Segmenting with Less Supervision: from Pseudo-Labels to Foundation Models

1 Introduction

Semantic segmentation is a very important field of Computer Vision that aims to classify the nature of each pixel of an input image. It is mostly used in medical imaging [1], autonomous driving [2], and general image segmentation [3].

A first approach could be to use image masks as targets to train segmentation models. The problem is that this type classification technique is pixel-wise and demands costly pixel labeling. This observation leads to think about other methods like Weakly Supervised Semantic Segmentation (WSSS) for instance. WSSS describes the techniques used to transform weak supervision signals into pseudo masks used to train segmentation models.

We asked ourselves: *Can weakly supervised segmentation approaches reach reasonable performance compared to full supervision?* Using the Oxford-IIIT Pet Dataset [5], we explored several techniques to generate pseudo-masks and train segmentation models. Among others, Class Activation Maps (CAM) [6] and bounding boxes [4] can help localize class-specific image regions. Moreover, we found out that Foundation Models (FMs) [7] [8] brought significant improvements in segmentation tasks. Our second, open-ended question then became: *What are the key limitations of pseudo mask generation in WSSS, and how can foundation models help overcome them?*

2 Methods

2.1 General Framework

Here is the detail of the framework used in most of the methods described below. It is made of two steps:

- (1) We use the weak labels of our images to generate a pseudo-mask using a weakly supervised model
- (2) We train a segmentation model using these binary pseudo-masks as labels

Let's also mention that the baseline model and the FM-based approach differ from this framework. The baseline model directly uses the segmentation model with the ground truth labels as targets, and FMs directly segment the objects in the image around given points.

2.2 Segmentation Models

The semantic segmentation models are trained on pixel-wise annotation, which can come from two different sources : (1) the ground truth masks that we use to compute the baseline and (2) pseudo-masks that we generated using different methods described below.

2.3 Weak Supervision via CAM

The first method we used to generate pseudo-masks was Class Activation Maps (CAM). The idea is to train a CNN-based classifier using only the image-level labels of the pictures (e.g., Bengal, Persian, etc.). Once those are trained, we use the last dense layer of the classifier to understand the regions of the images which were the most influential in the selection of the label, and compute a score translating the importance of each region. We then apply a threshold function to those scores to generate a pseudo mask, which will later be given to the segmentation model.

Three different CAM-based methods were tested : **(1) CAM** is the basic approach in which the deciding regions of the input image are obtained by linear combination of the last feature map and of weights of the last dense layer leading to the class chosen for the image. **(2) Grad-CAM** [9] computes importance weights using the gradients of the class score with respect to the feature maps. The localization map is then obtained by linear combination of those weights with the associated feature maps. **(3) Grad-CAM++** [10] refines the weight computation of Grad-CAM by including higher-order derivatives. Since performance of CAM-based methods heavily depend on the chosen threshold, we decided to perform an ablation study to find the best threshold for segmentation network predictions.

2.4 Weak Supervision via Bounding Boxes

This method consists in improving the pseudo-masks obtained through the CAM approach by enclosing the pets we want to segment in a bounding box and refining the pixels inside. We decided not to use the bounding boxes of the dataset to have more flexibility. To create our own bounding boxes, we generate the CAM of the image and apply a threshold to obtain an initial pseudo-mask. We then draw a bounding box enclosing this first pseudo-mask, and used this bounding box in three different ways :

- **(1)** We directly used this bounding box as labels for our segmentation model.
- **(2)** We applied the **the GrabCut** algorithm [11] within the box to create a refined second pseudo-mask. This algorithm iteratively uses GMMs and Graph Cuts to model the foreground and the background.The refined second pseudo-mask generated is used for the training of the segmentation model.
- **(3)** We generated two sets of pseudo-masks using both **GrabCut** and **the SuperPixel Simple Linear Iterative Clustering (SLIC)** algorithm [12] which groups pixels of the bounding box into meaningful clusters. For each picture, we take the overlapping section of the two pseudo-masks generated, to create a refined pseudo-mask. The refined pseudo-mask is then used for the training of the segmentation model.
- **(4)** We trained our segmentation model using raw bounding boxes for the first half of the training, and pseudo-masks generated by the mix of **GrabCut** and **SLIC** mentionned in (3) for the second half of the training

2.5 Foundation Models

Segment Anything Model 2 (SAM2) is a Visual Transformers-based model developed by MetaAI [13]. It allows to generate high-quality masks. There is no need for labeling as the model only requires object indications and outputs masks of possible objects (without assocaited label) around these points. We used model in four different ways :

- **(1)** We naively indicate the center of the image and generate only one mask. This mask is considered to be the predicted mask of the model.
- **(2)** We naively indicate the center of the image, generate several masks and only keep the mask with the most pixels to mitigate under-segmentation. This mask is considered to be the predicted mask of the model.
- **(3)** We naively indicate the center of the image and only keep the mask with the most pixels. This mask is then used as a pseudo-mask to train our segmentation model.
- **(4)** We used the best performing approach from the CAM experiments to find prominent activation points (and avoid selecting naively the center) and use them as pointers for our model to generate multiple segmentation hypothesis. Here again, we only keep the largest mask and use it as a pseudo-mask to train our segmentation model.

3 Experiments

3.1 Experimental Setup

For this study, we use the Oxford-IIIT Pet dataset which contains pixel-level annotations, image-level labels and bounding boxes. Data transformation techniques are applied to the images (resize and normalization) along with some data augmentation techniques (colour jitter or random horizontal flips) The latter are only applied to the training set. To evaluate our methods, We used the DICE score, Intersection over Union (IoU) score, pixel accuracy, precision & recall (per pixel). Models were evaluated on the test set, comparing the predicted mask of a given image to the ground truth mask

3.2 Baseline Comparison

Table 1 show the results obtained for the baseline framework, in which we train a segmentation model with the animals ground truth masks as labels. We evaluated UNet, DeepLabV3 (DLV3) and FCN, and used hyperparameters shown in table B. Models were trained with Binary Cross Entropy Loss and Adam optimizer, using a ReduceLROnPlateau scheduler to dynamically adjust the learning rate. All those models were trained using `baseline.py`.

Since the best performing model is DLV3, we will use it as a default segmentation model for the rest of the study. Results of the DLV3 predictions are shown in figure 1 of Appendix C

Table 1. Full supervision. Performance of baseline segmentation models.

Model	Pretrained	LR	Epochs	Dice	IoU	Acc	Prec	Rec
UNet	No	1e-4	25	0.9064	0.8287	0.9190	0.8866	0.9270
DLV3 (ResNet50)	Yes	1e-4	10	**0.9577**	**0.9189**	**0.9645**	**0.9640**	**0.9515**
FCN (ResNet50)	Yes	1e-4	10	0.9514	0.9074	0.9594	0.9633	0.9399

3.3 CAM-based Pseudo Masks

To obtain our CAMs, we trained three CNN-based classifiers (ResNet50, ResNet101, ResNet121) with *breed_ id* for target. We selected ResNet101, since it had the best test accuracy (92.61% vs 91.31% for ResNet50 and 90.41% for ResNet121). We then generated pseudo-masks using separately CAM, Grad-CAM & Grad-CAM++, and gave them as labels to our segmentation model DLV3. For a fixed threshold of 0.5, the best performance was achieved with ResNet101 and Grad-CAM (table 2.1). Therefore, we decided to keep those for the rest of our experiments and to use them for our ablation study on the threshold. Table 2.2 show that the best DICE score is obtained for a threshold of 1.10^{-2}.

You can visualize the Grad-CAM (with threshold 1.10^{-2}) generated pseudo-masks along with the predicted masks in fig. 2 and fig. 3 of Appendix C.

Table 2. Comparison of the performance of different CAM methods (left) and thresholds (right)

Table 2.1 CAM-based pseudo mask performance for a fixed threshold of 0.5

CAM Type	Dice	IoU	Acc.	Prec.
CAM	0.3364	0.2022	0.6522	0.8700
Grad-CAM	0.3589	0.2187	0.6581	0.8657
Grad-CAM++	0.3208	0.1911	0.6481	0.8724

Table 2.2 Grad-CAM thresholds performance

Threshold	DICE	IoU	Acc.	Prec.	Rec.
1.10^{-4}	0.5951	0.4236	0.4254	0.4238	**0.9986**
1.10^{-3}	0.6208	0.4501	0.4906	0.4530	0.9861
5.10^{-3}	0.6897	0.5229	0.6740	0.5784	0.8450
1.10^{-2}	**0.6918**	**0.5288**	0.7028	0.6159	0.7890
1.10^{-1}	0.6276	0.4680	0.7260	0.7234	0.569
5.10^{-1}	0.3656	0.2237	0.6570	0.8387	0.2338
7.10^{-1}	0.1996	0.1109	0.6175	**0.8651**	0.1128

The results of table 2 were obtained using `main_cam.py`. The hyperparameter used for both experiments are available in table 7 and 8 of Appendix B.

3.4 Bounding Box-based Pseudo Masks

Here, we created our own bounding boxes to generate the pseudo-masks used to train the segmentation model DLV3. To generate our pseudo-masks, we used the raw bounding boxes (Raw BB) as baseline, refined their content using GrabCut only (GrabCut) and the combination of GrabCut & SLIC Superpixel (Mix

GrabCut-Superpixel) mentionned in paragraph 2.4. Additionally, we trained DLV3 with both raw bounding boxes during 20 epochs, and the pseudo masks obtained through the combination of GrabCut & SLIC Superpixel during 20 more epochs (Raw BB + Mix GrabCut-Superpixel)

The code used to train these models is in `main_bbox.py` and the results are displayed in table 3. You can visaulize the generated pseudo-masks in figures 4, 5 and 6, 7 of Appendix C

Table 3. Bounding box-based pseudo mask performance

Method	Dice	IoU	Accuracy	Precision	Recall
Raw BB	0.7365	0.5830	0.7558	0.6772	**0.8073**
GrabCut	0.7832	0.6436	0.8144	0.7738	0.7926
Mix GrabCut-SuperPixel	0.7918	0.6554	0.8314	0.8287	0.7581
Raw BB + Mix GrabCut-SuperPixel	**0.8042**	**0.6726**	**0.8426**	**0.8483**	0.7645

3.5 Foundation Model-based Masks (SAM2)

For the experiments, we used SAM2 with the small Hiera variant, the ViT-H architecture type and the center of the image as object indication. The model was not fine-tuned. We generated a unique mask (single output), and multiple ones (multiple output). We also used the outputs of SAM2 as pseudo masks for DLV3 (SAM Pseudo + DLV3). Finally, we used CAM peaks (3 points with the highest CAM score) as SAM2 prompts to generate multiple segmentation masks, selecting the largest one (CAM-SAM + DLV3). To generate CAMs we used Grad-CAM algorithm along with a ResNet101 classifier.

Implementation details and evaluation procedures are available in the `main_fm.py` file. Examples are given in figures 8 and 9 of Appendix C

Table 4. SAM2 segmentation performance

Method	Dice	IoU	Accuracy	Precision	Recall
(single-output) Raw SAM	0.7843	0.6452	0.8434	0.9386	0.6736
(multi-output) Raw SAM	0.8635	0.7598	0.8908	0.9155	**0.8171**
SAM Pseudo + DLV3	**0.8692**	**0.7687**	**0.8993**	**0.9635**	0.7917
CAM-SAM + DLV3	0.7602	0.6132	0.8081	0.8056	0.7196

4 Results

We explored several methods and trained multiple models, the respective results are compared in table 5. As expected, DLV3 has the best results since it was trained in a fully supervised context, but this advantage comes with the fact that the model is heavy to train and depends on detailed pixel-level annotations, which we didn't use with our weakly supervised techniques.

Even with the best threshold, Grad-CAM only managed to highlight the most defining features (for instance, the head) of the pictures. The generated pseudo-masks are always circular and of a size anti-proportional to the threshold: they are too generic and prevent the segmentation model from generalizing correctly. Nevertheless, this issue was corrected by the bounding boxes approach, and in particular by the usage of algorithms such as GrabCut and SuperPixel SLIC which put an emphasis on the the creation of a refined pseudo-mask and allowed a +14.5% DICE improvement compared to the best Grad-CAM.

We then saw that training DLV3 with both raw bounding boxes and more refined pseudo-masks (mix GrabCut-SuperPixel) was very efficient as it allowed the model to understand first the raw location of the pet before creating a more precise and refined mask. This progressive approach led to a +2% increase compared to the lone mix GrabCut-SuperPixel method.

Finally, FMs like SAM2 allow to direclty generate masks that more accurately capture the complete structure of objects; the usage of SAM's output as pseudo-masks for DLV3 over-performs all the other methods by at least +7.4% in DICE score. Nevertheless, we notice that the usage of Grad-CAM to give object indication harms the performance: the generated object indications are sometimes not accurate and off target, leading to less precise pseudo-masks.

Table 5. Comparison of best models across supervision strategies

Method	**Dice**	**IoU**
Full Supervision (DLV3)	**0.9577**	**0.9189**
CAM (Grad-CAM ResNet101, 1.10^{-2})	0.6918	0.5288
Bounding box (Raw BB + Mix GrabCut-SuperPixel)	0.8042	0.6726
Foundation model (SAM2 Pseudo + DLV3)	0.8692	0.7687

5 Discussion

As expected, the FMs, while computationally heavy and not fine tuned, outperformed the other approaches. Regarding the latter, we discovered that generating good pseudo-mask is more important for the results than segmentation architecture. Our work has limitations that can be addressed in the future. We only used the Oxford-IIIT Pet dataset, which limits the scope of our study. Additionally, FMs are treated as black boxes, making it difficult to study, understand and further improve their behavior. Therefore, future directions include combining methods (e.g., FMs & bbox, CAM and CRF[14]), using larger datasets (e.g., Cityscapes), and leveraging newer models like SegFormer.

6 Conclusion

We built baseline models and evaluated several WSSS strategies. DLV3 with full supervision performed best, and FMs allowed to reach very close results. Ultimately, we noticed that pseudo-mask quality is more critical than segmentation architecture in WSSS.

Appendix

A Mathematical formulas for CAM, Grad-CAM and Grad-CAM++

– CAM: In this method, the deciding regions of the input image are obtained by linear combination of the last feature map and of weights of the last dense layer leading to the class chosen for the image

$$M_c(x, y) = \sum_k w_k^c \, f_k(x, y)$$

where:
- $M_c(x, y)$ is the class activation map for class c at spatial location (x, y).
- $f_k(x, y)$ is the activation of the k-th feature map at position (x, y).
- w_k^c is the weight corresponding to class c for feature map k.

– Grad-CAM : This method computes importance weights using the gradients of the class score with respect to the feature maps:

$$\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k}$$

where:
- α_k^c is the importance weight for the k-th feature map with respect to class c.
- y^c is the score for class c.
- A_{ij}^k represents the activation at spatial location (i, j) of the k-th feature map.
- Z is a normalization factor (typically the total number of spatial locations, i.e., $Z = \sum_i \sum_j 1$).

The Grad-CAM localization map is then computed as:

$$L_{\text{Grad-CAM}}^c = \text{ReLU}\left(\sum_k \alpha_k^c \, A^k\right)$$

where: the ReLU function ensures that only features with a positive influence on the class c are retained.

– Grad-CAM++ : Grad-CAM++ refines the weight computation by including higher-order derivatives:

$$\alpha_k^c = \frac{\sum_{i,j} \frac{\partial^2 y^c}{\partial (A_{ij}^k)^2}}{2 \sum_{i,j} \frac{\partial^2 y^c}{\partial (A_{ij}^k)^2} + \sum_{i,j} A_{ij}^k \frac{\partial^3 y^c}{\partial (A_{ij}^k)^3}}$$

where:

- The numerator is the sum of the second-order derivatives of the class score y^c with respect to the activations A_{ij}^k.
- The denominator consists of twice the sum of these second-order derivatives, plus an additional term that involves the third-order derivatives weighted by the activations A_{ij}^k.

The corresponding Grad-CAM++ localization map is given by:

$$L_{\text{Grad-CAM++}}^c = \text{ReLU}\left(\sum_k \alpha_k^c \, A^k\right)$$

where: again the ReLU function ensures that only positive contributions are considered.

B Hyperparameters used

Table 6. Full supervision. Hyperparameters

Hyperparameter	**Value**
Training batch size	64
Validation batch size	32
Test batch size	64
Validation split	0.2
Image resize	256

Table 7. Weak supervision - CAM. Hyperparameters

Hyperparameter	**Value**
Training batch size	64
Validation batch size	32
Test batch size	64
Validation split	0.2
Image resize	256
Epochs - classifier	15
Learning rate - classifier	1e-4
Epochs - segmentation	10
Learning rate - segmentation	1e-4

C Visualizations

Table 8. Ablation Study - Grad-CAM. Hyperparameters

Hyperparameter	**Value**
Training batch size	64
Validation batch size	32
Test batch size	64
Validation split	0.2
Image resize	256
Epochs - classifier	15
Learning rate - classifier	1e-4
Epochs - segmentation	5
Learning rate - segmentation	1e-4

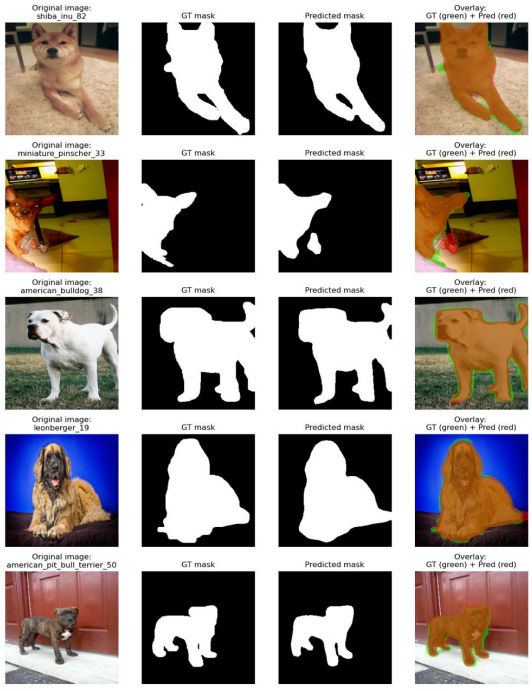

Fig. 1. Subsets of the results obtained for the fully supervised segmentation model DLV3. From left to right we have: the input image, the ground truth mask (GT), the predicted mask, and the overlay of GT (green) & the prediction (red)

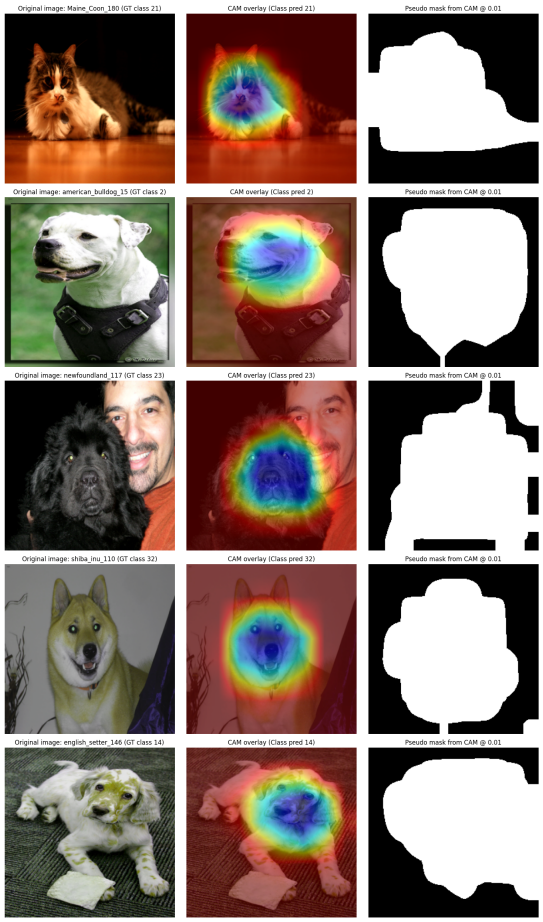

Fig. 2. Subsets of the pseudo-masks obtained with Grad-CAM ($1.10^{-}2$). From left to right we have: the input image, the CAM overlay heatmap, the resulting pseudo-mask

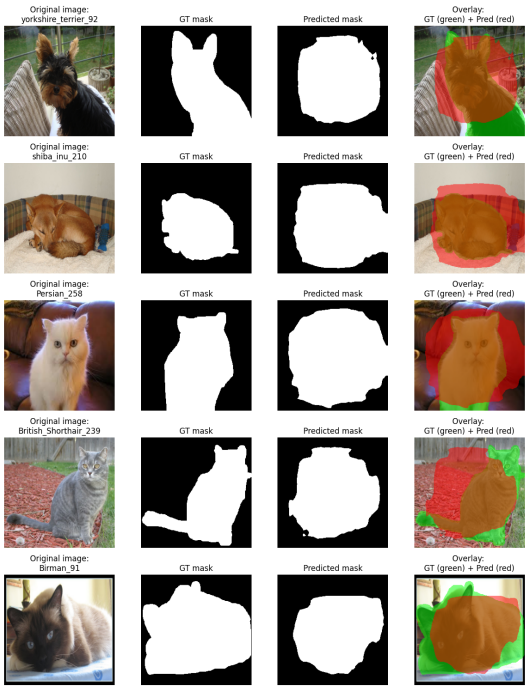

Fig. 3. Subset of the results of Grad-CAM (1.10^{-2}) to generate the pseudo masks and DLV3 to learn the segmentation. From left to right we have: the input image, the ground truth mask (GT), the predicted mask, and the overlay of GT (green) & the prediction (red)

Fig. 4. Subsets of the results obtained for the Raw Bounding Box method. From left to right we have: the input image, the ground truth mask (GT), the predicted mask, and the overlay of GT (green) & the prediction (red)

Fig. 5. Subsets of the results obtained for the BBOX-GrabCut method. From left to right we have: the input image, the ground truth mask (GT), the predicted mask, and the overlay of GT (green) & the prediction (red)

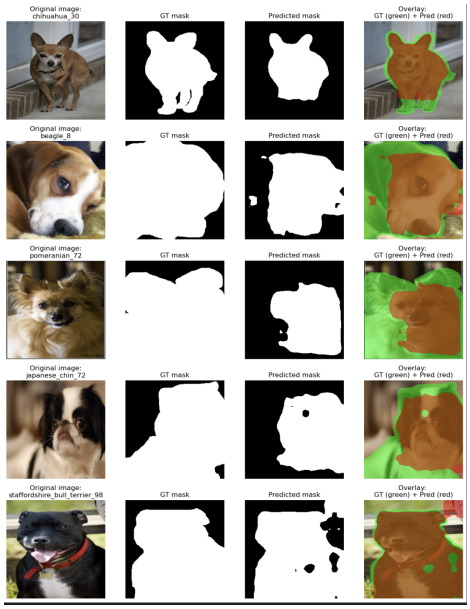

Fig. 6. Subsets of the results obtained for the mix GrabCut-SuperPixel method. From left to right we have: the input image, the ground truth mask (GT), the predicted mask, and the overlay of GT (green) & the prediction (red)

Fig. 7. Subsets of the results obtained for the Raw BB + mix GrabCut-SuperPixel method. From left to right we have: the input image, the ground truth mask (GT), the predicted mask, and the overlay of GT (green) & the prediction (red)

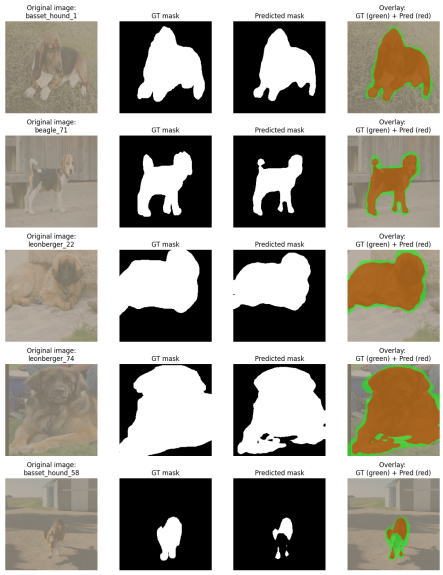

Fig. 8. Subsets of the results obtained for the zero-shot SAM2 + DLV3 method. From left to right we have: the input image, the ground truth mask (GT), the predicted mask, and the overlay of GT (green) & the prediction (red)

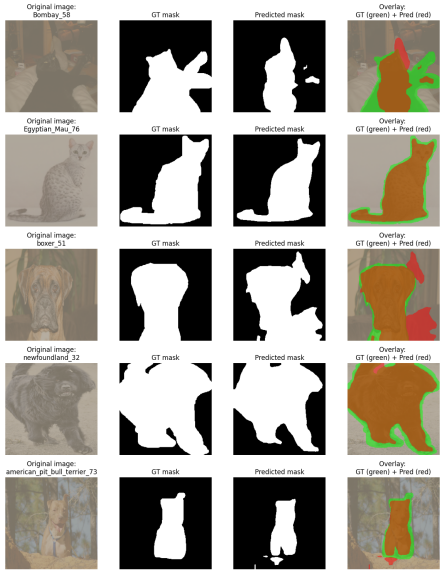

Fig. 9. Subsets of the results obtained for the CAM + SAM2 + DLV3 method. From left to right we have: the input image, the ground truth mask (GT), the predicted mask, and the overlay of GT (green) & the prediction (red)

References

1. Ronneberger et al., *net: Convolutional networks for biomedical image segmentation. Medical Image Computing and Computer-Assisted Intervention*, MICCAI, 2015.
2. Cordts et al., *he cityscapes dataset for semantic urban scene understanding. IEEE Conference on Computer Vision and Pattern Recognition*, CVPR, 2016.
3. Long et al., *Fully Convolutional Networks for Semantic Segmentation*, CVPR, 2015.
4. Khoreva et al., *imple does it: Weakly supervised instance and semantic segmentation* , 2016.
5. Parkhi et al., *Cats and Dogs*, CVPR, 2012.
6. Zhou et al., *Learning Deep Features for Discriminative Localization*, 2015.
7. Xiaobo et al., *Foundation Model Assisted Weakly Supervised Semantic Segmentation* , IEEE, 2024.
8. Wu et al., *Wps-sam: Towards weakly-supervised part segmentation with foundation models*, 2024.
9. Selvaraju et al., *Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization*, ICCV17, 2017
10. Chattopadhyay et al., *Grad-CAM++: Improved Visual Explanations for Deep Convolutional Networks*, WACV2018, 2018
11. Rother et al., *"grabcut": interactive foreground extraction using iterated graph cuts.*, ACM TOG, 2004.
12. Achanta et al., *SLIC superpixels*, EPFL Technical Report 149300, June 2010
13. Ravi et al., *SAM 2: Segment Anything in Images and Videos*, MetaAI, 2024
14. Liu et al., *CRF learning with CNN features for image segmentation*, 2015