MERLIN: Identifying inaccuracies in Multiple Sequence Alignments using Object Detection

Hiba Khodji, Lucille Herbay, Pierre Collet, Julie Thompson, Anne Jeannin-Girardon

MERLIN: Identifying inaccuracies in Multiple Sequence Alignments using Object Detection

Hiba Khodji[1 [0000−0002−5525−4863]], Lucille Herbay[1 [0000−0002−4201−8837]], Pierre Collet[1 [0000−0002−5284−4702]], Julie Thompson[1 [0000−0003−4893−3478]], and Anne Jeannin-Girardon[1 [0000−0003−4691−904X]]

University of Strasbourg, ICube Laboratory UMR7357
1 Rue Eugène Boeckel, 67000 Strasbourg, France
khodji@unistra.fr , lucille.herbay@etu.unistra.fr {collet, thompson, jeanningirardon}@unistra.fr

Abstract. Multiple Sequence Alignments set the basis for many biological sequence analysis methods. However, they are susceptible to irregularities that result either from the predicted sequences or from natural biological events. In this paper, we propose MERLIN (Msa ERror Localization and IdentificatioN), an object detector that consists in identifying such irregularities using visual representations of MSAs. Our model is developed using a state-of-the-art deep learning object detector, YOLOv4, and trained on a set of MSA images from an in-house built dataset with automatically annotated errors. Our object detector exhibits a mean Average Precision of 71.18% in predicting different types of errors within MSAs. We conducted a thorough examination of the obtained results which showed that our method correctly identifies certain inconsistencies that were missed by the automatic annotation algorithm.

Keywords: Multiple Sequence Alignment · Error detection · Deep Neural Networks · Object Detection.

1 Introduction

Multiple Sequence Alignment (MSA) is a fundamental task in bioinformatics that consists in arranging a set of biological sequences (DNA, RNA or proteins) in order to identify regions of similarity that reflect their biological relationships. However, MSAs are prone to errors which can originate either from natural biological variants or from MSA-generating algorithms. Algorithms used to predict protein-coding genes in DNA sequences, for instance, are not always accurate, and often lead to sequence prediction errors. Consequently, today's protein databases are riddled with inconsistencies [15].

Identifying inaccuracies in multiple sequence alignments is not a trivial task, but accurate MSAs are instrumental in developing solutions to address problems spanning different biology related areas of study [3], such as evolutionary studies, protein structure, analysis of effects of genetic mutations, *etc.* Numerous alignment methods have been developed to tackle the MSA dilemma, including, but

not limited to, dynamic programming, progressive multiple alignment, iterative alignment, Hidden Markov Models, and Genetic Algorithms [22]. Several other techniques have been proposed in order to evaluate and identify inconsistencies within MSAs [6, 12, 24]. However, one unprecedented approach to solving this issue consists in using a deep learning *object detector*. Object Detection aims to identify and locate one or several objects in an image or video. This challenging task has been the center of increasing attention in the field of computer vision with applications across a wide and diverse spectrum of fields including traffic monitoring [13], medical image analysis [8], and biology [10]. MSAs can be visually represented in the form of a colored alignment of sequences; we thus hypothesize that the structural information encoded in MSAs can be exploited by deep learning object detection algorithms, which are praised for their ability to capture increasing scale of information and extract contextual information from data.

In this paper, we propose an approach to identify inconsistencies within MSAs of protein sequences. Our method, MERLIN (Msa ERror Localization and IdentificatioN), consists in applying a state-of-the-art deep learning object detector, YOLOv4 [2], to detect and localize different types of errors (deletions, insertions/extension, and mismatches) using visual representations of multiple sequence alignments.

This paper is organized as follows: Section 2 introduces related works. Our dataset is presented in Section 3. Section 4 describes the experimental protocol used for the proposed approach. Section 5 presents and discusses the experiment results. Finally, Section 6 summarizes the conducted work and concludes this paper.

2 Related Works

Constructing an optimal multiple alignment of a set of sequences is a computationally expensive task, which led to the development of heuristic algorithms that sometimes introduce errors into the alignments. These errors can result either from inaccurate alignment algorithms or badly predicted sequences, and consist in the presence of unusual deletions, insertions/extensions, or mismatches that are inconsistent with their local context. A deletion error refers to the absence of one or more amino acids from a sequence. An insertion is the inclusion of an additional sequence segment between two amino acids, while an extension refers to an additional sequence segment which is added on either end (N- or C-terminal) of a sequence. Finally, a mismatch is represented by non-homologous amino acids in one or more columns of the alignment.

As an attempt to investigate the errors introduced into MSAs, the authors in [4] developed EvalMSA, a software tool that allows the detection of divergent sequences and "outliers" in MSAs. The basic idea behind this technique is to evaluate the contribution of each sequence to cause gaps in the alignment using a *gappiness* value (gpp); the sequence with the highest gpp value is deemed strongly divergent and is more likely to introduce gaps in the remaining se-

quences. EvalMSA was benchmarked with six different alignments derived from the Pfam database [7]. Artificial outliers were added and the sequences were realigned using different alignment algorithms (MUSCLE and ClustalW [21], as well as Expresso algorithm [1]). EvalMSA was able to correctly identify outliers regardless of the algorithm used to generate the MSAs. The method was compared to a similar outlier detection tool, called OD-Seq, introduced in [11]. This approach utilizes a gap metric to measure the similarity of gap placement between pairs of sequences and identify sequences with missing parts compared to the rest of the alignment. In another line of research, Khenoussi et al. [12] set out to identify inaccuracies in protein sequences using MSAs. The authors developed a Bayesien model called SIBIS, which takes an MSA as input and outputs an XML file where all identified inconsistencies are highlighted. The proposed algorithm was compared to MisPred [16] and a profile-based method [23] on a set of MSAs. The obtained results showed a higher sensitivity for SIBIS (81%) compared to 27% for MisPred and 62% for the profile-based approach. In terms of specificity, SIBIS suffered a slight loss (92%) compared to the profile-based method (96%). However, the loss in specificity was considered statistically insignificant.

In this paper we attempt to examine the possibility of using a deep-learning object detector to identify inconsistencies in MSAs. Object detection algorithms are primarily divided into two categories: *two-stage* and *one-stage* detectors. As the name suggests, a two-stage detector is carried out in two stages; in the first stage, the algorithm selects Regions Of Interests (ROIs) (*i.e.* regions with a high probability of containing an object). In the second stage, the region proposals are processed for object classification and bounding box regression. One-stage detectors, on the other hand, require only a *single* pass through the network to localize and classify objects. Although two-stage detectors exhibit a good detection accuracy, they come with a large cost in terms of time and computational power. As an attempt to overcome this limitation, Redmon et al. [19] proposed YOLO (You Only Look Once), a novel deep learning detection paradigm. YOLO owes its name to its ability to process and detect objects in an image/frame in a *single* pass of the network. Therefore, YOLO is a *one-stage detector* that waives the need for a region proposal technique, thereby increasing detection speed.

Here, we are interested in YOLOv4 [2], which is an advanced version of YOLO. YOLOv4 is a one-stage detector that improves upon earlier versions by introducing new techniques that enhance the training process and detection accuracy.

3 Dataset

In order to train and evaluate our object detector, we used multiple sequence alignments from an in-house built dataset [15]. These MSAs are extracted from the Uniprot reference proteomes [5] and RefSeq [17] databases, and were automatically annotated using SIBIS algorithm [12]. As described in Section 2, three types of errors can be found in an MSA: deletions, insertions/extensions, and

mismatches. It is important to note that, since the original MSAs were not manually annotated, some alignments may contain certain errors that the annotation program failed to detect [1]. The original MSAs were converted into images using ADOMA (Alternative Display Of Multiple Alignment) [28] which is a graphical interface that offers a colored alignment output where amino acids with similar physico-chemical properties share the same color. Figure 1 shows protein sequences in the color format as produced by ADOMA. We then removed the characters from the colored alignments and modified their color code to make the sequences more contrasted. The obtained HTML files were then converted into images using an open source command line tool [27].

Fig. 1: Example of protein sequences shown in the ADOMA [28] colored format.

Once we obtained the MSA images, we designed a simple program to automatically generate the corresponding annotations. To this end, we used a parser that takes an MSA in XML format (output by SIBIS [12]) as input and outputs all existing errors and their corresponding positions. The information provided by the parser is then used to produce annotations in YOLO format.

4 Experimental Setup

The architecture in YOLOv4 consists of a backbone, neck, and head. The backbone is based on CSPDarknet53 [25]; the neck includes a Spatial Pyramid Pooling (SPP) block [9] in order to increase the receptive field and separate out the most significant context features from the backbone; the feature aggregation in the neck is carried out using PANet [14]. The head is in charge of the final detection and implements YOLOv3 [20]. The implementation of YOLOv4 uses the Darknet framework [18]. For our experiment, we split the data (12 229 images) into 80% for training, 10% for validation, and 10% for testing. Since our MSA images are of a large rectangular shape, we set the network size[2] to 1024x512. For training, we used a batch size of 64; an initial learning rate of 0.001; momentum and weight decay are set to 0.9 and 0.0005, respectively. The training was processed for 14 000 iterations [3]. The distribution of classes in the training data is as follows: Internal deletions (32.7%), N-terminal deletions (31.4%), C-terminal deletions (9.57%), Internal insertions (9.07%), N-terminal extensions

[1] Since SIBIS sensitivity is "only" 81% (v.s. 92% specificity) it means that it is more prone to False Negatives.

[2] Images are resized to the network size during training and detection.

[3] Also known as max batches. It is defined as the number of classes * 2000 by Bochkovskiy et al. in [2]

(7.10%), Mismatches (6.76%), C-terminal extensions (3.40%). This distribution corresponds to the actual proportions of these error types across MSAs in the primate proteomes databank [15]: Internal deletions (36.9%), N-terminal deletions (29.34%), C-terminal deletions (8.87%), Internal insertions (8.82%), N-terminal extensions (6.43%), Mismatches (6.63%), C-terminal extensions (3.01%).

5 Results and Discussion

The performance of our object detector was evaluated using a set of different metrics as shown in Table 1. The mAP refers to the mean Average Precision, and it is a commonly used evaluation metric for object detection which measures model performance in terms of both classification and localization ability. In order to understand the mAP, we first need to define the Intersection over Union (IoU) and Average Precision (AP). The IoU is an evaluation metric that measures the overlap between a predicted bounding box and the ground truth box in an image; the larger the area of overlap the higher the IoU score. Given an IoU threshold $\alpha = 0.5$, the model determines whether a detection is a True Positive (IoU $\geq \alpha$), or a False Positive (IoU $< \alpha$); a False Negative is when the model fails to detect an object in the image. Based on these elements, the model computes the precision and recall. Precision quantifies the number of true positives over all positive predictions, while recall measures the number of true positives among all correct predictions. The Average Precision (AP) is then computed by finding the area under the the precision-recall curve. While AP is calculated for each class, the mAP is defined as the mean of APs across all classes.

Table 1: Evaluation results on the test set (containing a total of 1 223 MSAs). MERLIN achieved an accuracy of 71.18% on the test data.

Metric	**Value**	**Average precision (per class)**
mAP	71.18%	Internal deletion, AP = 75.85%
Recall	78%	N-terminal deletion, AP = 93.91%
Precision	72%	C-terminal deletion, AP = 85.83%
F1-score	75%	Internal insertion, AP = 76.57%
Average IoU	67.73%	N-terminal extension, AP = 88.02%
True Positives	2 667	C-terminal extension, AP = 66.80%
False Positives	1 014	Internal mismatch, AP = 11.32%
False Negatives	748	

Every 1000 iterations, the mAP is calculated on the validation set and the corresponding weights are saved in order to find the *best* weights with maximum accuracy. The optimal model weights yielded a validation mAP of 72.89%. We evaluated the performance of our object detector on the test set and reported the results in Table 1. With a default IoU threshold of 50%, our model reached a test accuracy of 71.18% with an average IoU of 67.73%. The results also show

that the model correctly identified 78% of all actual positives with a precision of 72%. In Table 1, the average precision (AP) evaluates the performance of the model for each class. To better interpret the obtained results, we compared the number of all annotated errors (*i.e.* errors identified by the Bayesian model SIBIS [12]) in the test set to the number of detected errors by our object detector for each error type. The results are reported in Table 2. Internal deletions are the most present in the test set with a total of 1 132 (representing 33.15% of the test set). The model detected 84.45% of correct internal deletions with an average precision of 75.85%. C-terminal extension errors are the least present in the test set with 92 occurrences (representing only 2.7% of the test set): 66% of these errors were correctly predicted with an AP of 66.80%. N-terminal deletions are the most accurately predicted errors with an AP of 93.91%, while internal mismatches were found to be the most difficult to detect with a low AP of 11.32%. This could be intuitively explained by the following: while deletions, insertions, and extensions alter the structural shape of an MSA, by introducing gaps or additional sequence segments, mismatches are represented by a succession of odd residues (colors) introduced into one or more columns, which makes this particular sequence inconsistency relatively more subtle and difficult to discern. This could also be due to the scarcity of training samples for this class which makes it difficult for the model to learn underlying characteristics of mismatch errors.

Table 2: Comparison results. We compared the number of all annotated errors by SIBIS [12] in the test set to the number of errors detected by our proposed object detector.

	Number of identified errors by SIBIS [12]	Number of identified errors by YOLOv4 [2]
Internal deletion	1 132	1 305 (TP = 956, FP = 349)
N-terminal deletion	1 033	1 139 (TP = 953, FP = 186)
C-terminal deletion	320	342 (TP = 260, FP = 82)
Internal insertion	282	325 (TP = 211, FP = 114)
N-terminal extension	203	253 (TP = 178, FP = 75)
C-terminal extension	92	96 (TP = 61, FP = 35)
Internal mismatch segment	224	221 (TP = 48, FP = 173)
Total	3 415	3 681

5.1 Detecting inconsistencies in MSAs: Qualitative Analysis

We examined the extent to which our model, MERLIN, is able to make good predictions on randomly selected MSAs from the test set: these MSAs contain different types and number of errors. MERLIN was also tested on MSAs that were annotated as error-free by SIBIS. Table 3 shows error detection results of our detector on randomly selected MSAs. The MSA identified as Q9UEE5

contains three different errors: N-terminal deletion, N-terminal extension, and internal deletion. All three errors were correctly detected by the model with high confidence scores[4]: 99%, 98%, and 93%, respectively. The second example, MSA Q9UHL4, contains an internal insertion error and a mismatch. While the model correctly identified both errors, the mismatch error has a low confidence score (40%) compared to the internal insertion (92%).

Table 3: Examples of prediction results on the test set. Sample MSAs with different inconsistencies correctly predicted by MERLIN.

MSA identifier	**Prediction: confidence score %**
Q9UEE5	N-terminal deletion: 99% N-terminal extension: 98% Internal deletion: 93%
Q9UHL4	Internal insertion: 92%; Internal mismatch: 40%

As stated in Section 3, the Bayesian model, SIBIS, used for data annotation is not 100% accurate; in terms of error detection, SIBIS exhibited a higher specificity of 92% with a sensitivity of 81%. Taking this statement into account, it is possible that certain MSAs contain at least one error that SIBIS has failed to detect. Moreover, it is important to note that MSAs may contain certain gaps as well as sequence segments that are structurally similar to deletions, extensions or insertion errors; however, these segments are simple shape variations in the MSA and do not constitute inconsistencies. In order to assess the ability of our object detector to distinguish between these variations and erroneous segments we used our model to identify errors on randomly selected examples amongst MSAs containing such characteristics. Detection results are shown in Figure 2: Q9NZI2 is an MSA with two N-terminal deletions correctly identified by the model. While the first deletion is a true positive with a high confidence score (98%), the second is a false positive with a lower confidence of 46%; however, a thorough manual examination of the MSA confirmed the presence of the second deletion, which was overlooked by SIBIS. We also observe that the MSA contains inserted sequence segments that were not misidentified by the model as inconsistencies. In Figure 2b, we present an MSA with a deletion error and two extended sequence segments. While our object detector successfully predicts the error with a confidence of 67%, it does not misidentify the extensions as erroneous. This particular MSA was manually found to contain a mismatch error that was overlooked by both SIBIS and our object detector.

More detection results on randomly selected MSAs are shown in Table 4. MSA Q9UBP4 has one extension error and three deletions. All detected errors are true positives identified by the object detector with high confidence scores

[4] Class-specific scores which encode the probability of a class appearing in the predicted box and how well the box fits the object.

(a) Alignment Q9NZI2

(b) Alignment Q9P2W7

Fig. 2: Detection results on the test set. Ability of our object detector to distinguish between sequence inconsistencies and non-erroneous sequence segments. (the annotation "SEQ_ERRORN2" refers to an N-terminal deletion).

(98% – 100%). It should be noted that the MSA contains two insertions that were not annotated as errors by SIBIS nor detected by our method. To determine whether these sequence segments constitute sequence inconsistencies, more information about the corresponding genomic sequences and the exon/intron structure is required. Three N-terminal deletions were found in the second example (MSA Q9UK12). While the first deletion is the only true positive prediction with a high confidence of 86%, a close inspection of the MSA revealed that the remaining deletions are true errors that the annotation tool has failed to identify. We also note that the alignment includes two deletions that were left undetected. The model made two correct predictions on MSA Q9UBK7: an extension error with a confidence score of 94%, and an internal insertion error with an equally high detection score of 95%. The third prediction corresponds to an internal deletion with a confidence of 39% that was not annotated by SIBIS, however a manual examination of the MSA suggested that this may be a correct positive prediction. The same observation can be made for MSA Q9Y664, where the model correctly identified three different inconsistencies: a deletion with a score of 48%, an extension with a score of 93%, and an insertion with a score of 34%. However, while the deletion and extension errors are true positives, the internal insertion is a manually confirmed error that was missed by SIBIS. In the last alignment, Q9NYB5, two mismatches were incorrectly detected by our model with low confidence scores.

As mentioned above, SIBIS is an automatic algorithm that sometimes fails to identify errors within MSAs. Therefore, it is reasonable to assume that some MSAs, containing at least one error, were dismissed as error-free. In order to validate this claim, we used MERLIN on three randomly selected alignments from a set of MSAs initially dismissed as containing no known inconsistencies by SIBIS. Prediction results are reported in Table 5. In the first example, MSA Q8WTV1, the model identified a deletion error with a high confidence (84%), while in the second MSA, an extension error was detected with a relatively low

Table 4: Additional detection results on the test set.

MSA identifier	**Prediction: confidence score %**
Q9UBP4	N-terminal extension: 99% N-terminal deletion: 100% N-terminal deletion: 98% N-terminal deletion: 100%
Q9UK12	N-terminal deletion: 86% N-terminal deletion: 78% N-terminal deletion: 78%
Q9UBK7	N-terminal extension: 94% Internal insertion: 95% Internal deletion: 39%
Q9Y664	C-terminal deletion: 48% C-terminal extension: 93% Internal insertion: 34%
Q9NYB5	Internal mismatch: 27% Internal mismatch: 33%

confidence of 48%. In the third example, the model identified an insertion with an even lower confidence of 26%. All the identified inaccuracies were manually confirmed by human expertise.

Table 5: Prediction results on MSAs initially annotated as error-free. MERLIN successfully identifies different types of errors that were overlooked by SIBIS [12]. These predictions were manually confirmed by human expertise.

MSA identifier	**Prediction: confidence score %**
Q8WTV1	N-terminal deletion: 84%
Q8TEV8	C-terminal extension: 48%
Q9GZW5	Internal insertion: 26%

It is important to note that, regardless of the detection confidence scores, the false positive predictions obtained by the detector warranted a manual re-evaluation of the MSAs which confirmed the accuracy of these predictions. These inconsistencies were not identified by SIBIS [12] in the original alignments [15], and therefore not annotated in our dataset. Thus, it is safe to assume that the number of true positives identified by our object detector is probably higher than 78.10%. Considering all of the above, our trained object detector is a promising tool for the quality assessment of multiple sequence alignments.

5.2 Performance Comparison

We evaluated the performance of our object detector against OD-Seq [11] and EvalMSA [4] on the test set. As described in Section 2, these tools rely on a gap

metric to identify outlier sequences in a given MSA; that is, divergent sequences that could harm the accuracy of an alignment. Our proposed approach, on the other hand, is designed to detect different inconsistencies within each individual sequence in an MSA. We conducted our comparison based on the number of outlier sequences identified by OD-Seq and EvalMSA, and all erroneous sequences found by MERLIN. Our detector yielded a precision of 83% compared to 77% by OD-Seq and 64% by EvalMSA; in terms of recall, our approach achieved 89% on the test set, outperforming both OD-Seq (29%) and EvalMSA (51%).

5.3 Interpretable predictions

Our proposed object detector proved effective in identifying inconsistencies within MSAs to assess their accuracy. However, the obtained predictions by the model, while valuable, are limited to the visual representation of the MSAs and cannot be directly exploited by bioinformaticians. To this end, we designed a program to parse all predicted errors for a given MSA into a more accessible format. The program takes YOLO predictions as input and outputs a text file which contains all identified inconsistencies with their corresponding position coordinates (*i.e.* the affected sequence, and start and end columns of the detected error). This information allows researchers to analyze an MSA with commonly used MSA visualization tools such as Jalview [26].

6 Conclusion

In this paper, we investigated a new approach by using a state-of-the-art object detector [2] to identify inconsistencies on visual representations of Multiple Sequence Alignments. We proposed MERLIN (Msa ERror Localization and IdentificatioN), an object detection model to detect and localize different types of errors (deletions, insertions/extensions, mismatches) within MSAs. Our model yielded an accuracy of 71.18% in error detection and achieved better precision (83%) and recall (89%), compared to gap penalty-based techniques, in identifying outlier sequences in MSAs. A qualitative analysis showed that MERLIN can also differentiate between sequence segments structurally similar to alignment errors and actual inconsistencies, and that it can identify and localize errors that were dismissed by annotation tools.

These encouraging results suggest that such an approach is suitable to localize and identify different types of errors in MSAs and, overall, evaluate their quality. This could prompt further research in this field in an effort to provide automated assistance to bioinformaticians working with Multiple Sequence Alignments.

Acknowledgements The authors would like to thank the BiGEst bioinformatics platform for technical support. This work was supported by the French Infrastructure Institut Français de Bioinformatique (IFB) ANR-11-INBS-0013, ANR ArtIC ANR-20-THIA-0006 and Institute funds from the French Centre National de la Recherche Scientifique and the University of Strasbourg.

References

1. Armougom, F., Moretti, S., Poirot, O., Audic, S., Dumas, P., Schaeli, B., Keduas, V., Notredame, C.: Expresso: Automatic incorporation of structural information in multiple sequence alignments using 3d-coffee. Nucleic acids research **34**, W604–8 (08 2006). https://doi.org/10.1093/nar/gkl092
2. Bochkovskiy, A., Wang, C., Liao, H.M.: YOLOv4: Optimal Speed and Accuracy of Object Detection. CoRR **abs/2004.10934** (2020), https://arxiv.org/abs/2004.10934
3. Chatzou, M., Magis, C., Chang, J.M., Kemena, C., Bussotti, G., Erb, I., Notredame, C.: Multiple sequence alignment modeling: methods and applications. Briefings in bioinformatics **2015** (11 2015). https://doi.org/10.1093/bib/bbv099
4. Chiner-Oms, A., González-Candelas, F.: Evalmsa: A program to evaluate multiple sequence alignments and detect outliers. Evolutionary Bioinformatics **12**, EBO.S40583 (2016). https://doi.org/10.4137/EBO.S40583, https://doi.org/10.4137/EBO.S40583, pMID: 27920488
5. Consortium, T.U.: UniProt: a worldwide hub of protein knowledge. Nucleic Acids Research **47**(D1), D506–D515 (11 2018). https://doi.org/10.1093/nar/gky1049, https://doi.org/10.1093/nar/gky1049
6. Dragan, M.A., Moghul, I., Priyam, A., Bustos, C., Wurm, Y.: Genevalidator: Identify problems with protein-coding gene predictions. Bioinformatics **32** (01 2016). https://doi.org/10.1093/bioinformatics/btw015
7. Finn, R.D., Bateman, A., Clements, J., Coggill, P., Eberhardt, R.Y., Eddy, S.R., Heger, A., Hetherington, K., Holm, L., Mistry, J., Sonnhammer, E.L.L., Tate, J., Punta, M.: Pfam: the protein families database. Nucleic Acids Research **42**(D1), D222–D230 (11 2014). https://doi.org/10.1093/nar/gkt1223, https://doi.org/10.1093/nar/gkt1223
8. Gao, Y., Zhang, Z.D., Li, S., Guo, Y.T., Wu, Q.Y., Liu, S.H., Yang, S.J., Ding, L., Zhao, B.C., Li, S., Lu, Y.: Deep neural network-assisted computed tomography diagnosis of metastatic lymph nodes from gastric cancer. Chinese Medical Journal **132** (2019). https://doi.org/10.1097/CM9.0000000000000532
9. He, K., Zhang, X., Ren, S., Sun, J.: Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. IEEE transactions on pattern analysis and machine intelligence **37 9**, 1904–16 (2015)
10. Hung, J., Goodman, A., Ravel, D., Lopes, S., Rangel, G., Nery, O., Malleret, B., Nosten, F., Lacerda, M., Ferreira, M., Renia, L., Duraisingh, M., Costa, F., Marti, M., Carpenter, A.: Keras R-CNN: library for cell detection in biological images using deep neural networks. BMC Bioinformatics **21** (07 2020). https://doi.org/10.1186/s12859-020-03635-x
11. Jehl, P., Sievers, F., Higgins, D.: Od-seq: Outlier detection in multiple sequence alignments. BMC bioinformatics **16**, 269 (08 2015). https://doi.org/10.1186/s12859-015-0702-1
12. Khenoussi, W., Vanhoutreve, R., Poch, O., Thompson, J.: Sibis: A bayesian model for inconsistent protein sequence estimation. Bioinformatics (Oxford, England) **30** (05 2014). https://doi.org/10.1093/bioinformatics/btu329
13. Komasilovs, V., Zacepins, A., Kviesis, A., Estevez, C.: Traffic Monitoring using an Object Detection Framework with Limited Dataset. In: VEHITS (2019)
14. Liu, S., Qi, L., Qin, H., Shi, J., Jia, J.: Path Aggregation Network for Instance Segmentation. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition pp. 8759–8768 (2018)

15. Meyer, C., Scalzitti, N., Jeannin-Girardon, A., Collet, P., Poch, O., Thompson, J.D.: Understanding the causes of errors in eukaryotic protein-coding gene prediction: a case study of primate proteomes. BMC Bioinformatics **21** (2020)
16. Nagy, A., Patthy, L.: Mispred: A resource for identification of erroneous protein sequences in public databases. Database : the journal of biological databases and curation **2013**, bat053 (01 2013). https://doi.org/10.1093/database/bat053
17. O'Leary, N.A., Wright, M.W., Brister, J.R., Ciufo, S., Haddad, D., McVeigh, R., Rajput, B., Robbertse, B., Smith-White, B., Ako-Adjei, D., Astashyn, A., Badretdin, A., Bao, Y., Blinkova, O., Brover, V., Chetvernin, V., Choi, J., Cox, E., Ermolaeva, O., Farrell, C.M., Goldfarb, T., Gupta, T., Haft, D., Hatcher, E., Hlavina, W., Joardar, V.S., Kodali, V.K., Li, W., Maglott, D., Masterson, P., McGarvey, K.M., Murphy, M.R., O'Neill, K., Pujar, S., Rangwala, S.H., Rausch, D., Riddick, L.D., Schoch, C., Shkeda, A., Storz, S.S., Sun, H., Thibaud-Nissen, F., Tolstoy, I., Tully, R.E., Vatsan, A.R., Wallin, C., Webb, D., Wu, W., Landrum, M.J., Kimchi, A., Tatusova, T., DiCuccio, M., Kitts, P., Murphy, T.D., Pruitt, K.D.: Reference sequence (RefSeq) database at NCBI: current status, taxonomic expansion, and functional annotation. Nucleic Acids Research **44**(D1), D733–D745 (11 2015). https://doi.org/10.1093/nar/gkv1189, https://doi.org/10.1093/nar/gkv1189
18. Redmon, J.: Darknet: Open source neural networks in c. http://pjreddie.com/darknet/ (2013–2016)
19. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 779–788 (2016). https://doi.org/10.1109/CVPR.2016.91
20. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. CoRR **abs/1804.02767** (2018), http://arxiv.org/abs/1804.02767
21. Tamura, K., Stecher, G., Peterson, D., Filipski, A., Kumar, S.: Mega6: Molecular evolutionary genetics analysis version 6.0. Molecular biology and evolution **30** (10 2013). https://doi.org/10.1093/molbev/mst197
22. Thompson, J.D.: Statistics for bioinformatics : methods for multiple sequence alignment. iSTE Press (2016)
23. Thompson, J.D., Linard, B., Lecompte, O., Poch, O.: A comprehensive benchmark study of multiple sequence alignment methods: Current challenges and future perspectives. PLoS ONE **6** (2011)
24. Vanhoutreve, R., Kress, A., Legrand, B., Gass, H., Poch, O., Thompson, J.: Leonbis: Multiple alignment evaluation of sequence neighbours using a bayesian inference system. BMC Bioinformatics **17** (07 2016). https://doi.org/10.1186/s12859-016-1146-y
25. Wang, C.Y., Liao, H.Y.M., Yeh, I.H., Wu, Y.H., Chen, P.Y., Hsieh, J.W.: Cspnet: A new backbone that can enhance learning capability of cnn. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) pp. 1571–1580 (2020)
26. Waterhouse, A.M., Procter, J.B., Martin, D.M.A., Clamp, M., Barton, G.J.: Jalview Version 2—a multiple sequence alignment editor and analysis workbench. Bioinformatics **25**(9), 1189–1191 (01 2009). https://doi.org/10.1093/bioinformatics/btp033, https://doi.org/10.1093/bioinformatics/btp033
27. wkhtmltopdf. https://wkhtmltopdf.org
28. Zaal, D., Nota, B.: Adoma: A command line tool to modify clustalw multiple alignment output. Molecular Informatics **35** (08 2015). https://doi.org/10.1002/minf.201500083