A Web-based Text Corpora Development System

Dan Bohuş, Marian Boldea

"Politehnica" University of Timişoara
Vasile Pârvan 2, 1900 Timişoara, Romania
{bd1206, boldea}@cs.utt.ro

Abstract

One of the most important starting points for any NLP endeavor is the construction of text corpora of appropriate size and quality. This paper presents a web-based text corpora development system which focuses both on the size and the quality of these corpora. The quantitative problem is solved by using the Internet as a practically limitless source of texts. To ensure a certain quality, we enrich the text with relevant information, to be fit for further use, by treating in an integrated manner the problems of morpho-syntactic annotation, lexical ambiguity resolution, and diacritic characters restoration. Although at this moment it is targeted at texts in Romanian, the system can be adapted to other languages, provided that some appropriate auxiliary resources are available.

1. System Overview

This paper presents a web-based text corpora development system which focuses on both their size and quality, built by adapting several existing tools to the necessities of the intended task and the peculiarities of the Romanian language, and by creating a couple of new ones.

The structure of the system is illustrated in Figure 1. It is composed of four main modules, each of them acting sequentially and performing some basic operation on the input text: acquisition, segmentation, dictionary lookup, and part-of-speech disambiguation. For an easier control, the modules are encapsulated by a graphical user interface built as a Tcl/Tk wrapper.

Since we use the Internet as a source of raw texts, the acquisition module performs text acquisition from both HTML and plain text files. A dedicated scripting language was developed to solve the problem of extracting the useful text from HTML files.

Next, the segmentation module translates the text into a stream of tokens, thus forming the Annotated Text Corpus. This module was built using the flex lexical analyzers generator (FSF-GNU, 2000a) , and support for dynamically extending the set of segmentation rules is included via the wrapper interface.

After segmentation, the dictionary lookup module identifies the dictionary words from which every word token could have been obtained, each of them with its corresponding lemma and morpho-syntactic description. A finite state automata software package (Daciuk, 1998) was used to compile the large scale morpho-syntactic dictionary (MSD) as a transducer, reducing the lookup time.

Furthermore, we use a probabilistic part-of-speech tagger to disambiguate between the different variants obtained for some word tokens from the dictionary lookup. For this purpose we adapted the ISSCO TATOO tagger (Robert, 1998) to the Romanian language.

The files in both the Plain Text Corpus and the Annotated Text Corpus are saved in SGML format, with appropriate annotations for each identified token. Attributes regarding each file (e.g. collection date, source, manual intervention flags) are also saved for corpus management purposes.

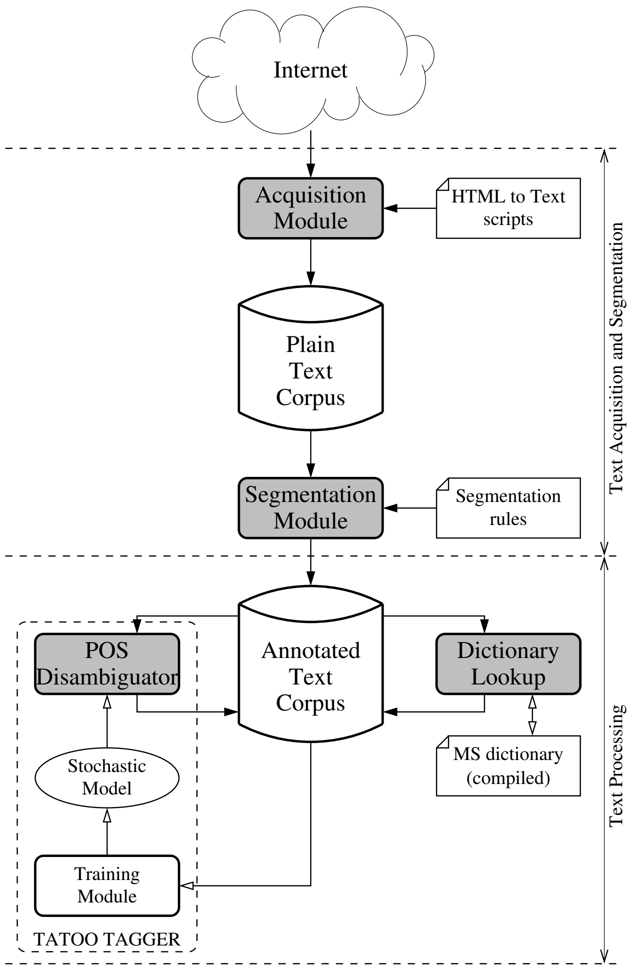

Figure 1: System overview

To make the system accessible even to non-specialists, and to facilitate manual interventions and corrections in the corpora, a graphical user interface was built using Tcl/Tk. The interface provides flexible access to all modules, encompassing them into a Corpora Development Workbench.

In the following sections of this paper we will present in detail each of these modules, then we will describe some experiments and results using this system.

2. Text Acquisition and Segmentation

The first issue addressed is plain text acquisition: lately the number of WWW sites that can be used as sources of texts (e.g. newspapers and radio-TV broadcast stations sites) has grown tremendously, so we took into account the Internet as a source of raw texts.

We have identified over 20 such (Romanian) newspaper-sites, a couple of magazine-sites, and a couple of news-sites. Most of them maintain archives with their past issues, thus providing a large quantity of text in HTML format. Another advantage is the fact that these texts are mostly free of grammatical errors. However, as nothing is perfect, there is also a downside: most of them lack the diacritic characters, and this is one of the main problems addressed by our system.

2.1. Text Acquisition

The actual task of downloading information from WWW sites was automated by means of standard UNIX tools such as wget (FSF-GNU, 2000b) and cron.

The acquisition module acts upon the downloaded files, gathering data from both plain text and HTML files. While acquisition from text files is straightforward, extracting the useful text from HTML files raises some problems. Due to the complex structure of the respective pages, a simple HTML-to-text conversion is not sufficient in most cases, as it would introduce a lot of undesirable text (links, advertisements, email addresses, etc.).

To solve this problem of text collection from HTML files, a dedicated scripting language and a simple interpreter were developed. As the example in Figure 2 shows, such an HTML-to-text (H2T in the sequel) conversion script is composed of 3 sections: ON EXTRACT, SEQUENCE and LAYOUT.

The mechanism is rather simple: the acquisition module advances a pointer through the HTML file according to instructions in the SEQUENCE section. Two types of instructions are allowed here: instructions for advancing the pointer until a certain string or HTML tag is encountered (SKIP TO <TAG|string>), and instructions for extracting text to a variable until a certain string or HTML tag (EXTRACT TO <variable> UNTIL <TAG|string>).

The ON EXTRACT section controls the behavior of the acquisition module during the extraction phase. Instructions for ignoring strings or HTML tags (IGNORE <TAG|string>), and instructions for replacing them with other strings (REPLACE <TAG|string> WITH <string>) were designed. As the example shows, this allows ignoring irrelevant text-formating tags and replacing certain characters in the HTML file with their standard SGML encoding.

Finally, the LAYOUT section instructs the acquisition module how to layout the extracted text in the output file. The example illustrated in Figure 2 uses a single variable (text) for text extraction, but the language allows for any number of variables, and the text collected to these variables is output in the order specified by the TYPE instructions.

```
ON EXTRACT
  IGNORE TAG /center
  IGNORE TAG center
  IGNORE TAG font
  ........
  REPLACE TAG br WITH '\n'
  REPLACE TAG p WITH '\n'
  ........
  REPLACE '\xAA' WITH '&Scedil;'
  REPLACE '\xBA' WITH '&scedil;'
  ........
END ON EXTRACT

SEQUENCE
  SKIP TO TAG /table
  SKIP TO TAG /center
  SKIP TO TAG /font
  SKIP TO TAG /div
  SKIP TO TAG /div
  SKIP TO TAG br
  SKIP TO TAG br
  EXTRACT TO text UNTIL TAG /td
END SEQUENCE

LAYOUT
  TYPE text
END LAYOUT
```

Figure 2: HTML-to-text script

Although this approach is very simple, we found it to be also very effective. The HTML files posted within the same archive generally share the same page structure, and the extraction language is expressive enough to cover the small differences that might exist (for more details see the Experiments section). Therefore, in many cases, writing an extraction script per archive should be sufficient, and support for easy development and testing of these scripts is included in the graphical interface.

As output, the acquisition module generates Plain Text Files (PTF in the sequel), thus forming a Plain Text Corpus. Each file is in SGML format, and contains the extracted text, preceded by a header with various information about the file itself: source, H2T script, acquisition date, a manual intervention flag, link to its correspondent file in the Annotated Text Corpus.

The correctness of the text extraction process can be automatically verified simply by checking for empty PTF files. As a secondary automatic check method, we perform a heuristic search for HTML tags that might have slipped in the extracted text. Finally, manual intervention can easily be performed from the embedding Corpora Development Workbench.

2.2. Text Segmentation

The next operation performed is text segmentation. The segmentation module, built upon the flex lexical analyzers generator (FSF-GNU, 2000a) translates the text into a stream of lexical units (tokens).

A basic set of regular expressions were written to identify 8 types of segmentation units: `WORD`, `PUNCTUATION`, `ABBREV`, `ACRONYM`, `SPECIAL`, `TIME`, `NUMBER` and `EOS` (end-of-sentence marker). The graphical interface provides support for refining these predefined rules, adding new types of segmentation units, and for performing easy manual interventions and corrections in the segmented files.

The segmentation module generates Annotated Text Files (ATF in the sequel), thus forming the Annotated Text Corpus. These files are also in SGML format and their structure is somewhat similar to the PTF files. The header contains general information about the file, and a set of flags indicating whether the file has also been passed through dictionary lookup and part-of-speech tagging. For each identified token, attributes such as token type, value, and a manual intervention flag are saved.

3. Text Processing

While using the Internet as a source of raw text helps to solve the quantitative problem, it also introduces a new one: the largest part of these electronic texts exhibits a troublesome lack of diacritics.

Several solutions can be used for dealing with this problem (Tufiş and Chiţu, 1999; Scheytt et al., 1998). We combined a couple of technologies to solve it: a compressed representation of a large-scale morpho-syntactic dictionary by means of finite state automata, and probabilistic part-of-speech tagging.

3.1. Dictionary Lookup

The first step in diacritics restoration is looking up each identified `WORD` token in a large scale morpho-syntactic dictionary. This operation is performed by the dictionary lookup module and results in a list of words from which the current token could have been obtained (by stripping off the diacritics), each word with its corresponding lemma and morpho-syntactic description.

The dictionary used for Romanian was developed in the MULTEXT-East project, together with similar resources for other five Central and Eastern European languages (Tufiş et al., 1998). It covers the full inflectional paradigms of the words appearing in the Romanian parts of the MULTEXT-East corpus, plus other common Romanian words. Each entry consists of a word, its corresponding lemma, and morpho-syntactic description (MSD), with a total of about 420000 entries, 33000 lemmas, and 611 MSDs.

There has been shown (Mohri, 1996) that very good compression ratios and lookup times can be obtained by representing the dictionary using finite state automata. To this end, we adapted the UTR package, written and made freely available for research purposes by Jan Daciuk (1998) at `www.pg.gda.pl/~jandac/fsa.html`.

The dictionary was brought to an appropriate form and compiled as a transducer using the above-mentioned software, thus reducing its size from 15 Mbytes to 849 Kbytes. This representation of the lexicon allows a fast search of all words that could have generated a `WORD` token, and also provides the corresponding lemma and MSD of each word, as outputs of the transducer.

The dictionary lookup module was written using the functions in the UTR library. It takes as input an ATF file, performs the lookup for each `WORD` token, and annotates it with the matching dictionary word(s), if any. The results classify the `WORD` tokens in 3 categories: (a) not found in the dictionary, (b) matching a single dictionary word, and (c) matching multiple dictionary words (for quantitative results see the Experiments section).

The unknown `WORD` tokens represent a problem, as diacritics can be restored only manually in their case. For the second category, the diacritics restoration problem is considered to be solved – each token is replaced with the corresponding dictionary word, as this is probably the only one that could have generated it. For tokens where multiple alternatives exist, a decision as to which might be the correct word is made using a probabilistic part-of-speech tagger.

3.2. Part-Of-Speech Tagging

The core task in part-of-speech tagging (or disambiguation) is choosing the most likely tag for each word in a context, given a set of possible tags (Armstrong et al., 1996). In our system, part-of-speech tagging is used not only for lexical ambiguity resolution, but also for diacritics restoration, by identifying the dictionary word that could have generated a certain `WORD` token. This was realized by adapting the ISSCO TATOO HMM-based tagger (Robert, 1998) to the Romanian language.

As Romanian is a highly inflectional language, the MULTEXT-East Romanian dictionary contains 611 different morpho-syntactic categories. Because creating a language model based on this large number of MSDs could have overcome the capabilities of our computing resources, we use a reduced tagset, as proposed by Tufiş in the tiered tagging approach (Tufiş, 1999). By reducing the number of tags to 92 (the reduced CTAG tagset), the size of the pre-tagged corpus necessary for building an accurate language model is also reduced. As the experiments will show, the `WORD` tokens still ambiguous after tagging, partly due to tagset reduction, represent less than 5%, and are more often than not the difficult cases in statistical disambiguation (Tufiş and Mason, 1998), making worth the compromise.

Once the tagset established, the only other language-dependent resource needed for building a model is a pre-tagged training corpus. For this purpose, we used a tagged version of Orwell's famous novel "1984", also developed within the MULTEXT-East project. The text was brought to an appropriate form, and the trainer module of TATOO (based on the Baum-Welch reestimation algorithm) was used to create a bigram language model over the CTAG tagset.

The POS disambiguator module uses the TATOO tagger as a child process to perform disambiguation, and works on ATF files already passed through dictionary lookup. When compiling the dictionary as a transducer, we actually replaced the MSDs with the corresponding CTAGs, since we use this reduced tagset approach in disambiguation. Therefore, in the lookup phase, all `WORD` tokens in an ATF file are annotated with the dictionary words that could have generated them, together with their corresponding lemmas and CTAGs.

Based on this information, the set of possible tags for each `WORD` token is constructed and passed to TATOO for disambiguation: its output is then processed, and the most likely tags string, together with the associated words, are annotated as correct to the corresponding tokens.

In most cases, the disambiguation is complete and uniquely identifies the dictionary word that generated a `WORD` token, thus solving the diacritics restoration problem. For a small number of cases (see the Experiments section), the disambiguation is not complete, partly due to the reduced tagset we use: although the tagger decided upon a CTAG as being the most likely one, there are more dictionary words corresponding to that CTAG. Lexical ambiguity resolution is not complete for these words (they are still morpho-syntactically ambiguous), but the CTAG identification can be in some cases enough to solve the diacritics restoration problem. The remaining unsolved tokens are corrected by manual intervention, which can be easily performed from the Corpora Development Workbench.

4. The Corpora Development Workbench

The described modules are all accessible within a single Corpora Development Workbench, which is a graphical user interface created using Tcl/Tk. The main rationale behind it was to make the system accesible even to non-specialists, and to offer an efficient way of performing manual interventions and corrections in text corpora.

To create a friendly interface we used the Tix 4.1 widget library over the extended Tcl (TclX) scripting language, in conjunction with various shell scripts. The main window provides easy access to all modules and processing steps: text acquisition, segmentation, dictionary lookup, and POS tagging/disambiguation.

The Workbench allows parallel work on multiple corpora. For each corpus, a separate configuration is kept, indicating the various resources used throughout the different processing stages: the set of H2T scripts, the segmentation rules, the dictionary used for lookup, and the stochastic model used for tagging.

The acquisition is launched by specifying a few parameters, such as source, type of text (plain text or HTML-formatted), H2T script to use, and target location. The task of writing HTML-to-text scripts is highly simplified by the H2T scripts editor – a framed window that allows a simultaneous view of an HTML file, an acquisition script, and the result of applying the script on the HTML file.

The Workbench includes support for refining and extending the existent segmentation rules. The manual segmentation window, illustrated in Figure 3, provides an easy way for performing manual corrections. Once the user has selected a location to browse, the tokens in the file are displayed as a list, with their corresponding type and manual intervention flag. By scrolling the contents of the file, the user can easily spot possible mistakes of the segmenter, and options such as inserting, removing, splitting, concatenating, and directly editing a token, are available.

The dictionary lookup process is also controlled from the Workbench, dictionary management can be performed

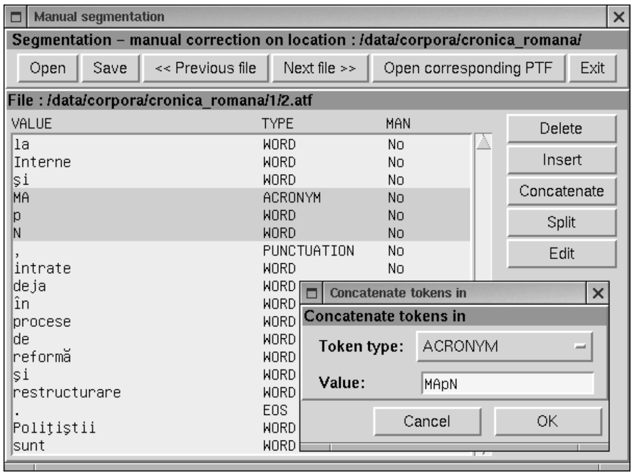

Figure 3: Manual segmentation window

directly from the interface, and commands for automatically creating lists with the unknown words from the ATF files and adding them to the dictionary are included.

One of the primary goals in designing this integrated graphical interface was to provide an efficient way of performing manual interventions and corrections in the text corpora, which otherwise can be a time-consuming, error-prone operation.

Two issues had to be solved here: first, the user must be able to easily spot and manually tag the words that are still ambiguous after the automatic process; and secondly, the user must be able to correct possible mistakes of the tagger. To accommodate both these problems, we used an approach similar to the Penn Treebank standard of ”slash encoding”, together with highlighting certain categories of tokens.

Figure 4 illustrates the manual intervention window. Depending on the task at hand, the user can activate various highlighting filters to spot the intended token categories. When moving the mouse cursor over the text, the token under the cursor is raised, and its attributes are displayed in the Details area. A right-click on a token opens a context-menu, from which the user can change the currently assigned CTAG and dictionary word to another variant found in the look-up process, or to a manually specified word / CTAG.

This way, the words that are still ambiguous after the tagging process can be easily spotted and disambiguated, and the same applies to the words for which diacritics restoration did not succeed: by using different colors for text and tags, even with slash encoding, the text stays readable, and the user can spot not only the words that are still ambiguous, but also the possible mistakes of the automatic diacritics restoration process.

The Corpora Development Workbench includes support for various other corpora management functions. Each operation performed on the data can be logged, and detailed statistics can be performed throughout the different stages of corpus acquisition and processing. All in all, it provides an efficient and user-friendly integrated environment for text corpora development.

	“Cronica Românã”	“Dimineaţa”	“22”	TOTAL
HTML files	3,350	3,234	794	7,378
Total size of HTML files (kB)	40,338	38,843	8,688	87,869
ACQUISITION time (hh:mm:ss)	18:15	21:50	9:27	49:32
Number of H2T scripts used	2	1	1	4
Empty PTF files after first extraction	602	0	3	605
PTF files successfully extracted	3,350	3,234	791	7375
PTF corpus size (kB)	8,580	18,414	8,189	35,185
SEGMENTATION time (hh:mm:ss)	8:13	5:39	1:42	15:34
Total number of tokens	1,262,213	3,179,274	1,492,513	5,934,000
Total number of `WORD` tokens	1,055,540	2,651,233	1,247,196	4,953,969
`WORD` tokens percentage	83.6 %	83.4 %	83.6 %	83.5 %
LOOKUP time (hh:mm:ss)	14:17	16:40	6:30	37:27
Unknown words	61,225	128,837	82,209	272,271
Unknown words percentage	5.8 %	4.9 %	6.6 %	5.5 %
Words with one dictionary form	441,946	1,123,510	510,366	2,075,822
Words with one dictionary form (%)	41.9 %	42.4 %	40.9 %	41.9 %
Words with multiple dictionary forms	552,369	1,398,886	654,621	2,605,876
Words with multiple dictionary forms (%)	52.3 %	52.7 %	52.5 %	52.6 %
TAGGING time (hh:mm:ss)	38:37:19	41:58:37	19:40:25	100:16:21
ATF corpus size (kB)	220,555	555,767	256,523	1,023,846
Remaining ambiguous words	47,357	117,868	53,926	219,151
Remaining ambiguous words (%)	4.5 %	4.4 %	4.3 %	4.4 %

Table 1: Sample corpora development statistics

5. Experiments

In this section we will detail some of our work in collecting and developing text corpora from web-sites. The experiments were performed on a Pentium II 333 MHz computer with 64 MB of RAM, running Red Hat Linux 5.2.

Three web-sites were selected for these experiments (Table 1): the first two, “Cronica Românã” (domino.kappa.ro/e-media/cronica.nsf) and “Dimineaţa” (domino.kappa.ro/e-media/dimineata.nsf), are daily Romanian newspaper-sites, while the other, “22” (www.dntb.ro/22/), is a weekly magazine-site, with slightly longer articles on various subjects, and all post archives with articles from past issues.

The first step was downloading the targeted archives: after quickly studying the structure of each site, shell scripts were written to download HTML files using wget (FSF-GNU, 2000b). As Table 1 shows, this operation resulted in 7,378 files, with a total size of about 87 Mbytes.

The next step was writing H2T acquisition scripts for these archives. For “Dimineaţa” and “22” a single one was enough to extract the useful text from the HTML files: the number of extraction errors for “22” (Table 1) was very small compared to the total number of files, so we merely discarded the resulting empty PTF files. By contrast, for “Cronica Românã” the acquisition resulted in 602 empty PTF files, and a second H2T script was written to cope with structural differences between the corresponding HTML files and the ones succesfully processed with the first. Text acquisition using this second H2T script resulted in no further errors.

Next, text segmentation and dictionary lookup were performed, and the results on the three PTF corpora are presented in the 3rd and 4th sections of Table 1.

The fourth and final operation was part-of-speech tagging. As the statistics illustrate, only about 4.4% of the `WORD` tokens are still ambiguous after the tagging process, and this relatively small number of words are to be disambiguated manually.

The time measurements show that text acquisition, segmentation and dictionary lookup have a tens of minutes order of magnitude, significantly faster than part-of-speech tagging, which took tens of hours for the given size of the processed corpora: leaving aside the manual work still necessary to verify and correct these corpora, in less than a week we obtained a quantity of texts large enough to e.g. start building language models for large vocabulary continuous speech recognition.

An anomaly can be noticed: while the number of `WORD` tokens in the “Dimineaţa” PTF corpus is about twice and a half that in “Cronica Românã”, the tagging times are quite close. This can be explained by the fact that new ambiguity classes were encountered and added to the model while tagging the first corpus, thus increasing the tagging time: the number of ambiguity classes was initially 92 (the ones corresponding to CTAGs after training with Orwell’s “1984”), and rose to 511 after tagging “Cronica Românã”. By contrast, only 13 new ambiguity classes were encountered when tagging “Dimineaţa”.

Manual verification and correction of the corpora is in progress, and evaluations concerning both diacritics restoration and lexical ambiguity resolution performance will be conducted upon its completion.

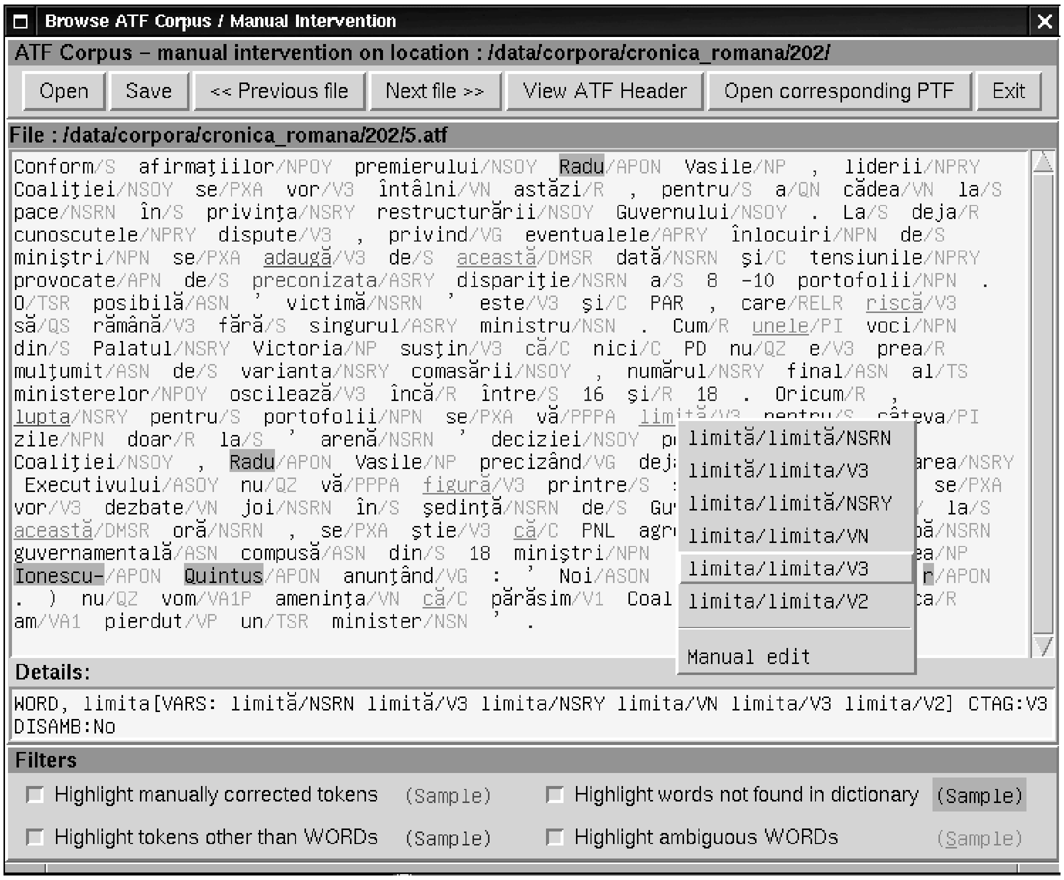

Figure 4: Manual intervention window (tagging/diacritics restoration)

6. Conclusion

In this paper we have presented our work so far on creating a web-based text corpora development system by integrating several existing tools with a couple of new ones, together with the first experiments on using it.

The manual verification and correction of the resulting corpora, necessary for a detailed performance evaluation, is still under way, but a preliminary statistic analysis of system's behavior demonstrates its efficiency as a means to quickly collect and process significantly large text corpora.

Although the system is targeted at Romanian, it can be adapted to other languages, given a couple of appropriate resources: a large-scale morpho-syntactic dictionary, and a pre-tagged text corpus.

Besides improvements of various modules, future work will concentrate on thorough performance evaluations.

7. References

Armstrong, S., G. Robert, and P. Bouillon, 1996. Building a Language Model for POS Tagging (unpublished).

Daciuk, J., 1998. *Incremental Construction of Finite-State Automata and Transducers, and their Use in Natural Language Processing*. Ph.D. Thesis, Technical University of Gdańsk, Poland.

FSF-GNU, 2000a. http://www.gnu.org/software/flex/.

FSF-GNU, 2000b. http://www.gnu.org/software/wget/.

Mohri, M., 1996. On Some Applications of Finite-State Automata Theory to Natural Language Processing. *Natural Language Engineering*, (2):1–20.

Robert, G., 1998. TATOO: ISSCO TAgger TOOl. http://issco-www.unige.ch/staff/robert/tatoo/.

Scheytt, P., P. Geutner, and A. Waibel, 1998. Serbo-Croatian LVCSR on the Dictation and Broadcast News Domain. In *Proceedings ICASSP'98*.

Tufiş, D., 1999. Tiered Tagging and Combined Language Models Classifiers. In F. Jelinek and E. Nöth (eds.), *Text, Speech and Dialogue*, volume 1692 of *Lecture Notes in Artificial Intelligence*. Springer.

Tufiş, D. and A. Chiţu, 1999. Automatic Diacritics Insertion in Romanian Texts. In *Proceedings International Conference on Computational Lexicography - COMPLEX'99*. Pecs, Hungary.

Tufiş, D., N. Ide, and T. Erjavec, 1998. Standardized Specifications, Development and Assessment of Large Morpho-Lexical Resources for Six Central and Eastern European Languages. In *Proceedings First LREC*. Granada, Spain.

Tufiş, D. and O. Mason, 1998. Tagging Romanian Texts: a Case Study for QTAG, a Language Independent Probabilistic Tagger. In *Proceedings First LREC*. Granada, Spain.