A Benchmark for Analyzing Chart Images

Zhipeng Luo(✉), Zhiguang Zhang(✉), Ge Li(✉), Lixuan Che(✉), Jianye He(✉), and Zhenyu Xu(✉)

DeepBlue Technology (Shanghai) Co., Ltd, Shanghai, China
{luozp,zhangzhg,lige,chelx,hejianye,xuzy}@deepblueai.com

Abstract. Charts are a compact method of displaying and comparing data. Automatically extracting data from charts is a key step in understanding the intent behind a chart which could lead to a better understanding of the document itself. To promote the development of automatically decompose and understand these visualizations. The CHART-Infographics organizers holds the Competition on Harvesting Raw Tables from Infographics. In this paper, based on machine learning, image recognition, object detection, keypoint estimation, OCR, and others, we explored and proposed our methods for almost all tasks and achieved relatively good performance.

Keywords: Image recognition · Document analysis · Text recognition and classification · Graphics recognition

1 Introduction

It is an efficient direction to get a structured description by inputting images, an excellent medium for data representation and interpretation. An effective chart summarization could help data analysts, business analysts, or journalists better prepare reports from data. However, understanding and getting insights from charts can be difficult and time-consuming. To get a better chart analysis algorithm, the CHART-Infographics Organizers holds the Competition on Harvesting Raw Tables from Infographics. The complex process of automatic chart recognition is divided into multiple tasks for this competition, including Chart Image Classification (Task 1), Text Detection and Recognition (Task 2), Text Role Classification (Task 3), Axis Analysis (Task 4), Legend Analysis (Task 5), Plot Element Detection and Classification (Task 6.a), Data Extraction (Task 6.b), and End-to-End Data Extraction (Task 7).

In this paper, we proposed our methods for the task1-6. Due to the difference between different tasks, we explored and proposed different methods. For task 1 and task 3, one is chart image classification another is text role classification task. So we based on Deep Convolutional Neural Network to complete task1 and Machine learning-based method for task3. For task2, which need to detect text blocks and Recognition the text, different from task 4, which need to detect tick point and associate with the corresponding value (a string). We based on object detection algorithm and OCR to complete task 2, see task4 as a standard keypoint estimation problem. Task 5, which need to detect and match legend with the data series name, we use an object detection based algorithm to solve it. For task 6, a relatively complex task needs to detect and classify each element in the plot area and output the raw data used to generate the chart image. We can only detect different elements separately, then classify them based on image features. Furthermore, deal with each class independently to transform the image coordinate system's element position to the raw data. Finally, based on our method, we achieved relatively good performance on almost all tasks.

2 Method

2.1 Task 1

The first sub-task target is to classify chart images by type both in Synthetic Dataset and UB PMC Dataset. These two independent sets of chart images have a different number of classes. For each type of Adobe Synthetic Dataset, we use 1000 for training and 200 for validation. Our baseline model is resnet-50 [1], and an ensemble of models from 5-fold cross-validation is used.

For UB PMC Dataset, a satisfactory performance can not be reached by the former method. We use grad-cam [2] to visualize the attention of the network to inspire the direction of improvement. Some different losses are tried, such as ArcFace [3], focal loss [22]. We choose Cross Entropy loss with Label-smoothing [6] finally.

We follow the BNNeck [7], adding a BN layer without the bias before the last linear layer of the network. Some methods to deal with long-tailed recognition are also tried for the unbalanced distribution. Apart from the conventional resample and re-weight methods, we try to decouple the representation from the classification by training the classifier using class-balanced data after freezing the backbone got from the instance-balanced data [8]. However, they fail to bring a positive impact because of the limited classes, perhaps.

2.2 Task 2

This sub-task concentrates on detecting and recognizing the text within the chart image. Based on the two-stages Object Detection model (Cascade R-CNN [9]), detect text boxes. In our results, the text box is easy to detect. The task's difficulty is recognizing the text from the rotated text. This task is to identify the text box with distinctive features, besides the foreground-background is clear, so the mask can easily extract the text block. We can get the mask and horizontal angle through the mask's minimum outer matrix and then make the text box horizontal box by an affine transformation.

As for the recognition algorithm, we adopt CRNN [10] and CTC Loss [11] as our pipeline. So that a better result can be achieved. In this task, we still focus on detecting the text box, and after trying some detection models, we adopted the current method for Object Detection. Because the detection target is simple, the model's proposal score is relatively high, and there are obvious errors in the proposal with a low score. We change the IOU threshold from 0.5, 0.6, 0.7 to 0.6, 0.7, and 0.8 to avoid learning samples with low quality. We also adjusted the anchor's scale and stride to fit the characteristics of this task. According to the above settings, localization performance improved through cascading refine boxes and suitable anchor parameters.

What's more, we explore the performance of different backbones such as ResNext [12], HRNet [13], GCNet [14].

2.3 Task 3

This sub-task focuses on identifying the role of each text block in a chart image, and text bounding boxes and transcripts are provided as input. Our technique includes two steps: feature extracting and classification using classifiers. The properties of the texts are used to define feature vectors [15]. The classifiers we use are Random Forest [16] and LightGBM [17].

The features are composed of the properties of the bounding boxes and the text content. These features are grouped into three categories. The first category contains the aspect ratio of the box, whether the text is numeric, whether the text is multi-line, angle of text, length of text, and the chart type. The second category includes three kinds of relative position information, which are the positions relative to the global bounding box for all elements, origin (The bottom left corner of the graph) and legend. The third category contains the number of horizontally/vertically aligned text boxes and horizontal/vertical range of the aligned text boxes. When judging whether the boxes are aligned, the centre points of the boxes, the upper left corners and the lower right corners are used, respectively.

The positions of the graph and legend mentioned above are obtained by a detection model. We train a CenterNet [18] with the backbone DLA-34 [19]. The two categories are graph and legend, and the input size is 512*512.

Random Forest and LightGBM are used to classify the roles of texts. One of the advantages of random forest is that it still has good performance when there are missing features. LightGBM has the advantages of high training efficiency and high accuracy. When training the models, each class is assigned a weight inversely proportional to its frequency in the input data.

Considering that there are cases where legends are not predicted by the model, two classification models are trained, using and not using legend location information respectively. In the inference stage, which model is selected for prediction depends on whether the legend is detected.

2.4 Task 4

In this task, we need to output the location and value of each tick mark on both the X-axis and Y-axis. Tick locations are represented as points and must be

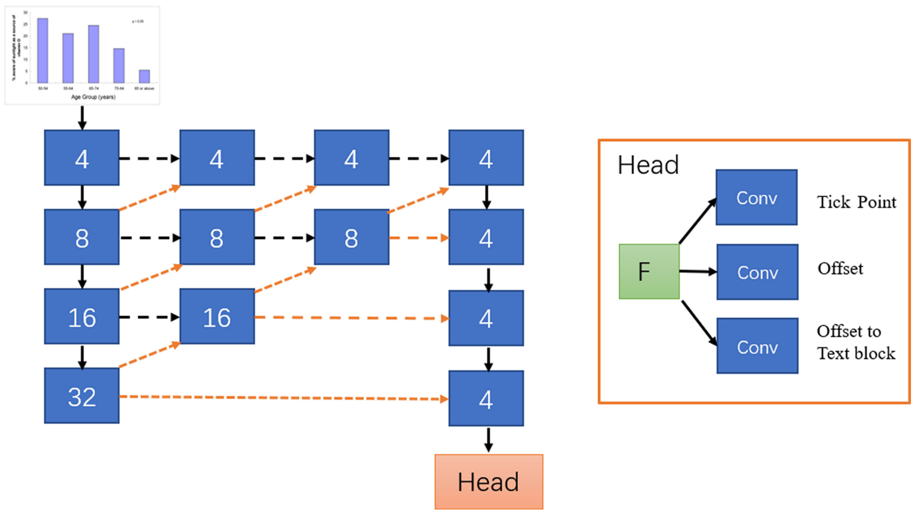


Fig. 1. The pipeline of our model. Dla34 as backbone and augment the skip connections with deformable convolution from lower layers to the head module. In the head module, has three output branches.

associated with the corresponding value (a string). To complete task 4, we split the task into two subtasks: tick point detection and matching.

Model. In the tick point detection task, we see the task as a standard keypoint estimation problem. We refer to CenterNet, use DLA-34 as backbone and add more skip connections from the bottom layers and upgrade every convolutional layer in upsampling stages to deformable convolutional layer [21], finally get output features with size 1/4 of the input and feed the features to head module. In the head module, there include three branches. First is tick point detection branch. Peaks in the heatmaps predict by this branch correspond to tick points and classify points to X-axis or Y-axis by two different heatmaps. To recover the discretization error caused by the output stride, we additionally predict a local offset acts on tick points locations in the offset branch. In the 3rd branch, we additionally predict an offset from tick points to the centre of corresponding text blocks(OTB), which only used in the training stage (see Fig. 1).

Loss Function. To train the tick point detection branch, we use focal loss as loss function to do pixel-wise logistic regression. To train the local offset prediction branch, we use Wing Loss [23] which is proposed in Facial Landmark Localisation task and designed to improve the deep neural network training capability for small and medium-range errors. To train the OTB branch, we simply use Smooth L1 [24] as loss function.

Matching. After getting the detection tick points to be classified into the X-axis or Y-axis, then use linear distribution check to filter out outlier points. To match tick points with text blocks, firstly, we use CenterNet trained by plot box to get the position of the plot box, then based on the plot box position and linear distribution check to further improve the classification accuracy of tick label text blocks. Finally, based on the L1 distance of X-axis or Y-axis between tick point and tick label text blocks, we match the tick points and text blocks.

2.5 Task 5

Task 5 is to associate each legend label text with the corresponding graphical style element within the legend area. We adopt a method of first detecting the legend elements and then matching the legend elements and the legend labels. In order to further improve the accuracy of matching, the legend pairs and legend elements are detected at the same time. The results of legend pairs are used to assist in matching.

Our method for Task5 is divided into three steps:

Firstly, the same method as Task 3 is used to classify the text boxes. And then the text boxes of the legend label category are filtered out.

Secondly, CenterNet is used to detect two categories of legend element and legend pair. The legend pair is obtained by merging the borders of legend elements and the corresponding texts.

Finally, legend elements and legend labels are matched. For each legend label, if there is a legend element in the same pair box, the element is matched. Then the Hungarian Algorithm is used to match the remaining legend elements detected and the legend labels. In order to reduce the impact of the classification error of task 3 on the results of task 5, post-processing is done. For the images without legend labels in the results of task 3, if the legend elements and legend pairs are detected at the same time, the legend elements are matched with all text boxes.

2.6 Task 6

For task 6, the goal is to detect and classify each element in the plot area and output the raw data that was used to generate the chart image. The representation of the element varies, but mainly contains two types of labels: box and point.

Task 6a. In this task, we detect elements by box-based detection algorithm and point-based detection algorithm separately. To detect box representation element like Grouped horizontal bar, Stacked horizontal bar, Grouped vertical bar, Stacked vertical bar, we use CenterNet, which is an anchor free object detection algorithm. To detect point representation element, we use the same point detection algorithm as Task4 and remove OTB branch. Moreover, based on the difference between the different type of chart, we predict different number of channels of heatmaps in tick point detection branch. Like Horizontal box and Vertical box, which have five points in each box, that can predict heatmaps with five channels to complete detection and classification. However, for Line, Scatter, it can only see all element as one class.

In this task some types of chart are need to group elements. For Horizontal box and Vertical box, we can group the five categories of points by L1 distance of X-axis or Y-axis. But for Line and Scatter, it is more complicated. For pictures with legends, the elements of lines and scatters are grouped according to legend elements. We extract the colour histogram features and Hog features [25] of the legend elements and points, respectively. Then each element is divided into a group corresponding to the legend based on the distance between the features. For pictures without legends, after extract the colour histogram features and Hog features of each point respectively, then use K-means to group them.

Task 6b. For task 6b, in the task 6a, we have got position representation in the image coordinate system of each element, the problem is to get position representation in the axis coordinate system. After analysis, we divided UB PMC dataset into five categories and Adobe Synth dataset into eight categories, then deal with each class independently. For data sequence, if the x values are numerical values, we computed x-axis values by interpolation. If the x values are string values, we use L1 distance to find the nearest character content. Moreover, do targeted treatment for different particular situations. For images containing bar and boxplot elements that need to group, the same method based on colour histogram features and Hog features are used like task 6a.

3 Experiments

3.1 Task 1

For Adobe Synthetic Dataset, average the output vector of the fully connected layer of each model in the 5-fold cross-validation. The maximum value of the averaged vector corresponds to the category of the prediction. The accuracy of our randomly divided validation set can reach 100%.

For UB PMC Dataset, some augmentation methods like cutmix [4] and mixup [5] are used in the training phrase, and TenCrop is adopted in the test phrase. The ablation experiment results on the validation set are shown in Table 1. Our results on the two test sets are shown in Table 2.

3.2 Task 2

Data-Augmentation. Because some of the text is slanted, we rotate the text lines to the horizontal position by projection, and our work involved applying an effective data augmentation strategy that included transformations informed specifically by the domain of the data(PMC dataset and Adobe Synth dataset)

Table 1. Ablation experiments for PMC.

Model	Accuracy	F1 score
resnet50(**baseline)**	0.947873	0.922636
resnet50 + cutmix	0.947251	0.924866
resnet50 + mixup	0.948833	0.0.923047
resnet50 + focal loss	0.949153	0.928227
resnet50 + ArcFace	0.948833	0.926474
resnet50 + Label Smoothing	**0.952350**	**0.931243**

Table 2. Task1 results on different datasets.

Dataset	Average Per-Class F-Measure
UB PMC Dataset	90.43%
Adobe Synth Dataset	100%

Training. We followed the training methodology they used and trained with the SGD optimizer using all of the default parameter values, including the base learning rate of 0.01. Also, as in both and, we exponentially decayed the base learning rate. For our experiments, which trained for 20 epochs, we applied an exponential decay rate of 0.99 per epoch, clamped to a minimum of 1e−6.

Results. Our results are shown in Table 3, we got 99.98 AP_{50} on the local validation dataset, there is a big gap between online and offline results probably due to the lack of extra training data.

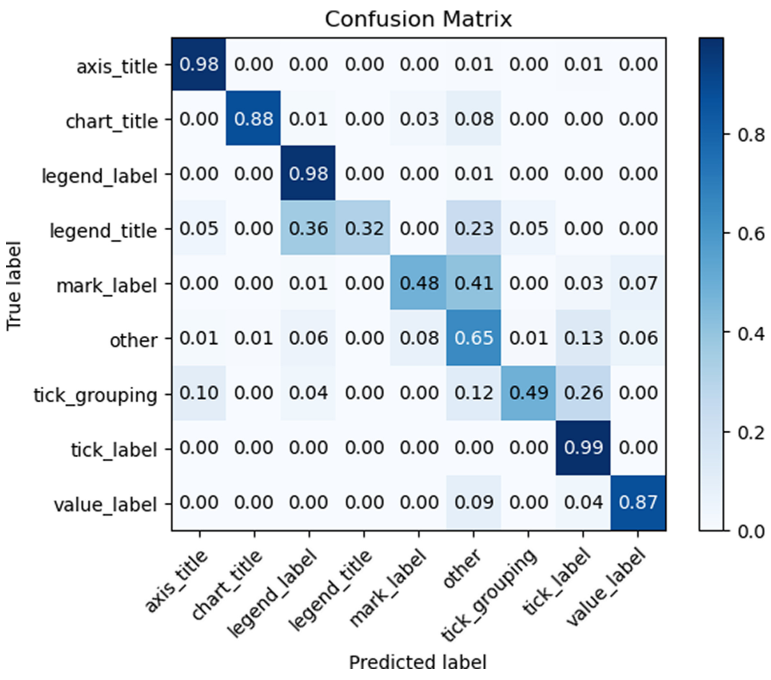
Table 3. Results on different datasets.

Dataset	IOU score	OCR score
UB PMC Dataset	**0.44**	0.70
Adobe Synth Dataset	**0.74**	0.58

3.3 Task 3

The two data sets are trained and inferred separately. For Random Forest and LightGBM, the number of trees is 1000. In order to improve accuracy, we use five-fold cross-validation and ensemble learning. The method of ensemble learning we use is weighted average strategy. Specifically, in the inference stage, for Random Forest and LightGBM, we predict the test images with 5 models trained with datasets generated by five-fold cross-validation. We average the output probability of 10(2*5) prediction results to get the final results. The final results on

PMC and Adobe Synthetic datasets are shown in Table 4. The confusion matrix on PMC dataset is shown as Fig. 2.



	axis_title	chart_title	legend_label	legend_title	mark_label	other	tick_grouping	tick_label	value_label
axis_title	0.98	0.00	0.00	0.00	0.00	0.01	0.00	0.01	0.00
chart_title	0.00	0.88	0.01	0.00	0.03	0.08	0.00	0.00	0.00
legend_label	0.00	0.00	0.98	0.00	0.00	0.01	0.00	0.00	0.00
legend_title	0.05	0.00	0.36	0.32	0.00	0.23	0.05	0.00	0.00
mark_label	0.00	0.00	0.01	0.00	0.48	0.41	0.00	0.03	0.07
other	0.01	0.01	0.06	0.00	0.08	0.65	0.01	0.13	0.06
tick_grouping	0.10	0.00	0.04	0.00	0.00	0.12	0.49	0.26	0.00
tick_label	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.99	0.00
value_label	0.00	0.00	0.00	0.00	0.00	0.09	0.00	0.04	0.87

Fig. 2. The confusion matrix on PMC dataset.

Table 4. Results for tasks 3.

Dataset	Synthetic	PMC
Average Per-Class F-measure	99.92%	77.19%

3.4 Task 4

In our experiment, we firstly split the training dataset to Train: Val: Test by 7:2:1. Fix the input resolution to 1024 1024 while training. Keep the original image size while verifying and testing. For data augmentation, we use random scale, blur, channel shuffle, hue saturation, invert, grayscale and so on. And, training the model use or not pre-trained model weight. In particular, during training, we exchange X-axis and Y-axis label of some types of charts like Horizontal bar, Grouped horizontal bar, Stacked horizontal bar, Horizontal box.

Table 5. Performance on the custom test dataset of Adobe Synth.

	Average recall	Average precision	Average F-measure
With pretrained	0.98913	0.99027	0.98970
Without pretrained	0.99594	0.99643	0.99619
+ OTB	0.99611	0.99651	0.99631
Ensemble	0.99826	0.99853	0.99839
Test challenge	–	–	0.9990

Table 6. Performance on the custom test dataset of UB PMC.

	Average recall	Average precision	Average F-measure
With pretrained	0.92032	0.93234	0.92629
Without pretrained	0.93460	0.94606	0.94029
+ OTB	0.94220	0.94434	0.94327
Ensemble	0.95642	0.95861	0.95751
Test challenge	0.81560	0.81000	0.81280

Furthermore, to get better performance, we split datasets into 5-folds for k-fold cross-validation, and ensemble results by Adaptive threshold point NMS algorithm. Finally, on the test challenge stage, we re-split the dataset by 10:2 as Train and Val set, and ensemble results of each type of chart by the performance on the Val set. As summarized in Tables 5 and 6.

3.5 Task 5

CenterNet with the backbone DLA-34 is used to detect legend element and legend pair. We use random flip, cropping, color jittering, and randomly shuffling RGB channels as data augmentation. The input resolution of the network is 1024*1024. The network is trained with Adam [20] for 200 epochs with the initial learning rate being 3.75e-4 and a batchsize of 32 (on 4 GPUs). The learning rate is reduced by a factor of 10 at 150 and 170 epochs, respectively. The final results on PMC and Adobe Synthetic datasets are shown in Table 7.

Table 7. Results for tasks 5.

Dataset	Synthetic	PMC
Average BBox Recall	92.82%	86.43%
Average BBox IOU	91.88%	81.78%

3.6 Task 6

Our experiment split the training dataset to Train: Val 10:2 and used the same method in task 4 to train the model. For the box representation element, point representation element with five channels output or one channel output, we train three models independently, finally, with the methods mentioned in task 6. We get our result on the Test set without k-fold cross-validation and ensemble, As summarized in Table 8.

Table 8. Performance on the test dataset of Task 6.

	Adobe synthetic	UB PMC
Average Visual Element Detection Score	90.67%	87.00%
Average Name Score	94.84%	78.54%
Average Data Score	70.30%	55.40%
Average Metric Score	76.43%	61.18%

4 Conclusion

Chart-info is a task to test the comprehensive ability, including machine learning, image recognition, object detection, keypoint detection, OCR, and other tasks. In every task, we explored and proposed our methods and achieved relatively good performance. However, it is still hard work to understand and get insights from charts, and we need to do more work.

References

1. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
2. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: visual explanations from deep networks via gradient-based localization. In Proceedings of the IEEE International Conference on Computer Vision, pp. 618–626 (2017)
3. Deng, J., Guo, J., Niannan, X., Zafeiriou, S.: Arcface: additive angular margin loss for deep face recognition. In: CVPR (2019). 6
4. Yun, S., Han, D., Oh, S.J., Chun, S., Choe, J., Yoo, Y.: Cutmix: regularization strategy to train strong classifiers with localizable features. arXiv preprint arXiv:1905.04899 (2019)
5. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: beyond empirical risk minimization. CoRR, abs/1710.09412 (2017). 7
6. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)

7. Luo, H., Gu, Y., Liao, X., Lai, S., Jiang, W.: Bag of tricks and a strong baseline for deep person re-identification. In: CVPR (2019)
8. Kang, B., et al.: Decoupling representation and classifier for long-tailed recognition. In ICLR (2020)
9. Cai, Z., Vasconcelos, N.: Cascade R-CNN: delving into high quality object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6154–6162 (2018)
10. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition[J]. IEEE Trans. Pattern Anal. Mach. Intell. **39**(11), 2298–2304 (2016)
11. Graves, A., Fernández, S., Gomez, F., et al.: Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 369–376 (2006)
12. Xie, S., Girshick, R., Dollár, P., et al.: Aggregated residual transformations for deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1492–1500 (2017)
13. Sun, K., Xiao, B., Liu, D., et al.: Deep high-resolution representation learning for human pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5693–5703 (2019)
14. Cao, Y., Xu, J., Lin, S., et al.: Gcnet: non-local networks meet squeeze-excitation networks and beyond. In: Proceedings of the IEEE International Conference on Computer Vision Workshops (2019)
15. Poco , J., Heer, J.: Reverse-engineering visualizations: recovering visual encodings from chart images. In: Computer Graphics Forum, vol. 36, no. 3. Wiley Online Library, pp. 353–363 (2017)
16. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
17. Ke, G., Meng, Q., Finley T., et al.: Lightgbm: a highly efficient gradient boosting decision tree. In: Advances in Neural Information Processing Systems, pp. 3146–3154 (2017)
18. Zhou, X., Wang, D., Krähenbähl, P.: Objects as points. arXiv preprint arXiv:1904.07850 (2019)
19. Yu, F., Wang, D., Shelhamer, E., et al.: Deep layer aggregation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2403–2412 (2018)
20. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. ICLR (2014)
21. Dai, J.: Deformable Convolutional Networks (2017)
22. Lin, T.-Y., Goyal, P., Girshick, R., He, K., Doll′ar, P.: Focal loss for dense object detection. In: ICCV (2017)
23. Feng, Z.H., Kittler, J., Awais, M., et al.: Wing loss for robust facial landmark localisation with convolutional neural networks. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. IEEE (2018)
24. Rashid, M., Gu, X., Jae Lee, Y.: Interspecies knowledge transfer for facial keypoint detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
25. Surhone, L.M., Tennoe, M.T., Henssonow, S.F., et al.: Histogram of Oriented Gradients. Betascript Publishing **12**(4), 1368–1371 (2016)