Automated Proof Reading of Clinical Notes

Jon Patrick and Dung Nguyen

School of Information Technology, University of Sydney,
1 Cleveland St, Sydney 2006, NSW, Australia
{jonpat, nguyend}@it.usyd.edu.au

Abstract. Misspellings, abbreviations and acronyms are very popular in clinical notes and can be an obstacle to high quality information extraction and classification. In addition, another important part of narrative reports is clinical scores and measurements as doctors infer a patient's status by analyzing them. We introduce a knowledge discovery process to resolve unknown tokens and convert scores and measures into a standard layout so as to improve the quality of semantic processing of the corpus. System performance is evaluated before and after an automatic proof reading process by comparing the computed SNOMED-CT codes to the coding created originally by the clinical staff. The automatic coding of the texts increased the coded content by 15% after the automatic correction process and the number of unique codes increased by 4.7%. Accuracy of the automatic coding and annotations in the notes which have not been coded by the clinical staff is suggested by the system output.

Keywords: clinical, proof reading, spelling, normalization, standardization.

1 Introduction

Clinical notes contain valuable information about patients' status, however, retrieving information from them is challenging because they may comprise up to 30% non-word tokens, idiosyncratic spellings, abbreviations and acronyms, and poor grammatical structure. Besides resolving misspellings, knowing the correct expansions of abbreviations and acronyms is critical to the understanding of the document for both automatic natural language understanding and human comprehension and interpretation (Pakhomov et al., 2005).

Proof reading is a process whereby a clinical text is validated to identify unknown tokens/words and their valid forms. Proof correcting is modifying the proofed text to make it notionally "correct" text and thereby more readily processible by automatic means. There are two principal tasks to be achieved, these are `normalization` and `standardization`. The normalization process changes the texts in a way so that a human reader would consider it as normal, such as correcting spelling, expanding abbreviations and acronyms. The standardization process converts the text into certain formats that an expert community has defined as standard; a good example is converting scores and measures into a standard layout.

At first glance it would seem that standardization should be done initially before normalization, however it is more likely that both will need to be performed multiple times in a repeated cycle of processing as there is interaction between the two processes. Standardization converts various instances of the same scores and measures into standard forms so that the system does not need to be concerned about their details for later processing steps such as normalization. For example: "HR 70" and "HR 78" are standardized to the representation of heart rate. However, most measurements and scores contain acronyms or abbreviations which may need to be expanded during the normalization process. Normalization could improve the standardization process by correcting misspelling and other token error within standard forms. If normalization is executed first, a large amount of tokens within standard layouts will need to be processed while they could be excluded if they were ringfenced by a standardization process.

Once both these tasks are completed the result is a corpus that is annotated by all of these processes. The final act is `proof correcting` or `transformation` which is to change the raw corpus into a proofed corpus, that is, the text can be read as a fully corrected corpus. The important process is to use the annotation properties of each token to change its representation in the source file to the correct form. This produces two versions of the source file, the uncorrected form and the corrected form. The former form has a set of annotations with properties defining the changes that needed to be made in the proof correcting process, and the latter form has all the text corrected and a set of annotations that define the original form of the token(s) and structure(s).

2 Related Work

The normalization process contains misspelling corrections, abbreviation and acronym expansions. Spelling error detection and correction can be classified into two categories (Ruch et al., 2003). The first category is word-based or context-free spelling correction which resolves errors for words that cannot be found in the lexicon (such as 'medicla' is a misspelling of medical). The second category is context-based or context sensitive spelling correction which concerns a valid word but misspelt within the context (for example, in 'a peace of paper' where 'piece' is misspelt).

The classical word-based spelling correction algorithm is minimum edit distance which ranks suggestion candidates by the minimal number of insertions, deletions, substitutions and transpositions needed to transform one string into the other (Levenshtein 1966). The Metaphone algorithm uses consonant symbols which represent their usual English pronunciations, the vowels 'AEIOU' are also used, but only at the beginning of the word (Philips 1990). In more recent research, Kukick (1992) maps every string into a key such that similarly spelt strings will have identical keys; this method is called similarity key. Some spelling suggestion tools such as Aspell and Gspell which combine multiple algorithms are now available for use and research. Aspell is a combination of the Metaphone algorithm and near-miss strategy by its predecessor Ispell (Atkinson 2006; Kuenning 2006). The mix of algorithms in Gspell includes the NGrams, metaphone, common misspellings, and homophone retrieval tools. Candidates are evaluated by the Levenshtein edit distance, and similar ranked candidates are re-ordered by use of word based corpus frequencies (Divita, 2003).

Much research on normalization has been developed in the medical domain due to a high frequency of misspellings, abbreviations and acronyms. A frequency-based technique combining a comprehensive and a medical dictionary configuration was developed to improve suggestion ranking of Aspell and Gspell (Crowell et al., 2004). Without a comprehensive dictionary, Turchin et al. (2007) identified misspelt words using prevalence analysis. Senger et al. (2010) used Aspell and user behavior to analyze drug misspelling characteristics in a drug query system.

Spelling correction is more effective when the method takes into account the context in which the word occurs. To improve spelling correction in the electronic patient record, Ruch et al. (2003) uses lexical disambiguation and named-entity recognition, and shows how a set of natural language processing (NLP) tools can be combined to improve the processing of clinical records. Emphasis on first suggestion accuracy in Patrick et al. (2010) introduced a high accuracy spelling corrector for clinical notes which uses a combination of rule-based suggestion generation and a context-sensitive ranking algorithm.

A comparative study of supervised acronym disambiguation in a corpus of clinical notes, using three machine learning algorithms: the naïve Bayes classifier, decision trees and Support Vector Machines (SMVs) has been conducted by Joshi et al. (2006). Instead of using three machine learners, Joshi el al. (2006) also developed three kernels for SVMs – one that makes use of knowledge derived from unlabeled text, the second using semantic knowledge from

ontologies, and finally a third, additive kernel consisting of the first two kernels – and studied their effect on the tasks of word sense disambiguation and automatic expansion of ambiguous acronyms. A method for collecting training data for supervised machine learning approaches to disambiguating acronyms has been introduced by Pakhomov S. (2002). The approach is based on the assumption that the expansion of an acronym and the acronym itself usually occur in similar contexts. The closest work to clinical document normalization is Wong et al. (2006), who integrated scores for spelling error correction, abbreviation expansion and case restoration.

Measurements and scores are text patterns that need to be identified in the clinical notes. Finite State Automata (FSA) is a pattern matching approach that is used in our standardization process. FSA have many applications in NLP such as pattern matching, named-entity recognition and partial parsing. A language-independent method of finite-state surface syntactic parsing and word-disambiguation is introduced and discussed in (Koskenniemi, 1990). In this work, finite-state machines represented syntactic constraint rules where each constraint excludes certain types of ungrammatical readings. From the view of computational efficiency, the use of finite-state automata is motivated by taking into account optimal time and space (Mohri, 1997). Abney (1996) used cascaded FSAs to parse free text while Aït-Mokhtar and Chanod (1997) utilized incremental finite-state machines to build a shallow parser. Both of these researchers highlighted the efficiency of the FSA in that they can be extended at modest cost, maintain broad coverage and linguistic granularity and do not necessarily involve trading off accuracy against speed. This characteristic of an FSA is very useful in recognition of scores & measurements in the clinical domain where there is a large number of different patterns and new incoming examples which require rapid re-training of FSA. Furthermore, Finite-state transducers are used for deterministic part-of-speech tagging and semi-structured data extraction from the web (Roche and Schabes, 1995; Hsu and Dung, 1999). For probabilistic FSA, CSSR is an algorithm that generates weighted FSA from training data used to identify named-entities in text (Padro and Padro, 2007). In the CSSR algorithm, the model changes its structure to satisfy new training data.

To our knowledge, our system is the first automated proof reading system for clinical data which combines normalization and standardization into the cycle of processing with evaluation of the result using a SNOMED-CT[1] concept identifier versus clinical staff's assigned codes.

3 Method

The important system requirement is to create a process for the automated proof reading of a clinical corpus so that it can be used to improve the information retrieval accuracy of clinical knowledge in the text.

The combination of normalization and standardization guarantees that standard expressions (scores and measures) are captured and each token contains its lexical verification information (abbreviation, acronym expansion and misspelling correction) whether the token is standing outside or within multi-token expression.

The result of normalization and standardization process will be stored as annotations in the notes. This approach enables different subsets of annotation type to be used in specific processing tasks or experiments (detailed in section 4).

3.1 Corpus Description

The corpus used in our experiment is the Concord hospital's clinical progress summary which contains 43712 anonymised records from 2003 to 2008. Each record contains information about Principle diagnosis, Additional diagnosis, SNOMED-CT Description Identifier (DID),

[1] Systematized Nomenclature of Medicine - Clinical Terms (SNOMED-CT). Available at http://www.ihtsdo.org/snomed-ct

Description and Progress working text. 22974 of the 43712 notes contain SNOMED-CT DID and description (one for each note) which are assigned manually by clinical staff.

3.2 Tokenization

Three kinds of tokenizing strategies for different levels of tokenization are utilized: the standard morphology tokenizer splits text into single tokens for lexical verification, while the ring-fencing tokenizer recognizes multi token expressions as standard types of scores or measurements, and the combined tokenizer. For example, the standard tokenizer can split "HR 72" into the word "HR" and number "72" so that HR can be further verified and classified into the acronym with expansion "Heart Rate". Then the ring-fencing tokenizer recognizes the phrase "HR 72" as a standard measurement of type heart rate. The combined annotation will have two overlapping levels: "HR" is an acronym, "72" is a number and "HR 72" annotated as a measurement of type HR (heart rate).

The standard tokenizer uses morphology defined by regular expressions (REs) for basic classification of word and non-word tokens. This tokenizer classifies non-word tokens into sub-classes: date (e.g. 3/7/02), time (e.g. 11.30am), range (e.g. 0.01->0.49), complex digit (e.g 0.52/0.44), digit (e.g 652), separator (e.g. ##, **), operator (e.g. +, -), punctuation (e.g. !, ?). Word tokens are divided into single word (e.g. patient) and compound words such as two_word_slash (e.g. d/c), two_word_hyphen (e.g beta-blockers), two_word_apostrophe (e.g. didn't) and more_than_two_words (e.g. behind-the-wheel). The lexical processing of each word component in compound words is similar to single word.

The ring-fencing tokenizer is a Finite State Recognizer (FSR) which uses training example patterns to recognize token patterns constituting a score or measurement that requires standardization (Patrick and Sabbagh, 2011). There are a large number of different scores and measurements in clinical notes. When using REs to describe patterns as more rules are developed to capture missed items, the rules became so complicated that it makes them difficult to update as any change has the risk of losing previously recognized patterns or introducing new false positives. Another problem is that the rule updating task requires an exhaustive knowledge about REs and a considerable amount of time modifying the rules. Consequently, the automated learning process to capture patterns using REs is particularly difficult. On the other hand, a trainable FSR can be built directly from training examples of data with high accuracy and efficient computational time. Some other types of measurements and scores in the training patterns are illustrated in table 1. The FSR training file has a simple format and contains two columns, first is the type and second is the span of the text which expresses the pattern. Training examples are then generalized so that the FSR can capture all the similar forms of these patterns.

Table 1: A subset of training Types.

Type	Pattern	Type	Pattern
BP	BP 140/65(84)	ABG	ABG's: 0355 7.41/41/103/26/2
SaO2	O2 sats 91%	Measurement	7mg/hr
Temp	37.6	O2-measurement	2L O2
Lipids	Lipids 10% at 20mls/hr	DRNAME	Dr. <:[A-Z][a-zA-Z]*<: [A-Z][a-zA-Z]*:>

The combined tokenizer uses the standard and ring-fencing tokenizer. It uses both morphology defined by REs and FSRs to tokenize text where measurements and scores can be recognized as multi-word expressions with no separated tokens, other tokens are split and classified by REs method.

3.3 Proof Reading and Proof Correcting

After standard tokenization, each token is passed through the lexical verification process and then inserted into the Lexicon Management System (LMS) which supports automated and manual resolution of unknown tokens. The LMS is a system developed to store the accumulated lexical knowledge and contains categorizations of spelling errors, abbreviations, acronyms and a variety of non-word tokens. It also has a web interface that supports rapid manual correction of unknown words with a high accuracy clinical spelling suggestor plus the addition of grammatical information and the categorization of such words into gazetteers (Patrick et al., 2010). The method of the clinical spelling suggestor is based on combining heuristic-based suggestion generation and ranking algorithms based on word frequencies and trigram probabilities. This approach achieved high accuracies on test data sets with over 93.5% for the Concord corpus. By using the LMS to resolve unknown words, the Concord lexicon database contains approximately 15000 tokens that have been manually corrected. Figure 1 illustrates the lexical verification process supported by our resources which includes 7 checking steps (1) Misspellings (2) Abbreviations (3) Acronym (4) Gazetteers (5) Moby lexical verifier for English[2] (6) UMLS dictionary of medical terms[3] (7) SNOMED-CT dictionary.

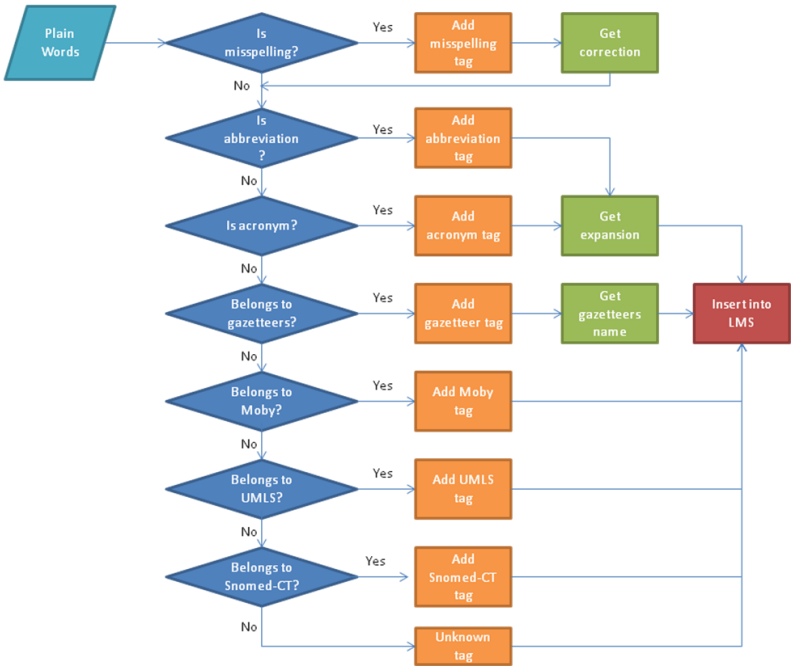

Figure 1: Lexical verification process.

The lexical verification process contains an additional step to resolve misplaced whitespace and punctuation. The LMS manages a developed process for manual spell correcting but it only considers single tokens hence errors of misplaced white space are not processed in it. The misplaced whitespace problem is dealt with in an external computational process. For example: "looka fter" should be "look after", and the LMS will give suggestions for each word. However, these words should be resolved together. The rate of this kind of error is not high in a single

[2] The Institute for Language, Speech and Hearing. Available at http://icon.shef.ac.uk/Moby

[3] Unified Medical Language System (UMLS). U.S National Library of Medicine, National Institutes of Health. Available at http://www.nlm.nih.gov/research/umls

document but for a large corpus, they are costly to manually resolve. Examples of incorrect punctuation problems are “natio;n” where it should be “nation” and “nation.The” has missing whitespace. Currently the LMS tokenizes each string on the left and right of the punctuation and so a correction has to be made in another process.

At the end of the proof reading step, annotation files which contain lexical information from a corpus are generated. This means every token is annotated so any correction or expansion can be made when needed. The misspelt words will have correction information while abbreviations and acronyms come with expansion within their annotations. If a token is correct, it should contain information about the resource where the token was validated and mapped to the name of gazetteer or dictionary. Table 2 shows tag type statistics used in the proof reading process.

Table 2: Frequencies of tag types used in the proof reading process.

Tag type	Frequency
Abbreviation	47,564
Acronym	149,372
Misspelling	93,477
Non-word	1,385,184
Unknown	3,162
Valid Words	4,608,664
Total	6,287,423

The proof correcting process generates the proof-corrected annotation files. They contain annotations that enable the user to see which tokens have been changed from the original corpus. In order to evaluate the effect of specific normalization types, proof correcting can generate corrected text files from annotation files with a selected set of annotation tags (misspellings, abbreviation, and acronym).

3.4 SNOMED-CT Code Annotating and Comparison

To evaluate the effect of the proof reading and correcting process, an algorithm for converting Text to SNOMED-CT (TTSCT) is used to annotate the corpus before and after the proof reading process to see the improvement and identify the distribution of SNOMED-CT concepts over the corpus (Patrick et al., 2007). TTSCT was developed so that SNOMED-CT concepts can be identified in free text narratives and to annotate them with the clinical reference terms. The accuracy of TTSCT is approximately 70% on a test corpus. Its improvement is ongoing research conducted by the authors. This evaluation is based on the assumption that if TTSCT can identify many more clinical terms in the proofed corpus than the original one, the proof reading process is considered effective because terms in the misspelt words, acronyms and abbreviations are now revealed. The measurements and scores patterns usually contain abbreviations and/or acronyms; so, we would expect there are more patterns after executing proof correcting using the misspelt tag only.

Another purpose for computing SNOMED-CT codes for the corpus is to compare with assigned DIDs from clinical staff and generate suggestions codes for unassigned DID notes. In the SNOMED-CT resource, each Concept Identifier (CID) may contain several DIDs. This step checks whether the manually assigned DID belongs to a computed SNOMED-CT concept in each note and generates CID candidates for unassigned notes (20738 unassigned notes / 43712 notes). When computing candidates for unassigned notes, we are especially interested in the two most popular assigned and computed classes: Clinical finding and Body structure.

4 Experiment Result and Discussion

Generally, when using all corrections and expansions, we found more SNOMED-CT codes instances (over 150,000 or 14.3%) and distinct concepts (842 new concepts or 4.7%) in every SNOMED-CT category except Record artifact (Table 3). When only applying the misspelling tag in the proof correcting process, there are only around 22,000 new SNOMED-CT instances and 268 new concepts. Consequently, most new SNOMED-CT codes come from expansion of abbreviations and acronyms (85.4%) as the SNOMED-CT recognizer could not map these concepts. We can conclude that when applying an automated proof reading process, the clinical note is more informative as more medical concepts and instances are revealed.

Table 3: Frequency of SNOMED-CT upper-level categories before and after proof reading. The figures in brackets show the number of distinct concepts in each category.

Category	Before	After	Difference
Clinical finding	441,498 (8,673)	497,486 (9,052)	55,988 (379)
Body structure	184,679 (2,462)	199,748 (2,542)	15,069 (80)
Procedure	93,267 (2,181)	108,165 (2,345)	14,898 (164)
Substance	75,630 (1,050)	79,509 (1,096)	3,879 (46)
Observable entity	74,229 (792)	86,926 (836)	12,697 (44)
Social context	70,243 (439)	104,141 (455)	33,898 (16)
Physical object	32,623 (556)	38,003 (567)	5,380 (11)
Situation with explicit context	26,275 (402)	32,531 (452)	6,256 (50)
Environment or geographical location	22,516 (287)	22,936 (305)	420 (18)
Pharmaceutical / biologic product	15,472 (467)	16,331 (486)	859 (19)
Event	10,576 (158)	12,407 (162)	1,831 (3)
Staging and scales	7,377 (19)	7,407 (21)	30 (2)
Physical force	7,026 (39)	8,364 (42)	1,338 (3)
Organism	5,450 (289)	5,475 (292)	25 (3)
Specimen	676 (35)	770 (39)	94 (4)
Record artifact	221 (13)	221 (13)	0 (0)
Total	1,067,758 (17,862)	1,220,420 (18,705)	152,662 (842)

On the other hand, most ring-fencing patterns contain abbreviations or acronyms. When using ring-fencing after misspelling corrections and abbreviation expansion, we lost more than 3000 patterns in the corpus (for example BP is an acronym which is widely used in most blood pressure ring-fencing examples, if we expand BP we may lose some of these patterns). When applying misspelling corrections only, we found slightly more patterns. Table 4 indicates the detailed number of high frequency types in scores & measurements before and after processing, other types with low frequency (<50) are PaO2, ST, Ward, F, Alb, PaCO2, PS, SiO2, RENAL, FiO2, BSL, Creat, TV.

Another experiment with the Concord database was to check the coincidence of manually assigned SNOMED-CT codes for each note from the principle diagnosis, additional diagnosis and progress text with values returned by TTSCT; we found 10465 over 22974 (45.48%) matches before proof reading with 1200 distinct concepts. After proof reading, the proportion of matched DIDs increased to 48.41% (10981 matches/ 22974 notes with DID) with the addition of 25 new concepts. From the experiments, it was found that the two most popular DID classes are Clinical finding (82.1%) and Body structure (6.2%). Furthermore, a maximum of 8 different DIDs were referenced to a single CID. In addition, some assigned DIDs can only be

found in the progress working text (1785 before and 1834 after proof reading process), this means in this case the clinicians may need to pay more attention to the content of the patient's note to decide the representative DID for that note rather than solely based on diagnosis sections. In approximately 12,000 notes that the system could not find any match between the assigned codes and the computed codes, some examples were analyzed and the common explanation for the mismatch is that the assigned DID is a general term (parent class) of the computed codes which are found in the notes. Combining multiple codes of the same class to infer the possibility of parent class could be a future enhancement for the system. Table 5 shows the distribution of matched DID within the notes.

Table 4: High frequency scores & measurements before and after proof reading using misspelling tags only.

Tag	Before	After	Difference
Measurement	13,638	13,650	12
BP	4,004	4,006	2
GCS	3,227	3,231	4
O2-Measurement	3,220	3,220	0
RR	2,988	2,988	0
HR	2,584	2,585	1
SaO2	2,270	2,276	6
Temp	1,836	1,846	10
Hb	846	847	1
PEARTL	838	846	8
PR	810	810	0
K	289	289	0
pH	142	142	0
Total	36,692	36,736	44

Finally, most of the unassigned notes contain information about at least one of two of the most popular classes which have been found in both assigned and computational SNOMED-CT codes. Within 20738 unassigned notes, 98.9% contain a clinical finding and 90.1% contains Body structure (86.2% contain both classes). This result means that many notes with unassigned SNOMED-CT DIDs might have the relevant codes computed from the annotated corpus. A list of CID candidates for each note is generated by the system and will be validated by clinical staff as well as the correctness of assigned DIDs against computed DIDs.

Table 5: Locations of matched DIDs in notes.

Section	**Before**	**After**
Principle Diagnosis	8594 (82.12%)	9058 (82.49%)
Additional Diagnosis	219 (2.09%)	251 (2.29%)
Working Progress	5315 (50.79%)	5670 (51.63%)

5 Conclusion

In this study, a general approach for proof reading of clinical notes is developed and evaluated. This work also introduces and illustrates the necessity of combining standardization and normalization into the cycle of the proof reading process in the clinical domain. The combination of REs and trainable FSR guarantees general classification of tokens and clinical pattern recognition. Automated lexical verification and high accuracy clinical spelling suggestion are supported in the LMS interface which then enable high accuracy and efficiency at manual resolution of unknown tokens. As a result, the proofed corpus becomes much more informative for automatic information extraction as well as human comprehension and interpretation. Finally, the method is easily adapted to other domains or others languages by re-defining the training patterns for FSR and central resources used in the proof reading process (dictionaries, gazetteers …).

The limitation in our work is that most abbreviations and acronyms are mapped directly from the accumulated dictionaries or manually expanded by using the LMS. The future development for our system is to apply an automated context-sensitive and probabilistic abbreviation, acronym expansion suggestion to support the lexical verification process and the LMS. A more general method for extracting and comparing computed SNOMED-CT codes with assigned codes to enable hierarchical inference is currently has our attention for future research. Furthermore, the clinical staffs will be involved in the evaluation process to have a better estimation of system performance.

References

Abney, S. 1996. Partial parsing via finite-state cascades. *Journal of Natural Language Engineering*, 2(4), 337-344.

Aït-Mokhtar S. and J.P. Chanod. 1997. Incremental finite-state parsing. *In Proceedings of the fifth conference on Applied natural language processing (ANLC). Association for Computational Linguistics*, pp. 72-79.

Atkinson, K. 2006. Gnu aspell 0.60.4, http://aspell.sourceforge.net/.

Crowell, J., Q. Zeng, L. Ngo and E.M. Lacroix. 2004. A Frequency-based technique to improve the spelling suggestion rank in medical queries. *Journal of the American Medical Informatics Association*, 11:179-185.

Divita, G. 2003. SPECIALIST Spelling Suggestion Tools (Gspell), produced by National Library of Medicine. Available at http://lexsrv3.nlm.nih.gov/LexSysGroup/Projects/gSpell/current/GSpell.html

Hsu C.N. and M.T. Dung. 1999. Generating finite-state transducers for semi-structured data extraction from the Web. *Journal of Information Systems*, 23(8), 521-538.

Koskenniemi, K. 1990. Finite-state parsing and disambiguation. *Proceedings of the 13th conference on Computational linguistics*, 2, 229-232.

Kuenning, G. 2006. International ispell 3.3.02. Available at http://lasr.cs.ucla.edu/geoff/ispell.html.

Kukich, K. 1992. Technique for automatically correcting words in text. *Association for Computing Machinery (ACM) Computing Surveys*, 24(4), 377-439.

Levenshtein, V. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8), 707–710.

Mohri, M. 1997. Finite-state transducers in language and speech processing. *Journal of Computational Linguistics*, 23(2), 269-311.

Padro, M. and L. Padro. 2007. ME-CSSR: an extension of CSSR using maximum entropy. *Proceeding of the Conference on Finite-State Methods for Natural Language Processing (FSMNLP)*, pp. 161-165.

Pakhomov S., T. Pedersen and C.G. Chute. 2005. Abbreviation and Acronym Disambiguation in Clinical Discourse. *Proceedings of the Annual Symposium on Biomedical and Health Infomatics (AIMA)*, 2005:589-593.

Pakhomov S. 2002. Semi-Supervised Maximum Entropy Based Approach to Acronym and Abbreviation Normalization in Medical Texts. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL)*, pp. 160-167.

Patrick J. and M. Sabbagh. 2011. An Active Learning Process for Extraction and Standardization of Medical Measurements by a Trainable FSA. *In Tokyo-Japan: Springer Berlin / Heidelberg*, pp. 151-162.

Patrick J., M. Sabbagh, S. Jain and H. Zheng. 2010. Spelling correction in Clinical Notes with Emphasis on First Suggestion Accuracy. *2nd Workshop on Building and Evaluating Resources for Biomedical Text Mining*, pp. 2-8.

Patrick J., Y. Wang and P. Budd. 2007. An automated system for conversion of clinical notes into SNOMED clinical terminology. *Proceedings of the 5th Australasian Symposium on Australasian Computer Science Week (ACSW) frontiers*, 68:219-226.

Philips, L. 1990. Hanging on the metaphone. *Computer Language Magazine*, 7(12), 38–44.

Roche E. and Y. Schabes. 1995. Deterministic part-of-speech tagging with finite-state transducers. *Journal of Computational Linguistics*, 21(2), 227-253.

Ruch, P., R. Bauda and A. Geissbuhlera. 2003. Using lexical disambiguation and named-entity recognition to improve spelling correction in the electronic patient record. *Artificial Intelligence in Medicine*, 29(2003), 169–184.

Senger C., J. Kaltschmidt, S.P.W. Schmitt, M.G. Pruszydlo and W.E. Haefeli. 2010. Misspellings in drug information system queries: Characteristics of drug name spelling errors and strategies for their prevention. *International Journal of Medical Informatics*, 79(2010), 832–839.

Turchin, A., J.T. Chu, M. Shubina and J.S. Einbinder. 2007. Identification of misspelled words without a comprehensive dictionary using prevalence analysis. *Proceedings of the Annual Symposium on Biomedical and Health Infomatics (AIMA)*, 11:751-755.