RETRIEVAL OF SONG LYRICS FROM SUNG QUERIES

Anna M. Kruspe

Fraunhofer IDMT
Ilmenau, Germany

Masataka Goto

National Institute of Advanced Industrial Science and Technology (AIST)
Tsukuba, Japan

ABSTRACT

Retrieving the lyrics of a sung recording from a database of text documents is a research topic that has not received much attention so far. Such a retrieval system has many practical applications, e.g. for karaoke applications or for indexing large song databases by their lyric content. We present a new method for lyrics retrieval. An acoustic model trained on singing is used to obtain phoneme probabilities from sung queries, which are then mapped to phoneme sequences. These are compared against lines of textual lyrics in a large corpus in order to retrieve the best-matching song.
The approach is tested on three sung datasets. Lyrics are retrieved from a set of 300 possible songs (12,000 lines of lyrics). The results are highly encouraging and could be used further to perform automatic lyrics alignment and keyword spotting for large databases of songs, or for retrieving lyrics from the internet.

Index Terms— Lyrics, Text retrieval, Singing, Automatic Speech Recognition, Music Information Retrieval

1. INTRODUCTION

Automatic Speech Recognition for singing is a topic that has started to receive more and more attention [1]. The research so far shows that most tasks are notoriously harder than on speech [2]. The reason for this is a multitude of differences between speech and singing, with most characteristics being much more varied in singing than in speech. Examples include pitch range, phoneme durations, pronunciation variants, semantic content, and many more.
Tasks like phoneme recognition, keyword spotting, or lyrics transcription therefore only achieve relatively low results so far [3, 4]. But there is one factor that could be beneficial to all of these tasks: The wide availability of textual lyrics on the internet. In contrast with the mentioned tasks, automatic alignment of lyrics to singing has already produced satisfactory results [2, 5]. Therefore, if the lyrics of a song can be found and then aligned, many other applications could profit.
In this paper, we present an approach to the task of automatically retrieving the lyrics for a sung recording from a corpus of known textual lyrics. In order to do this, we first generate phoneme posteriorgrams using an acoustic model trained on singing, and then map the results to symbolic representations (= phoneme sequences). These sequences are then compared against a database of lyrics using a modified Levenshtein distance.

2. STATE OF THE ART

Examples of research targeting singing without musical accompaniment include an automatic music-transcription system with lyrics recognition for a solo singing voice [6] , a robust speech modeling method applicable to even high-pitch signals such as those in singing [7], query-by-humming music retrieval using not only pitch (melody) but also lyrics [8], and a method that exploits the fact that the lyrics of popular songs are often repeated in a chorus [9]. However, research on singing that includes musical accompaniment is rare, though research has been done on the recognition of phonemes in singing under the limited conditions of known phoneme boundaries [10]. Additionally, while not strictly lyrics recognition, research has been reported on identifying the sung language [11] [12]. One topic that has received a lot of attention is the automatic alignment of lyrics to audio [13–22].
On the task of lyrics retrieval, Hosoya et al. developed a system that employs monophone HMMs trained on read speech for acoustic modeling (2005) [23]. These models are adapted to singing voices using the Maximum Likelihood Linear Regression (MLLR) technique [24]. Language modeling is performed with a Finite State Automaton (FSA) specific to the Japanese language. On a dataset of 238 children's songs, they obtain a recognition rate of .86 for the Top 1 result, and of .91 for the Top 10 results. In [25] and [8], more experiments are conducted. As a starting point, the number of words in the queries is fixed at 5, resulting in a recognition rate of .9 (Top 1 result). Then, a melody recognition is used to verify the matches proposed by the speech reognition step, raising the recognition rate to .93. The influence of the number of words in the query is also evaluated, confirming that retrieval becomes easier the longer the query is. However, even at a length of just three words, the recognition rate is .87 (vs. .81 without melody verification).
Similarly, Wang et al. presented a query-by-singing system in 2010 [26]. The difference here is that melody and lyrics

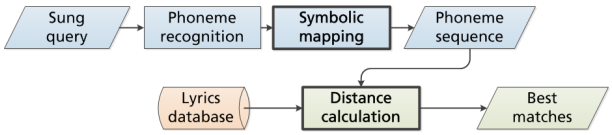


Fig. 1: Overview of the lyrics retrieval process.

information are weighted equally in the distance calculation. Lyrics are recognized here with a bigram HMM model trained on speech. The results are interpreted as syllables. A syllable similarity matrix is employed for calculating phoneme variety in the query, which is used for singing vs. humming discrimination. Assuming that only the beginning of each song is used as the starting point for queries, the first 30 syllables of each song are transformed into an FSM language model and used for scoring queries against each song in the database. The algorithm is tested on a database of 2154 Mandarin-language songs, of which 23 were annotated and the remainder are used as "noise" songs. For the Top 1 result, a recognition rate of .91 is achieved for the system combining melody and lyrics information, compared to .88 for the melody-only system.

Fujihara et al. presented a method that employs keyword spotting in singing to link text lyrics to sung lyrics [27]. Their approach employs a phoneme recognition step first, which is again based on the vocal re-synthesis method first described in [28]. Viterbi decoding using keyword-filler HMM is performed on the output of a phoneme model to detect candidate segments where keywords may occur. The link success rate for detecting the keywords is .3.

In 2009, Mesaros et al. picked Hosoya's approach back up by using MFCC features and GMM-HMMs for acoustic modeling, and also employing various MLLR techniques for adapting these models to singing [29]. These models are trained on the CMU ARCTIC speech corpus[1]. In [30] and [31], language modeling is added to the presented approach. Phoneme-level language models are trained on the CMU ARCTIC corpus as unigrams, bigrams, and trigrams, while word-level bigram and trigram models are trained on actual song lyrics in order to match the application case. The output from the acoustic models is then refined using these language models, and then also used for lyrics retrieval. This is the only purely lyrics-based system in literature. Retrieval is performed by recognizing words in queries with the full system, including language modeling, and then ranking each lyrics segment by the number of matching words (bag-of-words approach). The lyrics database is constructed from 149 song segments (between 9 and 40 seconds in the corresponding recordings). Out of these segments, recordings of 49 are used as queries to test the system. The Top 1 recognition rate is .57 (.71 for the Top 10).

[1] http://festvox.org/cmu_arctic/

3. DATA

In this work, the *DAMP* data set, which is freely available from Stanford University[2] [32], was used. This data set contains more than 34,000 recordings of amateur singing of full songs with no background music, which were obtained from the *Smule Sing!* karaoke app. Each performance is labeled with metadata such as the gender of the singer, the region of origin, the song title, etc. The singers performed 300 English language pop songs. The recordings have good sound quality with (usually) little background noise, but come from a lot of different recording conditions. We randomly selected 20 recordings per song to balance the data, resulting in a corpus of about 6000 songs, both male and female.

No lyrics annotations are available for this data set, but we obtained the textual lyrics from the *Smule Sing!* website[3]. These were, however, not aligned in any way. We performed such an alignment on the word and phoneme levels automatically using Viterbi alignment with models trained on speech (the *TIMIT* [33] corpus). This approach is described in [34]. This corpus of text lyrics is also used to test our approach. In a pre-processing step, each set of song lyrics was split into lines and converted into its constituting phonemes using the CMU Pronouncing Dictionary[4] with some manual additions of unusual words. This dictionary has a phoneme set of 39 phonemes. Splitting the lyrics for these 300 songs into lines results in a database of around 12,000 phoneme sequences.

For testing, we selected 20 female and 20 male song snippets corresponding to one line of lyrics (referred to as "Female" and "Male"). This selection was done based on the number of phonemes in the snippet and the clarity of pronunciation (in the sense that incomprehensible recordings were not selected). These test snippets were not included in the large training data set previously described.

In addition to this, the author performed 90 sung queries of lyrics included in the corpus, recorded with a mobile device (referred to as "Author").

4. PROPOSED APPROACH

For our new approach, we considered ways to improve upon the DTW-based method described in section 2. As shown in figure 1, we still start by extracting phoneme posteriorgrams. Then, we attempt to compress them down to a plausible sequence of phonemes, which can then be used to search directly on a textual lyrics database. Text comparison is much cheaper than the previous comparison strategy, and enables us to quickly expand the lyrics database.

Phoneme recognition is performed with Deep Neural Networks trained on the *DAMP* data set; the whole process is described in detail in [34]. The lyrics database is prepared by converting it into phoneme sequences as described in sec-

[2] https://ccrma.stanford.edu/damp/

[3] http://www.smule.com/songs

[4] http://www.speech.cs.cmu.edu/cgi-bin/cmudict

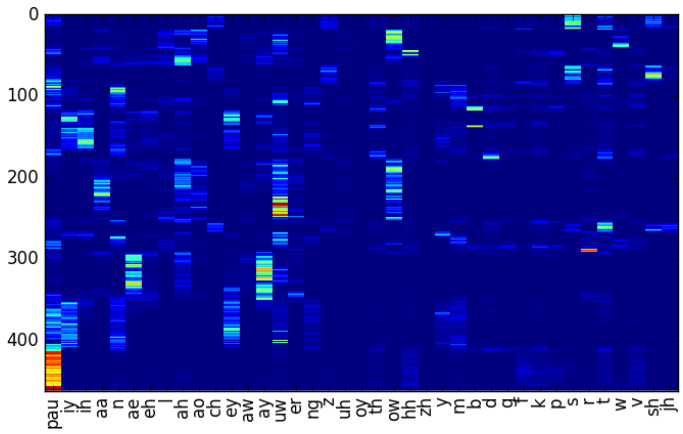
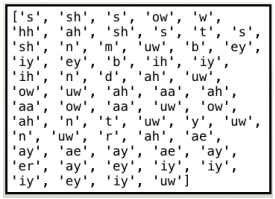
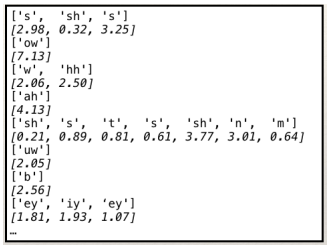


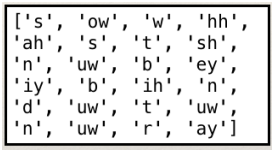
Fig. 2: Example of a phoneme posteriorgram.



(a) Original phoneme sequence



(b) Grouped into blocks (with probabilities in italics)



(c) Filtered result

Fig. 3: Example of the block grouping of the phoneme sequence and subsequent filtering by probabilities and confusions.

tion 3. The key algorithms are (a) how to generate plausible phoneme sequences from the posteriorgrams, and (b) how to compare these against the sequences in the database. These parts will be described in more detail in the following.

4.1. Symbolic mapping

Starting from the sung recording used as a query, MFCC features are extracted and run through the acoustic model trained on singing to recognize the contained phonemes. This produces a phoneme posteriorgram, such as the one shown in figure 2; i.e., probabilities of each phone over each time frame. These probabilities contain some noise, both due to inaccuracies in the model and due to ambiguities or actual noise in the performance. Obtaining a phoneme sequence from these posteriorgrams could be solved with a HMM approach, but there is not enough reliable training data. Instead, we undertake the following steps, which also allows us to take linguistic knowledge into account:

Smoothing We first smooth the posteriorgram along the time axis with a window of length 3 in order to remove small blips in the phoneme posteriorgrams.

Maximum selection and grouping Then, we select the maximum bin (i.e. phoneme) per frame and group consecutive results. This yields a first sequence of phonemes, each with duration and sum probability information, which is usually too long and noisy. In particular, we notice a lot of fluctuations between similar phonemes. An example is given in figure 3a.

Grouping by blocks and filtering through confusion matrix We attempt to solve this problem by first grouping the detected phonemes into blocks, in this case vowel and consonant blocks (shown in figure 3b). Then, we need to figure out which elements of these blocks are the "true" phonemes and which ones are noise. This is done by taking each phoneme's probability as well as the confusion between phonemes into account. The confusion is calculated in advance by running the classifier on an annotated test set; the result covers both the confusion by inaccuracies in the classifier as well as perceptual or performance-based confusions (e.g. transforming a long *[ay]* sound into *[aa - ay - iy]* during singing). An example is shown in figure 4. The product of the probabilities and the confusions are calculated for the highest combinations up to a certain threshold, and all other detected phonemes are discarded. This results in a shorter, more plausible phoneme sequence (figure 3c).

4.2. Distance calculation

We then need to calculate the distances between the extracted phoneme sequence and the ones provided in the lyrics database.

We first introduce an optional step to speed up the process. We count each sequence's number of vowels in advance, and do the same for the query sequence. Then, we only compare it to sequences with roughly the same amount of vowels (with some tolerance). This slightly decreases accuracies, but drastically speeds up the calculation time.

The similarity calculation itself is implemented with a modified Levenshtein distance. We again take into account the classifier's confusion between phonemes. These confusions are used as the Levenshtein weights for substitutions. Surprisingly, we discovered that also using them for insertion weights improves the results as well. This probably happens because of the effect described above: A singer will in many cases vocalize an phoneme as multiple different ones, particularly for vowels with long durations. This will result in an insertion in the detected phoneme sequence, which is not necessarily a "wrong" classification by the acoustic model, but does not correspond to the expected sequence for this line of lyrics. For this reason, such insertions should not be harshly penalized. For deletions, the weight is set to .5 to balance out the lower insertion and substitution weights.

	Female			Male			Author		
	Top1	Top3	Top10	Top1	Top3	Top10	Top1	Top3	Top10
Baseline	.25	.25	.35	.1	.1	.2	.1	.23	.36
Posteriorgram smoothing	.65	.7	.90	.5	.65	.65	.61	.67	.76
Filtering blocks by probabilities and confusions	.8	.85	.96	.75	.8	.85	.74	.78	.82
Substitution weights	.9	.95	.95	.65	.8	.9	.76	.78	.83
Substitution + Insertion weights	1	1	1	.75	.85	.9	.81	.84	.9

Table 1: Results of the retrieval algorithm for three test data sets with various improvement steps.

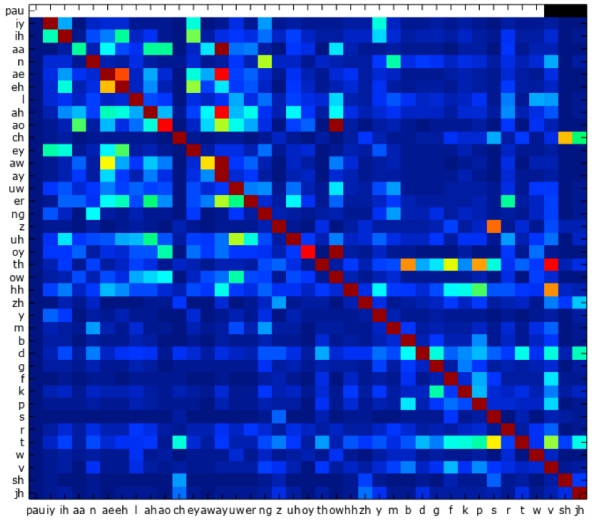


Fig. 4: Example of a confusion matrix for an acoustic model. (*[pau]* confusions are set to 0 to to avoid influence by pauses).

5. EXPERIMENTS AND RESULTS

Table 1 shows an overview over our experimental results. We tested the approach on the three test sets described in section 3 and report the retrieval rates for the Top 1 result (i.e. the one with the lowest Levenshtein distance), the Top 3, and the Top 10 results. Queries were allowed to be one to three consecutive lines of songs, increasing the number of "virtual" database entries to around 36,000 (12,000 lines as 1-, 2-, and 3-grams). It should be noted that a result counts as correct when the correct song (out of 300) was detected; this was done as a simplification because the same line of lyrics is often repeated in songs. For possible applications, users are most probably interested in obtaining the correct full song lyrics, rather than a specific line. Picking random results would therefore result in a retrieval rate of .003.

We then tested the various improvement steps of the algorithm as follows:

Baseline As a baseline system, we used the most straightforward approach: Directly pick the phonemes with the highest probabilities for each frame from the posteriorgram, group them by consecutive phonemes, and use the result of that for searching the lyrics database with a standard Levenshtein implementation. This results in retrieval rates of .25, .1, and .23 for the Female, Male, and Author test sets respectively. (The female test set generally produces higher results because the performance quality is somewhat better).

Posteriorgram smoothing This is the same as the baseline approach, but we smooth the posteriorgram along the time axis as described in section 4.1. This already improves the result by around .4 for each test set.

Filtering blocks by probabilities and confusions This includes the last step described in section 4.1. The result is improved further by .13 to .25.

Substitution and insertion weights Finally, we use the modified Levenshtein distance as described in section 4.2. When using the confusion weights for phoneme substitutions only, the result increases further, and even more so when they are also used for the insertion weights. The final Top 1 retrieval rates are 1, .75, and .81 for the three test sets respectively.

6. CONCLUSION

In this paper, we presented a new approach for retrieving textual lyrics from single lines of sung queries. To this end, we first extract phoneme posteriorgrams from the audio recording with acoustic models trained on a large singing database. Then, we employ a new algorithm to map the result to a symbolic representation (i.e. a phoneme sequence). This sequence is then compared to the ones in the lyrics database using a modified Levenshtein distance.

We found several improvements to this basic algorithm. First, we noticed that smoothing the posteriorgram along the time axis with a short window removes spurious probability fluctuations. The mapping from the posteriorgram to the symbolic representation can be improved by filtering the phonemes with the highest probabilities with the known phoneme confusions. Finally, we modified the Levenshtein distance by also utilizing these confusions for substitution and insertion weights. The final results range between .75 and 1 for the Top 1 result, and between .9 and 1 for the Top 10.

7. ACKNOWLEDGEMENTS

This work was supported by JST ACCEL Grant Number JPMJAC1602, Japan, and the German Research Foundation (DFG MU 2686/11-1, DFG AB 675/2-1).

8. REFERENCES

[1] M. Goto, "Singing information processing," in *ICSP*, 2014.

[2] H. Fujihara and M. Goto, *Multimodal Music Processing*, chapter Lyrics-to-audio alignment and its applications, Dagstuhl Follow-Ups, 2012.

[3] A. Mesaros and T. Virtanen, "Automatic recognition of lyrics in singing," *EURASIP Journal of Audio, Speech and Music Processing*, 2010.

[4] A. M. Kruspe, "Training phoneme models for singing with "songified" speech data," in *ISMIR*, 2015.

[5] A. Mesaros and T. Virtanen, "Automatic alignment of music audio and lyrics," in *DaFX-08*, 2008.

[6] C.-K. Wang, R.-Y. Lyu, and Y.-C. Chiang, "An automatic singing transcription system with multilingual singing lyric recognizer and robust melody tracker.," in *INTERSPEECH*. 2003, ISCA.

[7] A. Sasou, M. Goto, S. Hayamizu, and K. Tanaka, "An autoregressive, non-stationary excited signal parameter estimation method and an evaluation of a singing-voice recognition," in *ICASSP*, 2005.

[8] M. Suzuki, T. Hosoya, A. Ito, and S. Makino, "Music information retrieval from a singing voice using lyrics and melody information," *EURASIP J. Adv. Sig. Proc.*, 2007.

[9] M. McVicar, D. P. W. Ellis, and M. Goto, "Leveraging repetition for improved automatic lyric transcription in popular music," in *ICASSP*, 2014.

[10] M. Gruhne, K. Schmidt, and C. Dittmar, "Phoneme recognition in popular music," in *ISMIR*, 2007.

[11] W.-H. Tsai and H.-M. Wang, "Automatic identification of the sung language in popular music recordings," *J. New Music Res.*, vol. 36, no. 2, pp. 105–114, 2017.

[12] Anna M. Kruspe, "Improving singing language identification through i-vector extraction," in *DAFx-14*, 2014.

[13] A. Loscos, P. Cano, and J. Bonada, "Low-delay singing voice alignment to text," in *Proceedings of the ICMC*, 1999.

[14] C. H. Wong, W. M. Szeto, and K. H. Wong, "Automatic lyrics alignment for cantonese popular music," *Multimedia Syst.*, vol. 12, no. 4-5, pp. 307–323, 2007.

[15] M. Müller, F. Kurth, D. Damm, C. Fremerey, and M. Clausen, "Lyrics-based audio retrieval and multimodal navigation in music collections," in *ECDL*, 2007.

[16] M.-Y. Kan, Y. Wang, D. Iskandar, T. L. Nwe, and A. Shenoy, "Lyrically: Automatic synchronization of textual lyrics to acoustic music signals," *IEEE Trans. Audio, Speech & Language Processing*, vol. 16, no. 2, pp. 338–349, 2008.

[17] K. Chen, S. Gao, Y. Zhu, and Q. Sun, "Popular song and lyrics synchronization and its application to music information retrieval," *SPIE*, 2006.

[18] H. Fujihara, M. Goto, J. Ogata, K. Komatani, T. Ogata, and H. G. Okuno, "Automatic Synchronization Between Lyrics and Music CD Recordings Based on Viterbi Alignment of Segregated Vocal Signals," in *IEEE ISM*, 2006.

[19] D. Iskandar, Y. Wang, M.-Y. Kan, and H. Li, "Syllabic level automatic synchronization of music signals and text lyrics," in *14th ACM International Conference on Multimedia*, 2006.

[20] H. Fujihara and M. Goto, "Three techniques for improving automatic synchronization between music and lyrics: Fricative detection, filler model, and novel feature vectors for vocal activity detection," in *ICASSP*, 2008.

[21] H. Fujihara, M. Goto, J. Ogata, and H. G. Okuno, "LyricSynchronizer: Automatic Synchronization System Between Musical Audio Signals and Lyrics," *J. Sel. Topics Signal Processing*, vol. 5, no. 6, pp. 1252–1261, 2011.

[22] M. Mauch, H. Fujihara, and M. Goto, "Integrating additional chord information into hmm-based lyrics-to-audio alignment," *Trans. Audio, Speech and Lang. Proc.*, vol. 20, no. 1, pp. 200–210, Jan. 2012.

[23] T. Hosoya, M. Suzuki, A. Ito, and S. Makino, "Lyrics recognition from a singing voice based on finite state automaton for music information retrieval," in *ISMIR*, 2005.

[24] M. J. F. Gales and P. C. Woodland, "Mean and Variance Adaptation within the MLLR Framework," *Computer Speech & Language*, vol. 10, pp. 249–264, 1996.

[25] M. Suzuki, T. Hosoya, A. Ito, and S. Makino, "Music information retrieval from a singing voice based on verification of recognized hypotheses," in *ISMIR*, 2006, pp. 168–171.

[26] C.-C. Wang, J.-S. R. Jang, and W. Wang, "An Improved Query by Singing/Humming System Using Melody and Lyrics Information," in *ISMIR*, 2010.

[27] H. Fujihara, M. Goto, and J. Ogata, "Hyperlinking lyrics: A method for creating hyperlinks between phrases in song lyrics," in *ISMIR*, 2008.

[28] H. Fujihara, T. Kitahara, M. Goto, K. Komatani, T. Ogata, and H. G. Okuno, "Singer identification based on accompaniment sound reduction and reliable frame selection," in *ISMIR*, 2005.

[29] A. Mesaros and T. Virtanen, "Adaptation of a speech recognizer for singing voice," in *European Signal Processing Conference*, 2009.

[30] A. Mesaros and T. Virtanen, "Recognition of phonemes and words in singing.," in *ICASSP*, 2010.

[31] A. Mesaros and T. Virtanen, "Automatic understanding of lyrics from singing," *Akustiikkapäivät*, 2011.

[32] J. C. Smith, *Correlation analyses of encoded music performance*, Ph.D. thesis, Stanford University, 2013.

[33] J. S. Garofolo et al., "TIMIT Acoustic-Phonetic Continuous Speech Corpus," Tech. Rep., Linguistic Data Consortium, Philadelphia, 1993.

[34] A. M. Kruspe, "Bootstrapping a system for phoneme recognition and keyword spotting in unaccompanied singing," in *ISMIR*, 2016.