# Unsupervised Algorithm for Post-Processing of Roughly Segmented Categorical Time Series

Tomáš Kocyan[1], Jan Martinovič[2], Štěpán Kuchař[2], and Jiří Dvorský[1]

[1] VŠB - Technical University of Ostrava,
Faculty of Electrical Engineering and Computer Science,
17. listopadu 15/2172, 708 33 Ostrava, Czech Republic
{tomas.kocyan,jiri.dvorsky}@vsb.cz
[2] VŠB - Technical University of Ostrava,
IT4Innovations,
17. listopadu 15/2172, 708 33 Ostrava, Czech Republic
{jan.martinovic,stepan.kuchar}@vsb.cz

**Abstract.** Many types of existing collections often contain repeating sequences which could be called as patterns. If these patterns are recognized they can be for instance used in data compression or for prediction. Extraction of these patterns from data collections with components generated in equidistant time and in finite number of levels is now a trivial task. The problem arises for data collections that are subject to different types of distortions in all axes. This paper discusses possibilities of using the Voting Experts algorithm enhanced by the Dynamic Time Warping (DTW) method. This algorithm is used for searching characteristic patterns in collections that are subject to the previously mentioned distortions. By using the Voting Experts high precision cuts (but with low level of recall) are first created in the collection. These cuts are then processed using the DTW method to increase resulting recall. This algorithm has better quality indicators than the original Voting Experts algorithm.

**Keywords:** Voting Experts, Dynamic Time Warping, Time Series

## 1 Introduction

This paper is focused on processing of semistructured data such as text. It is commonly know fact that the amount of this data rapidly grows so that there are two subproblems: how to store this kind of data and retrieve any piece of information from the data. To efficiently store the data it is common to use data compression. Data compression methods [11] can treat the data as sequence of bytes, or they can use additional knowledge about the data, such as knowledge of the language of the data. With this additional knowledge data compression methods can improve their results for example by moving from byte oriented alphabet to alphabet of words. The word alphabet is connection to the second subproblem – information retrieval. Information retrieval algorithms usually do

J. Pokorný, V. Snášel, K. Richta (Eds.): Dateso 2012, pp. 81–92, ISBN 978-80-7378-171-2.

not process the input data as sequence of bytes, but they use even bigger pieces of the data, say words or generally some chunks of the data. This is the main motivation of the paper. How to split the input data into smaller chunks without a priori known structure of the input data? To do this, we use Voting Experts Algorithms in our paper. For test purposes, the Czech and English text was used as test bed for the segmentation algorithm, because the segmentation into words is known without any doubts for Czech or English text so that results of the Voting Experts Algorithm can be easily checked. During the future research, text inputs will be substituted by quantitative time series, such as river measured dicharge volume, and the typical patterns will be searched. These patterns will be further used in Case-Based Reasoning methodology as a input step for prediction.

The paper is organized as follows: in Sect. 2 a brief introduction of Voting Experts algorithm is given. Section 3 describes Dynamic Time Warping post-process of Voting Experts. Experimental results are provided in Sect. 4, and conclusion is given in Sect. 5.

## 2 Voting Experts

The *Voting Expert Algorithm* is a domain-independent unsupervised algorithm for segmenting categorical time series into meaningful episodes. It was first presented by Cohen and Adams in 2001 [3]. Since this introduction, the algorithm has been extended and improved in many ways, but the main idea is always the same. The basic Voting Experts algorithm is based on the simple hypothesis that natural breaks in a sequence are usually accompanied by two statistical indicators [4]: low internal entropy of episode and high boundary entropy between episodes.

The basic Voting Experts algorithm consists of following three main steps[3]:

- Build an nGram tree from the input, calculate statistics for each node of this tree (internal and boundary entropy) and standardize these values in nodes at the same depth.
- Pass a sliding window of length $n$ over the input and let experts vote. Each of the experts has its own point of view on current context (current content of the sliding window) and votes for the best location for the split. The first expert votes for locations with the highest boundary entropy, the second expert votes for locations with a minimal sum of internal split entropy. By this way, the votes are counted for each location in the input.
- Look for local maximums which overcome selected threshold. These points are adepts for a split of sequence.

Tests showed that the algorithm is able to segment selected input into meaningful episodes successfully. It was tested in many domains of interest, such as looking for words in a text [3] or segmenting of speech record [8].

There are several ways how to improve the basic Voting Experts algorithm. Simply we can divide these improvements into the two main groups. On the

[3] For detailed explanation of each of mentioned steps see [4].

one hand, custom "expert" can be added to voting process (for example Markov Expert in [2]) and receive additional point of view on your input. On the other hand, there are methods based on repeated or hierarchical segmenting of the input [5, 9].

One of the simplest ways how to slightly improve performance of segmenting is two-way passing of the sliding window. It means using classic voting algorithm supplemented by segmenting of reversed input. This idea was outlined in [5] which showed the way to make high-precision cut points by selection of higher values of the threshold. Additionally, reversing the corpus and segmenting the reversed input with Voting Experts generates a different set of backward cut points. The subsequent intersection of sets of cut points offers high precision segmenting. However, on the other hand, this high precision causes loss of recall.

## 3 Voting Experts Post-Process

Proposed solution for Voting Experts improvement takes the task of using Dynamic Time Warping algorithm (introduced below) and high precision cuts as a starting point for looking for typical patterns located in the input. Basic idea is to refine the sparse set of high precision cuts into regular sequences as correctly as possible. The mentioned refinement will be done by several types of post-processing methods and the results will be compared.

Methods will differ, but they share a common principle (as shown in Fig. 1). If there are high precision cuts in the input (such as cuts A, B, C a D in Fig. 1) and if the shorter sequence (bounded by cuts C and D) is subsequence of the longer one (bounded by cuts A and B), we can deduce new boundaries E and F by projecting the boundaries of common subsequence to the longer sequence. In this very simplified example the sequences were composed by definite number of values and limited length, so the evaluation is quite straightforward.

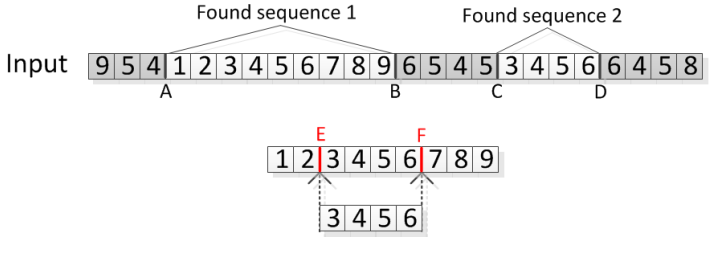


**Fig. 1.** Refinement of high precision cuts

### 3.1 Voting Experts DTW Post-Process

*Dynamic time warping* (DTW) is a technique to find an optimal alignment between two given sequences under certain restrictions [10]. The sequences are

warped in a nonlinear fashion to match each other. First DTW was used for comparison two different speech patterns in automatic speech recognition. In information retrieval it has been successfully applied to dealing with time deformations and different speeds associated with time-dependent data. For our purposes the DTW algorithm will be used as a tool for finding the longest common subsequence of two sequences.

In the case of application of previously mentioned process on distorted data, it is necessary to slightly modify it. Typical episodes of measurement of natural phenomena (such as precipitations, measured discharge volume etc.) are, unfortunately, subject to distortion in both time and value axes. For this reason, it is necessary to find out suitable mechanism that is able to deal with this deformation. The Dynamic Time Warping algorithm can be used for this purpose. The main idea of the Voting Experts DTW post-process is summarized into the following steps:

1. First of all, the high precision (but not complete) cuts are created by splitting the input with high level of threshold by the Two-Way Voting Experts method.
2. Let's suppose that there are $m$ unique sequences which have been created according to cuts from step 1.
3. A $m \times m$ distance matrix is build.
4. For each pair in this matrix, where the length of sequence $s_1$ is bigger than length of sequence $s_2$:
   (a) The optimal mapping of shorter sequence $s_2$ to longer sequence $s_1$ is found by using DTW modified for searching subsequences.
   (b) If the mapping cost does not overcome selected threshold, the longest sequence $s_1$ stores the shorter sequence $s_2$ into its own list of similar sequences. By this way, every sequence gets its own list of the most similar shorter sequences.
   (c) Each of the shorter sequences points to positions in the longer sequence, where it should be splitted. Because there are usually more than one similar shorter sequences, it is pointed to several locations whereas many of these locations are duplicated. For this reason, the votes are collected into internal vote storage.
   (d) After these votes are collected, the local maximums are detected. These places are suggested as new cuts in original input.
5. The granted votes from step 4d are summed with votes of frequency and entropy experts in the input. Subsequently, the local maximums of votes are searched again. The cuts are made in locations where the number of granted votes is higher than the specified threshold.
6. Algorithm ends or it can continue with step 2 for further refinement.

### 3.2 Variations of the proposed algorithm

For our algorithm improvement, several variants of each particular step were proposed and then their influence were tested on final results. The most important variants of the algorithm will be introduced in the following paragraphs.

**Method for finding similar sequences.** The algorithm works only with sequences that do not overcome specified cost threshold of mapping (see step 4b in Sect. 3.1). For this reason, it is necessary to specify this threshold and limit for splitting of long sequences only to reasonable number of subsequences. Test showed that when the length of subsequences is not limited to reasonable length with respect to the length of longer sequence, the input is broken into large number of short sequences. On one hand, it increases the recall, but on the other hand, it rapidly decreases the precision.

To avoid unwanted splitting we used two parameters – divisor and offset. Length of the longer sequence is divided by the *divisor* and then the $offset$ is added. In order to avoid removing the shorter sequence from list of similar sequences, its length has to be longer than the resulting number. Formally we can specify the necessary condition as:

$$\|shorter\ sequence\| > \frac{\|longer\ sequence\|}{divisor} + offset \tag{1}$$

**Method for voting.** Method for granting votes based on mapping shorter sequence to longer one was introduced in Sect. 3.1. Additional three modifications were proposed for this experiment. These methods and their modifications will be further called as $VotingMethod_1$, $VotingMethod_2$, etc. Their principles are described as:

- $VotingMethod_1$: Post-process runs as well as described in Sect. 3.1.
- $VotingMethod_2$: Post-process is the same as in the $VotingMethod_1$, but it is supplemented by boundary condition. This condition limits voting on boundary positions in found sequences. This restriction should avoid situations, in which the high precision cuts have some errors and whole pattern is longer than the area bounded by cuts. Automatic cuts on sequence boundaries may distort further computation, so they are ommited.
- $VotingMethod_3$: First of all, the algorithm groups all similar found sequences and then find the longest common prefixes and suffixes in original input. For example, if the three sequences of value '3456' are found (see Fig. 2), the common prefix is '2' and the longest common suffix is '78'. The resulting common sequence is '2345678'. Subsequently, the same procedure as in the step 1 is performed.
- $VotingMethod_4$: The last method of post-process also looks for common prefix and suffix. But in this case the DTW is substituted by the longest common substring method.

**Method for determining local threshold.** All subsequences that satisfy the condition (1) subsequently vote for places in which they should split the longest (parent) sequence. In this way, several potentional places for cuts are created. Now it is necessary to decide in which locations will be the input really cut. In our experiments, the required threshold was computed in two ways:

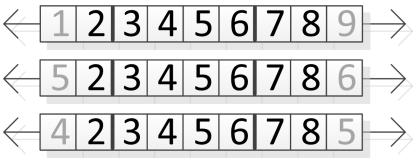


**Fig. 2.** Common prefix and suffix

1. Threshold was chosen as a multiple of maximum of votes (from 0.2 to 0.8).
2. Threshold was chosen as a multiple of nonzero votes average (from 1 to 2).

**Method for determing incrementing value.** Locations in which number of local votes overcomes the threshold are new proposals for cuts. This locations increment votes in original input. The question is how many votes should be these locations incremented. Three various types were suggested:

1. Incrementation of specified constant.
2. Incrementation of frequency with which the sequence appears in input and multiplied by specified constant.
3. Incrementation of multiple by which the threshold was overcome.

## 4 Experiments

Typical test of algorithm's verification of Voting Experts algorithm's performance is searching words in continuous text. In this text, spaces and punctuations (dots, dashes, new lines etc.) are removed and the goal of the algorithm is to put spaces back into correct places. Because the correct placement of spaces in the original text is known, it is very easy to quantify the algorithm's accuracy. To objectively compare the accuracy of suggested improvement with the basic method, experiments were performed on the same type of data, specifically on Jaroslav Hasek's novel named *Good Soldier Svejk*. To compare performance on different languages same text written in English and Czech language was chosen. English version was choosen as a default language, because original algorithm was tested on George Orwell's novel *1984*. Czech version was selected as a different type of language, which is characterized by large amount of possible word suffixes.

### 4.1 Evaluation

For the evaluation of proposed algorithm performace, precision and recall coefficients were defined. *precision coefficient* $P$ and *recall coefficient* $R$ rank among the most often used for the methods that are able to provide relevant documents in the information system. The precision coefficient is understood as the ratio of the amount of relevant documents returned to the entire number of returned

documents. Recall represents the ratio of the amount of relevant documents returned to a given query to the entire amount of documents relevant to this query. In our case, the precision coefficient will be understood as the ratio of the amount of correct spaces induced by algorithm to the entire number of induced spaces. Recall will represent the ratio of the amount of correct induced spaces to the entire amount of spaces in input. In order to simplify information about system effectivity, methods have been created to display precision and recall measured in a 1-dimensional space. One of these methods is Van Risjbergen's *F-measure*[10, 6]:

$$F_\beta = \frac{1+\beta^2}{\frac{\beta^2}{R}+\frac{1}{P}} = \frac{\left(1+\beta^2\right) R\ P}{\beta^2 P + R} \tag{2}$$

where $\beta$ indicates the ratio of significance between precision and recall. For example, when $\beta$ is an even 0.5, it means that the user is twice as interested in precision than in recall and when $\beta$ is an 2, the users interest is vice versa. $\beta$ was set to 1 in our experiment.

Each of the parameter combinations mentioned above were tested and the results were compared with basic algorithm. In all cases we observed percentage improvement of qualitative indicators.

### 4.2 Czech version results

The best configuration of input parameters for the Czech version is described in Table 1.

**Table 1.** The best parameter configuration for Czech version

| | | |
|---|---|---|
| Method for voting | Name | $VotingMethod_1$ |
| Method for finding similar sequences | Threshold | 0 |
| | Delimiter | 3 |
| | Offset | 0 |
| Method for determining local threshold | Name | Nonzero votes avg. |
| | Multiplier | 0.6 |
| Method for determining incrementing value | Name | Increment of constant |
| | Value | 2 |
| Votes threshold | Value | 3 |

Results of the best configuration applied on various input lengths are summarized in Table 2 and graphically in Fig. 3. It is evident that modified algorithm slightly decreases precision $P$ but considerably increases recall $R$ (up to 17.5%). Observed $F$-measure has an average improvement about +5.5%.

### 4.3 English version results

English input version reached the best results with configuration listed in Table 3 and its concrete qualitative indicators are specified in Table 4. In comparison

**Table 2.** Algorithm improvement for various input length of Czech input

| Input length | $F$-measure | Precision | Recall |
|---|---|---|---|
| 1000 | 7.02 | -1.59 | 13.33 |
| 2000 | 8.40 | -0.23 | 17.59 |
| 4000 | 3.81 | -3.67 | 11.84 |
| 5000 | 5.04 | -4.00 | 15.14 |
| 6000 | 5.88 | -3.50 | 16.57 |
| 8000 | 5.65 | -4.53 | 17.18 |
| 10000 | 4.25 | -5.78 | 15.54 |
| 15000 | 4.23 | -5.47 | 15.46 |

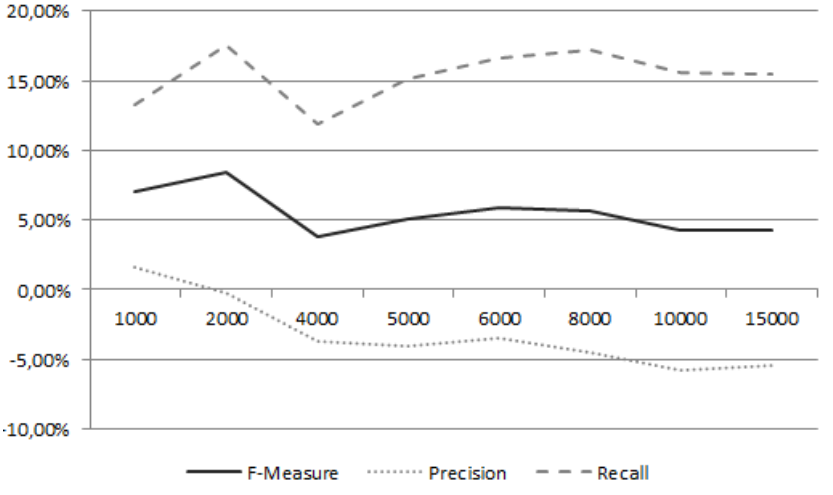


**Fig. 3.** Graphical representation of the best Czech configuration

with the Czech version, the English results are slightly worse. However, the average algorithm improvement of *F-measure* reaches 4.8%.

### 4.4 Overal methods evaluation

In previous sections, only the best configuration of parameters for each language were introduced. Now, overal success of each splitting method ($VotingMethod_1$, $VotingMethod_2$, etc.) regardless of remaining parameters and used language will be compared.

From each of the particular input size results, the top 300 configurations were selected and then the frequency, average frequency and relative frequency of concrete methods in this list were observed. These values are presented in Table 5.

The $VotingMethod_4$ reached the best result in average. This method appears in 42.38% of all selected configurations. It is probably caused by the fact that the

**Table 3.** The best parameter configuration for English version

| | | |
|---|---|---|
| Method for voting | Name | $VotingMethod_4$ |
| Method for finding similar sequences | Threshold | 0 |
| | Delimiter | 4 |
| | Offset | 1 |
| Method for determining local threshold | Name | Nonzero votes avg. |
| | Multiplier | 0.8 |
| Method for determining incrementing value | Name | Increment of constant |
| | Value | 2 |
| Votes threshold | Value | 3 |

**Table 4.** Algorithm improvement for various input length of English input

| Input length | $F$-measure | Precision | Recall |
|---|---|---|---|
| 1000 | 0.15 | -4.40 | 5.30 |
| 2000 | 6.66 | -2.05 | 15.33 |
| 4000 | 4.48 | -3.61 | 12.73 |
| 5000 | 6.78 | -2.23 | 16.15 |
| 6000 | 6.10 | -3.02 | 15.81 |
| 8000 | 4.48 | -3.97 | 13.93 |
| 10000 | 5.34 | -3.91 | 15.34 |
| 15000 | 3.87 | -5.45 | 14.21 |

longest common substring algorithm used in this method is designed specifically for strings. The second-best was the $VotingMethod_3$ with 29%. However, we hope that this method using DTW will overcome the $VotingMethod_4$ while testing on quantitative time series, because it is more robust against distortion in time axis.

**Table 5.** Algorithm improvement for various input length of English input

| *Used method* | *Frequency* | *Average frequency* | *Relative frequency* |
|---|---|---|---|
| $VotingMethod_1$ | 387 | 48.38 | 16.13% |
| $VotingMethod_2$ | 300 | 37.50 | 12.50% |
| $VotingMethod_3$ | 696 | 87.00 | 29.00% |
| $VotingMethod_4$ | 1017 | 127.13 | 42.38% |

### 4.5 Influence of algorithm configuration to results

During the tests we have not observed only the relative increment of $F$-measure, Presion and Recall, but we have also evaluated the influence of values of indu-

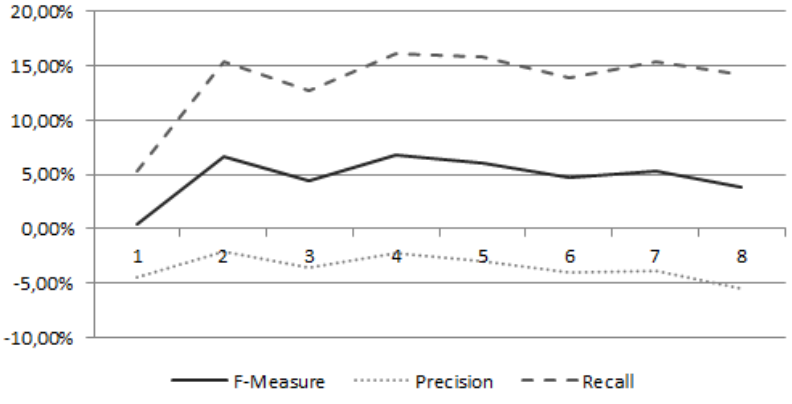


**Fig. 4.** Graphical representation of the best English configuration

vidual parameters to the results. This information will be further important for design algorithm parameters run on quantitative time series.

Test ran on grouped configurations by all parameters without the monitored one and the dispersion of resulting $F$-measure (caused by the ommited parameter) was observed. The dispersions of particular parameters are displayed in Fig. 5.

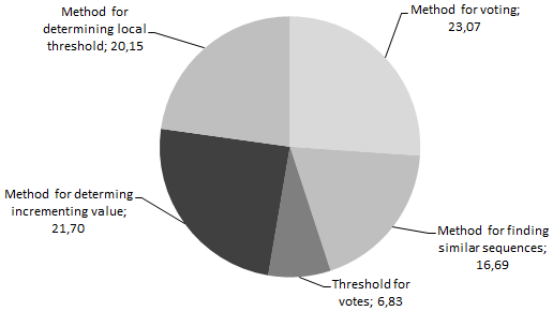


**Fig. 5.** Dispersions of particular parameters

It is evident that the proposed method is sensitive to the choise of the voting method, method for determining local threshold and incremental method. The algorithm is little less sensitive to the method for searching similar sequences and it is insensitive to the value of the threshold.

### 4.6 Reusing on other data collections

Once the best configuration for specific colection type has been found, it can be further reused on this type of input data (text data in our experiments). We verified this idea with the best English configuration on the another two English texts – Gorge Orwell's novel *1984* and Mark Twain's *Adventures of Huckleberry Finn*. In Fig. 6 you can see the results. Both inputs reached better results than the basic algorithm.

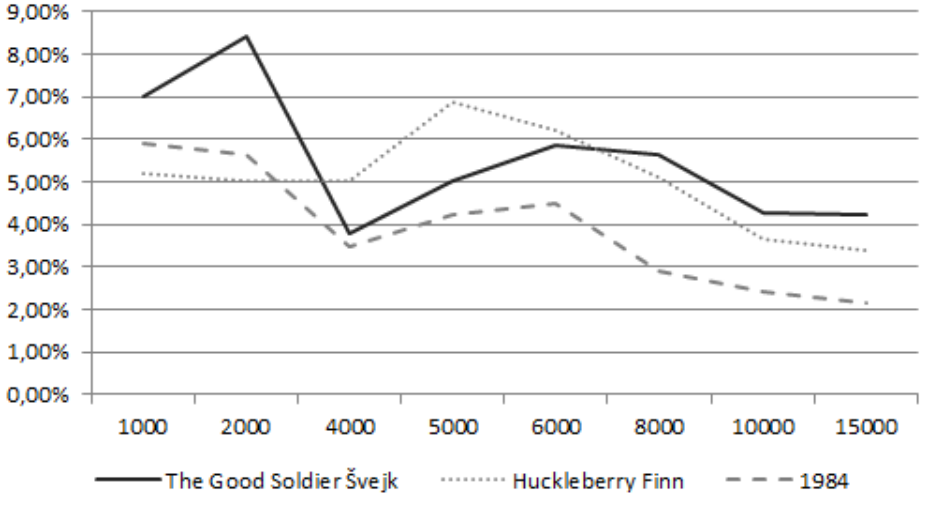


**Fig. 6.** Reusing the best configuration to other data collections

## 5 Conclusion and Future Work

Practical applications of the Voting Experts algorithm showed that it can be used in many domains in which we want to look for some meaningful episodes. Proposed solution overcomes qualitative indicators of original Voting Experts algorithm and offers different point of view to the solution of searching meaningful episodes. Experiments showed that if the proposed algorithm modification is trained to a specific type of data (such as English text, Czech text etc.) it can be further used on various inputs and it should always overcome the basic version of Voting Experts.

Our future work will be focused on searching typical patterns in measured and distorted time series, specifically on searching typical patterns in measured river discharge volumes. This found patterns will be further used for prediction using the Case-Based Reasoning method. This method requires suitable mechanism that is able to extract the most similar patterns from the input.

## Acknowledgement

This work was supported by the European Regional Development Fund in the IT4Innovations Centre of Excellence project (CZ.1.05/1.1.00/02.0070).

## References

1. G. Altmann. Prolegomena to Menzerath's law. Glottometrika 2 (1980). P. 1-10.
2. J. Cheng, and M. Mitzenmacher. Markov Experts. Proceedings of the Data Compression Conference (DCC). 2005.
3. P. R. Cohen, and N. Adams. An Algorithm for Segmenting Categorical Time Series into Meaningful Episodes. Proceedings of the Fourth Symposium on Intelligent Data Analysis, Lecture Notes in Computer Science. 2001.
4. P. R. Cohen, N. Adams, and B. Heeringa. Voting Experts: An Unsupervised Algorithm for Segmenting Sequences. In Journal of Intelligent Data Analysis. 2007.
5. D. Hewlett, and P. Cohen. Bootstrap Voting Experts. Proceedings of the Twenty-first International Joint Conference on Artificial Intelligence (IJCAI). 2009.
6. T. Ishioka. Evaluation of criteria on information retrieval. *Systems and Computers in Japan*, 35(6):42–49, 2004.
7. T. Kocyan, J. Martinovic, J. Dvorsky, V. Snasel. *Czech Text Segmentation Using Voting Experts and Its Comparison with Menzerath-Altmann law.* Computer Information Systems Analysis and Technologies, 2011, 978-3-642-27245-5, pages 152-160.
8. M. Miller, P. Wong, and A. Stoytchev. Unsupervised Segmentation of Audio Speech Using the Voting Experts Algorithm. Proceedings of the Second Conference on Artificial General Intelligence (AGI). 2009.
9. M. Miller, and A. Stoytchev. Hierarchical Voting Experts: An Unsupervised Algorithm for Hierarchical Sequence Segmentation. Proceedings of the 7th IEEE International Conference on Development and Learning (ICDL). (Best Paper Award, ICDL 2008)
10. M. Muller. *Dynamic Time Warping.* Information Retrieval for Music and Motion, Springer, ISBN 978-3-540-74047-6, 69–84, 2007.
11. D. Salomon. Data Compression: The Complete Reference. Springer-Verlag. 2007.
12. B. E. Swartz, and E. S. Goldensohn. Electroencephalography and Clinical Neurophysiology, in Electroencephalography and Clinical Neurophysiology, 106(2), 173 - 176, 1998.
13. C. J. Van Rijsbergen. *Information Retrieval, Second Edition.* Department of Computer Science, University of Glasgow, 1979.