On development of models and algorithms for automated metallographic measurement of visible metal slice grain sizes

V A Kovun and I L Kashirina
Department of Applied Mathematics, Informatics and Mechanics, Voronezh State University, 1, University Square, Voronezh, 394018, Russia
E-mail: sidav94@gmail.com

Abstract. The article proposes an approach to metallographic research based on solving the problem of semantic segmentation using a trained neural network classifier. To solve the problem of isolating grains in micrographs of longitudinal and transverse slices of metal, the neural network model U-Net was adapted. In order to obtain closed image contours, a post-processing algorithm was developed using the OpenCV open source computer vision library. The article describes the training of a neural network and the conversion of its results, as well as the comparative analysis of the histograms between the reference grain area distribution and the distribution obtained using the proposed algorithm.

1. Introduction
Metallography is an important area in metal science, a classic method of research and quality control of metallic materials. Quantitative metallography studies the quantitative characteristics of the microstructure of steels and alloys and provides the most objective assessment of the metal structure. According to GOST 5639-82, which is the most recent version of technical standard on steel and alloys accepted in the Commonwealth of Independent States, selection of grains in micrographs of the longitudinal and transverse slices of metal samples and measurement of their absolute and relative sizes is the one of the most important parts of the structural metallographic analysis of samples.

Recently, quantitative metallography has undergone significant changes due to the advent of automatic image analyzers (AAI) such as the Thixomet image analysis solution, but the task of fully automated evaluation of the metal structure is yet to be solved. Complexity is created by the presence of artifacts in the form of horizontal lines on the original image caused by the features of the technical process of creating slices as well as often poorly distinguishable grain boundaries.

The proposed article is dedicated to solving the following problems:

• Isolation of closed contours of metal grains in the original image;
• Processed image output with grains outlined;
• Automated calculation of grain areas;
• Building a histogram of the distribution of grain areas.
The proposed approach to metallographic research is based on segmentation using a trained neural network classifier alongside with the OpenCV open source computer vision library. Source data (100 micrographs of longitudinal and transverse slices of metal samples) were provided by the Laboratory of Metallurgy and Metallophysics of NLMK company, which is one of the leading international manufacturers of high quality metal products.

To solve the problem of isolation of grain boundaries in microphotographs of metal slices, a convolutional neural network of the U-Net architecture is used. That architecture was developed by Olaf Ronnenberger et al. in 2015 to solve the problem of segmentation of biomedical images [1]. The U-Net architecture took the first place by a wide margin in the 2015 ISBI competition for the allocation of cell boundaries in microphotographs of biological preparations. It should be noted that microphotographs of biological cells visually have much in common with microphotographs of metal grains, which was one of the main reasons for choosing this particular architecture to solve the problem. Other reasons include the fact that the U-Net network was originally designed to work with training samples of small size, which can significantly reduce the time required for marking training data, considering that this is a rather labour-consuming process when identifying metal grain boundaries. In addition, the U-Net network spends much less time on image processing than most classical computer vision algorithms. It is worth noting that in 2018 there was developed a modification of U-Net architecture called U-Net++ [2]. The differences of this modification from the base model are, in particular, in the form of an optimized quality functional as well as in the elimination of the semantic gap between the compressing and expanding convolutional layers.

2. Materials and methods

U-Net is a variant of a fully convolutional neural network [3], i.e. it does not contain fully connected layers. A general view of the architecture of the U-Net network is shown in figure 1 (taken from [4]), where each blue rectangle corresponds to a multi-channel feature map of examined objects. The number of channels is indicated at the top above each rectangle. The size of each card is indicated in the lower left corner of the rectangle. Arrows indicate various operations. The U-Net network contains a compression path (on the left) and an extension path (on the right), so its architecture is similar to the letter U (which is reflected in the network name). The compression path is similar to a typical convolutional network, it contains two consecutive 3 × 3 convolutional layers, after which the convolution result is processed using the ReLU activation function, and then fed to the input of the unification layer, on which the maximum value for each fragment of the 2 × 2 sized card is calculated in increments of 2. Each block of the extension path contains a layer inverse to the compression one which increases the size of feature maps, followed by a convolution layer with a convolution of 2 × 2, which reduces the number of feature channels. After that there is concatenation with the feature map from the compression path which is again followed by the two consecutive convolutional layers 3 × 3, after each of which the ReLU function is calculated again. On the last layer, the convolution 1 × 1 is used to form the required number of output classes from each of the received vectors of features. The U-Net network has several advantages, it is simple conceptually, poorly retrained, and it needs less data for training due to the relatively small number of parameters.

Examples of output images are presented on figure 2.

Unfortunately, some of the output images of the U-Net network contained open borders (poorly distinguishable in the input image as well, see figure 3), which led to erroneous values when calculating the grain areas. To close the boundaries of all the grains, an additional post-processing algorithm using the OpenCV library was developed, which includes the following steps:

- Pixel contrasting, image erosion and image dilation;
- The watershed algorithm usage for searching for segments in the image;
Approximation of the contours and areas of the segments in the algorithm output.

First, the image is subjected to binary threshold contrasting (1), where $p_{i,j}^n$ is the new pixel value at the coordinates $(i,j)$ in the image, $p_{i,j}$ is the old pixel value at coordinates $(i,j)$ in the image, $T$ is the brightness threshold, and 0 and 1 correspond to black and white, respectively.
Figure 3. Input and output of U-Net network with some open borders visible.

\[
p^o_{ij} = \begin{cases} 
1, & p_{ij} > T \\
0, & p_{ij} \leq T 
\end{cases}
\]  

To find the polygons corresponding to the segments, the watershed algorithm is used, followed by polygonal approximation of the regions. The watershed algorithm is a recursive image segmentation algorithm based on the simulation of filling a relief with water. In the present work a variant of the watershed algorithm proposed by F. Meyer is used [5]. The main idea of this algorithm is a special choice of starting points (markers) for the image to be filled from.

1. A set of markers (initial, correctly labeled image pixels) is selected. Markers are marked with different labels.

2. Pixels adjacent to the marked zones are queued in order of increasing priority, where priority is equal to the norm of the gradient of the image pixel: 
\[g = \sqrt{g_x^2 + g_y^2},\]
where \(g_x\) and \(g_y\) are the values of the increase in brightness along the x and y coordinates, respectively.

3. The highest priority pixel is retrieved from the queue. If the already marked neighbors of the extracted pixel have the same label, then the current pixel is marked with the same label. All unmarked neighbors that are not yet queued are now placed in the queue.

4. Step 3 is repeated until the queue is cleared.

Starting markers are selected as follows.

1. A map of Euclidean distances from each pixel to the nearest black (zero in brightness) pixel (1) is constructed for each pixel of the image, where \(d_i\) is the \(i\)-th output value of the distance, \((x_i, y_i)\) is the \(i\)-th coordinate input pixel, \((b_x, b_y)\) is the black pixel closest (in terms of Euclidean distance) to the \(i\)-th pixel in the input image.
\[d_i = \sqrt{(x_i - b_x)^2 + (y_i - b_y)^2}.\]

2. Local peaks are located on the obtained distance map. Local peaks are local maxima of values on the distance map, separated by at least \(k_{min}\) pixels from each other and having a value no less than some \(d_{min}\).

3. Pixels corresponding to local peaks as well as eight adjacent to each of them are selected as markers.
3. Results and discussion
To train the U-Net neural network model there were 20 slices manually marked out, both longitudinal and transverse. 14 marked slices were used for training, 6 were used for testing. To increase the training sample, augmentation using the elastic network (elastic_transform), as well as rotations and reflections, was additionally used.

For this task, in contrast to the original model, the following layer sizes were used: 256-128-64-32-16-32-64-128-256. To feed the network input, the original image was cut into fragments (batches) of size 256 × 256. At the output of the network, images of the same size are obtained. After the cutting and augmentation process the sizes of the data sets have been increased to 2400 and 700 for training and test datasets respectively. Final predictions are glued together. As a function of errors (loss), binary_crossentropy was used (since the network must distinguish between grain boundaries and background, that is, divide the image pixels into two classes).

After 100 iterations, the accuracy of the network in the training set was the following: value of the error function loss is 0.2479; fraction of correct answers when classifying acc is 0.9083. The accuracy on the test sample was val_loss: 0.2867, val_acc: 0.8946.

There’s an example of algorithm’s work on test data. Figure 4 shows a fragment of a micrograph of a thin section of metal for metallographic investigation. Figure 5 shows the output of the U-Net network (after glueing the resulting 256 × 256 fragments into a full-size image). Figure 6 shows the final metallographic structure of the metal section obtained by post-processing the output of the U-Net network.

![Figure 4. Metal slice original image.](image)

![Figure 5. U-Net network segmentation output (all batches are glued together).](image)

In addition to the black-and-white image, the developed program is capable of displaying a color image in which cells that fall into the same range of areas on the histogram are colored in the corresponding colors (figure 7).
The final result of the developed algorithm is a histogram of the distribution of the observed areas of steel grains (figure 8). Figure 9 shows a histogram of the distribution of grain areas in the reference image.

Figure 6. Post-processing algorithm output.

Figure 7. Post-processing algorithm output with color-coded grain areas.

Figure 8. A histogram of the distribution of grain areas in the resulting image.
To compare the reference and obtained histograms, the compareHist() function from the OpenCV library was used. The correlation value (3) was used as the main quality statistic, which allows to evaluate how well the two histograms coincide with each other:

$$d_1(H_1, H_2) = \frac{\sum_I (H_1(I) - \overline{H}_1)(H_2(I) - \overline{H}_2)}{\sqrt{\sum_I (H_1(I) - \overline{H}_1)^2 \sum_I (H_2(I) - \overline{H}_2)^2}},$$  \hspace{1cm} (3)

where $H_1$ is the histogram of the reference sample, $H_2$ is the histogram of the reconstructed image, and $H_1(I), H_2(I)$ are the frequencies of the corresponding histogram intervals, and

$$\overline{H}_k = \frac{1}{N} \sum_J H_k(J).$$  \hspace{1cm} (4)

As an additional quality statistic the Chi-square value was also used, which estimates the consistency of the obtained and reference distributions:

$$d_2(H_1, H_2) = \sum_I \frac{(H_1(I) - H_2(I))^2}{H_1(I)}.$$  \hspace{1cm} (5)

When calculating the Chi-square intervals containing less than 4 observations were combined with neighboring ones.

The average correlation value constructed using the proposed algorithm and reference histograms on the test sample was 0.996, and the Chi-square statistic was 12.11, which indicates the consistency of the obtained distributions (at a significance level of 0.05, the critical value is $\chi^2 = 22.4$).

In addition, specialists of NLMK company compared the results of the algorithm for detecting the structure of metallographic samples proposed in this article with the results obtained using the Thixomet microstructure analyzer of metals and alloys and confirmed the high quality of the developed method.
4. Conclusion
The proposed method demonstrates the flexibility of neural networks in image processing. The U-Net neural network proved to be applicable not only in its initial task of detecting cell contours in microphotographs of biological preparations, but also in segmentation of microphotographs of metal slices. A good result is achieved by combining the neural network approach with the classical segmentation algorithm, which allows both algorithms to mutually compensate for the shortcomings. Comparison of the results of the output of the program with the reference samples demonstrates a good correlation and high accuracy of the results.

References
[1] Ronneberger O, Fischer P and Brox T 2015 Lecture Notes in Computer Science Medical Image Computing and Computer-Assisted Intervention MICCAI 234–241
[2] Zhou Z, Siddiquee M M R, Tajbakhsh N and Liang J 2018 Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support Lecture Notes in Computer Science 3–11
[3] Long J, Shelhamer E and Darrell T 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 3431–3440
[4] Sankesara H 2019 U-Net URL https://towardsdatascience.com/u-net-b229b32b4a71
[5] Beucher S and Meyer F 1993 Mathematical Morphology in Image Processing 34 433–481
[6] Shelhamer E, Long J and Darrell T 2017 IEEE Transactions on Pattern Analysis and Machine Intelligence 39 640–651
[7] Tajbakhsh N, Shin J Y, Gurudu S R, Hurst R T, Kendall C B, Gotway M B and Liang J 2016 IEEE Transactions on Medical Imaging 35 1299–1312
[8] Zhou Z, Shin J, Zhang L, Gurudu S, Gotway M and Liang J 2017 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 4761–4772
[9] Fang Z, Yue W, Zhitao X, Lei G, Jun W, Yanbei L and Wen W 2019 Laser & Optoelectronics Progress 56 061005
[10] Hamarneh G and Li X 2009 Image and Vision Computing 27 59–68
[11] Mangan A and Whitaker R 1999 IEEE Transactions on Visualization and Computer Graphics 5 308–321
[12] Patel T R, Bangalore P V, Bodduluri S, Anthony T, Monroe W S, Kandhare P G, Robinson J P, Nakhmani A, Zhang C and Bhatt S P 2019 Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (learning) - PEARC ’19 1–4