MULTIPLE LYRICS ALIGNMENT: AUTOMATIC RETRIEVAL OF SONG LYRICS

Peter Knees [1]
peter.knees@jku.at

Markus Schedl [1,2]
markus.schedl@jku.at

Gerhard Widmer [1,2]
gerhard.widmer@jku.at

[1] Department of Computational Perception
Johannes Kepler University Linz
A-4040 Linz, Austria

[2] Austrian Research Institute for Artificial Intelligence (OFAI)
A-1010 Vienna, Austria

ABSTRACT

We present an approach to automatically retrieve and extract lyrics of arbitrary songs from the Internet. It is intended to provide easy and convenient access to lyrics for users, as well as a basis for further research based on lyrics, e.g. semantic analysis. Due to the fact that many lyrics found on the web suffer from individual errors like typos, we make use of multiple versions from different sources to eliminate mistakes. This is accomplished by Multiple Sequence Alignment. The different sites are aligned and examined for matching sequences of words, finding those parts on the pages that are likely to contain the lyrics. This provides a means to find the most probable version of lyrics, i.e. a version with highest consensus among different sources.

Keywords: Lyrics, Web Mining, Multiple Sequence Alignment.

1 INTRODUCTION

People like to sing their favorite songs or at least like to know what they are hearing. Because it is often hard to understand all of the words in a song, it is convenient to have them in a written form. The vast number of online lyrics portals is a response to this. However, even though the portals have large numbers of available lyrics in their databases, none is complete. Another fact is that very frequently the lyrics differ among the portals, e.g. due to simple typos, different words, or different annotation styles. As a consequence, a user may be forced sometimes to examine different sources and to figure out the "correct" lyrics. A simplified way to do so consists of meta-searching lyrics portals (and the rest of the web) by simply using a standard search engine like Google. Although this provides a fast way to find lyrics, an effort has to be made to obtain and compare them.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

©2005 Queen Mary, University of London

In this paper, we suggest a technique to automatically extract the lyrics of arbitrary songs from the Internet. Furthermore, the method presented provides a means to find the most probable version of lyrics, i.e. a version with highest consensus among different sources. Besides enabling users to easily retrieve lyrics, our approach can also serve as a basis for further applications like semantic analysis or automatic karaoke annotation.

2 RELATED WORK

To the best of our knowledge, no previous work on automatic lyrics extraction has been published yet. Nevertheless, work concerning the exploitation of semantics of lyrics in Music Information Retrieval exists. Scott and Matwin (1998) use two sets of more than 400 folk songs for text categorization experiments. Extending the traditional *bag-of-words* approach by integrating WordNet hypernyms, classification accuracy can be improved. In Baumann and Klüter (2002) ontology-based document retrieval is applied to characterize lyrics. Using this representation, similarities between songs based on the corresponding lyrics are evaluated. In Logan et al. (2004) about 16 000 lyrics are gathered and used to determine artist similarity. To analyze and compare the semantic content, Probabilistic Latent Semantic Analysis is applied.

From an abstract point of view this work is also related to the field of automatic text summarization, e.g. Radev et al. (2002). Since the central idea is to extract an interesting section from many similar documents and to remove unnecessary parts, the goal of this work could also be formulated as a multi-document summarization task. However, current off-the-shelf approaches are not applicable directly to this specific task.

Apart from scientific research on semantic lyrics analysis, supportive and convenient applications that assist users in the retrieval of lyrics are freely available on the Internet. The most notable among them is *EvilLyrics*[1]. EvilLyrics is capable of cooperating with the most popular music players and searches for lyrics as a song is played. For given artist and track name, the result of a Google query is examined for known lyrics providers. For each known lyrics page a filter consisting of characteristic sequences to determine start and end of the lyrics on the page has been written. The user is then presented with

[1] http://www.evillabs.sk/evillyrics/

the bare lyrics and can choose between multiple versions from different web pages. This is a handy utility which received a lot of positive feedback from users, indicating that automatic retrieval of lyrics is a desired feature for many people. Although results are displayed quickly and most are useful, this approach suffers from many drawbacks. The first is that the filters to extract the content are simply based on key sequences for each page. Presumably, most changes to the structure of the pages require the filter to be adapted manually. Furthermore, the retrieval is limited to a set of known lyrics portals, making it infeasible to exploit pages focusing, for example, only on lyrics from one particular artist, like official artist pages or fan pages. Finally, finding the best version of the lyrics is left to the user.

In contrast to this, our approach automatically extracts lyrics without knowledge about the structure of particular web pages. It is capable of using information from all kinds of web pages. Multiple sites are processed and examined for matching sequences of words, finding those parts on the pages that supposably contain the lyrics.

3 ANALYSIS OF LYRICS ON THE WEB

In this section, we want to give an overview of various annotation characteristics in lyrics. We do not claim the following listing to be complete; it is only intended to point out some of the difficulties that occur when comparing lyrics from multiple sources.

- *Different spellings* of words: Beside unintended typos, words can have different morphologic appearances. For example, the slang term *cause* is often found to be written as *'cause*, *'coz*, *cuz*, or even correctly as *because*. Although the semantic content is the same, these variations can not be handled with simple string comparison. Similar observations can be made for numbers (e.g. *40's* vs. *forties*) and censored words (*f**k*).

- *Differences in the semantic content*: These result from the simple fact that not all people contributing to lyrics pages understand the same words. Many perspicuous examples of misheard lyrics can be found on the web[2]. For this reason, most of the lyrics portals offer the possibility to rate the quality of lyrics or to submit corrections.

- *Different versions* of songs: Querying the web with the name of the track may result in a variety of lyrics which are all "correct", but highly inconsistent, e.g. due to cleaned versions or changes over time. Also translations of lyrics can be found frequently.

- *Annotation of background voices, spoken text, and sounds*: For songs where phrases are repeated, for example by background singers, in some lyrics versions these repetitions are written out. Sometimes even comments (*yeah I know*, etc.) or sounds from the background (**scratches**, etc.) are explicitly noted.

[2]http://www.amiright.com/misheard/

- *Annotation of chorus, verses, and performing artist*: Some authors prefer to give meta-information about the structure of the song and explicitly annotate the type of the subsequent section, e.g. *chorus*, *verse 1*, *hook*, *pre-chorus* etc. In duets, the artist to sing the next part is often noted (e.g. *[Snoop - singing]*).

- *References and abbreviations of repetitions*: To keep the lyrics compact, to minimize redundancies and to avoid unnecessary typing effort, lyrics are rarely found to be written completely. Rather, references to earlier sections (*repeat chorus*) and instructions for multiple repetitions (*x4*) are used, which are very difficult to handle for a machine, since they can occur in variable form (*x2*, *(4x)*, *repeat two times*, or even *chorus x1.5*).

Beside these differences in content, also trivial deviations like upper-/lowercase inconsistencies, or bracketing of words have to be handled.

4 METHODOLOGY

Our approach consists in three main steps: gathering the lyrics from the Internet, aligning the lyrics to find the corresponding parts, and producing an output string based on the alignment. The main idea behind the proposed method is to identify and extract the lyrics by finding large segments of text that are common to web pages returned by Google when queried with a song title. The consistent parts among these pages are considered to be the lyrics. The inconsistent rest is usually irrelevant page specific content like advertisements, site navigation elements, or copyright information.

4.1 Gathering the lyrics

The first step we have to perform is to obtain different web pages containing the lyrics. To retrieve pages we send queries of the form *"artist name" "track name" lyrics* to Google. From the retrieved pages we remove all HTML tags, as well as all links, and convert them to lower case, so only the plain content is used for lyrics extraction. We assume most of the resulting pages to contain lyrics or at least to contain a link to a page containing the lyrics. Since the decision on which pages contain the lyrics is a non-trivial task, we examined two approaches to exclude pages without lyrics.

4.1.1 Page selection

The first page selection approach simply collects pages containing the artist's name, the name of the track and the word *lyrics* in their title (in the following *kwit*, for keywords-in-title). Result pages that do not comply with these requirements are scanned for hyperlinks that contain the name of the track. The first linked page having all keywords in the title is then used instead. We examine up to 50 results to find at most ten pages fulfilling the *kwit* conditions.

The second approach (in the following denoted by *tf-corr*, for term frequency correlation) tries to find pages containing lyrics independently of the page title and is

thus intended to be the more general page selection method. To this end, the first ten accessible pages are retrieved. Each page is then transformed into a vector representation by counting the occurrences of all words (except for stopwords) appearing on the page (*term frequency*). Using these vectors, for all pages pairwise correlation is calculated, assuming high correlation for pages containing large portions of similar text. To avoid including a page containing only the hyperlink to the actual lyrics page, all pages (starting with the lowest in the Google ranking) are scanned for links containing the track title. If the average correlation for a linked page with the other pages in the set is higher than for the original page, the linked page is considered to contain the lyrics and is taken instead.

4.1.2 Page preprocessing

First experiences with the alignment algorithm (see below) showed that frequently the matching of multiple sources works well at the beginning of songs and increasingly gets confused after the second verse. This is caused by the fact that some pages simply refer to the first chorus (e.g. *repeat chorus*) while others rewrite the chorus each time. Also, different annotations of repetitive phrases raise a problem here (see Section 3 for a more detailed discussion). For this reason, we decided to perform a preprocessing step of the pages before actually trying to merge them. To this end we make use of the structure in the annotation, if available. The first thing to do is to find a paragraph that is preceded by a line starting with the words *chorus* or *refrain*. Subsequently, all other occurrences of lines starting with *(repeat) chorus* or *refrain* are replaced by this paragraph. After that, we perform an expansion step. Repeating phrases are inserted as indicated. Therefore, we search for occurences of the letter *x* in combination with a digit (e.g. *2x*, *x4*) and for the pattern *repeat* ⟨*digit*⟩ *times*. Insertion is applied for lines (having e.g. *x4* at the end of the line), as well as for paragraphs (*x4* at beginning or end of a paragraph). This improves performance of the following alignment step significantly. Furthermore, it is also advantageous to have the complete written form of a song, instead of an abbreviated version, since further applications, e.g. semantic analysis, may profit from using the actual content rather than using meta-notation.

4.2 Aligning the lyrics

The problem with most pages is that they do not solely contain lyrics. In most cases the lyrics are surrounded by advertisements, informations about the page, links to other lyrics, links to ringtones, or notices concerning copyright. In average, about 43% of the content on pages is irrelevant. To find the actual lyrics, we perform Multiple Sequence Alignment (MSA). This technique is borrowed from Bioinformatics, where it is used to align DNA and protein sequences. For our purposes, we can use it to find nearly optimal matching word sequences over all lyrics pages. Hence, MSA allows to discover the consistent parts in the pages (the lyrics), as well as the inconsistent (the irrelevant text fragments from the sites).

For alignment, we transform the web pages into sequences of words. To simplify the alignment, the words are reduced to a basic morphological version, i.e. all non-letters and non-digits like parentheses, brackets, curly brackets, dots, semicolons etc. are removed and special characters, for example characters with acute or dieresis, are substituted by their basic equivalent (e.g. á, à, ã, â, and ä are replaced by a).

4.2.1 Needleman-Wunsch algorithm

To align two sequences, we use the Needleman-Wunsch algorithm (Needleman and Wunsch, 1970). It is based on dynamic programming and returns the globally optimal alignment of two strings for a given scoring scheme. For our case, we decided to reward matching pairs of words with a high score (i.e. 10) and to penalize the insertion of gaps with a low value (i.e. -1). For mismatching words, we defined a score of 0. Using this configuration, we expect the algorithm to find large coherent segments without gap insertion. For non-matching terms within such a large segment, we want the varying terms to be aligned together rather than gaps in both alignments which might lead to both terms being output in some cases.

The algorithm itself uses a two-dimensional array MAT of size $(n+1)\times(m+1)$, where n is the length (the number of words) of the first sequence A and m the length of the second sequence B. The extra column and the extra row in the array are necessary to enable gaps at the beginning of each sequence. Every entry $MAT_{i,j}$ is interpreted as the optimal score of the partial alignment of $a_1, ..., a_i$ and $b_1, ..., b_j$. Thus, $MAT_{n,m}$ contains the score for the optimal alignment of A and B.

Entries of MAT are computed recursively. To calculate $MAT_{i,j}$ the optimal choice for the next alignment step is made by examining the three following cases (in the given order):

1. Alignment of a_i and b_j. This is equivalent to a diagonal step to the lower right in the array. Thus, the score at position $MAT_{i,j}$ is determined as the sum of $MAT_{i-1,j-1}$ and the score gained through alignment of a_i and b_j.

2. Alignment of a_i with a gap. This is equivalent to a step down. The score at position $MAT_{i,j}$ is then determined as the sum of $MAT_{i-1,j}$ and the penalty for gap insertion.

3. Alignment of b_j with a gap. This is equivalent to a step to the right. The score at position $MAT_{i,j}$ is then determined as the sum of $MAT_{i,j-1}$ and the penalty for gap insertion.

Considering these three options, the one to achieve the highest score is chosen. Substituting the values we chose for our lyrics alignment task, this gives us

$$MAT_{i,j} = max \begin{cases} MAT_{i-1,j-1} & + d(a_i, b_j) \\ MAT_{i-1,j} & - 1 \\ MAT_{i,j-1} & - 1 \end{cases} \quad (1)$$

where

$$d(x,y) = \begin{cases} 10, & \text{if x=y,} \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

After computation of all array entries, a traceback phase is necessary to determine the actual alignment from the scoring matrix. Starting from $MAT_{n,m}$ and depending on the origin of the score in the entries, the alignment is built backwards until $MAT_{0,0}$ is reached.

4.2.2 Multiple Sequence Alignment

The principle of the Needleman-Wunsch algorithm is theoretically easily extendable to more than two sequences. However, for two sequences this algorithm already uses $O(n \cdot m)$ in space and time. For every additional sequence the effort is further multiplied by the length of the sequence and is thus not practicable. To circumvent this problem, we decided to implement a hierarchical alignment, as proposed e.g. in Corpet (1988).

In the following, we will use the term *row* to denote a sequence within a complete alignment and the term *column* to denote the list of words that have been aligned together on a position in the alignment (cf. Figure 1). Furthermore, the term *length* will be used to refer to the number of words in a row (i.e. the number of columns in an alignment), while the term *depth* refers to the number of sequences in a column. For example, the alignment in Figure 1 has depth four.

Next, we will describe the hierarchical multiple sequence alignment. For all pairs of sequences, pairwise alignment using the Needleman-Wunsch algorithm is performed. The pair achieving the highest score is aligned. This step is performed again on the remaining sequences, until all are aligned (in case of an odd number of sequences, the last remains unaligned). The principle of pairwise alignment with respect to the highest score is then again applied to the group of aligned sequences, until only one alignment remains. For being able to perform pairwise alignment on pre-aligned sequences consisting of multiple rows, we have to adapt the Needleman-Wunsch algorithm. The basic idea is that words that already have been aligned, remain aligned. Thus, the algorithm must be capable of comparing columns of words, instead of single words. We achieved this by a simple modification of the scoring scheme.

$$MAT'_{i,j} = max \begin{cases} MAT'_{i-1,j-1} & + d_{col}(a'_i, b'_j) \\ MAT'_{i-1,j} & - 1 \\ MAT'_{i,j-1} & - 1 \end{cases} \quad (3)$$

where a'_i is the i^{th} column in alignment A' with depth k, b'_j the j^{th} column in alignment B' with depth l, and

$$d_{col}(x, y) = \sum_{i=1}^{k} \sum_{j=1}^{l} d(x_i, y_j), \quad (4)$$

with x_i denoting the i^{th} word in column x, and y_j the j^{th} word in column y.

4.3 Producing the result

Given the multiple lyrics alignment, we are able to produce a string containing the lyrics. To this end we examine every aligned column and find the most frequent word w in it. If the column's most frequent word is a gap, the column is skipped. Additionally, a threshold parameter t is introduced to determine if w is accepted. If the ratio of occurrences of w in the column and the depth of the alignment is below t, then the column is skipped too.

Before producing the actual output, a preliminary alignment using a threshold t of 0.3 is performed. This alignment is used to *eliminate pages* not containing the lyrics at all. For this purpose, a temporary output is generated as described above, and the agreement of each row in the alignment with the temporary output is computed. This is accomplished by examining every column which contributed to the result and checking for each row if the word in this column matches the result. The agreement of each sequence is then computed as the fraction of matching words and the total length of the result. Sequences with agreement below 0.33 are removed. After this elimination step the alignment is performed on the remaining sequences. In Figure 1 a sample section from one such alignment is depicted. It can be seen that different pages provide different lyrics. Despite the variations in the different sequences, our approach extracts the correct version. For final output generation, the original words are reused instead of the basic morphologic words.Therefore the most frequent version among the agreeing words is chosen.

4.3.1 Smoothing

When looking at the produced results, frequently, words not part of the lyrics can be found at the beginning and at the end. This is one of the negative effects of good aligning, since the algorithm strives to find as many matching words as possible. Therefore, results are often preceded by repetitions of the artist name, the track name, and the word *lyrics*, since these tend to occur on different pages above the lyrics. After the lyrics occurrences of words like *copyright* or *legal* can be observed. To remove such words we perform a smoothing on the output sequence. Words at the beginning and end tend to stand alone in large sequences of gaps, since their matching is rather coincidental. To exploit this finding, we analyze the coherence of the produced sequence. For every word in the output all rows agreeing with the output on this word are consulted. The number of agreements of the five preceding words and of the five subsequent words with the output is summed up. If this sum is below 35% of the number of totally examined words, the word is removed from the output. Although this procedure removes unrelated words in most cases, it is also at risk to remove words at the end or the beginning of coherent sequences, especially if agreement among the different pages is low.

5 EVALUATION

Evaluation of this approach is a non-trivial task, since lyrics can have countless different appearances, which are nonetheless all correct (see Section 3). In fact, the only way to decide if the generated result is correct, is to ask humans (in the optimal case the authors themselves). Since this would be infeasible, we have to be content with a simpler measurement. Therefore, we decided to sim-

...	it's	showtime	-	for	dry	climes	and	bedlam	is	dreaming	of	rain	when	the	hills	...
...	its	show	time	for	dry	climes	and	bedlam	is	dreaming	of	rain	when	the	hills	...
...	it's	showtime	-	for	dry	climes	and	bedlam	is	dreaming	of	rain	when	the	hills	...
...	it's	showtime	-	for	drag	lines	and	bedlam	is	dreamin'	of	rain	when	the	hills	...
...	it's	showtime	-	for	dry	climes	and	bedlam	is	dreaming	of	rain	when	the	hills	...

Figure 1: Section from alignment of song "Los Angeles is burning" by "Bad Religion". The four rows on the top are word sequences extracted from the web, the row at the bottom is the result obtained with any threshold value t below 0.75.

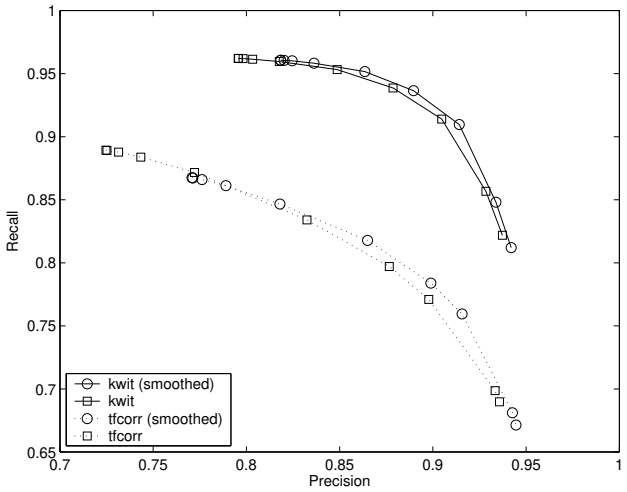

Figure 2: Precision-Recall-Plot for both page selection approaches (smoothed and unsmoothed). The leftmost point of each curve is obtained with threshold t=0.0, the rightmost with t=1.0.

ply compare our results to an "official" reference – lyrics printed in the booklets of CDs. The drawback of this method is that we have to expect the lyrics to be exactly like in the booklets, which also includes annotations of sections, references to sections and abbreviations. Also typos found in the booklet lyrics are transferred one-to-one to our reference set. Thus, experimental results have to be interpreted with these constraints in mind. Since compilation of such a test set involves the exhausting tasks of manual verification and typing, we use a small set limited to 258 songs from various genres[3].

To compare the multiple lyrics alignment result (mla result) and the reference lyrics, we simply reused the Needleman-Wunsch algorithm for optimal pairwise alignment. In contrast to the lyrics alignment, this time we were not interested in alignment of mismatching words. Therefore, instead of adapting the scoring scheme, we treat mismatching words as insertion of a gap in both sequences. Having the optimal alignment between retrieved and reference lyrics we can derive the characteristic information retrieval metrics *precision* and *recall*. Since precision is a measure of how well the approach performs in not adding non relevant words to the result, in our case, this is indicated by gaps in the aligned reference lyrics sequence. Recall is a measure of how well the approach performs in including the relevant words. This can be derived from the number of gaps in the aligned mla result sequence. Hence, we define them as follows, using len to denote the length of the alignment.

[3]the list of used lyrics is available online at http://www.cp.jku.at/people/knees/publications/258lyrics.html

Table 1: Absolute numbers of lyrics (from total 258) in specified recall intervals for values of t. Results were achieved using the unsmoothed kwit method.

	threshold t										
	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Rec=1.00	61	61	61	61	61	61	53	42	33	22	14
1.00>Rec≥0.98	95	95	95	95	95	90	82	69	54	34	29
0.98>Rec≥0.95	46	46	46	46	46	49	52	54	65	43	28
0.95>Rec≥0.90	20	20	20	20	20	21	30	42	36	51	50
0.90>Rec≥0.80	30	30	30	30	30	30	26	32	40	51	63
0.80>Rec≥0.65	5	5	5	5	5	5	12	13	17	30	34
0.65>Rec>0.00	0	0	0	0	0	1	2	5	12	26	39
Rec=0.00	1	1	1	1	1	1	1	1	1	1	1

$$Prec = 1 - \frac{|gaps_{ref}|}{len} \tag{5}$$

$$Rec = 1 - \frac{|gaps_{mla}|}{len} \tag{6}$$

To uncover the impact of the threshold t on precision and recall, we conducted systematic experiments, varying t from 0.0 (no lower bound for most frequent word in column to be chosen) to 1.0 (accept exclusively words that appear in all of the aligned sequences) in steps of 0.1. For each of the page selection approaches (kwit and tfcorr) and both unsmoothed and smoothed outputs, we calculated the average precision and recall over all 258 lyrics for all values of t. Results are visualized in a Precision-Recall-Plot in Figure 2.

Obviously, the kwit page selection approach performs significantly better than tfcorr. For values of t ranging from 0.0 to 0.5 the average recall reaches about 0.96, for 0.6 it is still around 0.95 using kwit. For higher values of t recall further decreases. Since retrieving complete lyrics is much more important than presenting no unnecessary words, in this task, recall is the crucial measure. Trying to keep a high recall value, while getting as much precision as possible, a value of around 0.5 or 0.6 for t seems reasonable. Furthermore, we can state that, at least for the

kwit approach with values of t below 0.8, smoothing increases precision without noticeable loss in recall. In fact, the curve with the smoothed kwit result is basically a right shifted version of the unsmoothed kwit curve.

To give further evidence for the high performance of the approach, Table 5 shows the absolute numbers of lyrics in specified recall intervals. For values of t from 0.0 to 0.5 the number of lyrics with a recall higher or equal than 0.95 is above or equal to 200 (from a total of 258), for t=0.6 it is still 187. For all values of t, there is one piece (*David Hasselhoff - Torero-Te Quiero*) with recall of 0, since no useful pages have been found for it.

6 CONCLUSIONS

We have presented an approach to automatically extract lyrics for arbitrary songs from web pages. Evaluation of the extracted lyrics on a set of 258 "official" lyrics taken from CD booklets yields a median recall above 98%. Using a hierarchical Multiple Sequence Alignment technique, we can find large coherent sequences in the pages with reasonable effort. Based on the alignment, the most probable sequence is determined by voting on each word. In contrast to existent applications, our method does not require explicit knowledge about the structure of the sites and is thus able to incorporate a broader variety of pages.

However, this approach also entails some disadvantages. As the results for the tfcorr page selection method show, the approach would strongly benefit from higher robustness against unrelated or hardly alignable pages. Hardly alignable means pages that actually contain the lyrics, but also other information, i.e. guitar tabs. This raises a problem, since the lines alternately contain chords and lyrics. Improvements could be achieved by applying methods from the field of co-derivative document identification, e.g. Bernstein and Zobel (2004).

Also the fact that important words are possibly omitted, poses a problem. Since the approach is intended to provide users with convenient access, returning complete results is mandatory to gain high acceptance among users. This can not be achieved with a simple local voting strategy as the one proposed here, making thus more sophisticated word selection approaches necessary to produce output with higher consistency. Furthermore, some of the omissions made are caused by improperly aligned sequences, suggesting that improvements could also be achieved by involving more complex MSA approaches. Taking another look at the field of Bioinformatics reveals that strategies have been evolved incrementally in the last years, leading to more powerful, but also more elaborate alignment techniques. Proposals for further enhancements can be found e.g. in Thompson et al. (1994), which covers interesting approaches like sequence weighting and position-specific gap penalties.

For future improvements also better multilingual support is desirable. In principle, the proposed technique is one-to-one applicable for lyrics in languages other than English, as has been proved for three lyrics by the German Hip Hop group *Absolute Beginner*, since it is completely free of language specific constraints. However, it could be valuable to perform searches using Google with keywords other than *lyrics* (e.g. *letras* or *songtext*) to yield more localized results. The proper choice for the additional keyword could then be accomplished by determination of frequent languages in the results.

To give consideration to the user, also the appearance of the output must be adapted. At the moment, output consists of a sequence of words without any line breaking or punctuation. To remedy this, an alignment of the result with the highest matching sequence could be performed to find probable positions for insertions. Thus, the underlying technique could be further obscured from the user by presenting results looking like manually produced lyrics.

ACKNOWLEDGEMENTS

This research is supported by the Austrian Fonds zur Förderung der Wissenschaftlichen Forschung (FWF) under project numbers L112-N04 and Y99-START, and by the EU 6th FP project SIMAC (project number 507142). The Austrian Research Institute for Artificial Intelligence is supported by the Austrian Federal Ministry for Education, Science, and Culture and by the Austrian Federal Ministry for Transport, Innovation, and Technology.

REFERENCES

S. Baumann and A. Klüter. Super-Convenience for Non-Musicians: Querying mp3 and the Semantic Web. In *Proc. of the 3rd International Conference on Music Information Retrieval (ISMIR)*, Paris, France, 2002.

Y. Bernstein and J. Zobel. A Scalable System for Identifying Co-Derivative Documents. In A. Apostolico and M. Melucci, editors, *Proc. of the String Processing and Information Retrieval Symposium (SPIRE)*, pages 55–67, Padova, Italy, 2004. Springer LNCS 3246.

F. Corpet. Multiple Sequence Alignment with Hierarchical Clustering. *Nucleic Acids Research*, 16(22):10881–10890, 1988.

B. Logan, A. Kositsky, and P. Moreno. Semantic Analysis of Song Lyrics. Technical Report HPL-2004-66, HP Laboratories Cambridge, 2004.

S. B. Needleman and C. D. Wunsch. A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins *Journal of Molecular Biology*, 48(3):443–453, 1970.

D. R. Radev, A. Winkel, and M. Topper. Multi Document Centroid-based Text Summarization. In *ACL 2002*, Philadelphia, PA, July 2002.

S. Scott and S. Matwin. Text Classification using WordNet Hypernyms. In S. Harabagiu, editor, *Use of WordNet in Natural Language Processing Systems: Proceedings of the Conference*, pages 38–44. Association for Computational Linguistics, Somerset, New Jersey, 1998.

J. D. Thompson, D. G. Higgins, and T. J. Gibson. CLUSTAL W: Improving the Sensitivity of Progressive Multiple Sequence Alignment through Sequence Weighting, Position-Specific Gap Penalties and Weight Matrix Choice. *Nucleic Acids Research*, 22(22):4673–4680, 1994.