The Interplay of Information Retrieval and Query by Singing with Words

Iman Amini
School of Computer Science
RMIT
Melbourne, Australia
iman.amini@rmit.edu.au

Alexandra L. Uitdenbogerd
School of Computer Science
RMIT
Melbourne, Australia
sandrau@rmit.edu.au

Mark Sanderson
School of Computer Science
RMIT
Melbourne, Australia
mark.sanderson@rmit.edu.au

Abstract *Speech recognition can be used in music retrieval systems to identify the words in users' sung queries. Our aim was to determine which of several techniques is most suitable for retrieving songs given a sung query with words. We used Sphinx for speech recognition, and tested several retrieval techniques on the output of the recognition system. The most effective retrieval technique was a combination of Edit Distance and Okapi, which persistently retrieved the correct song at the top one ranked results given that the queries were at least 50% correct. However, techniques performed differently when the queries were split into four buckets with varying level of correctness in the range of 0 to 73%.*

Keywords Pattern Matching, Ranking, Speech Recognition, Music Information Retrieval.

1 Introduction

Music Information Retrieval (MIR) involves finding music using a user query to a system. MIR systems differ in what information they use and how they use it as evidence to find music, songs or lyrics. Sung queries carry two types of relevant information, verbal and melodic. The type of MIR investigated in this paper focuses on the former, that is, words sung by the user to find the relevant lyric. A system that handles such queries can be divided into two sub-systems, namely, speech recognition and lyric matching.

Systems based on sung lyrics can fail due to poor "speech" recognition, or suboptimal matching techniques. Table 1 is an example that depicts the typical limits of speech recognition systems. The first column shows recognition output for the original input text given in the second column.

Proceedings of the 16th Australasian Document Computing Symposium, Canberra, Australia, 2 December 2011.

Acoustic characteristics of a singing voice vary from those of speech. Words are often pronounced differently, vowels are sustained longer, and some words are sung with varying pitch. Indeed, words sung by sopranos at high pitch lose clarity due to the overlap of pitch and vowel frequencies [6]. These factors can degrade the accuracy of the speech recognition. Others have worked on solutions to these problems by applying speaker adaptation techniques [13, 20].

In this paper, however, we focus on the lyric matching techniques for sung queries. We test and combine five techniques that can be categorized into three groups: word-based matching, edit distance, and phonetic matching.

Croft [2] earlier observed that ranking algorithms that have similar effectiveness can have a very low overlap in their result sets. We test 5 different matching techniques and combine our best word-based and phoneme- or character-based techniques. Results are promising, with combinations of Okapi similarity measure on words and an edit distance formulation (Okapi-Edit distance) and SAPS-L giving the best results.

The rest of this paper, describes related research work, followed by details of the range of lyric matching techniques used. Next, the retrieval experiment and results are outlined, before the paper concludes.

Transcription

The term transcription generally refers to two distinct set of outputs, both generated from a sung query.

1. Textual transcription, denotes the conversion of audio query to words.
2. Musical transcription, refers to the conversion of audio query to musical notes.

Both types of transcription involve analysis of frequency spectrograms generated from the audio data [13]. In the case of musical transcription, some

Recognized Segment	Original Segment
me been raining	it's been raining
gold who mind do fool around	don't fool around don't fool around
every say the a	every single day
let there be sang	let there be spring
morning light and we sang chan	morning light and we sang here

Table 1: Original lyrics misheard by the Speech Recognition generated by the PocketSphinx toolkit

of the frequencies have a direct correspondence to the pitch of notes that are sung, while others are related to such audio features as the timbre of the instrument or voice. A transcription algorithm for symbolic notes converts frequencies into its corresponding musical notes.

On the other hand, in order to produce a *textual* transcription of the sung query, speech recognition system uses a probabilistic language model such as n-gram or finite state grammar, to produce a sequence of uttered words.

2 Related Work

There are several main areas of prior work relevant to this paper: speech recognition – particularly for singing, speech retrieval, misheard lyric matching, and music information retrieval using sung text. We restrict our survey of speech recognition to work that is specifically about singing, and provide some examples of speech retrieval work.

Many researchers [11, 16], have used MLLR (Maximum Likelihood Linear Regression) to carry out speaker adaptation. In order to avoid customization to a specific user Suzuki et al. [20] used MLLR with 6 male singers employed to sing 127 choruses for training data. They experimented on singing recognition using a normal Finite State Automaton (FSA) and a modified FSA for a Japanese language grammar. Using the modified version of FSA which exploits a constraint in the Japanese grammar, they improved word recognition correctness from 83.2% to 86.0% and retrieval accuracy from 82% to 85% as compared to the normal FSA.

Mesaros and Virtanen [13] trained and configured their speech recognition to minimize the chance of errors in decoding the speech data, considering models that are specific to gender and singer. For retrieval they used a simple count of the words occurring in both the query and songs, achieving a top-ranked match for 57% of the queries for their dataset. Wang et al [21], in contrast, used a dynamic programming-based alignment of syllables for the same purpose. Their system used both melody and lyrics information. They first determined whether the user had sung or hummed by comparing the number of distinct phones decoded in the user's acoustic model, which was expected to be less for humming than for singing. A related but simpler problem that was studied is the task of audio and lyric alignment [3].

Phonetic matching is the task of retrieving words that are similar in pronunciation but not necessarily in spelling. There has been a number of studies on phonetic matching to improve name retrieval [1, 17], however, the problem of phonetic matching using sung queries has rarely been tackled. Zobel and Dart [24] showed that a phonetic-based edit distance (Editex) and a competing technique known as Ipadist were more effective than various commonly used alternatives when matching surname queries to a data set of 30,000 distinct surnames. Prior to the above experiment, Zobel and Dart [23] had proved that q-grams and the minimal edit distance were superior. While most of the phonetic approaches to string matching convert the strings into some canonical forms to provide the grounds to match phonetically similar strings, Syllable Alignment Pattern Searching (SAPS) is different in that it segments two phonetic strings into syllables to calculate their similarity score [4].

Ring and Uitdenbogerd [18] investigated the problem of misheard lyrics and found that standard edit distance was as effective as more sophisticated phonetic variants. Xu et al. [22] used a confusion matrix to solve the same task for lyrics in Japanese. Hirjee and Brown [5], trained a probabilistic model on examples of actual misheard lyrics and developed a phoneme similarity scoring matrix. This probabilistic method significantly outperformed all other techniques including SAPS-L and Phoneme Edit Distance, finding 5-8% more correct lyrics within the first five top results than the previous best method, Phoneme Edit Distance. This model was based on phonetic confusion data constructed from pairs of original misheard and correct lyrics found on misheard lyrics websites. For any given pair of phonemes a and b, the model generates a log-odds score outputting the likelihood of a being (mis)heard as b.

Speech retrieval is a similar problem to retrieval of music via sung queries with words, in that both involve retrieving audio by matching a representation of the words that the query and target document contain. Ng et al. [14, 15] found that word-based retrieval was consistently more effective, but, given reliable speech recognition, phoneme-based approaches had similar effectiveness, and had the advantage of handling out-of-vocabulary terms. They used n-grams of varying lengths, finding that 3-grams and 4-grams improved retrieval.

3 Matching

Here we describe the three classes of matching techniques used.

3.1 Word-based matching

Two forms of word-based matching were used in the experiments.

Our baseline technique counts the number of common words in the lyric and the query excluding the repeated words in the lyrics. For instance, if the word "school" happens to occur twice in the query and three times in the lyric, the algorithm will score two points [13]. We refer to this technique as *Word Count*.

We also tested *Okapi BM25* [7], which is a probabilistic similarity measure that exploits factors such as term frequency, and is known to be very effective for text retrieval. The formula and constant values used for our experiments are shown in equation 1.

$$BM25(Q,D) = \sum_{(t \in Q)} w_t . \frac{(k_1+1)f_{d,t}}{K + f_{d,t}} . \frac{(k_3+1)f_{q,t}}{k_3 + f_{q,t}} \quad (1)$$

where $K = k_1.((1-b) + \frac{b.L_d}{AL})$
Q query
D document
w_t is the Robertson-Spark Jones weight
$f_{d,t}$ and $f_{q,t}$ are the number of occurrences of term t in the document and query, respectively
L_d and AL are the document length and average document length
k_1,k_3 and b are parameters that are determined empirically
k_1 = 1.2, b = 0.75 and k_3 = 0

k_3 is often set to 0 because of the short nature of queries which often implies that words in the queries do not occur more than once.

3.2 Edit Distance

Two forms of edit distance were used in the experiments. The Levenshtein *edit distance* between two strings can be described as the minimum number of edit operations (e.g. insertions, deletions and substitutions) that are needed to transform the first string into the second one. Equation (2) demonstrates the recurrence relation for minimal Levenshtein edit distance, in which function $r(s_i, t_j)$ returns 0 in case of a two identical characters and 1 otherwise.

$$\begin{aligned} edit(0,0) &= 0 \\ edit(i,0) &= i \\ edit(0,j) &= j \\ edit(i,j) &= min[edit(i-1,j)+1, \\ & \quad edit(i,j-1)+1, \\ & \quad edit(i-1,j-1)+r(s_i,t_j)] \end{aligned} \quad (2)$$

Code:	0	1	2	3	4
Letters:	aeiouy	bp	ckq	dt	lr
Code:	5	6	7	8	9
Letters:	mn	gj	fpv	sxz	csz

Table 2: Editex letter groups

We also used *Editex* [24], which can be described as a phonetic version of the Edit Distance. Table 2 shows the letter groupings used in the matching process. The recurrence relation of Editex was modified to be local rather than global in its alignment of strings as shown in equation (3), due to the different length between query and song being matched. Function $d(a,b)$ is a redefined version of $r(a,b)$ used in Edit Distance, and is defined in more detail by Zobel and Dart [24].

$$\begin{aligned} edit(0,0) &= 0 \\ edit(i,0) &= 0 \\ edit(0,j) &= edit(0,j-1) + d(t_{j-1}, t_j) \\ edit(i,j) &= min[edit(i-1,j) + d(s_{i-1}, s_i), \\ & \quad edit(i,j-1) + d(t_{j-1}, t_j), \\ & \quad edit(i-1,j-1) + r(s_i, t_j)] \end{aligned} \quad (3)$$

3.3 Phonetic matching

The phonetic matching technique *SAPS* consists of three steps, described in Gong and Chan [4]. The modified version of SAPS, known as SAPS-L used in [18] and this paper does not perform global alignment on lyrics to avoid substitution penalties caused by the big length difference between lyrics and queries. SAPS-L slightly varies in all the three phases:

1. Preprocessing: SAPS-L transforms the plain text into a canonical form in order to obtain an accurate syllable segmentation. For example, it maps "tjV" (where V is any vowel) to "chV" and "ph" to "f".

2. Segmentation: The strings are segmented into "syllables". For example, "dancing" is segmented thus: "DanSing".

3. Alignment and Similarity Calculation: Alignment is similar to edit distance, but with more complex scoring. Table 3 are the parameters defining the scores for substitutions and gaps, s_n representing a substitution and g_n a gap. The initial condition set for the local alignment of the lyrics and global alignment of the query is given in equation 4. For further details, see Ring and Uitdenbogerd [18].

$$\begin{aligned} M[i,j] &= max \\ M[0,0] &= 0 \\ M[i,0] &= 0 \\ M[0,j] &= M[0,j-1] + g(S_2[j], -) \end{aligned} \quad (4)$$

Constant	Meaning	Value
s_1	match within strings	1
s_2	mismatch within strings	-1
s_3	start of a syllable	-4
s_4	match starting syllables	6
s_5	mismatch starting syllables	-2
g_1	gap not at start of syllable	-1
g_2	gap at start of syllable	-3

Table 3: Parameters defining the scores for substitutions and gaps

4 Experiment

Our aim was to test several matching techniques for effectiveness when used on words extracted from sung queries. In particular, we compared word-based techniques with character or phoneme-based matching, as well as combinations of both approaches, since this can often lead to better results [8, 9, 19]. We used the Sphinx decoder [10] for speech recognition, with input data as described in Section 4.1.

For the retrieval effectiveness evaluation, we assumed that a retrieved song was relevant if it was the same song as the query. For our collection this meant that there was only one relevant answer for each query. As such, we used reciprocal rank and success@n measures of effectiveness.

We explore two variables associated with the problem: query length and speech recognition correctness. Query length was achieved by truncating to a specific number of words, selecting the first 5, 10, and 15 query terms. Three queries had a length in the range 11–14. Creating speech recognition correctness as the second continuous variable involved splitting queries into four buckets with varying level of correctness. Queries were sorted in increasing order of correctness and grouped into different buckets (levels of correctness were 0%, 20%, 37%, 73%). Note that the overal overage of correctness being 32.5% for all the buckets is demonstrated in Table 5. For the 50 queries, two buckets contained 13 queries, two contained 12 queries. This allowed us to study the behaviour of techniques when queries are entirely incorrect (0%) and relatively correct (20-37-73%). Correctness was calculated based on exact matching of the words, therefore, we expected phoneme/character based techniques to show marginal success for the correctness being as low as 0, due to the existence of similar sounding words with different spellings.

4.1 Data Collection

Approximately three hours of training data was gathered from a speech and a singing database. The speech data used for training was about two hours in length and did not have any specific context. It was collected from the CMU Assignment package[1]. The singing collection consisted of one hour of singing in English from a variety of musical genres by two authors of the paper: one beginner male and one experienced female. Our singing collection from the male singer was about 40 minutes long, consisting of 27 songs and the rest of the singing material sung by the female singer consisted of 16 songs, approximately 20 minutes long. Some singing files were recorded using the Audacity software version 1.2.6 and configured to the setting which matched the speech data while others were recorded using Cubase.

4.2 Manipulating the Singing and Lyrics Database

The recordings were either made at 16,000 Hz sample rate in mono, or converted to that format as per the requirements for the speech recognition system. The recorded songs were segmented into 30 seconds chunks including silence gaps and Mel Frequency Cepstral Coefficients (MFCC) – the most salient features for training the speech recognition system [12] were extracted using the sphinx_fe program.

We gathered a lyrics database with 2,359 songs in English from the authors of [18] and we extend it to include the 43 lyrics used in our training set. The database was annotated and all the punctuation removed except the apostrophes, which had led to some inconsistencies during the early experiment, when removed. For instance, "I'll" has a different pronunciation to "Ill", Finally, all the words were reduced to lower case, and extra spaces removed.

The dictionary contains 2,262 unique words made up of 40 distinct phones and the filler dictionary has two symbols representing the silence gap which occurs at the beginning and in the middle, <s >, or at the end of every song $</s>$. We used sphinx web toolkit [2] to automatically generate the language model which constructs two n-gram models by default: unigram and bigram.

4.3 Speech Recognition Results

For the training purpose we sliced all the singing data to segments with length average of approximately 30 seconds. From the collection of 30 seconds tracks, we manually selected fifty queries. Depending on the length of the original track, we select 0 to 6 segments from each song, each representing a query. We calculated correctness for the truncated version of our query set, containing 10 terms for each query, and the original set with 15 terms on average, per query. Accuracy and error rates are shown in Table 4 and 5. Correctness was calculated using the formula:

$$Correctness = \frac{N-D-S}{N}$$

where N is the total number of words in the original lyrics, D is the number of deletions and S is the total

[1] available at `http://www.speech.cs.cmu.edu/15-492/assignments/hw1/hw1_data.zip`

[2] `http://www.speech.cs.cmu.edu/tools/lmtool-new.html`

number of substitutions. Accuracy was calculated as follows:

$Accuracy = \frac{N-D-S-I}{N}$
where I is the number of insertions.

Total words	Percent correct	Error	Accuracy
1155	40.78%	79.22%	20.78%
Insertions	Deletions	Substitutions	
231	34	650	

Table 4: Sentence and word accuracy generated for original query set

Total words	Percent correct	Error	Accuracy
493	32.5%	70.91%	29.92%
Insertions	Deletions	Substitutions	
19	15	314	

Table 5: Sentence and word accuracy generated for truncated query set

The results are fairly poor due to the small training set, and limited control over how the singing data was used compared to the speech data. However, we were interested in how retrieval techniques work across a range of recognition accuracies, and a low starting point allows this to be explored over a wider range.

4.4 Retrieval Results

In this section we analyze the retrieval results in detail, compare and evaluate the effect of different variables on the result sets. We measure the effectiveness of the results using success at 10 and to make a better distinction between the 4 top techniques we use reciprocal rank which shows the rate of success at 1 as well.

Initially we compare 5 techniques using the queries made up of the first 10 words in the raw speech recognition output. Table 6 shows that the best technique overall is SAPS-L finding the best match in top 10 results for 58% of the queries, while both word-based techniques (Word Count and Okapi) performed poorly. The t-test calculated on these results shows that the only technique to be statistically significantly worse than any other is the baseline Word Count with p value of 0.0041. We believe that a larger test collection can potentially reveal statistical difference amongst the top 4 techniques.

Word Count	SAPS-L	Editex	Okapi	Edit Distance
25.25%	58%	51.75%	37.5%	56.25%

Table 6: Percentage of success @ 10

Figure 1 shows that effectiveness increases with query length. Editex and SAPS-L appeared to benefit most from the longer queries with an increase of 12%.

Figure 2 shows the difference in retrieval effectiveness for queries binned into the four buckets depending on recognition accuracy for queries of ten words. Next we combine some of the word based and character based techniques to find the best

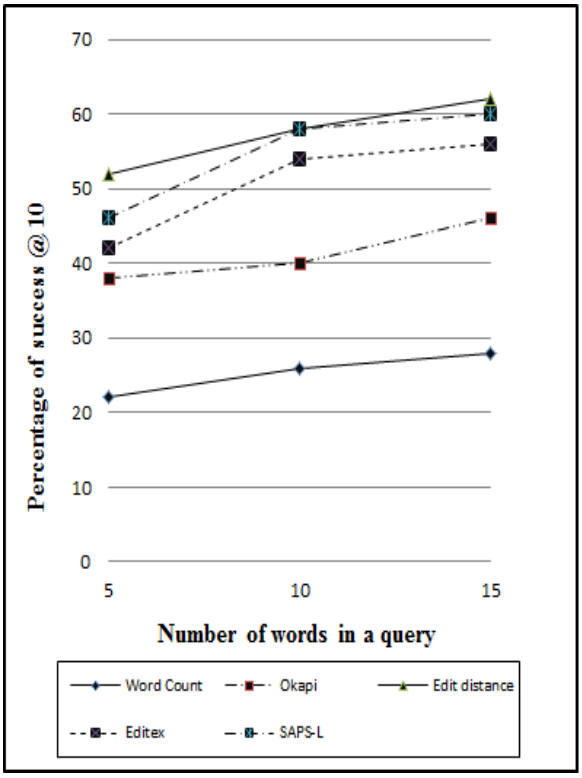

Figure 1: Trend in retrieval effectiveness with varying query length

Okapi-Edit Distance	Okapi-Editex
58%	56%

Table 7: Percentage of success @ 10 for combination of techniques

combination, using the same set of queries used for the previous experiment demonstrated in Figure 2 and results are shown in Table 7. Combination of techniques involved ranking all the lyrics twice for the two ranking techniques being combined against the given set of queries as shown in equation 5. We have tried different parameters to weight each ranking score and selected the most promising ones that led to better retrieval results. In the case of Okapi, the higher score implies more similarity, and for distance based rankings, higher similarities get lower scores.

$$\begin{aligned} \text{okapi-edit distance} &= 0.5(O) - 2(ED) \\ \text{okapi-editex} &= O - 2(EX) \end{aligned} \quad (5)$$

where
O : score generated by Okapi
ED : score generated by Edit Distance
EX : score generated by Editex

Combining Okapi and edit distance and SAPS-L appear to be the best techniques, however, no single technique is robust against varying speech recognition accuracy. Moreover, combinations of different pair of techniques did not lead to any improvement over the al-

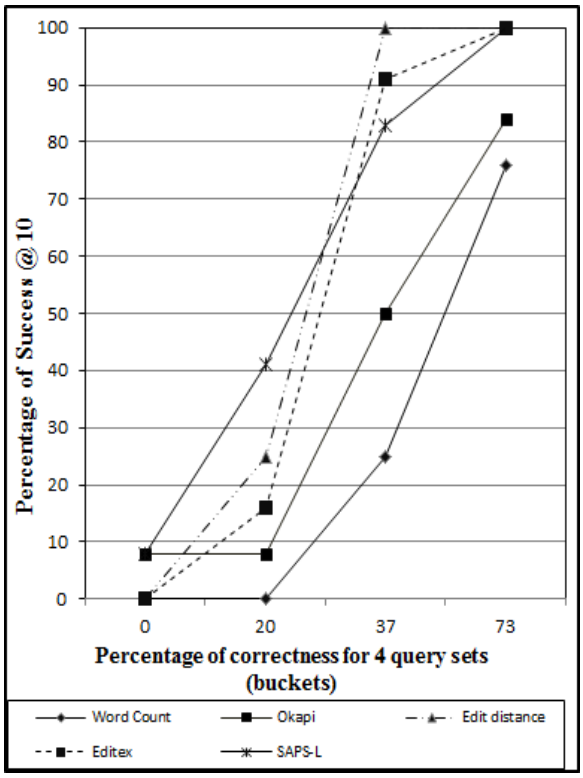

Figure 2: Trend of the retrieval techniques queried against four different sets of queries

ready combined techniques. In order to make better distinction amongst the top four techniques, we calculate reciprocal ranks for each of the queries. We intended to discover which technique is reliable and for what point of correctness. Figure 3 shows reciprocal ranks for each of the 50 queries plotted along the y axis with varying level of correctness along the x axis. Note that correctness was calculated for each individual query. We can see from the figure that SAPS-L performs 100% accurately for correctness level ranging from 80 to 100%. The combination of Edit Distance and Okapi, has the most desirable confidence point, finding the best match at top one ranked results, given that queries are at least 50% accurate. However, the marginal success of some techniques at 0% level of correctness, "Okapi" in particular, were questionable. Table 8 shows the query that retrieves the correct match, despite being highly inaccurate. This query has one word overlap with the target lyric which has a rare occurrence in the test collection (exploited by the Okapi), and is repeated many times in the original lyric, therefore giving a high score to the target lyric. The wrong position of the correct word, however, doesn't increase the correctness, which is 0%.

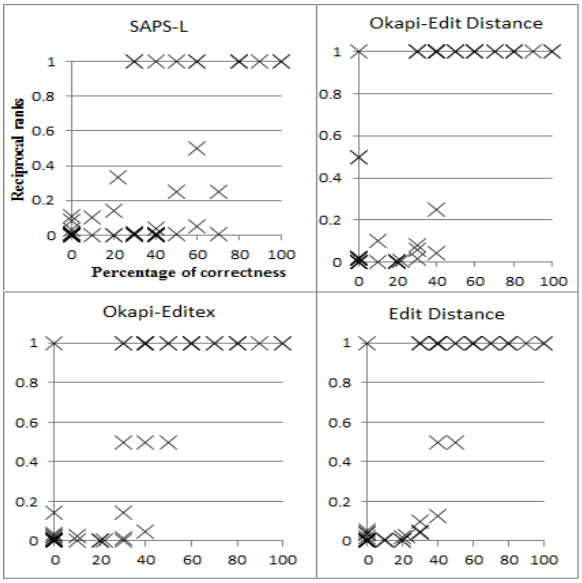

Figure 3: Reciprocal ranks for top 4 techniques

5 Conclusion

In this paper we attempted to learn ways to improve the effectiveness of query by singing with words, which uses speech recognition to train and recognize user queries. We have explored and tested some of the common matching techniques that have been tested in the context of previously published lyric-based retrieval.

Despite the faulty output produced by the speech recognition, some matching techniques performed reasonably well, although none of the four top techniques were statistically superior to the others. However, the simple "Bag of Words" technique previously used in Mesaros and Virtanen [13] as a proof of concept proves to be persistently and statistically inferior to the other six techniques.

Original:	happy	birthday	sarah	chan	happy	birthday	sarah	chan	happy	happy
Hypothesis:	on	to	have	died	sarah	chan	blind	and	we	sang

Table 8: Speech recognition hypothesis for a noisy query

There is much scope for future work. There are many alternative matching techniques that could be effective for matching transcribed lyrics. It would also be useful to compare recognition output that is naturally at a higher level of accuracy and to classify singing data based on the characteristics of singer's voice.

References

[1] R. Bhagat and E. Hovy. Phonetic models for generating spelling variants. In *Proceedings of the 20th international joint conference on Artifical intelligence*, pages 1570–1575. Morgan Kaufmann Publishers Inc., 2007.

[2] W. Bruce Croft. Combining approaches to ir (invited talk). In *DELOS Workshop: Information Seeking, Searching and Querying in Digital Libraries*, 2000.

[3] H. Fujihara and M. Goto. Three techniques for improving automatic synchronization between music and lyric. In *In proceedings of IEEE International Conference on Audio, Speech and Signal Processing*, pages 69–72, Las Vegas, USA, 2008.

[4] R. Gong and Tony K.Y.Chan. Syllable alignment: A novel model for phonetic string search. In *IEICE Transaction on Information and Systems*, pages 332–329, 2006.

[5] H. Hirjee and D. G.Brown. Solving misheard lyric search queries using a probabilistic model of speech sounds. *11th International Society for Music Information Retrieval Conference ISMIR*, pages 147–152, 2010.

[6] E. Joliveau, J. Smith, and J. Wolfe. Vocal tract resonances in singing: The soprano voice. *Journal of the Acoustical Society of America*, 116(4):2434–2439, October 2004.

[7] K. Spärck Jones, Steve Walker, and Stephen E. Robertson. A probabilistic model of information retrieval: development and comparative experiments - part 1. *Inf. Process. Manage.*, 36(6):779–808, 2000.

[8] B. Kantor, P. Cool, and Quatrain. Combining evidence for information retrieval. In Harman, editor, *Text Retrieval Conference(TREC)*, pages 35–44. National Institute of Standards and Technology Special, 1993.

[9] J. Lee. Combining multiple evidence from different properties of weighting schemes. In *Proc. ACM-SIGIR International Conference on Research and Development in Information Retrieval*, pages 180–188, 1995.

[10] K.F Lee, H.W. Hon, and R. Reddy. An overview of the sphinx speech recognition system. In *IEEE Transactions on Acoustics, Speech and Signal Processing*, pages 35–45, 1990.

[11] C. J. Leggetter and P. C. Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models. *Computer Speech and Language*, pages 171–185, 1995.

[12] B. Logan et al. Mel frequency cepstral coefficients for music modeling. In *International Symposium on Music Information Retrieval*, volume 28, page 5. Citeseer, 2000.

[13] A. Mesaros and Tuomas Virtanen. Automatic recognition of lyrics in singing. *EURASIP J. Audio, Speech and Music Processing*, 2010.

[14] Corinna Ng, R. Wilkinson, and J. Zobel. Experiments in spoken document retrieval using phoneme n-grams. *Speech Communication*, 32(1-2):61–77, 2000.

[15] Corinna Ng and J. Zobel. Speech retrieval using phonemes with error correction. In *SIGIR*, pages 365–366, 1998.

[16] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applicatoin in speech recognition. In Murray Hill, editor, *In Proceedings of the IEEE, Vol. 77, No.2*, pages 257–284, 1989.

[17] H. Raghavan and J. Allan. Matching inconsistently spelled names in automatic speech recognizer output for information retrieval. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 451–458. Association for Computational Linguistics, 2005.

[18] N. Ring and Alexandra L. Uitdenbogerd. Finding 'lucy in disguise': the misheard lyric matching problem. In *Prodeedings of AIRS 2009*, pages 157–167, 2009.

[19] Joseph A. Shaw and Edward A. Fox. Combination of multiple searches. In *TREC*, pages 35–44, 1994.

[20] M. Suzuki, Toru Hosoya, Akinori Ito, and Shozo Makino. Music information retrieval from a singing voice using lyrics and melody information. *EURASIP J. Adv. Sig. Proc.*, pages 151–151, 2007.

[21] Chung-Che Wang, Jyh-Shing Roger Jang, and Wennen Wang. An improved query by singing/humming system using melody and lyrics information. In *11th International Society for Music Information Retrieval Conference*, pages 45–50, 2010.

[22] X. Xu, M. Naito, T. Kato, and K. Kawai. Robust and fast lyric search based on phonetic confusion matrix. In K. Hirata and G. Tzanetakis, editors, *Proc. 10th International Society ofor Music Information Retrieval Conference*, pages 417–422, Kobe, Japan, October 2009. ISMIR.

[23] J. Zobel and Philip W. Dart. Finding approximate matches in large lexicons. *Softw., Pract. Exper.*, 25(3):331–345, 1995.

[24] J. Zobel and Philip W. Dart. Phonetic string matching: Lessons from information retrieval. In *SIGIR*, pages 166–172, 1996.