Artificial Intelligence in Medicine 53 (2011) 171–180

Contents lists available at ScienceDirect

Artificial Intelligence in Medicine

journal homepage: www.elsevier.com/locate/aiim

Statistical semantic and clinician confidence analysis for correcting abbreviations and spelling errors in clinical progress notes

Wilson Wong[a,b,*], David Glance[a]

[a] Centre for Software Practice, University of Western Australia, 35 Stirling Highway, Crawley, WA 6009, Australia
[b] School of Computer Science and Information Technology, RMIT University, 414–418 Swanston Street, Melbourne, VIC 3000, Australia

ARTICLE INFO

Article history:
Received 17 May 2010
Received in revised form 20 July 2011
Accepted 9 August 2011

Keywords:
Spelling error correction
Abbreviation expansion
Syntagmatic and paradigmatic similarity
Progress note cleaning

ABSTRACT

Motivation: Progress notes are narrative summaries about the status of patients during the course of treatment or care. Time and efficiency pressures have ensured clinicians' continued preference for unstructured text over entering data in forms when composing progress notes. The ability to extract meaningful data from the unstructured text contained within the notes is invaluable for retrospective analysis and decision support. The automatic extraction of data from unstructured notes, however, has been largely prevented due to the complexity of handling abbreviations, misspelling, punctuation errors and other types of noise.
Objective: We present a robust system for cleaning noisy progress notes in real-time, with a focus on abbreviations and misspellings.
Methods: The system uses statistical semantic analysis based on Web data and the occasional participation of clinicians to automatically replace abbreviations with the actual senses and misspellings with the correct words.
Results: An accuracy of as high as 88.73% was achieved based only on statistical semantic analysis using Web data. The response time of the system with the caching mechanism enabled is 1.5–2 s per word which is about the same as the average typing speed of clinicians.
Conclusions: The overall accuracy and the response time of the system will improve with time, especially when the confidence mechanism is activated through clinicians' interactions with the system. This system will be implemented in a clinical information system to drive interactive decision support and analysis functions leading to improved patient care and outcomes.

1. Introduction

Clinical records consist of a collection of clinical observations, treatment histories and medical test results produced at different stages of patients' health care. Elements of clinical records include admission notes, progress notes, radiology reports, pathology reports and discharge summaries. Despite containing some coded information, most of the important details in clinical records remain locked away in unstructured text [1]. For this reason, the automatic analysis of clinical records has become an important research area with significance to a variety of systems for reducing medical errors and improving other aspects of health care such as disease surveillance [2].

Our research focuses specifically on the automated analysis of clinical progress notes to provide clinicians with intelligent functions to support their everyday decision making. The urge amongst clinicians for faster text entry while attempting to retain semantic clarity has, however, contributed to the *noisy* structure of progress notes [3]. A progress note is considered as containing noise when there is potential difference or ambiguity between the surface form of the entered text and the intended content. For instance, a clinician could mistakenly enter "*pb*" instead of the intended "*bp*" or "*blood pressure*". Similarly, the preferential use of acronyms such as "*arf*" over "*acute renal failure*", which could potentially lead to ambiguous interpretations of notes, is another source of noise. The more noise clinicians introduce in their progress notes, the less intelligible the notes will become. Some of the common types of noise are *abbreviation*, *misspelling* and *punctuation error*. From here onwards, abbreviations will be taken to simply mean shortened forms of words (whether common or ad hoc), and can comprise acronyms, initialisms and so on.

A quick look at the literature reveals that noise remains a major hurdle towards achieving accurate and robust progress note analysis for real-world applications. In the words of Friedlin et al. [4], "*A chief barrier to accurate NLP classification was that many free text results disregarded grammatical rules. These reports*

* Corresponding author. Current address: School of Computer Science and Information Technology, RMIT University, 414–418 Swanston Street, Melbourne, VIC 3000, Australia. Tel.: +61 3 9925 9793; fax: +61 3 9662 1617.
E-mail address: wilson.wong@rmit.edu.au (W. Wong).

0933-3657/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.artmed.2011.08.003

```
NERUO: PT ALERT AND ORIENATED X3. mae and able
to follow commands cv: pt remains nsr, no ectopy
noted. hr 70-80's. sbp 120-100's. map 60's. NEO
GTT WEANED TO OFF AT 4 AM. CI 4.98-4.01.
```

Fig. 1. A noisy progress note obtained from http://physionet.org (last accessed 6 May 2011).

frequently contained incomplete sentences, spelling errors, and/or lacked proper punctuation". A missing punctuation mark or a few too many abbreviations can often render standard natural language processing tools such as part-of-speech taggers unusable. To date, most research into progress note analysis remains experimental, and deals with the *cleaning phase* (also known as *normalising*) by either ignoring it all together [4], requiring the input data to be cleaned beforehand [5], or using *machine learning* techniques that are expensive to set up and impractical to deploy. The challenge of achieving complete and accurate progress note analysis, therefore, begins with a robust, practical approach dedicated to cleaning noisy input. The focus of this paper is on addressing this challenge.

Recently, it was suggested that the problem of abbreviation disambiguation is simpler than that of general term disambiguation [6]. A quick look at an actual progress note in Fig. 1 would immediately suggest otherwise. While expanding abbreviations may be relatively straightforward with respect to the biomedical literature [6,7] or other properly curated texts [8], disambiguating abbreviations in clinical progress notes is definitely non-trivial. A number of problems contributed to the difficulty of achieving real-time performance in progress note analysis amongst existing systems. This paper focuses on the following: (1) nonconformity of the structure of progress notes to conventions, (2) noisy contextual information derived from progress notes, (3) inability of static dictionaries to easily reflect changes in the domain, and (4) problems with machine learning techniques. We will discuss more on these points and other related work in Section 2.

This paper addresses the four problems described above in the form of an *integrated clinical progress note cleaning system* based on *statistical semantics* and a novel *confidence mechanism*. The four main distinguishing points that separate the proposed system from existing ones are:

- The system automatically corrects both abbreviations and misspellings.
- The system is easy to set up without the trouble of training using hand-crafted data, or the complexity of preparing a variety of features (e.g. from static knowledge sources, part-of-speech tagging, etc.) for classification.
- The system uses statistical information derived from the distributional behaviour of words (i.e. statistical semantics) on the World Wide Web (or the Web from here on) to generate a set of scores to automatically produce the preliminary corrections.
- The system memorizes its interactions with clinicians over time to establish a confidence mechanism which is used to adjust the scores for improving future automatic corrections.

In short, the system provides realistic, real-time operation in the face of noisy progress notes with the ability to incrementally improve its performance through repeated interactions with clinicians. Section 3 describes the detail of this system. The system's sources of long forms or *senses* from here on, and the possible replacements for misspellings called *suggestions*, are discussed in Section 3.1. We evaluated the system's performance in two parts. First, we look at the response time of the system during real-time operations. Second, we assess the accuracy of the initial automatic corrections without the confidence mechanism. The results that arose from the evaluation are presented in Section 4. Section 5 discusses the limitations of the proposed system. We conclude this paper in Section 6.

2. Related work

Unlike properly curated texts, the noisy nature of clinical progress notes and the importance of complete and accurate analysis place extra burden on the cleaning phase, and hence, the significance of this research. A look into the literature immediately reveals the many problems that contribute to the challenging nature of this task. We first take a look at some of the existing work on disambiguating abbreviations and correcting spelling errors in Section 2.2. We then discuss in detail in Section 2.3 the four main problems that have to be addressed in order to create a robust cleaning system of reasonable performance.

2.1. Existing work in spelling error detection

The two main types of approach for generating suggestions for spelling errors are *minimum edit distance* and *similarity key*. The idea behind the edit distance technique was introduced by Damerau [9] and Levensthein [10]. Damerau–Levenshtein distance is the minimal number of insertions, deletions, substitutions and transpositions required to transform one string into another. For example, to change "*wear*" to "*beard*" will require a minimum of two operations, namely, a substitution of '*w*' with '*b*', and an insertion of '*d*'. Many variants were developed subsequently such as the algorithm by Wagner and Fischer [11]. The different techniques vary in the way each operation is assigned a cost. The second type of technique is similarity key. The main idea behind similarity key techniques is to map every string into a key such that similarly spelt strings will have identical keys [12]. Hence, the key, computed for each spelling error, will act as a pointer to all similarly spelt words (i.e. suggestions) in the dictionary. One of the earliest implementations is the Soundex system by Odell and Russell [13,14]. Soundex maps a word into a key consisting of its first letter followed by a sequence of numbers. Since then, many improved variants were developed such as the Metaphone and the Double-Metaphone algorithms by Philips [15], the Daitch-Mokotoff Soundex [16] and others [17].

2.2. Existing work in abbreviation disambiguation

Existing work on cleaning clinical progress notes has been mainly confined to the task of abbreviation disambiguation. Other types of noise such as spelling errors and improper letter casings are mostly disregarded, or dealt with using off-the-shelf tools. For instance, noisy laboratory messages and the absence of cleaning resulted in a large number of misses in the classification experiment by Friedlin et al. [4] using a tool called Regenstrief EXtraction (REX). Similarly, Gupta et al. [18] reported inaccuracies and even failures in their de-identification (De-Id) program when there were spelling errors or abbreviations. In the case of Wang [5], an off-the-shelf spelling corrector trained on a 60-million token corpus was used to clean its input text prior to extracting clinical named entities. Wang [5] manually disambiguated the abbreviations in the corpus using external knowledge sources, namely, UMLS, SNOMED-CT and MOBY. The reasonable performance of the author's entity extraction task was achieved because of the manual cleaning involved, and in the words of the author, "*The proofreading required considerable amount of human effort*...". In other words, existing work has

come to terms with the fact that noise is an inevitable part of clinical text analysis, and to date, no integrated systems dedicated to cleaning noisy progress notes can be found in the literature.

In regard to the mainstream focus on resolving abbreviations, research activities range from (1) extracting abbreviations and senses for constructing dictionaries to (2) creating techniques for disambiguation. Some examples of inventory construction include the technique by Schwartz and Hearst [19] which identifies abbreviations and their senses from Medline abstracts using surface patterns such as `long form (abbreviation)`. In an another example, Chang et al. [20] proposed a technique for building a biomedical abbreviation dictionary. The technique comprises four steps, namely, finding abbreviation candidates using the patterns as above, aligning the candidates to the long forms, converting the abbreviations and alignments into feature vectors, and scoring the feature vectors using a binary logistic regression classifier. The authors had to manually identify abbreviations and hand-annotate the alignments to create the training set. Existing techniques for abbreviation disambiguation include:

- Yu et al. [7] uses Support Vector Machine (SVM) to disambiguate abbreviation. To overcome the problem of preparing training and testing data, the authors created a technique to automatically extract the data by searching for the long and short forms from text. As we have mentioned earlier, this may be applicable to properly curated text such as biomedical literature, but not to clinical notes.
- Stevenson et al. [6] applied a word-sense disambiguation approach based on multiple knowledge sources for disambiguating biomedical abbreviations. The approach is based on supervised learning with a wide range of structural, linguistic and semantic features including bi-gram and tri-gram collocations, part-of-speech tags, unigram co-occurrence, and concept identifiers obtained from structured resources such as UMLS and MeSH. The authors compared the performance of the approach using three learning algorithms and a set of 21 three-letter abbreviations. Vector space model achieved the best accuracy of up to 99% as compared to SVM and Naive Bayes.
- Pakhomov [21,22] compared the use of the Web with Medline abstracts and a Mayo Clinic corpus (1.7 million notes) for gathering contextual information required for disambiguating abbreviations in clinical notes. The authors too concluded that methods based on the vector space model are more effective compared to classification techniques such as maximum entropy. The authors also found that the Web offers adequate context to achieve encouraging abbreviation disambiguation performance, which is better than the use of Medline abstracts and the Mayo Clinic corpus.
- Xu et al. [2] evaluated four methods for detecting abbreviations, and assessed the abbreviation and sense coverage of two resources. Ten clinical notes from the New York Presbyterian Hospital were selected and annotated for evaluation. Six were used for training with the remaining four as test data. The first method, which acts as a baseline, simply selects all unknown tokens in the text as abbreviations. The second method is based on heuristics concerning capital letters, the combination of numeric and alphabetic characters, punctuation marks and so on. The third and fourth methods use a decision tree classifier. It was found that the method based on a decision tree classifier with additional external knowledge achieved the best precision. The best recall, however, was achieved using heuristics. An interesting point to note is that abbreviations in lower case were not detected due to an inadequate training set.

2.3. Main challenges in clinical progress note cleaning

The four main problems that have continued to prevent existing clinical text cleaning systems from delivering reasonable, real-time performance are as follows:

- Nonconformity of the structure of progress notes to conventions: Common heuristics based on surface patterns to detect abbreviations or other noise are not straightforwardly applicable to noisy progress notes. Actual progress notes rarely conform to the conventions of abbreviation, letter casing, punctuation, and grammar. For instance, existing detection methods based on pattern matching are able to easily identify "*BSA*" in the clean text "*Lean BSA was obtained from height and lean body weight*. . ." [6] as an abbreviation. The same methods, however, would not work with the noisy sample shown in Fig. 1. The presence of misspellings and improper letter casings in the noisy sample complicates the disambiguation process. How do we know that unlike "*able*" and "*NO*", "*mae*" and "*CI*" are abbreviations based solely on surface patterns?
- Noisy contextual information derived from progress notes: The indiscriminate use of context for abbreviation or spelling error correction, as in the work of Ruch et al. [23], may not necessarily yield the same performance with noisy progress notes. Local (i.e. words surrounding the abbreviation) and global contexts (i.e. topic of the text or type of discourse in which the abbreviation occurs) are indispensable for cleaning noisy text. For instance, the abbreviation "*RA*" has over 20 senses [22] in UMLS. Using local context, "*right atrial*" would be a more probable long form compared to "*rheumatoid arthritis*" if that particular instance of "*RA*" appears with "*cardiovascular*" and "*ventricular ectopic activity*" in the text. However, how can we effectively employ context words for correcting misspellings or abbreviations if more than 60% of the words in the text are noise (e.g. abbreviation, misspelling). Consider "*RA*" being surrounded by "*vea*" and "*cv*" instead of "*ventricular ectopic activity*" and "*cardiovascular*".
- Inability of static dictionaries to easily reflect changes in the domain: Dictionaries of abbreviations and their senses are valuable resources for cleaning progress notes. This is evident by the intense research effort devoted to constructing more dictionaries using abbreviation–definition patterns [19], clustering [24], statistical techniques [25] and so on. However, with the increasing number of new senses and abbreviations (both standard and ad hoc) coming into use, it is becoming apparent that the sole reliance on individual abbreviation dictionaries is inadequate. For instance, UMLS only covered about 35% of the senses of abbreviations in hospital admission notes at the New York Presbyterian Hospital [2]. In the words of Pakhomov et al. [22], "*. . .established sources of acronyms and abbreviations may not be entirely suitable as sources of sense dictionaries for acronyms in clinical notes*".
- Problems with machine learning techniques: Most work conducted to date for expanding biomedical or clinical abbreviations remains highly experimental with a focus on offline processing using machine learning techniques. Some of the more pressing problems with these techniques are:
  - Supervised learning techniques require large amount of training examples that are representative of the noise at hand. Researchers working on supervised techniques face difficulties in preparing multiple manually labelled examples of each sense of an abbreviation in context for training. The issues of patient confidentiality and the diversity of noise in actual progress notes complicate the situation. Inadequate training data produce suboptimal classification as in the case of Xu et al. [2].
  - To address the problem of inadequate training examples, two common approaches based on abbreviation-definition patterns such as `long form (abbreviation)`, and the notion

of *paradigmatic substitutability* were proposed for (semi-)automatically gathering larger datasets. The first approach assumes that abbreviations and their definitions co-occur together within a reasonable window size [2]. For instance, Yu et al. [7] extracts training data by searching for short/long forms in Medline abstracts. The second approach assumes that abbreviations and their definitions are substitutable with one another in similar contexts. Pakhomov [21,22] gathers training data from Web data and other sources based on this approach. These approaches, however, are based on assumptions that rarely hold in the face of noisy clinical text. As an example, the noisy nature of contextual information in progress notes, as mentioned in point (2), introduces inaccuracies to the second approach.

- Many supervised and unsupervised learning techniques are looking to diversify the features used to compensate for inadequate training. However, many of these features are difficult to come by. For instance, Stevenson et al. [6] introduced a supervised learning system that makes use of a variety of features ranging from part-of-speech tags and syntactic features to semantic information from external knowledge sources. This approach does not work simply because noisy notes do not follow grammar conventions and parsing them using standard part-of-speech taggers will not yield the desired results. Using semantic information from knowledge sources as features too is not the best way because of their poor coverage as discussed in point (3).

3. Real-time clinical progress note cleaning

In essence, our system takes as input a noisy progress note, and produces the corresponding cleaned text as output. The system cleans a progress note in six steps. First, the system splits the input note by whitespace into a set of n words $W = \{w_1, \ldots, w_n\}$ (line 2 in Algorithm 1). For instance, the sentence "*Pt continues itnubated and on ventilatory support*" is broken down into $W = \{$"*Pt*", "*continues*", "*itnubated*", "*and*", "*on*", "*ventilatory*", "*support*"$\}$. Second, the system identifies words $w_i \in W$ that can be cleaned (line 6 of Algorithm 1). We will simply refer to the i-th word in the progress note, w_i as w from here on. Third, each w that requires cleaning is provided with a set of context words C_w whose composition is determined by a window size. The set C_w comprises the v number of words to the left as well as to the right of w (lines 7–22 of Algorithm 1). Continuing from our example above, assuming $v = 2$, the C_w for $w =$ "*on*" is $\{$"*itnubated*", "*and*", "*ventilatory*", "*support*"$\}$. Fourth, for each w that requires cleaning, a set of senses and suggestions S_w is created (lines 23–32 in Algorithm 1). The steps taken for sourcing senses and suggestions are provided in Section 3.1. Fifth, four types of scores based on statistical semantics and a confidence mechanism are calculated for each $s \in S_w$ (lines 35–39 in Algorithm 1). The computation of these scores is described in Section 3.2. Last, the sense or suggestion with the best combined score is taken as the ideal replacement for the abbreviation or spelling error (lines 40–43 in Algorithm 1).

Algorithm 1. Progress Note Cleaning (*input*, v)

```
1:  initialise output to input where input is simply the textual content of
    the progress note to be cleaned in the form of a string and v defines the
    number of words to the left and right of a current word to be
    considered as context.
2:  split input into set W.
3:  for each w_i ∈ W do
4:    initialise C_w and S_w to Ø
5:    initialise S_best to empty string, and ρ_best to 0.
6:    if w_i is not a number, a range of numbers, or a single-character word
    then
7:      l := w_{i−1}
8:      r := w_{i+1}
9:      lborder := 0
10:     if i − v > 0 then
11:       lborder := i − v
12:     end if
13:     rborder := |W|
14:     if i + v < |W| then
15:       rborder := i + v
16:     end if
17:     for j := i − 1; j ≥ lborder; j − − do
18:       C_w := C_w ∪ {w_j}//construct the set of context words
19:     end for
20:     for j := i + 1; j < rborder; j + + do
21:       C_w := C_w ∪ {w_j}
22:     end for
23:     B_w := senseAbbreviation(w)//construct the set of abbreviation
    senses
24:     S_w := B_w
25:     isword := isValidWord(w)
26:     if isword = FALSE ∧ |S_w| = 0 then
27:       A_w := suggestAspell(w)//construct the set of spelling suggestions
    using Aspell!
28:       Y_w := suggestYahoo(l,w,r)//construct the set of spelling
    suggestions using Yahoo!
29:       S_w := A_w ∪ Y_w
30:     else if isword = TRUE then
31:       S_w := S_w ∪ {w}
32:     end if
33:     if |S_w| > 0 then
34:       for each s ∈ S_w do
35:         calculate δ_s based on Eq. (2), χ_s using C_w and Eq. (3), π_s using l,
    r and Eq. (5), and σ_s using C_w and Eq. (7)
36:         ρ_s := δ_s χ_s π_s σ_s
37:         if ρ_s > ρ_best then
38:           ρ_best := ρ_s and S_best := s
39:         end if
40:       end for
41:     else
42:       S_best := w
43:     end if
44:   else
45:     s_best := w
46:   end if
47:   replace w in output with s_best
48: end for
49: return output
```

3.1. Alternative sources of senses and suggestions

Abbreviations evolve in many ways: more and more phrases may be abbreviated into the same short form, the same phrase may be referred to by increasingly different abbreviations, and entirely new abbreviations are introduced from time to time to accommodate emerging words. Intuitively, one would ask how could any dictionary be able to capture and reflect such rapid changes. This concern is especially valid if the dictionaries are manually created and maintained by a small group of individuals at non-negligible costs. Xu et al. [2] reported that UMLS only covered about 35% of senses of abbreviations in hospital admission notes at the New York Presbyterian Hospital. Besides being slow at reflecting changes and expensive to maintain, current established sources such as the UMLS provide coverage of restricted domains (e.g. biomedicine) and hence, certain generic senses that are required for correcting the diverse noise in clinical notes may not be present. As such, combining the strengths of more generic and collaboratively maintained resources on the Web with the specificity of hand-crafted vocabularies such as UMLS may be the direction to follow. This, however, is not the focus of our current work.

In this paper, we focus on harnessing (1) the collective efforts of abbreviation dictionary maintenance on the Web, and (2) the suggestions provided by clinicians during interactions with the system. There are several providers that maintain dictionaries of clinical abbreviations and their senses on the Web. The services mentioned below are free to use, and are

updated quite frequently by taking into account suggestions provided by users on the Web. They are http://www.all-acronyms.com/cat/7, http://www.medilexicon.com and http://www.abbreviations.com/bmc.aspx?c=MEDICAL. These and all subsequent URLs were last accessed on 6 May 2011.

At the moment, the proposed system uses the API for accessing abbreviations provided by abbreviations.com. The developer key can be obtained using the form http://www.abbreviations.com/abbr_api.asp. With the keys, developers are allowed up to 1000 queries per day using the request URL http://www.abbreviations.com/services/v1/abbr.aspx. A sample request URL is http://www.abbreviations.com/services/v1/abbr.aspx?tokenid=tk324324324&term=vea. There are over 15,000 abbreviations in abbreviations.com as of April 2011 that span across six sub-domains, namely, hospitals, laboratory, physiology, genome, oncology and veterinary. As for the replacements of spelling errors, the suggestions provided by Yahoo! and the standalone tool Aspell [26] are used. The Yahoo! spelling suggestion service is limited to 5000 queries per day using the URL http://search.yahooapis.com/WebSearchService/V1/spellingSuggestion. A sample query is http://search.yahooapis.com/WebSearchService/V1/spellingSuggestion?appid=YahooDemo&query=madnna. As for Aspell, the latest version 0.50.3 can be downloaded from http://aspell.net/win32/. Aspell uses only the Metaphone algorithm [15] and near-miss strategy by its predecessor Ispell [27] to generate suggestions for misspellings. Yahoo! spelling suggestion, on the other hand, is able to take into consideration contextual information to improve its suggestions. The use of suggestions from both Aspell and Yahoo! brings together the demonstrated performance of the former for generic text and the dynamic and broader coverage of the latter. An online interface to query Aspell is also available at http://suggest.aspell.net. It is worth noting that the proposed system is not in anyway tied down to a particular online service. We can always easily integrate new services in the future that provide better coverage.

Access to `abbreviations.com` and Aspell is implemented as the functions `senseAbbreviation` and `suggestAspell` which are queried using only the word w (lines 23 and 27 in Algorithm 1). The set of senses provided by `abbreviations.com` is denoted as B_w, while A_w represents the suggestions for a misspelling by Aspell. As for obtaining spelling suggestions from Yahoo!, the function `suggestYahoo` is called using the left and right context words l and r in addition to w (line 28 in Algorithm 1). The set Y_w is returned as the result. All senses and suggestions in B_w, Y_w and A_w are combined, in no particular order, to form a single set of potential replacements for both spelling errors and abbreviations S_w. If w is a valid English word, it will also be added to the set S_w for consideration (lines 30 and 31 in Algorithm 1).

3.2. Integrated scoring for sense disambiguation and misspelling correction

The next step after obtaining C_w and S_w is to score and rank each $s \in S_w$ using a *replacement score* denoted as ρ_s:

$$\rho_s = \delta_s \chi_s \pi_s \sigma_s \quad (1)$$

The individual scores that contribute to ρ are in effect the results of the various confidence and statistical semantic analyses. These scores are in place for systematically reducing the number of possible replacements by considering the various structural and semantic aspects of a noise word.

First, the *dominance score* (δ_s) reflects the *syntagmatic similarity* between the noise w (i.e. abbreviation, misspelling) and the replacement s (i.e. sense, suggestion). In simple terms, two words are syntagmatically similar if they co-occur significantly together within a certain predefined unit (e.g. n-gram, sentence-level, paragraph-level, document-level). In this sense, words that are syntagmatically similar such as "*knife*" and "*meat*" can be said as 'related' and hence, the notion of "*semantic relatedness*".[1] The score is computed as:

$$\delta_s = NWD(s, w) \quad (2)$$

where $NWD(s,w)$, which stands for Normalised Web Distance by Vitanyi et al. [28], is a measure for semantic relatedness based on search engine page count. In this work, δ_s quantifies which amongst the $s \in S_w$ is more commonly observed co-occurring together with w within the same text. This score is intuitively similar but extends the commonly used technique for identifying abbreviations which assumes that long forms precede their senses enclosed in parentheses (e.g. "*methyl methanesulfonate sulfate (MMS)*"). Instead of relying on very restrictive patterns, this score is based on the co-occurrence of abbreviations and their senses within the same document regardless of their adjacency (i.e. document-length size window). More importantly, clinicians compose progress notes with the assumption that the readers will have the necessary background knowledge to decode any abbreviations. As such, abbreviations rarely or perhaps never co-occur with their senses in noisy progress notes. For this reason, the co-occurrence values required to compute the score are obtained from Web data in the form of NWD instead of clinical corpora. Overall, this score provides a simple, initial estimate of the relatedness between s and w without considering any context. Contextual information will play a very important role in determining the subsequent three scores.

Second, clinicians interact with the proposed system through three main activities, namely, entering progress notes, making changes to the automatic corrections, and saving the cleaned notes. A vast amount of knowledge can be voluntarily acquired from clinicians with time in such an interactive system environment. We introduced a confidence mechanism to capture this knowledge and translate it into a *confidence score* (χ_s) to improve the automatic corrections. The score χ_s for each sense s reflects the clinicians' confidence, denoted as $p_s \in \{0, 1, 2, \ldots\}$, with regard to the sense's suitability as a replacement for noise w given its context C_w. During the automatic correction of abbreviations and misspellings not encountered before by the system, the χ scores for the corresponding senses are set to 0.5 due to $p = 5$. The p and hence, the χ score will change through clinicians' participations in three different occasions:

- The clinician can alter an incorrect automatic correction by selecting from a provided list based on the sets B_w, Y_w and A_w, or introduce a new sense or suggestion. The p_s value of the sense s selected or entered by the clinician will be incrementally increased. Each sense s and its confidence p_s are recorded in a *clinician confidence database* together with the corresponding abbreviation w and the context in which w exists, C_w.
- The clinician can reinforce a correct automatic correction. This will help to ensure that the corresponding abbreviation or misspelling is corrected using that same sense or suggestion given the same context in future progress note entry. The reinforcement process will result in s being assigned a higher p_s.
- The third way through which the clinician's participation will effect the χ score is the saving of the cleaned progress note. The saving of a cleaned progress note by a clinician will result in a similar reinforcement process as case (2) whereby all senses and suggestions replacing the abbreviations and misspellings,

[1] The notions of semantic relatedness and semantic similarity, which are fundamentally different, are often inappropriately used interchangeably.

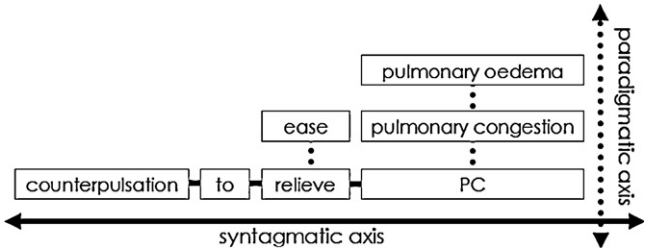


Fig. 2. The principle of syntagmatic and paradigmatic similarity upon which our two proposed scores π_s and σ_s are based. The syntagmatic similarity between two words depends on the extent to which one selects the other to precede or follow it (i.e. surface structure or syntax). Paradigmatic similarity, on the other hand, relies on the substitutability of a word with others of the same type or class (i.e. paradigm) to determine association.

whether automatically decided or manually corrected, will attain higher p_s.

Hence, the computation of the confidence score is based on both the confidence assigned by clinicians (i.e. p_s), and the overlapping of the contexts of w which is C_w and of its match w' which is $C_{w'}$ in the database. The score χ_s is defined as:

$$\chi_s = \begin{cases} \Omega_s \exp\left(\frac{-1}{p_s}\right) + \alpha & \text{if}(\Omega_s > 0 \wedge p_s \geq 1) \\ 0.5 & \text{otherwise} \end{cases} \quad (3)$$

where $\alpha = 1 - \exp(-1)$. The χ_s value is set to 0.5 if there is no contextual overlap (i.e. $\Omega_s = 0$) or no confidence has been assigned by clinicians (i.e. $p_s = 0$). Otherwise, α will ensure that χ_s exceeds $1 - \exp(-1) = 0.6321$. The *contextual overlap*, denoted by Ω_s, is defined as:

$$\Omega_s = \begin{cases} \exp - \left(\frac{|C_w|}{|C_w \cap C_{w'}|}\right) & \text{if}(C_w \cap C_{w'} \neq \varnothing) \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

and $0 \leq \Omega_s \leq \exp(-1)$. As the contextual overlap increases, Ω_s approaches $\exp(-1) = 0.3679$. Overall, this score is important in that (1) it allows the system to learn from clinicians and improve its automatic corrections, and (2) it acknowledges the fact that providing real-time cleaning with 100% accurracy consistently in real-word settings is extremely difficult without the involvement of clinicians.

Third, two words that are *paradigmatically similar* may be substituted for one another in the same context without violating the semantic well-formedness of the sentence. The words "*knife*" and "*blade*", for instance, are interchangeable in the context of "*cut with* *". In this sense, we can say that "*knife*" and "*blade*" are similar and hence, the notion of "*semantic similarity*". We introduce the *paradigmatic similarity score* (π_s) to capture this principle. Referring to Fig. 2, the words "*relieve*" and "*ease*" are paradigmatically similar in the context of "*counterpulsation to* * *pulmonary congestion*". The score π_s reflects the extent to which sense s is able to substitute the abbreviation w within its immediate contexts, which are the words to the left $l = w_{i-1}$ and the right $r = w_{i+1}$ of w. The higher the value of π_s, the more frequently s can be observed in between l and r. The score π_s is defined as:

$$\pi_s = \exp\left(\frac{\beta_s}{10}\right) \quad (5)$$

where the heuristically derived *substitutability score* β_s is:

$$\beta_s = \log \frac{0.8 n_{lsr} + 0.1(n_{ls} - n_{lsr}) + 0.1(n_{sr} - n_{lsr}) + 1}{n_w + 1} \quad (6)$$

where n_{lsr}, n_{ls} and n_{lsr} are the page counts for the trigram "*lsr*", the bigrams "*ls*" and "*sr*", respectively. n_w is the page count for the unigram w. To illustrate the purpose of the π_s score, consider the text "*intra-aortic balloon counterpulsation to relieve PC*". The abbreviation "*PC*" has over 10 senses such as "*present complaint*", "*prenatal care*", "*prostatic carcinoma*" and "*pulmonary congestion*". In this example, the use of the paradigmatic similarity principle reduces the number of replacements in S_w to only those that can take on the left context "*relieve*" such as "*pulmonary congestion*" and "*present complaint*". The list of possible replacements can be greatly reduced through this method.

Fourth, words that are syntagmatically similar occur together (i.e. co-occur) in the same context more often than usual. For instance, assume that "*kn*" stands for either "*knife*" or "*knob*". If "*kn*" appears in the same sentence together with "*cut*", we can immediately conclude that "*kn*" refers to "*knife*", regardless of their positions. In this case, "*knife*" and "*cut*" are syntagmatically similar. A *syntagmatic similarity score* (σ_s) is introduced to capture this principle. The σ_s score measures the extent of sense s co-occurring with the wider context of w in the same text. The more frequent s co-occurs with the context words in C_w, the more syntagmatically similar they are. This translates to s being a more suitable replacement for w, and hence higher σ_s score. This score, which is based on group-average relatedness, is defined as:

$$\sigma_s = \frac{\sum_{c \in C_w, c \neq l, c \neq r} NWD(s, c)}{|C_w|} \quad (7)$$

The use of the previous score π_s helps to reduce the number of replacements in set S_w to only those that are substitutable for w (i.e. paradigmatically similar). This last score, on the other hand, helps the system to further focus in on a single best replacement by considering contextual information (i.e. the syntagmatic axis of Fig. 2). Based on the same example phrase "*intra-aortic balloon counterpulsation to relieve PC*", the high co-occurrence of s = "*pulmonary congestion*" with the non-immediate context words such as "*intra-aortic*", "*balloon*" and "*counterpulsation*" will contribute to its higher σ_s score.

4. Evaluation results

For the initial evaluation, we randomly selected a test set of 30 samples from a corpus of 2433 actual de-identified progress notes by http://physionet.org.[2] The test set comprises 961 words, with each note containing an average of 32 words. Fig. 3 shows 4 of the 30 progress notes used in this evaluation. We performed the test in four steps. First, we typed the 30 progress notes word by word into the system's interface as shown in Fig. 4. The operation of the system as each word is being entered is as described in Algorithm 1. Second, we recorded the response time and the cleaned notes. Third, we allowed our domain expert to assess the automatically cleaned notes. Fourth, we constructed confusion matrices comparing the predicted outcomes and the actual values based on the domain expert's feedback. For this experiment, a caching mechanism is in place in the system to improve access to senses and suggestions from the Web.

In the next two sections, we look at the performance of the new system in terms of response time and accuracy.

4.1. System response time

In the first part of the evaluation, we look at the performance of the system in terms of its *response time*. Response time, in this evaluation, is defined as the interval between the instant at which

[2] A corpus of clinical progress notes can be downloaded from http://physionet.org/cgi-bin/receive?file=id.text.

```
CV: VSS, PA's 30's-50's/20's-30's. CVP 7-12.
Remains in AF. No VEA. KCL 40 MEQ GIVEN AT 1400.
```

```
HR 70 paced. BP 104-115/50's. Remains on neo at
320mcq. PAP 60's/27-30, wedge 28, CVP 27-29.
```

```
Neuro: Pt alert and orienated x3. Able to MAE
and follow commands. Pt difficult to understand
at times d/t history of stroke CV: Pt remains in
CHB.
```

```
CV: NSR, very frequent PACs, occassional PVCs.
Afebrile. Atiral pacing wires attached to pacer,
pacer turned off. Ventricular pacing wires
protected and secured to chest. PO lopressor
increased, diovan added. On and off Ntg gtt for
HTN. US of carotids done.
```

Fig. 3. Four of the 30 noisy progress notes from http://physionet.org used in our evaluation.

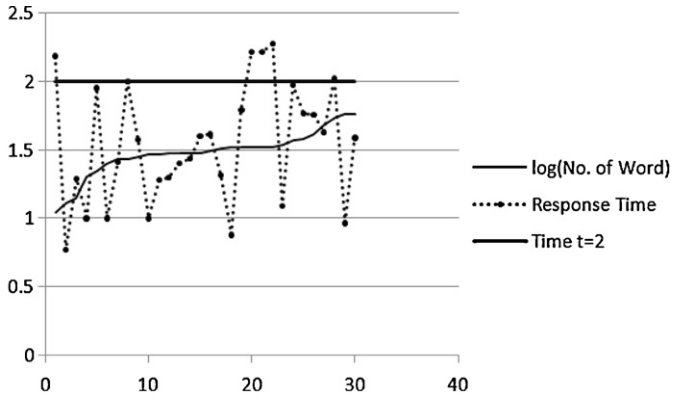
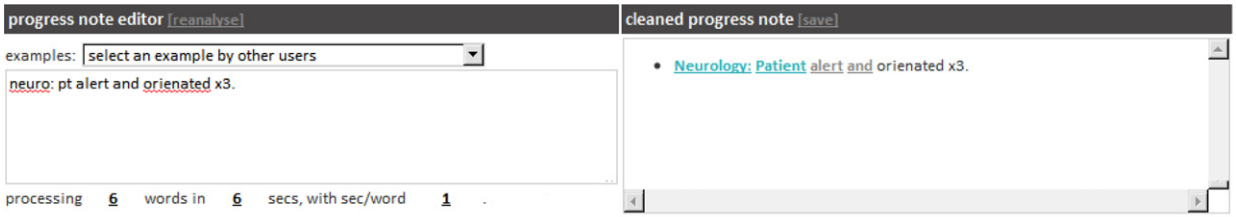


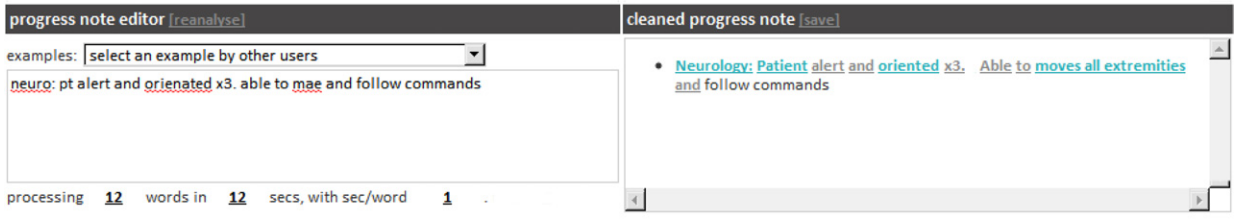
Fig. 5. The response time of the system and its relation with the number of words in the notes.

the first word is entered and the instant at which the final word of the response is received by the end user. The results showed that a total of 1499 s or 24.98 min is required to clean all 961 words in the 30 notes. In other words, the average response time of cleaning a note is 49.97 s. More precisely, an average of 1.56 s is required to process one word, with a standard deviation of 0.44.

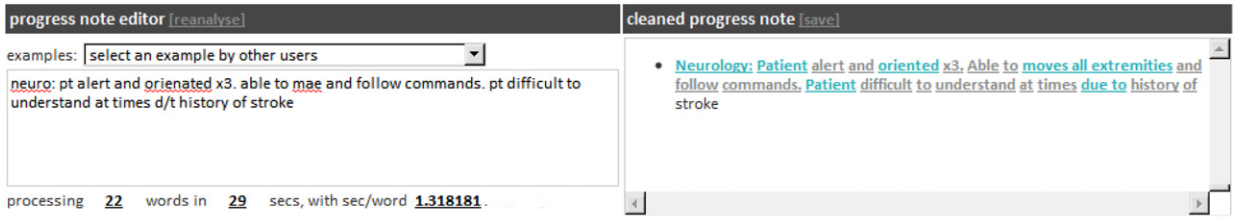
An interesting point to note is that the average typing speed for clinicians is around 30 words per minute [29], or equivalently, 2 s per word. As mentioned before, the average response time per word of the new system is about 1.56 s. The worst response time in this evaluation is 2.27. As shown in Fig. 5, only 5 notes are located above $t = 2$ (the thick solid line). In other words, more than 83% (=25/30) of the notes have a response time of less than 2 s. Intuitively, by the time a clinician finishes typing one word, the previous one will likely be cleaned already. This effect, however, will not be so obvious in the actual system since the response is buffered and displayed in groups of words. Fig. 5 also shows that the number of words in a note does not correspond linearly to the response time. As the number of words increases (the thin solid line), we do not see a corresponding rise in the response time (the dotted line). Instead, the response time was erratic, and it increases and drops irrespective of the number of words in the corresponding notes. One explanation is that while processing a note, the system encounters a varying number of replacements in set S_w for each word to be corrected. While a note may have a large number of words to be corrected, each of them may only have 2–3 replacements to be considered by the system. In contrast, a note that has only 2 words to be cleaned may take a considerably longer amount



(a) A snapshot of the system at time $t = 6$.



(b) A snapshot of the system at time $t = 12$.



(c) A snapshot of the system at time $t = 29$.

Fig. 4. Snapshots of the system output at three different time stamps.

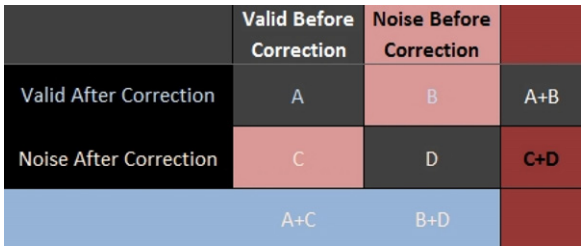


Fig. 6. The layout of the confusion matrices constructed to summarise the feedback of the GP regarding the automatically cleaned notes.

of time because each of these words may have over 10 replacements to be analysed.

To further illustrate the response time of the system, consider Fig. 4. This figure shows the snapshots of the system at three different time stamps as the test note "*neuro: pt alert and orienated x3. able to mae and follow commands. pt difficult to understand at times d/t history of stroke.*" was being entered. More specifically, Fig. 4(a), (b) and (c) shows the system output at time $t = 6$, $t = 12$ and $t = 29$, respectively. At time $t = 6$, 4 out of the 6 words provided were already cleaned. By time $t = 29$, 22 words had already been entered and out of these, 21 were cleaned. The cleaning of the final word was completed at $t = 30.6$. An average of 1.39 s was taken to clean a word in this note.

4.2. Accuracy of the initial cleaning

In the second part of this evaluation, we look at the overall accuracy achieved by the system for the initial corrections (i.e. without the involvement of clinicians' confidence or $\chi_s = 0.5$) together with its noise reduction, noise introduction and noise removal rates. As we have mentioned earlier, we seek the assistance of one general practitioner (GP) to assess the automatically cleaned progress notes. The feedback by the GP was used to establish the gold standard which is later structured as confusion matrices like the one shown in Fig. 6.

From the confusion matrices, we found out that the total number words which are considered as noise is 334. In other words, about 34.76% of the contents of the 30 test notes are actually noise. We had a closer look at the types of noise in the 30 notes. From the 334 noise words, we managed to isolate 322 as abbreviations, 11 as misspellings and 1 as a misspelt abbreviation. Out of the 322 noise words, there are 109 unique ones. There is an obvious tendency of about 96.41% towards abbreviations as noise in the sample notes selected for this experiment. We speculate that this trend is indicative of the actual distribution of noise in real-world composition of progress notes for two reasons. First, the principle of least-effort in language analysis [30] clearly explains the tendency of humans to write using familiar words and briefly. The brevity and prevailing use of abbreviations in progress notes reduces the chances of a clinician making spelling errors. Second, many of the medical and clinical jargons appear as abbreviations in clinical notes. From our observation of the samples, the remaining content is made up of a primarily limited set of everyday words (e.g. "*remain*", "*no*", "*increase*", "*effect*"). The repeated use of these words together with their 'commonness' also reduce the chances of a clinician mispelling them. Having discussed these two reasons, it remains possible that even abbreviations can be misspelt, and in this small experiment, that chance is 0.3%. Using the confusion matrices constructed for all 30 notes, we establish the performance of the system as such:

$$noisereduction = \frac{(B+D)-(C+D)}{B+D} = \frac{B-C}{B+D}$$

$$noiseintroduction = \frac{A}{A+C}$$

$$noiseremoval = \frac{B}{B+D}$$

$$accuracy = \frac{A+B}{A+B+C+D}$$

We discuss each of these performance metrics in the following paragraphs.

The *noise reduction rate* captures the difference in the amount of noise before and after cleaning. The higher the noise reduction rate, the more capable the system is at cleaning existing noise without introducing new noise. The average noise reduction rate of the system in this evaluation was 68.08% with a standard deviation of 19.05%. The least impressive and the best noise reduction rates were 25% and 100%, respectively. In other words, in a note of 4 noisy words, at least 1 will be cleaned in the worst scenario. While the *1-in-4* figure may not appear impressive, it is worth noting that this is achieved without the supervision of domain experts or the involvement of any manually crafted knowledge sources. From the perspective of involving clinicians, we can say that, on average, approximately 1 in 3 existing noise words in progress notes will require clinician's intervention to be resolved. Assuming an average of 1 s per mouse click, 3 clicks to manually correct a noise and that the clinician knows the correct senses and spellings without the need for any thinking, it would require $0.9576X$ s (= 1 s × 3 clicks × $0.3192X$ noise words per note that require manual correction) of the clinician's time to intervene to make each note noise-free, where X is the number of noise words in a note.

The *noise introduction rate* refers to the percentage of valid words that turn into noise after cleaning. Since it is possible for a valid word to still be subjected to cleaning and possibly incorrectly replaced, this measure will determine the system's tendency to be counterproductive by introducing new noise. In this evaluation, the system achieved an average introduction rate of only 1.66%, with a standard deviation of 3.31%. In other words, on average, less than 2 in 100 valid words will be incorrectly replaced and turned into noise after cleaning.

Next, we look at the *noise removal rate* of the system, which refers to the percentage of noise requiring cleaning (i.e. noise before correction) that was actually cleaned (i.e. valid after correction). A removal rate of 100% means that all existing noise in the input was removed and replaced with valid words by the system. In this experiment, the system achieved an average removal rate of 71.63%, with a standard deviation of 16.06%. This translates to the removal of an average of 7 out of 10 noisy words in a note.

As for the overall accuracy, it measures the ability of the system in achieving a higher percentage of word validity given a noisy input. A system with 80% accuracy shows that in a note of 10 words, 8 were valid in the final output. This measure does not distinguish whether the valid words in the output were the result of the cleaning or were already valid prior to processing. Nevertheless, accuracy provides an excellent overview of the ability of the system in producing clean outputs and not worsen the process by turning valid words into new noise. The system fared an average of 88.73% with a standard deviation of 7.32. In this evaluation, the accuracy percentages fall within the range of 66.67% and 100.00%. In other words, in a note of 10 words, close to 9 will be valid after applying the new cleaning system.

5. Limitations and discussions

The system is capable of removing on average 7 out of 10 noise words in a progress note. Considering the simplicity of the

interface (e.g. no user involvement in setting system parameters) and the absence of the need for expertise to maintain complex knowledge bases, such performance is commendable. Due to the relatively small test set, care should be taken when interpreting these figures. The current performance only signifies a baseline which will improve with time through the participation of clinicians. The participation can be as straightforward as simply saving the automatically cleaned notes to the slightly more elaborate action of correcting the automatic replacements. In other words, even though the performance of the system may fluctuate slightly when applied to a significantly larger dataset, the presence of the confidence mechanism will ensure the gradual improvement of the cleaning accuracy in the long run.

The system in its current version splits input text into tokens based on whitespace for processing. The presence of trailing punctuations in tokens is easily removed using heuristics. Moreover, the prevalence of abbreviations, which are obviously single words, amongst the 334 noise words eliminates the need to handle misspellings involving multi-word expressions in this experiment. As justified at the start of Section 4.2, the minimal need to handle misspellings, especially multi-word ones, in this experiment will likely persist into larger-scale evaluations. At the moment, we consider the simple tokenisation approach as adequate and any attempts to incorporate provisions for multi-word misspellings will unnecessarily complicate the system and increase the system's response time. However, we will still investigate the need for these provisions in the future as we deal with larger collections of noisier progress notes.

An issue that will also affect the response time of the system is the need for real-time access to several sources of data on the Web, namely, Yahoo! search and spelling suggestion, and abbreviation dictionaries. The response time of the system can be greatly enhanced by improving the caching mechanism for the data retrieved from the Web. This prevents the system from having to go out to the Web every time the senses of an abbreviation are required. For this reason, the system will initially perform slower. As the system processes more and more notes, the chances of having a local copy of the data needed by the system at any given moment will increase. To improve the system in this regard, we will, amongst others, increase the size of the cache, have a mechanism in place for refreshing the content of the cache and preload the common data (e.g. senses, spelling suggestions) in clinical progress notes.

Along the same line of work, we will carry out an elaborate study on the clinical sub-language, or more specifically, the set of terminology used by clinicians when composing progress notes. It would be interesting to see if the progress note terminology does indeed converge. To study this, we will gradually process an increasing amount of progress notes using the new system. At the back end, we will examine the possibility of arriving at a state whereby no new abbreviations or their corresponding senses are required to clean an ever increasing number of new notes. Similarly, following the same method, we will study whether or not the spelling errors made by clinicians actually belong to a confined finite set. For example, are we able to ascertain if there are only so many variants of the misspellings of the same word?

Another obvious future work is to extend the evaluation of the system. In particular, we will integrate this new cleaning system with a clinical information system currently being used in a number of hospitals and clinics by a variety of health professionals (MMEx). MMEx is a platform maintained by the University of Western Australia Centre for Software Practice (CSP) which offers electronic health record capabilities (http://www.mmex.net.au). MMEx currently supports approximately 6500 health professionals with around 300,000 patient records and transmitting 10,000 secure electronic messages per month. The progress note analysis system will be used to clean progress notes prior to saving and further analysis.

Last, this work represents the first step in the challenging attempt of achieving full semantic understanding of progress notes. This system may be able to recognise that "*AF*" refers to "*atrial fibrillation*" instead of 15 other different senses. A lot more work, however, is still required to create a system capable of understanding what "*atrial fibrillation*" is and the implications of its presence in the note. A possible next step after cleaning is to create a mapping mechanism to link extracted named entities to the content of some semantic knowledge base such as the UMLS.

6. Conclusion

Clinical progress notes are a rich source of valuable data for improving the many facets of health care. The problem, however, lies in the noisy nature of such texts, and the inability of many of the present technologies to adequately process and analyse them. Existing work either ignores the noise, requires the input data to be cleaned beforehand, or uses techniques that are expensive to set up and impractical to deploy.

In this paper, we presented a robust system for cleaning clinical progress notes with a focus on abbreviations and misspellings. Clinicians are able to enter progress notes through a simple interface and obtain the corresponding cleaned notes in real-time. Our initial evaluation demonstrated that the cleaning process is capable of reaching the average typing speed of clinicians which is about 1.5–2 s per word. The system initially relies on statistical semantic analysis using only Web data to produce the preliminary corrections. The results achieved using 961 words over 30 actual progress notes showed that an accuracy of 88.73% is possible using only statistical evidence from Web data.

References

[1] Friedman C. Semantic text parsing for patient records. In: Chen H, Fuller S, Friedman C, Hersh W, editors. Medical informatics. USA: Springer; 2005.
[2] Xu H, Stetson P, Friedman C. A study of abbreviations in clinical notes. In: Teich J, Suermondt J, Hripcsak G, editors. Proceedings of the AMIA annual symposium. Bethesda, MD: AMIA Press; 2007. p. 821–5.
[3] Demner-Fushman D, Chapman W, McDonald C. What can natural language processing do for clinical decision support? Journal of Biomedical Informatics 2009;42(5):760–72.
[4] Friedlin J, Grannis S, Overhage M. Using natural language processing to improve accuracy of automated notifiable disease reporting. In: Suermondt J, Evans R, Ohno-Machado L, editors. Proceedings of the AMIA annual symposium. Bethesda, MD: AMIA Press; 2008. p. 207–11.
[5] Wang Y. Annotating and recognising named entities in clinical notes. In: Dimalen D, Finkel J, Thomson B, editors. Proceedings of the ACL-IJCNLP student research workshop. Rochester, NY: ACL Press; 2009. p. 18–26.
[6] Stevenson M, Guo Y, Al-Amri A, Gaizauskas R. Disambiguation of biomedical abbreviations. In: Cohen K, Demner-Fushman D, Ananiadou S, Pestian J, Tsujii J, Webber B, editors. Proceedings of the NAACL HLT workshop on BioNLP. Rochester, NY: ACL Press; 2009. p. 71–9.
[7] Yu Z, Tsuruoka Y, Tsujii J. Automatic resolution of ambiguous abbreviations in biomedical texts using support vector machines and one sense per discourse hypothesis. In: Brown E, Hersh W, Valencia A, editors. Proceedings of the SIGIR workshop on text analysis and search for bioinformatics. NY: ACM Press; 2003. p. 57–62.
[8] Roche M, Prince V. Managing the acronymexpansion identification process for text-mining applications. International Journal of Software and Informatics 2008;2(2):163–79.
[9] Damerau F. A technique for computer detection and correction of spelling errors. Communications of the ACM 1964;7(3):171–6.
[10] Levenshtein V. Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady 1966;10(8):707–10.
[11] Wagner R, Fischer M. The string-to-string correction problem. Journal of the ACM 1974;21(1):168–73.
[12] Kukich K. Technique for automatically correcting words in text. ACM Computing Surveys 1992;24(4):377–439.
[13] Odell M, Russell R. U.S. patent numbers 1,261,167. U.S. Patent Office, Washington, DC; 1918.
[14] Odell M, Russell R. U.S. patent numbers 1,435,663. U.S. Patent Office, Washington, DC; 1922.

[15] Philips L. Hanging on the metaphone. Computer Language Magazine 1990;7(12):38–44.
[16] Lait A, Randell B. An assessment of name matching algorithms. Technical report. University of Newcastle upon Tyne; 1993.
[17] Holmes D, McCabe C. Improving precision and recall for soundex retrieval. In: Werne B, editor. Proceedings of the international conference on information technology: coding and computing (ITCC). Los Alamitos, CA: IEEE Computer Society Press; 2002. p. 22–6.
[18] Gupta D, Saul M, Gilbertson J. Evaluation of a deidentification (de-id) software engine to share pathology reports and clinical documents for research. American Journal of Clinical Pathology 2004;121(1):176–86.
[19] Schwartz A, Hearst M. A simple algorithm for identifying abbreviation definitions in biomedical text. In: Altman R, Dunker K, Hunter L, Jung T, Klein T, editors. Proceedings of the pacific symposium on biocomputing (PSB). Singapore: World Scientific Publishing; 2003. p. 451–62.
[20] Chang J, Schutze H, Altman R. Creating an online dictionary of abbreviations from medline. Journal of the American Medical Informatics Association 2002;9(6):612–20.
[21] Pakhomov S. Semi-supervised maximum entropy based approach to acronym and abbreviation normalization in medical texts. In: Isabelle P, editor. Proceedings of the 40th annual meeting on association for computational linguistics (ACL). Rochester, NY: ACL Press; 2002. p. 160–7.
[22] Pakhomov S, Pedersen T, Chute C. Abbreviation and acronym disambiguation in clinical discourse. In: Friedman C, Ash J, Tarczy-Hornoch P, editors. Proceedings of the AMIA annual symposium. Bethesda, MD: AMIA Press; 2005. p. 589–93.
[23] Ruch P, Baud R, Geissbuhler A. Using lexical disambiguation and named-entity recognition to improve spelling correction in the electronic patient record. Artificial Intelligence in Medicine 2003;29(1–2):169–84.
[24] Xu H, Stetson P, Friedman C. Methods for building sense inventories of abbreviations in clinical notes. Journal of American Medical Informatics Association 2009;16(1):103–8.
[25] Zhou W, Torvik V, Smalheiser N. Adam: another database of abbreviations in medline. Bioinformatics 2006;22(22):2813–8.
[26] Atkinson K. Gnu aspell 0.60.4, http://aspell.sourceforge.net; 2006.
[27] Kuenning G. International ispell 3.3.02, http://ficus-www.cs.ucla.edu/geoff/ispell.html; 2006.
[28] Vitanyi P, Balbach F, Cilibrasi R, Li M. Normalized information distance. In: Emmert-Streib F, Dehmer M, editors. Information theory and statistical learning. New York: Springer; 2009.
[29] Dawes M, Chan D. Knowing we practise good medicine: implementing the electronic medical record in family practice. Canadian Family Physician 2010;56(1):15–6.
[30] Zipf G. Human behaviour and the principle of least-effort. Cambridge, MA: Addison-Wesley; 1949.