Written Form Extraction of Spoken Numeric Sequences in Speech-to-Text Conversion for Ukrainian

Mykola Sazhok[1[0000-0003-1169-6851]], Valentyna Robeiko[2[0000-0003-2266-7650]], Ruslan Seliukh[1[0000-0003-2230-8746]], Dmytro Fedoryn[1[0000-0002-4924-225X]] and Oleksandr Yukhymenko[1[0000-0001-5868-8547]]

[1] International Research/Training Center for Information Technology and Systems, Kyiv, Ukraine
[2] Taras Shevchenko National University, Kyiv, Ukraine
{sazhok, valya.robeiko, vxml12, dmytro.fedoryn, enomaj}@gmail.com

Abstract. The result of automatic speech-to-text conversion is a sequence of words contained in a working dictionary. Hence each number must be added to the dictionary, which is not feasible. Therefore we need to introduce a post-processor block extracting numeric sequences by speech recognition response. We describe a sequence-to-sequence converter that is a finite state transducer initially designed to generate phoneme sequences by words for Ukrainian using the expert-specified rules. Further, we apply this model to extract numeric sequence by speech recognition response considering word sequences as well as time and speaker identity estimations for each word. Finally, we discuss experimental results and spot detected problems for further research.

Keywords: numeric sequence extraction, speech recognition post-processing, finite state transducer, rule-based conversions

1 Introduction

Human speech contains, depending on a domain, a significant amount of numeric sequences, which express cardinal numbers, time and date, addresses currency expressions and so on.

A speech-to-text system produces a sequence of items that are, typically, words contained in the system's dictionary.

The system's productivity depends on the dictionary amount. Taking more space and computational resources, a larger vocabulary induces additional hypotheses, which is a source for error increase.

If we consider each number as a valid word, this means that vocabulary expands as much as numbers might by expressed. Therefore, covering numbers between 1 and 1000000 would hypothetically mean that at least a million of words must be introduced to the vocabulary. For highly inflective languages, like Ukrainian, this amount is multiplied by the mean number of word forms. Moreover, most of these number are

unseen for the component of an ASR model constraining hypothetical word orders. So the data sparsity grows drastically.

From the other hand, quite a limited sub-vocabulary of lemmas (stems) is used to compose a spoken numeric. For Ukrainian, 20 lemmas are sufficient to compose spoken numbers from 0 to 19, nine stems are used for tens, nine stems are used to express hundreds and, finally, several lemmas name greater digit groups like thousand, million and billion. Therefore, in speech-to-text output all numbers are spelled as word sequences and finding their numeric form looks as a productive way.

The recent works aims to minimize the supervision, which varies much in dependence of the specific language [1,2,3]. The models using an end-to-end recurrent neural network are effective for English language, as an example, however, as reported, it does make errors with respect to the numeric value of the expression for highly inflective languages. Even such extremely rare cases would mislead about the message being conveyed that is completely unacceptable. The second type of models uses finite-state transducers constructed with a minimal amount of training data per inflectional form, on average, that is crucial for highly inflective languages like Ukrainian.

The referred approaches intensively exploit the number verbalization provided by a text-to-speech system and huge amount of synthesized speech as for the end-to-end model. That is what is paid to minimize the supervision, which requires huge computational resources and is not applicable to the matured and generally more productive HMM/DNN approach [4]. Instead, we retain a reasonable amount of supervision for tuning the finite-state automata based on [5] and use widely available language knowledge. This work reports the current state of the research applied to Ukrainian.

2 Selection of Hypothetical Numeric Subsequences

In general, we consider recognition response that includes, beside a word sequence, estimations for beginning and duration of each recognized word as well as speaker diarization labels. Therefore, we may avoid including into hypothetical numeric word sequences speech and speaker disruptions, since a long pause between speech segments as well as a speaker change likely cut a numeric sequence. Particularly, our assumption is that a speaker never continues pronouncing the number started by the previous speaker.

Each word is assigned with either *numeric* or *generic* or both labels. In Table 1 we can see a sequence of 10 words, (w_1, w_2, … , w_{10}), recognized in the beginning of the real news episode. The first word meaning “eighteen” starts at 15.08 s and its duration is 0.65 s as estimated by a speech-to-text converter. The second word is ambiguous and means either a number or an inflectional form of “magpie” word. As one can see, in Ukrainian, several numbers are homographs. Also among them are certain forms of words meaning two, three and five. In this work we label such words as a numeric word and include them to hypothetically numeric subsequences.

Hence, we selected two numeric word subsequences (w_1, w_2, w_3) and (w_7, w_8, w_9). From the first subsequence we intend to extract numbers, 18 and 45, whereas the second subsequence contains just one number, 2019.

Table 1. Numeric sequence extraction sample analysis.

No	Start time	Duration	Word in Ukrainian	Explanation in English	Labels	Intended output
1	15.08	0.65	вісімнадцята	eighteenth	numeric	18
2	15.74	0.19	сорок	fourty; *of magpies*	numeric; *generic*	45
3	15.95	0.31	п'ять	five	numeric	
4	16.26	0.33	факти	“Facts”	generic	факти
5	16.61	0.48	відкриває	is opening	generic	відкриває
6	17.11	0.23	новий	a new	generic	новий
7	17.34	0.18	дві	two	numeric	2019
8	17.52	0.36	тисячі	thousand	numeric	
9	17.90	0.73	дев'ятнадцятий	nineteenth	numeric	
10	18.63	0.21	рік	year	generic	рік

3 Rule-Based Sequence-to-Sequence Multidecision Conversion Model

A key issue in modeling the conversion between sequences is the question of how we define the correspondence between elements of source and target sequences. We consider a finite sequence of source elements $a_1^N=(a_1,a_2,\ldots,a_n,\ldots,a_N)$ where each element is taken from the set of input elements, *A*. Let us construct the conversion of this sequence to a set of sequences for output elements taken from *B*.

Consider an elementary correspondence *f* that maps a subsequence of a_1^N, starting from its *n*-th element, to an element from *B* set or an empty element:

$$f\left(a_n^N\right)=b, a_n^N\in \mathrm{Def}(f)\subset A, b\in B\cup\varnothing, 1\le n\le N. \quad (1)$$

Note that (1) is applicable only for the specified source sequences. Applying sequences of such functions, f_n^N, to the source subsequence a_n^N we attain a set of target subsequences:

$$\mathrm{F}\left(a_n^N\right)=\left\{\left(f_1^k\left(a_n^N\right),f_2^k\left(a_n^N\right),\ldots,f_{L_k}^k\left(a_n^N\right)\right)\in B^{L_k}\cup\varnothing, 1\leqslant k\leqslant K_F\right\}. \quad (2)$$

Here L_k is length of *k*-th target subsequence and the number of the target subsequences is K_F. Introduced correspondences (2) form $\overline{\mathrm{F}}$ set.

Now we define an operation that concatenates over the sets produced by F and G taken from $\overline{\mathrm{F}}$ as all possible combinations of target sequences generated by F followed by G:

$$\mathrm{F}\circ\mathrm{G}=\left\{\left(f_1^u,f_2^u,\ldots,f_{L_u}^u,g_1^v,g_2^v,\ldots,g_{L_v}^v\right), 1\leqslant u\leqslant K_F, 1\leqslant v\leqslant K_G\right\}. \quad (3)$$

Additionally, we assume that the connection result is empty if at least one of F and G is empty. Further, we specify ordered correspondences (2) and accomplish them with additional parameters attaining a set:

$$\widetilde{\overline{\mathrm{F}}}=\left(\mathrm{F}_{i,d_i,\delta_i}\right),\mathrm{F}\in\overline{\mathrm{F}},1\leqslant i\leqslant\left|\overline{\mathrm{F}}\right|,0<d_i,\delta_i=\{0,1\}, \tag{4}$$

where d_i we call an analysis step and δ_i is an exclusivity condition for the *i*-th correspondence. Within these parameters we construct restricted connections in form

$$\underset{i,n}{\circ}\,\mathrm{F}_{i,d_i,\delta_i}\left(a_n^N\right),1\leqslant i\leqslant\left|\overline{\mathrm{F}}\right|,1\leqslant n\leqslant N. \tag{5}$$

Assume that (5) has already been evaluated for certain index sets *J* and *M*, which are ordered, and we obtained

$$G_{J,M}=\underset{u\in J,v\in M}{\circ}F_{u,d_u,\delta_u}\left(a_v^N\right). \tag{6}$$

Let j and m be the last elements of *J* and *M* respectively. Then connecting the next correspondence, $\mathrm{F}_{i,d_i,\delta_i}\left(a_n^N\right)$, we proceed in accordance to (3), if the following conditions are met:

$$\left\{\left[\begin{array}{c} m+d_i=n; \\ \delta_r,1\leqslant r\leqslant i; \\ \underset{u\in J,v\in M}{\circ}\mathrm{F}_{u,d_u,\delta_u}\left(a_n^N\right)\circ\mathrm{F}_{r,d_r,\delta_r}\neq\varnothing,1\leqslant r\leqslant i,\text{if}\,\delta_i=1. \end{array}\right.\right. \tag{7}$$

Otherwise the connection is not applicable.

By means of expression (5) we can generate target sequences proceeding from a source sequence of elements.

We illustrate this process on the graph in Fig. 1. The sequence of Ukrainian words “сто”, “двадцять”, “три”, “тисячі” (“hundred”, “twenty”, “three”, “of thousand”) is accomplished with word sequence boundary elements “_”.

Thus we have a sequence of six elements a_1^N = (“_”, “сто”, “двадцять”, “три”, “тисячі”, “_”), N = 6. All valid correspondences for this example, $\mathrm{F}_{i,d_i,\delta_i}\left(a_n^N\right),1\leqslant n\leqslant N$, as it follows from the graph, are:

$\mathrm{F}_{1,1,0}\left(a_1^6\right)=\{(\_)\}$, $\mathrm{F}_{2,1,0}\left(a_2^6\right)=\{(100)\}$, $\mathrm{F}_{6,2,0}\left(a_2^6\right)=\{(120)\}$, $\mathrm{F}_{7,5,0}\left(a_2^6\right)=\{(123000,\_)\}$, $\mathrm{F}_{3,4,0}\left(a_3^6\right)=\{(23000,\_)\}$, $\mathrm{F}_{4,1,0}\left(a_3^6\right)=\{(20)\}$, $\mathrm{F}_{5,3,0}\left(a_4^6\right)=\{(3000,\_)\}$.

Moving alongside the arrows we generate expressions of the form (5) receiving the following hypothetical sequences of numbers: “123000”, “100 23000”, “100 20 3000” and “120 3000”.

On practice, we do not need to consider the entire subsequence a_n^N. Normally, we narrow the context to $a_n^{n-1+T_F}$, where $T_F\geqslant 1$ depends on the specific correspondence (2). In Fig. 1 hight of rectangles corresponds to the context widths.

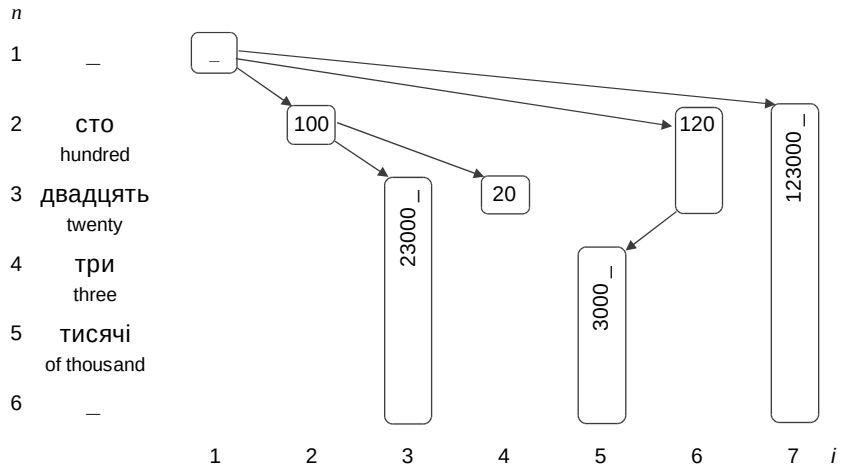

Fig. 1. Graph of multidecision conversion from a word sequence “сто”, “двадцять”, “три”, “тисячі” (“hundred”, “twenty”, “three”, “of thousand”) to hypothetical numeric sequences.

The expert can specify parameters of correspondences (4) as a template that is explained in the system description (Section 4). Note that we may apply the same or another set of correspondences to target sequences once more or multiple times. This way we introduce multiple levels for the conversion procedure. Generally, this allows for simplifying the model parameter specification and avoiding errors for the expert. Particularly, inserting zeros for skipped digits and digit groups is a bit tricky and level introduction does this work as we discuss in next section. The other benefit of introduced levels is the possibility to convert numbers, symbolic characters and abbreviations to their textual presentation within the same algorithm and to process multilingual text.

4 Numeric Sequence Extraction Structure

To extract numeric sequences from recognition response, firstly, we analyze word sequences and select all sub-sequences hypothetically containing numbers. Then we apply the conversion procedure described in Section 3 to the selected word subsequences and extract possible numeric sub-sequences. Finally, we connect all subsequences to get hypotheses of text with extracted numbers.

To select all word sub-sequences that may contain numbers we generated a list of valid numerical word forms proceeding from [6]. Then all words in recognition response [7,8] matching one of the generated word forms are marked as numeric. Thus, we obtain the input numeric word sequences each of which is split in accordance to the boundaries induced by pauses longer than a threshold and speaker alterations derived from the speaker diarization procedure [9].

The selected word sequences are processed further by the multilevel rule-based sequence-to-sequence converter as shown in Fig. 2.

At first stage, the language-dependent rules are applied to obtain language-independent symbol sequences. These symbols replaces Ukrainian words for ones (e), tens (d) and hundreds (c) and correspondences for digit groups of thousand (t), million (M), billion (B) and trillion (T). For instance, “дві тисячі двадцятий” (“two thousand twentieth”) is directly mapped to “2et2d”.

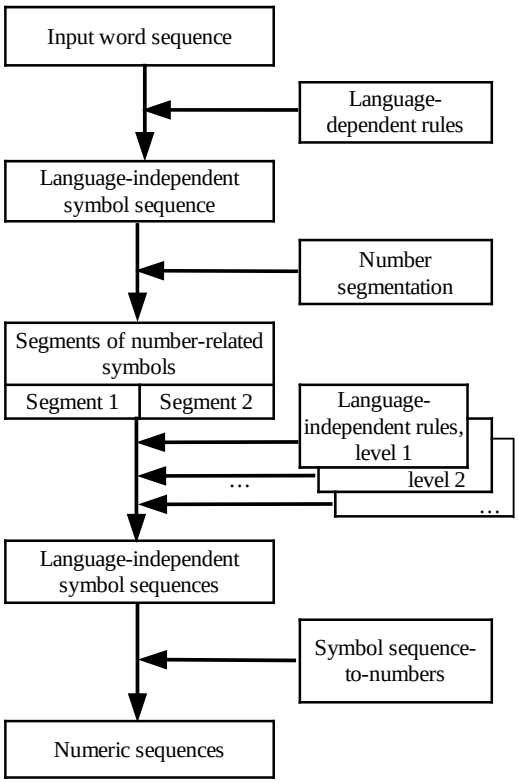

Fig. 2. Numeric word sequence conversion diagram.

At the next stage the cases when two or more numbers are pronounced in row are handled. Language-independent symbol sequences are segmented assuming that same or greater digit group starts a new number. This approach, however, might work improperly for descending numeric sequences. Fig. 3 illustrates number boundary detection for word sequence that means the time of 18:45. Three recognized words “вісімнадцята” (eighteen), “сорок” (forty) and “п’ять” (five) are mapped to language-independent symbol sequence “1d8e”, “4d” and “5e” and each word is connected to the first respective symbol with an arrow before and after segmentation on two numbers “1d8e” (18) and “4d5e” (45).

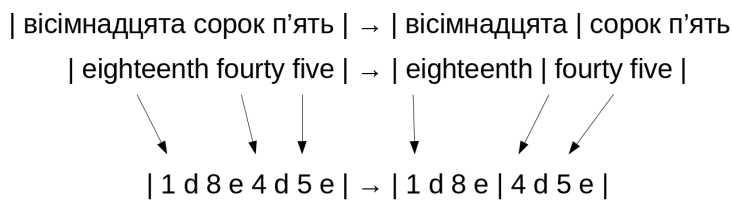

Fig. 3. Number boundary detection example for word sequence “вісімнадцята сорок п’ять” (“eighteenth forty five”).

At the last stage the extracted segments belonging to different numbers are accomplished with missing zeros and ones and, finally, cleaned from symbols that names digit positions and digit groups. So the example given in Fig. 3 does not require any mentioned accomplishments and can be mapped directly to numbers from the language-independent numeric presentation, | 1 d 8 e | 4 d 5 e | → | 18 | 45 |. In turn, if we consider the language-independent presentation of verbalized current, 2020, year, it requires inserting zeros, skipped by verbalization, at proper positions: | 2 t 2 d | → | 2 t 0 c 2 d 0 e | → | 2020 |.

5 Rule Specification

The rules are specified in text form as fields separated by tabulation, `<tab>`:

```
<source_subsequence_pattern><tab>[<condition><tab>]
<analysis_step1><target_subsequence_pattern1>
[<tab><analysis_step2><target_subsequence_pattern2> ...]
```

Here optional components of the template are shown in square brackets. The terminated with ellipsis block might be repeated with different values that will induce multiple decisions. We will refer to examples in Table 3 for rule specification illustrations.

`<source_subsequence_pattern>` consists of explicit characters as well as wildcards replacing any one symbol, `?`, and one or more symbols, `*`, like in examples 1 through 3 in Table 2. Furthermore, the expert may define a subset of characters taking them in square brackets (samples 4 and 5). Sequence elements are separated by whitespace.

The only `<condition>` is exclusivity introduced in (4). It is denoted as `-x` and used in the pattern that maps an unspecified element to itself like in example 6.

The next pairs of parameters may repeat as many alternative conversions are valid for the source subsequence.

`<analysis_stepX>` value stays for the analysis step introduced in (4), and `<target_subsequence_patternX>` explains how to generate a target subsequence. The wildcard `?`, as in samples 4 and 5, stay instead the actually matching character, i.e., for sample 5, matching to the source pattern “`M 2 c |`” will be mapped to “`M 2 c 0 d 0 e |`”.

Table 3. Examples for the rule specification.

No	Source subsequence	Step	Target subsequence	Explanation
1	**тисяч***	1	**t**	words starting with “тисяч”, a form of “thousand”, excluding “тисяч” itself → **t**
2	**п’ятис***	1	**5c**	words starting with “п’ятис”, a form of “five hundred” → **5c**
3	**дванадцят***	1	**1d2e**	words starting with “дванадцят”, a form of “twelve” → **1d2e**
4	**[TBM|]** t	1	**?1e**	inserting a one, **1**, before a thousand that was not pronounced
5	**[TBMt|]** [0123456789] **c [TBMt|]**	3	**??c0d0e**	zeroing skipped running pair **d** and **e**: inserts **0d0e** after **c** if **c** is followed by the element denoting a digit group boundary
6	*	1	*	any non-empty source element is mapped to itself (if no rules have been applied)

6 Implementation

The module that provides word-to-number extraction in accordance to section 4 is written in Perl and derived from the implementation of bidirectional text-to-pronunciation conversion [5]. The rules are specified as described in Section 5, one level per file. The file that corresponds to the next level is indicated in header.

The basic implementation is deployed online [10] and may be tested alongside with other rule-based sequence-to-sequence conversions.

For experiments, the data is read and written in time-marked conversations (ctm-file) format. In Table 3 the aligned input and output lines are presented for a real broadcast transcript leveraged by means of automatic speech recognition for Ukrainian broadcast media transcribing system [7]. The numbers, indicated with bold, are extracted as expected. A speech-to-text system produces a sequence of items that are, typically, words contained in the system’s dictionary.

7 Conclusions

The described multilevel rule-based system allows for generating hypotheses of word-to-number conversion. Best hypothesis selection is the subject of analysis of lexical, syntactic and prosodic contexts by large corpora.

To introduce a new language an expert just need to fill the language-dependent rules mapping to a language-independent number spelling presentation as illustrated in Table 2, rows 1 trough 3. This way a multilingual content might be introduced as well.

Further modeling will include appending a suffix for ordinal numbers and extraction of fractions, compound words (like “20-year-old”), time, sport scores and other numerical types.

Table 3. Input and output ctm-file comparison.

Input sequence				Output sequence		
Start	Duration	Word	Explanation in English	Start	Duration	Word
15.08	0.65	Вісімнадцята	Eighteenth	15.08	0.65	**18**
15.74	0.19	сорок	fourty	15.74	0.52	**45**
15.95	0.31	п'ять	five			
16.26	0.33	факти	“Facts”	16.26	0.33	факти
16.61	0.48	відкриває	is opening	16.61	0.48	відкриває
17.11	0.23	новий	a new	17.11	0.23	новий
17.34	0.18	дві	two	17.34	1.29	**2019**
17.52	0.36	тисячі	of thousand			
17.90	0.73	дев'ятнадцятий	nineteenth			
18.63	0.21	рік.	year.	18.63	0.21	рік.
18.84	0.45	Наступні	Next	18.84	0.45	Наступні
19.29	0.51	півгодини	half hours	19.29	0.51	півгодини
19.80	0.15	про	about	19.80	0.15	про
19.95	0.60	найголовніші	most important	19.95	0.60	найголовніші
20.55	0.33	події	events	20.55	0.33	події
20.88	0.33	другого	of second	20.88	0.33	**2**
21.21	0.42	січня	January	21.21	0.42	січня

References

1. Gorman, K., Sproat R.: Minimally supervised number normalization. Transactions of the Association for Computational Linguistics 4, 507–519 (2016).
2. He Y. et al.: Streaming End-to-end Speech Recognition for Mobile Devices. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 6381-6385. Brighton, United Kingdom (2019).
3. Peyser, C., Zhang, H., Sainath, T.N., Wu, Z.: Improving Performance of End-to-End ASR on Numeric Sequences. In: Interspeech 2019 Proceedings, pp. 2185-2189 (2019).
4. Hinton, G., Deng, L., Yu, D., Dahl, G. et al.: Deep Neural Networks for Acoustic Modeling in Speech Recognition. Signal Processing Magazine, IEEE 6 (29), 82–97 (2012).
5. Robeiko, V., Sazhok, M.: Bidirectional Text-To-Pronunciation Conversion with Word Stress Prediction for Ukrainian. In: 11th All-Ukrainian International Conference on Signal/Image Processing and Pattern Recognition UkrObraz’2012, pp. 43-46. UAsIPPR, Kyiv, Ukraine (2012).

6. Shirokov, V., Manako V.: Organization of resources for the national dictionary base. Movoznavstvo 5, 3–13 (2001).
7. Sazhok, M., Selyukh, R., Fedoryn, D., Yukhymenko, O., Robeiko V.: Automatic speech recognition for Ukrainian broadcast media transcribing. Control Systems and Computers 6 (264),p. 46-57 (2019).
8. Povey, D., Ghoshal, A., Boulianne, G. et al.: The Kaldi Speech Recognition Toolkit. In: IEEE 2011 Workshop on Automatic Speech Recognition and Understanding (2011).
9. Zewoudie, A., Luque, J., Hernando, J.: The use of long-term features for GMM- and i-vector-based speaker diarization systems. EURASIP Journal on Audio, Speech, and Music Processing, 14 (2018).
10. Bidirectional text-to-pronunciation conversion tool, www.cybermova.com/labs, last access 2020/02/20.
11. Allauzen, C., Riley, M., Schalkwyk, J., Skut, W., Mohri, M.: OpenFst: A General and Efficient Weighted Finite-State Transducer Library. In: Holub, J., Žďárek, J. (eds) Implementation and Application of Automata. CIAA 2007. Lecture Notes in Computer Science, vol 4783. Springer, Berlin, Heidelberg (2007).