Feature Extraction Based on Exponential-Weighted Higher-Order Local Auto-Correlation: An Approach to Improve Data Characterization

MARCELO KEESE ALBERTINI[1]
IVAN STOJMENOVIC[2]
RODRIGO FERNANDES DE MELLO[3]

[1]Federal University of Uberlandia, Faculty of Computing,
Campus Santa Monica, Bloco 1B, Uberlandia – MG, Brazil
[1]albertini@facom.ufu.br,
[2]University of Ottawa, School of Electrical Engineering and Computer Science,
800, King Edward av., Ottawa, Canada
[2]stojmenovic@site.uottawa.ca,
[3]University of Sao Paulo, Institute of Mathematics and Computer Sciences,
Department of Computer Science,
Av. Trabalhador Saocarlense 400, Sao Carlos - SP, Brazil
[3]mello@icmc.usp.br

Abstract. Motivated by complex phenomena embedded into time series, this paper proposes EHLAC (Exponential-Weighted Higher-Order Local Auto-Correlation), an approach to extract features from dynamic data based on polynomial relations over time. The main idea for this new approach is to preprocess data in order to improve modeling performance of different techniques. EHLAC extends the traditional HLAC (Higher-Order Local Auto-Correlation), introducing non-linear transformations in terms of its integrals, what inhibits or highlights the influences of observations within the auto-correlation function, highlighting a wider gamut of data characteristics. This approach is evaluated in a song classification scenario, whose results evidence that EHLAC complements the set of attributes of HLAC and improves modeling performance.

Keywords: feature extraction, high-order local auto-correlation

(Received October 10th, 2012 / Accepted March 3rd, 2013)

1 Introduction

The feature extraction area aims at assisting modeling techniques to represent phenomena by condensing and highlighting important data characteristics into features [21, 14, 10]. Feature extraction is typically considered when data is unstructured; when one needs a compact and meaningful representation of data; or when the direct modeling of raw data provides poor results due to unrelevant information, such as when dealing with texts, images [2] and audio [1, 20].

Features can be basically extracted from two types of scenarios: static and dynamic ones. In the first, the complete data set is available and, then, it is processed to obtain features. The second requires the online extraction of features as data arrives and is made available. The latter is interesting when studying phenomena such as climate change and pattern detection in videos. Those phenomena are typically unstable and influenced by external factors, which require the description of features in terms of dynamic components.

One currently relevant example whose effectiveness is greatly dependable on good-performing online feature extraction methods is the task of classifying songs into genre categories [1]. In that task, every song would be represented by a sequence of amplitude values. In order to conduct the classification, one needs to extract features from songs to represent them. Having such features, classification can be performed, however, when features do not represent songs adequately by providing enough information regarding their genres, the classification results are poor.

Due to this key role played by feature extraction, several well-known approaches have been applied to extract features such as Principal Component Analysis (PCA) [16, 22], Independent Component Analysis (ICA) [11], Wavelets [22], Discrete Cosine Transform [5] and High-Order Auto Correlation (HLAC) [8]. Despite the applicability of those approaches, some of them (e.g., PCA, ICA, and Wavelets) are not adequate to obtain features when data represents unstable phenomena. The main reason is that such data cannot be described in terms of static components. Conversely, HLAC has been widely considered to address the aforementioned scenario, demonstrating good results in several application domains [3, 17, 9, 6, 7]. HLAC employs mask patterns to extract features through the combination of auto-correlation functions. In that sense, HLAC can characterize complex data features in images, audio, etc.

When studying HLAC in dynamical scenarios, we observed that the way HLAC obtains features could be improved. We here propose the Exponential-Weighted Higher-Order Local Auto-Correlation approach (EHLAC), which employs polynomial factors to the terms of HLAC, introducing the possibility of non-linear transformations in every term. EHLAC makes possible to inhibit or highlight the influences of every observation in the auto-correlation, thus emphasizing a wider gamut of data characteristics. Classification experiments confirm this novel approach outperforms HLAC by means of a more informative data representation.

The organization of this paper aims at clarifying the implementation and verification of feature extraction using HLAC and EHLAC by providing code examples and evaluations with respect to analytical aspects. This was motived by sparse HLAC literature regarding its implementation, which may considerably affect the obtained results. This paper is structured as follows: Section 2 presents the original Higher-Order Local Auto-Correlation; Exponential-Weighted Higher-order Local Auto-Correlation (EHLAC) is introduced in Section 3; Experimental results comparing EHLAC against HLAC are shown in Section 4; Conclusions are drawn in Section 5; and, finally, references.

2 Feature Extraction with Higher-Order Local Auto-Correlation

Higher-Order Local Auto-Correlation (HLAC) is a method designed to extract features for a wide range of applications such as video processing and EEG (Electroencephalography) signal recognition [12]. HLAC is based on the computation of auto-correlation functions in order to describe linear relationships among multiple variables. The definition of higher-order correlation is presented in Equation 1, which relates the value of function $f(t)$ at the current time instant t to itself evaluated on N different time displacements $\{\tau_1, \tau_2, \ldots, \tau_N\}$.

$$x_f^{(N)}(\tau_1, \ldots, \tau_N) \triangleq \int_T f(t)f(t+\tau_1)\ldots f(t+\tau_N)dt \quad (1)$$

In order to better understand HLAC, consider a one -dimensional time series $X = \{x_{-\beta}, \ldots, x_{-1}, x_0\}$. Now consider vectors, which are also referred to as masks, whose length vary from two to $\beta + 1$. Every element in this vector can assume the discrete values $\{0, 1\}$. Let the first vector be given by $\begin{pmatrix} 1 & 1 \end{pmatrix}$, the second by $\begin{pmatrix} 1 & 0 & 1 \end{pmatrix}$, the third by $\begin{pmatrix} 1 & 0 & 0 & 1 \end{pmatrix}$, and so on. Every time we go for the next vector, we add an element zero in the middle of the previous one (when odd, you can either use functions floor or ceiling to define the central element). This mask vector informs which are the X observations to be considered by HLAC.

For example, at the beginning, the first mask vector $\begin{pmatrix} 1 & 1 \end{pmatrix}$ is set to the last observations of the sequence, i.e., x_{-1}, x_0, indicating that both will be considered in the auto-correlation. This means we will multiply both terms in the form $x_{-1} \cdot x_0$ and keep the resultant value on variable i_2 (the subscript value 2 indicates the length of the current vector under evaluation). Then, we move the vector backwards one element and set its position on observations x_{-2}, x_{-1}. Consequently, we compute $x_{-2} \cdot x_{-1}$ and add the result to i_2, thus, after going through the whole time series, i_2 will contain the integral of all multiplications for mask vector under length 2. After finishing this stage for vector length 2, the procedure is again conducted considering vector length 3 and consecutively, until the vector length is equal to the series length. After finishing all operations, we have the time series feature vector $\vec{I} = \begin{pmatrix} i_2 & i_3 & \ldots & i_{\beta+1} \end{pmatrix}$.

Program 1 presents the source code, in **R** [4], used to compute HLAC. The function parameters are the one-dimensional time series, the mask vector and the maximum number of series observations to be considered (*maxlag*). Program 2 presents the function *compute_hlac* which generates the mask vectors and apply them to the target time series. The result of this procedure is a feature vector including all integrals up to the maximum number of past observations, which is also referred to as *maxlag*.

```
hlac <- function (series, mask, maxlag=500) {
  i = length(series)
  integral = 0
  while (i >= length(mask) & maxlag > 0) {
    if (i >= length(mask)) {
      product = 1
      pos = i
      for (j in length(mask):1) {
        if (mask[j] != 0) {
          product = product * (series[pos] * mask[j
               ])
        }
        pos = pos - 1
      }
      integral = integral + product
    }
    i = i - 1
    maxlag = maxlag - 1
  }
  integral
}
```

Source code 1: Higher-Order Local Auto-Correlation

```
compute_hlac <- function(series) {
  result = c()
  for (j in length(series):1) {
    mask = rep(0,j)
    mask[1] = 1
    mask[length(mask)] = 1
    result = rbind(result, hlac(series, mask))
  }
  result
}
```

Source code 2: Test code to compute HLAC for any unidimensional series

Observe a comparison of results obtained with the application of the source code previously presented and the theoretical integral (Equation 1), which is rarely available. In this scenario we consider the feature extraction of audio signals, because they correspond to dynamic data, i.e., observations evolve over time. To introduce the effects of HLAC, we consider a first example using sine functions which are commonly considered to represent audio signals. This example is composed of $1,000$ observations of the sine function. One can plot analytical HLAC integrals with the Maxima System [15], as presented in Program 3, and compare it to the results from the sine function. By visual inspection, both results, in Figures 1 and 2, present the same shape for the feature vector, as expected. However, we generated the sequence for the sine function using a given sampling resolution for observations, making the feature scale (amplitudes of values in Figures 1 and 2) become different for analytical and experimental approaches. In this situation, we are more interested in having the same feature shape rather than the amplitudes, although the numerical results tend to approximate to the analytical ones when the sampling resolution are higher.

```
assume(tau > 0)$
f(t) := sin(t);
plot2d(integrate(f(t)*f(t-tau), t, 0, tau),[tau,0,15
     ]);
```

Source code 3: Commands to generate the analytical feature vector using the Maxima System

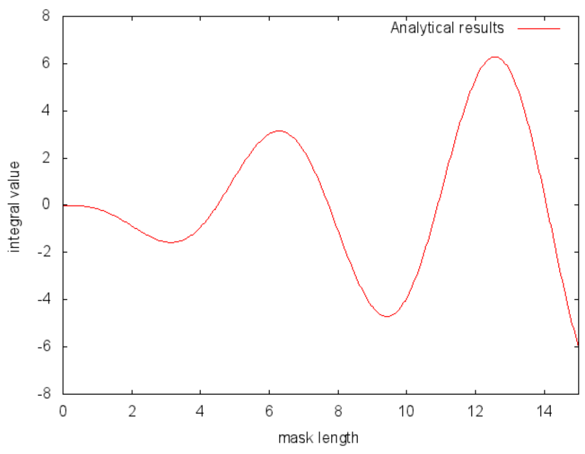

Figure 1: Analytical HLAC Results obtained with Maxima

3 Exponential-Weighted HLAC

The proposal of the Exponential-Weighted Higher-Order Local Auto-Correlation (EHLAC) aims at expanding the type of relations extracted by HLAC by inhibiting or highlighting the influences of certain data characteristics. This is conducted by applying power functions to the terms of HLAC which permit to introduce non-linear transformations in observations before computing the auto-correlation function. EHLAC, defined in Equation 2, is different from HLAC in the sense it introduces a set of weights $\mathbf{w} = (w_0, w_1, \ldots, w_N)$, in which $w \in \mathbb{R}^+$. Every i-th weight, i.e., w_i, is applied to the i-th term of a high-order correlation by raising it to the power of w_i. Thus, we employ Equation 2 using non-isometric combinations of weights in order to assess linear and non-linear features and obtain the

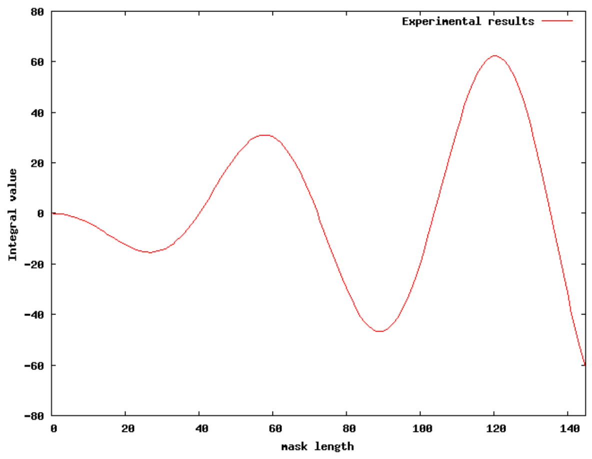

Figure 2: Experimental HLAC Results obtained with source code 1

```
ehlac <- function (series, mask, powermask, maxlag
     =500) {
  i = length(series)
  integral = 0
  while (i > length(mask) & maxlag > 0) {
    if (i >= length(mask)) {
      product = 1
      position = i
      for (j in length(mask):1) {
        if (mask[j] != 0) {
          product = product * ((series[position]
                                 * mask[j])**
                                      powermask[j])
        }
        position = position - 1
      }
      integral = integral + product
    }
    i = i - 1
    maxlag = maxlag - 1
  }
  integral
}
```

Source code 4: EHLAC source code

most representative ones. The right-hand side of such equation computes the correlation in between outputs of function $f(.)$ under inputs $\mathbf{r}, \mathbf{r}+\tau_1, \ldots, \mathbf{r}+\tau_N$, where τ_i, $i \in (1, N)$ are time displacements. A power is applied on every function $f(.)$, modifying the output value by highlighting or inhibiting it. The left-hand side represents the feature itself.

$$x_f^{(N)}(w_0, \tau_1(w_1), \ldots, \tau_N(w_N)) \triangleq \int_T f^{w_0}(t) f^{w_1}(t+\tau_1) \ldots f^{w_N}(t+\tau_N) dt \quad (2)$$

The EHLAC features extend HLAC by allowing non-linear relations to be included in feature extraction for better representation of complex phenomena. In order to better understand EHLAC, consider the function shown in Program 4, which receives a data sequence, a mask vector, a new parameter called *powermask* and finally the *maxlag*. The first, second and fourth parameters are the same as HLAC, and the third one is a vector, such as the mask, responsible for defining the powers for sequence observations. For example, consider the following mask vector $\begin{pmatrix} 1 & 0 & 0 & 1 \end{pmatrix}$. In this situation, only two terms of the series are considered, i.e., the ones in which the vector element is one. When using EHLAC, the same elements must have a power associated, for example, the power mask $\begin{pmatrix} 1 & 0 & 0 & 1 \end{pmatrix}$ would be the simplest one and it would generate the same results as HLAC. However, by modifying its elements, we interfere in observations before computing the auto-correlation.

For instance, consider the same one-dimensional sequence $X = \{x_{-\beta}, \ldots, x_{-1}, x_0\}$, previously presented. Now, let the current mask vector under evaluation be $\begin{pmatrix} 1 & 0 & 0 & 1 \end{pmatrix}$ and the power mask be $\begin{pmatrix} 3 & 0 & 0 & 2 \end{pmatrix}$. When computing integral i_4, every pair of observations x_{t-3}, x_t would be raised to ${x_{t-3}}^3, {x_t}^2$ before being multiplied.

Program 5 presents the function (written in **R** [4]) used to compute features for the sine function. Function *generateFeatures* is responsible for defining the powers and function *evaluate* creates the mask vector and the power mask to compute EHLAC. By calling *generateFeatures()*, we obtain nine feature vectors using the following powers: $(1,1)$, $(1,2)$, $(1,3)$, $(2,1)$, $(2,2)$, $(2,3)$, $(3,1)$, $(3,2)$ and $(3,3)$. The influences of those powers in EHLAC are exemplified in Equation 3 for powers $(3,2)$. In between those pairs, there are zeros indicating that intermediary elements of the mask vector will not be powered to any value (it is important to observe that those elements are not considered in the mask either).

$$x_{(3,2)} \triangleq \int_T sin^3(t) sin^2(t+\tau_1) dt \quad (3)$$

```
generateFeatures <- function() {
  sin = as.ts(read.table("sin.dat"))
  allfeatures = c()
  for (power1 in 1:3) {
    for (power2 in 1:3) {
      feature = evaluate(sin, power1, power2)
      allfeatures = cbind(allfeatures, feature)
    }
  }
  allfeatures
}
evaluate <- function(series, power1, power2) {
  result = c()
  for (j in length(series):1) {
    mask = rep(0,j)
    powermask = rep(0,j)
    mask[1] = 1
    mask[length(mask)] = 1
    powermask[1] = power1
    powermask[length(powermask)] = power2
    result = rbind(result, ehlac(series, mask,
         powermask))
  }
  result
}
```

Source code 5: Test code to compute EHLAC for any unidimensional series

In Figure 3, we present an example on how to generate features (function *generateFeatures*). By comparing them to the results of HLAC (analytical results shown in Figure 1 and experimental ones in Figure 2), we observe the presence of different characteristics in feature vectors. One of them is the same as HLAC, i.e., the pair $(1, 1)$, others inhibit or highlight the influence of sequence observations in the auto-correlation function. In the next section we confirm this influence improves the classification of sequence.

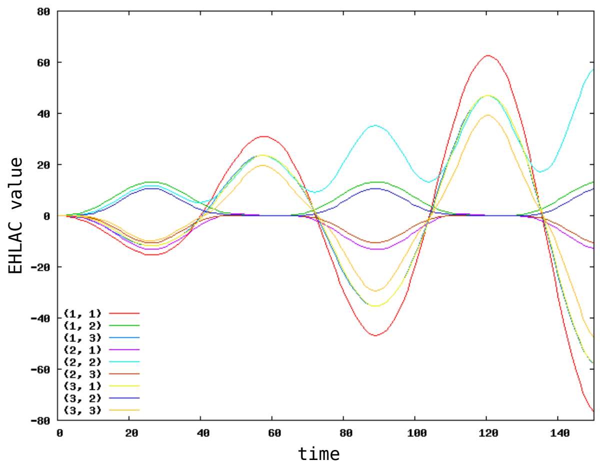

Figure 3: EHLAC results: nine feature vectors were generated using the outputs of the sine function sampling

4 Experiments

In order to evaluate and compare EHLAC against HLAC, we designed two sets of experiments: i) the first aimed at separating useful information from noise; ii) the second was designed to classify information. Those experiments were executed on a set of 32 songs of three different genres: Solitude on Guitar by Baden Powell (12 songs), Obscured by Clouds by Pink Floyd (10 songs) and Killers by Iron Maiden (10 songs). Those songs were converted into the RAW file format, which simply contains bytes, using the Linux command *sox* (Program 6). Songs were originally in the OGG format and, then, they were translated into binary files containing $1,000$-unsigned (option '-u') one-byte (option '-1') samples per second (1 kilohertz, see option '-r 1k'). It is relevant to mention that only the main song channel (first channel) was converted (option '-c 1').

```
sox -t ogg in.ogg -t raw -r 1k -u -1 -c 1 out.raw
```

Source code 6: Linux command used to convert songs

After such conversion, HLAC and EHLAC were applied on 4 intermediary seconds of every song[1]. Thus, those approaches were executed on 4 times $1,000$ samples per second, composing a sequence with $4,000$ observations. Feature vectors were then obtained and used to conduct the two sets of experiments. Every resultant vector was composed of $4,000$ coefficients or components of features.

In the first set, we extracted nine feature vectors for every song using the following powers: $(1,1)$, $(1,2)$, $(1,3)$, $(2,1)$, $(2,2)$, $(2,3)$, $(3,1)$, $(3,2)$ and $(3,3)$, as previously presented in Section 3. Those nine feature vectors were compared against a white noise generated using the **R** [4] function 'rnorm()'. This set of experiments aimed at confirming that feature vectors present relevant song information other than just random data and, consequently, they can indeed represent songs. The **R** code presented in Program 7 was executed to read the 9-feature vector of every song and compare them against another 9-feature vector of the white noise series using the multi-layer artificial neural network (**R** function 'nnet'). This network is available in **R** and implemented according to [13, 18].

The neural network was trained using 50% of features and tested against the other 50% of data. These first results confirmed that less than 1% (0.88% according to experiments) of every song contains characteristics that fail to be distinguished from noise. This was

[1]This value was chosen experimentally. There is no conclusive result in literature about the ideal length to consider, but most researchers take into account a range in $[3, 20]$ seconds.

a very important first step, which allowed us to confirm that features were representative and, therefore, we could carry on experimenting on real scenarios.

```
# loading the library nnet
library(nnet)
# reading song features
song = as.matrix(read.table("song.features"))
# generating white noise
noise = matrix(nrow=4000,ncol=9, data=rnorm(9*4000))
# classes for classification: song or noise
targets <- class.ind( c(rep("song", 4000),
                        rep("noise", 4000)) )
# defining the dataset to train and test by
# joining song features and noise
dataset = rbind(x, y)
# defines the first 4,000 coefficients to the song
# and the last 4,000 to the noise
# also that 2,000 coefficients of each class (song
# and noise) will be used to train
samp <- c(sample(1:4000,2000), sample(4000:8000,2000
       ))
# training of a neural network, 2 neurons in
# the hidden layer (size), weights randomly
# initialized in between [-0.1, 0.1] (rang),
# the maximum number of iterations was 200 (maxit)
result <- nnet(dataset[samp,], targets[samp,],
               size = 2, rang = 0.1,
               decay = 5e-4, maxit = 200)
```

Source code 7: Comparison of EHLAC against white noise

The second set of experiments compared our approach, EHLAC, against its predecessor HLAC. In this second set, we trained the same multi-layer artificial neural network with the features of 6 out of 12 songs by Baden Powell, 5 out of 10 songs by Pink Floyd and finally 5 out of 10 songs by Iron Maiden (such songs compose the dataset previously introduced). The same 4 intermediary seconds of every song was used to obtain the 9-feature vectors per song. The same **R** code was used in this situation, however, the number of neurons in the hidden layer of the artificial neural network was defined as 25, after a manual search for this parameter.

Figure 4 presents the error of the multi-layer artificial neural network during the training stage (it shows the average and standard deviation for the first 200 iterations – this experiment was executed 5 times in order to obtain the averages and standard deviations). We observe, as iterations perform, the error is still high when classifying songs using HLAC. This happens because HLAC feature vectors still have a high degree of similarity, what jeopardizes machine learning and also the separation of patterns. On the other hand, EHLAC results indicate a good tendency in reducing errors, confirming its features improve data representation and, thus, song classification.

Table 1 presents the classification results (averages and standard deviations for correct classifications) using HLAC and EHLAC. This experiment was executed for 5 times and correct (hits) and incorrect (misses) classification results were counted. We observe EHLAC outperforms HLAC under the same scenario, confirming it aggregates more information in feature vectors and improves learning. The most separable artist is Baden Powell because of the differences in his style. The drawbacks of HLAC were stronger when separating Pink Floyd and Iron Maiden songs, due to they present more similarities.

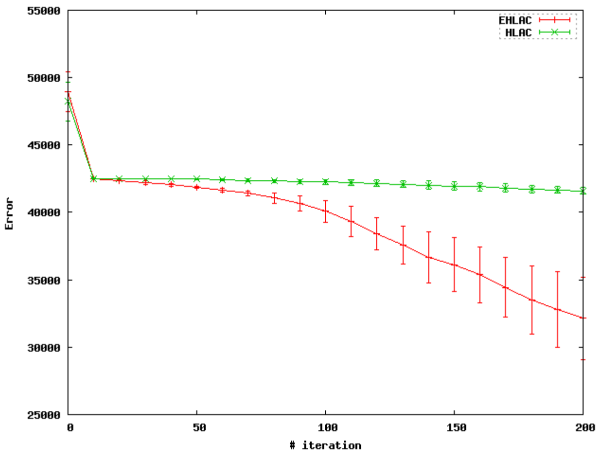

Figure 4: Training error for EHLAC decreases faster than for HLAC

Table 1: Results: hits per artist (average and standard deviation)

Artist	**HLAC**	**EHLAC**
Baden Powell	65.78% ± 1.21%	91.44% ± 3.98%
Pink Floyd	28.05% ± 0.97%	86.32% ± 3.06%
Iron Maiden	26.78% ± 1.98%	66.17% ± 4.97%

After obtaining such results, we decided to employ a Welch Two Sample T-Test, which is an adaptation of the Student's T-Test when two samples possibly have different variances as observed in this situation [19]. Thus, we defined three different hypothesis tests. The first compares EHLAC and HLAC results for Baden Powell songs, the second evaluates our approach in terms of Pink Floyd songs, and the last assesses results for Iron Maiden ones. The tests compare, for a given song set, the null hypothesis of equal average of correct classification using HLAC and EHLAC, against the alternative hypothesis, which states that the averages are different.

Table 2 presents the Welch Test results, which confirm all null hypotheses can be discarded as there is not enough intersection between every pair of distributions. This is evident by the Welch's T results, which confirm a far distance in between averages. Thus, we finally confirm EHLAC aggregates more information in feature vectors and improves results obtained with its predecessor HLAC.

Table 2: Results considering Welch's Two Sample T-Test

Artist	**Welch's T Result**	**Degrees of Freedom**	**ρ-value**
Baden Powell	14.0	4.8	4.7×10^{-5}
Pink Floyd	40.5	4.8	2.8×10^{-7}
Iron Maiden	17.0	5.3	7.3×10^{-6}

5 Conclusions and Future Work

This paper proposed a new feature extraction approach called EHLAC (Exponential-Weighted Higher-Order Local Auto-Correlation) which employs powers to the Higher-Order Local Auto-Correlation approach. By using powers, EHLAC introduces the possibility of non-linear transformations in every term of HLAC, inhibiting or highlighting the influences of observations in the auto-correlation function.

Experiments confirm EHLAC can improve results obtained with HLAC. The first set of experiments corroborates EHLAC aggregates more and important information to features, allowing to separate song features from noise. After this first step, we decided to classify songs according to their features extracted using HLAC and EHLAC. In that scenario, EHLAC strongly outperformed HLAC as confirmed by the hypothesis tests conducted. From that, we conclude EHLAC is useful to obtain information embedded in dynamic data. Although the experiments performed were based on arbitrarily chosen parameters in order to exemplify EHLAC usage, it is possible to devise a greedy algorithm to search for significant power coefficients and higher-order auto-correlations. Such algorithm can, for instance, create new features according to their level of information gain until it reduces to zero. As future work, we also intend to assess the feature space built using EHLAC.

6 Acknowledgments

This paper is based upon work supported by FAPESP – São Paulo Research Foundation, Brazil, under grant no. 2011/19459-8, "CAPES – Coordenadoria de Aperfeiçoamento de Pessoal de Nível Superior" research funding agency under grant no. PDEE-4443-08-0, CNPq – National Council for Scientific and Technological Development research funding agency under grant no. 304338/2008-7. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of FAPESP, CAPES and CNPq. This research is also partially supported by NSERC.

References

[1] Davis, S. and Mermelstein, P. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 28(4):357 – 366, aug 1980.

[2] de Mello, R. F. and Gondra, I. Multi-dimensional dynamic time warping for image texture similarity. In *SBIA*, pages 23–32, 2008.

[3] Hidaka, A., Kurita, T., and Otsu, N. Object detection by selective integration of HLAC mask features. In *BLISS*, pages 46–50, 2008.

[4] Ihaka, R. and Gentleman, R. R – available at: http://cran.r-project.org.

[5] Jing, X.-Y. and Zhang, D. A face and palmprint recognition approach based on discriminant dct feature extraction. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 34(6):2405 –2415, 2004.

[6] Kobayashi, T., Higuchi, T., Miyajima, T., and Otsu, N. Recognition of dynamic texture patterns using CHLAC features. In *Bio-inspired Learning and Intelligent Systems for Security, 2009. BLISS '09. Symposium on*, pages 58 –60, 2009.

[7] Kobayashi, T. and Otsu, N. Image feature extraction using gradient local auto-coorrelations. In Forsyth, D., Torr, P., and Zisserman, A., editors, *Computer Vision – ECCV 2008*, volume 5302 of *Lecture Notes in Computer Science*, pages 346–358. Springer Berlin / Heidelberg, 2008.

[8] Kurita, T., Otsu, N., and Sato, T. A face recognition method using higher order local autocorrelation and multivariate analysis. In *In 11th IAPR International Conference on Pattern Recognition*, pages 213–216, 1992.

[9] Lajevardi, S. and Hussain, Z. Novel higher-order local autocorrelation-like feature extraction

methodology for facial expression recognition. *Image Processing, IET*, 4(2):114 –119, 2010.

[10] Mueen, A., Keogh, E. J., 0002, Q. Z., Cash, S., Westover, M. B., and Shamlo, N. B. A disk-aware algorithm for time series motif discovery. *Data Min. Knowl. Discov.*, 22(1-2):73–105, 2011.

[11] Ozawa, S., Sakaguchi, Y., and Kotani, M. A study of feature extraction using supervised independent component analysis. In *Neural Networks, 2001. Proceedings. IJCNN '01. International Joint Conference on*, volume 4, pages 2958 –2963 vol.4, 2001.

[12] Popovici, V. and Thiran, J.-P. Pattern recognition using Higher-order Local Autocorrelation Coefficients. *Pattern Recognition Letters*, 25(10):1107 – 1113, 2004.

[13] Ripley, B. D. *Pattern Recognition and Neural Networks*. Cambridge University Press, January 1996.

[14] Sameti, M., Ward, R., Morgan-Parkes, J., and Palcic, B. Image feature extraction in the last screening mammograms prior to detection of breast cancer. *Selected Topics in Signal Processing, IEEE Journal of*, 3(1):46 –52, 2009.

[15] Schelter, W. Maxima, a computer algebra system – available at: http://maxima.sourceforge.net.

[16] Song, F., Guo, Z., and Mei, D. Feature selection using Principal Component Analysis. In *System Science, Engineering Design and Manufacturing Informatization (ICSEM), 2010 International Conference on*, volume 1, pages 27 –30, 2010.

[17] Toyoda, T. and Hasegawa, O. Extension of Higher Order Local Autocorrelation Features. *Pattern Recognition*, 40(5):1466 – 1473, 2007.

[18] Venables, W. N. and Ripley, B. D. *Modern Applied Statistics with S*. Springer, 4th edition, September 2003.

[19] Welch, B. L. The generalization of 'Student's' problem when several different population variances are involved. *Biometrika*, 34(1/2):28–35, 1947.

[20] Xiong, Z., Radhakrishnan, R., Divakaran, A., and Huang, T. Comparing MFCC and MPEG-7 audio features for feature extraction, maximum likelihood HMM and entropic prior HMM for sports audio classification. *Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03). 2003 IEEE International Conference on*, 5:V–628–31 vol.5, 2003.

[21] Yuille, A. L., Hallinan, P. W., and Cohen, D. S. Feature extraction from faces using deformable templates. *International Journal of Computer Vision*, 8:99–111, 1992. 10.1007/BF00127169.

[22] Zhu, X., Zhang, L., Wei, J., and Zhou, S. Application of wavelets and principal component analysis to process quantitative feature extraction. In *Control and Automation, 2007. ICCA 2007. IEEE International Conference on*, pages 2609 –2614, june 2007.