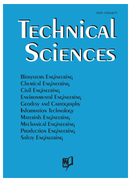
Technical Sciences, 2018, 21(4), 291–290

UNIWERSYTET WARMIŃSKO-MAZURSKI W OLSZTYNIE

Quarterly peer-reviewed scientific journal
ISSN 1505-4675
e-ISSN 2083-4527
TECHNICAL SCIENCES
Homepage: www.uwm.edu.pl/techsci/


Technical Sciences
Biosystems Engineering
Chemical Engineering
Civil Engineering
Environmental Engineering
Geodesy and Cartography
Information Technology
Materials Engineering
Mechanical Engineering
Production Engineering
Safety Engineering

WEATHER AND A PART OF DAY RECOGNITION IN THE PHOTOS USING A KNN METHODOLOGY

Tomasz Krzywicki

Faculty of Mathemathics and Computer Science
University of Warmia and Mazury
Polish Information Processing Society

Received 18 May 2018; accepted 16 November 2018; available online 21 November 2018.

Keywords: weather recognition, image analysis, machine learning, classification.

Abstract

This article presents a proposal for recognizing the weather and part of a day in digital photos encoded in the bitmap format, based on auctorial edge detection algorithm of horizon to demarcate the sky and k-nearest neighbours algorithm, to classify the daytime in the picture as “day” or “night” and to classify the weather as “sunny” or “cloudy”. To verify the effectiveness of the classification the Internal Bagging-5 model was applied. The data for surveys in the form of pictures was prepared on self-provision. To test the method in a different location, data from the Internet was used.

Introduction

Recognizing the weather based on the bitmap is difficult. There is a lack of basic sensory information which reflecting the actual weather as: temperature [°C], pressure [hPa], wind speed [m/s], downfall [mm/h], dampness [%], etc. The work’s aim is to propose the methods to analyze the picture and determine part of a day as “day” or “night”, also determining the weather as “sunny” or “cloudy”. The proposed solution was based on machine learning method – k-nearest neighbours (kNN) to learn the computer’s system weather recognizing in virtue of the examples in the form of 24-bitmap photos.

Correspondence: Piotr Artiemjew, Katedra Metod Matematycznych Informatyki, Uniwersytet Warmińsko-Mazurski, ul. Słoneczna 54, 10-710 Olsztyn, phone: 89 524 60 82, e-mail: tomasz.krzywicki@student.uwm.edu.pl

The inspiration to formulating the solution was trying to find the algorithms which automatically tag images, including significant surface of the sky. The method performance was determined with fivefold Internal Bagging-5 test. The results were presented as an average value of all tests. The solution has achieved a satisfactory effectiveness.

The picture exploration analysis

The photos which were used during the experiments were taken in Bartoszyce (Poland) by Microsoft LifeCam HD-3000 camera, within a period from 29th August 2017 to 7th September 2017, with a common frame. Photos confirming the effectiveness of the method used were taken in Le Morne (Mauritius), within a period 7th October 2018 to 6th November 2018, with a common frame.

The daytime characteristics

The part of a day is a time between sunrise and sunset. In the picture, in the RGB colour space, the daytime will be determining in the sum number ratio of red and green colour to blue colour, also the average intensity of pixel level. The pixel intensity should be understood as a brightness in shades of grey, in the field of <0;255>, where the value 0 is the lowest pixel intensity (black colour) and the value 255 is the highest (white colour). The part of a day classification will be automatically after learning the model method k-nearest neighbours.

The sum number ratio parameter of colours was formulated as

$$\frac{R+G}{B},$$

where:

R – the red colour sum number in the picture,
G – the green colour sum number in the picture,
B – the blue sum number in the picture.

The average pixel intensity parameter was formulated as

$$\frac{R+G+B}{3},$$

where:

R – the colour sum number in the picture,
G – the green colour sum number in the picture,
B – the blue sum number in the picture.

The graphic presentation of the layout pixel colours for the photos which were taken during the day and the night, were presented with the histograms (Figs. 1, 2).

The picture's histogram shows that during the day, it can be noticed, for example, for the value 100, the red colour occurred about 1,000 times. In the picture which was taken during the night, number of red colour pixels with intensity 100 was about 3,000. Analysing the colour distribution in RGB photos taken during the day and the night surface, the differences are noticeable. In the picture taken at night towards the picture taken during the day the intensity of blue colour is increasing, the intensity of red and green colour is falling.

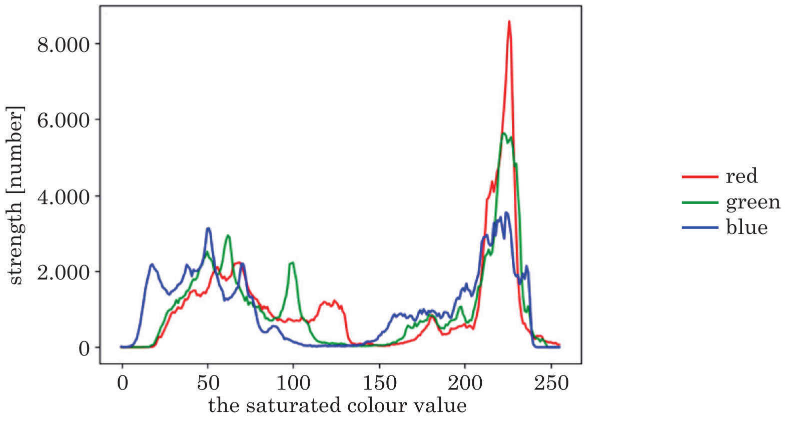


Fig. 1. The distribution of RGB values in the picture taken during the day

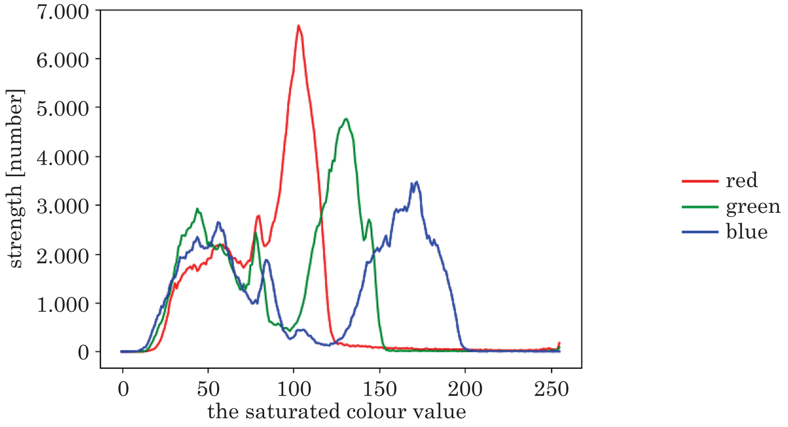


Fig. 2. The distribution of RGB values in the picture taken during the night

The picture histograms' observation taken every minute around the dusk, permitted to see that the sum number ratio parameter of colours and the average pixel intensity is decreasing around the dark (Fig. 3).

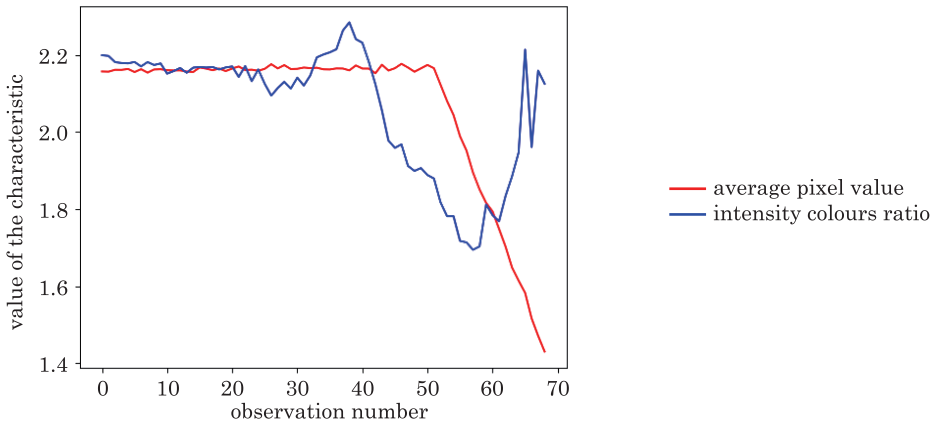


Fig. 3. Parameters' change in picture value along with night coming on

For the sake of clarity presented data, the intensity colour parameter ratio (coloured blue) was rescaled with a common logarithm. Each of the following observation number means a taken photo a minute later. It can be noticeable that from the moment when the night is coming on, the plurality of pixels is dark colour, whereby the parameter which is the ratio of colours number sum, is increasing again, as can be seen around the 60 minute of observation.

Weather characteristics

Intuitively, the sunny weather is seen as an under partly cloudy blue sky. The cloudy weather means no blue colour on the sky.

Empirically, it was noticed, that red colour doesn't influence on classifying the sunny or cloudy weather in the bitmap picture of the sky. Let us adopt that the sum number ratio of blue to green colour above the horizon's edge (the algorithm of detecting the horizon's edge was described in the next subsection) is a parameter which characterises the weather classification, it was formulated by B/G, where B is the sum number of blue colour, G is the sum number of green colour.

The pixel number ratio of higher intensity from the threshold level set of lower intensity from the threshold level, it will be called as a picture dichotomy. By taking threshold value 128, which is the middle range of set values by the pixel in the picture, in the scale of grey, it was noticed that the picture dichotomy taken during the sunny weather is higher than the dichotomy picture taken during the cloudy weather. Table 1 includes the parameters' values of pictures taken during the sunny and cloudy weather.

Table 1

Parameter values of pictures taken during the sunny and cloudy weather

Taken photo time	Weather	B/G	Dichotomy
28.08.2017 6:34 pm	sunny	1.281	1.643
29.08.2017 6:43 pm	sunny	1.124	1.608
29.08.2017 6:40 pm	sunny	1.154	1.499
29.08.2017 7:04 pm	sunny	1.011	1.206
01.09.2017 12:29 am	cloudy	0.960	1.109
01.09.2017 1:01 pm	cloudy	0.962	1.103
06.09.2017 12:11 am	cloudy	0.966	0.966
06.09.2017 6:57 pm	cloudy	0.970	0.810

The parameter values, which characterise the weather in the picture taken during the sunny weather are higher than the parameter values, which characterise the weather in the picture during the cloudy weather.

The edge of the horizon demodulation

With the aim of determination the edge of the horizon, it is necessary to determine outlines all of the subsets presented in the picture. Sobel (KAEHLER, BRADSKI 2017) operator shall produce the outlines in the picture in shades of grey, which as used during the determining the edge of the horizon. It was noticed that the pixels of outside edge contour, most commonly have their values at least threefold higher than average pixels values outline. For our purposes, the horizon will be called as lower edge of the sky outline (subset).

Determining edge of the horizon in the picture rely on finding the threshold value which is a lower limit (infimum). In order to analyse the weather, the land was divided from the sky. To that end, the function called get_threshold was constructed, its aim is to count three times the average pixel intensity of outline. Due to that function, the lower limit of the sky was determined. By means of get_threshold (Fig. 4) function, the edge of the horizon was determined, by creating get_horizon_edge (Fig. 5) function.

```
function get_threshold(image)
    sum_values = 0
    pixel_count = 0
    for i = 0 to image.height-1 do
        for j=0 to image.width-1 do
            if image[i][j] > 0 then
                sum_values += image[i][j]
                pixel_count += 1
            endif
        endfor
    endfor
    return round((sum_values/pixel_count)*3)
endfunction
```

Fig. 4. Function calculating three times the average pixel intensity

```
function get_horizon_edge(image, threshold)
    horizon_edge = []
    for i = 1 to image.width-2 do
        j = 1
        while j < image.height-1 and obraz[j][i] <= threshold do
            j += 1
        endwhile
        horizon_edge.append(new Point(i, j))
    endfor
endfunction
```

Fig. 5. Function dividing the land from the sky

The function called get_horizon_edge (Fig. 5) determines from the designated picture pixels list, which establish the horizon. In Figures 6 and 7 are original photos before and after the operation of the get_horizon_edge function result. In Figure 6 is noticeable an original photo, in Figure 7 is symbolically pixels list of the horizon.

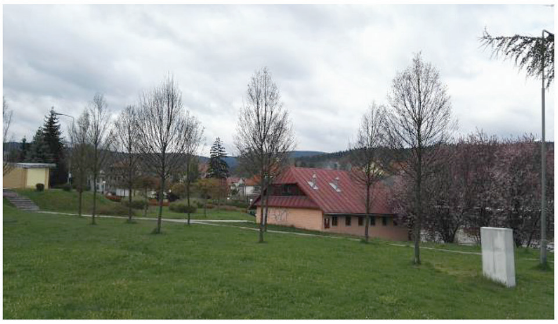


Fig. 6. An exemplary photo showing the horizont covered by trees

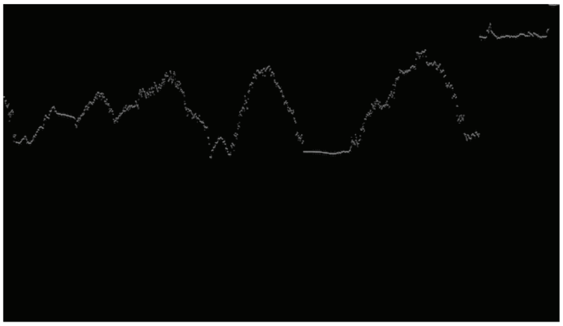


Fig. 7. Symbolically pixels list affixed to the photo horizont number 1

The detection algorithm of the horizon edge has disadvantages, which can negatively influence on the parameter values of current weather in the picture. When analysing photos both light-struck and low resolution, also overly contrasty, it was noticed, that the edge of the horizon is not always being determined well. The list of horizon dots determined with get_horizon_edge sometimes has deviated from some pixels under or above a real horizon line.

kNN algorithm

The kNN algorithm (k-nearest neighbours) is a machine model learning which facilitates multiclass classification. The kNN model belongs to lazy classifier- learning from the examples. The kNN algorithm working principle is performed by checking only k objects in the training set, which features in the expanse are close to classified object as possible. Basing on just founded k-similar objects the decision-making class is chosen, which total weight is the best. The kNN algorithm needs two parameters: k-nearest neighbours number and a metric to assess proximity of objects.

Selected metric remoteness

The most common measure of distance is Euclidean metric, used in scores of kNN algorithm implementation as default. Euclides distance of two objects is formulated as:

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (a_i(x) - a_i(y))^2},$$

where:

$d(x, y)$ – Euclidean distance in n-dimensional space of real numbers,
$a_i(x)$ – the value of i coordinate – that x object's point in n-dimensional space of real numbers,
$a_i(y)$ – the value of i coordinate – that y object's point in n-dimensional space of real numbers.

Manhattan metric is a metric used in the cities, where a grid system of streets run north-south and east-west. The distance determined as Manhattan metric is the sum of the absolute differences coordinate points' values in space, which is formulated as:

$$d(x, y) = \sum_{i=1}^{n} (a_i(x) - a_i(y)),$$

where:

$d(x, y)$ – Manhattan distance in n-dimensional space of real numbers,
$a_i(x)$ – the value of i coordinate – that x object's point in n-dimensional space of real numbers,
$a_i(y)$ – the value of i coordinate – that y object's point in n-dimensional space of real numbers.

Chebyshev metric is a distance metric used in chess game, in determining the minimum number of moves, to go from one square to another one. Chebyshev distance is formulated as:

$$d(x, y) = \max.(|a_i(x) - a_i(y)|), \text{ for } i = 1, 2, \ldots, n,$$

where:

$d(x, y)$ – Chebyshev distance in n-dimensional space of real numbers,
$a_i(x)$ – the value of i coordinate – that x object's point in n-dimensional space of real numbers,
$a_i(y)$ – the value of i coordinate – that y object's point in n-dimensional space of real numbers.

General kNN algorithm procedure:

– loading test system (X, A, c) and training system (Y, A, c), where: X is objects testing set, Y is objects training set, A is conditional objects' features, c is a class objects set;

– the d metric's choice of counting the distance between objects and k-nearest neighbours number;

– the classification of all tested objects by using k-nearest objects, for each of the training objects' class set – the decision is a class, which objects are the closest to tested object, based on before stated d metric.

Testing classifier efficiency

As the result method of classifier efficiency there was chosen Internal Bagging-5 method, which is based on frequentative Bagging method's carrying out, in training set to matching the most effective parameter of k-nearest neighbours number, and then checking the classification effectiveness in testing set. The each test results are averaged out, which gives classifier's estimate effectiveness better and closer to reality.

Bagging

The foundation of Bagging method is repeated performing Bootstrap test. The each test results are levered (ARTIEMJEW 2016).

In Bootstrap method from n-objects set of original decision – making system, selecting at random is restored of n-objects, which are the training set. It is easy to spot that not all of the objects will be chosen, the other ones will appear over and over again. Following that procedure there is a chance that in the training set the same objects will be many a time. In case of appearing an empty test set, we repeat its draw. The Bootstrap method gives inaccurate, perfunctory evaluation of classifiers effectiveness, contignent on random, a single split on training and testing set (ARTIEMJEW 2016).

Results discussion

During the survey there were taken 2,611 photographs. There were randomly chosen of 1,000 photos showing a day and a night, sunny and cloudy weather, with shared of 500 photos for each *clas*s. In Figures 8 and 9 is performed a parameters visualization for 1,000 objects of a daytime and 1,000 weather's objects.

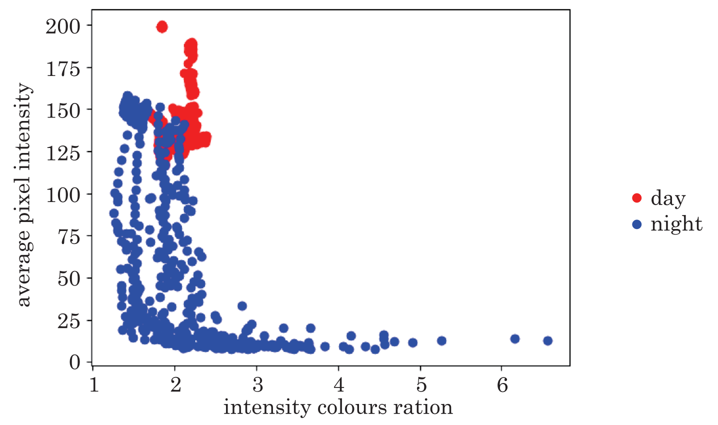


Fig. 8. Visualisation of a daytime parameters

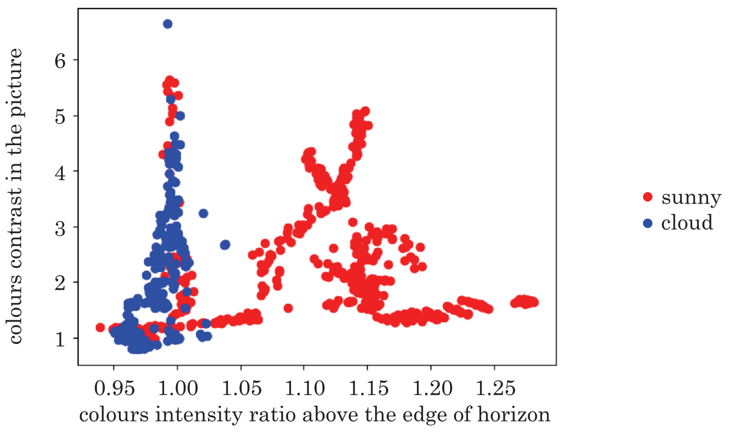


Fig. 9. Visualisation of weather parameters

The parameter values for day and cloudy weather are more focused rather than for night and sunny weather (Fig. 8, 9).

The kNN validation model with Euclidean distance metric for recognising a part of a day and weather in the photos, was done by using fivefold Internal Bagging-5 method test, on 1,000 objects containing daytime parameters and on 1,000 objects containing weather parameters, with fives *k*-parameter (Tabs. 2, 3). After the test, results are presented in Table 4.

Table 2

Parameter's fives used during the daytime classifier test by Internal Bagging-5 method

Part of a day					
Test number	$k1$	$k2$	$k3$	$k4$	$k5$
Test 1	$k1$=2	$k2$=2	$k3$=4	$k4$=2	$k5$=2
Test 2	$k1$=3	$k2$=2	$k3$=2	$k4$=2	$k5$=2
Test 3	$k1$=2	$k2$=2	$k3$=2	$k4$=2	$k5$=2
Test 4	$k1$=4	$k2$=2	$k3$=3	$k4$=2	$k5$=2
Test 5	$k1$=2	$k2$=4	$k3$=2	$k4$=2	$k5$=2

Table 3

k parameter's five used during the Feather classifier test by Internal Bagging-5 method

Weather					
Test number	$k1$	$k2$	$k3$	$k4$	$k5$
Test 1	$k1$=3	$k2$=2	$k3$=2	$k4$=2	$k5$=2
Test 2	$k1$=3	$k2$=2	$k3$=2	$k4$=2	$k5$=2
Test 3	$k1$=2	$k2$=2	$k3$=2	$k4$=2	$k5$=2
Test 4	$k1$=3	$k2$=2	$k3$=2	$k4$=2	$k5$=4
Test 5	$k1$=2	$k2$=2	$k3$=2	$k4$=2	$k5$=2

Table 4

Test results of daytime and weather classifier by Internal Bagging method

Type of problem	Arithmetic average	Standard deviation
Part of a day	0.943548097603988	0.010142184895868
Weather	0.958484382006594	0.008661892051049

709 photos from different location were used to confirm the effectiveness of the method. There were randomly chosen of 300 photos showing a day and a night, sunny and cloudy weather, with shared of 75 photos for each class.

For the data checking the effectiveness of the method, the test Internal Bagging-5 was performed again. The results are presented in Table 5.

Table 5

Test results of daytime and weather classifier by Internal Bagging method
in the different location

Type of problem	Arithmetic average	Standard deviation	The most common k
Part of a day	0.991473908231302	0.005949486731013	2
Weather	0.911436556374957	0.030718308098330	3

Conclusions

A key aim of the article was to performance a proposal of recognizing a part of a day and the weather, based on the learning machine algorithm k-nearest neighbours, in virtue of digital photos containing significant sky surface. Also, there was shown the new approach to the edge of the horizon detection in the picture, noted for Adobe Photoshop.

The suggested solution has achieved a satisfactory effectiveness, confirmed by fivefold Internal Bagging-5 test: about 94% to daytime classification and about 96% to weather classification. The suggested solution in the different location has also achieved a satisfactory effectiveness, confirmed by fivefold Internal Bagging-5 test: about 99% to daytime classification and about 91% to weather classification.

Future works

In the future it is planned to check effectiveness in other locations and at different times of the year. For this purpose photos from locations: North and South Europe, Central Africa, Eastern Asia are collected.

In the future is also planned a comprasion of the proposed method with other solutions performing similar tasks.

References

ARTIEMJEW P. 2016. *Wybrane paradygmaty sztucznej inteligencji*. PJATK Publishing House, *Warszawa*.

ESPOSITO R., SAITTA L. *2003. Monte Carlo Theory as an Explanation of Bagging and Boosting*. Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence IJCAI-03 Contents al and Data Integration, p. 499–504, http://www.ijcai.org/Proceedings/03/Papers/074.pdf (access: 27.01.2018).

KAEHLER A., BRADSKI G. 2017. *OpenCV 3. Komputerowe rozpoznawanie obrazu w C++ przy użyciu biblioteki OpenCV*. Helion Publishing House, Gliwice.

LEMMENS A., CROUX C. 2006. *Bagging and Boosting Classification Trees to Predict Churn*. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.196.6659&rep=rep1&type=pdf (access: 9.02.2018).

MACHOVÁ K., BARČÁK F., BEDNÁR P. 2006. *A Bagging Method using Decision Trees in the Role of Base Classifiers*. http://people.tuke.sk/kristina.machova/pdf/bagging_o6.pdf (access: 12.04.2018).

MOLINARO A.M., SIMON R., PFEIFFER R.M. 2005. *Prediction error estimation. a comparison of resampling methods*. Bioinformatics, 21(15): 3301–3307.