# The MODIS software for word like motif discovery and its use for zero resource audio summarization

Nathan Souviraà-Labastie, Laurence Catanese, Guillaume Gravier, Frédéric Bimbot


# The MODIS software for word like motif discovery and its use for zero resource audio summarization

Nathan Souviraà-Labastie , Laurence Catanese , Guillaume Gravier
Frédéric Bimbot

TECHNICAL REPORT

N° 439

July 2013

Project-Team
PANAMA/TEXMEX

ISSN 0249-6399

# The MODIS software for word like motif discovery and its use for zero resource audio summarization

Nathan Souviraà-Labastie[1] , Laurence Catanese [2], Guillaume Gravier[3]
Frédéric Bimbot[4]
Project-Teams PANAMA/TEXMEX

Technical Report N° 439 — 26/07/2013 —18 pages.

**Abstract:** MODIS is a free audio motif discovery software developed at IRISA Rennes. Motif discovery is the task of discovering and collecting occurrences of repeating patterns in the absence of prior knowledge, or training material. In the case of speech, those motifs could be word since MODIS is tolerant to motif variability. The algorithm implementation allows to process large audio streams at a reasonable speed where motif discovery often requires huge amount of time. It may therefore find many uses, such as summarization of large databases.

**Key-words:** Motif discovery; acoustic summary; speech processing, free software

[1] IRISA/Université de Rennes 1 – nathan.souviraa-labastie@irisa.fr

[2] INRIA Centre de Rennes – Bretagne atlantique – laurence.catanese@inria.fr

[3] IRISA/CNRS UMR 6074 – guig@irisa.fr

[4] IRISA/CNRS UMR 6074 – frederic.bimbot@irisa.fr


RESEARCH CENTRE
BRETAGNE ATALANTIQUE

Campus universitaire de Beaulieu
35042 Rennes Cedex France

# Le logiciel de découverte de mots clefs MODIS et son utilisation pour générer des résumés audio non-supervisés

**Résumé :** MODIS est un logiciel gratuit de découverte de motif audio développé à l'Irisa. La découverte de motif est la tâche qui consiste à découvrir et collecter les occurrences des motifs qui se répètent en l'absence de connaissance a priori ou de données d'apprentissage. Dans le cas de la parole, ces motifs seraient alors des mots. La tolérance de MODIS à une certaine variance des motifs permet son utilisation sur de la parole. L'implémentation de l'algorithme permet de traiter des flux audio de grande dimension en un temps raisonnable alors que la découverte de motif requiert généralement beaucoup de temps de traitement. Ainsi, il peut répondre aux contraintes d'un bon nombre de tâches, comme par exemple celle du résumé automatique.

**Mots clés :** Découverte de motif ; résumé acoustique ; traitement de la parole, logiciel gratuit

**1. INTRODUCTION ..... 7**

**2. MODIS SOFTWARE ..... 8**

2.1. GENERAL OVERVIEW OF MODIS ..... 8
2.2. IMPLEMENTATION ..... 10
*2.2.1. Analysis of the software behaviour ..... 10*
*2.2.2. Improvements ..... 11*
2.3. IMPROVEMENTS EVALUATION ..... 11
*2.3.1. Experimental Protocol ..... 11*
*2.3.2. Performance Indicators ..... 12*
*2.3.3. Results ..... 12*
*2.3.4. Intrinsic performances ..... 13*
*2.3.5. CPU time performance ..... 13*

**3. SPEECH DATA SUMMARIZATION ALGORITHM ..... 14**

3.1. KEYWORD LIBRARY BUILD ..... 14
3.2. MAXIMIZING THE RELEVANCE OF THE SUMMARY ..... 15
3.3. A ZERO RESOURCE TASK ..... 16

**4. CONCLUSION ..... 17**

**5. REFERENCES ..... 17**

# 1. Introduction

The amount of audio data grew significantly these past years given the expansion and the velocity of digital media creation and diffusion. Audio compression and digital storage techniques contribute to this evolution by increasing the storage possibility. One of the current challenge is now to handle those huge amounts of information and make it available in a succinct form for human user. Automatic mining tools are required for this purpose. The identification of repeating patterns in the audio data is a promising method to mine audio assuming that repeating extracts carry meaningful information on the main content of the data. Such patterns can be directly used to summarize for instance speech data that is one of the most meaningful audio signal for human. Recent works have already studied the motif discovery problem. A pioneering work in word discovery is presented in [1] and an algorithm is proposed in [2] to automatically extract repeating patterns in multimedia stream.

This report presents the open source software MODIS[5] that aims at performing unsupervised discovery of arbitrary repeating spoken patterns in large audio streams. It is based on a generic approach to mining repeating sequences, tolerant to motif variability [3]. Motifs properties like their identity and their number are unknown as well as the number of occurrences, their length and their locations. This unsupervised approach allows the search to be language and topic free. Multiple spoken-document summarization and topic clustering are two possible applications of MODIS. These two ideas are developed respectively in [5] and [6], where repetitions automatically discovered at the acoustic level are processed by matching learning methods. A method to summarize speech data based on MODIS outputs is then described, as well as its use on real data, *i.e.* segmented TV news report. The final results are available under Texmix[6] demo platform.

[5] MODIS: https://gforge.inria.fr/projects/motifdiscovery/

[6] http://texmix.irisa.fr/modis/

# 2. Modis software

This part aims at introducing the software and explaining the functioning for a user. A general overview of the MODIS project is presented, giving the baseline architecture of this audio motif discovery software and its functioning. Secondly, critical points relative to a motif research in large audio streams will be underlined. Indeed, motif discovery algorithms need efficient implementations to keep reasonable computation times. The internal structure has been made to reveal the functionalities that requires a lot of computation time. Some solutions to this issue are presented. Finally, an experimental protocol is presented and a performance evaluation is reported in sub-section 2.3. This software is based on the algorithm presented in [3]. The goal of these last part is to strengthen previously obtained results in [3] and to show the benefits of the implementation choices made regardless of the structure and the code to minimize the computation time.

## 2.1. General overview of MODIS

The general input/output system of MODIS is presented in Figure 1. The feature extraction has to be performed toward a representation suited for the task and the data type. The software handle for example classic mel frequency cepstral coefficient (MFCC) and posteriorgrams. Motif discovery in a stream $X$ can be seen as the problem of finding all the pairs $[a, b]$ and $[c, d]$ in $X$ such as:

- $H(X_a^b, X_c^d) < \epsilon$ (1)
- $b - a < L_{min}$ (2)
- $a < b < c < d$ (3)

Equation (1) states that two segments $X_a^b$ and $X_c^d$ are occurrences of a motif if the cost function $H()$ applied to the pair of segments is below a given threshold $\epsilon$. Condition (2) imposes a constraint on the motif minimum length while (3) prevents considering overlapping segments as instances of the same motif to avoid trivial matches. The result of the motif discovery process is a library $LB_{i,j}$ of $I$ motifs. Each motif is a set of occurrences of size $m_i$ (at least 2), each occurrence being represented by a starting and an ending point. Output motifs can be words, groups of words or non-linguistic spoken patterns like breathing in the case of speech, and entire song, chorus or even verses with more tolerance on variability, in the case of music. MODIS is developed in C, and used under Linux. As shown on Figure 1, this software requires the SPRO library[7] installed on the system. It is an open library widely used for handling operations of access to audio feature buffers. OpenCV[8] that provides

[7] Spro: http://www.gforge.inria.fr/projects/spro

[8] Opencv: http://opencv.willowgarage.com/wiki/

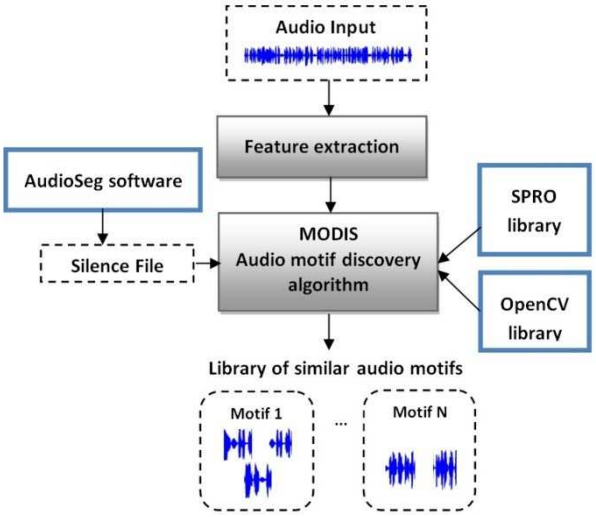


**Figure 1: MODIS software architecture. MODIS software processes an input feature file generated from an audio file. The released output indicates the portions of audio segments that are estimated similar by the system. MODIS relies on SPRO and OpenCV (optional) library A file generated by Audioseg software can optionally be used as an input to process the stream without silences.**

useful image-video processing tools is an optional library that MODIS can use to perform some additional processing like self-similarity matrix computations.

A summary of the global internal architecture is shown in Figure 2. The process is based on a general framework called *seeded* discovery. The incoming stream is decomposed into two adjacent audio segments: a "seed block" which is a potential fragment of a motif, and its "near future" where a potential match is searched. After a fixed length fragment of a motif is found, the final occurrences are grown by their entire length through a match extension procedure. Motifs discovered are stored and a library of motifs is incrementally built. Instead of looking for a seed match in the entire stream, similarities are searched in this library at first, permitting long term occurrences and avoiding search over the entire stream. The search for local similarities between the seed block and a buffer is performed using variations of the popular dynamic time warping (DTW) procedure. The technique used is called "segmental locally normalized dynamic time warping" (SLNDTW). It aims at detecting local alignments relaxing the boundary constraint of the classical algorithm since starting points of potential matches are unknown. In order to improve the robustness to the variability of speech patterns, a template matching technique based on the comparison of self-similarity matrices (SSM) is used on speech sequences. This technique allows confirming or invalidating the similarity between speech patterns complementary to a DTW-based matching and thus provides a more robust system. This technique has shown some benefits although it requires more computation time.

Among other optional features, two main parameters of the audio motif search are left to be defined by the user:

- The length of the seed and the near-future buffer in which the search is performed into can be modifiable. The length of the seed has to be set with respect to the expected size of the output motifs. The computation time generally grows with buffer sizes.
- The threshold $\epsilon$ used for the similarity detection corresponds to the value below which two sequences are considered similar. Changing this value affects precision and recall scores

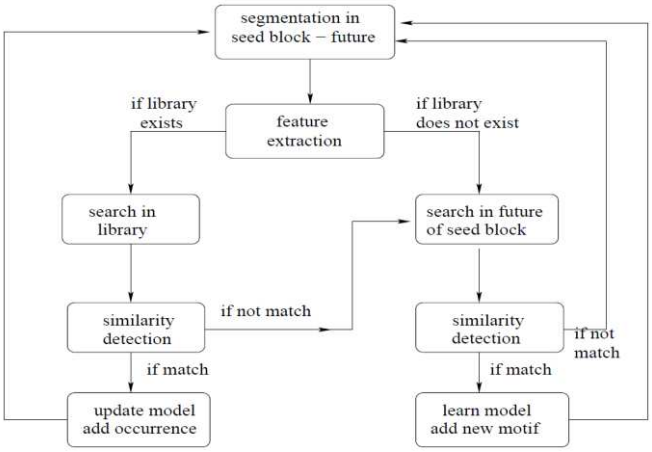


**Figure 2: Seeded discovery algorithm design. At each step the seed block is compared with the models in the library and, if not found, is searched in its near future. At the following step the pair "seed block-future" shifts of a seed block length along the audio stream and the process is iterated.**

## 2.2. Implementation

A modular implementation of the algorithm permits to identify critical parts that make the motif discovery task very long to process. It also ensures the durability of the software. Indeed the functions, could be replaced or completed for potential future developments like other form of DTW for instance.

### 2.2.1.Analysis of the software behaviour

Profiling over the module underlines the main issues relative to the data process. Main results are presented Table 1. The SLNDTW algorithm searches a match between a seed represented by M feature vectors and a buffer (or motif model) represented by N feature vectors. It is called each time a new seed search is made. First, a matrix M by N is filled with the distance between each feature vectors of the seed and the buffer (or motif) where the seed is searched. This step takes the biggest part of execution time (40% on average). Then a second pass is made over the same matrix to compute the corresponding cumulated distances and to find a list of promising endpoints. This seed match search takes approximately 20% of the time computation. The SLNDTW complexity is $O(M * N)$ because every entry in the cost matrix must be filled. The matrix size grows quadratically with the size of the seed and buffer. The function that gives the minimum between three cumulated distances is fast but is called a huge amount of time (actually each time a value of the matrix is computed).

| *Functions* | *Time* |
|---|---|
| Distance matrix filling | 40% |
| SSM computation | 30% |
| Seed match search | 20% |
| Minimum | 8% |
| Others | 2% |

**Table 1: Approximate amount of time used per functionalities**

It is also important to notice that the search of a motif is called in more than 90% of the cases for the library search. Indeed, before searching the seed in the buffer, it is searched in the already existing library of motifs. While no match is found, the seed is compared with every motif of the library whose size grows during the process. The search time grows exponentially. A last remarkable point is the execution time of the functions relative to the self-similarity matrix checking takes almost 30% of the total execution time. In comparison, the seed extension procedure and the storage of the motif in the library are not significant and do not need improvements.

#### 2.2.2.Improvements

Some additional functionalities have been implemented to improve the software performances.

The input sequence can be down-sampled. To process a low resolution representation of the input stream allow saving a huge amount of computation time. The use of this approximate technique is extremely advantageous to recognize pattern with low variability.

An implementation has been made to allow processing the stream without silence segments. The expected output result is a library free from silence patterns and more focused on relevant ones. To this end, MODIS can optionally process an input text file containing silence time codes corresponding to the audio input stream. The format file needed is the one produced by an external audio segmentation toolkit called Audioseg[9]. In the same way, other segmentations can thus be use such as speech/music segmentation.

Some functions have been set in macro type like the much called function that gives the minimum between cumulated distances. The functions used for the computation of self-similarity matrices used a lot of pre-written functions from OpenCV and so are difficult to optimize. Gains have been made concerning the allocation memory management.

### 2.3. Improvements evaluation

#### 2.3.1.Experimental Protocol

To show the software performances, experiments were performed on a 4h subset of the ESTER corpus [4], comprising four different French broadcast news shows (France Inter), sampled at 16 KHz and concatenated. All features are extracted at a 100Hz frame rate. The module is tested with MFCCs, 39 dimensional vectors and two types of posteriorgram: Gaussian posterior features that are 50 dimensional vectors obtained by Gaussian mixture model (GMM) and French state posteriorgram (FRA) which vectors size is 115. Each posterior frame has been computed from the same MFCCs used for the experiments.

For all the experiments, the seed block length is fixed to 0.25 seconds and the search buffer length to 90 seconds (average length of a news report). We also compare both the DTW only system and the

[9] AudioSeg: https://gforge.inria.fr/projects/audioseg/

self-similarity matrix based system and use different types of features. All the tests were made on the same computer whose characteristics are 3.9Gb RAM and a Intel Core2 Duo processor at 3GHz.

### 2.3.2.Performance Indicators

In order to assess rigorously the performances of the algorithm, an evaluation framework at the phonetic level is proposed. Evaluation at the phonetic level can be achieved by relying on the annotations in the form of word transcriptions and phonetic alignments. To identify a true hit in a motif, each occurrence is compared to a reference which is the median occurrence of the motif, defined as the closest to all the other strings in average. The performance indicators are:

- **Nmt**: The number of motifs discovered for which a phonetic representation is available according to the phonetic alignments.
- **P**: The global precision is computed by averaging all the precisions of the motifs in the library. Precision quantifies the capability of the system of detecting true instances of a motif and discarding false ones. The precision of a motif is the fraction of instances close enough to the reference among all strings detected for the motif.
- **Nocc**: Number of motifs yielding an acceptable precision (more than 0.5).
- **R**: The global recall is computed by averaging all the recalls of motifs in the library. The recall of a motif is the fraction of instances of the motif close enough to the median occurrence among all strings in the corpus that are close to the median occurrence.
- **CPU**: computation time required to perform the task.

### 2.3.3.Results

Experiments done in [3] are reproduced here as a baseline already published, to compare results of the algorithm before and after the final implementation in the software. The new version of the algorithm is compared to the old one with a threshold set to 2,5. The corresponding performances indicators are reported in Table 2 and 3. Table 4 shows results of the software for another threshold ($\varphi$ = 2).

| | *Nmt* | *P* | *Nocc* | *R* | *CPU* |
|---|---|---|---|---|---|
| MFCC (dtw only) | 1100 | 0,68 | 665 | 0,37 | 4h54min |
| MFCC (dtw+ssm) | 673 | 0,82 | 482 | 0,40 | 3h39min |
| GMM (dtw only) | 4495 | 0,20 | 872 | 0,58 | 11h18min |
| GMM (dtw+ssm) | 3732 | 0,26 | 966 | 0,58 | 14h38min |
| FRA (dtw only) | 3866 | 0,38 | 1445 | 0,58 | 23h17min |
| FRA (dtw+ssm) | 2733 | 0,50 | 1396 | 0,57 | 26h22min |

**Table 2: Initial Performances of the algorithm with $\varphi$ =2,5**

| | *Nmt* | *P* | *Nocc* | *R* | *CPU* |
|---|---|---|---|---|---|
| MFCC (dtw only) | 1100 | 0,68 | 665 | 0,37 | 2h43min |
| MFCC (dtw+ssm) | 652 | 0,81 | 481 | 0,40 | 3h03min |
| GMM (dtw only) | 4532 | 0,20 | 894 | 0,61 | 4h41min |
| GMM (dtw+ssm) | 3752 | 0,27 | 1009 | 0,57 | 9h10min |
| FRA (dtw only) | 3930 | 0,37 | 1418 | 0,58 | 10h50min |
| FRA (dtw+ssm) | 2735 | 0,51 | 1436 | 0,58 | 8h14min |

**Table 3: Performance of the algorithm implemented in MODIS with $\varphi$ = 2,5**

| | *Nmt* | *P* | *Nocc* | *R* | *CPU* |
|---|---|---|---|---|---|
| MFCC (dtw only) | 84 | 0,79 | 36 | 0,28 | 56min |
| MFCC (dtw+ssm) | 53 | 0,82 | 24 | 0,38 | 59min |
| GMM (dtw only) | 3296 | 0,38 | 1207 | 0,55 | 4h01min |
| GMM (dtw+ssm) | 2778 | 0,44 | 1210 | 0,53 | 5h04min |
| FRA (dtw only) | 2167 | 0,64 | 1371 | 0,52 | 6h15min |
| FRA (dtw+ssm) | 1676 | 0,73 | 1226 | 0,52 | 5h41min |

**Table 4: Performance of the algorithm implemented in MODIS with φ = 2**

#### 2.3.4.Intrinsic performances

To stress the intrinsic performances of the software, we recall the influence of some of the main parameters. The posterior features based on French phone (FRA) report the best results in comparison to the other type of features. Besides good results for the precision, MFCCs allow discovering only few motifs. For comparable values of recall, the average precision over the motifs and the related *Nocc* parameter (number of motif with a high precision) are widely higher than the ones for posterior features based on GMM. The spectral threshold represents the amounts of intra-motif variability that are admitted. The choice of this threshold value is very important to balance the results between a good precision and a good recall. In most of the cases, when φ becomes lower, precision increases because the number of false hits is reduced as the similarity condition becomes more selective. However, it also reduces the capability of collecting motif occurrences with too high variability, which reduces the recall.

The increased values of precision for the SSM-driven system relatively to the DTW-based one indicate that the SSM comparison is successful in retaining the correct hits and discarding the false ones. This is confirmed by the fact that *Nocc* remains almost constant unlike the total number of motifs found. The improvement with respect to the system only based on DTW can reach 13% for MFCC, 7% for the GMM and 14% for the French phone posterirogram.

#### 2.3.5.CPU time performance

Time execution is not necessarily greatly increased by the SSMs checking despite of the fact that the SSMs comparison adds a large amount of calculations. It is also a pruning mechanism that significantly reduces the number of motifs discovered (see Nmt) and the size of the library. Since some type of features are more able to retrieve motifs, the computation time increase as well. However the higher dimension of the corresponding vectors implies a higher computation time too due to the distance matrix computation in each DTW-based comparison.

The DTW threshold is also a critical parameter for the complexity of the approach. The number of validations of the found matches (and so the library size) grows with the threshold value.

We can notice that a lot of execution time has been saved. The algorithm is now able to process the entire four hours long stream in less than 11 hours for the highest value of φ in the worst case. The best improvement concerns the French phone posteriorgram with SSM calculation whose time fall from more than 26 hours to 8 hours.

# 3. Speech data summarization algorithm

Starting from the postulate "What is repeated is meaningful" , we aim at providing an acoustic summary of an audio based on information provided by MODIS. The process can be regarded as the audio equivalence of the key scenes extraction task in video in which the salient structure of the video sequence is grasped by a small number of representative still images.
Since we work on TV reports summarization, MODIS will aim at discovering keywords type motifs.
In this section, we present a concrete applicative framework where MODIS is used. It is divided in two parts, first the keyword library is build based on MODIS software, then an algorithm selects the most informative audio sections to produce the summary. As a proof of concept the final results are display in Texmix[10] demo that permits to navigate in a collection of TV news report.

## 3.1. Keyword library build

MODIS provides a fast and flexible tool to extract for instance keywords of a TV report. Thus, we use MFCC to detect speech repetitions, and no down sampling is applied since the reports are relatively short and we expect the keywords to be variable. However, parameters such as seed size and threshold still need to be tuned.
Previous experiences show that the output libraries contents from different seed size parametrization were complementary. Two discoveries with different seed size have often non-mutual information in their respective output libraries, in both directions. In our case, too short seed size will lead to pollution of output libraries by inspirations or short words such as articles, for instance. We hence choose empirically to process two discoveries, one with a seed size of 500ms and one with 1s seed size. Those values are well adapted to the speed of speech in TV news report.
Once the choice of a set of seed sizes made, an appropriate threshold for each of the chosen seed size is search. We stand for appropriate threshold, a threshold that leads to a library of motifs that is not empty and not corrupted. Usually the more the libraries are large the more they contain false motifs, however we also need a minimum number of keywords to produce a relevant summary. We determine the tradeoff between the precision and the recall empirically for each seed size depending on the length of the audio input (*i.e.* the report), *i.e.* for a 500ms seed size discover, 20 occurrences of motifs for a 2 minutes audio input and for 1s seed size discovery 10 occurrences of motifs for a 2 minutes audio input. The first observation is generalized by the following formula :

$$Expected\ number\ of\ occurrences = \hat{f} = (20\ occ.\ or\ 10\ occ.) * (1 + log10(length(X)/2min)) \quad (4)$$

The thresholds that lead to the closest number of motifs for each seed size are selected:

[10] http://texmix.irisa.fr/modis/

$$LB^{selected} = \text{argmin}_{LB}(abs(\hat{J} - \sum_i m_i)) \quad (5)$$

A large range of threshold are tested to provide good results and to be independent from the audio content and quality. Thanks to the efficient implementation of MODIS, those several discoveries don' t cost so much in term of time and resources. As a final step, the two selected libraries are merged into a single one that will be used to produce the summary. This single output library $LB_{i,j}$ of $I$ motifs of $m_i$ occurrences has a fully equivalent shape to an output library of MODIS.

### 3.2. Maximizing the relevance of the summary

As a simple and efficient manner of selecting intelligible content, the summary is built from a set of portions of the stream to summarize. In our approach, we consider tree main properties of relevance during the build of the summary:

- Its content meaning
- Its length
- And its continuity or clarity

In other words, we want the summary to contain a maximum number of keywords in a minimum total size and with a minimum number of audio sections. The problem is formulate as finding the summary $U_P = \left\{X_{s_1}^{e_1}, \dots, X_{s_p}^{e_p}\right\}$ of the stream $X$ based on the previously discovered library $LB_{i,j}$ such as:

$$U_P = \left\{X_{s_1}^{e_1}, \dots, X_{s_p}^{e_p}\right\} = \arg\max_{U_P} C\ (U_P)$$

$$= \arg\max_{U_P} \frac{\sum_i \delta(LB_i \cap U_p \neq 0)}{I} * \left(1 - \frac{\sum_p (e_p - s_p)}{25sec}\right)^3 * (1 - 0{,}2 * (1 - P)) \quad (6)$$

Where $\delta(condition) = 1\ if\ condition\ is\ true,\ 0\ otherwise$

The criterion $C$ that we want to maximize is composed of 3 sub-criterions which correspond to the tree previously presented properties expected for the summary, *i.e.* presence of keywords, acceptable length and continuity. Each sub-criterions have been tuned empirically to satisfy the listener expectation.

Once this criterion defined, its maximization is proceed as follows: the summary is initialized to empty and the best found criterion to zero. Then for each possible number of motifs we choose the portion that maximizes the criterion (the smallest in this case because the first and the third sub-criterion are fixed). For each chosen $X_{s_1}^{e_1}$, we remove the corresponding motifs from the library and do the same while there is more than one motif in library. At each step, we store the current summary $U_P$ if the criterion is higher than the previously stored one. In our case, only a two depth level research is performed as shown by the description of algorithm 1.

As a final step, the silence segmentation of the stream, provided by AudioSeg, is used to extend the border of the summary, *e.g.* $s_1,\ e_1,\ s_2,\ e_2$, until the next detected silence. Empty 200ms are then added from those new starting and ending points. This avoids odd starts and stops, such as sliced words or expressions. Those extends are also applied before the criterion calculation during the algorithm 1.

**Algorithm 1** Partial search of the summary $U_P^{best} = \{X_{s_1}^{e_1}, X_{s_2}^{e_2}\}$ that maximize $C(U_P^{best})$

$init: U_P^{best} = \{\emptyset\}\ such\ as\ C(U_P^{best}) = 0$

$for\ 2 \leq m \leq I$

$U_P = \{X_{s_1}^{e_1}\} = \arg\max_{U_P} C(U_P)\ with \sum_i \delta(LB_i \cap U_P \neq 0) = m$

$U_P^{best} = U_P\ if (C(U_P^{best}) < C(U_P)$

$LB^{updated} = LB - (LB \cap X_{s_1}^{e_1})$

$for\ 2 \leq k \leq m$

$U_P = \{X_{s_1}^{e_1}, X_{s_2}^{e_2}\} = \arg\max_{U_P} C(U_P)\ with \sum_i \delta(LB_i^{updated} \cap U_P \neq 0) = k$

$U_P^{best} = U_P\ if (C(U_P^{best}) < C(U_P)$

$end$

$end$

$return\ U_P^{best}$

Other postulates can also be considered as starting points such as "What is the most repeated is the most meaningful" or the inverse "what is not repeated is meaningful" . In the first case, the first sub-criterion should be:

$$\frac{\sum_i m_i \delta(LB_i \cap U_p \neq 0)}{\sum_i m_i} \quad (8)$$

This sub-criterion is ideal in the ideal case where there is no false discoveries or non-words discoveries but in practice this method rises this kind of little discovery mistakes. For instance, if one keyword or one non-word audio object is massively detected, the audio summary will be totally corrupted. Summarization techniques based on the last postulate do not belong to discovering redundancy techniques, hence MODIS is not adapted to this kind of approach.

### 3.3. A zero resource task

This task is totally unsupervised and unimodal. It means that motif length, number of occurrences, location in the stream, amount of variability allowed among motif occurrences, are unknown, and no additional clues about the presence of motifs is gathered from video or textual sources. No train and test strategies are used contrary to most supervised models for recognition, such as in modern ASR systems, in which linguistic and acoustic trained models are designed prior to the discovery. We can also highlight that language models are absent of our scenario, while acoustic models are learnt and refined during the actual discovery process. In the end, this approach does not suffer of the out of vocabulary problem, and language dependency. It is also sensibly faster than textual transcription approaches.

# 4. Conclusion

This report presents the MODIS open-source software that permits to mine repetitive patterns in audio content with tolerance to variability. It targets tasks such as multimedia indexing and large archives summarization. As a proof of concept, an empiric algorithm produces acoustic summary of TV report based on keywords library given by MODIS. Using MODIS in this application highlights its adaptability to mine relevant information without any other resources than the input stream and with few computational time and power. The results are promising but still need to be evaluated. For this purpose, the already existing transcription could be a the transcription could give an easy-ot-use ground truth.

The future main challenge for MODIS project is to handle large data sets and tolerate more type of variability among instances of the same underlying pattern. Although the process time grows exponentially with respect to the size of the library built over the time, optimizations allow reducing significantly the computation time keeping in the meantime similar performances for the precisions and recalls.

# 5. References

[1] A. S. Park, Unsupervised pattern discovery in speech: applications to word acquisition and speaker segmentation, Ph.D dissertation, MIT, Cambridge, MA, 2006.

[2] Herley, C. ARGOS: Automatically extracting repeating objects from multimedia streams. IEEE Transactions on Multimedia, 2006

[3] A. Muscariello, G. Gravier, Member, F.Bimbot, "Unsupervised motif acquisition in speech via seeded discovery and template matching combination" . Accepted for publication in IEEE Trans. on Audio, Speech and Language.

[4] Galliano, and al, "The ESTER evaluation campaign for the rich transcription of French broadcast news" Interspeech, 2005.

[5] X. Zhu, G. Penn and F. Rudzicz, "Summarization multiple spoken documents: finding evidence from untranscribed audio" , 47th annual meeting of the ACL and the 4th IJCNLP and of the AFNLP, 2009, 549-557

[6] M. Dredze, A. Jansen, G. Coopersmith and K. Church, NLP on Spoken Documents without ASR, Empirical Methods in Natural Language Processing, 2010


RESEARCH CENTRE
BRETAGNE ATALANTIQUE

Campus universitaire de Beaulieu
35042 Rennes Cedex France

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr
ISSN 0249-6399