Multilingual Number Transcription for Text-to-Speech Conversion

R. San-Segundo[1], *J.M. Montero*[1], *M. Giurgiu*[2], *I. Muresan*[2], *S. King*[3]

[1]Speech Technology Group, ETSI Telecomunicación. UPM. Spain.
[2]Dept. of Telecommun., Tech. Univ. of Cluj-Napoca, Cluj-Napoca, Romania.
[3]Centre for Speech Technology Research, University of Edinburgh, UK.

lapiz@die.upm.es

Abstract

This paper describes the text normalization module of a text to speech fully-trainable conversion system and its application to number transcription. The main target is to generate a language independent text normalization module, based on data instead of on expert rules. This paper proposes a general architecture based on statistical machine translation techniques. This proposal is composed of three main modules: a tokenizer for splitting the text input into a token graph, a phrase-based translation module for token translation, and a post-processing module for removing some tokens. This architecture has been evaluated for number transcription in several languages: English, Spanish and Romanian. Number transcription is an important aspect in the text normalization problem.

Index Terms: Multilingual Number Transcription, text normalization, fully-trainable text conversion.

1. Introduction

Although Text to Speech (TTS) conversion is the area where more effort is devoted to text normalization, dealing with real text is a problem that also appears in other applications such as machine translation, topic detection and speech recognition when it is necessary to associate a phoneme sequence to a written word. In an ideal situation, there would be an unambiguous relationship between spelling and pronunciation. But in real text, there are non-standard words: numbers, digit sequences, acronyms, abbreviations, dates, etc. The main problem of a text normalization module consists of converting Non-Standard Words (NSWs) into regular words. This problem can be seen as a translation problem between a real text with NSWs and an ideal text where all the words are standard: unique relationship between word spelling and its pronunciation.

2. State of the art

One of the main references focused on text normalization is [1]. In this reference, authors propose a very complete taxonomy of NSWs considering 23 different classes grouped in three main types: numerical, alphabetical and miscellanea. Sproat et al describes the whole normalization problem of NSWs, proposing several solutions for some of the problems.

Additionally, it is also important to mention other references that have addressed specific problems included in the text normalization research line. Focused on abbreviations and acronyms, there are several efforts focused on extracting them from text automatically [2] and other efforts trying to model how they are generated [3]. Numbers [4] and proper names [5, 6] have been also the target of other research works. Number transcription has been missing from previous efforts and this paper contribute to complete this work [7]. Nowadays, much effort on text normalization is focused on SMS language interchanged through mobile phones and social networks like Facebook or Twitter [8, 9].

Due to the important advances obtained in machine translation in the last decade, there has been an increasing interest on using machine translation capabilities for dealing with the problem of text normalization [10, 11]. Text Normalization is an important aspect, not only for Text-to-Speech Conversion but also for Text Categorization [12] or Text Classification. [13].

3. Architecture description

Figure 1 shows the architecture diagram proposed in this paper. This architecture is composed of three main modules: a pre-processing module that splits the text input into a token graph, a phrase-based translation module based on Moses software, and a post-processing module for removing some tokens.

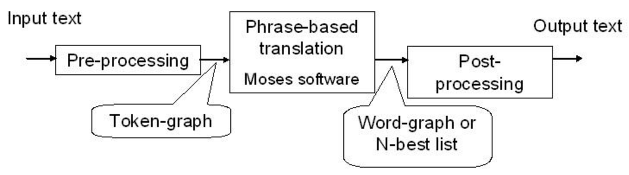

Figure 1: Architecture diagram

3.1. Pre-processing: Sentence Tokenization

In this first module, the input text is split into tokens. This process is carried out in two different steps. At the first step, a preliminary token sequence is generated considering a small set of rules. As one of the main targets of this work is to provide a language independent architecture, the main rules should be language independent:

- The first rule assumes that blank characters provide an initial segmentation in tokens.
- The second rule subdivides initial tokens considering some homogeneity criterions: tokens must have only alpha or numerical characters. If there is a change from alpha to number or vice-versa, the token must be subdivided. Secondly, punctuations characters are independent tokens.

Secondly, some of the tokens are re-written in a different format in order to facilitate their posterior translation. In this step, the idea is to classify each token as a standard word (W) or as a non-standard word (NSW). This classification can be done considering a dictionary of standard words in this language or considering a more complex classifier based on some features obtained from the target token and its context: character language model, vowels, capitals, etc. In this work, detecting numbers is quite simple based on the characters componing the token.

If the token is classified as a NSW, it is split into letters including some separators at the beginning and at the end of the letter sequence. For example, UPM (Universidad Politécnica de Madrid in Spanish) is rewritten into # U P M #. This way of rewriting an alpha token tries to introduce a high flexibility to facilitate the text normalization process. Considering sequences of letters, some unseen acronyms could be normalized by spelling (using the translations of its graphemes individually).

Also, all the numbers are rewritten dividing the token into digits. This work has considered two alternatives. In the first one, every digit is complemented with its position in the number sequence. For example: 2013 is rewritten as 2_4 0_3 1_2 3_1, where 2_4 means the digit 2 in the 4th position (position beginning from the right). The second alternative consists of dividing the number sequence in sets of three digits in sequence, complementing with its position in the sequence, and including additional tags to separate 3-digits sequences: 2013 is written as 2_1 tag1 0_3 1_2 3_1. The Roman numbers are first translated into Arabic ones and then, rewritten digit by digit. Ordinal numbers are not treated in this paper.

As it will be shown in the next section, the translation module can deal with graphs of token as input. Thanks to this possibility, it is possible to work with fuzzy decisions when classifying every token as standard word or NSW. Considering a token graph, both alternatives can be considered with different weight if necessary. Figure 2 shows an example of token graph for the sentence "Welcome to UPM2013"

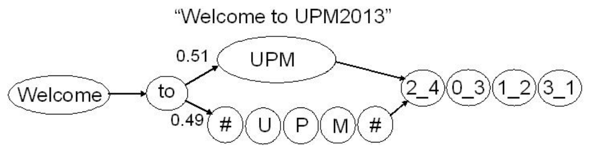

Figure 2: Token graph for the sentence "Welcome to UPM2013"

The token "UPM2013" is divided into two tokens: UPM and 2013. UPM, is rewritten considering two possibilities: as it is, and letter by letter. The second one is a number and it is rewritten digit by digit, considering the first alternative commented above.

3.2. Token Translation

The token translation is performed using a phrase-based system. The phrase-based translation system is based on the software released from NAACL Workshops on Statistical Machine Translation in 2012. The translation process uses a phrase-based translation model and a target language model. These models have been trained in accordance with these steps, see Figure 3.

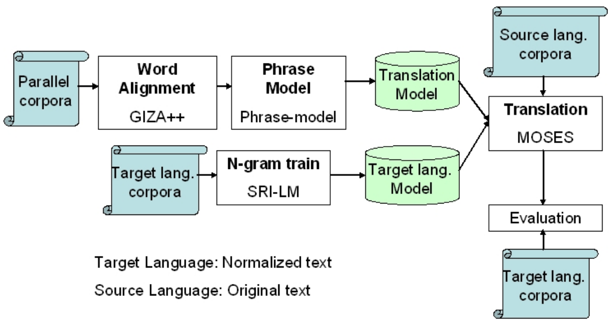

Figure 3: Process for training the translation and target language models

The first step is word alignment computation using the GIZA++ software [14]. In order to establish these alignments, GIZA++ combines the alignments in both directions. As there are many standard words, they are the same tokens in source and target languages, being important reference points for the alignment.

The second step is phrase extraction [15]. All token phrase pairs that are consistent with the token alignment are collected. Finally, the last step is phrase scoring. In this step, the translation probabilities are computed for all phrase pairs. Both translation probabilities are calculated: forward and backward.

The Moses decoder, available at http://www.statmt.org/moses/, is used for the translation process. This program is a beam search decoder for phrase-based statistical machine translation models. The N-gram language model has been generated with the SRI language modelling toolkit [16].

3.3. Post-processing

This module performs several actions in order to generate the normalized text to the speech synthesiser. One of the main actions is to remove unnecessary tokens. For example, if after the translation module there is any # token, used for defining the limits of the letter sequences, it must be removed.

Additionally, given that the translation module can generate a token graph or a sequence of N-best token sequence, it would be possible to add new translation modules in order to improve the translation process by considering new language models for reordering the N-best token sequences or searching the output token graph.

4. Multilingual Number Transcription

This section reports the experiments to adapt the text normalization module for multilanguage number transcription: English (EN), Spanish (ES) and Romanian (RO).

The database for the experiments has been generated randomly by taking into account several patterns of numbers including numbers with decimals. These numbers have been dividied into three sets: training (800), tuning (1000) or testing (5000). Every number belongs to only one of the sets. The evaluation measurements are BLEU, WER (Word Error Rate) and SER (Sentence Error Rate).

4.1. Number Tokenization

This section evaluates the two tokenization alternatives described above. In the first one, every digit is complemented with its position in the number sequence. For example: 2013 is rewritten as 2_4 0_3 1_2 3_1. The second alternative consists of dividing the number sequence in sets of three digits in sequence, complementing with its position in the sequence, and including additional tags between 3-digit sequences: 2013 is written as 2_1 tag1 0_3 1_2 3_1. In these experiments, the default values for the translation architecture have been considered: a 3-gram language model and grow-diag-final alignment. The sizes of the sets are 800 for training, 1000 for tuning and 5000 for testing.Table 1 shows improvement when considering the second tokenization alternative.

4.2. Token Translation

In the current experiments we have used in the training stage three types of alignments between the tokens in the source and the words in the target text with the aim to estimate statistical parameters for the number transcription system. Two types of word-to-word alignments ("tgt-

Tokenization			
EN	BLEU	WER	SER
1st alternative	97.9	1.6	7.4
2nd alternative	**98.5**	**0.8**	**6.8**
ES	BLEU	WER	SER
1st alternative	97.8	1.9	6.8
2nd alternative	**98.2**	**0.9**	**6.1**
RO	BLEU	WER	SER
1st alternative	98.5	0.9	5.4
2nd alternative	**99.2**	**0.5**	**4.6**

Table 1: Experiments with different tokenization

Aligment for training the translation model			
EN	BLEU	WER	SER
grow-diag-final	98.5	0.8	6.8
srctotgt	**99.4**	**0.4**	**4.4**
tgttosrc	98.1	0.7	6.4
ES	BLEU	WER	SER
grow-diag-final	98.2	0.9	6.1
srctotgt	98.2	0.9	6.1
tgttosrc	**98.5**	**0.7**	**4.5**
RO	BLEU	WER	SER
grow-diag-final	**99.2**	**0.5**	**4.6**
srctotgt	98.9	0.5	5.1
tgttosrc	98.5	0.8	7.8

Table 2: Experiments with different alignments used when training the translation model

tosrc" - target to source, and "srctotgt" - source to taget), as well as the "grow-diag-final" heuristic model, which consists of a number of intersections and unions aimed to cope with the asimetry of the word alignment models (Table 2). The best result has been obtained with different alignments depending on the language. Although the differences are not very high, the best alignment must be adapted depending on the language. The alignments are slightly dependent on the language due to the irregularities in number transcription and the use of language-specific function words. For example the number 30.000 is "thirty thousand" in English, but "treizeci de mii" in Romanian (functional preposition "de" is used), while 3.000 is "three thousand" in English, and "trei mii" in Romanian (without to use the preposition "de"). Therefore the heuristic alignment "grow-to-diag" would be more appropriate for Romanian. Analyzing the average number of words needed for transcribing the same number, we obtain 9.9 words per number for English, 10.1 words for Spanish and 13.3 words for Romanian, considering the same set of numbers with an average length of 8.3 digits, including decimals, dots and commas. For experiments in Tables 2 and 1, the three original sets were considered:

training (800), tuning (1000) or testing (5000).

Training Set Size			
EN	BLEU	WER	SER
200 numbers	98.3	0.8	6.4
400 numbers	99.3	0.4	4.6
800 numbers	99.4	0.4	4.4
4000 numbers	**99.9**	**0.2**	**1.8**
ES	BLEU	WER	SER
200 numbers	97.3	1.5	8.8
400 numbers	98.2	0.9	6.0
800 numbers	98.5	0.7	4.5
4000 numbers	**99.6**	**0.2**	**1.0**
RO	BLEU	WER	SER
200 numbers	95.2	2.3	20.2
400 numbers	97.7	1.3	11.2
800 numbers	99.2	0.5	4.6
4000 numbers	**99.7**	**0.2**	**2.4**

Table 3: Experiments with different training sets

Finally, we have evaluated the influence of the size of the training set (Table 3). When decreasing the number of training examples the error increases. This increment is higher for those languages that need more words for transcribing a number. In these cases, we need more data to train the models. Analysing the errors, we have realized that many errors comes from the decimal part. In this case, the tokenization is not appropiate for this part. In order to analyze the influence of the decimal part, we carried out experiments with and without the decimal part considering only 200 numbers for training (Table 4). As it is shown, the error reduction is significant.

The influence of decimal part (200 numbers)			
EN	BLEU	WER	SER
With decimal part	98.3	0.8	6.4
Without decimal part	**99.3**	**0.4**	**3.6**
ES	BLEU	WER	SER
With decimal part	97.3	1.5	8.8
Without decimal part	**98.2**	**0.8**	**6.0**
RO	BLEU	WER	SER
With decimal part	95.2	2.3	20.2
Without decimal part	**97.9**	**1.1**	**11.2**

Table 4: Experiments without the decimal part using 200 numbers for training

4.3. Postprocessing

In this step, the main target is to avoid the presence of input tokens in the output sentence. If one initial token has not been translated, the postprocessing step replaces this token with the most probable translation: digits in our case. In this work, the postprocessing step did execute very few replacements, less than 0.1%.

5. Conclusions

This paper has presented a text normalization module to be integrated in a text to speech fully-trainable conversion system and its application to number transcription. The text normalization module proposed is based on statistical machine translation techniques. This module is composed of a tokenizer for splitting the text input into a token graph , a phrase-based translation module and a post-processing module for removing some tokens. This architecture has been evaluated for number transcription in English, Spanish and Romanian. For all the languages, the reached performance has been very good, specially for numbers not including decimals. When increasing the amount of data used for training the system, the results are better. Finally, it is necessary to comment that the system tuning, as the aligment of the token translator, must be adapted to the language in order to get the best results. Comparing to previous works, for example in [7], authors compare the language dependent (language specific) - rule based approach with the SMT and suggest to post-correct the results of LS-rule based by applying the SMT. This paper directly use the SMT, without any rule or language specific interventions. The system, at the end, only does minor post-corrections at a very small amount of data (eg. 0.1%). In [7], for a larger training dataset (eg. 3000 sentences) they obtain a BLEU=94,4, while in these experiments, for the smallest training set of 200 sentences, the BLEU is 95.2 (RO), 97.3 (ES) and 98.3 (EN). In [9], for SMS and Twitter messages, the BLEU is 99,2 for a larger training set, 90.000 sentences.

6. Acknowledgements

The work leading to these results has received funding from the European Union under grant agreement (n 287678). It has also been supported by TIMPANO (TIN2011-28169-C05-03), INAPRA (MICINN, DPI2010-21247-C02-02) and MA2VICMR (CAM, S2009/TIC-1542) projects. Authors also thank the other members of Simple4All project for the discussion on these topics.

7. References

[1] Sproat, R., Black, A., Chen, S., Kumar, S., Ostendorf, M., and Richards, C., 2001. *A Normalization of non-standard words.* Computer Speech and Language, 15(3), 287-333, 2001.

[2] Chang, J.T., Schtze, H., and Altman. R.B., 2002. *Creating an Online Dictionary of Abbreviations from MEDLINE. JAMIA.*

[3] Pennell D., and Liu Y., 2011a. *Toward text message normalization: Modeling abbreviation generation.* Proceedings of the IEEE. pp. 5364-5367.

[4] Sproat, R., 2010. *Lightly Supervised Learning of Text Normaliza-*

tion: Russian Number Names. IEEE Workshop on Spoken Language Technology, Berkeley, CA, 2010.

[5] Jonnalagadda and Topham. 2010. *NEMO: Extraction and normalization of organization names from PubMed affiliations*. J Biomed Discov Collab (2010) vol. 5 pp. 50-75.

[6] Ning Xia, Hongfei Lin, Zhihao Yang, Yanpeng Li, 2011 *Combining multiple disambiguation methods for gene mention normalization*. Expert Systems with Applications, Volume 38, Issue 7, July 2011, Pages 7994-7999

[7] Schlippe, T., Zhu, C., Gebhart, J., Schultz, T., 2010. *"Text Normalization based on Statistical Machine Translation and Internet User Support"*. Interspeech. 2010.

[8] Brody, S., Diakopoulos. N., 2011. *Cooooollllllll!!!!!!!! Using Word Lengthening to Detect Sentiment in Microblogs*. EMNLP'11.

[9] Han B., and Baldwin, T., 2011. *Lexical normalisation of short text messages: Makn sens a #twitter*. ACL 2011.

[10] Aw, et al. 2006. *A phrase-based statistical model for SMS text normalization*. ACL, 2006.

[11] Pennell D., Liu Y., 2011. *A Character-Level Machine Translation Approach for Normalization of SMS Abbreviations*. IJCNLP.

[12] Nouman Azam, JingTao Yao. 2012. *Comparison of term frequency and document frequency based feature selection metrics in text categorization*. Expert Systems with Applications, Volume 39, Issue 5, April 2012, Pages 4760-4768

[13] Lei Shi, Xinming Ma, Lei Xi, Qiguo Duan, Jingying Zhao. 2011. *Rough set and ensemble learning based semi-supervised algorithm for text classification* Expert Systems with Applications, Volume 38, Issue 5, May 2011, Pages 6300-6306

[14] Och J., Ney. H., 2003. *"A systematic comparison of various alignment models"*. Computational Linguistics, Vol. 29, No. 1 pp. 19-51, 2003.

[15] Koehn P., F.J. Och D. Marcu. 2003. *"Statistical Phrase-based translation"*. Human Language Technology Conference 2003 (HLT-NAACL 2003), Edmonton, Canada, pp. 127-133, May 2003.

[16] Stolcke A. 2002. *"SRILM - An Extensible Language Modelling Toolkit"*. ICSLP. 2002. Denver Colorado, USA.