International Journal of Advanced Intelligence
Volume 8, Number 1, pp.1-11, May, 2016.
© AIA International Advanced Information Institute

Retrieving Vaguely Remembered Lyrics Using N-Gram Edit Distance

Manabu Sasayama
Department of Information Engineering, Kagawa National College of Technology
551 Kohda, Takuma cho, Mitoyo, Kagawa, Japan
sasayama@di.kagawa-nct.ac.jp

Kazuyuki Matsumoto
Institute of Technology and Science, The Tokushima University
Minamijosanjima cho 2-1, Tokushima, Japan
matumoto@is.tokushima-u.ac.jp

Received (10 June 2015)
Revised (10 Sep. 2015)

Current text based music information retrieval systems are based on full-text retrieval engines or matching the exact keywords. If a user vaguely remembers lyrics, those systems are incapable of searching for lyrics. The major type of vaguely remembered is spelling variants. In this paper, we propose using kana for the retrieval of lyrics where queries may contain spelling variants. First, we construct a standard inverted index over the kana converted lyrics. Next, we filter the search result using n-gram Levenshtein distance. We demonstrate the effectiveness of the system through an experiment using queries containing one, two or three spelling variants. From the experiment, all accuracy rates were higher than 90% when the query contains one, two or three spelling variants.

Keywords: Vaguely remembered lyrics; Spelling variants; Kana; n-gram Levenshtein distance.

1. Introduction

Users desire searching for songs they are interested in and often whose name they do not remember. In order to meet this need lyric retrieval systems, such as Uta Map[1] and Evesta[2], and smartphone applications, such as Midomi[3] and Shazam[4], have been created. Such systems are classified by the type of query they support. The first type of query uses melody in which a user hums or sings parts of the desired song[34]. The second type of query makes use of song metadata, such as the artist's name or the song's title[12]. However, current lyric retrieval systems are incapable of searching for lyrics which the user vaguely remembers. In this paper, we describe a system that facilitates the search of vaguely remembered lyrics using n-gram edit distance.

Recently, researchers are examining the problem of searching for vaguely remembered lyrics. Xu et al.[5] propose a retrieval system that uses acoustic similarity between words. They achieve about 60% accuracy on queries containing unseen

Table 1. The type of the vaguely remembered lyrics and its examples.

The type	An original lyrics	A vaguely remembered lyrics
	akirame nai	akirame nai
Spelling variants 1	あきらめ ない	諦め ない
	watasi ni totte	watasi ni totte
Spelling variants 2	わたし にとって	私 にとって
	ikite yuku tikara ga	ikite yuku tuyosa ga
Replacement of word 1	生きてゆく 力 が	いきてゆく 強さ が
	tatoe donna owari	tatoe donna mirai
Replacement of word 2	たとえどんな 終わり	たとえどんな 未来
	seisyun no hibi subete wo egaki	seisyun no hibi egaki
Deletion of word 1	青春の日々全てを 描き	青春の日々描き
	itumo sokoni kimi ga	itumo kimi ga
Deletion of word 2	いつも そこに 君が	いつも君が
	taiyou no youni tuyoku	taiyou no youni kimi wo tuyoku
Insertion of word 1	太陽のように強く	太陽のように 君を 強く
	kimi wo aisiteita	tada kimi wo aisiteita
Insertion of word 2	君を愛していた	ただ 君を愛してた

vaguely remembered lyrics. However, this accuracy is not high enough for adoption by end users. Matsumoto et al.[6] propose a retrieval system with improved accuracy that takes into account artists' word usage. In particular, they focus on queries in which a word is replaced by another or removed. In contrast, the current work focuses on queries in which the lyrics contain spelling variations, e.g. misspellings and alternate spellings. Other notable work includes that of Sasaki et al.[7] who present unknown lyrics to users, but does not aim to provide search of vaguely remembered lyrics.

Our previous research[8] investigates the use of query expansion for searching lyrics. We identify and compare instances of vaguely remembered lyrics to the original artist lyrics. We collect the instances from questions and answers posted on a Q&A site[9]. This investigation reveals that there are five predominate types of vaguely remembered lyrics: 1) spelling variants, 2) word replacement, 3) word deletion, 4) word insertion, and 5) other less common variations. Spelling variations account for 43% and word replacement 38% of the instances. Examples of the types of vaguely remembered lyrics are shown in *Table 1*. The underlined word represents a variation from the original lyrics.

Japanese texts contain many spelling variants that cause mismatches between queries and lyrics. These large amount of spelling variations arises due to the Japanese writing system making use of three different character sets: hiragana, katakana, and kanji. For example, "会う (meet)" can be written either as "会う", "あう" or "アウ". Further compounding the problem is that the word can be written using a different kanji, i.e. "逢う". In addition, Japanese text input requires an indirect input method called kana-kanji conversion. Users may choose to convert inputted kana into kanji or leave it as kana depending on their personal preference. In the example of "会う" all variations share the same kana forms and same pronunciation (“au”).

In this paper, we propose using kana for the retrieval of lyrics where queries may contain spelling variants. We convert queries and lyrics into kana form using a dictionary. Full text search is performed using the standard inverted index over the kana converted lyrics. Conversion and search using kana facilitates all possible forms of word to be matched. Using kana queries, the lyrics retrieval can search with a query which is different from kana lyrics. For example, given the query “会う”, the system can match lyrics which are written using “会う”, “逢う”, “あう” or “アウ”. If we don’t use kana, then we are unable to match differing variations and must perform additional searches or form more complex queries. We propose to use the n-gram Levenshtein distance in order to filter results and handle spelling variations. We construct word-based n-gams over both the queries and lyrics and use the Levenshtein distance over each word n-gram. The total distance is used as the similarity between the query and lyrics. In the next section we explain the basic framework of the proposed method. In section 3 we show the validity of the system by conducting experiments and in section 4 we present the conlusion and future work.

2. Proposed Method

2.1. *Framework*

Methods[10] have been proposed to retrieve lyrics from queries with partial spelling variations. However, the number of queries exponentially increases when a word is deleted from or inserted into a query. In addition, retrieving lyrics is difficult using a query containing many spelling variants. Hence, we propose two methods: 1) lyric retrieval using an inverted index consisting of lyrics in kana and 2) filtering search results using n-gram Levenshtein distance. The combination of these two methods eliminates omission and prevents an increase in the number of queries for queries containing many spelling variations. The process and an example of the proposed method is shown in Fig. 1.

Section 2.2 gives the details of the kana-based query methodology and Section 2.3 gives the details of the n-gram based Levenshtein distance.

2.2. *Lyric retrieval using kana converted queries and lyrics*

The flow of construction of an inverted index is as follows. The lyric database is constructed containing the lyric IDs, song’s names, artist names and lyrics. The first step is word segmenting the lyrics using Mecab. This step is necessary since Japanese does not separate words by spaces.

Next, the word-kana dictionary is built based on IPADIC.

The kana of a word is written in hiragana. Japanese words can be constructed using only kanji, hiragana, or katana or by mixing kanji with hiragana or katakana. Katakana appearing in mixed words are converted into hiragana. An excerpt from

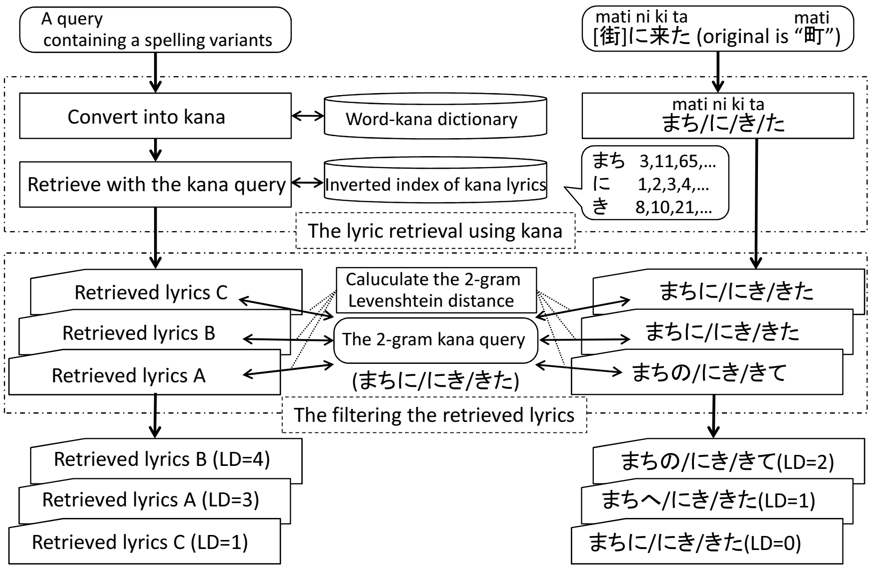

Fig. 1. The process and an example of the proposed method.

Table 2. The word-kana dictionary.

A headword(English)	Kana 1(the reading)	Kana 2	Kana 3
君 (you)	くん (kun)	きみ (kimi)	-
愛 (love)	あい (ai)	ちか (tika)	めぐみ (megumi)
トヨタ自動車 (Toyota Motor)	とよたじどうしゃ (toyota jidosya)	-	-

the constructed word-kana dictionary is shown in *Table 2*. After performing morphological analysis, lyrics are converted into kana using the word-kana dictionary. If a word has multiple possible kana readings, it is converted into all kanas. Because of the use of an inverted index, the mapping to all possible kana doesn't affect the speed of retrieval. Moreover, due to the use of the n-gram Levenshtein distance the number of spurious results are also minimized. We build an inverted index of words and lyric IDs from the kana versions of the lyrics. We exclude digits and alphabetic characters.

Querying of the inverted index begins by first converting the query into its kana form using Mecab and the previously constructed word-kana dictionary. Words which have multiple possible kana forms are mapped into the first candidate. Normalizing queries and lyrics into their kana forms increases the number of lyrics returned by the search engine. In the next section, we explain a methodology for filtering the search results in order to not overload the user.

2.3. *Filtering search results using the n-gram Levenshtein distance*

We present a method to filter the search results using the n-gram Levenshtein distance. Previous reserch[12] have shown that search taking into account of the order of words in lyrics results in higher accuracy than not taking into account of the order of words. Thus, we use an n-gram search methodology. The output of the retrieval system is the query separated in words and converted into kana and the lyrics matching the query. The similarity between the query and each of the retrieved lyrics is calculated using the n-gram Levenshtein distance. The similarity is calculated as the sum of the similarities between each query n-gram and each lyric n-gram using the Levenshtein distance. The shortest Levenshtein distance between a query n-gram and a lyric n-gram is determined as the Levenshtein distance between the one n-gram and lyrics. The equation for the total Levenshtein distance(nLD_{total}) between query(Q) and a lyric(L) is shown in equation (1).

$$nLD_{total}(Q, L) = \sum_{Q_{n\text{-}gram} \in Q} \mathop{\arg\min}_{L_{n\text{-}gram} \in L} \Big(nLD(Q_{n\text{-}gram}, L_{n\text{-}gram}) \Big). \tag{1}$$

In the equation, nLD expresses the Levenshtein distance between one n-gram of query and one n-gram the lyrics. $Q_{n\text{-}gram}$ expresses an n-gram of query, $L_{n\text{-}gram}$ expresses an n-gram of the lyrics. The similarity between the query and a lyric is calculated as the inverse of the Levenshtein distance.

3. Evaluation Experiment

In this section, we evaluate the lyric retrieval system in order to show that searching with queries containing spelling variants is possible. In this experiment, we apply 2-gram(n=2) Levenshtein distance.

3.1. *Experimental Conditions*

We prepare queries containing spelling variants and associated lyrics IDs. We construct querys containing one spelling variant, two spelling variants and three spelling variants. There were no more than three spelling variants in an instance of vaguely remembered lyrics. There are 145 kinds of spelling variants. We use the 84 kinds with the highest frequency. We generate 5 queries for each of the 84 kinds of spelling variants for a total of 420 queries. We use three types of queries for evaluation(total 1260 queries). Example queries for each type is shown in *Table 3*. The kinds of spelling variants and how to generate queris are described in next section.

For each of the queries, lyrics are retrieved from the search engine and ranked using the 2-gram Levenshtein distance. We evaluate the ranked retrieved lyrics from a twofold perspective of general retrieval and practical lyric retrieval. We calculate the MRR(Mean Reciprocal Rank) from the first perspective and the appearance of the correct lyric in the ranked retrieved lyrics(there is only one correct lyric[a] for

[a]The case that a same lyrics is sung by different artists.

Table 3. An example of queries for each type.

A type of query	Original lyrics	Lyrics containing spelling variants
One spelling variant	wasure nai itu made mo [忘れ] ない いつ まで も saigo ni atta sono toki 最後 に 会っ た その [瞬間] に	wasure nai itu made mo [わすれ] ない いつ まで も saigo ni atta sono toki 最後 に 会っ た その [時] に
Two spelling variants	meguri ae ta toki yori mo めぐり [逢え] た [時] より も ano hito ha ima toui mati あの 人 は [今] 遠い [町]	meguri ae ta toki yori mo めぐり [会え] た [とき] より も ano hito ha ima toui mati あの 人 は [いま] 遠い [街]
Three spelling variants	kimi no eran da koto da kara [君] の [選ん] だ [こと] だ から kimi ga kokoro ni subete tukare ta toki [君] が 心 に [すべて] 疲れ た [とき]	kimi no eran da koto da kara [きみ] の [えらん] だ [事] だ から kimi ga kokoro ni subete tukare ta toki [きみ] が 心 に [全て] 疲れ た [時]

each query). The equations for the MRR are shown in the equation (2) and (3). RR adds the reciprocal of the rank appearing of the correct lyric. MRR is obtained by dividing RR by the number of queries. In case of a tie, we return the rank plus the number of ties. For example, we regard r=2 as r=6, when the rank of the correct lyric is two and there are five lyrics tied for the second rank.

We use the ratio of queries with the correct lyric in the top 20 for the second perspective. We choose the top 20 as we assume a user can confirm no more than 20 retrieved lyrics(1 lyrics per 1 minute). The accuracy is obtained by dividing the number of correct lyrics appearing in the top 20 divided by the number of queries returning two or more lyrics. We regard RR as zero when the correct lyric is not in the retrieved lyrics.

$$RR_i = \frac{1}{r} \tag{2}$$

$$MRR = \frac{1}{C}\sum_{i}^{C} RR_i \tag{3}$$

r : Rank of correct lyrics
C : The number of queries which have two or more retrieved lyrics

3.1.1. *Generating queries containing spelling variations*

In this section, we explain how to generate queries containing spelling variations for the evaluation experiment. The queries are randomly generated from all lyrics in the lyrics database. The process is as follows.

Step 1 A line in a lyric containing the original version of a spelling variation in the spelling variants dictionary (see previous discussion) is taken as a candidate query.
Step 2 Original words in the candidate query are converted into spelling variations.

Table 4. The linguistic resources used in the experiment.

The linguistic resources	The amount of songs/words	A note
The lyric database	91212 (songs)	Lyrics IDs, song's names, artist names and lyrics
The word-kana dictionary	249262 (words)	Based on IPADIC
The inverted index consisting of kana lyrics	96849 (words)	Based on the lyric database containing function words
The normal inverted index consisting of lyrics	122385 (words)	Based on the lyric database containing function words

Step 3 Search the lyric database with the candidate query keeping only those queries which have multiple search results. Additionally, only queries with three morphemes or more are used.

The spelling variants dictionary consists of spelling variants words and their associated original word. The spelling variant and its original are taken from the previously acquired instances of vaguely remembered lyrics and associated original lyrics. There are a total of 145 kinds of spelling variations.

3.1.2. *Linguistic resources*

The linguistic resources used in the experiment are shown in *Table 4*.

3.2. ***Results***

Table 5 shows the results of filtering lyrics using the 2-gram Levenshtein distance after retrieving lyrics using a kana query. "HIT" represents the number of queries which have one or more retrieval results. "INCORRECT" represents the number of queries where the correct lyric is not contained in the results. "NO HIT" represents the number of queries that did not have any returned lyrics. As is seen in Table 5, the number of "HIT" decreases and "INCORRECT" increases as the number of spelling variations increases. For comparison, *Table 6* shows the results of filtering lyrics using the 2-gram Levenshtein distance retrieving lyrics using a normal query, i.e. not converting into kana. The retrieval results indicate that the queries containing spelling variations were unable to be matched.

Table 7 shows the percentage of queries with the correct lyric found within the top 20 search results. Queries containing one spelling variation resulted in a very high accuracy of 96% and MRR of 0.795. Queries with two or three spelling variations also had high accuracies of 91.5% with and MRR of 0.750 and 90.5% with and MRR of 0.740 respectively. However, when using a normal query the results were mostly "NO HIT" or "INCORRECT".

In the kana retrieval system, queries containing no spelling variations also had a very high accuracy of 98.8% with an MRR of 0.829. This indicates that the proposed method can retrieve lyrics not only when the query contains spelling variations but

Table 5. The results of filtering using lyrics using the 2-gram Levenshtein distance after lyrics retrieval with a kana query. Each type has 420 queries.

A type of query	HIT	INCORRECT	NO HIT	MRR
One spelling variants	401	8	19	0.795
Twe spelling variants	387	18	33	0.750
Three spelling variants	380	22	40	0.740
A normal	420	0	0	0.829

Table 6. The results of filtering using the 2-gram Levenshtein distance after lyrics retrieval with a normal query. Each type has 420 queries.

A type of query	HIT	INCORRECT	NO HIT	MRR
One spelling variants	195	177	225	0.071
Twe spelling variants	143	139	277	0.024
Three spelling variants	86	84	334	0.023
A normal	420	0	0	0.863

Table 7. The percentage of queries with the correct lyric found within the top 20 search results.

	Number of Spelling Variations			
	One	Two	Three	None
The kana retrieval+LD	96.5%	91.5%	90.5%	98.8%
	(387/401)	(354/387)	(344/380)	(415/420)
The normal retrieval+LD	8.7%	2.8%	2.3%	99.0%
	(17/195)	(4/143)	(2/86)	(416/420)

also for queries that don't contain spelling variations, i.e. a normal query.

3.3. *Discussion*

We confirmed effectiveness of the proposed method because the accuracy rates of the kana-based lyric retrieval system with queries containing spelling variations were high compared to the low rates for the normal lyric retrieval system. The proposed method is not influenced by the kanji-to-kana conversion or the expressions contained in the lyric sheets. We can say this, because rates were higher than 90% for queries containing one, two or three spelling variations.

The queries for which the system was unable to retrieve any lyrics can be classified into two types. The first type is when the kana form of the spelling variant is different from the kana form of the original word in the lyric, which results in "INCORRECT" or "NO HIT". This type can be further broken down into seven types. Three of these were due to the songwriter arbitrarily substituted a kanji character. For example, the reading of "時" and "瞬間" are "toki" and "syunkan" but songwriters would arbitrarily substitute "瞬間" for "時". In this case 'syunkan' is stored in the inverted index and queries for "時" and its spelling variations would

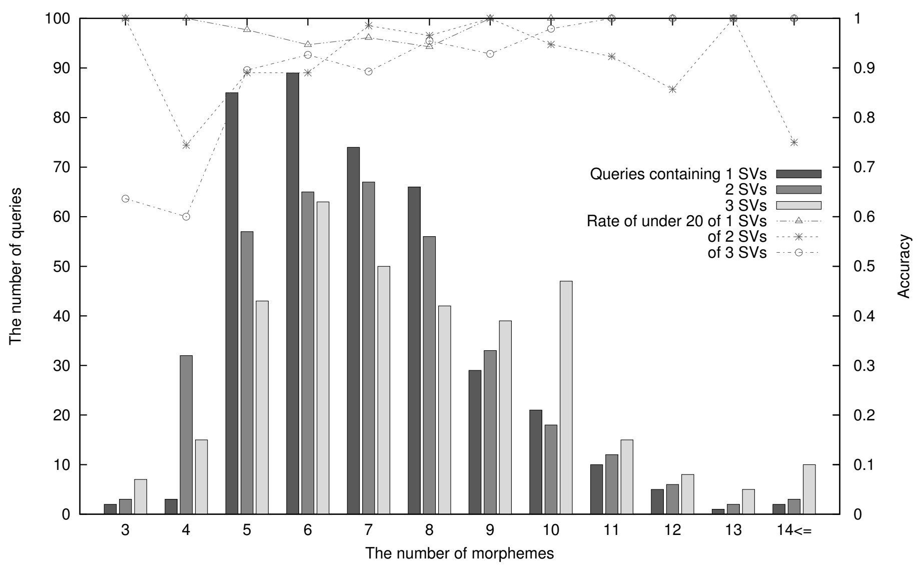

Fig. 2. The relation between the number of morphemes and the accuracy.

not match.

The second type of error is caused when the Levenshtein distance returns zero. This is due to the to common lyrics, for example "/忘れ/て/い/た/想い/". In this case "忘れ" can also be expressed as "わすれ", and "想い" can be expressed as "思い" or "おもい". So, the retrieval results of six combinations were obtained by the lyrics retrieval with a kana query.

Fig. 2 shows the effect of the number of morphemes on the accuracy of the system. The bar graph illustrates the relation between the number of morphems and the number of queries for each query type(one, two, or three spelling variations(SVs).). The line graph indicates the relation between the number of morphemes and the accuracy. As seen in the figure the accuracy is lower when the number of morphemes is four or less. When the number of morphemes is small, the lyrics with has the same 2-grams increase. In order to solve this problem, we need to use information other than lyrics. For example, we use the sex of an artist and the lyric's date of release. Users typically remember those information when they are searching for lyrics.

To continue our discussion let use examine the computing time of the 2-gram Levenshtein filtering method. We measured the computing time using ten queries. For each of the ten queries we ran the system and report the average across the trials. *Table 8* lists the timing results. The average number of morphemes per query was 7.0, the average of the number of retrieved lyrics per query was 295 and the average of the number of morphemes per lyrics was 276.0. The average computing time of the 2-gram Levenshtein distance between one query and its retrieved lyrics

Table 8. The computing time of filtering the retrieved lyrics using the 2-gram LD.

The average of the computing time of the 2-gram LD per a query	The average of the computing time of the 2-gram LD par a query per a lyrics
6.017 (seconds)	0.020 (seconds)

was 6.017 seconds, and between one query and a single retrieved lyrics was 0.020 seconds. The result indicates the method of filtering the retrieved lyrics using 2-gram Levenshtein distance takes time. To reduce the number of calculations, computing can be finished when the distance between the shortest Levenshtein distance at that time and the Levenshtein distance in process of calculation becomes two.

4. Conclusion and Future Work

In this paper, we proposed a lyric retrieval system using kana with the aim of constructing facilitating retrieval for queries containing spelling variants. We constructed an inverted index consisting of kana lyrics. In addition, we proposed to use the n-gram Levenshtein distance in order to filter the retrieval results. In order to verify the effectiveness of such proposed methods, we conducted an evaluation using queries containing one to three spelling variants appearing in an instance of vaguely remembered lyrics. From the experiment, all accuracy rates were higher than 90% when the query contains one, two or three spelling variants. In future work, we would like to develop a methodology for when the kana of the word of spelling variants is different from the kana of the original word.

References

1. Uta Map, `http://www.utamap.com/`.
2. Evesta, `http://www.evesta.jp/lyric/`.
3. Midomi Sound Hound, `http://www.soundhound.com/`.
4. Shazam, `http://www.shazam.com/`.
5. X. Xu, M. Naito, T. Kato, H. Kawai, An Introduction of a Fuzzy Text Retrieval System For Music Information Retrieval, *IPSJ SIG Technical Report*, 2008-MUS-078, pp.41–46, 2008.
6. K. Matsumoto, M. Sasayama, Q. Xiao, A. Fujisawa, M. Yoshida and K. Kita, Reranking the Search Results for Lyric Retrieval Based on the Songwriters' Specific Usage of Words, *The proceedings of the 4th international conference on electronics*, communications and networks (CECNet2014), Beijing, Dec. 2014.
7. S. Sasaki, K.Yoshii, T.Nakano, M. Goto, and S. Morishima: LyricsRadar: A Lyrics Retrieval System Based on Latent Topics of Lyrics, *Proceedings of the 15th International Society for Music Information Retrieval Conference* (ISMIR 2014), pp. 585–590, October 2014.
8. M. Sasayama, K. Matsumoto, Query Expansion Using Semantic Information for Lyric Search System, *Human Communication Group Symposium*, Vol.HCG2013-B-1-3, pp.38–42, Dec. 2013.
9. Yahoo!, `http://chiebukuro.yahoo.co.jp/`.
10. H. Sakamoto, Y. Kitamura, T. Fukushima, T. Yoshino, Evaluation of Multilingual Parallel Text Search Method Using N-gram, *Institute of Electronics, Information, and Communication Engineers*,Technical Report AI2010–52（2011–02）, pp.51–56, 2011.
11. MeCab: Yet Another Part-of-Speech and Morphological Analyzer.
12. Y. Mitsuishi, M. Sasayama, K. Matsumoto, Investigation of spelling variations for vague

memory lyrics, *Journal of Shikoku-Section Joint Convention of the Institutes of Electrical and Related Engineers*, p.320, Sep. 2014.

Manabu Sasayama

He received the Ph.D degree in 2008 from Faculty of Engineering, the University of Tokushima. He is currently an a lector at the Kagawa National College of Technology. His research interests include dialogue processing and Natural Language Processing. He is a member of IPSJ.

Kazuyuki Matsumoto

He received the PhD degree in 2008 from Tokushima University. He is currently an assistant professor of Tokushima University. His research interests include Affective Computing, Emotion Recognition, Artificial Intelligence and Natural Language Processing. He is a member of IPSJ, ANLP, IEICE and IEEJ.