TICCLops User and Demonstrator Documentation

version 2.0

Martin Reynaert
Tilburg centre for Cognition and Communication
Tilburg University
Centre for Language and Speech Technology
Radboud University Nijmegen

25 maart 2015

Inhoudsopgave

Hoofdstuk 1

Introduction to version 2.0 – Introducing TICCLops v. 0.2 at INL

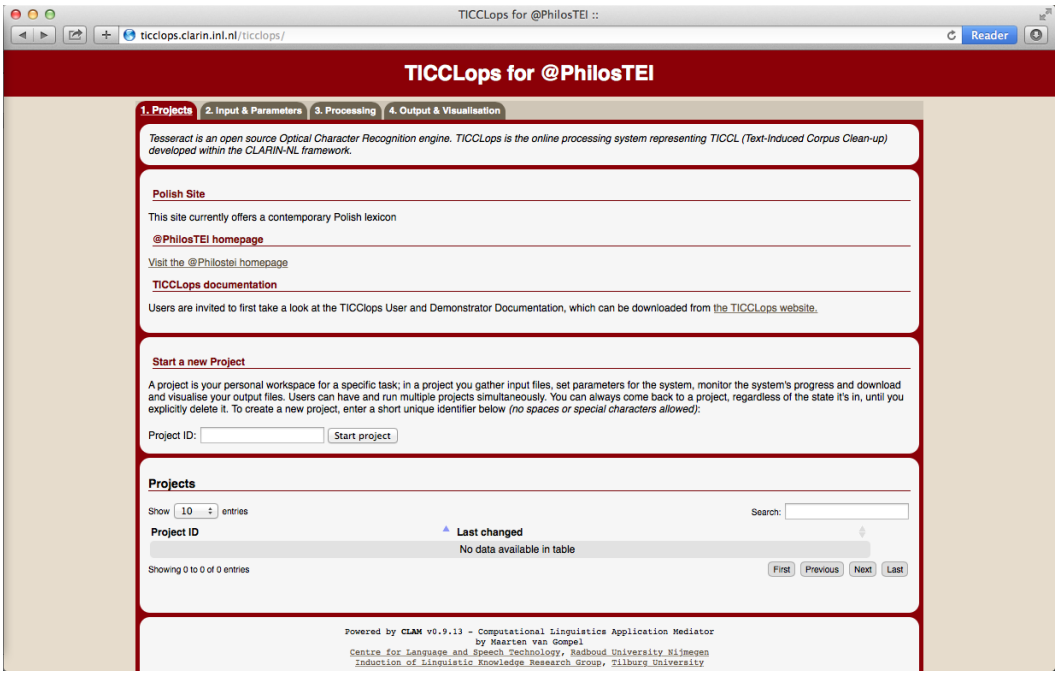
Almost in step with the end of the CLARIN-NL project we are happy to here be able to announce that we have succeeded in fully updating the TICCLops web application and service we have had online at CLARIN Centre INL since after Call 1. This is in large part due to our fruitful involvement in CLARIN-NL Call 4 project @PhilosTEI under the project coordination of Prof. dr. Arianna Betti of the University of Amsterdam (UvA).

- The main idea behind this CLARIN-NL project was to enable philosophers to submit scans of philosophical works they need for their work to the online system and to receive back an electronic version suitable for further processing into a critical edition. The preferred format for this is TEI xml. Due to @PhilosTEI we have gained experience with integrating an OCR-engine, i.e. Tesseract, into our corpus building work flow.

- The philosopher-users study works in a broad range of European languages. The OCR post-correction tool TICCL has therefore been made multilingual within this project. In the @PhilosTEI system, we currently have dedicated TICCL web applications and services for 18 European languages/language varieties.

While the above clearly shows that TICClops is now an integral part of a broader system geared at corpus building starting from digital images of e.g. a book's pages, TICCLops's OCR post-correction functionalities remain

in place and are in fact separately addressable through the CLAM-interface, available at `http://ticclops.clarin.inl.nl`.

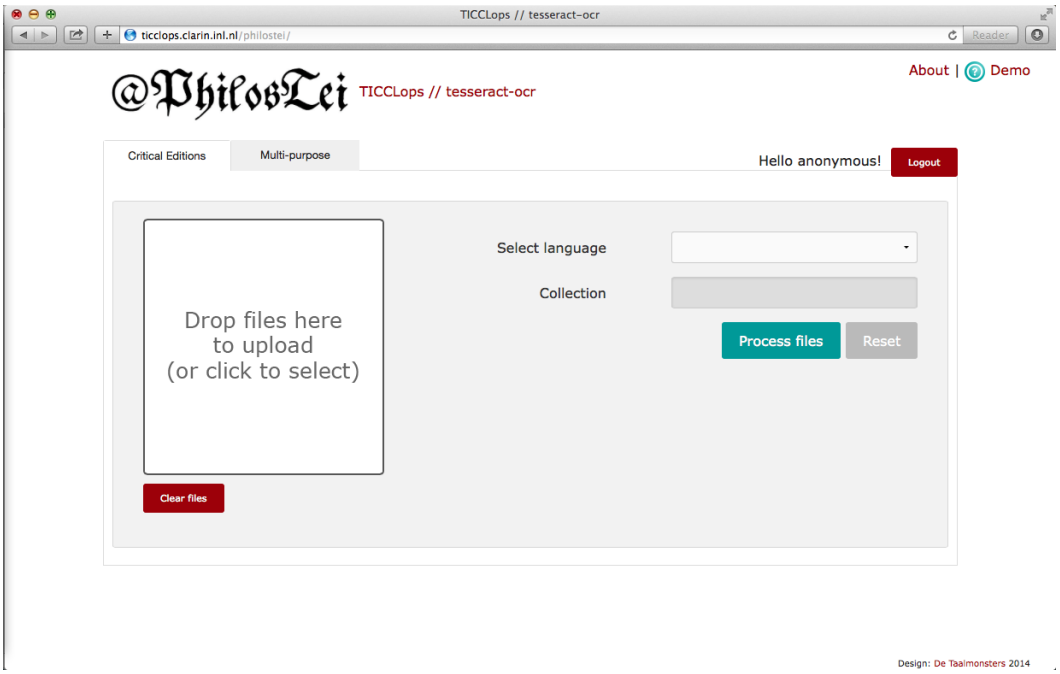


Figuur 1.1: Classic CLAM-interface to the Polish version of @PhilosTEI and TICCLops

As of TICCLops v.0.2 we can now also offer a new JavaScript based interface to TICCLops. This is available in the CLARIN infrastructure from: `http://philostei.clarin.inl.nl`. The functionality of TICCLops in the new interface is available through the tab 'Multi-Purpose' selectable in the top left of the interface.

The new interface, in contrast to its classical CLAM counterpart, provides the user a 'guided tour'. The contents of this tour is dynamically adapted to the tab chosen, i.e. different if you chose tab 'Critical Edition' rather than 'Multi-Purpose'. Access to the guided tour is obtained after clicking on the link 'Demo' in the top right of the interface.

The functionality present under the tab 'Critical Edition' offers the functionality desired by the User Group of the CLARIN-NL closed call 4 project @PhilosTEI. This project ran under the direction of project coordinator Prof. dr. Arianna Betti, herself a philosopher and eager to start building a corpus of open critical editions of major philosophical works. It makes the least demands possible on the users, in fact requiring them only to



Figuur 1.2: New interface to the multilingual/multiscript @PhilosTEI OCR and OCR post-correction system developed by De Taalmonsters

- Select from the drop-down menu the language the work to be digitised was printed in
- To enter a distinctive name for the digital collection about to be built (File names may not start with digits)
- To upload their images, which currently may be in either PDF, DjVu or TIFF formats.
- Press the 'Process' button

The guided tour should suffice to make clear to the users what happens after the previous steps were completed. It must be understood that the system requires time in proportion to the size of the job submitted in order to OCR and OCR post-correct and where necessary perform the required format conversions.

Depending on the users' screen, screen settings and browser used, it may not immediately be obvious to them that the system outputs information about the processing and its output (possibly) underneath the visible screen surface.

The functionality present under the tab 'Multi-Purpose' is highly comparable to that offered under the classic CLAM-interface and in fact links to the very same underlying programs.

One way in which the new interface differs is in its visualisation of the correction results. Under the tab 'Multi-Purpose' the user has the possibility of taking a closer look at both input and output files. When users chose the tab 'Output', they are presented with a list overview of all the system's output files.

For viewing TICCL FoLiA XML output files, select one of the *folia.ticcl.xml files. A special FoLiA XML viewer shows the TICCL corrected file. In the text shown, words that were corrected are highlighted in a red background. The word form in the text is in fact the correction effected. When one hovers over a highlighted word, a pop-up appears which shows the original, replaced word form and the correction candidates proposed by TICCL. These are ranked according to the correction confidence scores TICCL has assigned to each. The best-first correction candidate is the one that replaces the original word form in the text.

Hoofdstuk 2

Introduction to version 1.01

The spelling and OCR-error correction system Text-Induced Corpus Clean-up (TICCL) gradually developed by Martin Reynaert in prior projects is now TICCLops (TICCL online processing system). TICCLops is a fully operational web application and RESTful web service. This is in large part thanks to the Computational Linguistics Application Mediator (CLAM), developed by Maarten van Gompel.

CLAM and TICCLops are the result of a project in Call 1 of the CLARIN-NL programme, a Dutch initiative in the wider European CLARIN framework. The project TICCLops (CLARIN-NL/09-011) was to run from February, 1st to July, 31st of 2010. The project was coordinated by Martin Reynaert. Its scientific programmer was Maarten van Gompel. A student-assistant, Martha van den Hoven did part of the work on the documentation and built the OCR post-correction gold standard for the demonstrator book.

TICCL is a later reincarnation of TISC (Text-Induced Spelling Correction) which was developed as part of Martin Reynaert's PhD work from 2001 to 2005. TICCL extends some of the ideas explored in TISC to the sometimes graver and sometimes different types of errors present in digital texts as produced by machines, in contrast to texts produced by humans. Humans' errors are commonly ascribed to mechanical failure as in typing, dubbed "typo's", or to cognitive causes (not knowing, failure to recall, ...) dubbed "cognitive errors". In contrast, machines' text digitization errors may have a plethora of causes. Their main effect is nevertheless defective text, even if the system in the final analysis faithfully managed to reproduce the original human typesetter's lapse.

Figure 2.1 shows an example of a typesetting error in a historical newspaper.

In Figure 2.2 we see the OCR-ed text as available online. As can be seen, the OCR-process faithfully reproduced the original typesetting error. It does

CHARLES A. LINDBERGH

DE VLIEGENDE DWAAS, DIE EEN LUCHTHELD WERD

Lindbergh, een blonde jonge man van 25 jaar, wordt door zijn medevliegers de „bolfer" Lindbergh genoemd. Hij is zijn vliegersloopbaan begonnen bij den Amerikaanschen postdienst, als zoovele jonge vliegers, alleen om een bestaan te vinden. Twee welgeslaagde paracnutesprongen uit een onklaar vliegtuig, kort na elkaar, gaven hem zijn bijnaam. Hij liet zich kennen als een uitstekend vlieger, doch zijn naam was buiten de vliegvelden veertien dagen geleden nog zoo goed als onbekend. Toen hij echter den 11en Mei plotseling neerdaalde op het vliegveld van St. Louis, na een vlucht van 14 uur van Sint Diego en den volgenden dag in 10 uur tijds doorvloog naar Nieuw York om op zijn eentje aan den wedvlucht over den Oceaan deel te nemen, was

COBHAM HEEFT LINDBEGH GEINTERVIEWD

WAT HIJ TE WETEN KWAM

Uit Londen wordt gemeld:

De geestdrift over Lindbergh is in Londen even groot als elders en wordt door het wansucces van Carr en Gillman in de Perzische Golf niet getemperd. De Express heeft een omstandig verhaal van Lindbergh zelf, maar Cobham, die dadelijk naar Parijs overgevlogen is, doet ook in de Mail een verhaal, waarin veel van Lindberghs eigen vertelling opgenomen is. Lindbergh noemt het onjuist, dat sommigen den indruk voor al op zijn overigens onbetwistbaar geluk leggen. De tocht was zorgvuldig voorbereid

Figuur 2.1: Newspaper article captions from 'Het Vaderland : staat- en letterkundig nieuwsblad, 23-05-1927, front page' showing a typesetting error (Source: http://kranten.kb.nl/. Query for 'Lindbegh'. The query's result is shown highlighted on the page's image.)

also show a fair number of OCR misrecognition errors, however.

The ultimate goal for TICCLops is to eventually be able to correct this text to:

COBHAM HEEFT <sc variants="LINDBEGH">Lindbergh</sc> GEINTERVIEWD

WAT HIJ TE WETEN KWAM

<sc variants="I.'il">Uit</sc> Londen wordt gemeld:

De geestdrift over Lindbergh is in Londen even groot <sc variants="al-,">als</sc> elders en wordt door het wansucces van Carr en Gillman in de <sc variants="Perz'S'-l.e">Perzische</sc> Golf niet getemperd. De Express <sc variants="heelt">heeft</sc> een

```
omstandig verhaal <sc variants="van.">van</sc>
Lindbergh zelf, maar <sc variants="Cobharii,">Cobham
</sc> die dadelijk naar Parijs overgevlogen is, <sc
variants="doel">doet</sc> ook in de Mail een verhaal
, waarin <sc variants="vee!">veel</sc> van <sc
variants="Linclberglis">Lindberghs</sc> eigen
vertelling opgenomen is. Lindbergh noemt het onjuist
, <sc variants="clat">dat</sc> sommigen den indruk
vooral op zijn overigens onbetwistbaar geluk leggen.
 De tocht was zorgvuldig voorbereid en al
```

As can be seen, we do not aim to overwrite what the system has determined to be erroneous, but rather we (aim to) add the correct word form in an xml-format to the original OCR-ed text. At this point in time, we too need to add our own disclaimer: it will be some time yet before the above level of perfection is achieved by any automatic spelling correction system. The reader can discern the real-word errors 'heelt' (E. 'heals') for 'heeft' (E. 'has') and 'doel' (E. 'goal') for 'doet' (E. 'does'). Correction of real-word errors is still an active research topic. Also 'Linclberglis' for 'Lindberghs' represents a challenge which we cannot satisfyingly meet yet.

Figuur 2.2: OCR-ed text of the newspaper article to the right in the previous image from 'Het Vaderland : staat- en letterkundig nieuwsblad, 23-05-1927, front page' showing a disclaimer about the OCR quality (Source: http://kranten.kb.nl/. Query for 'Lindbegh'.)

Some of the original ideas partially developed in TISC have to date not been expanded and or reinvestigated in the TICCL(ops) framework presented here. Noticeably, the valuable idea of employing local lexical context to properly determine a particular word string's intended 'identity' has not to date been reimplemented in TICCL and therefore TICCLops. We maintain

that local context will be the major factor enabling proper resolution of short word OCR-errors.

The original main idea in the TICCLops project proposal was to build a shell around TICCL to turn it into TICCLops, the TICCL online processing system, i.e. a fully functional web application and web service. To Maarten van Gompel's credit he extended this idea to build a generic shell that can almost effortlessly be wrapped around any existing linguistic application to turn it into a web application and service. The shell soon got a generic shells' name: clam.

CLAM comes with its own extensive documentation. This is to be found at: http:/ilk.uvt.nl/clam.

CLAM is maintained at github. To install the software, follow either of these routes:

Do a git clone: git clone git://github.com/proycon/clam.git

Or download the tarball from https://github.com/proycon/clam/tarball/master

Please consult the CLAM Manual for further instructions. After reading the manual, it is recommended to watch the CLAM tutorial videos at the bottom of the web page.

For additional support and announcements please join the CLAM Mailing List at https://lists.uvt.nl/mailman/listinfo/clam.

Development Notes

TICCLops was designated the first step in the language technology tools workflow being developed within the CLARIN-NL project TTNWW. As a consequence, both TICCLops and to a larger extent CLAM are still under development. As such, this documentation is not definitive, but a work in progress.

In TTNWW we are to add support for Alto xml and FoLiA xml to TICCL. Alto xml is the intermediate xml format used by the KB to provide the necessary link between a digitized page's image and the text derived by OCR from it. FoLiA xml has been developed in the wake of the CLARIN-NL project TICCLops and the STEVIN project SoNaR to provide a better format for linguistically richly enriched texts.

Hoofdstuk 3

Functional description of TICCL

3.1 Global description of TICCL

TICCL (Text Induced Corpus Clean-up) is a system that is designed to search a corpus for all existing variants of (potentially) all words occurring in the corpus. This corpus can be one text, or several, in one or more directories, located on one or more machines. TICCL creates word frequency lists, listing for each word type how often the word occurs in the corpus. These frequencies of the normalized word forms are the sum of the frequencies of the actual word forms found in the corpus. TICCL is a system that is intended to detect and correct typographical errors (misprints) and OCR errors (optical character recognition) in texts. When books or other texts are scanned from paper by a machine, that then turns these scans, i.e. images, into digital text-files, errors occur. For instance, the letter combination 'in' can be read as 'm', and so the word 'regeering' is incorrectly reproduced as 'regeermg'. TICCL can be used to detect these errors and to suggest a correct form.

One of TICCL's main goals is to output a list of focus words together with their variants, or correction candidates. A focus word is the word form that is found in the input corpus. What are considered to be variants depends on the maximum divergence allowed in terms of difference in characters, which is an input parameter of TICCL. This maximum edit distance is measured in terms of Levenshtein distance. If there is a one-letter difference in a word pair, this counts as 1 edit cost. It does not matter if the one letter is an extra letter in the variant (insertion), a missing letter in the variant (deletion), or a different letter (substitution). If two letters should change places, this counts as an edit cost of 2 (transposition).

Tabel 3.1: Edits

Correct word	Variant	Name	Cost
regering	regeering	insertion	1
regering	regerng	deletion	1
regering	regeriug	substitution	1
regering	regernig	transposition	2
regering	regermg	substitution + deletion	1 + 1

We give some examples in Table 3.1.

If TICCL is run with a maximum edit distance of 1, the first three variants given in the example will be paired up with the correction candidate 'regering', the others will not. The maximum possible edit distance that TICCL works with is 3. The bigger the edit distance used, the higher the number of false positives, meaning supposed variants of a focus word are retrieved, that are in reality not variants of the focus word, but different words. If the maximum edit distance is chosen too small, not all variants of a focus word are retrieved. A maximum edit distance of 2 usually gives the best results.

TICCL returns the number of types (distinct word forms) and tokens (total number of words in the corpus). When we count how often words are used in a corpus, we in essence compile a frequency list. In relation to word frequency lists we talk of **word types** as contrasted to **word tokens**. When one derives a frequency list from a text, the tokens are counted and each distinct type with its frequency count is added to the list. The word types that make up the list define the vocabulary as observed in the corpus. The list then contains all the **word forms** observed. The word forms retain their inflections in a frequency list, they are not typically reduced to their **lemma**, the word form under which the various inflected forms would fall in a dictionary. The vocabulary contained in a corpus should not be taken to constitute the language's vocabulary: there may actually be many words in the language that happen not to be in the corpus at all, however large this corpus is. The same goes for the vocabulary provided to a spelling error detection and correction system. Even though such a system's **dictionary** may be very large, it cannot be complete.

Also the type/token ratio is returned. In a normal corpus, this ratio will be about 50%. However, if there are many OCR-errors in the texts, the ratio can be much higher.

TICCL can run very well without a supporting dictionary or lexicon because the vocabulary present in the corpus plays an integral role in the variant retrieval process. This makes TICCL far less domain sensitive than most

other approaches to spelling correction. In fact, experiments have shown that the larger the corpus TICCL is asked to process, the more reliable the observed word frequency statistics are and the better the results.

It is also possible to evaluate how well TICCL is doing. In order to do that, a so-called gold standard evaluation file is needed, which contains the text of the corpus. In this file are all misrecognised words in the OCR-ed version of the book, lined up with their correct or canonical form. Any variant of a focus word returned by TICCL, which is also in the gold standard, is counted as a true positive. Any variant returned by TICCL that is not in the gold standard, is a false positive. Any variant not returned by TICCL, but that exists in the gold standard, is counted as as false negative. In this way of measuring how well TICCL performs no account is taken of the correct words in the OCR, the true negatives, for which TICCL does not report anything. For more in depth information about spelling correction evaluation, please see Chapter 6 or refer to [**?**].

The user can influence the amount and type of words that TICCL tries to find variants for. It is possible to set criteria for word frequency and word length, so TICCL works only on the words that match those criteria from the word frequency list it builds from the corpus.

TICCL assumes the corpus it is used on is not tokenized. TICCL relies on word form normalization to be able to deal with common OCR-errors that mistakenly put punctuation inside words. TICCL can without problem be set to work on tokenized text, however.

TICCL is in principle largely language independent. However, the resource files that currently come with TICCL are for Dutch only.

3.2 Definitions

- corpus: the input texts that contain the words for which the variants in a given lexicon or in the corpus are looked up.

- lexicon: the lexicon TICCL uses as a list of validated word forms. TICCL is released with several lexicons, see elsewhere for a description.

- canonical word form: the form of a word as found in the validated word list (lexicon).

- focus word: the word form used to look up variants, taken from the

corpus

- normalized variant: word forms as found in the corpus but normalized using a reduced alphabet.

- surface form: the word form exactly as it is found in the corpus, without normalization.

- character confusion: a possible confusion of one or more characters. In the earlier example 'regeermg' vs. 'regeering' the confusion is 'm' - 'in'.

3.3 TICCLops' address

TICCLops is online at CLARIN Centre INL. TICCLops is available to the parties having been granted access and user rights at the following URL (password required):

http://ticclops.dev.inl.nl

TICClops is also due to be deployed at TiCC, Tilburg University, soon.

Hoofdstuk 4

TICCLops online user environment: Step-by-step description

In this chapter we describe the TICCLops environment as presented to the user.

The following sections describe the simple steps a user should take to be able to upload a corpus for correction by TICCLops and to be able to download the results after processing has been completed.

In this chapter we present screen shots of the various screens presented by TICCLops. As these may be hard to read due to their limited size, we list them all again in larger and more readable size in the appendix.

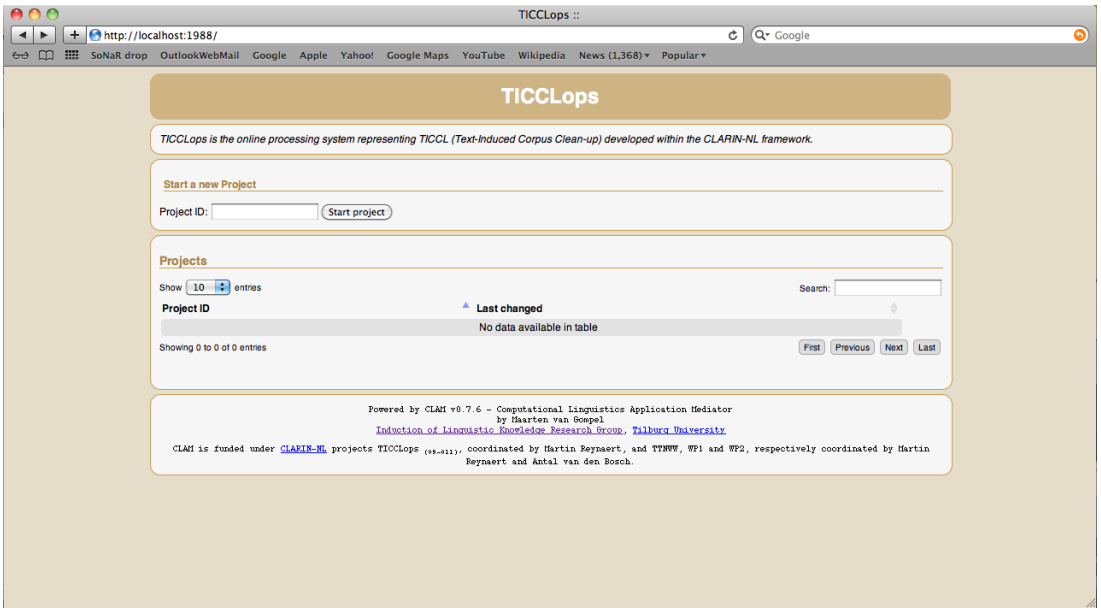
4.1 Step 1 Choose project

Figure 4.1 shows the TICCLops start-up project screen.

The first thing to do when working with TICCLops is to define a new project or to choose an existing one.

- If you want to start a new project, type the name in the field Project ID and click *Start project*.
- In order to choose an existing project, you can click on its name in the list of Project IDs. The project will then be reactivated.

Figure 4.2 shows the TICCLops project screen with two predefined projects.



Figuur 4.1: TICCLops start-up project screen

To run the demonstrator, you should create a new project by choosing and entering a name you consider suitable. We explain in more depth how to explore TICCLops by way of the demonstrator in Chapter 7.

4.2 Step 2: Specify input

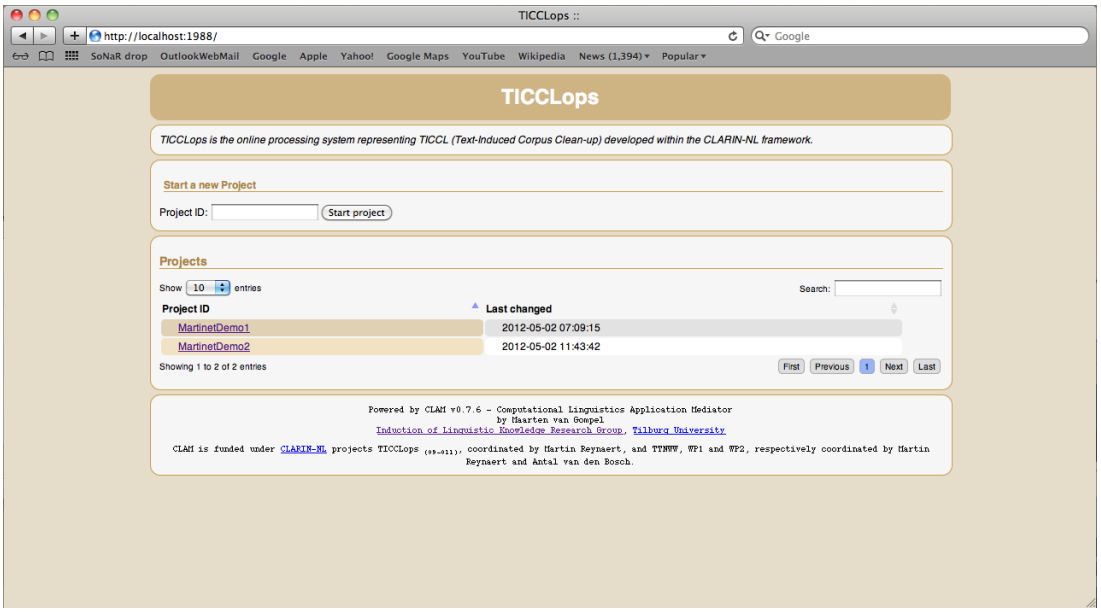
Figure 4.3 shows the TICCLops screen for input selection.

A choice has to be made what input source TICCL should use. The next step is to specify the corpus files you want TICCL to process. There are several options:

A pre-installed corpus is available. The TICCLops demonstrator comes with the Martinet book as such a pre-installed corpus. You may choose to explore the system by way of this.

Figure 4.4 shows the TICCLops screen for input selection after the Demonstrator data have loaded. Note that the interface for selecting more input files has now moved down.

If you now want to select one of the pre-installed lexicons, select one from the drop-down menu under 'Demo data' and click 'Add resource' again. The lexicon chosen will then be loaded and be made available to the system. Figure 4.5 shows the TICCLops screen for input selection after the lexicon



Figuur 4.2: TICCLops project screen with two predefined projects

chosen has loaded and after we clicked on the button 'Last' to the right under the list of loaded resources.

Having now loaded both a corpus for processing and a lexicon we can now proceed down screen for setting TICCLops specific processing parameters.

However, we first briefly discuss the alternative input specification options.

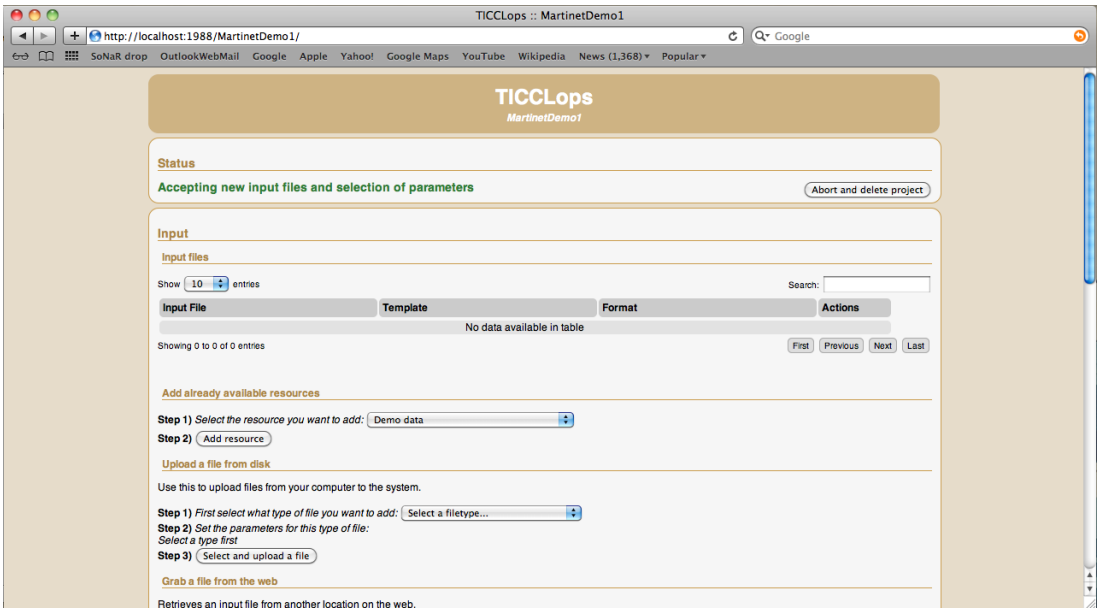
- Upload a file from disk

  Figure 4.6 shows the TICCLops screen for uploading files from your own computer onto the TICCLops server.

  First you need to specify the correct input format of the file that will be uploaded. This is done with the dropdown list. The choices available in this demo are:

  - Koninklijke Bibliotheek XML-formaat [utf-8]
  - Lexicon list
  - Plain Text Format (not tokenized) [utf-8]

  Next, click button *Select and upload a file*, to choose the file from your file system.



Figuur 4.3: TICCLops screen for selecting the input (top half)

As we explain more in depth in Chapter 8 the list of available options is due to expand very soon.

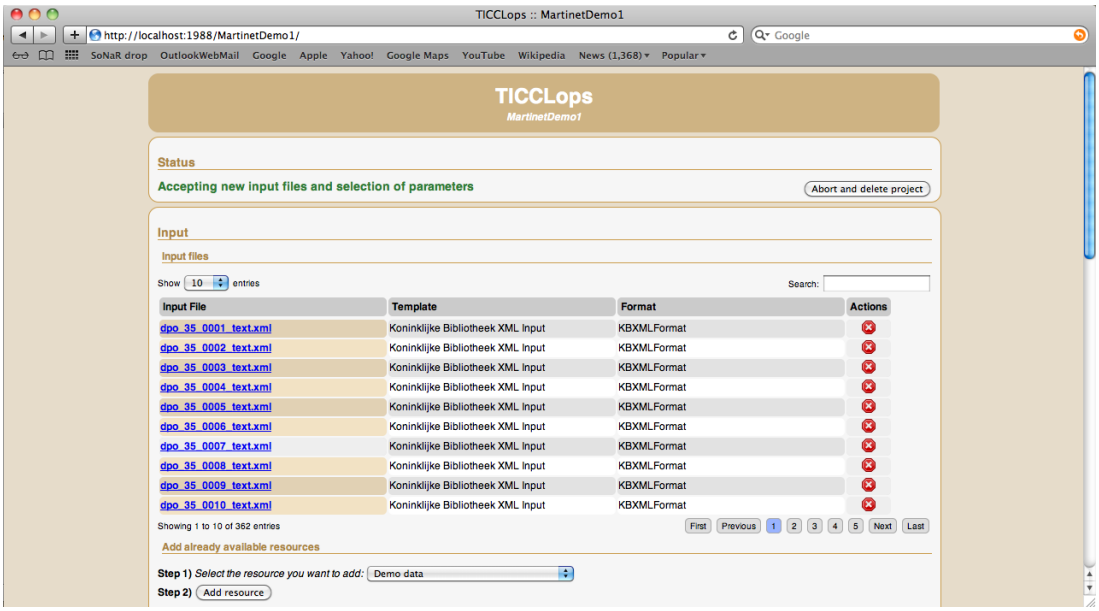
- Grab a file from the web

  When choosing to grab a file that is available on the internet, the correct format must also be specified, using the dropdown list. The available formats are the same as for the previous choice. Next, the URL of the file to be retrieved must be entered, and then click button *Retrieve and add file*. Note that TICCLops in its current version does not know how to deal with HTML.

- Add input from browser

  It is also possible to create an input file using an editor. In order to do this, click button *Open Live Editor* and an editor will be started, as shown in Figure 3. Now text can be typed or copied into the input field, and saved under a chosen name. Also here the format of the text should be specified using the dropdown list.

  Press *Add to input files* to save the document as an input file, or press *Cancel* to leave this screen without saving.



Figuur 4.4: TICCLops screen after DEMO data have loaded

4.3 Step 3: Parameter selection

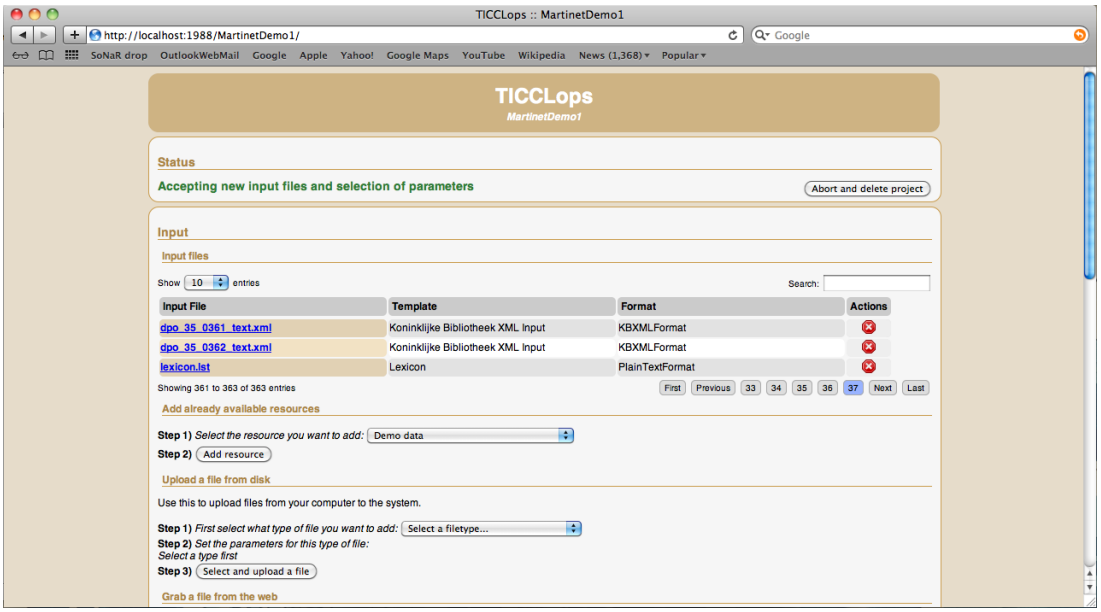
Next, parameters have to be selected, to run TICCLops with the desired options. Figure 4.7 shows the TICCLops screen for parameter selection, i.e. the lower half of the input selection page.

- Parameter: Focus word selection

  The selection criteria for the focus word need to be chosen.

  Specify the limits in frequency and word length. Minimum and maximum frequency can be specified, and minimum and maximum word length in characters.

  - The word frequency minimum and maximum can have a value between 0 and 10 million, and the minimum and maximum length of the focus word can be set anywhere from 0 to 100. Because the maximum frequency is not known before running TICCL, it can be useful to set this at a high value.
  - It is not advisable to run TICCL on very short words, i.e. shorter than 5 characters, on badly OCR-ed corpora. When going for shorter words, the LD-limit must be conservative.



Figuur 4.5: TICCLops screen after the lexicon selected has loaded

  - Example: 1000000-2-5-50 selects all words whose frequency lies between 2 and one million occurrences (including 2), with a word length between 5 and 50 (including 5 and 50).

- Parameter: Edit Distance

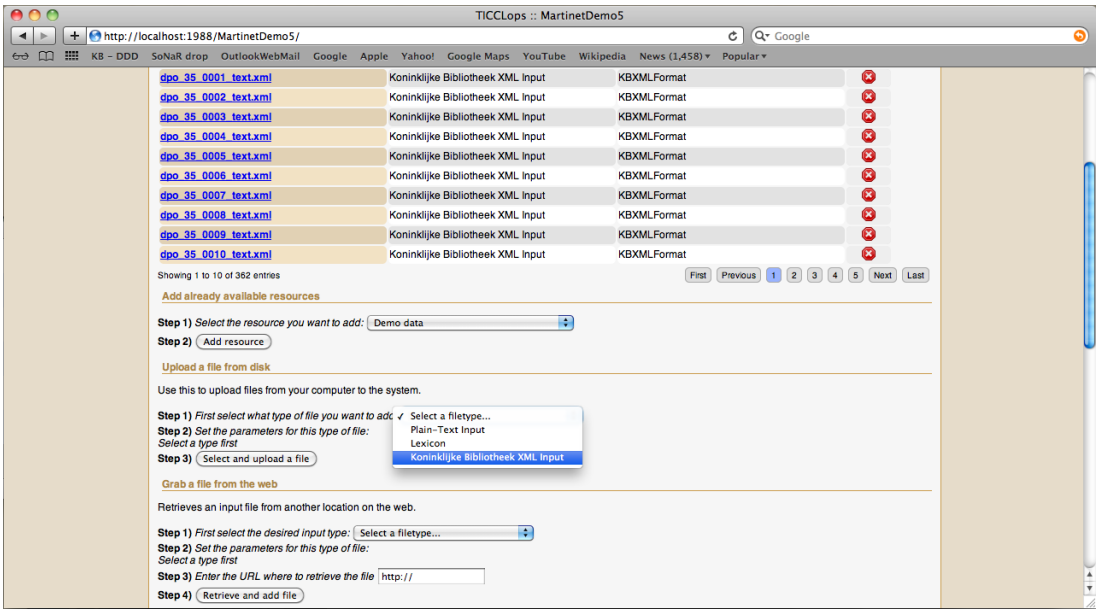
  Another parameter that needs to be set is the maximum edit distance (Levenshtein distance) between variants. This can be 1, 2 or 3. This parameter defines the scope of the search for lexical variants within the corpus.

- Number of output variants

  With N-best ranking the number of output canonical variants of a corpus word is limited. A choice can be made from 1, 2, 3, 5, 10 or 20 N-best ranked. In all cases the variants output are ranked, best candidate first.

Finally there are some options, that can be switched on with the check-boxes. These options are off by default.

- Evaluation: This option only makes sense when indeed a gold standard for the corpus is available, as is the case for the TICCLops demon-



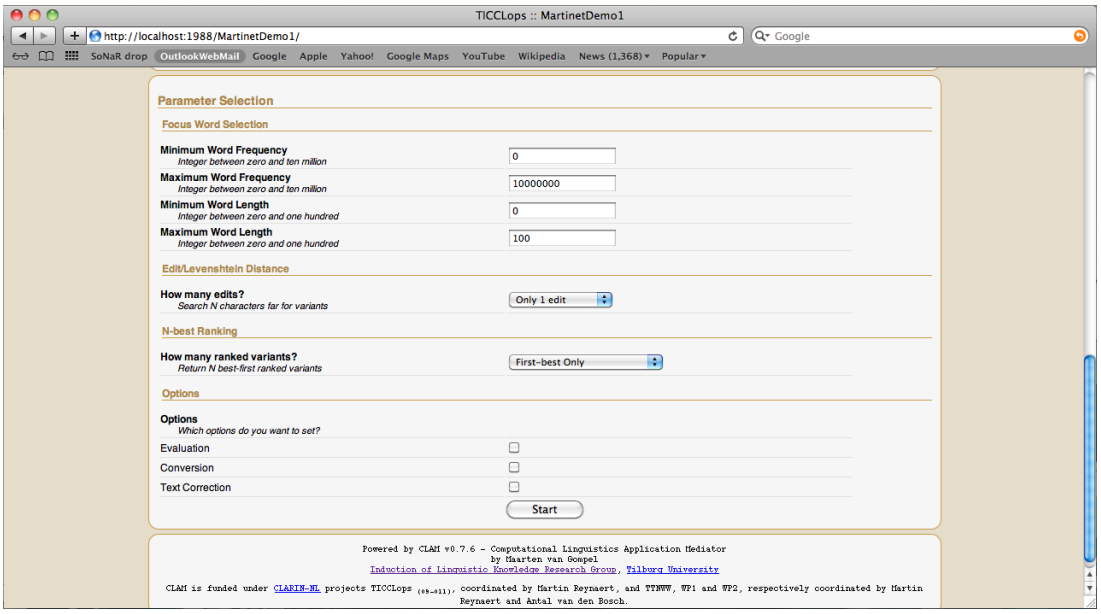
Figuur 4.6: TICCLops screen for file upload: file type selection dropdown list

strator. By checking this box, the output of TICCLops given its input parameter settings is compared with the expected outcome given the gold standard and scored on various levels. A full report of this can be found in the log file after processing has finished.

- Conversion: Tick this box when your input data are in UTF-8 encoding.
- Text Correction: when this box is checked your input files will be rewritten and the canonical forms for variants within your corpus will be added. The original word form is not overwritten, but rather enriched with the number of n-best canonical forms requested earlier (if, indeed, so many were found).

For demonstration purposes you might choose the following:

- Load the pre-installed demonstrator corpus
- Load a lexicon: Historical-contemporary Dutch lexicon
- Set the system parameters
  - word frequency: 1-250



Figuur 4.7: TICCLops parameter selection screen (lower half input selection page)

  - word length: 6-100
  - maximum edit distance: 2
  - N-best ranking: Up to 3 N-best ranked
  - check boxes ‘Evaluation’ and ‘Conversion’ are to be checked
  - check box ‘Text Correction’ if you want the actual input files to be rewritten

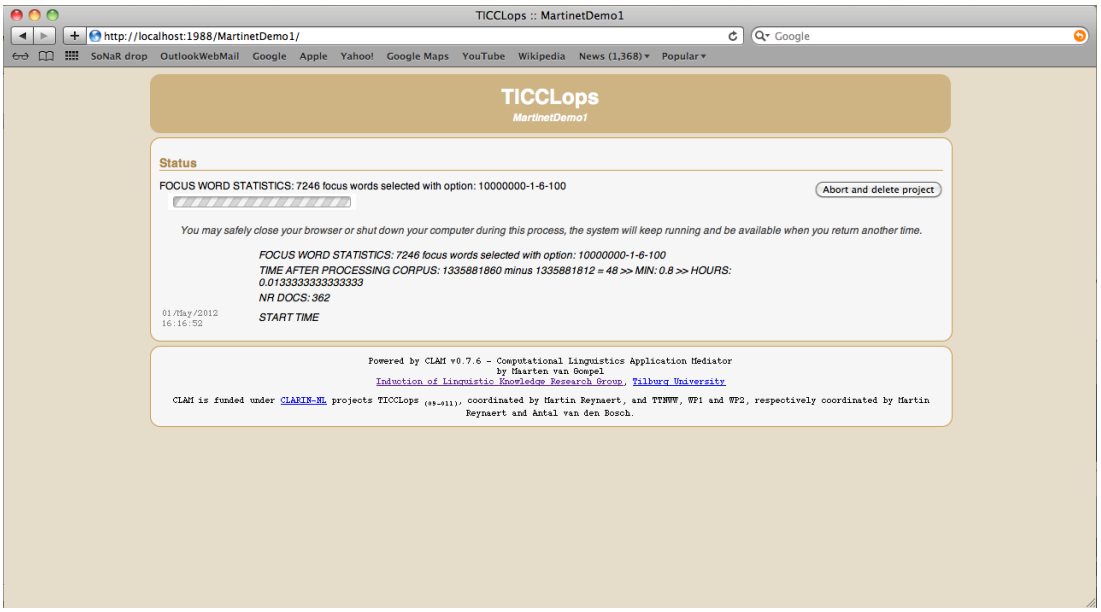
- Next, click *Start*.

4.4 Step 4: Processing

Figure 4.8 shows the TICCLops screen for monitoring processing progress.

While TICCL is running, the status of the process is shown on the screen, and is regularly updated.

It is not necessary to leave the browser open while the process is running. You can close the window and come back later. If you want to reopen the process you were running, start TICCL again and choose the project you started previously from the list.



Figuur 4.8: TICCLops processing progress screen

It is also possible to stop TICCL running, by clicking *Abort and delete project*. This stops the process, and deletes all the uploaded input files and possibly generated output files, and the project.

4.5 Step 5: Output

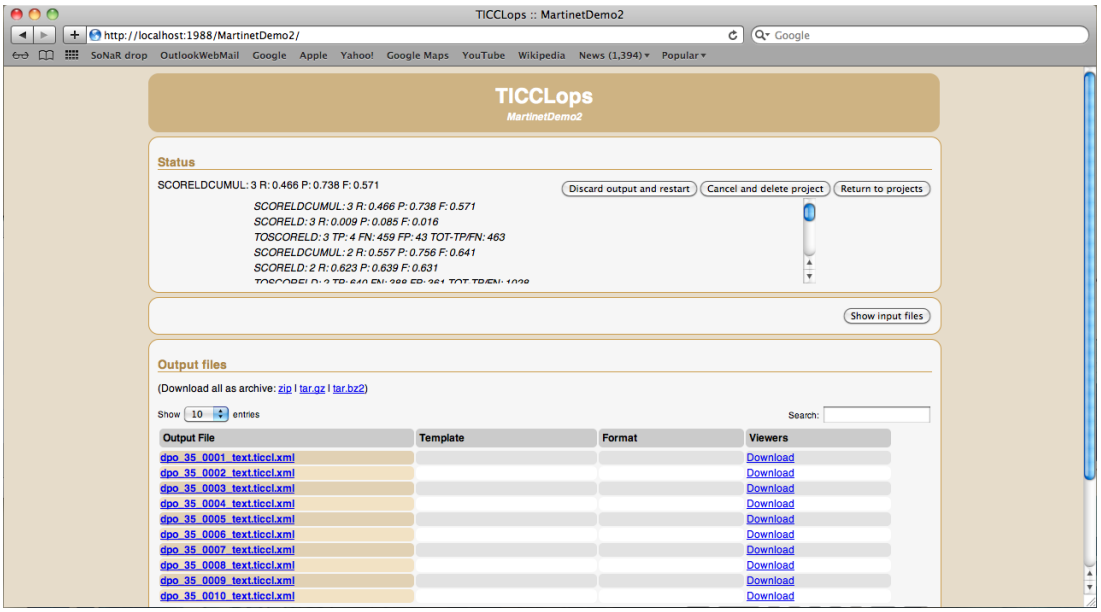
Figure 4.9 shows the TICCLops screen for output viewing or downloading of the corrected files and the variant list produced for the corpus.

After processing has finished and when an evaluation was performed as in this screen shot, the evaluation scores are shown. We will describe these in far more detail in Section 7.2.2.

After TICCL processing is done, the output files are listed. They can be viewed separately by clicking on them, and they can be downloaded bundled together as an archive.

Available archive formats: Several archive formats are possible: zip, tar.gz and tar.bz2. To download the files in compressed format, click the format of choice.

To continue: You now have three different options in order to continue.



Figuur 4.9: TICCL output viewing or downloading

- Clicking *Discard output and restart* deletes all the generated output files, and restarts the project on the basis of the input files uploaded earlier.
- Clicking *Cancel and delete project* deletes all the uploaded input files and possibly generated output files, and the project itself.
- Clicking *Return to projects* takes you back to the initial project screen where you then may define and initiate a new project.

Temporarily absenting yourself while the system is running: If you want to temporarily leave your project, you can close your browser, and open your project again at a later time.

Hoofdstuk 5

TICCLops Output

TICCL gives a list as output, consisting of word pairs where the left hand word is the canonical form, and the right-hand word is the normalized variant of the word, as found in the corpus. This variant can differ from the canonical form because of a typesetting error, spelling or typographical error, OCR error, historical spelling difference, or a combination of these.

Corrected files: The major use a spelling correction system is conventionally put to is to actually correct the input texts it is fed.

TICCLops in fact affords this possibility but only in case it is specifically asked to do so. This is because TICCLops works in an unsupervised fashion: it does not time and again ask the user what she thinks about this or that edit, but goes ahead fully automatically. It is for this very reason that, given the state of the art where there is as yet insufficient certainty about the changes the system may effect, that changes are not made unnoticeably. This means changes may be made, but only in such a fashion that these should be prominently inspectable by the user.

All changes made are effected in an xml-like format.

5.1 Output files

- Variant output file (.varout)

  The system gives as output a list containing the focus words and their variants retrieved form the corpus. Each line of the list consists of the normalized variant, corpus frequency of that variant, focus word, corpus frequency of the focus word, the Levenshtein distance between the two, and the surface forms in pseudo-XML. This XML-structure is necessary, because the surface forms may contain any character, including the delimiter. This delimiter cannot appear in normalized forms or focus

words.

Example:

???

- Log file (.log) contains information about which corpus files were processed, along with the processing time per file, the number of word forms (types) and the total number of words (tokens) found in the corpus, and the type/token ratio. The log file is currently only interpretable by humans. Future versions of TICCLops will also provide a machine-interpretable log file.

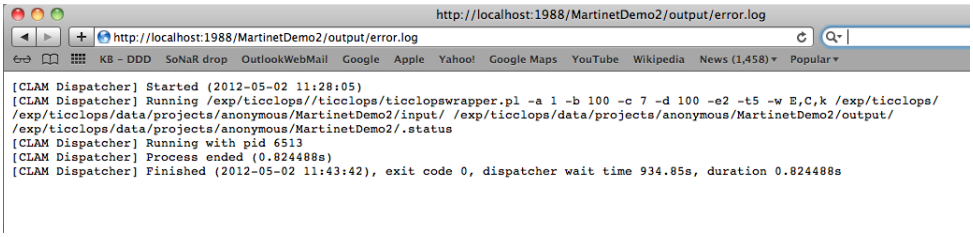
  If TICCL was run with an evaluation file, the results of the evaluation are also in the log file. The log file ends with the total time run.

  See Section 6 for more information about what we understand under evaluating a spelling and OCR-error system.

- Error log file (error.log)

  Each time TICCLops is run an error log file is created. This does not necessarily imply that something went wrong. By default, the error log file contains metadata about the run, e.g. the specific parameter settings that were chosen by the user for the run.

  An example of the error after a faultless run is shown in Figure 5.1:



Figuur 5.1: TICCLops error.log after a faultless run. The parameter settings for the run are logged as shown.

  When something does go wrong during TICCLop's processing the event is logged in this file.

  An example of the error log when something did go wrong:

5.2 An introduction to the effects of lexical variation

On invitation by the National Library of The Netherlands (Koninklijke Bibliotheek - Den Haag) we have worked on contemporary and historical text collections. The contemporary collection comprises the published Acts of Parliament (1989-1995) of The Netherlands, referred to as 'Staten-Generaal Digitaal' (henceforth: SGD)[1]. The historical collection is referred to as 'Database Digital Daily Newspapers' (henceforth: DDD)[2], which comprises a selection of daily newspapers published between 1918 and 1946 in the Netherlands. The historical collection was written in the Dutch spelling 'De Vries-Te Winkel', which in 1954 was replaced by the more contemporary spelling as found in the SGD from 1954 onwards. This collection was digitized in a pilot project for extensive digitization projects underway in which a selection of 8 million pages from the Dutch newspaper collection present in the National Library is being made publicly available online. A nice consequence of the fact that both collections we have worked on here are available online is that any example we may give can be independently verified. If we claim that the English word 'restoring' is in fact an OCR-misrecognition of the word 'regeering' (i.e. De Vries-Te Winkel spelling for the contemporary word 'regering' ('government'), any reader can look it up on the DDD website and see the actual word highlighted in yellow in the digital image from which the OCR-ed text version was created.

Fig. 5.2 shows the Vocabulary Growth Curves (VGCs) [**?**] obtained with the zipfR package due to [**?**] for three text collections, two of which we later deal with in this paper. VGCs show how many new, previously unseen, words are seen the more text is read. The attendant growth lines for hapax legomena are also plotted. TWC02 is a contemporary one-year newspaper corpus, covering the year 2002, composed mainly of 5 national Dutch newspapers. We use the corpus here as a kind of reference for the 'normal' vocabulary growth one would expect to see given 'born-digital', well edited, published text. Its curve at first shows a greater vocabulary growth than the SGD. This is easily explained by the fact that in newpapers far more topics are touched upon than in a typical debate in Parliament, which is likely centered around far fewer topics, calling upon a smaller vocabulary. At around 60M words the vocabulary growth of the TWC02 starts to slow down, and the SGD vo-

[1]URL: http://www.statengeneraaldigitaal.nl/. The full collection which is now available runs from 1814 to 1995

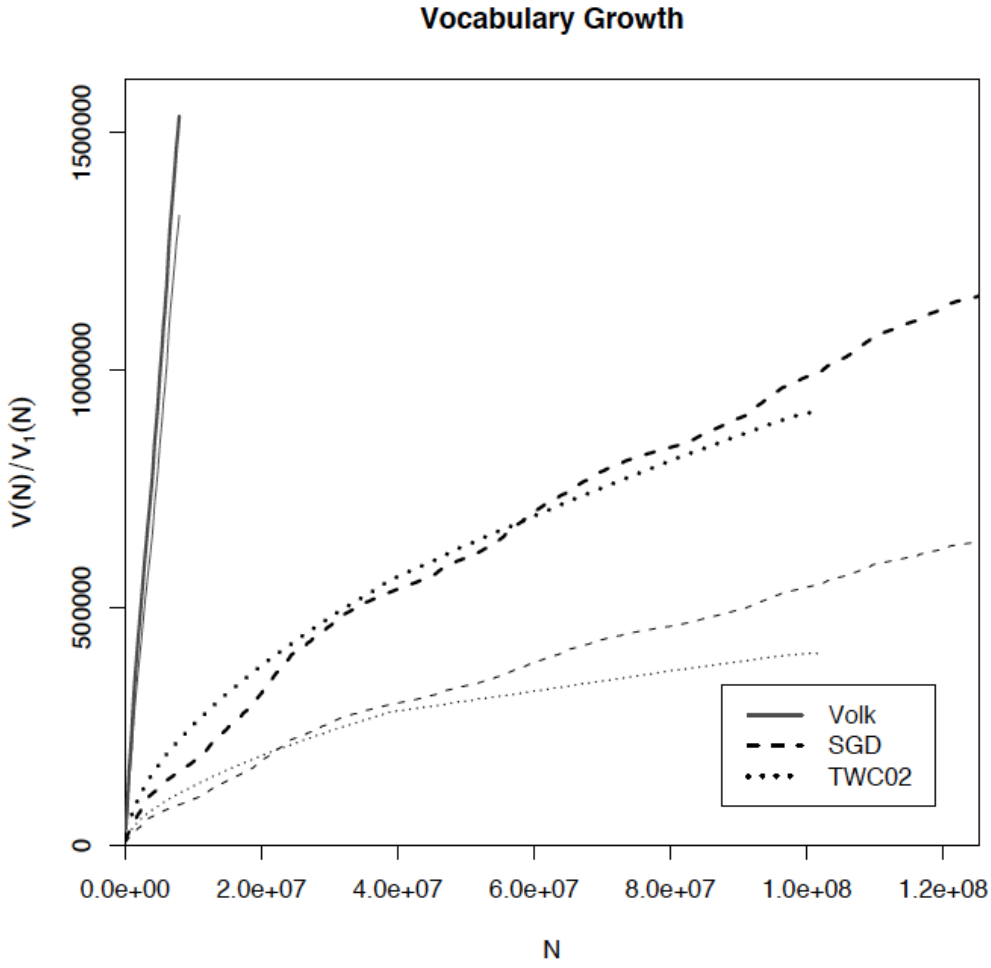
[2]URL: http://kranten.kb.nl/ In actual fact, this collection represents the results of a pilot project which will in due course be incorporated into the DDD.

cabulary continues to rise. We take this to be the effect of OCR-errors within the SGD. In sharp contrast to both these curves, the VGC of 'Het Volk' for the year 1918 (further: Het Volk 1918), one of the daily newspapers in the DDD, exhibits a markedly sharp ascent.

We list more statistics on the corpora in Table 5.1. Note the tremendous type-token ratio observed for 'Het Volk' (articles - 1918). This effectively shows the effect OCR has on the vocabulary one finds in OCR-ed text collections. TICCLops main aim is to reduce this explosion of non-desirable word forms in one's corpus so as to afford more reliable lexical statistics and to enhance search within the corpus.

Corpus	Lang.	Origin	Tokens	Types	TTR
TWC2	CD	BD	92,793,519	914,026	0.985%
SGD	CD	OCR	125,209,007	1,156,998	0.924%
DDD	HD	OCR	7,950,950	1,535,529	19.31%

Tabel 5.1: Corpora Statistics: Corpus, language (CD: Contemporary Dutch, HD: Historical Dutch), origin: born-digital (BD) or OCR-ed (OCR), number of word tokens, number of word types, type-token ratio (TTR)

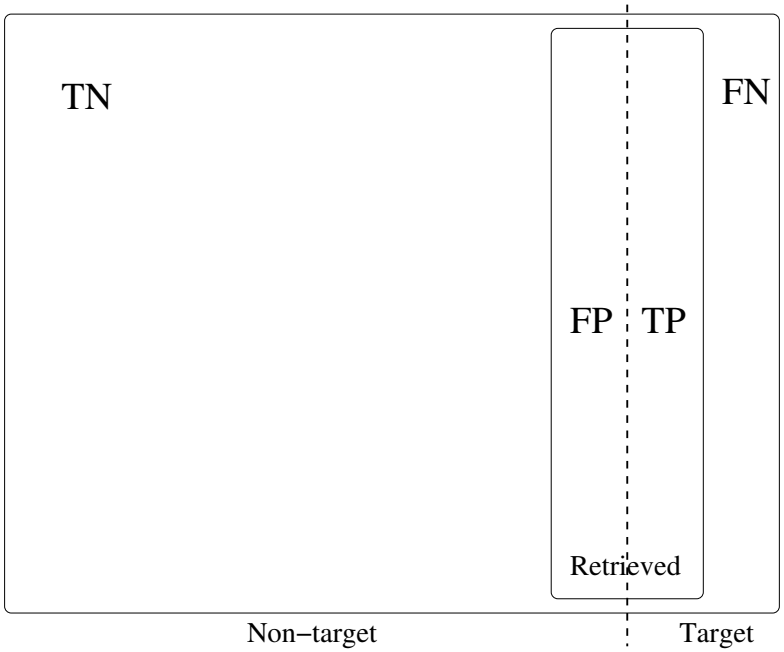


Figuur 5.2: Vocabulary Growth Curves for the contemporary reference corpus Twente Newspaper Corpus 2002 or TWC02, for the contemporary OCR-ed parliamentary debates corpus or SGD and the historical newspaper corpus 'Het Volk, articles, 1918'. Attendant finer lines depict the growth in terms of the hapax legomena

Hoofdstuk 6

Evaluation

6.1 Defining the 'task'



Figuur 6.1: Schematic representation of the task faced by a spelling error detection and correction system.

Figure 6.1 gives a graphic representation of the tasks involved in spelling error detection and correction. The large box represents the set of word strings in a text or language. The typically much larger, left portion depicts

	Target	Non-target	
Selected	TP	FP	
Not selected	FN	TN	
Totals	P	N	TOTAL

Tabel 6.1: Confusion matrix. P = positive, N = negative T = true, F = false.

the correct or acceptable word forms, the smaller portion to the right the incorrect or unacceptable word forms. These are split by a dashed line, representing the fact that the boundary between these two categories is not always razor-sharp, what is and what is not correct depending on the definition used. Words to the left of the boundary are non-target items for spelling error correction, the words to the right form the target. The diagram therefore describes the problem of distinguishing between correct words (true negatives or TN) and incorrect words (the target). The system selects a set of words of which it assumes that they are incorrect (the selected set). The false positives or FP are those retrieved that are in fact correct word forms. The part of the target which was not retrieved by the system forms the false negative or FN set. The intersection between words retrieved and the target set defines the set of incorrect words correctly identified as such and corrected (true positives or TP). The aim of any spelling error detection and correction system will be to maximize the overlap between the target and retrieved sets, achieving perfection when this is 100%.

6.2 Evaluation metrics

The interrelations between true and false positives and negatives are conventionally represented in a confusion matrix (often referred to as a contingency table). This is shown in Table 6.1. From the confusion matrix many metrics can be derived.

From the TP, FN and FP we can derive recall and precision [**?**], as follows [**?**] (p. 268-269):

- Recall = $\boldsymbol{R} = \frac{TP}{TP+FN}$
- Precision = $\boldsymbol{P} = \frac{TP}{TP+FP}$

The harmonic mean of $\boldsymbol{R}$ and $\boldsymbol{P}$, the simplified F measure, $\boldsymbol{F}$, is given by:

System	Returned	Corrected	R	P	F
	1-best True Scores				
A	10,000	100	1	0.010	0.020
B	100	50	0.50	0.50	0.50

Tabel 6.2: Results for two hypothetical correctors on the basis of a fictitious 10,000 word token text containing 100 typos. Shown are items returned, items corrected, recall, precision and F-score.

- $\boldsymbol{F = \frac{2PR}{R+P}}$

We have in fact written more about how we think spelling spelling correction systems are best evaluated in [**?**].

Now, in the hope of helping you in interpreting recall and precision scores, please consider these two hypothetical correction systems: for a 10,000 token text containing 1% of typos, i.e. 100 typos, system A returns the full 10,000 item list with 100 best-first ranked corrections. System B returns only 100 items with 50 best-first ranked correction candidates. The scores are listed in Table 6.2. The F score for system B tells us that the job was half done. For system A the F score clearly indicates that this is not so, with precision stating the obvious fact that the full list returned is a hundred times longer than the one returned by system B.

System B thus requires only one hundreth times the work to get half the job done right than system A requires to get the full job done. In the absence of fully automatic systems with great precision as well as great recall, we think system B, requiring us to examine a 100 item list to reduce error with 50% is the better option than system A requiring us to examine the full list to get the job done to perfection.

To conclude, we like to point out that recall and precision allow for direct interpretation of the results. In plain words the scores obtained by system B can actually straightforwardly be read: 'the system manages to correct half the errors present in the test set. For every error corrected it has erroneously changed one correct word in the test set into another correct word, producing 50 real-word errors in the text'.

Hoofdstuk 7

TICCLops demonstrator

A demonstration of the TICCL system is available. This demonstration uses an 18th century Dutch book, 'Kort begrip der waereld-historie voor de jeugd' by J.F. Martinet, printed in 1789.

7.1 Demonstrator book

7.1.1 Description of the digitized version

This book has been scanned and its text digitized by means of optical character recognition. It is this text, which does suffer from OCR-misrecognition errors, that we use to demonstrate TICCLops's correction capabilities. The text of the book as it should be, without any errors – the so called gold standard, is also available. The gold standard for the OCR-process has been produced at the Institute for Dutch Lexicology or INL in the framework of the European project IMPACT and was kindly put at our disposal.

Within the TICCLops project we have produced a gold standard for OCR post-correction. The main difference is that an OCR gold standard faithfully reproduces what the OCR process should have recognized what is in fact on the page, the post-correction gold standard documents exactly what should have been on the page, disregarding the actual splitting of words at the end of lines or even the fact that printed books may suffer from typesetting errors.

The TICCLops demonstration can be started using the CLAM interface by surfing to the address given in Section 3.3. CLAM is a universal wrapper for Natural Language Processing tools, developed within the CLARIN-NL framework, that handles validation of the input, and the call to the actual tool, TICCL in this case. The output of TICCL is made available by CLAM for viewing or downloading. How to set up and start CLAM is described in

Van Gompel (2010).

7.1.2 Introduction to the book's textual characteristics

As a further prelude to our discussion of the TICCLops's demonstrator we first briefly introduce some of the pertinent textual characteristics of the demonstrator book. We list why these are pertinent in view of TICCLops's current capabilities.

This text was printed in 1789 and was written in a non-standard version of Dutch. The book runs to 362 pages, but many of these carry little and some no text at all. The digital version amounts to less than 60,000 untokenized word tokens. As corpora go, this is not a lot of text to derive very reliable lexical statistics from.

Many of the word tokens are short words, the text being aimed at children. A lot of longer words were split up by the OCR-process. The current version of TICCLops does not deal with split words. This requires working with bigram frequency lists, i.e. lists providing the frequencies of two consecutive words in the texts. We have chosen to defer doing this for OCR-ed text to a later time.

We do not have a validated lexicon for the time period in which the book was written. What we have is a lexicon that has the 1914 version of the official Word List for Dutch, borrowed from the Project Gutenberg.

The text deals with world history as it was conceived in rather biblical terms at the time. As such it has a great many names of historical figures and more or less foreign peoples and places, a lot of which are in a fickle spelling. Some of these are not too familiar to readers today.

The print was printed in what is known as 'Fraktur': many of the characters we would now regard as 's' where then printed in what closely resembles an 'f' but did not usually have the serif. The OCR software that was used by the KB to digitize the book was developed for contemporary print, i.e. does not have specific provisions for dealing with older print. Indeed, the main purpose of the European project IMPACT was to improve OCR-technology for older print.

In summary: the current TICCLops demonstrator book presents a real and interesting challenge to any spelling or OCR post-correction system.

In the following section we will describe and discuss how well TICCLops manages to cope. In the process we will contrast some of the available parameter settings and lexicon choices and try to elucidate the results obtained by using them.

7.2 The Martinet demonstrator

In what follows we will focus on what the system makes of just a few of the book's paragraphs, taken from demonstrator input file dpo_35_0344_text.xml, which after processing is renamed to dpo_35_0344_text.ticcl.xml. These paragraphs deal with the exploits of Ferdinand Magellan and Hernán Cortés (English spelling for these names borrowed from the English Wikipedia).

In the following subsections we faithfully reproduce the system's XML output.

Be advised that when one opens one of these files, e.g. the original named above or the version produced by TICCLops, one's browser is likely to interpret the XML. This results in an identical view of both files in that only the original textual content is shown, as in Figure 7.2. If one, therefore, would like to inspect TICCLops's corrections, added in XML as in the examples below, one needs to first download the file and then open it in an editor such as e.g. emacs. Another option is to select to view the page's source code in your browser.

7.2.1 Using the Historical dictionary

We present the result of TICCLops's rewriting of a few sample paragraphs from the demonstrator file named above.

This result was obtained with the Historical dictionary. The settings chosen were the following:

- We loaded the pre-installed demonstrator corpus
- We loaded the Historical-contemporary Dutch lexicon
- We set the system parameters as follows:
  - word frequency: 1-100
  - word length: 7-100
  - maximum edit distance: 2
  - N-best ranking: Up to 3 N-best ranked
  - checkboxes Evaluation and Conversion were checked
  - We also checked box Text Correction
- Next, we clicked on *Start*.

A note on the number of variants presented: TICCLops allows for up to 20 variants to be retrieved. The user should be aware that in all cases the number actually set is the number of variants maximally returned by the system. In a number of cases this may be less, which is then the result of the fact that within the limits set by the edit distance parameter allowed, no variants were in fact encountered.

Also, the variants returned by the system are in all cases ranked best-first.

```
<p> A. Ferdinand <sc variants="nagellak;magellaan">
    Magellan</sc> <sc variants="ontdekt;ontdekte;
    ontdekten">cntdekte,</sc> na het door Zeilen eener
    Straat, naar hem daarna genoemd, dat men, om den <
    sc variants="zuidelijken;zuidelyke"> Zuidelyken</sc>
     uithoek van America, in de groote ftille Zuidzee
    kon komen, en dus de <sc variants="wereld;weergeld;
    gereld">Waereld</sc> omzeilen. </p>
<p> 38 V. Maar werdt Cortes niet <sc variants="
    treflijk"> treflyk</sc> beloond voor zulke
    uitmuntende daaden ten beste van Spanje? </p>
<p> A. Hy vondt nydige <sc variants="menschen;Munchen"
    > menfchen,</sc> en het ging hem, gelyk Columbus:
    want hy werdt ook federt met <sc variants="verging;
    overacting;verachting"> veragting</sc> door het
    Spaanfche Hof behandeld, hoewel hy daarna California
     ontdekte, en flierf dus , flegt beloond , in het
    jaar 1558.</p>
```

- Short words:

  What we here observe is that a number of shorter words that do show OCR-errors and/or spelling variation are not corrected, i.e. 'hy' which today is spelled 'hij' (E: he) and 'slierf' a non-word that should have read 'stierf' (E: died) in the last sentence. This is explained by the fact that we set the word length parameter to '7' .

  In terms of performance this is important: if one allows the system to try and correct short words, one may correct more errors. However, the system is also likely to not be able to rank the correction candidates, i.e. the possible variants it retrieves, properly because the confusion between these is much larger - there being many more - for shorter words. This will then have a, possibly serious, cost in the precision of the system.

One should carefully consider what it is what one hopes to achieve by having TICCLops clean-up a corpus. Does one want to have the text fully corrected, at the cost of probably having to spend manual effort? Or does one want to improve the recall of searches within the corpus by automatically cleaning-up say only longer words at a lower cost?

- Ranking issues

  The more common Dutch variant 'Magellaan' for the name 'Magellan' is retrieved, but ranked in second position after 'nagellak' (E. nail polish).

  This example shows that ranking the correction candidates can still be improved in the system.

- Correctly resolved OCR-errors

  The historical word form 'Waereld' is resolved best-first to its contemporary, uncapitalised variant 'wereld'.

  Both for 'cntdekte' and 'menfchen' we have plausible best-first ranked variants, although the first represents another morphological variant and the second is resolved to its correct historical spelling, not to its contemporary spelling. The latter is explained by our setting of LD = 2: resolving 'menfchen' to 'mensen' comes at the cost of 3 edits, which was therefore out of TICCLops's reach in this run.

7.2.2 Examining the evaluation scores

We next present and discuss some of the evaluation scores as can be found in de log file.

```
TOSCORELD: 1 TP: 655 FN: 612 FP: 49
TOSCORELD: 2 TP: 640 FN: 388 FP: 361
TOSCORELD: 3 TP: 4 FN: 459 FP: 43
SCORELD: 1 R: 0.517 P: 0.930 F: 0.665
SCORELD: 2 R: 0.623 P: 0.639 F: 0.631
SCORELD: 3 R: 0.009 P: 0.085 F: 0.016
SCORELDCUMUL: 1 R: 0.513 P: 0.921 F: 0.659
SCORELDCUMUL: 2 R: 0.557 P: 0.756 F: 0.641
SCORELDCUMUL: 3 R: 0.466 P: 0.738 F: 0.571
```

For each Levenshtein distance we get the counts for the true positives, the false negatives and the false positives. So 'TOSCORELD: 1' lists these for LD 1.

Next we get the score as calculated for that LD only, so SCORELD: 2 lists the recall, precision and F score obtained on pairs of canonical word forms that differ from their OCR-error variants by just 2 edits.

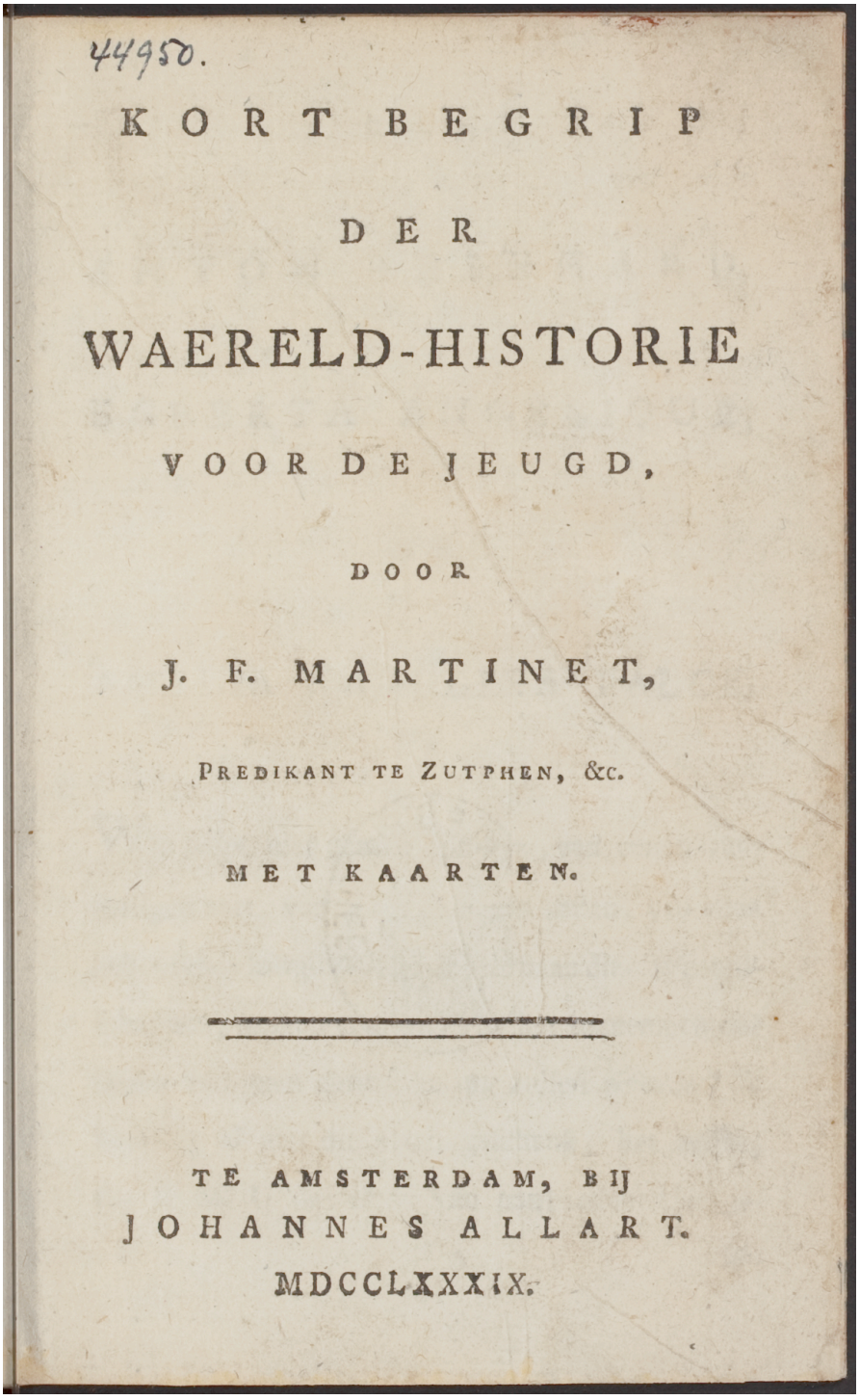
In SCORELDCUMUL: 2 we then get the recall, precision and F score averaged over both LD 1 and 2 pairs.

It can be seen that the system performed very badly at LD 3. In fact, we ran it with LD limit 2 so this should come as no surprise.

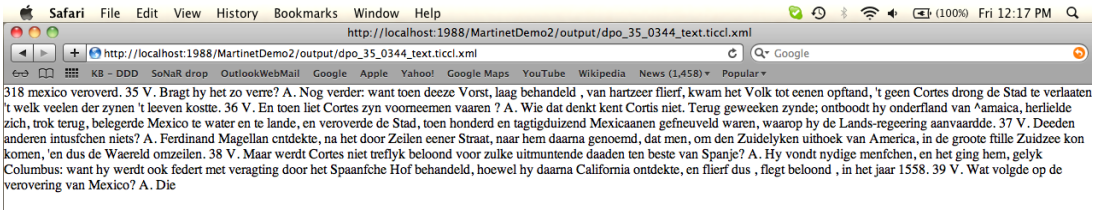
The system performs less well in terms of recall at LD 1 than at LD 2. This too is not surprising: there are typically more LD 1 errors in a corpus than larger edit errors. We have in fact very clear statistics on this in [**?**]. Also LD 1 errors are more likely in shorter words. Shorter words in turn are on the one hand more frequent in a corpus and were on the other hand also not targeted in this run.

Further, it is clear that given that any word typically has less neighbours at LD 1 than at LD 2, the system is more likely to retrieve and correctly rank the correction candidates at LD 1. This has a clear beneficial effect on precision at that level.

Given the cumulated scores at LD we can now state that with these settings, given the difficult task presented by the demonstrator book, TICCLops neverheless performs quite well. The system corrected more than one in two of the errors in the OCR-ed pages. Only in about 1 out of every four corrections made, the system was wrong.



Figuur 7.1: TICCLops demonstrator book (1789 paper version)



Figuur 7.2: The text of Demonstrator file dpo_35_0344_text.xml as viewed with a browser, here Safari

Hoofdstuk 8

Future TICCLops developments

The current version of the TICCLops demonstrator in fact presents a simple if not somewhat naive approach to corpus clean-up. In its essence it works on raw text, whether or not maskerading as xml. It relies on the focus word approach to spelling correction, which is fine for shorter texts but not conducive to efficient clean-up of the large corpora we increasingly have at our disposal. It is agnostic of the actual language(s) that appear in the texts it processes.

All this is due to change within the next few months.

TICCL continues to be developed in projects such as NWO Middelgroot Political Mashup, CLARIN Flanders and NL project TTNWW and CLARIN-NL project VU-DNC.

Results of these new developments will eventually be available through the web application and web service TICCLops as well.

8.1 XML processing

These developments include full xml-parsing and correction capabilities.

In Political Mashup we have added streaming xml processing to TICCL which scales to the largest corpora sizes. This was deployed to correct the full texts of the Acts of Parliament running from 1814 to 1995, which were digitized by the Koninklijke Bibliotheek.

8.2 Language recognition

The Acts often hold non-Dutch text. We have extended TICCL to perform language recognition on the level of text paragraphs by incorporating the language recognizer TextCat developed by Gertjan van Noord (URL: http://odur.let.rug.nl/ vannoord/TextCat/).

8.3 FoLiA and Alto XML support

XML processing in TICCL relies on XPath expressions. This allows for efficient selection of only those text components that require clean-up, while safely leaving the componnets that should not undergo any changes, untouched.

XML is often very specific and task-oriented. This creates a bewildering diversity which it is impossible to cater for in a single tool. For this reason we promote the use of FoLiA xml, developed in the framework of TTNWW. TICCLops will soon be extended with full FoLiA support.

In cooperation with the TICCLops project user Koninklijke Bibliotheek, we are also adding Alto XML support to TICCL. The text of the Martinet book digitized by the KB incorporated in the current version of TICCLops is the basic XML version that is employed by the KB to be presented to the user of its digitized collections when the user asks to see the digitized text after she has performed a query. The texts are nevertheless also present in a more developed XML format, Alto XML, which are in fact instrumental in highlighting the term queried for on the image retrieved.

Hoofdstuk 9

Appendix

9.1 Acknowledging TICCLops

If TICCL has proven valuable to in your own research, please refer to the following publication:

[?]

9.2 TICCLops screen shots

Figuur 9.1: TICCLops start-up project screen

Figuur 9.2: TICCLops project screen with two predefined projects

Figuur 9.3: TICCLops screen for selecting the input (top half)

Figuur 9.4: TICCLops screen after DEMO data have loaded

Figuur 9.5: TICCLops screen after lexicon data have loaded

Figuur 9.6: TICCLops screen for file upload: file type selection dropdown list

Figuur 9.7: TICCLops parameter selection screen (lower half input selection page)