Vol. 28, No. 1/4, 2019

Machine GRAPHICS & VISION

International Journal

Published by
The Institute of Information Technology
Warsaw University of Life Sciences – SGGW
Nowoursynowska 159, 02-776 Warsaw, Poland
in cooperation with
The Association for Image Processing, Poland – TPO

Data Augmentation Techniques for Transfer Learning Improvement in Drill Wear Classification Using Convolutional Neural Network

Jarosław Kurek[1], Izabella Antoniuk[1], Jarosław Górski[2], Albina Jegorowa[2],
Bartosz Świderski[1], Michał Kruk[1], Grzegorz Wieczorek[1], Jakub Pach[1],
Arkadiusz Orłowski[1], Joanna Aleksiejuk-Gawron[3]
[1]*Institute of Information Technology*
[2]*Institute of Wood Sciences and Furniture*
[3]*Institute of Mechanical Engineering*
Warsaw University of Life Sciences – SGGW, Warsaw, Poland
jaroslaw_kurek@sggw.pl

Abstract. This paper presents an improved method for recognizing the drill state on the basis of hole images drilled in a laminated chipboard, using convolutional neural network (CNN) and data augmentation techniques. Three classes were used to describe the drill state: red – for drill that is worn out and should be replaced, yellow – for state in which the system should send a warning to the operator, indicating that this element should be checked manually, and green – denoting the drill that is still in good condition, which allows for further use in the production process. The presented method combines the advantages of transfer learning and data augmentation methods to improve the accuracy of the received evaluations. In contrast to the classical deep learning methods, transfer learning requires much smaller training data sets to achieve acceptable results. At the same time, data augmentation customized for drill wear recognition makes it possible to expand the original dataset and to improve the overall accuracy. The experiments performed have confirmed the suitability of the presented approach to accurate class recognition in the given problem, even while using a small original dataset.

Key words: convolutional neural networks, data augmentation, deep learning, tool condition monitoring.

1. Introduction

During a furniture manufacturing process the drill sharpness is one of many important aspects ensuring the quality of the final product. When the drill is not sharp enough it may have a negative impact on different furniture elements, which in turn may result in financial losses for the company. Therefore, determining the moment when an operator should replace the current drill with a new one is an important and genuine problem. Merely observing the drill state with an unarmed eye is not efficient enough, hence the research to automatize this stage and the use of a computer system for drill wear recognition is conducted.

When approaching this problem usually three classes of drills are considered: red, yellow, and green, to automatically asses the current state of the drill. The red class indicates that the drill should be replaced straight away, since its further use in the production process would probably result in too many damaged elements. The yellow class points to the drill which is suspected of being worn out. Such a drill should be evaluated and either disposed of, or used in further production, depending on the expert opinion. Final, the green state signifies the drill which is still in good condition and can be used in the production process without any changes.

Automatization process for this problem (often called tool condition monitoring or TCM) is not a new concept. Previous solutions, related to drill assessment, usually required a large collection of various sensors used for measuring different signals. Those signals were then used to define the diagnostic features for each drill state. Some of the most common signals are those related to noise, vibrations and acoustic emission, the cutting torque or feed force [4]. Although such setups can generate accurate results, they usually require numerous preprocessing stages before any considerable precision can be reached. Some of the prior steps may include an appropriate sensor and signal choice, the generation and selection of the best diagnostic features, or building the classification model. Such solutions are usually quite complex, time consuming and often expensive in preparation, not to mention that even simple mistakes at any given step may in result bring unacceptable and unusable products. In previous works [1,2,3] the authors took into account diverse features that were generated on the basis of different registered signals, yet the accuracy of presented solutions still did not exceed 90% for the recognition of the three classes defined.

One of the goals of this and the previous work (see [5]) is to decrease the complexity of the entire solution. Therefore, similarly as in the former paper, we relied on the assessment of images of drilled holes. In such an approach the only required external equipment is a camera used for taking pictures of drilled holes, which then are used to asses the state of the drill. Recently, the Convolutional Neural Network (CNN) is considered as one of the most effective solutions [6,9] which additionally does not require any specialized diagnostic features to be defined. It has also been proven effective in previous tests in a similar setup [5], therefore we decided to base the current solution on similar concepts.

In the previous solution [5] a CNN was applied to this problem on a limited set of training data (only 242 images, representing three classes: 102 samples for green, 60 samples for yellow and 80 samples for red class). Using such limited collection, the prepared algorithm was able to achieve 85% accuracy for a pretrained CNN algorithm, using AlexNet model created by Krizhevsky, Sutskever and Hinton [11,12,13]. Accuracy was improved to 93.4% by applying the Support Vector Machine (SVM) as an alternative

(a) Green (b) Yellow (c) Red

Fig. 1: Examples of holes produced by each drill wear class: Green, Yellow and Red.

to the final CNN layer. In another paper [6] the data were expanded by using simple image operations. CNN with classical deep learning approach was used to recognize two classes of drill wear (sharp enough or not sharp enough). In that case, 900 images were used (300 images for the first class and 600 images for second class), and the original system achieved 66.6% accuracy. The final precision was improved by increasing the number of original images through simple operations (rotation, scaling, adding noise), and resulted in 89% accuracy for 11700 images and 95.5% accuracy for 33300 images. Unfortunately, the learning process was lengthy in that case and lasted over 20 hours.

This paper proposes an approach combining the best elements of both solutions: usage of pretrained CNN applying the transfer learning methodology [11, 12, 14] in order to decrease the training time and expanding original the data set with artificially augmented data, to achieve better accuracy of final solution [6]. We will use three classes for drill wear determination: red (for worn out), yellow (as a warning for an operator) and green (to indicate good condition of the drill). The aim of the current work is to show that the application of transfer learning, even with a very small data set, expanded using data augmentation methods, can achieve high classification precision, without the need for costly equipment or time consuming CNN training (as in the classic approach). The initial data set, similarly as in [5], contains 242 samples of images which represent the three classes. The same dataset was used in our other paper in this volume [15], where we further extend our approach by applying an ensemble of classifiers.

2. Dataset preparation

The data samples containing images of drilled holes for all three classes were collected in cooperation with Faculty of Wood Technology, at Warsaw University of Life Sciences using standard Buselatto JET 100 CNC vertical machining centre. For the test purposes drilling process was performed on standard laminated chipboard (Kronopol U 511 SM), which is typically used in furniture industry. Dimension of the test piece were $150 \times 35 \times 18$ mm. For the drilling process a regular 12 mm FABA drill equipped with a tungsten carbide tip was used.

Used data set has the same structure as in [5]. Out of 242 samples, three subsets of original images were prepared: 102 samples for green class, 60 samples for yellow class and 80 samples for red class. Typical images for each of the specified classes are presented in Fig. 1a, 1b, 1c (images in this Figure and partly in Fig. 2 are similar to those used in our previous papers on drill wear classification due to that we used the same initial set of images).

To expand the initial data set, for each of the original images additional collection of augmented samples was produced. Total of 18 different operations were defined. After performing each of those procedures, the final data set consisted of 4598 images, including 1938 samples for green class, 1140 samples for yellow class and 1520 samples for red class. Operations used for data augmentation were as follows:

1. ColorToGrayscale – convert image to gray-scale values.
2. ColorBrightJitter – adjust brightness of image by random offset in range [-0.3, -0.1].
3. ColorContrast1 – adjust contrast of image by scale factor in range [1.2, 1.4].
4. ColorContrast2 – adjust contrast of image by scale factor in range [1.4, 1.6].
5. ColorHueJitter1 – change image hue by random offset in range [0.05, 0.15].
6. ColorHueJitter2 – change image hue by random offset in range [0.15, 0.30].
7. NoiseGauss – add Gaussian noise to the image.
8. Noise1 – add salt and pepper noise to the image, with strength factor 0.2
9. Noise2 – add salt and pepper noise to the image, with strength factor 0.4
10. Reflection – create a reflection that transforms original image by flipping it in each dimension with 50% probability.
11. Rotate30 – rotate image by random angle in range [0, 30].
12. Rotate45 – rotate image by random angle in range [30, 60].
13. Rotate90 – rotate image by random angle in range [60, 90].
14. Rotate120 – rotate image by random angle in range [90, 120].
15. Scale1 – scale image by random factor in range [1.2, 1.5]
16. Scale2 – scale image by random factor in range [0.8, 0.9]
17. Shear – shear transformation (horizontal), with angle selected randomly in [-30, 30].
18. Translate50 – translate image both vertically and horizontally by random number of pixels in range [-50,50].

Example results for each transformation are presented in Fig. 2. For a detailed description of the used transformations please refer to [20].

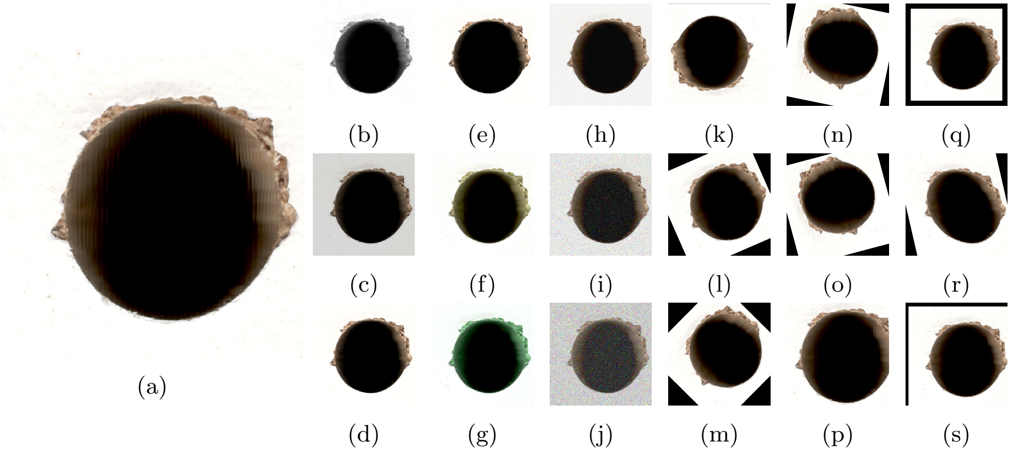

Fig. 2: Examples of augmented data set for a single original image (**a**): (**b**) ColorToGrayscale, (**c**) ColorBrightJitter, (**d**) ColorContrast1, (**e**) ColorContrast2, (**f**) ColorHueJitter1, (**g**) ColorHueJitter2, (**h**) NoiseGauss, (**i**) Noise1, (**j**) Noise2, (**k**) Reflection, (**l**) Rotate30, (**m**) Rotate45, (**n**) Rotate90, (**o**) Rotate120, (**p**) Scale1, (**q**) Scale2, (**r**) Shear, (**s**) Translate50

3. Deep learning approach using CNN

Deep learning approach is one of very popular solutions to classification problems, especially ones meant for images [7, 8, 9, 10]. Usually in such cases CNN would be used as a main function. Due to network structure, and manner in which elements of different type and size are processed, there is no need for the user to specify diagnostic features used to describe particular problem. Deep learning is capable of finding those features without any manual intervention of the user. What is even more important, features extracted in that way are more universal and can work on other types of images with reasonable accuracy.

Contrary to regular methods, deep learning analyses blocks of pixels with fixed sizes (i.e. 5×5, 10×10 or 15×15 for large images). Output signals are filtered, first with linear then nonlinear transformations, and finally use pooling operation to reduce the size of processed blocks. The most significant difference between classical machine learning and deep learning approaches lies in a way in which diagnostic features are extracted. In case of classical machine learning this process constitutes of separate stage of processing (usually consisting of various, often complicated steps). In case of deep learning, generating features is done automatically in hidden layers of neural network. Later on extracted features can be used as input data for softmax classifier or serve as input attributes for external classifiers. For more detailed description of used CNN solution refer to [5].

4. Pretrained CNN AlexNet

Since the database used in current approach is inadequate to start proper training of CNN from the scratch, the only possibility to achieve acceptable results requires using model that was earlier pretrained on large set of different images. Following the methodology of transfer learning [12, 13] we will use AlexNet model [11, 12, 13], that is implemented in Matlab [14]. It was pretrained using over a million images, with 1000 different classes [14, 16].

The AlexNet structure used in presented classification task is built of 9 layers composed of 25 sub-layers, and uses softmax classifier. This function takes u-vector and calculates i-th component of output vector, representing given number of classes: m according to equation:

$$\mathrm{softmax}(u)_i = \frac{\exp(u_i)}{\sum_{j=1}^{m} \exp(u_j)} \tag{1}$$

This function is normalized exponentially while its values are in the range [0, 1] and are treated as probabilities for each class to occur. Component with highest value indicates recognized class. For our application of three classes cross entropy has been used in calculations of softmax function.

Since AlexNet model requires for all images to have the same size ($227 \times 227 \times 3$), we additionally transform our input data to fit those requirements. In our solution first six trained layers were applied without any changes, taking mechanism for creating diagnostic features directly from classes of images used in pretraining stage of AlexNet. Last three, fully connected layers were adjusted during learning process to best fit our image classification problem.

5. Pretrained CNN AlexNet with augmented dataset

Since in previous approach pretrained CNN model achieved promising results, but evidently required more data than the original amount to achieve satisfactory results, in this approach we decided to artificially expand data set. We decided to use augmented data instead of collecting more samples, to minimize the amount of work user needs to perform, to achieve final results. Since taking and preparing images for further use in chosen model can take considerable amount of time it was decided, to instead use basic operations to generate new instances from the ones already collected. We performed 18 operations in total on original data set, choosing random values for most of them to achieve maximal diversity in generated set.

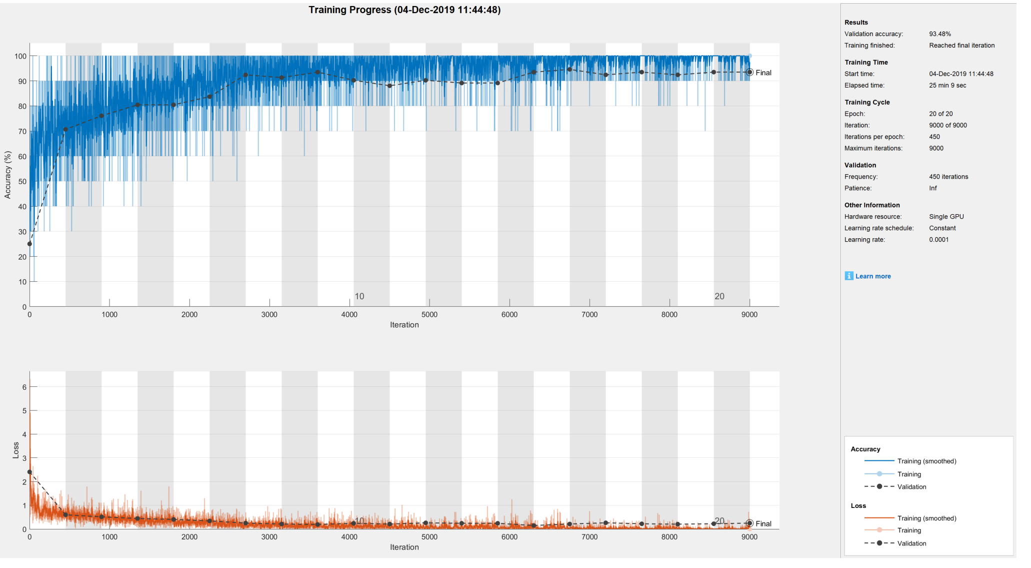

Fig. 3: Outline of training process for CNN model with the augmented data set.

While considering results of previous experiments and CNN characteristics in general, we assumed, that this approach would increase overall performance of our algorithm, without the need to use more complex solutions (like pretrained CNN model, with SVM as a classifier) or significant increase in computation time (like it was the case with classical CNN learning from scratch). At the same time, since additional images are generated, they remain close enough to original ones, to meet the required criteria, while improving overall data set diversity. For an overview of the learning process for this approach see Fig. 3.

6. Results

Original data set used in experiments contained 242 images representing three different classes. After performing data augmentation, extended set for final algorithm contained 4598 images, changed using basic operations such as rotation, translation, adding noise or others (see Fig. 2).

For first two algorithms set of 242 images was split randomly into 10 subsets, where 9 were used during training process, while the final one was used in testing mode. Performed experiments have been repeated 10 times in 10-fold cross validation mode, exchanging test part of the data for each iteration. For third algorithm similar approach

Tab. 1: Accuracy of chosen algorithms applied to the problem of drill condition classification with 3 classes (green, yellow and red). First two algorithms use 242 input images, while last algorithm used artificially expanded set of 4598 images.

Deep learning algorithm	**Accuracy** [%]
Standard CNN	35
Pretrained CNN	85
Pretrained CNN with augmented data	93

was used, but in that case 95% of the data in each iteration was used for training, while remaining 5% acted as testing set. Obtained statistical results in the form of the mean value of class recognition accuracy are presented in Table. 1. Subsequent results refer to standard CNN, which learned from scratch ("Standard CNN"), CNN using pretrained AlexNet model with softmax as a classifier ("Pretrained CNN") and the same pretrained AlexNet model, used on the augmented data set ("Pretrained CNN with augmented data"). Outline of training process for CNN algorithm with an augmented data set is shown in Fig. 3.

As in the previous paper [5], the advantage of using the pretrained CNN is clearly visible. Furthermore, by artificially expanding initial data set, presented approach was able to achieve acceptable accuracy of 93.48%. Such precision is even slightly higher than achieved by solution combining CNN with SVM (Support Vector Machine) used as the final classifier, which was best in previous work (93.4% accuracy). Since former approach is more complex than currently used one, final outcome of presented algorithm is more than acceptable.

7. Conclusion

In this paper we presented an improved method, that applied AlexNet CNN network to problem of recognising drill wear state, based on images of drilled holes. In our approach we used data augmentation, to artificially increase initial data set, and by using pretrained model, we achieved high class recognition accuracy. In previous works such precision was achieved either by using more complex solution or greatly increasing initial data set (in case of learning from scratch, without pretrained network), which in turn resulted in prolonged computations.

Presented results of numerical experiments confirm, that used approach was able to achieve acceptable level of accuracy when predicting states of the drill. The results were compared both to traditionally learned CNN and previous version of pretrained AlexNet CNN without the data augmentation. Final algorithm was able to achieve slightly higher accuracy than more complex solution, which used SVM as a classifier for AlexNet CNN network, while learning time did not exceed 30 minutes. Similarly as in previous approach, pretrained network required only minimal interference in the last layers of CNN. This process was relatively quick, while the resulting model had good generalization properties for the class recognition.

References

[1] K. Jemielniak, T. Urbański, J. Kossakowska, S. Bombiński. Tool condition monitoring based on numerous signal features. Int. J. Adv. Manuf. Technol., vol. 59, pp. 73-81, 2012

[2] S. S. Panda, A. K. Singh, D. Chakraborty, S. K. Pal. Drill wear monitoring using back propagation neural network. Journal of Materials Processing Technology, vol. 172, pp. 283-290, 2006.

[3] R. J. Kuo. Multi-sensor integration for on-line tool wear estimation through artificial neural networks and fuzzy neural network. Engineering Applications of Artificial Intelligence, vol. 13, pp. 249-261, 2000.

[4] J. Kurek, M. Kruk, S. Osowski, P. Hoser, G. Wieczorek, A. Jegorowa, J. Górski, J. Wilkowski, K. Śmietańska, J. Kossakowska. Developing automatic recognition system of drill wear in standard laminated chipboard drilling process. Bulletin of the Polish Academy of Sciences. Technical Sciences, vol. 64, pp. 633-640, 2016.

[5] J. Kurek, G. Wieczorek, M. Kruk, A. Jegorowa, S. Osowski. Transfer learning in recognition of drill wear using convolutional neural network. 18th International Conference on Computational Problems of Electrical Engineering (CPEE) (pp. 1-4). IEEE. September 2017.

[6] J. Kurek, B. Swiderski, A. Jegorowa, M. Kruk, S. Osowski. Deep learning in assessment of drill condition on the basis of images of drilled holes. Proc. SPIE 10225 Eighth International Conference on Graphic and Image Processing (ICGIP 2016), pp. 102251V, February 8, 2017.

[7] L. Deng, D. Yu. Deep Learning: Methods and Applications. Foundations and Trends in Signal Processing, vol. 7, pp. 3-4, 2014.

[8] Y. Bengio. Learning Deep Architectures for AI. Foundations and Trends in Machine Learning, vol. 2, no. 1, pp. 1-127, 2009.

[9] I. Goodfellow, Y. Bengio, A. Courville. Deep learning, MIT Press, 2016.

[10] J. Schmidhuber. Deep Learning in Neural Networks: An Overview. Neural Networks, vol. 61, pp. 85-117, 2015.

[11] A. Krizhevsky, I. Sutskever, G. Hinton. Image net classification with deep convolutional neural networks. Advances in Neural Information Processing Systems, vol. 25, pp. 1-9, 2012.

[12] O. Russakovsky, J. Deng, H. Su et al. ImageNet Large Scale Visual Recognition Challenge International Journal of Computer Vision (IJCV), vol. 115, no. 3, pp. 211-252, 2015.

[13] BVLC AlexNet Model. Online: `https://github.com/BVLC/caffe/tree/master/models/bvlc_alexnet`.

[14] *Matlab 2017a – User manual*, Natick, MA, USA: The MathWorks, Inc, 2017.

[15] J. Kurek, I. Antoniuk, J. Górski, A. Jegorowa, B. Świderski, M. Kruk, G. Wieczorek, J. Pach, A. Orłowski, and J. Aleksiejuk-Gawron. Classifiers ensemble of transfer learning for improved drill wear classification using convolutional neural network. *Machine Graphics & Vision*, 28(1/4):13–13, 2019. doi:10.22630/MGV.2019.28.1.2.

[16] ImageNet. Online: `http://www.image-net.org`.

[17] B. Scholkopf, A. Smola. *Learning with Kernels*, Cambridge: MIT Press, 2002.

[18] M. Kruk, B. Świderski, S. Osowski, J. Kurek, M. Słowińska, I. Walecka. Melanoma recognition using extended set of descriptors and classifiers. Eurasip Journal on Image and Video Processing, vol. 43, pp. 1-10, 2015.

[19] V. N. Vapnik. *Statistical Learning Theory*, New York: Wiley, 1998.

[20] Description of Matlab image transformations. Online: `https://www.mathworks.com/help/deeplearning/examples/image-augmentation-using-image-processing-toolbox.html`.

Classifiers Ensemble of Transfer Learning for Improved Drill Wear Classification Using Convolutional Neural Network

Jarosław Kurek[1], Izabella Antoniuk[1], Jarosław Górski[2], Albina Jegorowa[2], Bartosz Świderski[1], Michał Kruk[1], Grzegorz Wieczorek[1], Jakub Pach[1], Arkadiusz Orłowski[1], Joanna Aleksiejuk-Gawron[3]

[1]*Institute of Information Technology*

[2]*Institute of Wood Sciences and Furniture*

[3]*Institute of Mechanical Engineering,*

Warsaw University of Life Sciences – SGGW, Warsaw, Poland

jaroslaw_kurek@sggw.pl

Abstract. In this paper we introduce the enhanced drill wear recognition method, based on classifiers ensemble, obtained using transfer learning and data augmentation methods. Red, green and yellow classes are used to describe the current drill state. The first one corresponds to the case when drill should be immediately replaced. The second one denotes a tool that is still in a good condition. The final class refers to the case when a drill is suspected of being worn out, and a human expert evaluation would be required. The proposed algorithm uses three different, pretrained network models and adjusts them to the drill wear classification problem. To ensure satisfactory results, each of the methods used was required to achieve accuracy above 90% for the given classification task. Final evaluation is achieved by voting of all three classifiers. Since the initial data set was small (242 instances), the data augmentation method was used to artificially increase the total number of drill hole images. The experiments performed confirmed that the presented approach can achieve high accuracy, even with such a limited set of training data.

Key words: classifiers ensemble, convolutional neural networks, data augmentation, deep learning, tool condition monitoring.

1. Introduction

Manufacturing furniture can be lengthy and complex process, where each decision can result either in good quality product or large losses in income for the company, when entire piece or its elements do not meet quality requirements. One of many factors that can influence final outcome is drill sharpness, and pointing out exact moment when tool should be replaced to avoid budget losses due to poor product quality. Since manual observation of the drill state is not efficient enough, researching solutions that can automatize this stage was necessary.

Usually when tool wear problem is approached three classes would be used to describe its state: red, green and yellow. If evaluated tool would be assigned to first class, it would indicate that it already is in a poor state, and should be replaced immediately, since its further use in production process can generate loss for the company. Contrary, second class would denote tools that are still in good shape, and can be further used without any risk. Final class contains tools that are suspected of being to worn out, and therefore requiring human expert to manually evaluate their state – depending on given opinion, they can be either discarded or used further in production process.

Tool condition monitoring problem (or TCM) is not a new concept, similarly as evaluating drill condition specifically. When it comes to the latter, any described solutions usually required vast collections of different sensors for signal measurement. Collected signal data will then be processed and used to generate diagnostic features for each drill state. Commonly used signals can be related to: feed force, noise, cutting torque, vibration or acoustic emission [4]. While above setups can render accurate classifications, to achieve acceptable results usually numerous preprocessing stages will be required, including steps such as appropriate sensor or signal choice, generating and selecting best diagnostic features or building the classification model for given set of input data. Because of that, solutions from that set are quite complex, require lengthy preparations, and usually will be rather expensive due to various equipment parts required to even start measuring required signals. Moreover if later on some of selected input elements would prove unnecessary or inadequate, it would automatically generate loss (in terms of equipment required for measurements and time needed for setup configuration), and any mistakes during early stages may result in final product being not accurate enough. Additionally, in [1,2,3] even though authors took into account various features generated from different signals, accuracy of presented procedures did not exceed 90% for given task of recognizing three drill wear classes.

One of the goals of the presented approach (which should be treated as a continuation of the research presented in [5] and in our other paper in this volume [6]) is finding a more efficient and less complicated way (especially in terms of required equipment) for achieving the same classification task, with acceptable accuracy, exceeding the 90% threshold. Similarly as in [5, 6] the images of drilled holes will be used as a base for the assessment process. Since in the previous tests the transfer learning proved to give satisfactory results, it will also be used in the current approach, but instead of using a single model, three of the most commonly used pretrained convolutional neural networks (CNN) [7, 10] will be adjusted to our classification problem and used in the final classifier ensemble. Since the initial training set is rather small (242 images, representing three classes: 102 samples for green, 60 samples for yellow and 80 samples for red drills), our solution also includes the data augmentation method, to artificially increase the number of examples, similarly as in [7] and [6].

In this paper an approach is proposed that includes the creation of the classifier ensemble, composed of three pretrained convolutional neural networks. Out of the available

(a) Green (b) Yellow (c) Red

Fig. 1: Examples of holes produced by each drill wear class: green (a), yellow (b) and red (c).

solutions we chose AlexNet, VGG19 and VGG16 and adjusted them to our classification problem using transfer learning and data augmentation methodologies (with additional requirement that each individual model will achieve accuracy above 90%). The final result is achieved by counting votes from all the classifiers. Similarly as in previous approaches to this subject, we are using three classes to describe the drill wear state, while the initial image set used for training and evaluation is the same as in [5, 6].

2. Data augmentation methodology

Data samples used in our experiments were collected in cooperation with Faculty of Wood Technology at Warsaw University of Life Sciences, using standard Buselatto JET 100 CNC vertical machining centre. For the test purposes drilling process was performed on standard laminated chipboard (Kornopol U 511 SM) that is typically used in furniture industry. Dimensions of the test piece were $150 \times 35 \times 18$ mm. Regular 12 mm FABA drill equipped with a tungsten carbide tip was used for the drilling process.

Data set that is used for CNN learning has the same structure as the one presented in [5]. From 242 examples present in initial set, we have the following attribution for each of the three classes: 102 images for green, 60 images for yellow and 80 images for red. Examples representing each of the defined classes are shown in Fig. 1 (images in this Figure and in Fig. 2 are similar to those used in our previous papers on drill wear classification due to that we used the same initial set of images, and the image transformations used in [6]).

Since the initial set is not sufficient for most of deep learning approaches, we decided to artificially expand it, using basic image operations. 18 different operations were used

Tab. 1: Parameters of the pretrained convolutional neural networks chosen for classifier ensemble.

CNN	**Depth**	**Size [MB]**	**Parameters** [$\times 10^6$]	**Input Image Size**
AlexNet	8	227	61.0	227×227
VGG16	16	515	138	224×224
VGG19	19	535	144	224×224

for each of the original images from the initial data set. After performing those our training data set consisted of total 4598 examples, where 1938 samples were assigned to green class, 1140 samples to yellow class and 1520 samples for red class. Image operations that were used for expanding the original data set are as follows:

1. ColorToGrayscale – convert image to gray-scale values.
2. ColorBrightJitter – adjust brightness of image by random offset in range [-0.3, -0.1].
3. ColorContrast1 – adjust contrast of image by scale factor in range [1.2, 1.4].
4. ColorContrast2 – adjust contrast of image by scale factor in range [1.4, 1.6].
5. ColorHueJitter1 – change image hue by random offset in range [0.05, 0.15].
6. ColorHueJitter2 – change image hue by random offset in range [0.15, 0.30].
7. NoiseGauss – add Gaussian noise to the image.
8. Noise1 – add salt and pepper noise to the image, with strength factor 0.2
9. Noise2 – add salt and pepper noise to the image, with strength factor 0.4
10. Reflection – create a reflection that transforms original image by flipping it in each dimension with 50% probability.
11. Rotate30 – rotate image by random angle in range [0, 30].
12. Rotate45 – rotate image by random angle in range [30, 60].
13. Rotate90 – rotate image by random angle in range [60, 90].
14. Rotate120 – rotate image by random angle in range [90, 120].
15. Scale1 – scale image by random factor in range [1.2, 1.5]
16. Scale2 – scale image by random factor in range [0.8, 0.9]
17. Shear – shear transformation (horizontal), with angle selected randomly in [-30, 30].
18. Translate50 – translate image both vertically and horizontally by random number of pixels in range [-50,50].

The final operation that was required for artificially augmented data set was the image size adjustment, since each of pretrained convolutional neural networks that was chosen for our approach requires images to be saved with specified dimensions. Example results for each of denoted image transformations are presented at Fig. 2. Specific description of each of those operations are in [20]. Parameters of selected models are presented in Table 1.

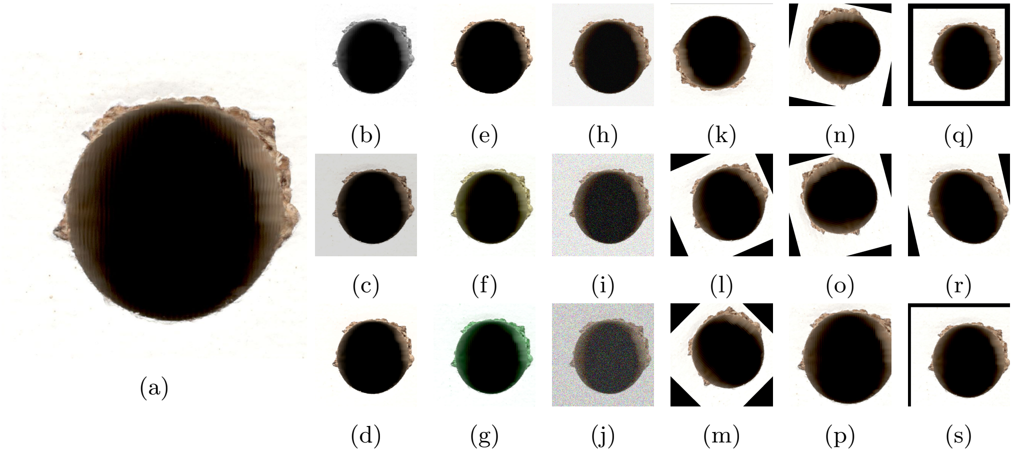

Fig. 2: Examples of augmented data set for a single original image (**a**): (**b**) ColorToGrayscale, (**c**) ColorBrightJitter, (**d**) ColorContrast1, (**e**) ColorContrast2, (**f**) ColorHueJitter1, (**g**) ColorHueJitter2, (**h**) NoiseGauss, (**i**) Noise1, (**j**) Noise2, (**k**) Reflection, (**l**) Rotate30, (**m**) Rotate45, (**n**) Rotate90, (**o**) Rotate120, (**p**) Scale1, (**q**) Scale2, (**r**) Shear, (**s**) Translate50

3. Classical deep learning approach (CNN)

When it comes to classification problems, in recent years one of most commonly used solutions (especially if different types of images are considered) are different deep learning approaches [8, 9, 10, 11]. Among those CNN (convolutional neural network) would be typically used either as a main function, or as one of the core parts of entire approach. Each of hidden layers of nonlinear processing would begin extracting diagnostic features (starting from elements such as points, edges, corners, etc.), while each successive layer will slowly derive higher level features from the more basic ones, creating hierarchical data representation of input image. Thanks to that there is no need for the user to manually specify diagnostic features for given problem, as they will be obtained during training process. Additionally, features found in such a way tend to be more universal and usually give good results for different types of images, with acceptable accuracy.

Deep learning approach uses blocks of pixels with fixed size for analysis (i.e. 5 by 5, 10 by 10 or in case of larger images 15 by 15 pixels), contrary to regular methods which iterate through image pixels. In this approach few different filtrations are used to achieve final results first starting with linear filtration. In the next step nonlinear transformation is applied to filter output signals (usually rectified linear unit or ReLU). Finally, reducing size of the actually processed blocks, pooling operation is performed.

Main difference between deep learning and classical approach lies in diagnostic features extraction. When classical machine learning is considered this process is considered as separate step, usually consisting of different stages, often complicated in themselves. In case of deep learning all features are generated automatically as an embedded internal process that is done in hidden layers. Extracted features can later be used as attributes for external classifier (i.e. support vector machine) or input values for softmax classifier (which is an integral part of CNN). For detailed description of CNN solution used as reference in this work see [5].

4. Transfer learning approach (CNN AlexNet)

Database used in our approach consists of 242 images for three defined classes (102 for green, 60 for yellow and 80 for red class). Such number of training examples is not sufficient enough for training CNN from the scratch. Therefore, in order to ensure better classification accuracy, using model pretrained on much larger set of different and unrelated images was one of possible solutions. Using transfer learning methodology [13, 14], AlexNet model, created by Krizhevsky, Sutskever and Hinton [12, 13, 14] and prepared with Matlab [15], was applied to our classification problem. AlexNet was pretrained using more than a million images, where each of them represented one of 1000 defined classes [15, 16].

In our approach structure of used pretrained CNN is made of 9 layers, containing 25 sublayers. Softmax classifier, which is part of this solution, calculates i-th component of output vector (which represents given number of classes M) by inserting u-vector in the following equation:

$$\mathrm{softmax}(u)_i = \frac{\exp(u_i)}{\sum_{j=1}^{m} \exp(u_j)} \tag{1}$$

Softmax function uses exponential normalization and its values range from 0 to 1. Final classifier value is treated as possibility of specified class, while component with highest value will indicate chosen class. To calculate softmax function for our case of three classes (M=3), cross entropy has been used.

AlexNet requires for all images to have the same size, which is 227 by 227 by 3, so all input images used in experiments were additionally adjusted to meet those specifications. In tested solution from all trained layers, first six were applied without any changes and implemented the same mechanism for creating diagnostic features as in classes of images used in pretraining stage of AlexNet. The adjustments were made in last three, fully connected layers, to better adapt entire setup for our specific classification problem.

5. Classifier ensemble

In the previous work [5] the pretrained AlexNet CNN model achieved promising results (see Table 2), but was clearly losing in terms of training data. Therefore, we decided to expand the data set in our current approach. Since even without the need to use specialized equipment in the data acquisition process, collecting samples of drill hole images could take a considerable amount of time, it was decided that artificial data augmentation will be used, with basic image operations. This approach would also ensure that user needs to take minimal number of actions in order to achieve acceptable accuracy of final classifier. Total of 18 operations for different types of image transformations were chosen (like rotation, adding noise, scaling etc.), and all elements from initial data set were processed through each of them, with random values for most operations. Such approach ensured maximal diversity in final set, without adding results that would be too different from accepted standards.

To further increase overall performance of our solution, instead of using single classifier we decided to apply classifier ensemble. We chose three of the most popular, pretrained CNN networks: AlexNet, VGG16 and VGG19. Taking into account previous experiments and CNN general properties, we assumed that this kind of approach will heighten final algorithm accuracy, without the need to use more complex solutions (like adding SVM as a final classifier as it was done in [5]) or significantly increasing calculation time (as with classical CNN, without using transfer learning methodology).

Each of the classifiers was prepared separately, while final class recognition is done by all three classifiers voting. For our approach, class that got most votes is the one that is assigned to presented image. For overview of learning process for each of chosen classifiers refer to Fig. 3.

6. Results

Initial data set prepared for the training and experiment purposes contained 242 images that represented three defined classes. After data augmentation process was performed, the set was extended to 4598 images, where each of new images was generated from image in original set, using basic image operations (see Fig. 2).

Due to different algorithms used to evaluate final solution, we use few methodologies to prepare and evaluate out experiments. For the first two algorithms only initial data set was used (242 images). This set was randomly divided into 10 subsets, and in further experiments 9 of those were used in training process, while remaining set was used for testing. We repeated those experiments 10 times in 10-fold cross validation mode, with test data being exchanged for each subsequent run. For three pretrained classifiers, using artificially extended data set, we used similar approach, but in that case 95% of data was assigned as training set, and remaining 5% was used for testing purposes. For classifier

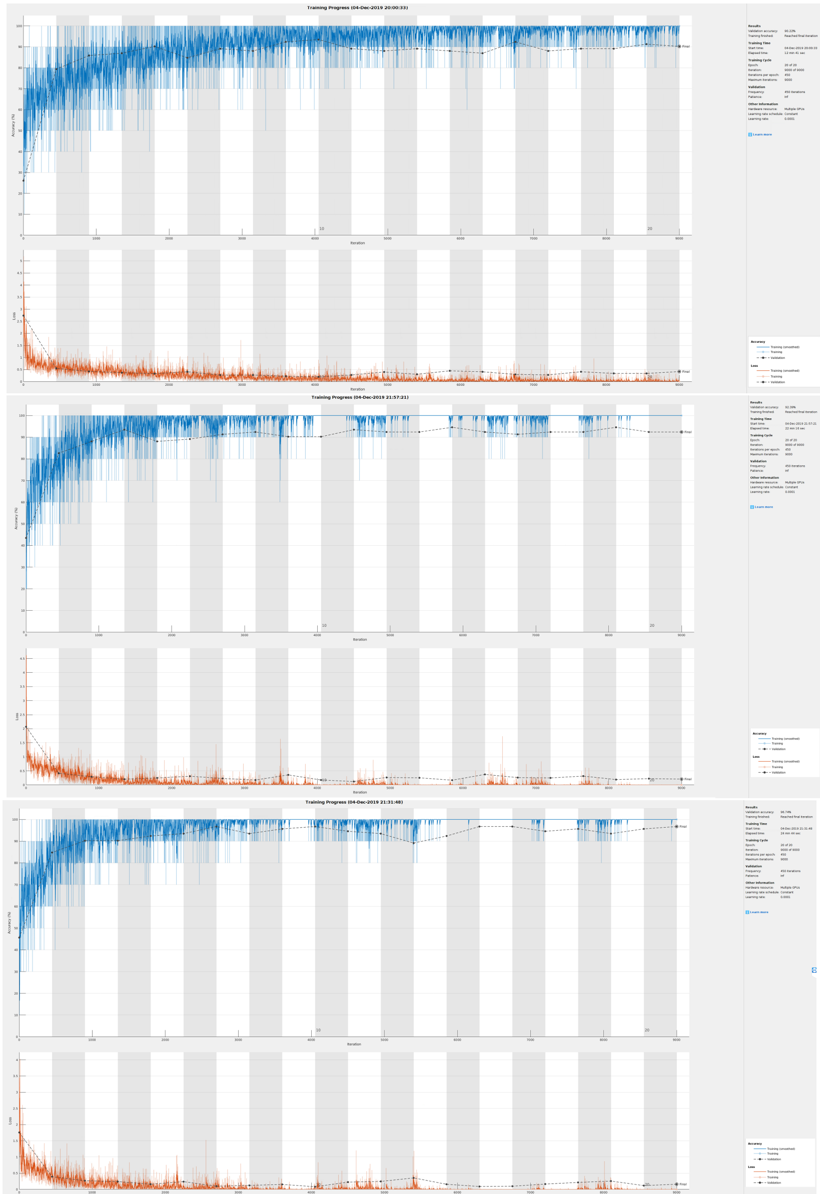

Fig. 3: Outline of training process for CNN models with augmented data set: AlexNet (top), VGG16 (middle) and VGG19 (bottom).

Tab. 2: Accuracy of chosen algorithms applied to the problem of drill condition classification with 3 classes (green, yellow and red). First two algorithms use 242 input images, while rest of the algorithms used artificially expanded set of 4598 images.

No.	Deep learning algorithm	Accuracy[%]
(1)	Standard CNN	35%
(2)	Pretrained CNN	85%
(3)	AlexNet CNN	94.30%
(4)	VGG16 CNN	92.39%
(5)	VGG19 CNN	96.73%
(6)	Classifier ensemble	95.65%

ensemble solution, trained classifiers were voting for final class assignment. Obtained results are prepared in the form of mean value of class recognition accuracy and are presented in Table. 2. Subsequent table rows refer to:

1. **Standard CNN** – CNN which learned from scratch.
2. **Pretrained CNN** – CNN using pretrained AlexNet model with stoftmax classifier.
3. **AlexNet** – pretrained AlexNet CNN, using augmented dataset.
4. **VGG16** – pretrained VGG16 CNN, using augmented dataset.
5. **VGG19** – pretrained VGG19 CNN, using augmented dataset.
6. **Classifier ensemble** – solution using AlexNet, VGG16 and VGG19 models, adjusted to presented classification problem with classifier voting as a method to select final class assignment.

As in previous work [5] it is clearly visible that using pretrained CNN for such a small dataset has a great advantage. Moreover, by artificially expanding the data set we were able to successfully imitate a larger number of samples, and to increase the overall accuracy both for single classifiers (AlexNet – 94.30% accuracy, VGG16 – 92.39% accuracy and VGG19 – 96.73% accuracy) and for the final solution, combining votes of all three of them (95.65% accuracy). Even though VGG19 classifier achieved higher accuracy than the final solution, the presented approach still achieved more than acceptable outcomes and has grater generalization capabilities.

7. Conclusion

In this paper we presented a method for drill wear state recognition, based on drilled holes. Our approach improves on previous solutions [5] by using classifier ensemble of three most popular, pretrained convolutional neural networks (AlexNet, VGG16 and VGG19), bringing into play transfer learning and data augmentation methodologies to further improve final class recognition accuracy. Previously to increase precision of given classifications required either more complex solutions (like using SVM as a final classification method), or significantly larger data set used for learning (which could result in prolonged computations, like it was the case with CNN learning from scratch in [7], where similar accuracy of 95.5% was achieved for 33300 images, with training time that lasted over 20 hours).

Obtained results confirm that presented approach was able to achieve satisfying accuracy while predicting drill wear state. Outcome of classifier ensemble solution was compared both to traditionally trained CNN and previous version of current method, using pretrained AlexNet without data augmentation techniques, and accuracy rate was better in both cases. Additionally final algorithm was able to achieve higher precision than solution using SVM as a final classifier (see [5]). When it comes to training time, none of the classifiers in final solution required more than 25 minutes before this process was finished (12 min 41 s for AlexNet, 22 min 16 s for VGG16 and 24 min 44 s for VGG19). As it was the case in the previous approach, the used pretrained networks required only minimal adjustments in the last layers of CNN, before they were redy to use in the presented classification problem. The entire process was not very time consuming, while three resulting models had good generalization properties for drill wear class recognition, both individually, and as the classifier ensemble.

References

[1] K. Jemielniak, T. Urbański, J. Kossakowska, S. Bombiński. Tool condition monitoring based on numerous signal features. Int. J. Adv. Manuf. Technol., vol. 59, pp. 73-81, 2012.

[2] S. S. Panda, A. K. Singh, D. Chakraborty, S. K. Pal. Drill wear monitoring using back propagation neural network. Journal of Materials Processing Technology, vol. 172, pp. 283-290, 2006.

[3] R. J. Kuo, Multi-sensor integration for on-line tool wear estimation through artificial neural networks and fuzzy neural network. Engineering Applications of Artificial Intelligence, vol. 13, pp. 249-261, 2000.

[4] J. Kurek, M. Kruk, S. Osowski, P. Hoser, G. Wieczorek, A. Jegorowa, J. Górski, J. Wilkowski, K. Śmietańska, J. Kossakowska. Developing automatic recognition system of drill wear in standard laminated chipboard drilling process. Bulletin of the Polish Academy of Sciences. Technical Sciences, vol. 64, pp. 633-640, 2016.

[5] J. Kurek, G. Wieczorek, M. Kruk, A. Jegorowa, S. Osowski. Transfer learning in recognition of drill wear using convolutional neural network. 18th International Conference on Computational Problems of Electrical Engineering (CPEE), pp. 1-4. IEEE. September 2017.

[6] J. Kurek, I. Antoniuk, J. Górski, A. Jegorowa, B. Świderski, M. Kruk, G. Wieczorek, J. Pach, A. Orłowski, and J. Aleksiejuk-Gawron. Data augmentation techniques for transfer learning improvement in drill wear classification using convolutional neural network. *Machine Graphics & Vision*, 28(1/4):3–12, 2019. doi:10.22630/MGV.2019.28.1.1.

[7] J. Kurek, B. Świderski, A. Jegorowa, M. Kruk, S. Osowski. Deep learning in assessment of drill condition on the basis of images of drilled holes. Proc. SPIE 10225 Eighth International Conference on Graphic and Image Processing (ICGIP 2016), pp. 102251V, February 8, 2017.

[8] L. Deng, D. Yu. Deep Learning: Methods and Applications. Foundations and Trends in Signal Processing, vol. 7, pp. 3-4, 2014.

[9] Y. Bengio. Learning Deep Architectures for AI. Foundations and Trends in Machine Learning, vol. 2, no. 1, pp. 1-127, 2009.

[10] I. Goodfellow, Y. Bengio, A. Courville. Deep learning. MIT Press, 2016.

[11] J. Schmidhuber. Deep Learning in Neural Networks: An Overview. Neural Networks, vol. 61, pp. 85-117, 2015.

[12] A. Krizhevsky, I. Sutskever, G. Hinton. Image net classification with deep convolutional neural networks. Advances in Neural Information Processing Systems, vol. 25, pp. 1-9, 2012.

[13] O. Russakovsky, J. Deng, H. Su et al. ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV), vol. 115, no. 3, pp. 211-252, 2015.

[14] BVLC AlexNet Model. Online: `https://github.com/BVLC/caffe/tree/master/models/bvlc_alexnet`.

[15] Matlab 2017a – User manual, Natick, MA, USA: The MathWorks, Inc, 2017.

[16] ImageNet. Online: `http://www.image-net.org`.

[17] B. Scholkopf, A. Smola. Learning with Kernels, Cambridge: MIT Press, 2002.

[18] M. Kruk, B. Świderski, S. Osowski, J. Kurek, M. Słowińska, I. Walecka. Melanoma recognition using extended set of descriptors and classifiers. Eurasip Journal on Image and Video Processing, vol. 43, pp. 1-10, 2015.

[19] V. N. Vapnik. *Statistical Learning Theory*, New York: Wiley, 1998.

[20] Description of Matlab image transformations. Online: `https://www.mathworks.com/help/deeplearning/examples/image-augmentation-using-image-processing-toolbox.html`.

BCT Boost Segmentation with U-net in TensorFlow

Grzegorz Wieczorek, Izabella Antoniuk, Jarosław Kurek, Bartosz Świderski,
Michał Kruk, Jakub Pach, Arkadiusz Orłowski
Institute of Information Technology
Warsaw University of Life Sciences – SGGW, Warsaw, Poland
grzegorz_wieczorek@sggw.pl

Abstract. In this paper we present a new segmentation method meant for boost area that remains after removing the tumour using BCT (breast conserving therapy). The selected area is a region on which radiation treatment will later be made. Consequently, an inaccurate designation of this region can result in a treatment missing its target or focusing on healthy breast tissue that otherwise could be spared. Needless to say that exact indication of boost area is an extremely important aspect of the entire medical procedure, where a better definition can lead to optimizing of the coverage of the target volume and, in result, can save normal breast tissue. Precise definition of this area has a potential to both improve the local control of the disease and to ensure better cosmetic outcome for the patient. In our approach we use U-net along with Keras and TensorFlow systems to tailor a precise solution for the indication of the boost area. During the training process we utilize a set of CT images, where each of them came with a contour assigned by an expert. We wanted to achieve a segmentation result as close to given contour as possible. With a rather small initial data set we used data augmentation techniques to increase the number of training examples, while the final outcomes were evaluated according to their similarity to the ones produced by experts, by calculating the mean square error and the structural similarity index (SSIM).

Key words: breast cancer, breast conserving therapy, image segmentation, U-net, Keras, TensorFlow

1. Introduction

Breast cancer is a type of tumour that develops in breast region when cells located there rapidly divide. In result, those rapidly dividing cells can grow out of control, finally forming large, abnormal masses. It can be characterized as anomalous swelling of a part of the body or growth of tissue, where there is no local inflammation. Tumours can be roughly divided into two main types: benign and malignant. First type of cancer occurs, when its cells spread only to areas surrounding tumour, while second type means that cells travelled by lymph nodes to different organs, spreading through either body and producing metastasis. Breast tumour can be seen with X-ray and can be felt during standard test as a lump, that is irregular in shape. Such lumps will be fixed to the tissue in the breast and can have firm or solid filing because of that. Depending from the specific case, they can be painful or painless, also tumours can start from different parts of breast. Usually breast cancer will start in the ducts (ductal cancers), while some of them will start in the glands (lobular cancer).

Recent years have shown advancements both in image diagnostics as well as available treatments. At this time prevention is discussed besides potential medication and surgery. Because the actual disease is better understood, there were considerable changes in treatment methodologies. Such shift allowed multidisciplinary approach to the problem itself, changing its understanding and finding more efficient methods for both diagnosing breast cancer as well as ways to introduce improvements during different stages of treatment [1].

One of the core problems that influence final stages of the breast conserving therapy is denoting boost area that remains after removing recognized tumour for further radiation treatment. When boost area is not correctly marked it can result in actual treatment missing its target (completely or partially), while healthy tissue, that otherwise could be spared will be targeted. Even when entire boost area is covered by the procedure, from cosmetic point of view it is best to limit its overlay with healthy cells, saving as much of the breast as possible. Because of above reasons more precise and efficient method for denoting boost area from CT images was necessary.

When it comes to different visual recognition tasks, especially in recent years, deep convolutional neural networks have performed considerably well (see [5,6]). Since for the convolutional neural network (or CNN) usually large amount of data is required before it will render acceptable results, application to medical images was rather limited. In case of such applications sample sizes are usually rather small (like in our case, when we have 246 samples total), while acquisition methods are lenghty and complicated. Another problem is that while typically CNN would be applied to classification task, in medical images usually additional data, such as localization is needed, therefore requiring for any proposed algorithm to assign class value to each pixel in input image.

One of recent solutions called U-net is focusing on medical image segmentation for determining location of single cells in given samples [2]. Proposed network uses two paths: contracting and expansive. Contracting paths consists of repeated application of two 3 by 3 unpadded convolutions, each followed by a rectified linear unit (ReLU) and a 2 by 2 max pooling operation with stride 2 for downstamping. For the expansive path each step includes upsampling feature map, next followed with 2 by 2 up-convolution (halving number of feature channels), concatenation with corresponding cropped feature map and finally two 3 by 3 convolutions, each of them followed by ReLU. In that approach both small initial sample size as well as other problems concerning medical image processing are considered, and we use proposed U-net network as a base for out further research when it comes to our case of post-surgery boost area segmentation for breast cancer. For outline of U-net network structure see Fig. 1

Due to its large success we decided to use U-net as a base of our solution and adjust it to boost area segmentation problem. In our approach we use CT images, that were

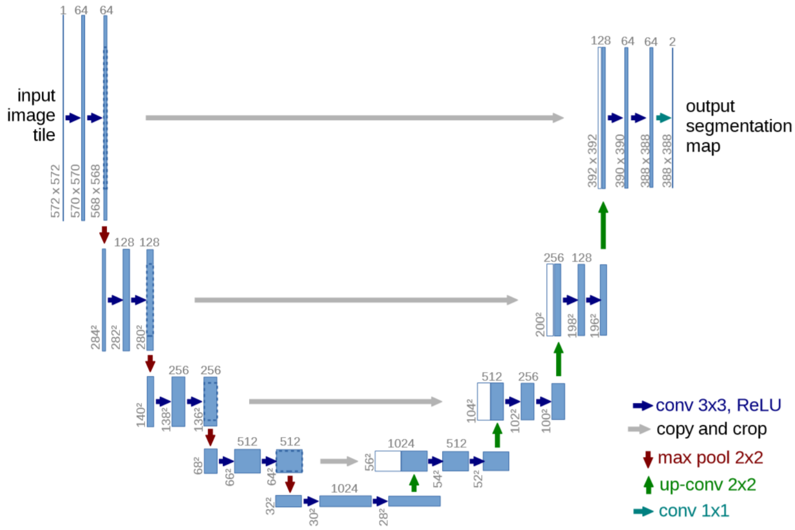

Fig. 1: U-net network architecture example for 32x32 pixels in lowest resolution [2].

originally saved in DICOM format. Each sample contained numerous slices along with expert outlines denoting boost area for those examples (see Fig. 2 for example slice along with GT expert outline and Fig. 3 for more specific indication of boost area). For those slices segmentation process was performed, in which we tried to achieve region selection as close to area pointed out by the expert as possible. We use Keras U-net network along with TensorFlow as a base for our methodology and adjust them to this specific problem using data augmentation methodologies for increasing total number of instances for the training process (to further increase accuracy of final segmentation algorithm). Please refer to [8] for Keras library documentation and [9] for TensorFlow.

2. Network architecture

Image segmentation for medical purposes can be tiresome and time consuming process, especially when high precision is necessary, like it is the case with boost area denotation. Usually for such application general machine learning algorithms and convolutional neural networks specifically would be prepared. Main issue occurring in this type of problems is small number of available samples, hence network learning from scratch cannot be used, simply because it would not achieve acceptable precision. Another problem lies with the fact, that while CNN is quite good for classification problems it requires some adjustments for area denotation, some of which were addressed in [2]. In one of previous

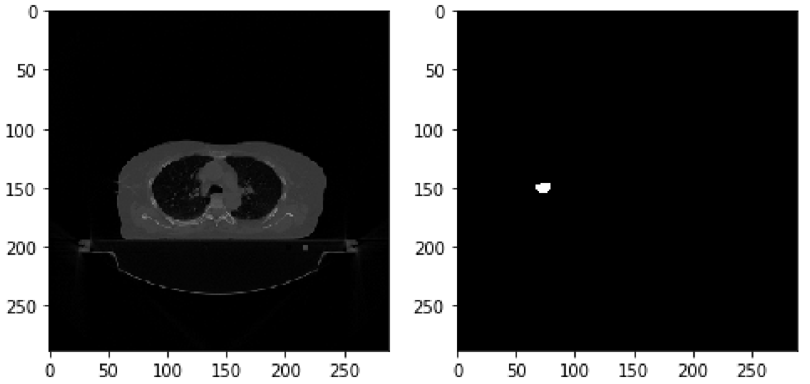

Fig. 2: Example of a single CT image (left), with expert outline of boost area (right).

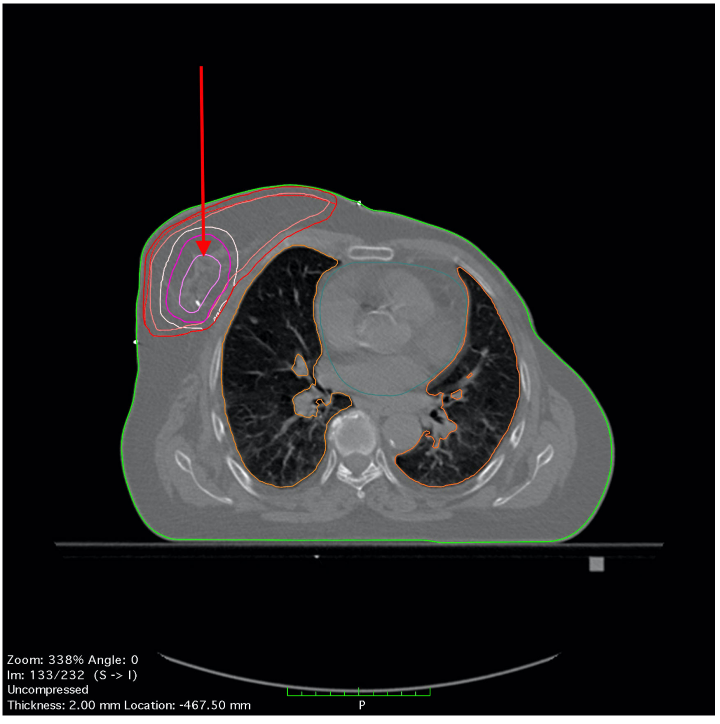

Fig. 3: Identification of area that needs to be outlined relatively to entire CT image.

works [7]CNN was used for drill wear recognition. Similarly as in this case, very small number of samples were available for training (only 242 examples for three recognized classes). Presented approach showed, that using pretrained model and adjusting it to different problem by exchanging classification layers is a very promising approach. We chose similar solution, basing our algorithm on Keras version of U-net algorithm.

Since our images have size 288 by 288 pixels, we needed to adjust network structure to accommodate that. To better fit entire solution to boost area segmentation problem, we removed last 3 layers of Keras U-net network, and retrained them using available samples. Full network structure is described in Table. 1.

3. Training

Solution presented in this paper is based on Keras version of U-net network, that was adjusted to accommodate different image size (288x288 in our case), and had last [??] layers retrained to better fit our segmentation problem. Initial set of data contained 246 samples. We used 216 of them for the training process, 24 for validation and remaining 6 samples were denoted as test data.

Such small dataset is a common occurrence in case of medical images, therefore instead of trying to get more original images, we used data augmentation to artificially increase number of samples we could use. By performing operations such as scaling, rotation or simple translations we could efficiently increase number of available images, without the need to perform lengthy operations required for obtaining original images. Such approach reduces time required for preparations, while still significantly increasing obtained accuracy (augmented samples will not differ greatly from original images, while still increasing overall neural network accuracy). Example images after performing simple image operations are presented at Fig. 4.

4. Experiments

After preparing and augmenting initial data set, we used obtained images to train Keras U-net. All experiments presented in this paper were done on Ubuntu operating system (version 18.04.3 LTS), with 2x AMD Ryzen Threadripper 2990WX 32-Core Processor, 128GM of Ram, and two graphic cards Nvidia TITAN RTX. We chose Python [10] as our programming language (version 3.7.4) since due to its specifics as well as availability of extensive libraries it was most comprehensive approach for combining different solutions, including Keras library (version 2.3.1) and TensorFlow (verision 2.0.0).

Evaluation was performed for all 6 samples of test data. Obtained segregation results are presented at Fig. 5. To further evaluate final regions selected by prepared solution, we calculate mean squared error according to equation 1:

Tab. 1: Structure of BCT boost segmentation network.

Layer (type)	**Output shape**	**Param #**	**Connected to**
input_8 (InputLayer)	(None, 288, 288, 1)	0	
conv2d_64 (Conv2D)	(None, 288, 288, 8)	80	input_8[0][0]
max_pooling2d_22 (MaxPooling2D)	(None, 144, 144, 8)	0	conv2d_64[0][0]
conv2d_65 (Conv2D)	(None, 144, 144, 16)	1168	max_pooling2d_22[0][0]
max_pooling2d_23 (MaxPooling2D)	(None, 72, 72, 16)	0	conv2d_65[0][0]
conv2d_66 (Conv2D)	(None, 72, 72, 32)	4640	max_pooling2d_23[0][0]
max_pooling2d_24 (MaxPooling2D)	(None, 36, 36, 32)	0	conv2d_66[0][0]
conv2d_67 (Conv2D)	(None, 36, 36, 32)	1056	max_pooling2d_24[0][0]
up_sampling2d_22 (UpSampling2D)	(None, 72, 72, 32)	0	conv2d_67[0][0]
concatenate_22 (Concatenate)	(None, 72, 72, 64)	0	up_sampling2d_22[0][0], conv2d_66[0][0]
conv2d_68 (Conv2D)	(None, 72, 72, 32)	8224	concatenate_22[0][0]
up_sampling2d_23 (UpSampling2D)	(None, 144, 144, 32)	0	conv2d_68[0][0]
concatenate_23 (Concatenate)	(None, 144, 144, 48)	0	up_sampling2d_23[0][0], conv2d_65[0][0]
conv2d_69 (Conv2D)	(None, 144, 144, 24)	4632	concatenate_23[0][0]
up_sampling2d_24 (UpSampling2D)	(None, 288, 288, 24)	0	conv2d_69[0][0]
concatenate_24 (Concatenate)	(None, 288, 288, 32)	0	up_sampling2d_24[0][0], conv2d_64[0][0]
conv2d_70 (Conv2D)	(None, 288, 288, 16)	2064	concatenate_24[0][0]
conv2d_71 (Conv2D)	(None, 288, 288, 64)	1088	conv2d_70[0][0]
dropout_8 (Dropout)	(None, 288, 288, 64)	0	conv2d_71[0][0]
conv2d_72 (Conv2D)	(None, 288, 288, 1)	65	dropout_8[0][0]

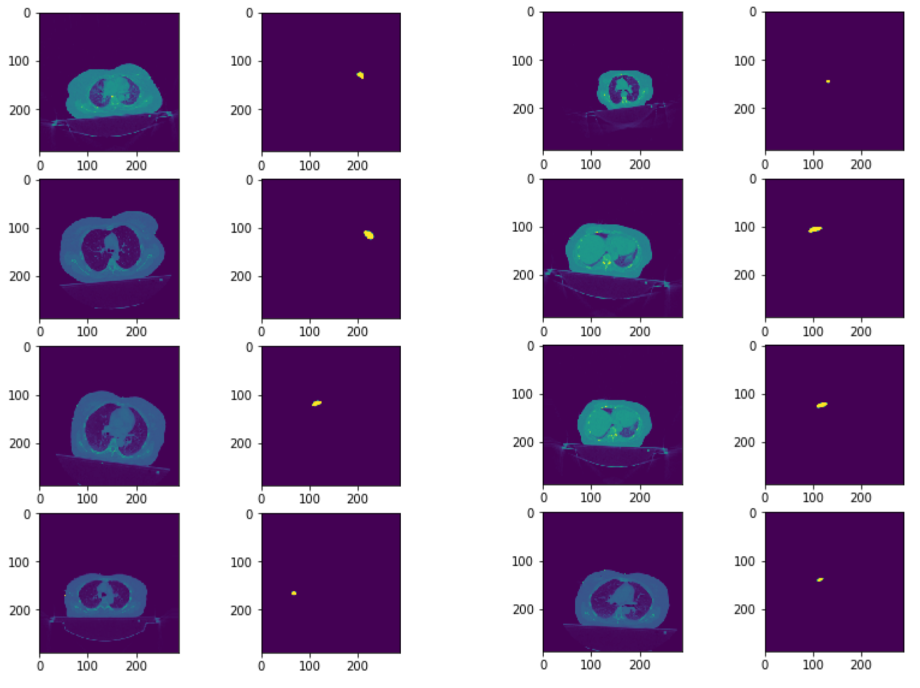

Fig. 4: Examples of images obtained after performing simple image transformations (i.e. rotation, scaling, translation), along with denoted boost area.

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2 \tag{1}$$

Second used parameter is SSIM (Structural Similarity Index) [4] in the form of equation 2.

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{2}$$

Using SSIM is not a standard solution, but we decided, that in case of comparing two selected regions determining exactly how similar our solution is to the target outline will be an important factor. SSIM index values range from -1 to 1, where first value denotes completely different elements, without common parts, while second value denotes perfect similarity. Results obtained for each of test samples are presented in Table. 2.

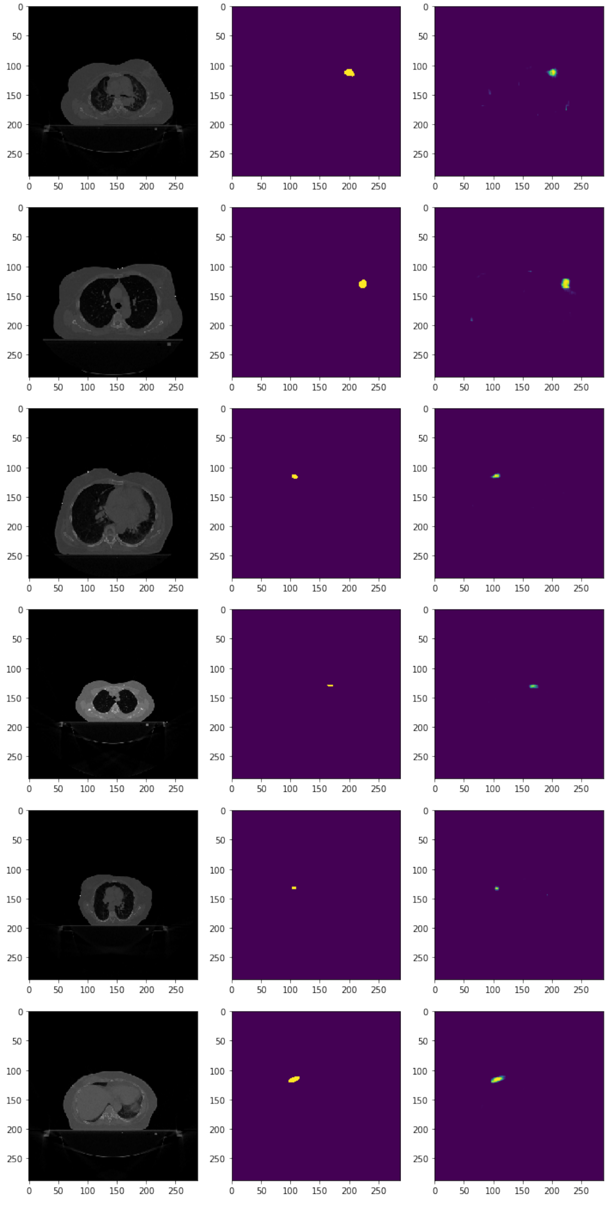

Fig. 5: Example segregation results: original image (left), GT denoted by expert (middle) and result obtained with our solution (right).

Tab. 2: Mean square error (MSE) and structural similarity index (SSIM) obtained for all test samples by our boost area segmentation method.

Sample number	**MSE**	**SSIM**
1	0.0003858	0.9975356
2	0.0008077	0.9965344
3	0.0004701	0.9964890
4	0.0004581	0.9971912
5	0.0003375	0.9982249
6	0.0004581	0.9980749

5. Conclusion

In this paper we present machine learning approach to segmentation problem of boost area in breast conserving therapy, which is a result of performed surgery. Since outline obtained during that process will later be used to determine region for radiation treatment, precise and exact definition is crucial for limiting exposure of healthy tissue and minimizing risk of missing the target. Our solution uses U-net network available in Keras library, modified and adjusted to specified problem, and is able to accurately denote specified region. We used relatively small number of initial samples, while entire set used for network training was artificially expanded using simple image operations.

We evaluated our results using MSE (mean square error) and SSIM (structural similarity index) as main parameters, apart from visual confirmation of final outline quality. For given set of test examples (6 samples in total) we were able to achieve satisfactory results. All of samples achieved similarity index above 0.99 (with lowest similarity index equal 0.9964890 for sample number 3), while calculated errors remained at relatively low level (highest obtained error equalled 0.0008077 for sample number 2). Visually all obtained results were very close to the GT prepared by expert.

Overall presented solution has great potential and requires further study. Our approach still needs to be evaluated on larger set of test images. It also might require additional methods for more complex cases (i.e. when more than one area will need to be outlined). Despite that current version of presented solution was able to generate precise outlines of boost areas in given set of CT images, and can be used during medical procedures as an support tool for the expert.

References

[1] M. Kaufmann, G. Von Minckwitz, J. Bergh, p. F. Conte, S. Darby, et al. Breakthroughs in research and treatment of early breast cancer: an overview of the last three decades. Archives of gynecology and obstetrics, 288(6), 1203-1212, 2003.

[2] O. Ronneberger, P. Fischer, T. Brox. U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention. (pp. 234-241). Springer, Cham, 2015, October.

[3] C. J. Hansen, E. de Winton, S. Guglani, E. Vamvakas, D. Willis, B. H. Chua. Target localisation for tumour bed radiotherapy in early breast cancer. Journal of medical imaging and radiation oncology, 56(4), 452-457, 2012.

[4] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing, 13(4), 600-612, 2004.

[5] R. Girshick, J. Donahue, T. Darrell, J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014.

[6] A. Krizhevsky, I. Sutskever, G. E. Hinton. Imagenet classification with deep convolutional neural networks. In: NIPS. pp. 1106–1114, 2012.

[7] J. Kurek, G. Wieczorek, M. Kruk, A. Jegorowa, S. Osowski. Transfer learning in recognition of drill wear using convolutional neural network. 18th International Conference on Computational Problems of Electrical Engineering (CPEE) (pp. 1-4). IEEE. September 2017.

[8] Keras deep learning library for python. Online: `https://keras.io/`

[9] TensorFlow machine learning platform [online]. Online: `https://www.tensorflow.org/`

[10] Python main web page. Online: `https://www.python.org/`

Textural Features Based on Run Length Encoding in the Classification of Furniture Surfaces with the Orange Skin Defect

Jakub Pach[1], Leszek J. Chmielewski[1], Arkadiusz Orłowski[1], Michał Kruk[1], Jarosław Kurek[1], Bartosz Świderski[1], Izabella Antoniuk[1], Grzegorz Wieczorek[1], Katarzyna Śmietańska[2], Jarosław Górski[2]
[1]*Institute of Information Technology*
[2]*Institute of Wood Sciences and Furniture*
Warsaw University of Life Sciences – SGGW, Poland

Abstract. Textural features based upon thresholding and run length encoding have been successfully applied to the problem of classification of the quality of lacquered surfaces in furniture exhibiting the surface defect known as *orange skin*. The set of features for one surface patch consists of 12 real numbers. The classifier used was the one nearest neighbour classifier without feature selection. The classification quality was tested on 808 images 300 by 300 pixels, made under controlled, close-to-tangential lighting, with three classes: *good*, *acceptable* and *bad*, in close to balanced numbers. The classification accuracy was not smaller than 98% when the tested surface was not rotated with respect to the training samples, 97% for rotations up to 20 degrees and 95.5% in the worst case for arbitrary rotations.

Key words: quality inspection, furniture surface, orange skin, textural features, run length coding, thresholded image, one nearest neighbour, leave-one-out testing.

1. Introduction

In various sectors of the manufacturing industry, the computer vision methods are used to perform inspection and measurement tasks, while in the furniture industry it seems that the inspection with the unarmed human eye is the most commonly used tool. In this way, the tasks like the evaluation of esthetic quality of an object or a surface can be performed in a straightforward way and at a moderate cost. This state does not exclude the research on computerized image-based inspection in the application to furniture.

The application of our interest will be the classification of a painted surface with respect to the defect known as *orange skin* or *orange peel*. Up till now, the research on this defect was performed to a limited extent. In [10] the images of *orange peel* were generated and its visibility for humans was studied, but neither automatic detection nor classification was considered. In [2] a complex system of moving lights and cameras was investigated which improved the visibility of a variety of defects, including the *orange peel*, so that the local adaptive thresholding could be used as the defect detection method. The defects on furniture were mentioned within the general domain of defect detection in [9]. The quality of raw materials was considered in [17]. In many patents the methods to perform the painting process so that orange skin is removed or avoided, for example [1], but with no reference to image analysis. This state of the lack of interest in automatic image-based inspection in the furniture industry is interesting in view of that in the timber industry the analysis of a whole range of raw wood defects is widely applied and described [3, 16]).

The quality of wood products has been the domain of our interest in a number of papers (shape and dimensional accuracy in [5, 13, 14, 15] and surface quality in [5, 6, 7, 11, 21]). In particular, in [5] we have demonstrated that structured light-based scanning of the surface can not reveal the orange skin defect reliably. In [6] we have tested the applicability of such conventional image processing techniques like special lighting, differentiation and thresholding. The use of conventional and novel textural features was tested in [7] and [11]. A set of seven feature selection methods were used to find the best features for this task [21], with the best attained classification precision of 95.8%.

Finding the best features in the sense of precision as well as calculation speed is still an interesting problem. As far as moderately expensive equipment is considered, this can not be solved with such general but hardware-intensive and costly solutions like deep learning. In this paper we present yet another set of features which are relatively simple to calculate to be used in the orange skin detection and classification problem. In the algorithm a set of features based upon the classic run length encoding is applied. The features similar to those presented here have been successfully used in the problem of finding the handwriting area in a manuscript [19]. The accuracy of the classification expressed with its accuracy, can attain values over 98%, despite of the relatively simple structure of the features.

The remaining part of the paper is organized as follows. In the next section the images to be analysed will be described. In Section 3 the algorithm of finding the features will be proposed. The classifier will be briefly introduced and the attained classification quality will be presented in Section 4. The results will be discussed in Section 5. The paper will be concluded in Section 6.

2. Images

Orange skin is a defect of lacquered and hardened surfaces which can be seen as shallow hollows which form a characteristic pattern. The defect can emerge due to insufficient quantity or bad quality of dilutent, excessive temperature difference between the lacquer and the surface, bad pressure or distance of spraying, excessive air circulation during spraying or drying, and insufficient air humidity. The defect can be the reason for treating the surface as unacceptable for vending, or as acceptable as a lower quality product. Therefore, the surface fragments considered have been divided into three classes: *good*,

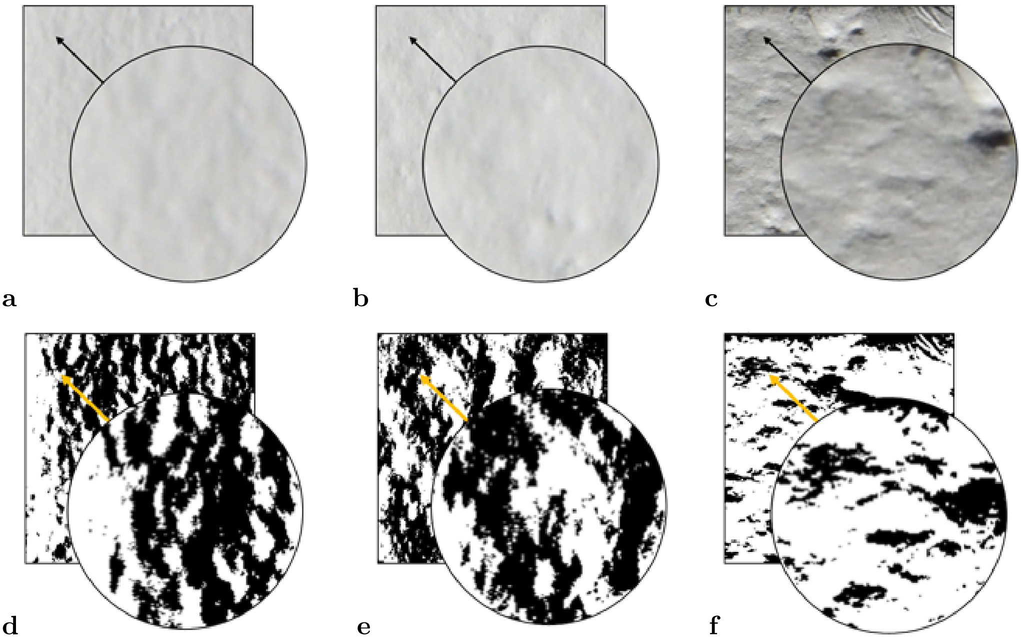

Fig. 1. Examples of images of the surface without and with the orange skin defect. (**a**) Class *good*, defect absent; (**b**) class *acceptable*, weak defect; (**c**) class *bad*, defective. Binarized images: (**d**) *good*; (**e**) *acceptable*; (**f**) *bad*.

acceptable and *bad*. The examples of such surfaces with fragments enlarged are shown in Figure 1a, b and c, respectively.

The set of images, described already in [11,21], contained images 300×300 pixels each, cut from a smaller number of images taken with the Nikon D750 24 Mpix camera with the Nikon lens F/2.8, 105 mm. The distance from the focal plane to the object surface was 1 m and the optical axis of the camera was normal to the surface. The surface was illuminated with a flash light located at 80 cm from the object, with the axis of the light beam inclined by 70° from the normal to the surface (lighting close to tangential). The images were made in colour mode, with lossless compression, transformed into CMY colour model, and the Y component was taken for further processing.

There were $N = 808$ images in total: 282 *good*, 278 *acceptable*, and 248 *bad* (92 images were excluded from the set considered in [21] due to that it was very problematic to which class they should be assigned). All these images were used in the assessment of the classification quality, due to that the cross-validation procedure was used, as it will be described further in Section 4.

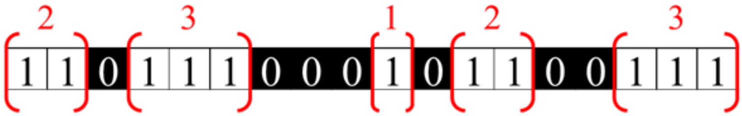

Fig. 2. Illustration of a run length code of a line of a binary image. Ones represent white pixels, zeros represent black ones.

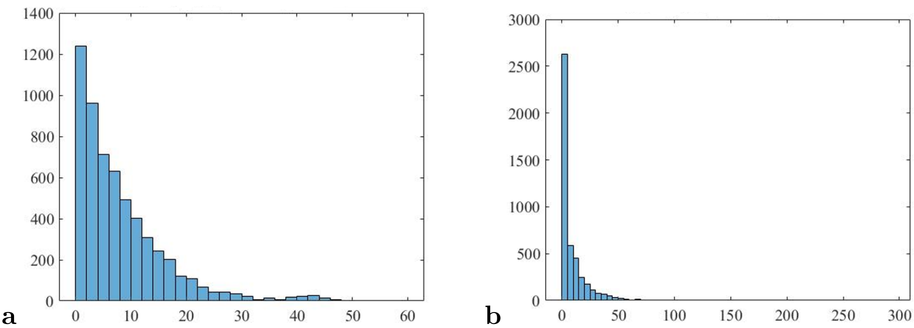

Fig. 3. Histograms or run lengths of ones in lines of a binary image of the orange skin defect. (**a**) Lines taken horizontally; (**b**) lines taken vertically. The histograms shown have bins containing more than one length of the runs (for example, histogram **a** has bins containing two consecutive lengths each: 1 and 2, 3 and 4, etc.); in the features, the histograms containing 10 consecutive lengths in each bin are used.

3. Features

The features used previously to successfully differentiate the regions with handwritten text from other regions [19] will be used. These were the components of the histogram of runs of ones, in run length encoded binarized image of the sheet of a manuscript. A hypothetic line of a binary image is shown in Fig. 2 in the form of the run length code. It occurs that the longer the run, the less frequent it is in an image. Histograms of run lengths of ones in an image coded by rows (horizontally) and by columns (vertically) are shown in Fig. 3.

In the histograms, each bin contains L consecutive lengths, and the lengths up to $l_{\max}$ are considered. To find the features, $L = 10$ lengths are counted in each bin, and lengths up to $l_{\max} = 60$ are considered, for horizontal as well as for vertical lines. This gives $n = l_{\max}/L = 6$ features for each direction, which makes $m = 2n = 12$ features for an image. The numbers of the runs are normalized to give the sum equal to one,

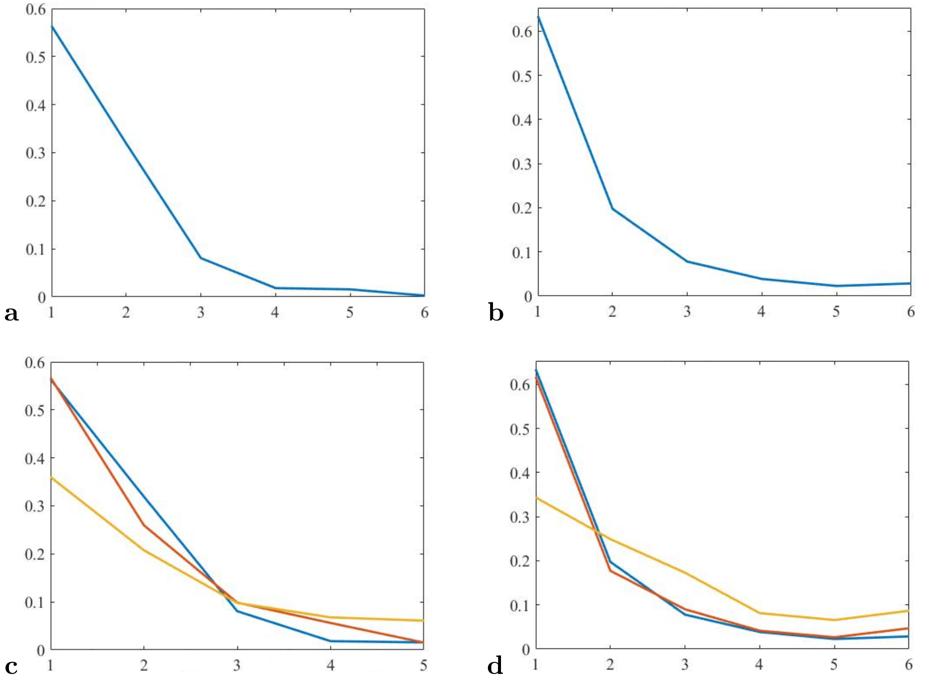

Fig. 4. Features, that is, normalized histograms for horizontally nad vertically coded images. (**a**) Example features of an image coded horizontally, corresponding to the raw histogram of Fig. 3a. (**a**) Example features of an image coded vertically, corresponding to the raw histogram of Fig. 3b. (**c**) Features of an image coded horizontally, belonging to three classes: blue – class *good*, red – class *acceptable*, and yellow – class *bad*. Zero values of feature 6 not shown. (**d**) Same as **c**, but the image coded vertically. Lines joining the values for the integer indexes of features have no meaning; shown only to indicate the relations between the points.

so the features are real numbers (rational fractions). Normalization makes the features insensitive to the size of an image, but they remain sensitive to its scale.

In Fig. 4 the features, that is, the normalized histograms for a horizontally and vertically encoded image, corresponding to the histograms in Fig. 3, and the features for the three classes of the surface are shown.

To binarize the images, the classical Otsu method [18] was used due to that this classic parameter-less method still has many citations and applications and its performance is good in cases where the numbers of dark and bright pixels do not differ much [20].

4. Classification

4.1. Classifier

The classical nearest neighbour classifier was used, like in [19]. The simplest one nearest neighbour (1-NN) version was applied. The more complex nearest neighbours schemes (see for example [8]) were not tested due to that in this paper the effectiveness and simplicity of the features, not the classifier, is of the greatest interest, and the results appeared to be satisfactory.

As the distance function between objects e_i and e_j represented with features $F(e) = \{f_k(e), k = 1, \ldots, m\}$ the Manhattan distance was used

$$d(f_i, f_j) = \sum_{k=1}^{m} |f_k(e_i) - f_k(e_j)| \; . \tag{1}$$

4.2. Errors and classification quality

4.2.1. Data without transformations

The quality of the classification in the setting defined above was tested with the cross-validation method in its extreme form, that is, the leave-one-out method. Besides some well theoretically justified criticism [12], this method is still treated as attractive and applicable (let us cite [4] as just one example of many possible ones). This method consists of taking one pattern away from the set of known N patterns, training the classifier with the remaining $N-1$ ones, and then classifying this left pattern with the so trained classifier. After cycling through all the N patterns in the set, the errors made in the N classifications are calculated. In the case of 1-NN classifier, there is actually no training, and in our case the set of features is fixed so the feature selection process is absent either. Therefore, this otherwise time-consuming procedure is effective in this case and is an attractive solution.

The errors found with the cross-validation method are shown in Table 1. As the global measure of classification quality, the accuracy expressed as the ratio of the number of

Tab. 1. Confusion matrix C_{ij} for the three classes: *good* $c = 1$, *acceptable* $c = 2$ and *bad* $c = 3$.

	c	Actual class		
Class no. $= c$		1	2	3
Classification result	1	275	2	0
	2	7	274	3
	3	0	2	245

proper classifications to the number of all the patterns was used

$$A = \frac{\sum_{i=1}^{3} C_{ii}}{\sum_{i,j=1}^{3} C_{ij}}\,, \tag{2}$$

with the attained value $A = 98.29\%$.

4.2.2. Testing the sensitivity to rotation

In an industrial quality inspection process the location of the tested object can be easily restricted to that which corresponds to the conditions in which the teaching of the classification system was performed. However, the sensitivity of the testing system to such transformations as rotation and scale should be assessed. From the two transformations mentioned, the scale seems to be easier to control by setting the distance from the focal plane of the camera to the object surface to the desired value. The influence of rotation, which can result from that the direction of cutting the manufactured object from the painted surface can be arbitrary, is more difficult to restrict, so the sensitivity of the proposed features to rotation has been tested.

To achieve this goal, each of the images used in the cross-validation process was rotated, by $k = 360$ angles from $1°$ to $359°$. The images should have the same dimensions and scale as those without rotation, so before the rotation each image was complemented at its edges and corners with its six mirror images, such extended image was rotated, and its center 300×300 was cut, as shown in Fig. 5. In this way, the appearance of the texture within the image was maintained.

For each of the rotated images, the cross-validation procedure was repeated, with the $N - 1$ reference images without rotation, and the classified image with rotation, by each of the k angles. The accuracy A obtained in this way, versus the rotation angle, is shown in the graph in Fig. 6. The difference between the best and worst accuracy is close to 3.5%. The accuracy is the worst (95.5%) for the angle close to $310° = 360° - 50°$, and is the best at $270°$ and happens to surpass 99%. The shape of the graph follows the commonsense expectation that the rotations by angles close the multiples of $90°$ deteriorate the classification quality to the smallest extent (or even improve it), while the rotations by $45°$ reduce it the most. The variability of inter-class errors is illustrated in Table 2 with the standard deviations of the errors.

5. Discussion

The classification accuracy obtained with the features presented has been measured with a different methodology than in the previous papers [11,21], where cross-validation with the set of images was partitioned into nine subsets, 90 images each. Also the set of images was better prepared in this paper than in [21], in that the images for which the class was

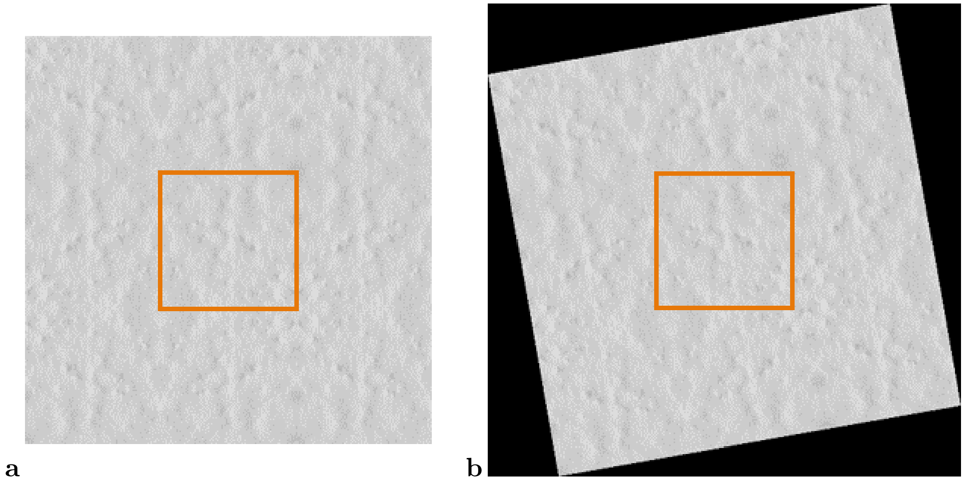

Fig. 5. Illustration of rotating an image with the scale and dimensions maintained. (**a**) Image 300×300 (colour frame) complemented at the edges and corners with its mirror images, forming a 900×900 extended image. (**b**) Window 300×300 (colour frame) cut from the rotated extended image.

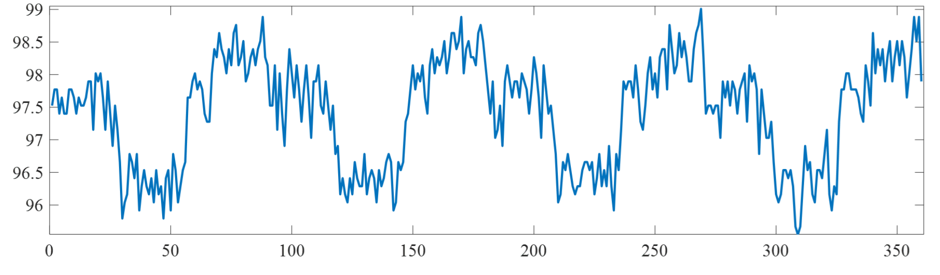

Fig. 6. Global classification accuracy [%] versus rotation of the classified image [°], in the range [1°, 359°].

Tab. 2. Variability of the confusion matrix for the experiment with rotation illustrated by standard deviations of the inter-class errors for three classes.

	c	Actual class		
Class no. = c		1	2	3
Classification result	1	2.21	1.05	0.72
	2	2.13	3.28	2.56
	3	0.44	2.95	2.88

difficult to assess by a human eye were excluded. Nevertheless, it can be observed that the single set of features, relatively simple to calculate, appeared to perform better, or at least not worse, than the large set of state-of-the-art features selected with advanced methods. The best average accuracy attained in [21] was 95.88%. The comparison with the accuracy of 98.29% found in this paper indicates that the features proposed have a potential in the application of our interest. It should be also noted that here there was entirely no confusion between the classes *good* and *bad*, which is important from the point of view of the application to quality inspection.

The sensitivity of the classification accuracy to the rotation of the image was not tested previously. The variability shown in Table 2 can be considered relatively small with respect to the global accuracy. From the graph shown in Fig. 6 it follows that the accuracy should not fall below 95.5% obtained for the rotation of the image close to 45°, which is the worst case if it is taken into account that the original features were found for angles close to zero. For the angles differing from those close to the multiple of 90° by not more than ±20° the accuracy is not worse than 97%. This result indicates that the decrease of classification accuracy related to small rotations is negligible, and that the sensitivity of the classification quality to arbitrary rotations can be considered small.

6. Conclusion

Quality of the surface of manufactured elements is usually inspected by a naked eye of human inspectors in the furniture industry practice. As a continuation of our previous research aimed at changing this practice, a set of textural features based upon thresholding and run length encoding, positively tested in other applications, have been successfully applied to the problem of classification of the quality of lacquered surfaces in furniture exhibiting the surface defect known as *orange skin*. The set of proposed features was formed by 12 easy to calculate real-valued features for each object. The classifier used was the simple one nearest neighbour classifier, with the Manhattan distance, and without feature selection. The classification method was tested on 808 images 300×300 pixels, made under controlled, close-to-tangential lighting, and classified by an expert in the way of inspection by an unarmed eye into three classes: *good*, *acceptable* and *bad* surface. The numbers of examples from each class were close to each other. The classification quality was assessed by cross-validation with the leave-one-out method. The attained accuracy was not smaller than 98% when the tested surface was not rotated with respect to the training samples, 97% for rotations up to ±20° and 95.5% in the worst case for arbitrary rotations. In the case without rotation, there were no classification errors between the classes *good* and *bad* surface.

The proposed features seem to be a simple and effective alternative to the advanced features tested previously for this defect.

References

[1] M. Allard, C. Jaecques, and I. Kauffer. Coating material which can be thermally cured and hardened by actinic radiation and use thereof, September 27 2005. US Patent 6,949,591.

[2] L. Armesto, J. Tornero, A. Herraez, and J. Asensio. Inspection system based on artificial vision for paint defects detection on cars bodies. In *2011 IEEE International Conference on Robotics and Automation*, pages 1–4, May 2011. doi:10.1109/ICRA.2011.5980570.

[3] V. Bucur. Techniques for high resolution imaging of wood structure: a review. *Measurement Science and Technology*, 14(12):R91, 2003. doi:10.1088/0957-0233/14/12/R01.

[4] G. C. Cawley and N. L. C. Talbot. Efficient leave-one-out cross-validation of kernel Fisher discriminant classifiers. *Pattern Recognition*, 36(11):2585 – 2592, 2003. doi:10.1016/S0031-3203(03)00136-5.

[5] L. J. Chmielewski, K. Laszewicz-Śmietańska, P. Mitas, A. Orłowski, et al. Defect detection in furniture elements with the Hough transform applied to 3D data. In R. Burduk et al., editors, *Proc. 9th Int. Conf. Computer Recognition Systems CORES 2015*, volume 403 of *Advances in Intelligent Systems and Computing*, pages 631–640, Wrocław, Poland, 25-27 May 2015. Springer. doi:10.1007/978-3-319-26227-7_59.

[6] L. J. Chmielewski, A. Orłowski, K. Śmietańska, J. Górski, et al. Detection of surface defects of type 'orange skin' in furniture elements with conventional image processing methods. In F. Huang and A. Sugimoto, editors, *Image and Video Technology – PSIVT 2015 Workshops*, volume 9555 of *Lecture Notes in Computer Science*, pages 26–37, Auckland, New Zealand, 23-27 Nov 2015. Springer, 2016. doi:10.1007/978-3-319-30285-0_3.

[7] L. J. Chmielewski, A. Orłowski, G. Wieczorek, K. Śmietańska, and J. Górski. Testing the limits of detection of the 'orange skin' defect in furniture elements with the HOG features. In N.T. Nguyen, S. Tojo, et al., editors, *Proc. 9th Asian Conference on Intelligent Information and Database Systems ACIIDS 2017, Part II*, volume 10192 of *Lecture Notes in Artificial Intelligence*, pages 276–286, Kanazawa, Japan, 3-5 Apr 2017. Springer. doi:10.1007/978-3-319-54430-4_27.

[8] A. Jóźwik, S. Serpico, and F. Roli. A parallel network of modified 1-NN and k-NN classifiers – Application to remote-sensing image classification. *Pattern Recogn. Lett.*, 19(1):57–62, January 1998. doi:10.1016/S0167-8655(97)00155-4.

[9] D. A. Karras. Improved defect detection using support vector machines and wavelet feature extraction based on vector quantization and SVD techniques. In *Proc. Int. Joint Conf. on Neural Networks, 2003*, volume 3, pages 2322–2327, July 2003. doi:10.1109/IJCNN.2003.1223774.

[10] J. Konieczny and G. Meyer. Computer rendering and visual detection of orange peel. *Journal of Coatings Technology and Research*, 9(3):297–307, 2012. doi:10.1007/s11998-011-9378-2.

[11] M. Kruk, B. Świderski, K. Śmietańska, J. Kurek, L. J. Chmielewski, J. Górski, and A. Orłowski. Detection of 'orange skin' type surface defects in furniture elements with the use of textural features. In K. Saeed, W. Homenda, and R. Chaki, editors, *Proc. 16th IFIP TC8 Int. Conf. Computer Information Systems and Industrial Management Applications CISIM 2017*, volume 10244 of *Lecture Notes in Computer Science*, pages 402–411, Białystok, Poland, 16-18 Jun 2017. Springer, Cham. doi:10.1007/978-3-319-59105-6_34.

[12] W. J. Krzanowski and D. J. Hand. Assessing error rate estimators: The leave-one-out method reconsidered. *Australian Journal of Statistics*, 39(1):35–46, 1997. doi:10.1111/j.1467-842X.1997.tb00521.x.

[13] K. Laszewicz and J. Górski. Control charts as a tool for the management of dimensional accuracy of mechanical wood processing (in Russian). *Annals of Warsaw University of Life Sciences-SGGW, Forestry and Wood Technology*, 65:88–92, 2008.

[14] K. Laszewicz, J. Górski, and J. Wilkowski. Long-term accuracy of MDF milling process–development of adaptive control system corresponding to progression of tool wear. *European Journal of Wood and Wood Products*, 71(3):383–385, 2013. doi:10.1007/s00107-013-0679-2.

[15] K. Laszewicz, J. Górski, J. Wilkowski, and P. Czarniak. Analysis of dimensional accuracy of milling process. *Wood Research*, 58(3):451–463, 2013.

[16] F. Longuetaud, F. Mothe, B. Kerautret, et al. Automatic knot detection and measurements from X-ray CT images of wood: A review and validation of an improved algorithm on softwood samples. *Computers and Electronics in Agriculture*, 85(0):77–89, 2012. doi:10.1016/j.compag.2012.03.013.

[17] E.-C. Musat, E.-A. Salca, F. Dinulica, et al. Evaluation of color variability of oak veneers for sorting. *BioResources*, 11(1):573–584, 2016. doi:10.15376/biores.11.1.573-584.

[18] M. Otsu. A threshold selection method from gray-level histograms. *IEEE Trans. Systems, Man and Cybern.*, 9(1):62–66, 1979. doi:10.1109/TSMC.1979.4310076.

[19] J. L. Pach. *Identification of the author of Latin manuscripts with the use of image processing methods*. Ph.d. thesis, Warsaw University of Technology, Faculty of Electronics and Information Technology, Warsaw, 2019.

[20] M. Sezgin and B. Sankur. Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic Imaging*, 13(1):146–165, 2004. doi:10.1117/1.1631315.

[21] B. Świderski, M. Kruk, G. Wieczorek, J. Kurek, K. Śmietańska, L. J. Chmielewski, J. Górski, and A. Orłowski. Feature selection for 'orange skin' type surface defect in furniture elements. In L. Rutkowski et al., editors, *Proc. Int. Conf. on Artificial Intelligence and Soft Computing ICAISC 2018*, volume 10842 of *Lecture Notes in Artificial Intelligence*, pages 81–91, Zakopane, Poland, 3-7 Jun 2018. doi:10.1007/978-3-319-91262-2_8.

Context-Based Segmentation of the Longissimus Muscle in Beef with a Deep Neural Network

Karol Talacha[1], Bartosz Świderski[2], Jarosław Kurek[2], Michał Kruk[2], Andrzej Półtorak[1], Leszek J. Chmielewski[2], Grzegorz Wieczorek[2], Izabella Antoniuk[2], Jakub Pach[2], Arkadiusz Orłowski[2]

[1]*Institute of Human Nutrition Sciences*

[2]*Institute of Information Technology*

Warsaw University of Life Sciences – SGGW, Poland

Abstract. The problem of segmenting the cross-section through the *longissimus* muscle in beef carcasses with computer vision methods was investigated. The available data were 111 images of cross-sections coming from 28 cows (typically four images per cow). Training data were the pixels of the muscles, marked manually. The AlexNet deep convolutional neural network was used as the classifier, and single pixels were the classified objects. Each pixel was presented to the network together with its small circular neighbourhood, and with its context represented by the further neighbourhood, darkened by halving the image intensity. The average classification accuracy was 96%. The accuracy without darkening the context was found to be smaller, with a small but statistically significant difference. The segmentation of the *longissimus* muscle is the introductory stage for the next steps of assessing the quality of beef for the alimentary purposes.

Key words: beef carcasses, context-based, segmentation, longissimus muscle, classification, deep convolutional network, beef quality.

1. Introduction

Quality of food is one of the important concerns in contemporary living. Quantities in which meat is produced makes the ocular assessment of various features of meat too time-consuming and prone to subjectivity and errors resulting from fatigue. Computer vision methods are introduced to the meat production industry and research on accurate and effective methods is necessary.

In this paper we shall study the case of segmenting the cross-section of the *longissimus* muscle (called also *longissimus dorsi*). We will consider only the initially prepared transverse sections received during the partition of beef carcasses in which this muscle is the largest connected region, so we shall not go into the anatomic considerations. The knowledge on the location of the *longissimus* will be represented by the images annotated by a human expert.

The segmentation of the images of cross-sections through beef into the region belonging to the muscle of our interest and the other regions is the introductory stage of the analysis aimed at estimating the marbling of this muscle, which is an important factor influencing the quality of meet as an alimentary product. The marbling itself is out of scope of the present paper.

As the classifier we shall use the convolutional neural network (CNN) AlexNet proposed by Alex Krizhevsky et al. in 2017 [9]. Since then this network proved to be successful in many applications.

Survey on the use of computer vision in the meat quality evaluation can be found in [13] from 2004 and in a more recent paper [12] from 2019.

The following papers treat the question of analysis of the images of beef. Jackman, Sun and Allen [7] investigated the *longissimus dorsi* muscle in beef and its marbling and presented a customary system based on clustering and thresholding with contrast enhancement. They have presented very good results of the segmentation in comparison to a method requiring manual intervention of an expert. Agustin and Dijaya [1] recognized the quality and freshness of beef to enable a fast decision to be made at the butcher's. They have shown that a k-nearest neighbour classifier with the colour and textural features is able to recognize the type of meat. Andaya et al. [3] went beyond the muscle segmentation and assessed the meat marbling with a fuzzy logic-based classifier. They have investigated beef as one of the meats.

The research on other kinds of meet, fresh as well as cooked, was the subject of a number of papers. De Guzman et al. [6] assessed pork quality with the support vector machine. Quality of chicken meat was tested with color camera by Barbin et al [4] and it was found theat the results similar to those found with a colorimeter can be found. Color is a very important source of information in meat. Al-Sarayreh, Klette et al. [2] presented interesting results in testing red meat adulteration with hyperspectral imaging. Such imaging was also used by Kamruzzaman et al. [8] to monitor the colour of red meet online. Sun, Young, Liu et al. [11] also studied the relation between the colorimeter results and the colour computer vision results for pork meat.

There are also numerous papers on the recognition of other kinds of food with the image processing methods. For example, Du, Iqbal and Sun [5] analysed the quality of various cooked food.

Finally, convolutional neural networks are used in the domain of meat image analysis. Muñoz, Gou and Fulladosa [10] investigated the contents of intramuscular fat in dry ham slices. Among other things, they pointed to the question of manual labelling the training data which can be problematic.

To our best knowledge, until now AlexNet has not been used to the segmentation of beef, including the *longissimus dorsi*, although many other methods were used in the application to processing the images of meet in general, and beef in particular.

The remaining part of this paper will be organized as follows. In the next Section we shall present the images we have used for this research. In Section 3 the training and classification will be discussed together. Configuration of the network will be shown in

Subsection 3.1. More attention will be paid in Subsection 3.2 to our simple but effective proposition of using the context of a pixel. The accuracy of the results found with the context-based method will be reported in Subsection 3.3, and that received with the typical neighbourhood-based method – in Subsection 3.4. The results will be briefly discussed in Section 4 and the paper will be concluded in Section 5.

2. Images

In the study the images of sections through the body of the cow, with the cross-section through the *musculus longissimus* were analysed. From the images of cross-sections made under various ribs, the subject of our interest were those under ribs 5 and 6, from both sides, which makes four images per cow. There were 28 cows giving 112 images, one image was missing for one cow, so there were 111 images in total for the analysis. Example images from the cow no. 3 are shown in Figs. 1a, b, e and f.

In these images, an expert marked manually the regions belonging to the cross-section through the em musculus longissimus, shown in Figs. 1c, d, g and h. Such marked images, together with their originals, served as data for the training, classification and testing aimed at the segmentation of the images into the two classes.

3. Training and classification

3.1. Configuration of the network

To show the configuration of AlexNet in a detailed and univocal way, the contents of the configuration file is directly presented in Listing 1. The line numbers are shown explicitly. For presentation only, the long lines of text are shown broken, with the continuations deeply indented; in the original file they were long, unbroken lines.

3.2. Presentation of the images and the context

In AlexNet, as the objects for training as well as for classification, the image fragments are shown. At the training stage each fragment is labelled with the class index (muscle: 1, other region: 0). In the classification phase, the image fragment is given at the input, and the index of the class is returned by the network at the output.

For the presentation of the images to the network the concept of context-based recognition was used. We thought of applying something meaningful but simple, going beyond the presentation of a pixel together with its small neighbourhood. As a solution, it was proposed to present the pixel together with the contents of the whole image surrounding it, in which the immediate circular neighbourhood was presented directly, and the remaining part of the image was darkened. In this way, the pixel was shown together with

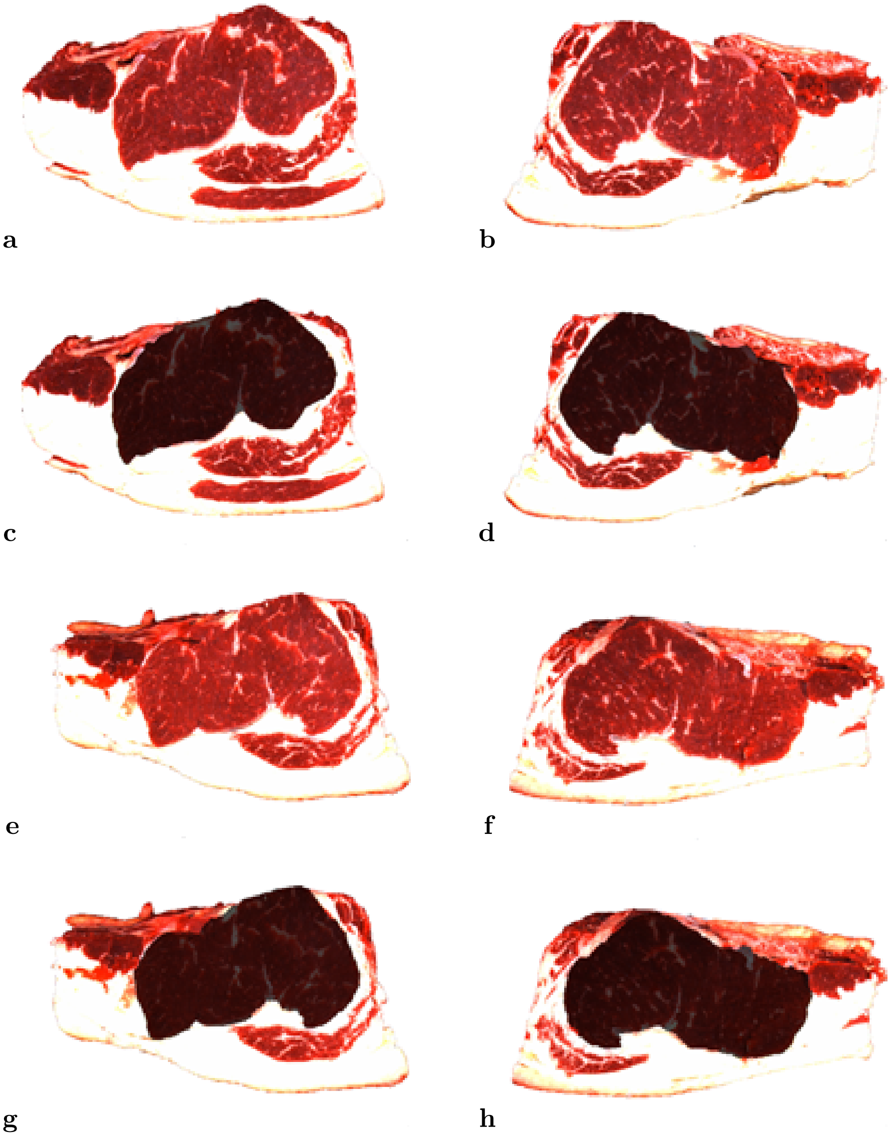

Fig. 1. Images with cross-sections through the *musculus longissimus* and surrounding tissues. **a**, **b**, **e**, **f**: original images; **c**, **d**, **g**, **h**: images as above, with the region of *musculus longissimus* marked in the way of darkening the respective pixels.

```
  1   'data'     Image Input                    227x227x3 images with
                                                   'zerocenter' normalization
  2   'conv1'    Convolution                    96 11x11x3 convolutions with
                                                   stride [4  4] and padding [0
                                                   0]
  3   'relu1'    ReLU                           ReLU
  4   'norm1'    Cross Channel Normalization    cross channel normalization with
                                                   5 channels per element
  5   'pool1'    Max Pooling                    3x3 max pooling with stride [2
                                                   2] and padding [0  0]
  6   'conv2'    Convolution                    256 5x5x48 convolutions with
                                                   stride [1  1] and padding [2
                                                   2]
  7   'relu2'    ReLU                           ReLU
  8   'norm2'    Cross Channel Normalization    cross channel normalization with
                                                   5 channels per element
  9   'pool2'    Max Pooling                    3x3 max pooling with stride [2
                                                   2] and padding [0  0]
 10   'conv3'    Convolution                    384 3x3x256 convolutions with
                                                  stride [1  1] and padding [1  1]
 11   'relu3'    ReLU                           ReLU
 12   'conv4'    Convolution                    384 3x3x192 convolutions with
                                                  stride [1  1] and padding [1  1]
 13   'relu4'    ReLU                           ReLU
 14   'conv5'    Convolution                    256 3x3x192 convolutions with
                                                  stride [1  1] and padding [1  1]
 15   'relu5'    ReLU                           ReLU
 16   'pool5'    Max Pooling                    3x3 max pooling with stride [2
                                                  2] and padding [0  0]
 17   'fc6'      Fully Connected                4096 fully connected layer
 18   'relu6'    ReLU                           ReLU
 19   'drop6'    Dropout                        50% dropout
 20   'fc7'      Fully Connected                4096 fully connected layer
 21   'relu7'    ReLU                           ReLU
 22   'drop7'    Dropout                        50% dropout
 23   'fc8'      Fully Connected                1000 fully connected layer
 24   'prob'     Softmax                        softmax
 25   'output'   Classification Output          crossentropyex with 'tench',
                                                  'goldfish', and 998 other
                                                  classes - two classes (pixel
                                                  belonging / not belonging to
                                                  musculus longissimus)
```

Listing 1. Parameter file for AlexNet. Broken lines shown here as deeply indented.

its immediate neighbours, and the broader context. In detail, the following operations were done for each trained as well as classified pixel.

1. Image was centred at the given pixel.
2. Image around the pixel was trimmed to 227×227 pixels. If the pixel was too close to the image boundary to do this, the missing part was padded with white pixels.
3. Pixels outside the small circular region centred at the pixel, with the area of approx. 1% of the trimmed image area (radius ≈13 pixels), were darkened by reducing the image intensity by a half.

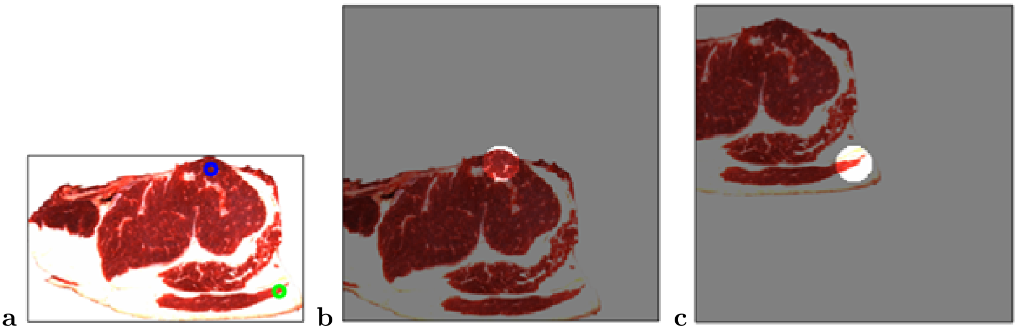

Fig. 2. Image (of Fig. 1a) with two pixels prepared for training or classification. **a**: image with two interesting pixels marked with colours; **b**, **c**: images presented to the network for pixel marked with blue and green, respectively. Grey frames indicate the borders of the pixels.

The examples of these operations are shown in Fig. 2. For the training, the image of Fig. 1b was presented to the network together with the label 1, and the one of Fig. 1c – with the label 0, as an example.

For comparison, the same was done without darkening the context of the pixels. This will be referred to further.

3.3. Classification accuracy

The assessment of the classification accuracy was performed with the cross-validation method. Images from all the cows, except one cow left out for testing, were used to train the classifier, and the results were checked on the images belonging to that testing cow. Such a procedure was repeated with until each cow was the testing one.

The training was a time-consuming task, so for the training phase only a part of pixels of the training images was drawn at random. There were two versions of training: with drawing 3000 pixels for training in each training image, and with drawing 100 pixels for training in each image. In both cases, the testing was performed on all pixels of all images of the testing cow.

The classification quality was measured with the measure of accuracy:

$$A = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}} , \tag{1}$$

where TP, FP, TN and FN are the numbers of true positive, false positive, true negative and false negative classifications of the pixels.

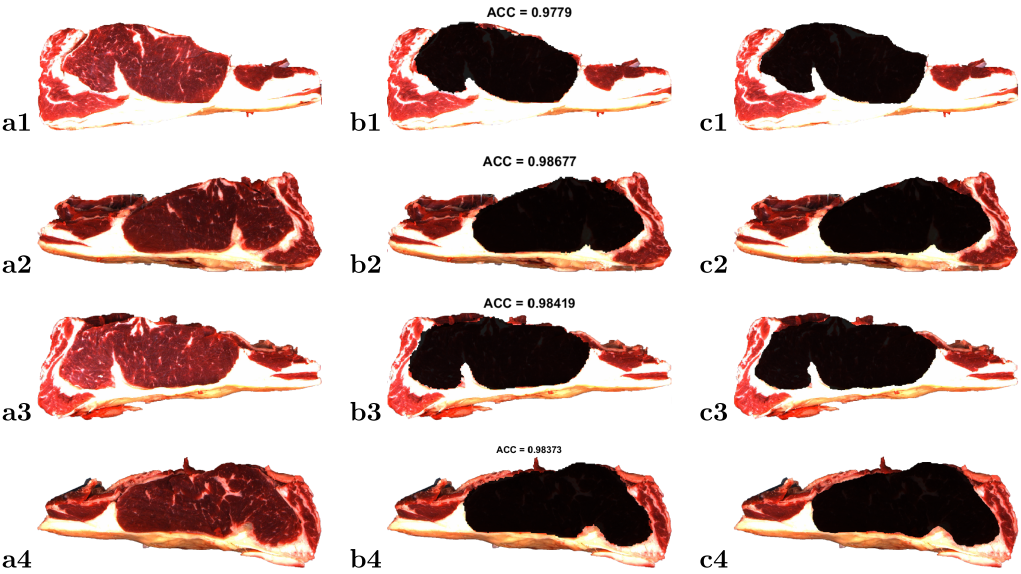

Fig. 3. Examples of results and accuracies found with 3000 training pixels drawn from each image, in a single validation round with just one testing cow (cow no. 1). (**a**) raw image; (**b**) classified image; (**c**) ground truth – training image; (**x1-x4**) subsequent images for this cow.

The example results and accuracies found with 3000 pixels drawn for training from each cow, in a single validation round with just one testing cow, are shown in Fig. 3.

The experiment with all the cows and with drawing 100 pixels from each image for training was performed. To find the overall accuracy, the accuracies were found for each image for each testing cow separately, and averaged for the images. This overall accuracy was found to be 96.27% for 111 images (minimum 93.04%, maximum 98.37%). Selected results are shown in Figs. 4 and 5.

3.4. Accuracy without darkening the context

The same experiment was repeated without darkening the context of the pixels presented to the network. In this case, 100 pixels were drawn from each image for training, as above. The overall accuracy was found to be 95.26% (minimum 91.00%, maximum 97.43%). In 88/111 cases, that is 79%, the results with the darkened context were better than those without darkening. The difference was found to be statistically significant at the p-value much lower than 0.01, which was tested with the *sign test* and *signrank* tests.

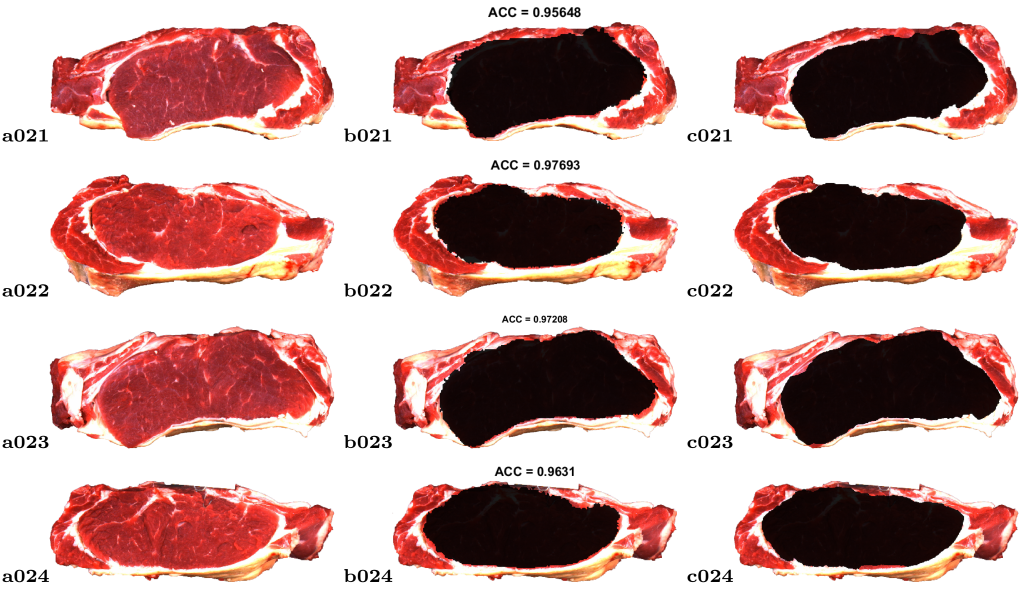

Fig. 4. Examples of results and accuracies found with cross-validation, with 100 training pixels drawn per image, for cow no. 2. (**a**) raw image; (**b**) classified image; (**c**) ground truth – training image; (**xiij**) image for cow *ii*, image *j*.

4. Discussion

The classification method proposed appeared to perform with the average accuracy 96% which can be considered as a promising result. This result was achieved for 100 training pixels per image. Such a small number was a compromise between accuracy and training speed. The accuracy with more training pixels could be larger, which has been confirmed with the experiment with 3000 pixels per image and with a single cross-validation round (one testing cow and 27 training ones), in which the average accuracy was 98.4%.

The speed of classification was limited with the necessity of preparing the neighbourhoods and context for each classified pixel. These operations are simple image shifting and convolutions, and could be performed much more efficiently if dedicated software, or better, dedicated hardware were used.

For the difference in the classification accuracies between the versions of the presentation of pixels to the classifier, with darkened context and with unchanged neighbourhood, it has been demonstrated that the version with darkened context as descried in Section 3.2 was significantly better than that with showing the pixel simply together with its large neighbourhood.

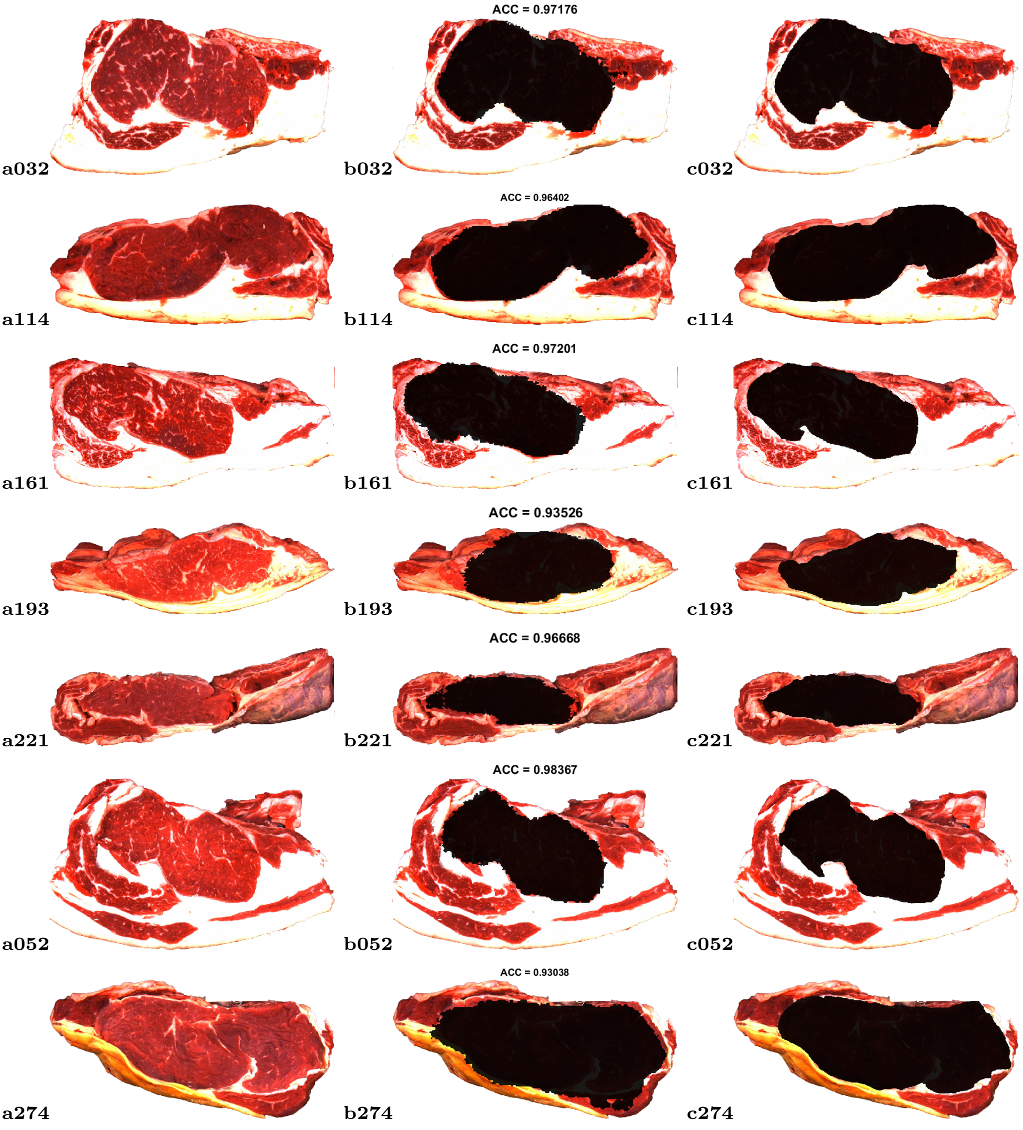

Fig. 5. Examples of results and accuracies found with cross-validation, with 100 training pixels per image, for selected cows and their images. (**a**) raw image; (**b**) classified image; (**c**) ground truth – training image; (**xiij**) image for cow *ii*, image *j*. The last two examples are the extremal cases of accuracy: maximum 0.98367 for cow 5 and minimum 0.93038 for cow 27.

5. Conclusion

To assess the quality of beef carcasses with such measures like, for example, marbling, it is first necessary to find the region occupied by the *longissimus* muscle, which is important from the point of view of cookery. The image segmentation problem stated in this way was the problem of our interest.

As the classifier, the AlexNet deep convolutional neural network was used. As the classified objects, pixels of the images of the cross-sections were considered. The data analysed consisted of 111 images of cross-sections, taken from 28 cows (four images per cow, one image missing). Training data were available for these images in the form of images with pixels belonging to the muscle sought marked manually by human experts. It was proposed to present to the network each pixel together with its small circular neighbourhood, and also together with its context represented by the broader neighbourhood, that is, the further neighbourhood, darkened by reducing the image intensity by a half. The average classification accuracy received with the cross-validation method was at the level of 96% and could be improved to around 98% by performing the training with more data. The accuracy without darkening the context was found to be smaller (95%). The difference was small but statistically significant (p-value < 0.01, for *sign test* and *sigrank* tests).

The proposed methodology, including the context-based classification and application of the deep convolutional network, seems to be a promising possibility in the domain of segmentation of meet images.

References

[1] S. Agustin and R. Dijaya. Beef image classification using k-nearest neighbor algorithm for identification quality and freshness. *Journal of Physics: Conference Series*, 1179:012184, July 2019. doi:10.1088/1742-6596/1179/1/012184.

[2] M. Al-Sarayreh, M. M. Reis, W. Q. Yan, and R. Klette. Detection of red-meat adulteration by deep spectral-spatial features in hyperspectral images. *Journal of Imaging*, 4(5), 2018. doi:10.3390/jimaging4050063.

[3] A. E. Andaya, E. R. Arboleda, A. A. Andilab, and R. M. Dellosa. Meat marbling scoring using image processing with fuzzy logic based classifier. *Int. J. of Scientific & Technology Research*, 8(8):1442–1445, Feb. 2018. `http://www.ijstr.org/research-paper-publishing.php?month=aug2019`.

[4] D. F. Barbin, S. M. Mastelini, S. Barbon, G. F. C. Campos, A. P. A. C. Barbon, and M. Shimokomaki. Digital image analyses as an alternative tool for chicken quality assessment. *Biosystems Engineering*, 144:85–93, 2016. doi:10.1016/j.biosystemseng.2016.01.015.

[5] C.-J. Du, A. Iqbal, and D.-W. Sun. Quality measurement of cooked meats. In D.-W. Sun, editor, *Computer Vision Technology for Food Quality Evaluation (Second Edition)*, chapter 8, pages 195–212. Academic Press, San Diego, second edition, 2016. doi:10.1016/B978-0-12-802232-0.00008-6.

[6] R. J. S. De Guzman, D. N. N. Niro, and A. C. F. Bueno. Pork quality assessment through image segmentation and support vector machine implementation. *Journal of Technology Management and*

Business, 5(2), Feb. 2018. `https://publisher.uthm.edu.my/ojs/index.php/jtmb/article/view/2261`.

[7] P. Jackman, D.-W. Sun, and P. Allen. Automatic segmentation of beef longissimus dorsi muscle and marbling by an adaptable algorithm. *Meat Science*, 83(2):187–194, 2009. doi:10.1016/j.meatsci.2009.03.010.

[8] M. Kamruzzaman, Y. Makino, and S. Oshita. Online monitoring of red meat color using hyperspectral imaging. *Meat Science*, 116:110–117, 2016. doi:10.1016/j.meatsci.2016.02.004.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017. doi:10.1145/3065386.

[10] I. Muñoz, P. Gou, and E. Fulladosa. Computer image analysis for intramuscular fat segmentation in dry-cured ham slices using convolutional neural networks. *Food Control*, 106:106693, 2019. doi:doi.org/10.1016/j.foodcont.2019.06.019.

[11] X. Sun, J. Young, J. H. Liu, L Bachmeier, R. M. Somers, K. J. Chen, and D. Newman. Prediction of pork color attributes using computer vision system. *Meat Science*, 113:62–64, 2016. doi:10.1016/j.meatsci.2015.11.009.

[12] A. Taheri-Garavand, S. Fatahi, M. Omid, and Y. Makino. Meat quality evaluation based on computer vision technique: A review. *Meat Science*, 156:183–195, 2019. doi:10.1016/j.meatsci.2019.06.002.

[13] J. Tan. Meat quality evaluation by computer vision. *Journal of Food Engineering*, 61(1):27–35, 2004. doi:10.1016/S0260-8774(03)00185-7.

Constraint-Based Algorithm to Estimate the Line of a Milling Edge

Marcin Bator[1] and Katarzyna Śmietańska[2]
[1] *Institute of Information Technology*
[2] *Institute of Wood Sciences and Furniture Manufacturing*
Warsaw University of Life Sciences – SGGW, Poland

Abstract. Each practical task has its constraints. They limit the number of potential solutions. Incorporation of the constraints into the structure of an algorithm makes it possible to speed up computations by reducing the search space and excluding the wrong results. However, such an algorithm needs to be designed for one task only, has a limited usefulness to tasks which have the same set of constrains. Therefore, sometimes is limited to just a single application for which it has been designed, and is difficult to generalise. An algorithm to estimate the straight line representing a milling edge is presented. The algorithm was designed for the measurement purposes and meets the requirements related to precision.

Key words: constraint-based algorithm, line, milling, measurement.

1. Introduction

Image processing becomes increasingly popular in the area of quality inspection of furniture elements. It is applied in surface monitoring [4], wood defects detection [5] or tools monitoring [3, 6], among others.

One of intermediate wood products is a medium density fibreboard (MDF) which is a wood composite. The melamine-faced MDF is a single layer MDF covered with melamine. It is a commonly used material in furniture manufacturing [1].

Any imperfection in a surface, created by drilling or milling, reduces the value of a final product. One of the reasons of imperfections like delamination is the wear of tools used in machining the final product. To investigate the relation between the tool wear and the delamination, a delamination factor (explained for example in [7]) is usually analysed (see e.g. [6]).

A delamination factor can be calculated, for example, using the depth of chips or the area of chips as the measure of imperfection intensity. In massive analysed, the measurement of such coefficients without image processing would be labour-intensive, expensive, and subject to the risk of subjective judgement. To have repeatable and precise results, the task of measurement of the area and depth of chips should be performed with the use of an image analysis method.

The task of segmenting out the region which represents the chips on a milling edge is not an easy one. This region has, more or less, the same colour as the groove created by milling. It was observed that one of the ways to distinguish between these regions is the analysis of the spatial relationships. Pixels representing the chips are between the milling edge and the untouched melamine. It should be taken into account that on the samples produced with new mills there can be no chips at all.

An algorithmic approach suggests to find what is invariant. In the analysed case, in all samples, the groove and the melamine is always visible. This makes it possible to reduce the problem of indicating the pixels which represent the chips to finding the pixels between the line representing the edge of the milling and the melamine. If there is no delamination, there is nothing between these objects and the delamination factor is equal to zero. When the delamination appears, the border line of the untouched melamine is no longer straight and it cannot be treated as the milling edge.

The object to be estimated is the line representing the milling edge. A common approach to estimating a straight line in a picture is the Hough transform [2]. It is a robust method based on voting of pixels potentially belonging to the line to be detected. In the case of our interest it is not easy to apply that method. Typically the edge detection filter is applied and then pixels having a value above a chosen threshold are used to vote. However, because the difference in brightness between melamine and the remaining part of the image is quite significant, this would generate the set of voting pixels from those on the border of melamine. In consequence, the estimated line would be shifted. Although there is a shadow related to the depth of the groove, it is subtle and not regular enough to be treated as invariant.

To handle the problems mentioned above, in Section 2 the input data and the examples of important aspects related to the difficulties in the described problem will be analysed. In Section 3 a simple and fast algorithm which handles all of the mentioned problems will be presented. At the end, some of the results of application of the algorithm to the data will be shown.

2. Material

In the analysis, the pictures of melamine-faced MDF were used. It is important that we have considered the delamination of various intensities, including the large intensities never occurring in the industrial practice. This was done in order to test the proposed delamination factor measurement method in typical as well as in extreme working conditions.

There were 450 samples of MDF pieces 10×16 cm. On each sample there were three milled edges (one groove with two edges and one single-sided groove, both 5 mm deep). The details interesting from the wood processing point of view have been described in [6]. The samples were scanned with 600 dpi. Examples of scanned samples are presented in Fig. 1. From each scan, three pictures were taken and rotated if necessary. In this way,

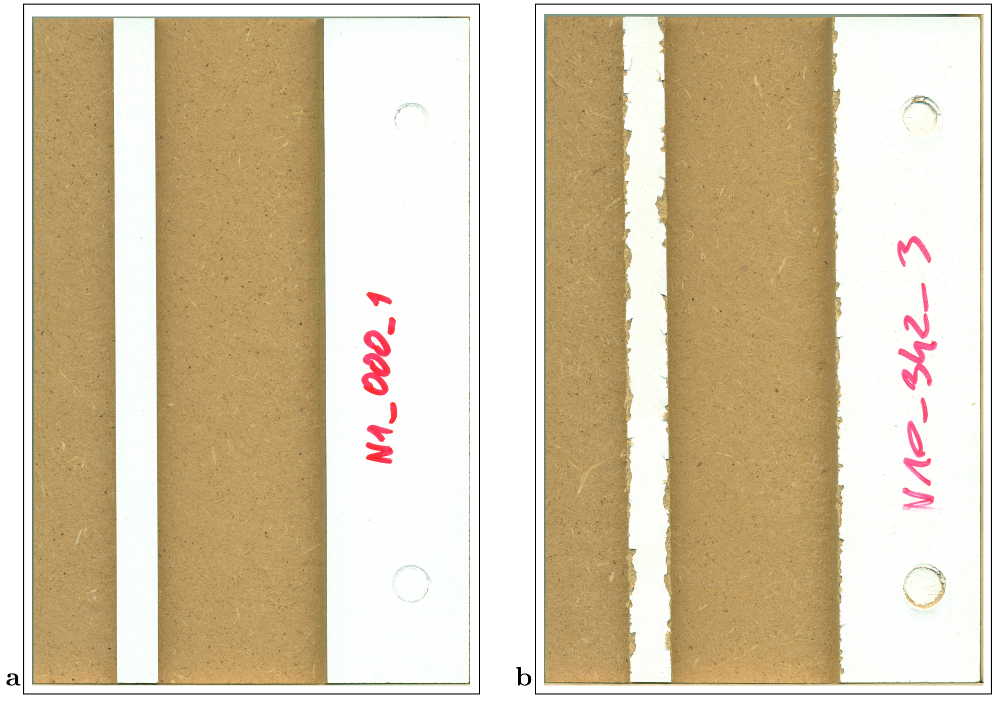

Fig. 1. Examples of scanned samples: (**a**) milled with a new tool; (**b**) milled with a very used-up tool.

1350 images 3840×240 pixels were produced for analysis. Each image was rotated in such a way that the melamine was at the same side (in the visualisation in Fig. 1 it is at the right-hand side). The coordinate y increased from the side of the groove towards the melamine side (to the right in Fig. 1). The general layout of the cutting edges was the same in each image; hence, instead of performing three times the detection of edges at the sides of a groove, it was necessary to perform three times the task of detecting a well-defined edge.

In Fig. 2 the examples of difficulties encountered in the measurements are shown in magnified images. The first problem is that the milling line is not horizontal so its position changes in the y direction (Fig. 2a, b), so the line itself should be detected. The second problem is that if delamination appears, the edge between melamine and darker regions is not straight, although the milling line was so. This is the reason why the edge cannot be analysed locally.

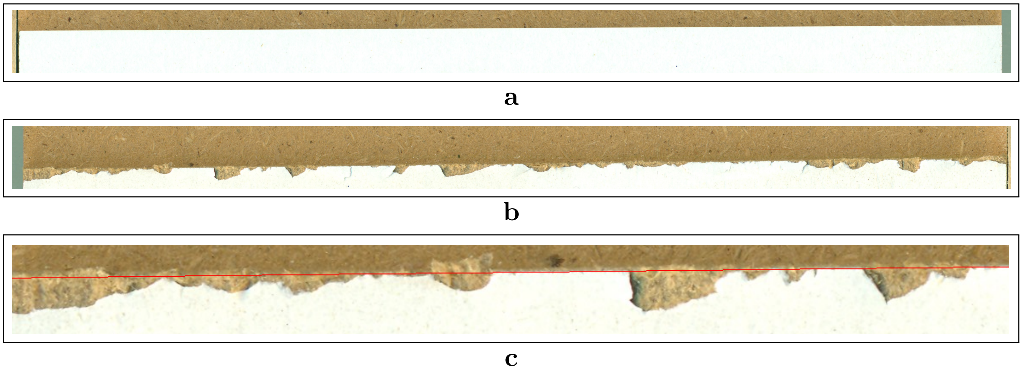

Fig. 2. Examples of difficulties to solve: (**a**, **b**) the milling line is not horizontal; (**c**) the delamination makes the edge heavily depart from being straight.

3. Algorithm

Constrains generally have two aspects. The first aspect is that constrains limit the freedom, the set of possibilities. The second aspect is that they allow to reject wrong hypotheses. By using the constraints it is possible to design effective algorithms. The computations can be speeded up and better results can be obtained. An example of an algorithm to detect trees in LIDAR data in which constraints were taken into account can be found in [8].

In the case of our interest an algorithm to estimate the straight line which represents the milling edge was needed. The result of the algorithm had to be precise enough to be used for the measurement purpose. As long as milling removes some of the melamine, the constraint was defined such that pixels representing melamine cannot occur inside the area of a groove. At the same time, the pixels representing melamine should belong to the edge except the cases where delamination occurred. An assumption was made that at least two pixels representing melamine were preserved at the edge. Even in the worst cases of delamination, the number of such pixels is much larger.

In other words, the algorithm had to find a straight line that *touches* as many of the pixels representing melamine as possible, but containing no pixels from the other side of the line, where the darker material of the inside part of the specimen is visible.

3.1. Preprocessing

Under the assumptions given above, it was enough to take for the further analysis only the pixels representing the boundary of the melamine area. It is assumed that the boundary extends over the major part of the image (at least there are pixels of the cutting line in both halves of the image). It is assumed that the cutting line is close to horizontal (very roughly) so it can be parameterized with the variable x, with the axis Ox pointing to the right, and the axis Oy points down (melamine is in the lower part of the image, like in Fig. 2). To prepare the required information, the preprocessing steps described below were executed.

1. Recognize the pixels representing melamine by selecting pixels brighter than a chosen threshold (200 in all RGB channels) and create a binary image.
2. For each x, take the minimum value of y representing melamine.
3. If the value of y is 0 then this is the margin (the region of interest including the edge was taken with a margin to compensate for an imprecise placement of the sample on the scanner); such points will be further excluded from the analysis.

In this way, an array Y_x of numbers was formed in which indexes represent subsequent values of x (in pixels) and values represent the minimum value of y coordinate of melamine pixels for current x.

3.2. Edge detection

The algorithm is described below.

1. Find x_0 that has the minimum value of y.
2. Check whether x_0 is in the left or in the right half of the image. The steps below describe the case when this pixel is on the right (if it is on the left then the algorithm below goes in the opposite direction, that is, from larger to smaller values of x).
3. Set x_1 as the first pixel on the left in the axis Ox (the smallest x possible).
4. Move x_1 to the right until the first local minimum of Y_x is found.
5. Use the equation of the line passing through two points $(x_0, Y_{x_0}), (x_1, Y_{x_1})$ to calculate its coefficients. This line represents the current position of the cutting line.
6. Verify if any point in Y_x violates the constrains (if for any x_i it is $Y_{x_i} < y$, where $y = a(x_i) + b$, and a and b are the coefficients calculated in step 5). Check this condition going in the same direction as previously (here, to the right). If a violation occurs, repeat from step 4. Otherwise, proceed.
7. Return the coefficients.

In the verification in step 6 it is taken into account that pixels have integer coordinates and the line equation gives real values.

The iterations of the algorithm are illustrated in Fig. 3 (to make it possible to enlarge the images to the maximum extent they have been turned left by a quarter of a revolution, so now Ox points up and Oy points to the right). The current position of the line is marked in red. The two points are marked with black arrows, with x_0 in a constant position and x_1 moving towards larger values of x from one iteration to another. The point in which the assumptions are violated is marked with a blue arrow. The line is not widened for the illustration due to that this would obscure the view of single pixels, which are very important in this algorithm.

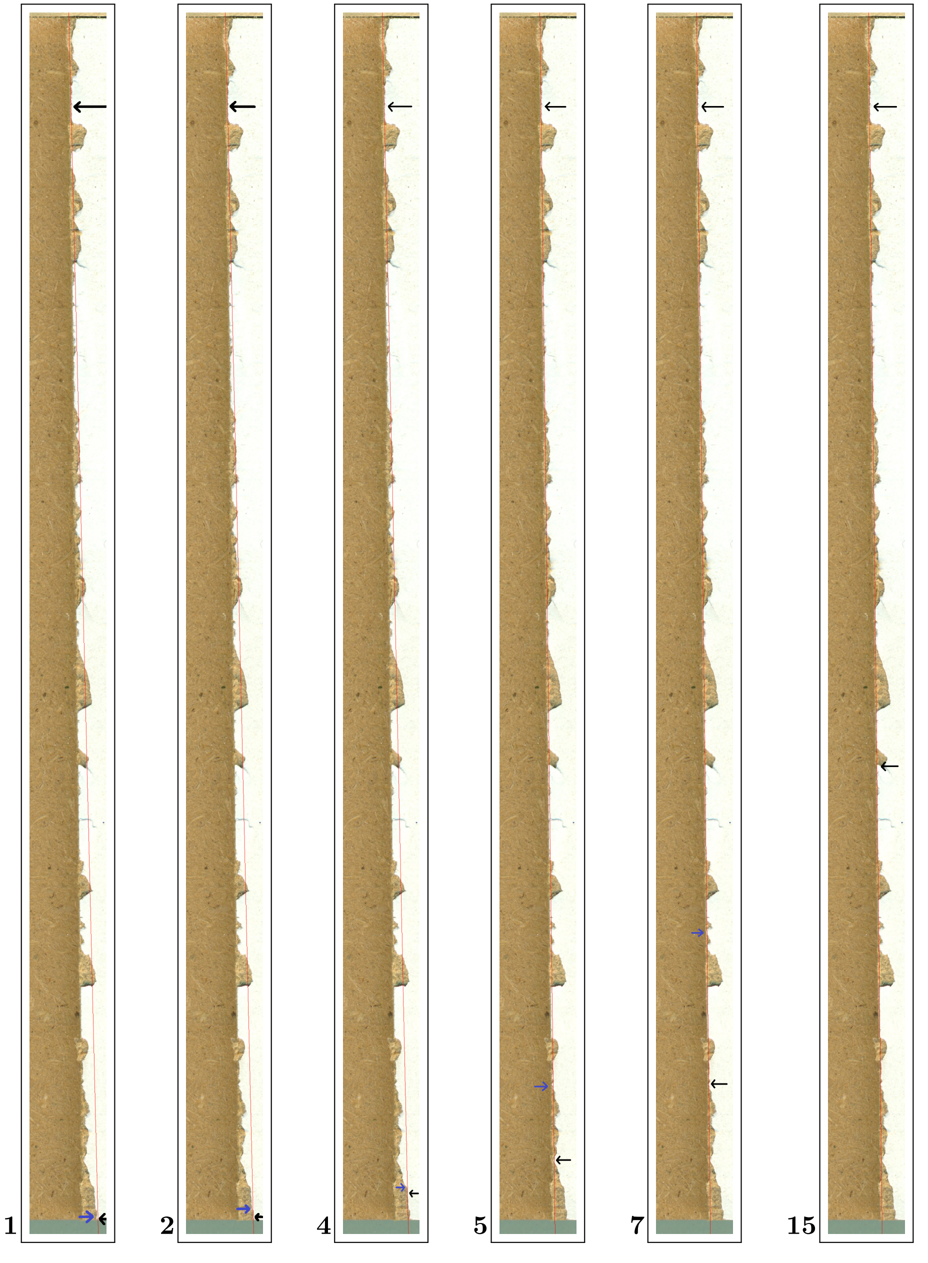

Fig. 3. Selected iterations of the algorithm: 1, 2, 4, 7, 15 (the last one). See also text.

4. Results

The algorithm was tested on a set of 450 samples, three images for each, making 1350 images (these images were used in the analysis of the delamination factor described in [6]). The correctness of the results was verified by inspection of the images with marked pixels representing the boundary of the melamine region and the line representing the milling edge. The results in all 1350 images were assessed as well marked. Some of the assessed results can be seen in Fig. 4.

5. Conclusion

Constraint-based algorithms usually have the advantages of accuracy and speed. The algorithm proposed in this paper fits this scheme. It is fast, has a simple structure, and returns correct results. All of those objectives were obtained as a consequence of the proper analysis of the problem and appropriate definition of constrains. At the same time, a constraint-based algorithm usually has a limited area of applicability and it is difficult to be generalized. The algorithm proposed was designed to solve the well-defined task on a large set of similarly structured, but varied images. It gave proper results for all the images in that set, which made it possible to effectively solve the technical problem posed by a practical application.

References

[1] T. Drouet, A. Leśnikowski, W. Ońisko and A. Starecki. *Technologia Tworzyw Drzewnych*. Wydawnictwa Szkolne i Pedagogiczne, Warszawa, 1980.

[2] J. Illingworth and J. Kittler. A survey of the Hough transform. *CVG&IP*, 44(1):87–116, 1988. doi:10.1016/S0734-189X(88)80033-1.

[3] A. Jegorowa, J. Górski, J. Kurek and M. Kruk. Initial study on the use of support vector machine (SVM) in tool condition monitoring in chipboard drilling. *Eur. J. Wood Prod.*, 77:957—959, 2019. doi:10.1007/s00107-019-01428-5.

[4] L. J. Chmielewski, A. Orłowski, G. Wieczorek, K. Śmietańska and J. Górski. Testing the Limits of Detection of the 'Orange Skin' Defect in Furniture Elements with the HOG Features. In Intelligent Information and Database Systems (ACIIDS 2017), PT II Proc. volume 10192 of *Lecture Notes in Artificial Intelligence*, pages 276–286 Kanazawa, JAPAN, 3-4 Apr 2017. Springer. doi:10.1007/978-3-319-54430-4_27

[5] G. A. Ruz, P. A. Estevez and P. A. Ramirez. Automated visual inspection system for wood defect classification using computational intelligence techniques. *International Journal of Systems Science*, 40(2):163–172, 2009. doi:10.1080/00207720802630685

[6] K. Śmietańska, P. Podziewski, M. Bator and J. Górski. Automated monitoring of delamination factor during up (conventional) and down (climb) milling of melamine-faced MDF using image processing methods. *Eur. J. Wood Prod.*, xx:xx—xx, 2020. (in review) (editorial postprocessing: doi:10.1007/s00107-020-01518-9)

Fig. 4. Examples of images used to evaluate the correctness. Colours of pixels: blue – pixel on the border of melamine, red – estimated line, purple – overlap.

[7] K. Szwajka and T. Trzepieciński. The influence of machining parameters and tool wear on the delamination process during milling of melamine-faced chipboard. *Drewno* 60(199):118–131, 2017. doi:10.12841/wood.1644-3985.189.09

[8] L. J. Chmielewski, M. Bator and M. Olejniczak. Advantages of using object-specific knowledge at an early processing stage in the detection of trees in LIDAR data. In Computer Vision and Graphics: Proc. ICCVG 2014, volume 8671 of *Lecture Notes in Computer Science*, pages 145—154, Warsaw, Poland, 15-17 Sep 2014. Springer. doi:10.1007/978-3-319-11331-9_18

Image Annotating Tools for Agricultural Purpose – A Requirements Study

Marcin Bator and Maciej Pankiewicz
Institute of Information Technology
Warsaw University of Life Sciences – SGGW, Poland

Abstract. Images of natural scenes, like those relevant for agriculture, are characterised with a variety of forms of objects of interest and similarities between objects that one might want to discriminate. This introduces uncertainty to the analysis of such images. Requirements for an image annotation tool to be used in pattern recognition design for agriculture were discussed. A selection of open source annotating tools were presented. Advices how to use the software to handle uncertainty and missing functionalities were described.

Key words: image annotation, agriculture, uncertainty.

1. Introduction

With the growing advancements in the field of supervised learning, the need for a labelling tool that optimizes the process of image annotating (both automatic and manual) increases. There are both one-fits-all tools available as well as products tailored to the specific needs of certain applications, e.g. medical diagnosis [1] or robotics competition [2]. Several attempts to comparing features of the available tools have been proposed by the research community, e.g. in respect of its ability to utilize ontologies [3, 4, 5], its appropriateness for e-health and generally for the medical domain [6, 7] or suitability for the biomedical appliances [8]. However, research community in the field of life sciences, especially in agricultural sciences, faces several problems occurring specifically in this area and this article addresses these specific issues.

Increasingly more publications presenting various subjects for pattern recognition in the field of agriculture appears (e.g. cows health monitoring [9], Spider Mites detection [10], weeds detection [11], precision herbicide application [12]). Pattern recognition in this field is quite often difficult due to the diversity of forms of the same objects and similarity of a sought object to other objects. This difficulty may introduce uncertainty about the border of an object or uncertainty about object classification.

As long as there is no agriculture dedicated annotating software available, in the next section of this paper a best selection of software that could be used is presented. In the following sections the several uncertainties of a different kind which could be met in tasks of pattern recognition in agriculture, and suggestions how to handle them are described.

Machine GRAPHICS & VISION 28(1/4):69–77, 2019. DOI: 10.22630/MGV.2019.28.1.7 .

2. Selected software

In the search executed for this paper, no software dedicated to the agricultural image annotations were found. This shifted the interest to search explicitly for the software that is open source. Open source software has many advantages in the analysed case. One of the main advantages of the open source systems is the ability of free adaptation of the software to the needs. It may be easily carried out by the community if the feature request is submitted. The community may develop and share the software for free. So, this adaptation does not necessarily impose having programming skills. But if one has programming skills, one can modify the software according to needs. The licence and access to source code allow for it.

Fortunately, there is a wide range of the free or open source software available. This analysis does not suffer from the exclusion of the proprietary software from the analysis, as the open source tools described in the following section are meeting the state-of-the art requirements of the image labelling.

The last reason for focusing explicitly on the free software in this analysis is related to the financial aspect. Although many proprietary tools provide free or trial access to the software (Labelbox [13], oclavi.com, rectlabel.com, supervise.ly), other products usually require purchase (e.g. taqadam.io, prodi.gy, scale.com, appen.com, thehive.ai).

The criteria used to include a software in this analysis were the following: 1) software has been accessible through the online search engines with the phrase "image annotation tool"; 2) the software has been available on the lists of the top image annotation tools found through the online search engines with the phrase "top image annotation tools". Finally, ten projects of annotating tools were selected.

The properties helping in comparison of the searched software are summarized in the table 1 (offline tool), and table 2 (online tools), sorted by the number or contributors.

The following criteria were chosen to summarise the software so that it should be helpful in consideration what would be the best solution in a specific case.

Online/Offline It is important if the process requires data sharing, e.g. if the images are annotated by several experts and their inputs need to be viewed, e.g., for the purpose of comparison or evaluation. Offline tools provide an option of exporting annotations usually in multiple data formats, therefore the information may be exchanged. However, this leads to creating multiple parallel data repositories and data synchronization may become a problem. Online tools support collaborative work on data sets and access of multiple users to one repository is possible. However, the size of the images in repository may be a problem, impacting the speed of annotation process, if the internet connection is slow. Always an online software may be used with advantages of an offline tool, if installed as a service on a local machine.

License type The code of the open source software developed by the community is made available at the popular software version control services (e.g. Github, Gitlab) with

Tab. 1. Examined offline software.

	License	Language	Annotation types	Labels	# of contr.	update 2019
LabelImg [14]	MIT	Python	b-box	User definable	57	yes
Sloth [15]	GNU GPL	Python	b-box point, polygon	User definable	8	no
Mask Editor [16]	MIT	Matlab	b-box, polygon, ellipse, freehand	User definable	1	no
Ratsnake [1]	Free software	Java	b-box, polygon, freehand	User definable	n/a	n/a

an appropriate license. New features may be added depending on the community demand. If required, individual and specific features may be freely added by every user with programming background.

Language Language denotes the primary programming language used for the development of the tool. Online projects mostly utilize more than one programming language or technology. For the online tools the main backend technology is provided.

Annotation type Several types of annotation types are usually available in the form of the following shapes: rectangle (bounding box, b-box), circle, ellipse, point, or line. Using a predefined shape speeds up the process. More complex shapes may be marked using the polygon feature or freehand shape. There exists also an option of creating annotations based on several examples of the class to annotate.

Labels Every annotation may be assigned to one or more categories which are usually defined by the user. Online tools may provide an option of sharing labels created by different users, however in some projects label categories may be imposed and defined by an administrator.

Number of contributors and updates This information makes it possible to take into consideration the probability of experiencing the development of the software and the emergence of new features. Repositories of the open source projects are publicly available and the history of changes in the project may be tracked. Several of the selected projects are constantly developed by large groups of contributors.

Tab. 2. Examined online software.

	License	Language	Annotation types	Labels	# of contr.	update 2019
CVAT [17]	MIT	Python	b-box, polygon, polyline, points	User definable	50	yes
LabelMe [18]	GNU GPL	Python	Polygon, b-box, circle, line point	Free text	31	yes
imglab [19]	MIT	Java Script	b-box, polygon, circle, ellipse, point	User definable	26	yes
Image Tagger [2]	MIT	Python	b-box, polygon, line, point	User definable	13	yes
Via [20]	BSD-2	Java Script	b-box, polygon, circle, ellipsis, polyline, point	User definable	9	yes
Rhoban Tagger [21]	MIT	PHP	Examples	Admin defined	1	no

3. Agriculture requirements

Pattern recognition of agriculture objects cause a need to add some special requirements to the image annotating tools. All of them are driven by the uncertainty imposed by nature. Some of living forms try to look like others to hide. Some of living forms change their look to adapt to the environment. Different look is also related to different life stage or growth phase.

Annotation tools presented already could meet some of the extra requirements just by the untypical use of the software functionality, but some of them need an extension of the software. Of course at first the requirements for such tools should be defined.

As an introduction to the requirements for the image annotation tools, reasons for pattern recognition being a hot topic in agriculture will be presented. The first reason is the ecological approach to agriculture. It can have a different scale from just reduction of use of chemicals to even something called 'organic farming'. Secondly, economics are considered, due to the observable trend of equipment becoming cheaper and labour costs growing larger.

A current philosophy applied to agriculture is not to eliminate pests or weeds but to allow them to exist, as long as it is kept on a safe level. The consequence of such approach is resignation from preventive use of pesticides. To reduce usage of chemicals the knowledge about the condition of the crop is needed. This means that the crop should be monitored. In organic farming chemistry can not be used at all and weeds should be removed mechanically. The consequence of the above, is that possibilities of usage of pattern recognition in that field are widely investigated. In such circumstances a lot of AI is required to analyse increasing number of samples and to navigate robots. On the other hand the AI needs annotated learning material. The consequent requirement is development of the good tool for annotating, that is taking into account difficulties of such kind of images.

The diversity of possibilities for pattern recognition is significant. It reflects the diversity of agriculture aspects. The diversity can be categorised in several ways. At first it can be grouped by animals and plants. The second aspect is the location of image collection: field, buildings, laboratories. This strongly influences possible assumptions about what can be invariant and what can vary (e.g. lighting, relative position of objects and the camera, etc.). The third aspect is related to the requirements for the result provided by a system: classification of images, detection of objects, measurement of objects (quantitative description).

As long as the source of difficulties is the same in all of agricultural pattern recognition cases – the Nature, an actual problem usually is specific for a project. In the following analysis some of examples are presented.

3.1. Labelling uncertainty

As was mentioned before, it is a part of survive strategy to look similar. But it is not the only reason indicating of a problem of difficulty in discrimination. Lets consider an example. There is a task of monitoring of influence of different treatment for activity of some insects. An idea is to measure the activity indirectly through the count of their excrements. Unfortunately, at least for a non biologist, such excrements (annotated) look like other objects existing on the same picture that are not annotated. Someone responsible for the pattern recognition aspects, would like to know what have to be determined as the object and what out of the similar objects is not, and in which cases

Fig. 1. An example image presenting many aspects. a poppy in wheat is a weed. Poppies are in different life stage (flower, green capsule, dried that make its stem similar to wheat). Overlapping between poppies, and between poppies and wheat. (source: `https://commons.wikimedia.org/wiki/File:Klatschmohn_in_Weizenfeld.JPG`)

the expert was not sure of the correct determination. Even for a biologist, without additional knowledge, it may be difficult to discriminate correctly the specified objects. If at the current stage of a project there is only the picture available to the expert, the available label should be yes/no/not sure.

The mentioned requirement may be solved with the software. The uncertainty can be annotated by adding to one object few labels. All of the presented software allow to do that. The problem is a shift to the following stage of system development, that means to the stage of metrics defining and proper interpretation of labels of annotation.

3.2. Uncertainty of borders

The problem of borders is something more than a fuzzy border and problem of decision which pixels to choose. Of course, this is a problem to annotate the object when the border is not sharp. But a solution can be quite easy. The software should allow to put an annotation in annotation. In such a case more label categories should be added. Then the maximum range of an object should be annotated at first, and in the next step the minimum range marked into the maximum one.

More difficult problem is when objects overlap each other (eg. like the wheat and poppy on fig. 1). The requirements of quantitative description became a difficult task. If e.g. the relation of numbers of objects of different kind is expected then the whole object should be counted as one. In order to be able to learn and evaluate the result, within annotations there should be such an information.

3.3. Risk of omission

Omission of an object during annotating has two main consequences. The first consequence is that the interesting object (that might be treated as an example for the learning process) would be lost. The second, and even more important, consequence is that a proper answer of a system verified on such data would be wrongly judged as returning en error of a false positive type. The consequent requirement is to ensure that all areas of a picture have been analysed. In some cases pictures have such a size that it is not possible to have on a screen the whole picture and still to see necessary details of objects or even to identify them. It would be good if a software scrolls automatically the picture in a resolution set while the expert is analysing the picture. Unfortunately the analysed software do not have such functionality and scrolling can be made only manually.

4. Conclusions

Some of the described requirements for an annotating tool may be satisfied with existing software, some of them require enhancement of the functionality of the software. Even though, the proposed non typical usage of software would allow one to handle the specified problems, it should be made carefully and some additional processing of results of annotation should be made. Functionality is one aspect of a good software, another key aspect is ergonomics. A software with good ergonomics can speed up the work and reduce the number of mistakes. As the number of project and their scale increase, the need for dedicated tools also increases.

Good annotating software is just the first step in the preparations to build a system for pattern recognition. To handle the uncertainty of the knowledge incorporated in the process of annotation of images the next step is required. This step is to properly define or choose the metrics that could rank potential systems in the way which reflects the expectations.

References

[1] D. K. Iakovidis, C. Smailis, T. Goudas, and I. Maglogiannis. Ratsnake: A Versatile Image Annotation Tool with Application to Computer-Aided Diagnosis. *The Scientific World Journal*, vol. 2014, Article ID 286856, 12 pages, 2014. doi:10.1155/2014/286856.

[2] N. Fiedler, M. Bestmann, N. Hendrich. ImageTagger: An Open Source Online Platform for Collaborative Image Labeling. In Holz et al., editors,*RoboCup 2018: Robot World Cup XXII.* vol 11374 *Lecture Notes in Computer Science*, Springer 2019 doi:10.1007/978-3-030-27544-0_13 `https://github.com/bit-bots/imagetagger`

[3] Erdmann, M., Maedche, A., Schnurr, H. P., and Staab, S. From manual to semi-automatic semantic annotation: About ontology-based text annotation tools. In Proceedings of the COLING-2000 Workshop on Semantic Annotation and Intelligent Content (pp. 79-85). 2000, August.

[4] Sazedj, P., and Pinto, H. S. Time to evaluate: Targeting Annotation Tools. In SemAnnot@ ISWC. 2005, November.

[5] Dasiopoulou, S., Giannakidou, E., Litos, G., Malasioti, P., and Kompatsiaris, Y. A survey of semantic image and video annotation tools. In Knowledge-driven multimedia information extraction and ontology evolution (pp. 196-239). Springer, Berlin, Heidelberg. 2011.

[6] S. Seifert, M. Kelm, M. Moeller, S. Mukherjee, A. Cavallaro, M. Huber and D. Comaniciu. Semantic annotation of medical images. In Medical Imaging 2010: Advanced PACS-based Imaging Informatics and Therapeutic Applications (Vol. 7628, p. 762808). International Society for Optics and Photonics. 2010, March.

[7] K. Chehab, A. Kalboussi, and A. H. Kacem. Study of Annotations in e-health Domain. In International Conference on Smart Homes and Health Telematics (pp. 189-199). Springer, Cham. 2018, July.

[8] M. Neves, U. Leser. A survey on annotation tools for the biomedical literature. Briefings in Bioinformatics, 15(2):327—340, March 2014. doi:10.1093/bib/bbs084

[9] T. Wirthgen, G. Lempe, S. Zipser and U. Grünhaupt. Level-set based infrared image segmentation for automatic veterinary health monitoring. In L. Bolc et al., editors, *Computer Vision and Graphics: Proc. Int. Conf. ICCVG 2012*, volume 7594 of *Lecture Notes in Computer Science*, pages 685–693, Warsaw, Poland, 24-26 Sep 2012. Springer.

[10] L. Yang, D. Zhang, J. Luo, Z. Wang and C. Wu. Automatic Recognition for Cotton Spider Mites Damage Level Based on SVM and AdaBoost. *Nongye Jixie Xuebao/Transactions of the Chinese Society for Agricultural Machinery* 50(2):14-20 2019.

[11] A. Farooq, X. Jia, J. Hu and J. Zhou. Multi-resolution weed classification via convolutional neural network and superpixel based local binary pattern using remote sensing images. *Remote Sensing* 11, 2019. doi:doi:10.3390/rs11141692

[12] R. Aravind, M. Daman, and B. S. Kariyappa. Design and development of automatic weed detection and smart herbicide sprayer robot. In *2015 IEEE Recent Advances in Intelligent Computational Systems, RAICS 2015* 257–261, 2015.

[13] Labelbox. [Online; accessed 10 Dec 2019]. `https://www.labelbox.io/`.

[14] Labelimg. [Online; accessed 10 Dec 2019]. `https://github.com/tzutalin/labelImg`.

[15] M. Bauml. Sloth documentation. [Online; accessed 10 Dec 2019]. `http://sloth.readthedocs.io/en/latesthttps://github.com/cvhciKIT/sloth`.

[16] C. Zhang, K. Loken, Z. Chen, Z. Xiao and G. Kunkel. Mask editor: an image annotation tool for image segmentation tasks (2018). arXiv preprint arXiv:1809.06461. `https://github.com/Chuanhai/Mask-Editor`.

[17] Computer Vision Annotation Tool (CVAT). [Online; accessed 10 Dec 2019]. `https://github.com/opencv/cvat`.

[18] B. C. Russell, A. Torralba, K. P. Murphy and W. T. Freeman. Labelme: a database and web-based tool for image annotation. *Int. J. Comput. Vis.*, 77(1–3):157—173 2008. doi:10.1007/s11263-007-0090-8. `https://github.com/wkentaro/labelme`.

[19] imglab. [Online; accessed 10 Dec 2019]. `https://github.com/NaturalIntelligence/imglab`

[20] A. Dutta ad A.Zisserman. *The VIA Annotation Software for Images, Audio and Video* In Proceedings of the 27th ACM International Conference on Multimedia (MM '19), October 21–25, 2019, Nice, France. ACM, New York, NY, USA, 4 pages. doi:10.1145/3343031.3350535

[21] Rhoban. [Online; accessed 10 Dec 2019]. `http://rhoban.com/tagger`, `https://github.com/Rhoban/tagger`.

Appendix: Repositories of presented open source annotation tools

- `https://github.com/bit-bots/imagetagger`
- `https://github.com/Rhoban/tagger`
- `https://github.com/cvhciKIT/sloth`
- `https://github.com/Chuanhai/Mask-Editor`
- `https://github.com/wkentaro/labelme`
- `https://github.com/opencv/cvat`
- `https://github.com/NaturalIntelligence/imglab`
- `https://github.com/bit-bots/imagetagger`
- `https://github.com/tzutalin/labelImg/tree/master`
- `https://gitlab.com/vgg/via`