Towards low-altitude image analysis: Object-enhanced concept detection

Emmanouil Christakis, Stefanos Demertzis, Konstantinos Stavridis,
Athanasios Psaltis, Anastasios Dimou, Petros Daras
Visual Computing Lab, Information Technologies Institute
6^{th} Km Charilaou-Thermi, Thessaloniki, Greece
{manchr, demertzis.s, staurid, at.psaltis, dimou, daras} @iti.gr

Abstract

In this report, the technical details of the VCL-team submission to the Disaster Scene Description and Indexing (DSDI) challenge, are presented. Although the modern Deep Learning (DL) concept detection schemes can be trained to detect, classify and index a huge variety of concepts under condition of high diversity, they rely solely on global-level information. However, the core part of LADI, i.e. the low altitude, oblique perspective of the imagery and disaster-related features, is quite unique in computer vision literature, therefore it raises challenges that need to be adequately addressed in the design of NN architecture. In order to evaluate the effect that objects have in scene understanding, a set of experiments/runs was conducted using different schemes for incorporating local-level information (e.g. objects, entities).

1. Introduction

Being at the centre of global climate change, humans should start thinking of measures not only proactive to natural disasters but also reactive ones. Civil protection shows particular interest in applications that help mitigate the impact of such disasters by providing fast and reliable access to information and analytics in the aftermath of them. Machine learning has been successfully applied in a plethora of computer vision tasks (e.g. object detection, concept detection, action recognition, etc.) achieving remarkable performance even in case of complex, cluttered, and highly-occluded environments. Having a closer look at the core part of LADI, it consists of low altitude images taken in various disaster-related environments, comprising a fairly unique benchmark in computer vision literature. This work addresses the Trecvid 2020 evaluation [1] in the Disaster Scene Description and Indexing (DSDI) subtask.

The purpose of this task is the application of computer vision capabilities on natural disasters response. More specifically, the teams were asked to develop systems that can label video clips with the correct disaster related features. These features are split in 5 high-level concept *categories* (namely damage, environment, infrastructure, vehicle, and water), being further divided in low-level sub categories giving in total 32 *features*. It was the first time that this task was included in the Trecvid evaluation. The DSDI dataset consists of over 20000 annotated images, based on the presence of the disaster related features in each image and was based on the Low Altitude Disaster Imagery (LADI) dataset [6]. In order to evaluate their submissions, teams were given 5 hours of video from a recent natural disaster event, split in small clips of 60 seconds maximum duration and were asked to give for each feature, a list of the clips that contained this feature. This list had to be ranked based on the confidence that the feature was present in a clip.

2. Preparing the Dataset

As mentioned in the previous section, the DSDI dataset contains approximately 20000 images. To ensure the quality of the annotations, only images with file size over $4MB$ were included in the dataset.This choice was made based on the assumption that images with larger size and therefore higher resolution are more likely to contain clearer concepts that may lead to better manual annotation. As a pre-processing step, the selected images were resized/downsized (224x244) to meet the computational constraints set by convolutional neural networks (CNNs)

Another issue that we faced with the provided dataset was the conflicting labels given by the annotation workers. In those cases we assumed the feature was present if at least 1 worker annotated it.

2.1. Object-based Annotations

The annotations provided by the dataset of the DSDI track included 32 concepts of interest. Given the high challenges posed by this dataset, we argue that the integration of

local-level features into the analysis and more specifically, the localization and classification of core image elements, would lead to improved performance in overall scene understanding. Based on this hypothesis, we trained an object detection model to detect entities within the images providing their bounding boxes and class. Given the lack of ground truth information for this task, we opted to annotate a small portion of the dataset ourselves. We plan to provide our annotations in the immediate future to aid the teams participating in the DSDI challenge in the coming years. Our annotations concern only features in the 'vehicle' and 'infrastructure' categories as the features in the other categories can not be precisely localized with bounding boxes in an image. The output of the object detection model was utilized in some of our submissions.

2.2. Frame Extraction

Given that the evaluation was on videos, we decided to classify them based on multiple frames that were subsampled from each video. In order to have a limited but descriptive set of images for evaluation, we had to create a frame export strategy that would adjust to the diverse size and frame rate of the clips. This included the following steps: first we extracted 1 out of 3 frames if clip had less than 10 fps, 1 out of 100 frames if the clip had less than 20 and 1 out of 200 frames if the clip had more or equal to 20 fps. With this approach we ended up having 2 - 4 frames per given video clip to feed to our networks. We assume that a clip is characterized by the predictions exported from the model for each of the individual images that compose it.

3. Our Submissions

To address the challenges posed by the DSDI dataset, several CNN-based approaches have been investigated. The first two submissions are utilizing only the provided annotation during the training phase, while the rest also exploit the object-based annotation performed by the team.

In the following paragraphs we give a detailed overview of the implemented architectures, highlighting the important parts of each as well as the considerations made during the training phase.

Five Independent ResNet101 Classifiers *(L_VCL_1)*: In the first submission, five different Resnet101 [4] classifiers were employed, one for each category. During training, the original dataset was split into 5 smaller parts, each one used to train a different classifier. Each of the smaller datasets contained only images along with their annotations for one of the 5 categories. The classifiers were pretrained on ImageNet [3] and fine-tuned on the 5 smaller datasets. Due to its simplicity, we regard this approach as our baseline with which to compare the other submissions. A drawback of this method lies in its inference speed as the prediction on one image requires running 5 ResNet101 networks.

A multi task classifier *(L_VCL_5)*: In the second submission, to mitigate the drawback of the first one, we built a network with a common ResNet101 backbone and 5 separate classifier heads on top, one for each separate task n. We made 5 dataset loaders so we could iterate over them to generate batches separately and calculate the loss $\mathcal{L}_n$ for only that task head of the network. The backbone was extracted from a pretrained on ImageNet Resnet101 classifier. We were only perform back propagation once. We run each dataset's batch through and calculate the loss $\mathcal{L}_n$ for each task n. Once we have the all the auxiliary losses computed, we aggregate them on a global loss $\mathcal{L}$ and perform back propagation.

$$\mathcal{L} = \frac{1}{5}\sum_{n=1}^{5}\mathcal{L}_n$$

With this approach we assumed that the backbone can extract common features that can be utilized by the different classifier heads. This has the advantage that during the inference on an image, the ResNet101 backbone, which has many more parameters than the classifier heads, has to be forward passed only once, which significantly speeding up the process compared to the first submission. However, the average precision drops substantially using this approach.

Classifiers for concept / Object Detection for objects *(O_VCL_3)*: In this we begin to utilize the formed object localization annotations, based on the observation that targeted features (e.g. objects) can make a significant contribution the the final classification result. The aim here is to apply object detection networks to detect features from the vehicle and infrastructure categories consisting of 13 features in total. To achieve this we fine-tuned a commonly used object detector, Faster R-CNN [7], using the object-specific annotations. The original network was trained on the COCO dataset [5]. The input for this network is an image and the output are bounding boxes coordinates for each category with a confidence score for each box. However since we want to know only if an image contains a feature or not and do not care for the localization task, where the feature is located in the image, we had to convert the output of Faster R-CNN to predict only the presence or absence of a category. To achieve this we extracted the class probability array S from trained Faster R-CNN network and initialize an image class probability vector $\mathcal{P} = \mathbf{0}$. Where S a $100 \times n$ array, $\mathcal{P}$ a n dimension vector and n the number of classes for 100 best predicted ROIs. Let (i, j) be the coordinates of the first maximum value of S

$$(i, j) = \{(i, j)|S_{ij} = max(S)\}$$

We store the value to image class probability vector $\mathcal{P}$,

$$\mathcal{P}_j = max(S)$$

Algorithm 1: Convert ROIs to class labels

input : Scores Array S. A $100 \times n$ array
output: Vector $\mathcal{P}$

$\mathcal{P} \leftarrow 0_n$;

for $k \leftarrow 1$ **to** n **do**
  m $\leftarrow$ max(S) ;
  i , j $\leftarrow$ position of m in S ;
  S[:, j] $\leftarrow 0$;
  S[i, :] $\leftarrow 0$;
  $\mathcal{P}$[j] $\leftarrow$ m ;
end

Then we set zero whole ith row an jth column of S array. We repeat the process until $\mathcal{P}$ vector is full constructed as shown in Algorithm 1. If this score $\mathcal{P}_j$ is over a threshold we assume that the feature is present in the image. For the features in the other three categories we use the classifiers implemented in L_VCL_1.

Classifiers over Faster R-CNN Features Map *(O_VCL_6)*: This is similar to submission O_VCL_5 where we have Resnet101 backbone and 5 classifier heads on top. However, instead of using a pretrained ResNet101 on ImageNet like in O_VCL_5, we use the trained Faster R-CNN from O_VCL_3, keeping only it's Resnet backbone while removing the region proposal network and classifier heads. Like O_VCL_5 we made 5 dataloaders to iterate over datasets separately and calculate the auxiliary loss $\mathcal{L}_n$ for only the task head of the network. We back-propagate 5 times, once per task. The weights of the backbone were kept frozen during the training of this submission and only the five classifier heads were being updated.

Classifiers over Faster R-CNN Features Map with attention *(O_VCL_2)*: In the previous approach, we do not take advantage of the bounding boxes detected by the Faster R-CNN and only use it's backbone network to extract base features map, which in turn are passed to the classifier heads. Here, we intend to create attention masks from the detected bounding boxes and then apply these on masks on the base features map extracted from the Faster R-CNN backbone.

Inspired by the Cascade R-CNN [2] learning approach, the proposed network intend to achieve increased concept detection performance by simultaneously training local-level (object detection) and global-level (concept detection) classifiers. In particular, an object detection classifier is trained to refine its input regions in such a way that highlights salient areas which then were fed as an attention mask to a second task classifier.

Let B be the extracted base features map from the cascade stage 1, forming a $1024 \times 14 \times 14$ tensor and the proposed normalized attention mask M_{norm} is a 14×14 array. The goal is to highlight regions in the features where Faster R-CNN has detected vehicle or infrastructure objects, assuming that "highlighting" these regions would boost the classification performance of the classifier heads.

In order to create the normalized attention mask M_{norm}, for each ROI prediction $b^{[\ell]}$ with confidence level greater than 0.5

$$b^{[\ell]} = \left[x^{[\ell]}_{min}, y^{[\ell]}_{min}, x^{[\ell]}_{max}, y^{[\ell]}_{max}, score^{[\ell]}\right]$$

we calculate the set of the coordination pairs of the mask array covered by the ROI

$$A^{[\ell]} = \left\{ (i,j) \middle| y^{[\ell]}_{min} \leq i \leq y^{[\ell]}_{max}, x^{[\ell]}_{min} \leq j \leq x^{[\ell]}_{max} \right\}$$

and we compute the normalized attention mask

$$M_{norm} = \frac{1}{max(M)} M$$

where,

$$M = \sum_{\ell} \left(\sum_{(i,j) \in A^{[\ell]}} \left(O_{ij} + score^{[\ell]} \right) \right)$$

Finally we apply the attention mask to base features map using Hadamard product to each one of 1024 base features map layers k

$$B_{k_{new}} = B_k \circ (M_{norm} + J_{14})$$

where J_{14} the 14×14 all-ones matrix. Features map and attention mask visuals can be found in Figure 1.

Exactly like O_VCL_6 we made 5 dataloaders to iterate over datasets separately and calculated the 5 losses $\mathcal{L}_n$ for every task head of the network. We back-propagate 5 times, once per task. The weights of the backbone are kept frozen during the training of this submission and only the five classifier heads are being trained.

Five Independent ResNet101 Classifiers with attention *(O_VCL_4)*: Similarly to L_VCL_1, in this submission we have used 5 different classifiers, one for each category. The only difference from L_VCL_1 is that we apply the normalized attention mask from O_VCL_2 over base features of the 3^{rd} layer C_3 of the ResNet101 backbone. However this method did not perform as expected.

4. Results and analysis

In this section we analyze the performance achieved by our submissions in terms of mean average precision (mAP). In figure 2 the mAP achieved by our submissions on the

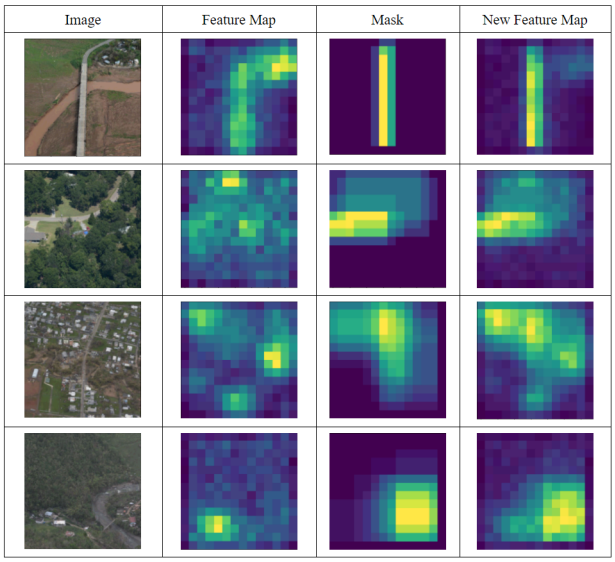

Figure 1. Base Features Map before and after normalized attention mask application.

	L_VCL_1	O_VCL_2	O_VCL_3	O_VCL_4	L_VCL_5	O_VCL_6
damage (misc)	0.198	0.144	0.198	0.013	0.035	0.125
flooding	0.629	0.512	0.629	0.135	0.266	0.512
landslide	0.066	0.274	0.066	0.061	0.045	0.116
road washout	0	0.001	0	0.001	0.001	0.006
rubble / debris	0.236	0.262	0.236	0.029	0.048	0.224
smoke / fire	0.064	0.02	0.064	0.001	0.007	0.2
dirt	**0.744**	0.689	**0.744**	0.471	0.494	0.686
grass	**0.991**	0.977	**0.991**	0.922	0.842	0.944
rocks	0.242	0.267	0.242	0.154	0.12	0.183
sand	0.006	0.001	0.006	0.003	0.002	0.002
shrubs	**0.253**	0.208	**0.253**	0.195	0.148	0.247
snow/ice	0.088	0.034	0.088	0.013	0.02	0.114
trees	0.978	0.954	0.978	0.862	0.807	0.965
bridge	0.167	0.187	0.227	0.047	0.057	0.178
building	0.731	0.718	0.792	0.558	0.502	0.698
dam / levee	0.035	0.095	**0.168**	0.033	0.091	0.165
pipes	0.035	0.085	0.14	0.03	0.016	0.007
power lines	0.1	0.205	**0.346**	0.041	0.085	0.237
railway	0.067	0.036	**0.256**	0.057	0.015	0.036
radio towers	0.021	0.057	**0.191**	0.008	0.012	0.018
water tower	0.032	0.049	**0.082**	0.005	0.024	0.025
aircraft	0.013	0.048	0.163	0.267	0.016	0.056
boat	0.247	0.333	0.414	0.017	0.089	0.195
car	0.442	0.348	**0.539**	0.026	0.109	0.382
truck	0.335	0.225	0.375	0.012	0.066	0.215
flooding	0.5	0.246	0.5	0.177	0.418	0.169
lake / pond	0.14	0.207	0.14	0.092	0.181	0.223
ocean	0.003	0.04	0.003	0.103	0.006	0.038
puddle	0.041	0.168	0.041	0.031	0.098	0.05
river / stream	0.591	0.643	0.591	0.5	0.577	0.68
road	0.77	0.809	**0.865**	0.606	0.537	0.83
mAP	0.283	0.285	0.333	0.176	0.185	0.275

Table 1. mAP for every feature for each of our submissions. In bold the mAP achieved was the highest in the competition. For features not in the infrastructure and vehicle categories L_VCL_1 and O_VCL_3 have identical mAP.

test video clips provided by the DSDI task for evaluation is presented. It can be seen in the figure 2 that the third submission (O_VCL_3), which utilized the formed object localization annotations, surpassed by large margin the rest. L_VCL_5 and O_VCL_4 performed substantially worse than the other submissions. O_VCL_6 and O_VCL_2 achieve more or less the same performance with L_VCL_1. As mentioned in section 3, the first submission would be a benchmark to compare against. Submission 3 managed to significantly outperform submission 1 achieving 5% higher mAP. For this reason our analysis will focus on this submission. As mentioned before, submission 3 uses Faster R-CNN to detect features belonging in the infrastructure and vehicle categories and then checks for each feature whether at least one bounding box with sufficient confidence score is is found. For the other three categories these submissions are identical. In figure 3 the test precision for the features in the vehicle and infrastructure for submission 1 and 3 is shown. Object detection based submission 3 outperforms classification based submission 1 for all the features in these categories. This occurs despite the fact that the images we annotated to train Faster R-CNN are by far less than the labeled images provided in the original dataset. The LADI dataset consists of aerial images that have a very high information content. For example an image may contain a very large number of buildings, cars, trees etc. As such, we hypothesize that it is quite challenging for the classification network to locate the required features in the image based only on label annotations. For this reason we believe that more dense annotations with bounding boxes are superior for such datasets even if we are only interested in classification and not object detection as was the case in the DSDI task. An extension of this idea would be to use semantic segmentation networks and annotations to handle other feature categories like environment and water that can be segmented in an image. However, due to the high annotation cost of semantic segmentation we were not able to pursue this avenue. Compared to the competition O_VCL_3 performed the best for 10 out the 32 total features, with 7 out these 10 belonging to the infrastructure and vehicle categories, where O_VCL_3 is very competitive. For two more features in these categories, and specifically the "building" and "truck" objects, the mAP achieved is very close to the best of the competition. We believe that the provided results validate the object-guided approach for the vehicle and infrastructure features. These results are demonstrated in figure 4, where the mAP of O_VCL_3 for the features in these 2 categories is compared against the highest achieved mAP by any team for these features. The mAP of our submissions for each of the 32 features is shown in table 1.

5. Conclusion

To conclude, we relied on CNN classification networks and specifically ResNet101. We tried a) 5 different classifiers one for each feature category and b) one classifier with a common ResNet101 backbone and 5 different classifier

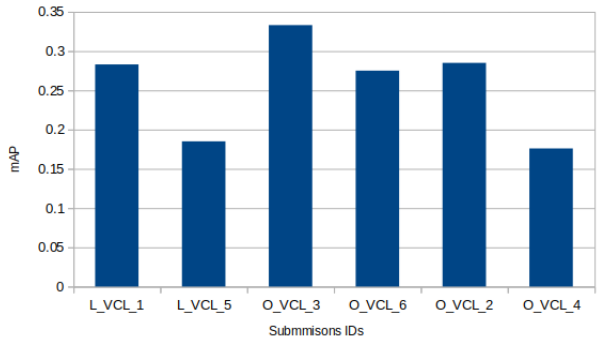

Figure 2. mAP achieved on the test videos by our submissions.

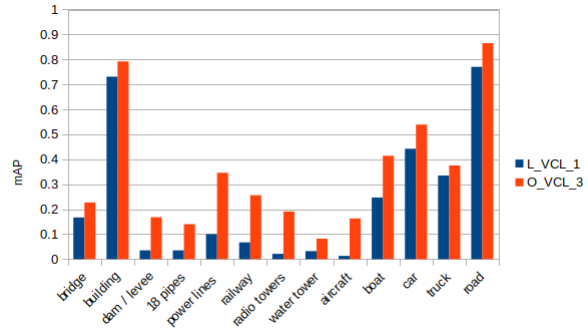

Figure 3. mAP comparison for submissions 1 and 3 for features in infrastructure and vehicle categories.

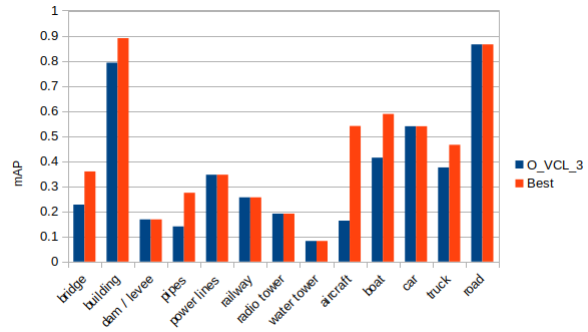

Figure 4. mAP of submission 3 vs best mAP for each feature in the infrastructure and vehicle categories.

heads on top. The first approach performed much better and we used it as a baseline with which to compare our next efforts. Following that, we decided to employ object detection using the Faster R-CNN network. This however required extra annotations with bounding boxes from our side. Object detection was applied only on features in the infrastructure and vehicle categories as features from the other categories can not be precisely located in an image with bounding boxes. A combination of the Faster R-CNN and 3 of the 5 classifiers for the other 3 categories from our initial approach yielded the best results among our submissions. This submission performed very competitively among all teams on infrastructure and vehicle features, highlighting the effectiveness of object detection for this kind of features. Next, we utilized the Faster R-CNN backbone as a feature extractor and 5 classifier heads on top. Finally the features extracted from the Faster R-CNN backbone were refined with masks formed by the Faster R-CNN detected bounding boxes, before passing them to the classifier heads. However, these did not manage to surpass our best submission and performed more or less equally with our baseline.

References

[1] G. Awad, A. A. Butt, K. Curtis, Y. Lee, J. Fiscus, A. Godil, A. Delgado, J. Zhang, E. Godard, L. Diduch, J. Liu, A. F. Smeaton, Y. Graham, G. J. F. Jones, W. Kraaij, and G. Quénot. Trecvid 2020: comprehensive campaign for evaluating video retrieval tasks across multiple application domains. In *Proceedings of TRECVID 2020*. NIST, USA, 2020.

[2] Z. Cai and N. Vasconcelos. Cascade r-cnn: Delving into high quality object detection, 2017.

[3] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[4] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[5] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft coco: Common objects in context, 2015.

[6] J. Liu, D. Strohschein, S. Samsi, and A. Weinert. Large scale organization and inference of an imagery dataset for public safety. In *2019 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–6, Sep. 2019.

[7] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, 2016.