PitchKeywordExtractor: Prosody-based Automatic Keyword Extraction for Speech Content

Iurii Lezhenin∗, Artyom Zhuikov∗, Natalia Bogach∗, Elena Boitsova†, Evgeny Pyshkin‡

∗Institute of Computer Science and Technology Peter the Great St. Petersburg Polytechnic University
194021 St. Petersburg Polytechnicheskaya, 21 Email: bogach@kspt.icc.spbstu.ru
†Institute of Humanities Peter the Great St. Petersburg Polytechnic University
194021 St. Petersburg Polytechnicheskaya, 19 Email: el-boitsova@yandex.ru
‡Software Engineering Lab. University of Aizu
Aizu-Wakamatsu, 965-8580, Japan Email: pyshe@u-aizu.ac.jp

Abstract—Keyword extraction is widely used for information indexing, compressing, summarizing, etc. Existing keyword extraction techniques apply various text-based algorithms and metrics to locate the keywords. At the same time, some types of audio and audiovisual content, e.g. lectures, talks, interviews and other speech-oriented information, allow to perform keyword search by prosodic accents made by a speaker. This paper presents PitchKeywordExtractor — an algorithm with its software prototype for prosody-based automatic keyword extraction in speech content. It operates together with a third-party automatic speech recognition system, handles speech prosody by a pitch detection algorithm and locates the keywords using pitch contour cross-correlation with four tone units taken from D. Brazil discourse intonation model.

I. INTRODUCTION

KEYWORDS make the semantic backbone of a text. As keywords reflect the text ideas and convey text meaning they are used for text indexing, analysis, summarizing compression, etc. [1]. In modern world of on-line information abundance automatic keyword extraction techniques are extremely in-demand ([2], [3]).

There is a great number of research in the area of automatic keyword extraction either for individual documents e.g. [4], [5], or large document corpora [6], as well as for specific types of on-line content like e-newspapers [7] or micro-blogs on Twitter [8]. Content-based retrieval research [9] is also highly relied upon the keywords [10].

Some of these techniques use document corpora, while others do not. When a document corpus is used, a function which balances a measure of a keyword within a document (frequency, location or co-occurrence) with a similar measure from the corpus is applied. When corpus is unavailable, keyword extraction techniques use lexical or semantic analysis or keywords co-occurrences over an individual document. An excellent literature review on automatic keyword extraction techniques is presented in [11]. Automatic keyword extraction techniques for text compression and summarizing can be found in [3].

In comparison with text processing techniques specific audio and audio-visual speech content keyword extraction algorithms are less developed. Meladianos et al. [12] report on a high demand for speech processing from the point of view of information mining. The actual research in this area is usually based on a preliminary audio-to-text conversion by means of automatic speech recognition system (ASR) and further application of content-sensitive text-based techniques (e.g. see Elakiya K. et al. [13] or G. Alharbi [14]).

At the same time, speech content has an inherent powerful feature, namely, speech prosody (i.e. intonation, rhythm, tempo, pausing, etc.) that can help to locate and extract keywords. We use the term "prosody" exactly in the sense of D. Brazil system of discourse intonation (DI) [15], [16], [17] and refer to his tone units to define the prosodic patterns for PitchKeywordExtractor. The working hypothesis of the present research is based on concept that keywords being the most informative parts of speech are prosodically highlighted by a speaker, and, therefore, they must have specific discernible prosodic characteristics.

Speech prosody is observable by measuring the fundamental frequency (pitch) and there exist a variety of speech processing tools e.g. see Praat or Visi-Pitch or TarsosDSP [18] to analyse prosodic characteristics as per pitch detection and estimation algorithms [19], [20]. A perfect guideline for special software operation can be found in [21].

There have been much research, discussion and criticism on prosody-based methods applicability and limits. Now they go far beyond simple pitch measurement and exist as components for complex analytic frameworks: e.g. see P. Roach [22] or A. Meftah et al. in [23] for prosody-based systems of emotion recognition. A deep insight into the contribution of prosody-based techniques to corpus linguistics was made by M. Warren [24].

On the assumption that automatic keyword extraction can benefit from prosody-based analysis we propose to add processing of prosodic features to automatic keyword extraction algorithms as far as speech content is concerned. We present PitchKeywordExtractor — a prosody-based tool for automatic keyword extraction. Operating together with a third-party ASR and speech processing software PitchKeywordExtractor searches for keywords in speech content by matching their prosodic characteristics to ASR output text.
II. METHODOLOGY AND MATERIALS

It is widely recognized that keywords in speech have not only statistically measurable features or occupy a certain sentence position, but are usually highlighted by intonation because they frequently act as speech signals for given and new information [25]. The way to make this tonal emphasis may be different depending upon the context and background of the speakers. For our analysis we have taken 4 tones from the tonal model of discourse intonation developed by D. Brazil [16] which is widely used in linguistics to describe the semantic aspect of speech prosody. This model comprises 5 principle tones of English speech: fall, rise, fall-rise, rise-fall and level. D. Brazil also defines the speech situations when each of these tones occurs.

Fall tone (p-tone) and rise-fall tone (p+-tone) are defined by Brazil as proclaiming tones, so they are used to mark new information introduced by a speaker, therefore, these tones may indicate the keywords entries. Among those the rise-fall tone is defined as "dominant proclaiming" and it highlights not only new, but important information, so it can be a strong keyword entry marker.

At the same time, fall-rise and rise tones are "referring", r-tone and r+-tone respectively. In speech they mark the already known information, i.e. the common ground of the speakers. These tones may also indicate keyword entries.

We consider four model tone units (fall, rise, fall-rise, rise-fall) to be searched for in speech. Strictly speaking, Brazil tones describe phrasal intonation and refer not to one word but to a whole semantic unit, i.e. a syntagm. A tone pattern has complex structure, namely, a pre-head, head, nucleus and tail and refers to a part of a phrase (or to a whole phase, if it is short); while a keyword can be marked by the nucleus only. However, the entire tone pattern can be located more accurately with correlation, while nucleus is too short to provide a good correlation peak. Thus, we are looking for keywords inside a phrasal tone pattern provided corresponding phrase pitch contour is obtained, compare to model tone units and map it to ASR output to retrieve the keywords.

The architecture of PitchKeywordExtractor consists of 4 main parts (see Fig.1): 1) Pitch Detector 2) Tone Unit Detector 3) Speech Recognizer 4) Segment-to-word Mapper

A. Pitch Detector

Pitch Detector obtains pitch series $s[k]$ for a given speech record. We use a third-party YIN [26] pitch detection algorithm provided with TarsosDSP [18].

B. Tone Unit Detector

Pitch series $s[k]$ are subsequently processed by Tone Unit Detection Algorithm (see Sec. III for details). The output of Tone Unit Detector is a set of segments (time intervals) where model tone units were found.

C. Speech Recognizer

Speech Recognizer produces text for a given speech record to create the reference wordlist. Sphinx [27] is used in PitchKeywordExtractor prototype by now, while this block may be implemented with any alternative solution for speech recognition.

D. Segment-to-word Mapper

The segments received from Tone Unit Detector and Speech Recognizer output file are mapped to each other to locate a word within a segment (see Sec.IV for details). Segment-to-word Mapper output is the final keyword list.

III. TONE UNIT DETECTION ALGORITHM

Tone unit detection is based on the correspondence of a syntagm pitch contour and one or more model tone units.

A. Preliminary Assumptions

Tone unit detection is performed on evenly distributed pitch series $s[k]$ obtained as per pitch detection algorithm.
Let us define 4 discrete-time limited basic functions: $\phi^f_w(x), \phi^r_w(x), \phi^{rf}_w(x), \phi^w(x)$ of $w$ length, $w \in [w_{\text{min}}, w_{\text{max}}]$, where $w_{\text{min}}, w_{\text{max}}$ are the empirically chosen syntagm boundaries; $x \in \mathbb{Z}, 0 \leq x \leq w$. These functions correspond to Brazilian tone model (see Fig. 2) as follows:

- $\phi^f_w(x)$ - “fall tone” p-tone
- $\phi^r_w(x)$ - “rise-fall” p+tone
- $\phi^{rf}_w(x)$ - “fall-rise” r-tone
- $\phi^w(x)$ - "rise" r+-tone

**B. Pre-Processing**

1) Median filtering [28] is applied to remove single prominences in $s[k]$.
2) $s[k]$ is divided into the datasets $\{s_j[k]\}$, bounded by natural pauses in speech (silence).
3) Too short datasets $\{s_j[k]\}$ are not processed as statistically inconsistent.

**C. Processing**

The following Algorithm 1 is subsequently applied to all datasets $\{s_j[k]\}$ and all model tone units $\phi_w(x)$. Values of correlation coefficient $r_\phi(k, w) \in [-1, 1]$ are used to estimate the similarity between the model tone unit $\phi_w$ and the pitch contour of a segment, which starts at $k$ and ends at $k + w$. $r_\phi(k, w)$ is calculated only for full-size segments, i.e., $k$ varies in the range of $[0, K_j - w]$ that discards the edge issues. Eq.1 shows Algorithm 1 output.

$$
\begin{align*}
\begin{array}{ccc}
\phi_0(k, w_{\text{min}}) & \ldots & \phi_0(k - w_{\text{min}}, w_{\text{min}}) \\
\phi_0(k, w_{\text{min}} + 1) & \ldots & \phi_0(k - (w_{\text{min}} + 1), w_{\text{min}} + 1) \\
& \vdots & \vdots \\
\phi_0(k, w_{\text{max}} - 1) & \ldots & \phi_0(k - (w_{\text{max}} - 1), w_{\text{max}} - 1) \\
\phi_0(k, w_{\text{max}}) & \ldots & \phi_0(k - w_{\text{max}}, w_{\text{max}})
\end{array}
\end{align*}
$$

Thus, each segment is defined by $k$, $w$, $\phi_w$, and $r_\phi(k, w)$.

**D. Post-Processing**

Post-Processing (see Algorithm 2) is applied to all the segments and comprises 4 steps (see Fig. 3):

1) Tresholding
2) Resolving horizontal segment overlap for different $k$ at fixed $w$
3) Resolving vertical segment overlap for different $w$
4) Resolving tone unit collision

The first three steps are applied to each group of segments referring to one tone unit $\phi_w$, while the last step is applied only to segments where several tone units were found.

Tresholding checks the statistical significance of correlation. Tresholding parameter $Q_{\text{Treshold}}$ sets the significance level, e.g., 0.95 or 0.98.

To locate model tone unit accurately $k$ takes all the integer values in $[0, K_j - 1]$. For two neighbour values $k_1$, $k_2$ the corresponding $r_\phi(k_1, w)$, $r_\phi(k_1, w)$ will be very close to each other, because they are calculated over almost identical datasets leading to a significant redundancy of the output data. We call this issue "horizontal overlap". It is resolved now by keeping the only one segment with the largest $r_\phi(k, w)$.
Algorithm 2 Tone Unit Detection (Post-processing)

1: for all \( w_{\min} \leq w \leq w_{\max}, w \in \mathbb{Z} \) do
2:   for all \( 0 \leq k \leq K_j - w \) do
3:     if \( r_\phi(k, w) \geq Q_{\text{thresh}} \) then
4:       CREATE_SEGMENT(tone, k, w, \( r_\phi \))
5:   end if
6: end for
7: end for
8: for all SEGMENTS do
9:   RESOLVE_HOR_OVERLAP(SEGMENT)
10: end for
11: for all SEGMENTS do
12:   RESOLVE_VERT_OVERLAP(SEGMENT)
13: end for
14: for all SEGMENTS do
15:   PRIORITIZE(SEGMENT)
16: end for

for further processing among all the overlapping segments for given \( w \), which are discarded.

"Vertical overlap", i.e., the overlap of segments with different \( k \) and \( w \), is also possible. It is resolved in exactly the same manner. Again, only the segment with the largest \( r_\phi(k, w) \) is kept for further processing.

The last step processes tone unit collision, i.e., the overlapping segments which correspond to different model tone units. In this case, the priority is given to "complex" units (p+ and r).

IV. SEGMENT-TO-WORD MAPPER

Keyword search is performed by Segment-to-word Mapper, which operates with an ASR output text labelled with the timestamps and Tone Unit Detector output file containing the segments. The goal of Segment-to-word Mapper is to find a word that was pronounced during the given segment; this word is deemed to be a keyword. Partial coincidence between segments and word timestamps is allowed and can be set in Algorithm 3 by ratio parameter. Fig. 4 illustrates a fragment of Segment-to-word mapping results achieved for the online lecture The Great Reversal: The "Rise of Japan" and the "Fall of China" after 1895 as Historical Fables delivered by Benjamin Elman from Harvard University’s Fairbank Center for Chinese Studies. Table I shows 35 keywords marked with proclaiming tones (fall and rise-fall) found by Segment-to-word Mapper in a 2-minute piece of lecture. The keywords are sorted in the same order as they are mentioned in the text; keywords given in boldface refer to Fig.4.

V. RESULTS AND DISCUSSION

To summarize, an algorithm to process ASR output text for keywords by their prosodic features is presented. The first prototype has custom Tone Unit Detector and Segment-to-word Mapper, it also operates with pitch detection and speech recognition performed by third-party tools. As the result, a list of possible keywords is generated.

For our experiments we used a number of audio samples including academic lectures, presentation talks and news recordings. A particularly interesting case is the online lecture of B. Elman mentioned in Section IV and used for segment-to-word mapping evaluation. This use case refers to (not very common but still possible) situations when audio tracks are available with no explicit metadata describing the substance and the internal content of the recorded material. This, apart from obvious applications of the proposed algorithm and related tools, we can also consider solving a problem of mapping the processed recordings to a variety of external resources such as online encyclopedias, historical books, geographical maps, etc. In such a case the process of audio playback (together with prosody-based keyword extraction performed in background) can be enhanced by delivering additional visual and text information retrieved with using the extracted keywords.

ACKNOWLEDGMENT

This work is partially supported by the grant 17K00509 of Japan Society for the Promotion of Science (JSPS).

Authors would like to thank Karina Vylegzhanina, for if it was not for her we would not have taken up this research. Our discussions and work together have greatly influenced this paper.

REFERENCES
