Analysis of the MoNuSAC 2020 challenge evaluation and results: metric implementation errors

Adrien Foucart, Olivier Debeir, Christine Decaestecker

***Abstract*— The MoNuSAC 2020 challenge was hosted at the ISBI 2020 conference, where the winners were announced. Challenge organizers, in addition to the leaderboard, released the evaluation code and visualisations of the prediction masks of the "top 5" teams. This shows a very high level of transparency, and provides a unique opportunity to better understand the challenge results. Our analysis of the code and all released data, however, shows three different problems in the computation of the metric used for the official ranking: a coding mistake resulting in erroneous false positives; another resulting in missed false positives; and a problem with the metric's aggregation method. We demonstrate the errors, and confirm that the mistaken version of the code was indeed used to rank the algorithms in the challenge. Our results can be fully replicated with the code provided on GitHub.**

***Index Terms*— Digital pathology, challenge, nuclei segmentation, nuclei classification**

I. Introduction

A. Background

The MoNuSAC 2020 challenge was an official satellite event of ISBI 2020[1]. Code for reading the annotations and for computing the evaluation metric was released on GitHub, and examples of the submission file's format were provided to the challenge participants via Google Drive. The results and an analysis of the key findings were published after the challenge [1], with supplementary materials containing all the teams' submitted methods and more detailed metrics available on Google Drive. Additionally, the "color-coded ground truth masks and predictions of the top five teams" were released on the challenge website and made available alongside the challenge's training and test data. This shows a great transparency in the intentions of the organizers, who provide much more information than in many other digital pathology challenges. Unfortunately, in the case of MoNuSAC 2020, a review of the available material reveals several errors in the code for calculating the evaluation metric. The results of the challenge should therefore be re-computed and the subsequent publication revised accordingly.

A full technical description of the errors as well as all the code necessary to reproduce our results are available on GitHub: https://github.com/adfoucart/monusac-results-code-analysis

B. Overview of the dataset and predictions

The full description of the dataset can be found in the challenge publication [1]. We will focus in this section on the facts that are necessery to understand the errors and their potential impact.

The task of the challenge was to detect, segment and classify cell nuclei in histological images. The test set contained 25 images from 25 patients. For each patient's image, regions (i.e., sub-images) were extracted and their nuclei annotated. The annotations are stored as .xml files containing the vertices of polygons contouring the nuclei, and associated with one of four classes (epithelial, lymphocyte, neutrophil, macrophage). The sub-images vary largely in their sizes and numbers of objects of interest, from tiny 100x100 pixels images with one or two nuclei, up to large 1500x1500 pixels images with hundreds of objects. Participants were expected to provide for each sub-image and for each class with at least one cell nucleus in the sub-image, a separate .mat file containing the labelled objects of that class in that sub-image (the "n-ary mask"). Each .mat files therefore contains a "n-ary mask" where all the pixels of a segmentend instance are being assigned a unique positive integer, with 0 reserved for the background.

However, the predictions of the "top teams" were not released in this format. Instead, "color-coded" predictions were released, with a single RGB image per sub-image (each class being associated with a color), and borders being added to the objects to show the separation of close or overlapping nuclei. This means that, unfortunately, we do not have access to the raw .mat files with the actual labels and precise contours of each object that would allow us to fully reproduce the results of the challenge. The provided color-coded images, however, are sufficient to demonstrate the problems in the computation of the metric. In our experiments, we simply removed the borders and relabelled the objects, leading to slightly smaller objects in the prediction masks we used than in those submitted by the teams.

This paragraph of the first footnote will contain the date on which you submitted your paper for review.

A. Foucart is with the Laboratory of Image Synthesis and Analysis (LISA) at the Université Libre de Bruxelles (ULB), Brussels, Belgium (e-mail: adrien.foucart@ulb.be)

O. Debeir is with the LISA and the Center for Microscopy and Molecular Imaging (CMMI), ULB, Gosselies, Belgium.

C. Decaestecker is with the LISA and the CMMI, and Senior Research Associate with the F.N.R.S. (email: Christine.Decaestecker@ulb.be)

[1]Challenge website on grand-challenge.org

C. Metric and described evaluation method

The metric used by the challenge is based on the "Panoptic Quality" (PQ), introduced in [2], and which essentially combines a segmentation score based on the average Intersection over Union (IoU) of the true positives, and a detection score, which is simply the F1-score. The PQ is computed per-class c and per-image i as:

$$PQ_c^i = \frac{\sum_{(p,g)\in TP_c^i} IoU(p,g)}{|TP_c^i| + \frac{1}{2}|FP_c^i| + \frac{1}{2}|FN_c^i|}$$

Where (p, g) are pairs of matching objects identified from the prediction masks and ground truth masks. A "match" is defined as a pair of objects where $IoU > 0.5$.

The evaluation method described in [1] consists in:

- Computing the PQ per-class c and per-image i (PQ_c^i)
- Computing the image PQ $PQ^i = \frac{1}{C}\sum_{c=1}^{C} PQ_c^i$
- Computing the average PQ over the entire dataset $aPQ = \frac{1}{N}\sum_{i=1}^{N} PQ^i$

The text of the publication indicates that "Participants submitted a separate output file for each of the 25 test images" and "Arithmetic mean of the 25 PQ^i scores formed the final average panoptic quality". As there are 25 *patients* but more "sub-images" in the test set, this means that the "per-image" computation has to be done at the level of the *patient* (meaning that the IoUs, True Positives, False Positives and False Negatives have to be aggregated over all the "sub-images" of the patient before computing the PQ_c^i). This makes sense given the highly variable image sizes and contents, to avoid a small image with two nuclei and a large image with a hundred times more nuclei having the same contribution to the average score.

II. Error in the computation of the metric

A. Description of the problem

In the released code of the challenge, a method computes the PQ for one class of one sub-image. It takes as input the ground truth n-ary mask (i.e. the mask of labelled objects) and the predicted n-ary mask of that class. To compute the "detection" part of the metric, the number of True Positives (TP), False Positives (FP) and False Negatives (FN) need to be determined. To determine the number of FPs, a list of existing predicted objects indices is initialized. Each time a matching pair of objects is found, the index of the predicted object is removed from that list. The length of the list after going through all the matches gives the number of FPs.

The problem happens when removing the index of the predicted object. The published code removes the elements where `pred_indx_list == [indx][0]`, when it should be `pred_indx_list == matched_instances[indx][0]`. In the challenge version, `[indx][0]` will resolve to `indx`, which is the *ground truth object index*. If this particular ground truth index is not present in the predicted index list, no object will be removed, and a False Positive will be incorrectly added to the tally.

The resulting error will have no effect if and only if all the indices in `np.unique(ground_truth_image)` are present in `np.unique(predicted_image)` (as which particular index is removed from the `pred_indx_list` doesn't matter for the metric). In contrast, the effect will be particularly strong if the two lists are completely "unaligned". For instance, if the indices in the ground truth image are `[1,2,3,4]` and the indices in the predicted image are `[5,6,7,8]`, even if all the objects are perfectly matched, 4 False Positives will be counted by the provided method. In the exemple submissions provided by the challenge, the indices follow each other from class to class in the same sub-image (so that, for instance, indices of the "lymphocyte" class may start at 10 if there were 9 nuclei annotated in the previous classes), but nothing in the rules prevented the teams from starting the indices at 1 for each class, thus making the situation of non-alignment of lists (described above) quite likely for some teams.

In our code on GitHub, we use synthetic data to check that the published code behaves as we described above, and that replacing the problematic line in the code with our fix makes the problem disappear. We also test the impact of the bug using a single image from the "SJTU 426" team's prediction. We demonstrate that offsetting the label indices without changing anything else about the prediction mask leads to computed PQs ranging from 0.385 (completely unaligned indices) to 0.501 (completely aligned indices) for the Lymphocyte class.

Based on the information that can be found on the challenge's GitHub, this problem does appear in the final challenge results. Looking at the history of the `PQ_metric.ipynb` file[2], we can see a "result dump" for several of the participating teams, showing the per-image, per-class PQ computed on the entire test set. According to this result dump, the score computed for the image and class mentioned above of the SJTU 426 team was 0.381, which is very close to our "worst case scenario" score of 0.385. The difference is likely to be due to our PQ being computed based on the color-coded predictions instead of the raw .mat files provided by the participants.

B. Error in the reporting of the detailed per-organ results

Using that same "result dump", we can recompute the detailed table of results published in the supplementary materials of the post-challenge publication. The SJTU 426 is the "L2" team in the result table. Comparing the published per-organ, per-class PQs and the same metric recomputed from the result dump shows that they are identical, except that the Macrophage and Neutrophil classes are systematically inverted (see Table I). Looking at the code, it seems likely to be an error in reporting the results in the table, as throughout the code the order used for the classes puts neutrophils before macrophages, and they are also stored in that order in the .xls sheet in which the results are written by the code.

[2]PQ metric file on March 20, 2020 (around the time of the publication of the leaderboard)

TABLE I
PER-ORGAN, PER-CLASS RESULTS PUBLISHED IN THE SUPPLEMENTARY MATERIALS OF THE POST-CHALLENGE PUBLICATION, AND RECOMPUTED FROM THE RESULT DUMP, FOR THE MACROPHAGE AND NEUTROPHIL CLASSES.

Organ	Class	PQ (reported)	PQ (recomputed)
Breast	Macrophage	0.446	0.342
	Neutrophil	0.342	0.446
Kidney	Macrophage	0.441	0.465
	Neutrophil	0.465	0.441
Lung	Macrophage	0.566	0.621
	Neutrophil	0.621	0.566
Prostate	Macrophage	0.612	0.426
	Neutrophil	0.426	0.612

III. UNDETECTED FALSE POSITIVES

Another part of the code compiles the results per-class and per-(sub-)image into a .xls file. It is from that file that, presumably, the PQs were then averaged to get the final results shown in the leaderboard. For each team, the code processes by starting from a list of files taken from *the ground truth directory*. Each of these files will contain the reference n-ary mask for one class on one sub-image. Iterating through this list, the code then finds the corresponding .mat file in the team's predictions directory. The PQ is then computed based on these two n-ary masks, and added to the .xls worksheet.

The problem is that there is nothing in the provided code that checks for additional files in a *team predictions directory* without corresponding files in the *ground truth directory*. For example, if a team found Lymphocytes in a sub-image that didn't actually contain any, due to the lack of a corresponding ground truth file, this erroneously predicted "Lymphocytes" mask would never be opened and no "False Positives" would be counted.

To check if this was the case in the challenge results, we recomputed the full PQ metric of the SJTU 426 team based on the color-coded masks. Using the erroneous PQ code described in section II, we compute the sub-image PQ^i using two different strategies: either to take the average of the PQ_c^i where there is at least one instance of class c in the ground truth (which appears to be what the challenge has done), or where there is at least one instance of class c in the ground truth and/or in the predictions (accounting for all false positives and all false negatives). Our results obtained using the incorrect FP count ($PQ = 0.554$) are much closer to the published results ($PQ = 0.579$) than those obtained using the corrected FP count ($PQ = 0.424$). The remaining difference in results here are again likely to be due to the fact that our results come from the color-coded predictions, and with labels which will be different than in the team's n-ary masks, which given the PQ computation bug will impact the results as well.

Based on our analysis (see supplementary materials on GitHub), 439 nuclei detected by the SJTU 426 team are with no corresponding ground truth object but are not counted as False Positives. In fact, 66 submitted n-ary masks are ignored because of a lack of corresponding ground truth masks, and the evaluation finally takes into account only the 162 masks with ground truth annotations.

IV. PROBLEM WITH THE METRIC'S AGGREGATION METHOD

As mentioned in section I-C, the post-challenge publication makes it clear that the PQ metric should be computed per-class c and per-patient p as PQ_c^p, and not per sub-image i as PQ_c^i. This is, however, not the method that allowed us to reproduce the results in Table I. In the result dump, all the PQ provided were computed per-class, per-sub-image, and we had to average them per-organ in order to match the results of the challenge. Similarly, we can recompute the overall result for the SJTU 426 team by computing the $PQ^i = \frac{1}{C}\sum PQ_c^i$ for each sub-image, and then averaging them over the entire test set. This gives us the expected 0.579 reported in the leaderboard. Nothing in the provided code performs the necessary operation of first compiling the IoUs, TPs, FPs and FNs per-patient *before* computing the PQ_c^p. This discrepancy between the evaluation code and the published methodology is problematic, as it will harshly penalize teams which made mistakes on the smaller images compared to teams that may be worse overall, but whose mistakes were more often found in larger images.

V. CONCLUSIONS

We have shown that the results published in [1] and on the MoNuSAC 2020 leaderboard are based on an erroneous implementation of the PQ metric. More specifically, the results:

- Mistakenly detect False Positives when there are indices in the set of ground truth labels which are not present in the set of prediction labels.
- Do not count False Positives when there was no annotated object of that class in the ground truth.
- Compute the per-class "PQ_c^i" metric, and the per-image PQ^i metric on each sub-image of the test set instead of first aggregating the IOUs, TPs, FPs and FNs per-patient, as stated in the publication. This hugely penalizes teams which made a single mistake in very small images with very few cell nuclei.

This analysis also shows the importance of a transparent and collaborative process when evaluating challenge results. Most challenges do not report any details about the teams predictions, nor release their evaluation source code. The implementation of metrics, especially non-standard ones, is quite prone to errors that can be difficult to detect. To avoid such errors, evaluation codes should be published as early as possible in the challenge process, and participating teams should be encouraged to test and validate the code themselves, in order to have more confidence in the final results and rankings. If mistakes can be found in a challenge like MoNuSAC because of its transparency, one wonders what mistakes could be found in challenges that do not have this transparency.

REFERENCES

[1] R. Verma *et al.*, "Monusac2020: A multi-organ nuclei segmentation and classification challenge," *IEEE Transactions on Medical Imaging*, pp. 1–1, 2021.

[2] A. Kirillov *et al.*, "Panoptic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.