Object Detection as Probabilistic Set Prediction - Supplementary Material

Georg Hess[1,2], Christoffer Petersson[1,2], and Lennart Svensson[1]

[1] Chalmers University of Technology, Gothenburg, Sweden georghe@chalmers.se
[2] Zenseact, Gothenburg, Sweden

A Cost Matrix

Suppose that we have a PMB density with Poisson intensity $\lambda(\cdot)$ and with m Bernoulli components where the i-th Bernoulli component has probability of existence r_i and existence-conditioned object density $p_i(\cdot)$. Note that we can model an MB as a PMB with Poisson intensity $\lambda(\cdot) = 0$, which means that it is enough to describe how to handle PMB densities. To evaluate the multi-object density of a PMB, we have to calculate all possible assignments, which can be computationally intractable when working with many elements. However, we can approximate the PMB likelihood by only considering the assignments with the highest likelihood. This section shows how to find these assignments by solving an optimal assignment problem and how to select the corresponding cost matrix.

Before formulating the optimal assignment problem, we remind ourselves of the problem setting. For an object with state $y_j \in \mathbb{Y} = \{y_1, \dots, y_n\}$, the single object likelihood is proportional to $\lambda(y_j)$ if it is associated to the PPP and proportional to $r_i p_i(y_j)$ if it is associated to the i-th Bernoulli component. If this Bernoulli component is not associated to any object states, then the likelihood is $1 - r_i$.

Next, to formulate the optimization problem of finding the assignment with highest likelihood we introduce an association variable. Define a surjective association $\theta : \{1, \dots, n\} \to \{0, 1, \dots, m\}$ such that $\theta(i) = \theta(j) \in \{1, \dots, m\}$ if and only if $i = j$. If $\theta(j) = 0$, object y_j is associated to the PPP, while $\theta(j) = i > 0$ means that object state y_j is associated to the i-th Bernoulli component. Further, let Θ be the set of all such θ. Then, the PMB likelihood for the set of objects $\mathbb{Y}$ can be expressed as

$$\begin{aligned} f_{\text{PMB}}(\mathbb{Y}) &= \sum_{\theta \in \Theta} \prod_{j:\theta(j)>0} r_{\theta(j)} p_{\theta(j)}(y_j) \prod_{i:\nexists \theta(j)=i \forall j} 1 - r_i \prod_{j:\theta(j)=0} \lambda(y_j) \exp\left(-\bar{\lambda}\right), \\ &\propto \sum_{\theta \in \Theta} \prod_{j:\theta(j)>0} r_{\theta(j)} p_{\theta(j)}(y_j) \prod_{i:\nexists \theta(j)=i \forall j} 1 - r_i \prod_{j:\theta(j)=0} \lambda(y_j). \end{aligned} \tag{1}$$

When searching for the most likely associations θ, we disregard $\exp\left(-\bar{\lambda}\right) = \exp\left(-\int \lambda(y')dy'\right)$ since it is independent of θ.

We note that (1) captures the association of every Bernoulli component. If the i-th Bernoulli component does not appear in the second product in (1), then it must appear in the first product in (1). We also note that the factor $\prod_{i=1}^{m}(1-r_i)$ is independent of the association θ. This means that dividing (1) by $\prod_{i=1}^{m}(1-r_i)$ yields

$$f_{\text{PMB}}(\mathbb{Y}) \propto \sum_{\theta\in\Theta} \prod_{j:\theta(j)>0} \frac{r_{\theta(j)}p_{\theta(j)}(y_j)}{1-r_{\theta(j)}} \prod_{j:\theta(j)=0} \lambda(y_j), \tag{2}$$

and the association that maximizes the PMB likelihood can be found as

$$\theta^* = \arg\max_{\theta} \prod_{j:\theta(j)>0} \frac{r_{\theta(j)}p_{\theta(j)}(y_j)}{1-r_{\theta(j)}} \prod_{j:\theta(j)=0} \lambda(y_j). \tag{3}$$

Equivalently, we can search for the association that minimises the negative logarithm of (3), which gives us

$$\theta^* = \arg\min_{\theta} -\sum_{j:\theta(j)>0} \log \frac{r_{\theta(j)}p_{\theta(j)}(y_j)}{1-r_{\theta(j)}} - \sum_{j:\theta(j)=0} \log \lambda(y_j). \tag{4}$$

In order to formulate this minimization as an optimal assignment problem, we make a slight change in notation. Each association map θ can be represented by a $(m+n)\times n$ assignment matrix A consisting of 0s or 1s. There is a bijective mapping between θ and A: for $i=1,\ldots,m$, $j=1,\ldots,n$, $A_{i,j}=1$ if and only if $\theta(j)=i$. For $i=m+j$, $j=1,\ldots,n$, $A_{i,j}=1$ if and only if $\theta(j)=0$, whereas $A_{i,j}$ is always 0 when $i>m$ and $i\neq m+j$. The assignment matrix hence must satisfy the constraints $\sum_i A_{i,j}=1$, $\forall j$, and $\sum_j A_{i,j}\leq 1$, $\forall i$.

The cost matrix of the corresponding assignment matrix A is the $(m+n)\times n$ matrix C where

$$C_{i,j} = -\log \frac{r_i p_i(y_j)}{1-r_i},\ i=1,\ldots,m,\ j=1,\ldots,n, \tag{5}$$

and where the lower part of C is a "diagonal" matrix such that $C_{m+j,j} = -\log\lambda(y_j)$, $j=1,\ldots,n$, and entries not on the diagonal are ∞. The cost of assignment matrix A is then given by the Frobenius inner product

$$\text{trace}(A^T C) = \sum_{i=1}^{m+n}\sum_{j=1}^{n} C_{i,j}A_{i,j}, \tag{6}$$

and the problem of finding the optimal assignment A^* becomes

$$A^* = \arg\min_{A} \sum_{i=1}^{m+n} \sum_{j=1}^{n} C_{i,j} A_{i,j} \tag{7a}$$

$$\text{s.t.} \quad \sum_{i=1}^{m+n} A_{i,j} = 1, \sum_{j=1}^{n} A_{i,j} \leq 1, \tag{7b}$$

$$C_{i,j} = \begin{cases} -\log\left(\frac{p_i(y_j)}{1-r_i} r_i\right) & \text{if } i \leq m, \\ -\log \lambda(y_j) & \text{if } i = m + j, \\ \infty & \text{otherwise.} \end{cases} \tag{7c}$$

B Experimental Details

B.1 Model Implementation

For implementing the DETR, RetinaNet and Faster-RCNN models, we used the probabilistic extension [2] of the Detectron2 [3] object detection framework. In that framework, models are trained to predict the covariance matrix $\boldsymbol{\Sigma}_b$ for the corresponding bounding box b. Specifically, models output the parameters of a lower triangular matrix $\boldsymbol{L}$ of the Cholesky decomposition $\boldsymbol{\Sigma}_b = \boldsymbol{L}\boldsymbol{L}^T$. While originally trained with a Gaussian distribution, we found that using an independent Laplace distribution for each parameter in b yielded better results. Using the diagonal elements of $\boldsymbol{L}$ as $[\sigma_1, \sigma_2, \sigma_3, \sigma_4]$, we find the scale of each Laplace distribution as $s_i = \sigma_i/\sqrt{2}$. The choice of a diagonal matrix is partially motivated by the evaluations in [2]. While their study was limited to Gaussian distributions, they found that diagonal covariance matrices perform on par with, or better than, their full equivalent.

B.2 Training Details

Fine-tuning toward MB-NLL was done given the pre-trained weights in [2][3]. For DETR and Faster-RCNN, models trained with ES were used as a starting point, while the model trained with NLL was used for RetinaNet. Faster-RCNN and RetinaNet were fine-tuned for 135,000 gradient steps, where the learning rate was dropped by a factor of 10 at 105,000 iterations, and again at 125,000 iterations. The initial learning rate was set to 0.001 for RetinaNet and 0.0025 for Faster-RCNN. DETR was also fine-tuned for 135,000 iterations, but with an initial learning rate of $5 \cdot 10^{-5}$ and with learning rate drops at 60,000 and 100,000 iterations. Otherwise, no changes to hyperparameters from the standard Detectron2 framework were done.

As both RetinaNet and Faster-RCNN rely on non-maximum suppression to remove duplicate detections, we applied NMS when training with the MB-NLL

[3] `https://github.com/asharakeh/probdet`

loss. Here, we used the standard IoU threshold of 0.5 and used the top 100 detections. For DETR, this was not necessary since it predicts the set of objects directly.

B.3 Inference Details

Following the COCO standard, detectors are limited to 100 predictions per image. We do not apply any confidence thresholding, but for RetinaNet and Faster-RCNN the 100 predictions with highest existence probability after NMS are used. For DETR, no selection is needed as the model only produces 100 predictions per image.

C Additional Results

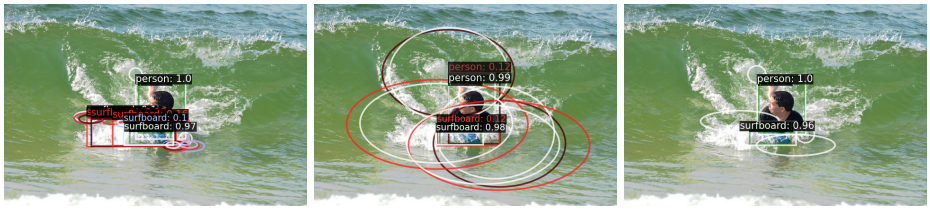
C.1 Qualitative Results

In addition to the example detections shown in Section 4, we provide further examples for DETR, RetinaNet and Faster-RCNN in figures 1, 2 and 3. We can identify similar trends in these examples as in the ones described in our results. First, Fig. 3b shows additional examples where the assignment is ambiguous. There, the cat predictions for Faster-RCNN trained with MB-NLL either have large regression or classification errors, depending on which one of them is assigned to the true object. Second, we see that MB-NLL reduces the number of confident false detections across all models. When comparing the models trained with ES and MB-NLL, the reduction of false detections can also be interpreted as a different representation of spatial uncertainty. In both Fig. 1a and Fig. 2a, there is uncertainty in where the surfboard ends on the left side. The MB-NLL models have a single prediction with larger regression uncertainty, while the ES models have many detections, each with relatively small spatial uncertainty. Further, the MB-NLL loss is normalized with the number of predictions during training.

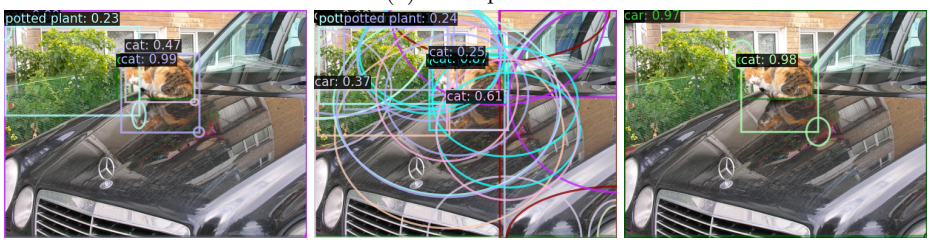
C.2 PMB-NLL Decomposition

To complement the PMB-NLL decomposition in Table 2, we also provide corresponding histograms for DETR, RetinaNet and Faster-RCNN in figures 4, 5 and 6. The histograms show how predictions contribute to the overall PMB-NLL in the assignment with the highest likelihood. Further, for matched predictions, histograms are also decomposed based on the size of the true object. Here, we follow the COCO standard for defining small, medium and large objects. For the regression of matched Bernoullis and PPP, the values have been limited to 40 for enhanced visualizations. For the classification of matched Bernoullis, the upper limit is set to 3.

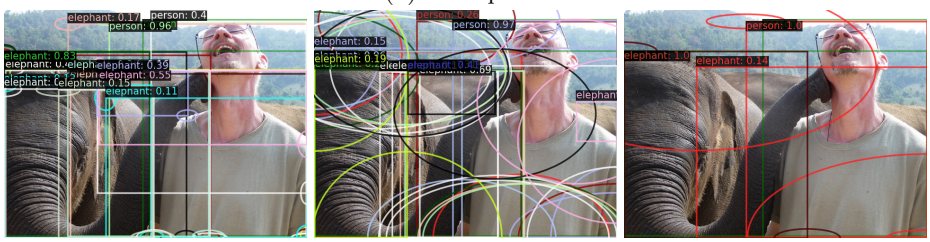
Generally, the models are worse at detecting small objects, which is shown by them being assigned to the PPP more often than medium or large objects.



(a) Example 1.

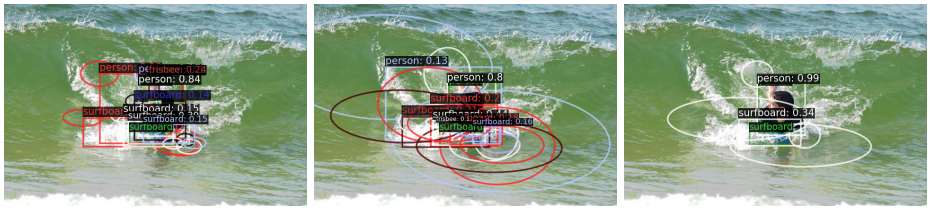


(b) Example 2.



(c) Example 3.

Fig. 1: Examples from COCO validation data with predictions made by DETR detectors trained with ES (left), NLL (middle), and MB-NLL (right). True objects are shown in green and without confidence values. Predictions with $r < 0.1$ are not shown.

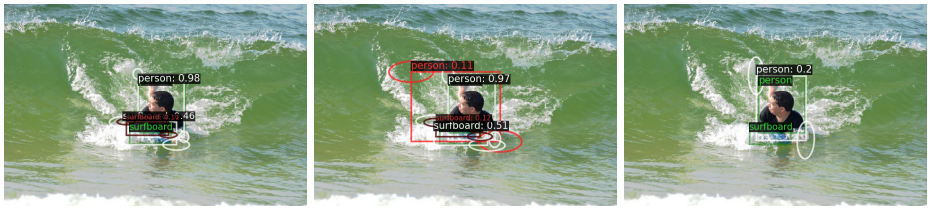


(a) Example 1.

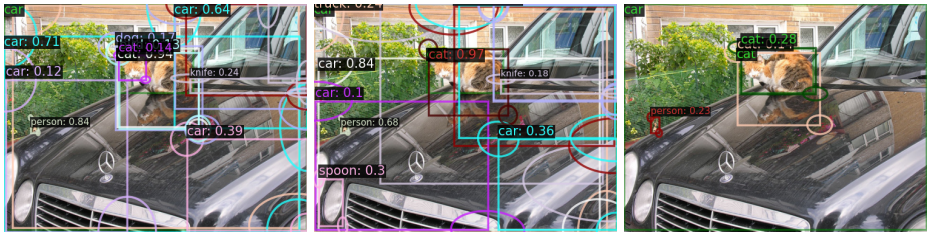
(b) Example 2.

(c) Example 3.

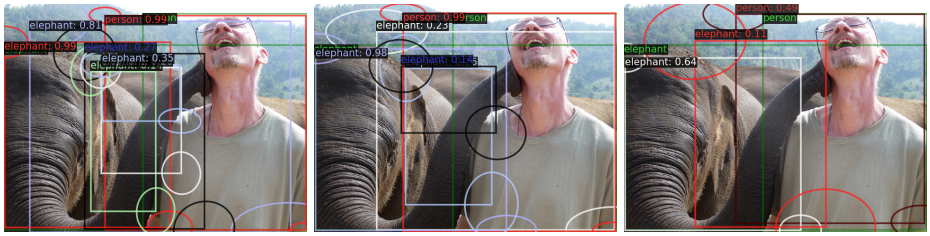
Fig. 2: Examples from COCO validation data with predictions made by RetinaNet detectors trained with ES (left), NLL (middle), and MB-NLL (right). True objects are shown in green and without confidence values. Predictions with $r < 0.1$ are not shown.



(a) Example 1.



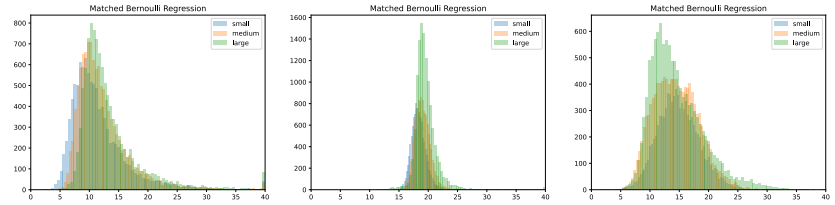
(b) Example 2.



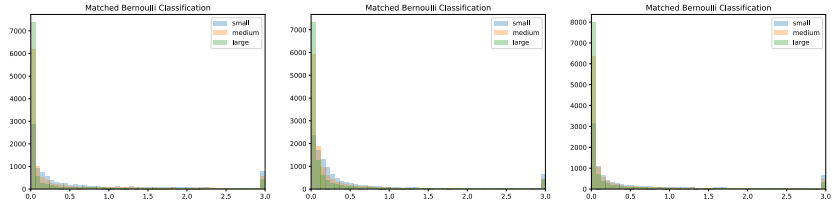
(c) Example 3.

Fig. 3: Examples from COCO validation data with predictions made by Faster-RCNN detectors trained with ES (left), NLL (middle), and MB-NLL (right). True objects are shown in green and without confidence values. Predictions with $r < 0.1$ are not shown.

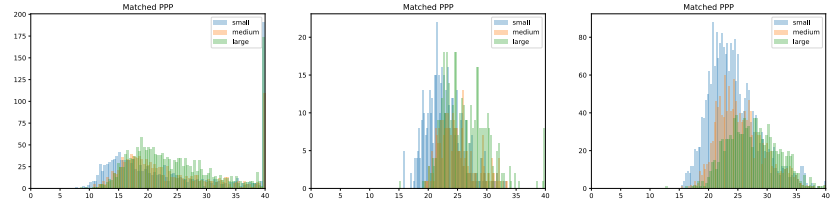
Further, the detectors are more confident when predicting larger objects, as can be seen from the classification histograms over matched Bernoullis. This is of course expected as large objects are inherently easier to detect. Lastly, we can observe that the histograms for models trained with ES or NLL tend to have more outliers, i.e., predictions whose values have been clipped to the visualization limits.



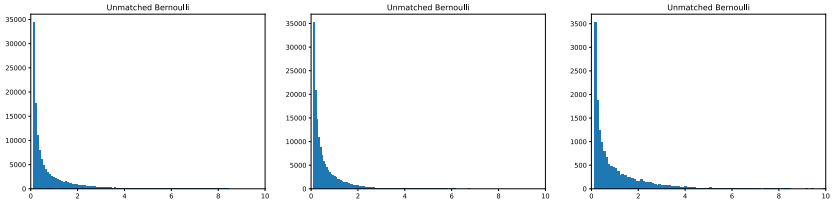
(a) Matched Bernoullis regression.



(b) Matched Bernoullis classification.

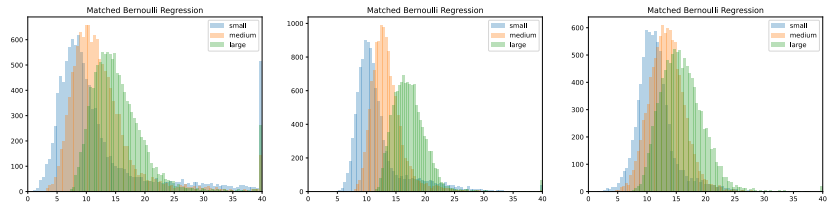


(c) Matched PPP.

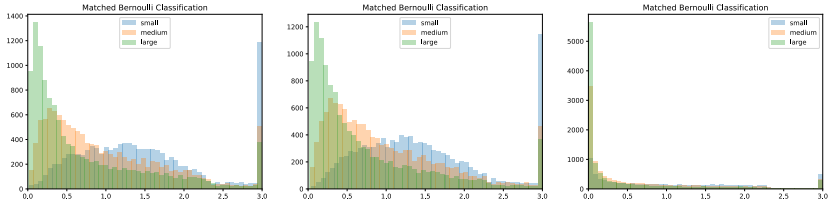


(d) Unmatched Bernoullis.

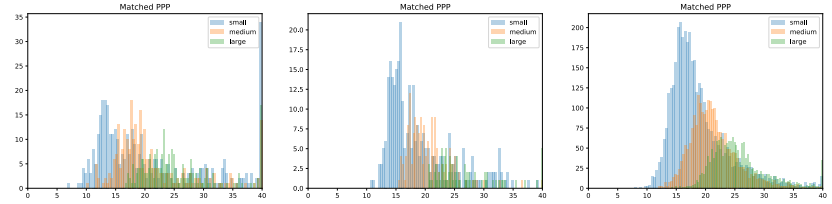
Fig. 4: Histograms over PMB-NLL decomposition for DETR trained with different loss functions: ES (left), NLL (middle), and MB-NLL (right). Note varying y-axes across models.



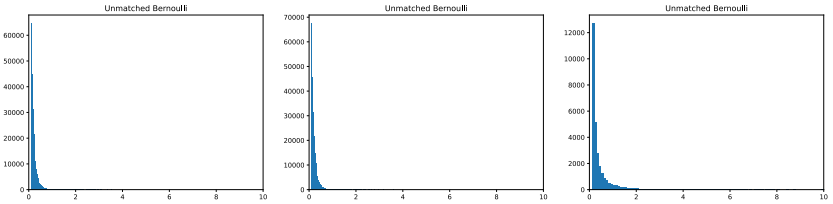
(a) Matched Bernoullis regression.



(b) Matched Bernoullis classification.

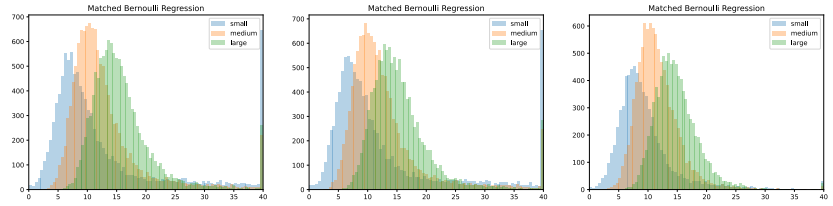


(c) Matched PPP.

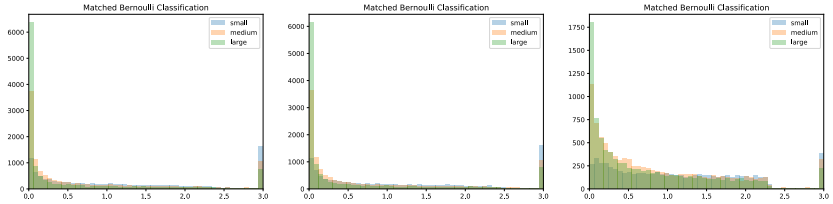


(d) Unmatched Bernoullis.

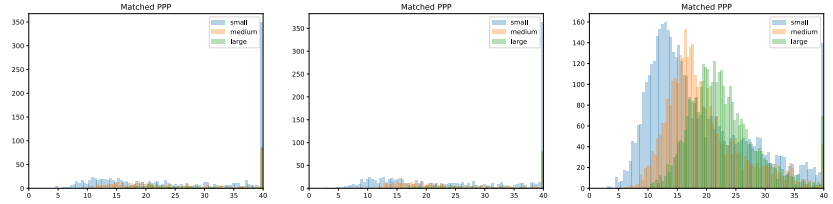
Fig. 5: Histograms over PMB-NLL decomposition for Retinanet trained with different loss functions: ES (left), NLL (middle), and MB-NLL (right). Note varying y-axes across models.



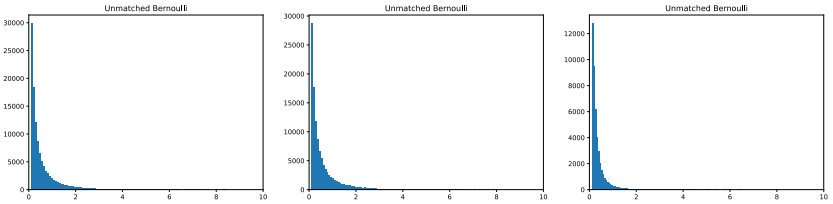
(a) Matched Bernoullis regression.



(b) Matched Bernoullis classification.



(c) Matched PPP.



(d) Unmatched Bernoullis.

Fig. 6: Histograms over PMB-NLL decomposition for Faster-RCNN trained with different loss functions: ES (left), NLL (middle), and MB-NLL (right). Note varying y-axes across models.

D Comparison to DETR loss

The DETR object detector [1] popularized the concept of treating object detection as a direct set prediction task. Similar to our method, they rely on a one-to-one matching between predictions and ground truth objects. We aim to compare their formulation to ours, highlighting similarities and differences.

D.1 DETR Loss Revisited

We start by reviewing the methods used in DETR. To find the matching between predictions and true objects, they rely on the Hungarian algorithm to minimize

$$\hat{\sigma} = \underset{\sigma \in \mathfrak{S}_N}{\arg\min} \sum_{i}^{N} \mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)}), \tag{8}$$

where $\hat{y} = \{\hat{y}_i\}_{i=1}^{N}$ is the set of predictions and y is the ground truth set of objects, where we assume y to be padded to size N with $\varnothing$ (no object). An object $y_i = (c_i, b_i)$ consists of a class label (which can be $\varnothing$) and a bounding box $b_i \in \mathbb{R}^4$, while a prediction $\hat{y_i} = (\hat{p}_{i,\text{cls}}(c_i), \hat{b}_i)$ consists of a class distribution $\hat{p}_{i,\text{cls}}(c_i)$ which assigns probabilities to all classes including $\varnothing$ and a predicted bounding box $\hat{b}_i$. Further, $\mathfrak{S}_N$ is the set of all permutations of N elements and $\mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)})$ is a pair-wise matching cost defined as

$$\mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)}) = -\mathbb{1}_{c_i \neq \varnothing} \hat{p}_{\sigma(i),\text{cls}}(c_i) + \mathbb{1}_{c_i \neq \varnothing} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)}), \tag{9}$$

with

$$\mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)}) = \lambda_{\text{iou}} \mathcal{L}_{\text{iou}}(b_i, \hat{b}_{\sigma(i)}) + \lambda_{\text{L1}} ||b_i - \hat{b}_{\sigma(i)}||_1, \tag{10}$$

where hyperparameters are typically set to $\lambda_{\text{iou}} = 2$ and $\lambda_{\text{L1}} = 5$. We can note that the matching cost for $c_i = \varnothing$ is zero.

Given the optimal matching $\hat{\sigma}$, the final loss is calculated as

$$\mathcal{L}_{\text{Hungarian}}(y, \hat{y}) = \sum_{i=1}^{N} \left[-\log(\hat{p}_{\hat{\sigma}(i),\text{cls}}(c_i)) + \mathbb{1}_{c_i \neq \varnothing} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\hat{\sigma}(i)}) \right]. \tag{11}$$

However, rather than using the negative log-likelihood for class predictions, the log-probability term is down-weighted by a factor 10 when $c_i = \varnothing$. This is motivated in [1] to handle class imbalance.

D.2 MB-NLL Relation to DETR

This section aims at describing the MB-NLL using the same notation as DETR; to simplify the comparison we focus on MB-NLL rather than PMB-NLL. We start by comparing the matching costs before moving on to the loss formulation.

Matching Cost. We can compare (9) with the cost of matching objects to Bernoulli predictions used in this work. As described in Section 3.1, we use $r = 1 - \hat{p}_{\text{cls}}(\varnothing)$. Further, for the Bernoulli predictions, the class distribution is assumed to be conditioned on existence, i.e., it has non-zero probability only for foreground classes. Hence, to clarify the relation to DETR we define

$$p_{\text{cls}}(c) = \begin{cases} 0 & \text{if } c = \varnothing, \\ \hat{p}_{\text{cls}}(c)/r & \text{otherwise,} \end{cases} \tag{12}$$

as the class distribution over foreground classes. We scale the predicted distribution $\hat{p}_{\text{cls}}(c)$ by $\frac{1}{r}$ such that $p_{\text{cls}}(c)$ becomes a proper distribution and fulfills $\sum_c p_{\text{cls}}(c) = 1$.

In Appendix A, we showed how to find the assignment that maximizes the likelihood (1) and consequently minimizes the negative log-likelihood. To enable easier comparison, we want to use the same notation as DETR and formulate a minimization over permutations

$$\hat{\sigma} = \underset{\sigma \in \mathfrak{S}_N}{\arg\min} \sum_{i}^{N} \mathcal{L}_{\text{match,MB}}(y_i, p_{\sigma(i)}), \tag{13}$$

where $\mathcal{L}_{\text{match,MB}}$ is a pair-wise matching cost. Using (1), we can express $\mathcal{L}_{\text{match,MB}}$ as

$$\begin{aligned} \mathcal{L}_{\text{match,MB}}(y_i, p_{\sigma(i)}) &= -\mathbb{1}_{c_i \neq \varnothing} \left(\log(r_{\sigma(i)} p_{\sigma(i)}(y_i)) \right) - \mathbb{1}_{c_i = \varnothing} \log\left(1 - r_{\sigma(i)}\right) \\ &= -\log(\hat{p}_{\sigma(i),\text{cls}}(c_i)) - \mathbb{1}_{c_i \neq \varnothing} \log(p_{\sigma(i),\text{reg}}(b_i)), \end{aligned} \tag{14}$$

since the cost of assigning a prediction to a true object is $-\log(r_{\sigma(i)} p_{\sigma(i)}(y_i))$, while the cost of assigning it to background is $-\log(1 - r_{\sigma(i)})$. In (14), we have also used the fact that $p_i(y_i) = p_{i,\text{cls}}(c_i) p_{i,\text{reg}}(b_i)$, and the relation

$$\hat{p}_{i,\text{cls}}(c_i) = \begin{cases} r p_{i,\text{cls}}(c_i) & \text{if } c_i \neq \varnothing \\ 1 - r & \text{if } c_i = \varnothing, \end{cases} \tag{15}$$

to obtain an expression that resembles (9) and (11).

Further, if we assume $p_{\sigma(i),\text{reg}}(b_i)$ to be a Laplace distribution with independent box parameters $b = [b^1, b^2, b^3, b^4]$, with means $\hat{b} = [\hat{b}^1, \hat{b}^2, \hat{b}^3, \hat{b}^4]$ and scales $\hat{s} = [\hat{s}^1, \hat{s}^2, \hat{s}^3, \hat{s}^4]$, we can rewrite

$$\begin{aligned} -\log(p_{\sigma(i),\text{reg}}(b_i)) &= -\log\left(\prod_{k=1}^{4} \frac{1}{2\hat{s}^k_{\sigma(i)}} \exp\left(-\frac{|b_i^k - \hat{b}^k_{\sigma(i)}|}{\hat{s}^k_{\sigma(i)}} \right) \right), \\ &= -\sum_{k=1}^{4} \log\left(\frac{1}{2\hat{s}^k_{\sigma(i)}} \exp\left(-\frac{|b_i^k - \hat{b}^k_{\sigma(i)}|}{\hat{s}^k_{\sigma(i)}} \right) \right), \\ &= \sum_{k=1}^{4} \frac{|b_i^k - \hat{b}^k_{\sigma(i)}|}{\hat{s}^k_{\sigma(i)}} + \log\left(\hat{s}^k_{\sigma(i)} \right) + \log(2)\,. \end{aligned} \tag{16}$$

As log(2) is present in all matching costs between pairs of predictions and true objects, it does not affect the optimal assignment and can be disregarded. Further, if we let $\hat{s}_i^k = s, \forall i, k$, we can use the same argument to disregard $\log\left(\hat{s}^k_{\sigma(i)}\right)$ and replace $-\log(p_{j,\text{reg}}(b_i))$ in (14) with $||b_i - \hat{b}_{\sigma(i)}||_1/s$, obtaining

$$\mathcal{L}_{\text{match,MB}}(y_i, p_{\sigma(i)}) = -\log(\hat{p}_{\sigma(i),\text{cls}}(c_i)) + \mathbb{1}_{c_i \neq \varnothing}||b_i - \hat{b}_{\sigma(i)}||_1/s. \tag{17}$$

We can now compare the expressions for the matching losses in (17) and (9), used by MB-NLL and DETR, respectively, and analyze their similarities and differences. Rather than using the log-probabilities, the original DETR matching loss uses the class probabilities directly. In [1] this is motivated as making the classification part of the cost comparable to the regression part. Interestingly, we find that the classification log-probability is comparable to the L1 regression under the constant scale assumption. Further, the classification cost in (9) only evaluates the probability of the true class when the object is not $\varnothing$. For background, the cost is set to zero. In contrast, the MB matching cost also considers the log-probability of the prediction being background and favours predictions for which the probability of background $\hat{p}_{\text{cls}}(\varnothing)$ is small. The reason for also considering the cost of assigning predictions to background in MB-NLL is that predictions with large existence probabilities infer large penalties in the final loss function if they are not assigned to a true object.

We further highlight the difference in how $\hat{p}_{\text{cls}}(\varnothing)$ is handled by the two matching costs with an example. Imagine a scenario that contains two predictions and one true object, a car. Both predictions have the same regression error, but differ in their classification. Suppose that both predictions have the same $\hat{p}_{\text{cls}}(\text{car}) = r p_{\text{cls}}(\text{car})$ but that the first prediction has a small r and a large $p_{\text{cls}}(\text{car})$ whereas the second prediction has a large r and a small $p_{\text{cls}}(\text{car})$. In (9), both these predictions would be treated as equally good. For MB-NLL however, it is better to assign the second prediction to the car, simply because that implies that the first prediction, which has a small r, is assigned to the background.

For the regression part, (9) contains both an additional IoU-loss, and the hyperparameters $\lambda_{\text{iou}}, \lambda_{\text{L1}}$ when compared to (17). While the IoU-part has no related term in (17), we find an inverse relationship for λ_{L1} and the assumed constant Laplace scale s, i.e., $\lambda_{\text{L1}} = \frac{1}{s}$. Thus, the choice $\lambda_{\text{L1}} = 5$ is equivalent of assuming a constant Laplace scale $s = 0.2$, and increasing λ_{L1} translates to assuming smaller spatial uncertainties.

Loss Function. The MB-NLL training loss

$$\mathcal{L}_{\text{MB}}(y, \hat{y}) = \sum_i^N \left[-\log(\hat{p}_{\hat{\sigma}(i)}(c_i)) + \mathbb{1}_{c_i \neq \varnothing} \sum_{k=1}^4 \frac{|b_i^k - \hat{b}^k_{\hat{\sigma}(i)}|}{\hat{s}^k_{\hat{\sigma}(i)}} + \log\left(\hat{s}^k_{\hat{\sigma}(i)}\right) \right], \tag{18}$$

is very similar to the matching loss, but makes use of the scaling parameters $\hat{s}^k_{\sigma(i)}$ predicted by our networks, whereas the matching loss uses a fixed scaling parameter s.

We see that both (11) and (18) contain two terms, one for classification and one for regression. However, in contrast to the original DETR loss (11), we do not down-weigh the log-probability when predictions are assigned to $\varnothing$. We believe this is one of the contributing factors to the reduced number of false detections when training with the MB-NLL loss. By down-weighing the penalty for predictions assigned to $\varnothing$, the detector is encouraged to produce artificially high classification confidence. In mAP, the measure that DETR most likely has been optimized toward, the high classification confidence is generally not penalized as it only relies on the ranked confidences among predictions and not the absolute confidence values.

Similar to the matching cost, (18) lacks the IoU cost found in (11), but has an identical L1-loss when $\lambda_{\text{L1}} = \frac{1}{\hat{s}^1_{\sigma(i)}} = \cdots = \frac{1}{\hat{s}^4_{\sigma(i)}}$. Naturally, L1-losses increase with larger λ_{L1}, but using (18) this can also be explained as assuming smaller regression uncertainties. Finally, when training with MB-NLL, the relation between the Laplace scale and λ_{L1} also shows how the network can self-regulate the regression penalties. For the network to chose suitable scales, it has to balance the two terms $\log\left(\hat{s}^k_{\hat{\sigma}(i)}\right)$ and $\frac{|b^k_i - \hat{b}^k_{\sigma(\hat{i})}|}{\hat{s}^k_{\hat{\sigma}(i)}}$. While the first term encourages smaller scales, choosing too small scales may yield a large cost if the regression performance $|b^k_i - \hat{b}^k_{\sigma(\hat{i})}|$ is poor.

References

1. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: European Conference on Computer Vision. pp. 213–229. Springer (2020)
2. Harakeh, A., Waslander, S.L.: Estimating and evaluating regression predictive uncertainty in deep object detectors. In: International Conference on Learning Representations (2021)
3. Wu, Y., Kirillov, A., Massa, F., Lo, W.Y., Girshick, R.: Detectron2. `https://github.com/facebookresearch/detectron2` (2019)