A Multi-lingual TN/ITN Framework for Speech Technology

Hyongsil Cho[1,2], *Daniela Braga*[1,2], *Cristiano Chesi*[1,2], *Daan Baldewijns*[1], *Manuel Ribeiro*[1], *Kaisa Saarinen*[1], *Jeppe Beck*[1], *Silvia Rustullet*[1], *Peter Henriksson*[1], *Miguel Dias*[1,2], *Heiko Rahmel*[3]

[1] Microsoft Language Development Center, Portugal
[2] ISCTE-Lisbon University Institute, Portugal
[3] Microsoft Speech Components Group, Redmond, USA

{v-hych, i-dbraga, v-crches, v-daanb, i-manrib, v-kasaa, v-jeppeb, i-sirust, Miguel.Dias, heikora}@microsoft.com

Abstract

Although the research community pays little attention to (Inverse) Text Normalization (TN and ITN), this is an essential module in Text-to-Speech (TTS) and Speech Recognition (SR) systems. It has a significant development timeline and requires deep linguistic expertise. One of the main issues is ambiguity resolution, which is particularly problematic when handling numerals in different languages, especially those with gender or case variation. In this paper, we present a framework that can deal simultaneously with TN and ITN and which was applied to twelve different languages. The rules were tested and subsequently refined. The overall performance of the system is presented and discussed.

Index Terms: (inverse) text normalization, text-to-speech, automatic speech recognition

1. Introduction

Text normalization (TN) is a core module in any speech synthesis or Text-to-Speech (TTS) system. As a part of text pre-processor, the text normalization module converts a raw text file, which contains non-standard words such as sequences of digital bits or abbreviations, into a well-defined sequence of linguistically-meaningful units. The same module can be also used by speech recognition systems (SR), where the module performs the same task but in the opposite way and the operation is called then Inverse Text Normalization (ITN).

Text pre-processing is an essential part of any NLP system, since the characters, words, and sentences identified at this stage are the fundamental units passed to all further processing stages, from analysis and tagging components, such as morphological analyzers and part-of-speech taggers, through applications, such as information retrieval and machine translation systems [2].

The usual approach for this is to use phonetic lexica, i.e., a list of entries and their respective orthographic expansion or immediate phonetic transcription. However, a list-based approach has a number of important drawbacks.

First, a list-based approach is linear. So, to include all numbers up to a million, the same number of list entries would be required. More complex structures as dates, times, currency units, mathematical expressions and telephone numbers – which often have many possible orthographic representations – are even more problematic to cover exhaustively.

A second issue is the problem of ambiguity for TN. A non rule-based approach hardly ever offers descriptions or solutions for the different cases of ambiguities in this module, and for which disambiguation rules need to be proposed. For example, while a native language speaker of Portuguese language might find it relatively easy to decide whether *SPA* should be read as an acronym and therefore spelled out as *Sociedade Portuguesa de Autores*, or as an acronym and pronounced as a single syllable when it means *Salutem per Acquam*, this is not true of a computer application, which requires context information and rules to make such a decision. The same happens when with Roman numerals, such as *I, V, X, C, D* or *M,* which are written using letters and therefore cause ambiguities with acronyms. For example, a sequence like *D.* occurring in a text written in Portuguese may mean not only the number *500* but also *Dom* or *Dona*, depending on the context

This problem is also related to the last disadvantage of list-based text normalize. A simple list, when performing TN, is unable to resolve agreement between the item to be normalized and its context. In Portuguese, the number 1 can have a masculine or feminine readout, depending on the context; and for highly inflected languages like Finnish which has a system of 15 cases, the chance of having this kind of problem is even bigger.

In this study, we created a rule-based multi-lingual TN/ITN algorithm and tested its performance over a number of languages from different families.

2. Domains and issues of text normalization

2.1. Domains of text normalization in Speech Synthesis and Recognition Systems

The size of a text normalization module varies depending on its applications. A number of studies have defined objects of text normalization and identified characters of each object.

Some previous works investigated the problems of named entity recognition in informally inputted texts and proposed improving the performance of personal name recognition in emails using two machine-learning based methods [6][1]. For email data cleaning, a cascaded approach by employing Support Vector Machines and rules has been proposed in [10]. While some studies pay attention to the case restoration problem [4][3][5], others concentrated to the normalization of non-standard words in texts, including numbers, abbreviations, dates, currency amounts, and acronyms. They propose taxonomy of non-standard words and apply n-gram models, decision trees and weighted finite-state transducers to the normalization [8].

Our approach to the text normalization focuses on the non-standard words normalization. To do this, we first defined the categories of non-standard words to be normalized by our text

normalization module. Each of the categories can be described by its example as follow:

- Cardinal (positive/negative numerals)
  - Integers of range 1 - 999: 2, -59
  - Integers of range 1,000 - 10,000: 5000, 3,589 or 3589
  - Large integers : 301,020 or 301020
  - Decimals : 0.1, 0.03, 0.987
- Ordinal: 1st, 2nd
- Roman Numbers: I, II, V
- Date
  - Date expression by digits: 5/10/2000
  - Stand alone year: 1989, '89
  - Day and month: 5/10, 5-10
  - Month and year: 10/2000, 10.2000
  - Decade: 60', 70'
- Time : *time expression with a variety of signs* (:, -, a.m., p.m., etc)
- Fraction: ½, ¼, ¾
- Telephone Number
  - Local numbers: 555-1212
  - Local numbers with area code: (267) 555-1212
  - Local numbers with country code: +1-212-735-0989
  - International dialing: +31 24 323 5647
- Measurement: 20l, 300km, $5m^2$
- Degree: 480°
- Address
  - Street address line 1: *Street number and name*
  - Street address line 2: *Name of town, State (abbreviation) + ZIP code*
  - Optional street address line: *any optional information*
  - Post office box: P.O.BOX 1320
- Currency
  - ISO Standard: 200$, $200
  - Subdivision of currencies: 1.75$
- Contact Names
  - (Honorific) Titles: Mr., Mrs., Dr.
- URL/e-mail address: abc@12f3.net
- Range expression for numerals and date: 1~20, 5/10/2000~10/10/2000

2.2. Language specific issues

To normalize correctly all of the categories described above, a number of language specific issues must be considered.
For example, in some West Germanic languages, numerals between 11 and 99 need a special attention for their specific word order. The following example illustrates the case of the Dutch language:

Ex) 23: drieëntwintig (3+and+20 in one word)
44: vierenveertig (4+and+40 in one word)

Moreover, in case of the French language, the expression of some numbers is made in a very particular way:

Ex) 10 : dix
60: soixante
70: soixante-dix (60+10)
80 : quatre-vingts (4 times of 20)

Also, as described briefly in the section 1, the gender distinction is very important for Romance languages and a part of Germanic languages.
The forms and the expression of numerals get a bigger diversity when it comes to the telephone number: the number of units in a standard telephone number is different from a country to another and the way of reading a telephone number is also highly varied depending on the country.
For example, a French standard telephone number is composed by 5 times of two unit numbers and all of them are read as tens.

Ex) 01 91 28 64 32
: zéro un quatre-vingt-onze vingt-huit soixante-quatre trente-deux

Whereas a Korean telephone number must be read digit-by-digit.

Ex) 361-2839
: 삼(3)육(6)일(1)에(-) 이(2)팔(8)삼(3)구(9)

In some languages, text normalization modules need to consider the phonetic context of items. For instance, in Italian, the common abbreviation for *Saint* (i.e. S.) should be normalized not only according to the gender of the saint's name:

Ex) S. Marco: San Marco ("Marco" is masculine)
S. Maria: Santa Maria ("Maria" is feminine)

but also keeping into account if the name starts by vowels or not:

Ex) S. Antonio : Sant'Antonio
("Antonio" is masculine and begins with a vowel)
S. Anna: Sant'Anna
("Anna" is feminine and begins with a vowel)

Concerning to the context, another critical issue is that ambiguities in normalization are not always locally solvable, that is, it is not enough to look at the adjacent words to guess the correct normalization. In that sense, the item *S.*, mentioned in the previous examples, can be a hard element to handle.

Ex) S.S. Appia 7: Strada Statale Appia sette

In this example, without a deep understanding of the word which follows *S.S.*, a text normalization module may make the fatal mistake to get *Santissima Appia sette* or *Santo Santa Appia sette* instead of *Strada Statale Appia sette*.

To handle all these issues, we built a rule-based TN/ITN module with possibilities of implementing and controlling the information about context.

3. A Rule-based multi-lingual TN/ITN framework

3.1. Structure and hierarchy of the rules

We built, in 2.1, a list of domains of non-standard words. In our TN/ITN module, these domains will be associated with top-level rules, with each one being basically independent. Although the rules are self-referent and the domains intercross, the fact that they are top-level rules enables them to be

independent without being influenced by the behavior of other rules (except for cases where there are identical inputs.

Text normalization rules are structured based on a language called TNML (Text Normalization Mark-up Language), which is an adaptation of SSML (Speech Synthesis Markup Language) for the TN module. SSML has been recommended by the W3C (World Wide Web Consortium) since 2004 and is currently one of the most commonly used mark-up languages.

However, while the language is easy to understand, the sheer number of rules and constant referencing makes the process too intricate to be handled in a text editor. For this reason, we decided to make easier the process of writing down the text normalization rules by creating an internal tool that establishes a bridge between the rules and the language in which they are written. The TNML programming language and testing process presented in this paper are protected by patent (Patent Serial No. 12/361,114).

Using this tool, we can create a TNML file that we call a "map". The structure of a TN map is basically a tree-type scheme, with successive function-based references and positions. Four types of rules have been defined in TNML: terminals, sequence rules, list rules and top level rules. Thus, a map is composed by one or more top level rules and a top level rule possess one or more sequence rules, list rules and/or terminals. The characteristics of each rule are described in the following sections.

3.1.1. Terminals

Terminals are the elements placed at the bottom in the rules tree of a text normalization map. Terminals contain the information to be used during TN/ITN process and establish a one-to-one relation between a display form and a spoken form. For example, in case of the Portuguese language (see 1), we put the information that the numeral '1' must be normalized as *um* or *uma* is contained on the terminal level by making two different terminals, one with *um* and the other with *uma*, for one same item '1'.

In French language's case, some items may have a bigger number of terminals.

Display	Spoken
4	Avril
4	quarante
4	quart
4	quatorze
4	quatre

Table 1: *Terminals related to the item '4' in French*

All the other levels of rules are authored based on the terminals. In a rule which normalize numerals 0~9, the terminal 4⇔quatre will be used whereas a rule to normalize the date, especially the month, will need the terminal 4⇔Avril.

3.1.2. List rules

List rules represent their references in a vertical way. Their main characteristic is that they allow for only of the different available normalizations in the list. This allows for a grouping of the elements in a way that they can be reused or referenced by other rules where only one of the elements is selected. For example, the list rule shown in the Table 2 regroups a number of terminals of English language which may function in the same way.

List rule name	Composition (Terminals)	
	Display	**Spoken**
Cardinal 1 to 9	1	one
	2	two
	…	…
	8	eight
	9	nine

Table 2: *A list rule*

A list rule can be a direct constituent of a top level rule, a sequence rule or another list rule.

3.1.3. Sequence rules

Differently from the list rules which regroup terminals and rules of a same class, the sequence rules concatenate terminals rules by defining their order within a sequence. Given a terminal 3⇔thirty and the list rule shown in the Table 2, we can create a sequence rule which will normalize all the numerals between 31 and 39.

Sequence rule name	Composition	Example
Cardinal 31 to 39	(3⇔thirty)+(Cardinal 1 to 9)	31⇔ thirty one … 39⇔thirty nine

Table 3: *A sequence rule*

A list rule can be a direct constituent of a top level rule, a list rule or another sequence rule.

3.1.4. Top level rules

Top-level rules are the elements placed at the top in the rules tree of a text normalization map. All other rules run towards the top-level rules, which are the entry gate to a TN map. When starting a normalization process, the system always starts off by using a top-level rule. In other words, any terminal, list rule or sequence rule which is not included in a top level rule will not make any effect in the normalization. As such, all rules must directly or indirectly be associated with a top-level rule.

In our TN/ITN module, we built one top level rule per category (defined in 2.1).

3.1.5. Other components

In addition to the four types of rules described above, our TNML maps may contain some supplementary information such as:

- Spaces control between the constituents of a sequence rule. They can be toggled on or off.

Name	Display	Spoken
SpcToSpc	<sp>	<sp>
SpcToNo	<sp>	<ns>
NoToSpc	<ns>	<sp>
NoToNo	<ns>	<ns>

Table 4: *Space control*

In fact, in the sequence rule shown in Table 3, NoToSpc is applied between <3⇔thirty> and <Cardinal 1 to 9> and that's how the rule could produce <31⇔ thirty one> and not <31⇔ thirtyone>.

- Priorities: Assuming a map for TN (used in speech synthesis) and ITN (used in speech recognition), priorities allow for assigning values to disambiguate identical expressions, by using a system of weight defined for each constituent of a list rule.

- Agreement: Agreements make it possible for the TN rules to bridge into an annotated lexicon and extract information from it to facilitate disambiguation in specific cases.

3.2. TN/ITN framework applied to 12 languages

The performance of our TN/ITN framework was tested for a number of languages from different language families. The languages are en-GB, de-DE, fr-FR, it-IT, pt-PT, ca-ES, es-ES, nb-NO, da-DK, nl-NL, sv-SE and ko-KR.

To test the performance, a text normalization map was created. During the whole process of building and re-fining the text normalization map, 3 types of tests were carried out:

- Accuracy tests in internal tool
- Performance tests on a large sized text corpus
- Overall tests on a set of pre-selected items

First, the accuracy tests in internal tool were performed during the coding and fine-tuning of rules. These are simple tests (input/output) normally associated with a top level rule. The purpose of these preliminary tests is to assess the functionality of the rules as they are being created, and to fix any small concatenation and referencing errors, such as extra or missing spaces, agreement issues, reference or structure errors, etc.

Although the number might vary depending on the rule, a minimum number of nine cases have been established for each top-level rule. A few other cases were added to sub-rules in order to test their efficiency during the rule-writing process. The intended mark for these preliminary tests is a 100% accuracy rate.

Second, the performance tests on a large sized text corpus assume the existence of a beta version of the text normalization map, with all domains finalized and preliminary tests completed. During these tests, TN/ITN rules are applied on corpus composed by 50,000~100,000 sentences, depending on the language and the nature of corpus, collected from various sources in order to obtain and analyze approximately 20,000 normalized items. For example, to test the performance of European Portuguese map, a corpus of approximately 60,000 random phrases was used.

The goal of these tests is that our map recognizes any non-standard word occurring in the raw text data and normalizes it correctly. After analyzing the results, the rules are reviewed again in order to add unexpected patterns and fix any errors found. By analyzing the errors, we could observe how our TN/ITN map works when it is combined with the other modules of a TTS or SR system.

To finish, overall tests on a set of pre-selected items were made. A text data set composed by approximately 1,000 non-standard words (distributed over the all domains defined in 2.1) was built. Differently from the corpus used for the previous performance tests, the data set built for the overall tests doesn't contain any standard shape of sentence but only non-standard words like 11/5/1989 for date or 378-2684 for telephone number. Each of those items has the (pre-defined) expected spoken form and the goal of the overall test is i) our TN/ITN map of the given language places a given item in the right category, ii) the rules applied to the item generate the correct expected spoken form for the given item and iii) the map works also correctly in the ITN direction (from the spoken form to the display form).

After having run a first normalization, the errors were categorized, analyzed and fixed, and a final hit rate of a 100% has been achieved.

3.3. Result and discussion

For a text normalization module, it is a very hard task to get a 100% accuracy rate on an infinitely large corpus composed from various sources. There will always be word sequences or characters that contain unexpected problems for a phonemic conversion algorithm. Therefore, building and maintaining a text normalization module is inevitably work in progress and will always be a cyclic process.

In this sense, our rule-based multi-lingual TN/ITN framework gave a reasonable result by getting a 100% success rate in the overall tests on preselected items. The result leads us to conclude that the rule-based approach provides solutions to ambiguities and contextual agreement, and allows a much higher level of exhaustiveness. However, it must be realized that reviewing and modifying the maps and the whole framework will remain necessary.

4. Conclusion

We built a rule-based multi lingual TN/ITN framework, in which maps for a number of languages were authored by a group of language experts. The performance tests of each language's map provided satisfactory results.

In the future, improvement of our module and enhanced accuracy are sought by focusing on a specific text domain: for instance medical text, bible texts, historical text, etc. It will be interesting to implement an auto-tuning function to refines the maps depending on the application (TTS, SR, or other application). We will also look for ways of reducing the size of maps without loss of performance or accuracy.

5. References

[1] Carvalho V. R. and W. W. Cohen. 2004. Learning to Extract Signature and Reply Lines from Email, Proc.of CEAS 2004.

[2] David D. Palmer, "Text Pre-processing", in Handbook of Natural Language Processing, Second Edition, Goshen, Connecticut, USA, 2010

[3] Golding A.R.and D. Roth. 1996. Applying Winnow to Context-Sensitive Spelling Correction, Proc. Of ICML'1996.

[4] Lita L. V., A. Ittycheriah, S. Roukos, and N. Kambhatla.2003. tRuEcasIng, Proc. of ACL 2003.

[5] Mikheev, A. 2000. Document Centered Approach to Text Normalization, Proc. SIGIR 2000.

[6] Minkov E., R. C. Wang, and W. W. Cohen. 2005. Extracting Personal Names from Email: Applying Named Entity Recognition to Informal Text, Proc. Of EMNLP/HLT-2005.

[7] Palmer D. D. and M. A. Hearst. 1997. Adaptive Multilingual Sentence Boundary Disambiguation, Computational Linguistics, Vol. 23.

[8] Sproat R., A. Black, S. Chen, S. Kumar, M. Ostendorf, and C. Richards. 1999. Normalization of nonstandard words, WS'99 Final Report. http://www.clsp.jhu.edu/ws99/projects/normal/.

[9] Stone, H.S., "On the uniqueness of the convolution theorem for the Fourier transform", NEC Labs. Amer. Princeton, NJ. Online: http://citeseer.ist.psu.edu/176038.html, accessed on 19 Mar 2008.

[10] Tang J., H. Li, Y. Cao, and Z. Tang. 2005. Email data cleaning, Proc. of SIGKDD'2005.