 702	0.0842
się z	r.p. with	191 529	0.0717
się do	r.p. to	183 245	0.0699
linki zewnętrzne	external links	174 269	0.0653
w latach	in years	169 156	0.0633
w polsce	in Poland	165 278	0.0619
nie ma	does not have	129 060	0.0483
zobacz też	look also	119 127	0.0446
nie jest	is not	114 487	0.0429
się że	r.p. that	111 635	0.0418
roku .	year.	108 944	0.0408
0 0	0 0	102 788	0.0385
jest to	is this	102 244	0.0383
że nie	that no	92 798	0.0347
na przykład	in example	90 209	0.0338
i w	and in	88 003	0.0329
w gminie	in municipality	83 990	0.0314
w stanie	in state	82 088	0.0307
pod względem	regarding	81 790	0.0306
między innymi	including	79 331	0.0297
o tym	about this	76 952	0.0288