STUDIA INFORMATICA 2013
Volume 34 Number 2A (111)

Aldona ROSNER, Marcin MICHALAK
Silesian University of Technology, Institute of Informatics
Bożena KOSTEK
Gdańsk University of Technology, Audio Acoustics Laboratory

A STUDY ON INFLUENCE OF NORMALIZATION METHODS ON MUSIC GENRE CLASSIFICATION RESULTS EMPLOYING *k*NN ALGORITHM

Summary. This paper presents a comparison of different normalization methods applied to the set of feature vectors of music pieces. Test results show the influence of min-max and Zero-Mean normalization methods, employing different distance functions (Euclidean, Manhattan, Chebyshev, Minkowski) as a pre-processing for genre classification, on *k*-Nearest Neighbor (*k*NN) algorithm classification results.

Keywords: MIR (Music Information Retrieval), music genre classification, normalization, pre-processing, *k*NN

BADANIE WPŁYWU METOD NORMALIZACJI NA WYNIKI KLASYFIKACJI GATUNKÓW MUZYCZNYCH Z WYKORZYSTANIEM ALGORYTMU *k*NN

Streszczenie. Artykuł przedstawia porównanie różnych metod normalizacji zastosowanych do zbioru wektorów cech utworów muzycznych. Wyniki testów prezentują wpływ zastosowania metod normalizacji min-max oraz Zero-Mean z użyciem różnych funkcji odległości (Euklidesowej, Manhattan, Czebyszewa, Minkowskiego) w procesie wstępnego przetwarzania w klasyfikacji gatunków muzycznych z wykorzystaniem algorytmu klasyfikacji *k*-Najbliższych Sąsiadów (*k*NN).

Słowa kluczowe: ekstrakcja informacji muzycznych, klasyfikacja gatunków muzycznych, normalizacja, przygotowanie danych, *k*NN

1. Introduction

Due to the growing demand for the Internet-connected social networks and entertainment, mobile devices, such as smart-phones, netbooks and tablets should provide an easy access to multimedia data the user is interested in. This requires tools that make the search for specific data possible, fast and reliable. More and more frequently the searching parameters are not only metadata (such as e.g.: music / movie title, singer / actor name), but they often represent users' need in the context of their mood to see a specific type of movie or listen to the certain music genre. Since those data are not always available (in the form of metadata information) and multimedia data grows exponentially, it is important to provide a possibility to search for such type of data, e.g.: genre, mood, tempo automatically. To make it happen, it is necessary to retrieve data features from music, which may be then used in the process of classification. This area is called Music Information Retrieval (MIR) and uses knowledge from domains as diverse as music, signal processing, machine learning, information retrieval, etc. Basing on such music content as: rhythm, tempo, appearance of specific instruments, high level features (e.g. mood, singer age, singer gender [1]), etc., it is possible to recommend music based on user's behavior [2][3] or get the humming of the melody [4]. Since such vital feature as genre is nowadays often used for content based analysis [5], thus it is necessary to improve music genre classification , which is still far from being satisfying.

As mentioned before, one of the domain used in the MIR area is machine learning, which consists of supervised, unsupervised, semi-supervised and active learning [6]. Classification is considered as supervised learning, since learning is based on the labeled training examples [7].The most popular methods for music classification are [7]: Support Vector Machine (SVM), Artificial Neural Networks (ANN), Decision Trees, Rough Sets and Minimum-distance methods, to which the *k*-Nearest Neighbor (*k*-NN) method belongs.

An important step in the whole process of any classification method is correct data preparation. Some of the factors influencing correct data preparation are as follows [8]:

- Clearing the input data filtering the data in order to remove noisy or erroneous data;
- Integrating data – integration of various data sources;
- Selecting data (data reduction) – selecting/ retaining the most important attributes;
- Transforming data –in order to approve to further exploration;
- Exploring data – analyzing the data to retrieve useful knowledge for the users;
- Rating data – finding interesting data knowledge schemas/patterns;
- Representing data – visualizing and representing the data to present explored knowledge.

Examples of data transformation steps are: data normalization and data discretization (transformation of continuous data into discrete values or into a finite set of intervals).

In this article the authors present the process of automatic genre classification and the results of the influence of data transformation on exploration/classification performance, where transformation is understood as usage of different normalization methods and different distance function. For the purpose of the conducted tests the *k*-NN was selected as a classification algorithm.

1.1. *k*-NN

The *k*-Nearest Neighbor (*k*NN) method is based on the minimum distance between a test sample and the elements of the training set. Basically, for each object, its *k* nearest objects are found in order to produce the class to which those objects belong. Such an approach precludes errors resulting from mistakes in the training sequence. The idea of distance is associated with metrics defined in the property space. Choosing the adequate metrics is one of the decisions affecting the results.

2. Normalization as a pre-processing for classification algorithms

2.1. Normalization methods

Data normalization is a scaling of original data (e.g. In data) to a specified range defined by an user, which is useful in data exploration and specifically for neural networks. Normalization utilized at the pre-processing stage is also useful in algorithms which theoretically don't need normalization, like *k*-nearest neighbor (*k*NN) algorithm. The *k*NN efficiency depends on pre-scaling original data, otherwise data containing huge values can surpass the data of small values, and thus the metrics can be deformed [2]. The pre-scaling process has to be performed for each dataset (attribute /parameter), so wherever it is referred: min value, max value, mean value, etc. it is meant as a min/max/mean value for a specific attribute/parameter.

The most popular methods of normalizations are: min-Max, Zero-Mean (ZScore), decimal scaling. In this article authors presents the results for the first two (min-Max and Zero-Mean) normalization methods.

2.2. Normalization min-Max

Min-Max normalization is a linear transformation on the original data, usually to the range [0, 1] or [-1, 1]), according to Equation:

$$newV = \frac{V - \min}{\max - \min} * (newMax - newMin) + newMin, \quad (1)$$

where: newV – value after normalization; V – value before normalization, [min, max] – min/max value of original dataset range, [newMax, newMin] – min/max value of data range after normalization.

The advantage of this method is that it doesn't change the relation between values, however a problem may occur when values in test data exceed the original dataset range [2].

In tests presented in this article the original dataset range is calculated on the min-Max values from the training dataset, which means that it may happen that the test dataset values exceed the original dataset range. To avoid this problem, the authors normalize the test dataset using the same data range as in the training set. Resulted from this operation the normalized test dataset can exceed the original range [0, 1], but in this way the relationship between objects is kept (in each dimension) and the scale is also acceptable (e.g. the values of normalized test dataset will be [-0.2, 1.3]). So that we can represent the calculation of normalized data using min-Max method with the following Equations:

$$newTrainV = \frac{TrainV - Train\min}{TrainMax - TrainMin} * (newMax - newMin) + newMin\,, \tag{2}$$

where: newTrainV –value after normalization in train dataset; V – value before normalization in train dataset, [TrainMin, TrainMax] – min/max value of train dataset range, [newMax, newMin] – min/max value of data range after normalization,
in our case: new dataset range is: [0,1], which gives:

$$newTrainV = \frac{TrainV - Train\min}{TrainMax - TrainMin}\,. \tag{3}$$

And the test dataset is normalized using Eq. 4:

$$newTestV = \frac{TestV - Train\min}{TrainMax - TrainMin}\,. \tag{4}$$

As can be seen, the min and max values are the same in both calculation (from the training dataset), and the only difference is the value before normalization (which is from the training dataset/test dataset for respectively calculating value after normalization in both datasets).

2.3. Normalization Zero-Mean

Zero-Mean normalization takes into account the fact that the mean value should equal zero after the normalization process. The data transformation of original data can be calculated with the following Equation:

$$newtV = \frac{V - mean}{std\_dev}\,, \tag{5}$$

where: newV – value after normalization, V – original value before normalization, mean – mean value of the data (for specific attribute), std_dev – standard deviation.

The normalization of training and test datasets are performed in the same way as for min-Max normalization – the mean and standard deviation values are calculated only for training dataset, and only the current value is retained from training and test datasets (used respectively for normalization of training and test datasets).

3. Distance function

This paper presents the results of music genre classification using the *k*NN algorithm, which is based on minimum distance between objects. The most popular distance functions used for *k*NN are: Euclidean, Manhattan, Chebyshev and Minkowski. In this paper these four distance functions were tested as a possible option to improve the classification results.

3.1. Minkowski distance

Minkowski distance (L_m) is a generalized distance between two points [9][11]. In a plane with point p_1 at (x_1, y_1) and p_2 at (x_2, y_2), it is $(|x_1 - x_2|^m + |y_1 - y_2|^m)^{1/m}$ [4] . The specific case of Minkowski distance is Euclidean distance (L_2). Rectilinear (paths, lines, etc. which are always parallel to axes at right angles), Manhattan or Hamming (the distance between two strings) distance is L_1. Chebyshev distance is the maximum distance (L_∞).

3.2. Euclidean distance

The Euclidean distance is a straight line distance between two points. In a plane with p_1 at (x_1, y_1) and p_2 at (x_2, y_2), it is $\sqrt{((x_1 - x_2)^2 + (y_1 - y_2)^2)}$, what in N-dimensional case can be described with Equation:

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \,, \tag{6}$$

where: n – number of dimensions, and x_i, y_i – coordinates of x and y in dimension i.

3.3. Manhattan distance

The Manhattan distance is a distance between two points measured along axes at right angles [10]. In a plane with p_1 at (x_1, y_1) and p_2 at (x_2, y_2), it is $|x_1 - x_2| + |y_1 - y_2|$.

3.4. Chebyshev distance

The Chebyshev (or Tchebyschev) distance is a maximum metric (L_∞), so it is equal $max(|x_1 - x_2|, |y_1 - y_2|)$.

4. Tests results

Tests were run on the vectors of parameters of 2700 music tracks, selected randomly, which represent four music genres: Pop (250 files), Latin Music (250 files), Rap&Hip-Hop (200 files) and Jazz (200 files). Those genres were selected from over 10 genre classes as the ones representing genres which are quite similar (such as: Pop versus Latin) and the ones which differ from each other (Pop vs Rap&Hip-Hop). The feature vector contains 173 attributes and class information (genre), chosen as the representative ones for music tracks in SYNAT database [12]. List of attributes contained in the feature vector are presented in Table 1.

The *k*NN algorithm was chosen as a classification algorithm, and a cross-validation method was used to calculate the average results. The cross-validation split the original dataset (2700 vectors of parameters) into *n* subsets in equal proportions, and for each run (*n-1*) subsets is a training set and the remaining subset are the test set. In our study the original dataset was split into three 900-data subsets: 2 training data and 1 test data subset. The results for each type of settings are the average values from three runs (each time another subset becomes the test set). The test procedure was repeated for different *k* (number of neighbors) value – from 3 to 45 in step of 3.

Table 1

List of attributes

Nr of attributes	Attribute Name	Comment
1	TC	Temporal Centroid
1	SC	Spectral Centroid
1	SC_V	Spectral Centroid variance
29	ASE 1- 29	Audio Spectrum Envelope (ASE) in 29 subbands
1	ASE_M	ASE mean
29	ASEV 1 - 29	ASE variance in 29 subbands
1	ASE_MV	Mean ASE variance
1	ASC	Audio Spectrum Centroid (ASC)
1	ASC_V	ASC variance
1	ASS	Audio Spectrum Spread (ASS)
1	ASS_V	ASS variance
20	SFM 1 - 20	Spectral Flatness Measure (SFM) in 20 subbands
1	SFM_M	SFM mean
20	SFMV 1- 20	SFM variance

Table 1

List of attributes

1	SFM_MV	SFM variance of all subbands
20	MFCC 1-20	Mel Function Cepstral Coefficients (MFCC) – 20 first
20	MFCCV 1-20	MFCC Variance – 20 first
3	THR_1RMS_TOT / THR_2RMS_TOT / THR_3RMS_TOT	Number of samples higher than the values of RMS / double value of RMS / triple value of RMS
3	THR_1RMS_10FR_MEAN / THR_2RMS_10FR_MEAN / THR_3RMS_10FR_MEAN	Mean value of parameter THR_1/2/3RMS_TOT for 10 time frames
3	THR_1RMS_10FR_VAR / THR_2RMS_10FR_VAR / THR_3RMS_10FR_VAR	Variance value of parameter THR_1/2/3RMS_TOT for 10 time frames
1	PEAK_RMS_TOT	A ratio of peak to RMS (Root Mean Square)
2	PEAK_RMS10FR_MEAN / PEAK_RMS10FR_VAR	A mean/variance of PEAK_RMS_TOT for 10 time frames
1	ZCD	Number of transition by the level Zero
1	ZCD_10FR_MEAN	Mean value of ZCD for 10 time frames
1	ZCD_10FR_VAR	Variance of ZCD for 10 time frames
3	1RMS_TCD / 2RMS_TCD / 3RMS_TCD	Number of transition by the level RMS / double level of RMS / triple level of RMS
3	1RMS_TCD_10FR_MEAN / 2RMS_TCD_10FR_MEAN/ 3RMS_TCD_10FR_MEAN	Mean value of parameter 1/2/3RMS_TCD for 10 time frames
3	1RMS_TCD_10FR_VAR / 2RMS_TCD_10FR_VAR/ 3RMS_TCD_10FR_VAR	Variance value of parameter 1/2/3RMS_TCD for 10 time frames
173	**SUM of attributes**	

Table 2

Settings of *k*NN algorithm parameters

Test number	Test name	Normalization	Distance function
1	DistCheb	NONE	Chebyshev
2	DistCheb_NormMinMax	min-Max	Chebyshev
3	DistCheb_NormZMean	Zero-Mean	Chebyshev
4	DistEucl	NONE	Euclidean
5	DistEucl_NormMinMax	min-Max	Euclidean
6	DistEucl_NormZMean	Zero-Mean	Euclidean
7	DistManhattan	NONE	Manhattan
8	DistManhat-tan_NormMinMax	min-Max	Manhattan
9	DistManhattan_NormMean	Zero-Mean	Manhattan
10	DistMink	NONE	Minkowski
11	DistMiink_NormMinMax	min-Max	Minkowski
12	DistMink_NormZMean	Zero-Mean	Minkowski

The tests were conducted using WEKA [13] library for Java [14], version 3.7.5. The evaluation model and implementation of *k*NN (including distance function) and cross-validation methods employed Weka library. The normalization methods are the authors' implementation and they are based on definitions presented in Section 2. Table 2 presents specific test settings of the *k*NN algorithm, including types of normalization and distance function.

The results of tests are presented in Table 3, where *TP* – True Positive Rate and *Prec* – Precision.

True Positive Rate is defined with the following Equation:

$$TP(Id) = \frac{CCP(id)}{TNE(id)} * 100\%, \tag{7}$$

where: *Id* - class identifier, *TP* – percentage of true positives of class *Id*, *CCP* – Correctly Classified Positives of class *Id*, *TNE* – total number of elements in class *Id*, meaning that a class *Id* will be classified correctly with *TP* probability.

Precision is defined as:

$$\Pr ecision(Id) = \frac{CCP(id)}{TCP(id)} * 100\%, \tag{8}$$

where: *Id*, *CCP*, – as above, *Precision* – percent value of precision of class *Id*, *TCP* – total number of objects classified as class *Id* (including false positives), meaning that if an instance *X* is classified as an object in class *Id*, then with probability equal to Precision value it is truly class *Id*.

As Table 3 shows, the best results were obtained using Manhattan distance function with Zero-Mean normalization type. Such configuration enabled the *k*NN algorithm to obtain 62.63% of correct answers in classification, thus having over 10% better results in comparison to settings with the same distance function but without normalization (51.93%), and over 2% better than the default Weka settings for *k*NN (Euclidean distance with Min-Max normalization), which led to 60.59%.

Moreover, the configuration that obtained the highest correctly classified ratio proved to be the most optimum for three out of four tested classes: Pop, Latin and Jazz. The fact that different settings were better for classification of Rap&Hip-Hop genre might be explained by its dissimilarity to the other three classes. However it is worth mentioning that for the best test settings, the precision result for Rap&Hip-Hop was only 0.1% worse than for the best configuration. The improvement of effectiveness in each class for the best test settings is presented in Table 4.

Table 3

Test results for different *k*NN normalization and distance function settings

Total classified		JAZZ				RAP & HIP-HOP				LATIN MUSIC				POP				
correctly	K	K	Prec. [%]	K	TP [%]	K	Prec. [%]	K	TP [%]	K	Prec. [%]	K	TP [%]	K	Prec. [%]	K	TP [%]	Test Settings
48.52	45 18	9	46.06	9	32.50	6	63.35	45	79.00	6	46.38	27	48.93	45	42.12	3	51.47	DistCheb
54.30	30	15	64.07	30	43.17	15	73.84	21	64.83	30	48.18	45	67.47	39	46.78	6	55.07	DistCheb_NormMinMax
52.67	18	12	62.56	12	40.33	9	71.02	21	59.50	12	47.83	42	65.60	18	45.20	3	56.53	DistCheb_NormZMean
50.22	45	6	47.13	36	41.83	3	64.16	39; 42	80.00	45	49.80	45	46.80	45	43.80	3	52.40	DistEucl
60.59	45	12	67.46	39	57.50	6	79.94	18	71.50	3	58.77	42	70.27	45	51.42	3	61.73	DistEucl_NormMinMax
62.59	18	9	68.57	27	57.33	**6**	**80.46**	15; 18	72.17	18	60.88	45	69.73	18	51.47	3	62.53	DistEucl_NormZMean
51.93	45	9	50.15	45	53.00	3	62.76	**39**	**81.17**	45	53.09	24; 27	46.27	45	45.11	3	52.00	DistManhattan
62.15	21	12	68.58	45	59.83	12	80.26	18; 21; 24	72.17	12	60.07	42	70.13	**42**	**51.84**	3	61.47	DistManhattan_ NormMinMax
62.63	**30**	**6**	**70.68**	**45**	**60.33**	9	80.32	21	73.00	**3**	**62.82**	**45**	**70.40**	45	51.69	**6**	**62.80**	DistManhattan_ NormZMean
50.22	45	6	47.13	36	41.83	3	64.16	39; 42	80.00	45	49.80	45	46.80	45	43.80	3	52.40	DistMink
60.59	45	12	67.46	39	57.50	6	79.94	18	71.50	3	58.77	42	70.27	45	51.42	3	61.73	DistMink_NormMinMax
62.59	18	9	68.57	27	57.33	**6**	**80.46**	15; 18	72.17	18	60.88	45	69.73	18	51.47	3	62.53	DistMink_NormZMean
62.63		**70.68**		**60.33**		**80.46**		**81.17**		**62.82**		**70.40**		**51.84**		**62.80**		**BEST**
	30	**6**		**45**		**6**		**39**		**3**		**45**		**42**		**6**		**for K**

Table 4

Comparison of improvement for each class with and without normalization

Class	TP without normalization	TP after normalization	Prec. Without normalization	Prec. After normalization
Pop	52%	62,80%	45,11%	51,68%
Latin	46,27%	70,40%	53,09%	62,81%
Rap& Hip-Hop	81,17%	73%	62,76%	80,31%
Jazz	53%	60,33%	50,15%	70,68%

As can be seen the overall improvement of true positive value varies between 10%-24%, and precision 6-17% for different classes.

Figure 1 shows that for the same *k* parameter the results were significantly better with normalization than without it. The three best *k* values were chosen according to results presented in Table 3.

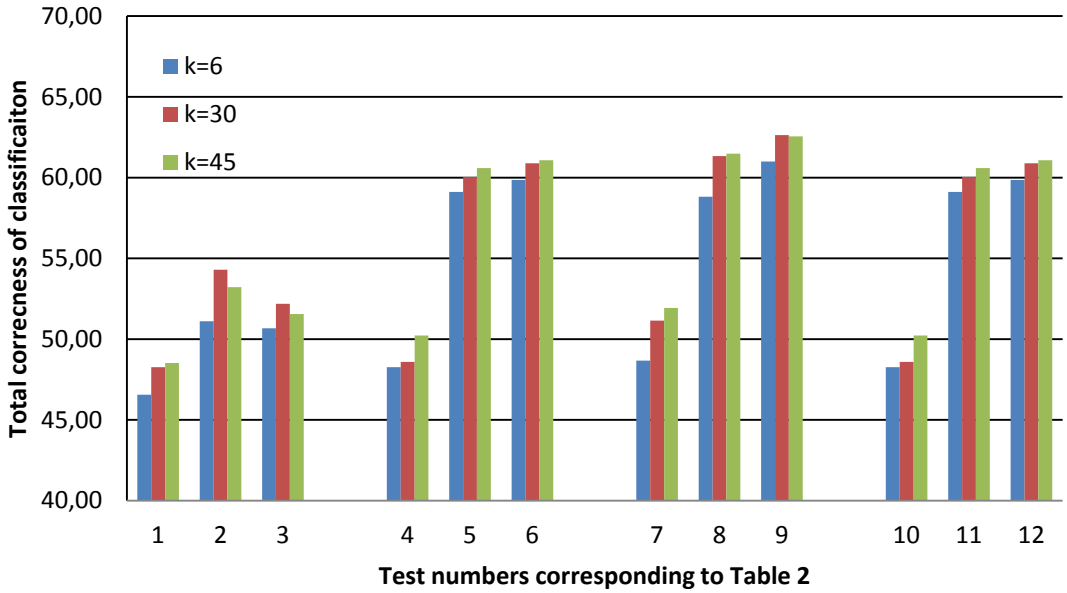

Fig. 1. Results of different normalization methods for the the three best *k* values
Rys. 1. Wyniki dla różnych metod normalizacji dla trzech najlepszych wartości *k*

5. Summary and future work

The test results show that the pre-processing of data normalization is a significant step in the whole process of genre classification. For future research the authors plan to apply and analyze its influence on other classification algorithms including SVM, ANN, decision trees, etc., in order to obtain more satisfying results. These plans have been imposed, among others, by the fact that(as can be seen from Table 3) the most promising results were obtained for different *k* parameter for each class and in *k*NN-based classification this parameter should be constant for all classes.

Moreover the authors will work on optimizing the descriptor list by identifying and discarding those that have the negative influence on correctness and/or expanding it with new parameters. Such new features may also include high level features related to emotions, mood, appearance of musical instruments in a music piece, etc., to provide insight into how the music genre is formed. Also, experiments are to be carried out employing more music genres.

6. Acknowledgements

This work has partially been supported by the European Union from the European Social Fund (grant agreement number: UDA-POKL.04.01.01-00-106/09). and also by the project No. SP/I/1/77065/10 entitled: "Creation of universal, open, repository platform for hosting and communication of networked resources of knowledge for science, education and open society of knowledge", a part of a Strategic Research Program "Interdisciplinary system of interactive scientific and technical information" funded by the National Centre for Research and Development (NCBiR, Poland).

BIBLIOGRAPHY

1. Weninger F., Woelmer M., Schuller B.: Automatic Assessment of Singer Traits in Popular Music: Gender, Age, Height and Race. 12th International Society for Music Information Retrieval Conference (ISMIR), 2011, http://ismir2011.ismir.net/papers/OS1-4.pdf.
2. Bogdanov D., Herrera P.: How Much Metadata Do We Need in Music Recommendation? A Subjective Evaluation Using Preference Sets. ISMIR 2011, http://ismir2011.ismir.net/ papers/PS1-10.pdf.
3. Hu Y., Ogihara M.: Nextone Player: A Music Recommendation System Based on User Behavior. ISMIR 2011, http://ismir2011.ismir.net/papers/PS1-11.pdf.
4. De La Bandera C., Barbancho A., Tardón L., Sammartino S., Barbancho I.: Humming Method for Content-Based Music Information Retrieval. ISMIR 2011, http://ismir2011.ismir.net/papers/PS1-2.pdf.
5. McKay C., Fuginaga I.: Automatic Genre Classification Using Large High-Level Musical Feature Sets. 5th International Society for Music Information Retrieval Conference (ISMIR), 2004, http://www.music.mcgill.ca/~cmckay/papers/musictech/ISMIR_04.pdf.
6. Han J., Kamber M., Pei J.: Data Mining Concepts and Techniques. Third Edition, 2011.

7. Kostek B.: Perception-Based Data Processing in Acoustics. Applications to Music Information Retrieval and Psychophysiology of Hearing. Springer Verlag, Series on Cognitive Technologies, Berlin, Heidelberg, New York 2005.
8. Nowak-Brzezińska A.: Przygotowanie danych w środowisku R. 2011, zsi.tech.us.edu.pl/~nowak/ed/ped_PD.pdf.
9. Minkowski Distance: http://xlinux.nist.gov/dads//HTML/lmdistance.html.
10. Manhattan Distance: http://xlinux.nist.gov/dads//HTML/manhattanDistance.html.
11. Cormen T., Leiserson C., Rivest R.: Introduction to Algorithms. MIT Press, 1990.
12. Kostek B., Kupryjanow A., Żwan P., Jiang W., Raś Z., Wojnarski M., Świetlicka J.: Report of the ISMIS 2011 Contest: Music Information Retrieval, [in:] Kryszkiewicz M., Rybiński H., Skowron A., Raś Z. W. (eds.): Foundations of Intelligent Systems. Springer Verlag, Heidelberg 2011, p. 715÷724.
13. WEKA 3: Data Mining Software in Java, http://www.cs.waikato.ac.nz/ml/weka/.
14. WEKA Library for Java, ver. 3.7.5, http://sourceforge.net/projects/weka/files/weka-3-7-windows-jre-x64/3.7.5/.

Wpłynęło do Redakcji 16 stycznia 2013 r.

Omówienie

W artykule przedstawiono wpływ różnych metod normalizacji zastosowanych do zbioru wektorów cech utworów muzycznych. Utwory muzyczne reprezentowane są za pomocą wektorów parametrów opisujących ich cechy. Zaprezentowano wpływ metod normalizacji min-max oraz Zero-Mean z użyciem różnych funkcji odległości (Euklidesowej, Manhattan, Czebyszewa, Minkowskiego), jako wstępnego procesu przetwarzania w klasyfikacji gatunków muzycznych, na wyniki klasyfikacji. Testy prowadzono z wykorzystaniem klasyfikacji *k*-Najbliższych Sąsiadów (*k*NN). Wyniki przeprowadzonych analiz przedstawia tabela 3. Dane wejściowe reprezentują cztery gatunki muzyczne: pop, latin, rap&hip-hop, jazz. W eksperymentach wykorzystano 200-250 utworów na klasę, przedstawionych w postaci wektorów 173 cech (tabela 1) obliczonych jako wartości średnie dla 20 okien czasowych 30-sekundowych fragmentów utworów.

Addresses

Aldona ROSNER: Silesian University of Technology, Institute of Informatics, Akademicka 16, 44-100 Gliwice, Poland, aldona.rosner@polsl.pl.

Marcin MICHALAK: Silesian University of Technology, Institute of Informatics, Akademicka 16, 44-100 Gliwice, Poland, marcin.michalak@polsl.pl.

Bożena KOSTEK: Gdańsk University of Technology, Audio Acoustics Laboratory, Narutowicza 11/12, 80-233 Gdańsk, Poland, bokostek@audioacoustics.org.