Project CORTEX: Trustworthy AI

Agenium Space

In the context of the project CORTEX, a study was carried out investigate the possibility to associate a confidence score to the predictions of a Deep Neural Network (DNN) in Earth Observation (EO) scenario. This work package studied 2 classification usecases to test such an add-on, on satellite optical and SAR imagery. 17k tiles were cropped in 37 Sentinel-2 products to create a modulable database for the first usecase: binary classification of tiles containing ships or not. 33k downsampled and cropped images were used for the second usecase: classification of ten geophysical phenomena from Sentinel-1 wave mode SAR imagery. In both cases, a classification DNN with VGG16 architecture was trained on a part of the dataset with low capabilities of generalization, and the aim was to associate a confidence score on the predictions made on a harder dataset.

Introduction

Most of the DNNs are designed to predict a class, a segmentation map or detections, no matter it is interpolation or extrapolation. Some paper [1] even thinks that DNNs always extrapolate. Then, a confidence score answer to the need of having interpretable outputs and it could help an AI4EO end-user to take a decision.

Ship S2 AIS detection dataset

The database was generated using Sentinel-2 images distributed through the Copernicus open access hub (https://www.copernicus.eu/en, https://scihub.copernicus.eu/). These Sentinel-2 images are L1C products acquired in Danish sovereign waters in 2019, 2021 and 2022. More specifically, 31 10980x10980 L1C sentinel-2 products were selected (25 for the train dataset, 6 for the test dataset displayed in Figure 1), each of them with a cloud coverage below 10% according to the cloud mask products. The database gathers small (80x80) tiles extracted from Sentinel-2 products.

The reason for selecting this region is the availability of data of ships location: the Danish government made available the AIS (Automatic Identification System) data around Denmark from 2009 until now1. AIS data are tabular data geolocating ships every few minutes. Using the acquisition time of the Sentinel-2 images and the AIS data, image thumbs of ships have been extracted. Negative thumbs have also been created by taking random crops In the Sentinel-2 images (at positions with no AIS signal).

Overall, 1890 tiles containing ships and 11874 tiles stored as 'negatives' have been extracted.

The raster extracted provide 5 bands: Blue, Green, Red, SWIR 1, SWIR 2, that are band 2, 3, 4, 11, 12 from Sentinel2 product. The first 3 bands have a 10m spatial resolution while the last 2 bands have a 20m spatial resolution. Nearest neighbour interpolation was proceeded for SWIR bands in order to match BGR bands spatial resolution.

The tiles are 80x80 pixels images corresponding to 800*800 square meters areas. the labels of the tiles come from the AIS data. Knowing the date and time of acquisition of the Sentinel-2 image, the AIS signals of all ships in a relevant time frame (almost 10 minutes) around the acquisition were extracted. With this process, we obtain several positions for each detected ship and a simple interpolation allows us to deduce the precise position of the ship at the exact time of acquisition. This process ensures to find a ship in the "projected area" with 98% precision for 24x24 neighbourhood, and 99%

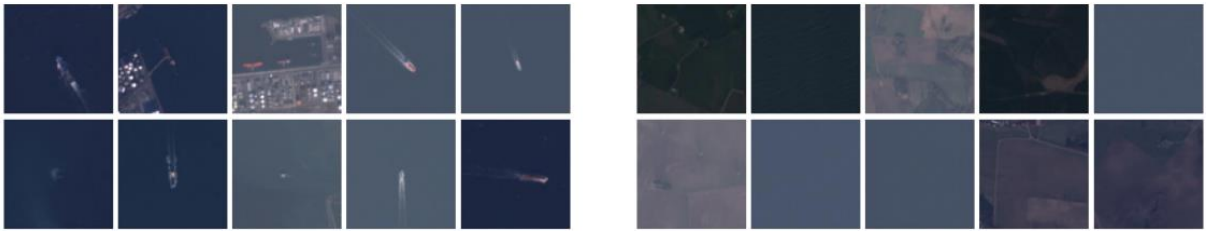

Figure 1: example tiles showing (left) 10 ships and (right) 10 negatives

precision for a 80x80 neighbourhood (800*800 square meters areas). No manual validation of the tiles was performed which means that the database obviously contains errors, but we tried to significantly lower their number regarding the first version of the database (still available on Zenodo).

In order to validate the results and qualify the robustness of the classifier, the inference can be run on the entire Sentinel-2 product after being subdivided into 22500 tiles.

ARD Ocean Features database

The second dataset was already made available by 10.1002/gdj3.73, and was only subsampled by a factor of 2 and cropped at a 224x224 size.

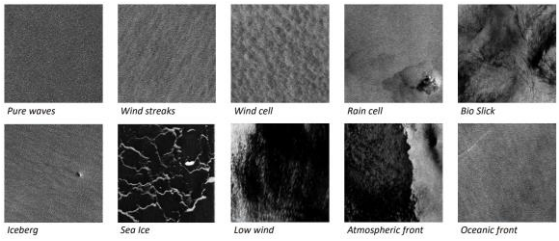

Figure 2: 10 vignette examples of expertly-defined geophysical phenomena (pre-processed)

Investigated methods

Trendy classification or segmentation architectures embed a softmax layer at the end, so that predictions are built like probabilities, this method improves the interpretability of the outputs of a DNN but comes with side effects.

First of all, for example with the ship detection usecase, if a DNN was trained to separate calm sea or ship tiles, what should it predict when seeing an aircraft in an airport ? The fact that the probabilities over every classes sum to 1 implies that there will be a class with bigger probability than 0.5 whether the prediction is trustworthy or not. In correlation with the previously enunciated bias, the network has a high probability of being very confident on his prediction, it is unlikely that it will predict a 50% output for both class. So this unfounded prediction will be hard to track. There is then a need of interpretability of the outputs of a DNN taking more information than "class * is predicted, with a probability of *%".

We then investigated one promising method [1] and compared it with entropy.

Entropy

The entropy can be used to describe the distribution of the output probabilities of a n-classification network (can be extended to semantic segmentation if the implemented method is a par pixel classification).

$$Entropy = \sum_{i=1}^{n} p_i . log(p_i)$$

Remark: with 2 classes, it does not give more information than just taking MCP into account, but with more classes it has good results and is often used to filter predictions that have a too big entropy.

$$MCP = max(p_1, p_2)$$

$$Entropy_{2\ classes} = \sum_{i=1}^{2} p_i . log(p_i) = MCP . log(MCP) + (1 - MCP) . log(1 - MCP)$$

Following the same idea, but since a low probability has a low impact on entropy, other metrics were designed not taking all output probabilities into account but only a few (2, 3, 4, …) of the highest class probabilities. This methods should produce similar results to entropy and we did not investigate.

ConfidNet approach

The main interest of the confidNet approach is that it does not need any retrain from the network we want to assess the confidence. In the example (example from the paper) above, there are 2 models that share a convolutional architecture to extract features from an image. The main idea is that a DNN can characterize In distribution/Out Of Distribution from the latent space and extract more information than it would with the outputs of the networks. Since it is an add-on a main DNN, the aim is to estimate if the prediction is trustworthy or not. The first phase consists in training that main DNN, and then the confidNet will be trained (classifier model layers are frozen) trying to minimize this l2 loss :

$$L_{conf}(\theta;\ D) = \frac{1}{N} \sum_{i=1}^{N} (c(x_i, \theta) - c^*(x_i, y_i^*))^2$$

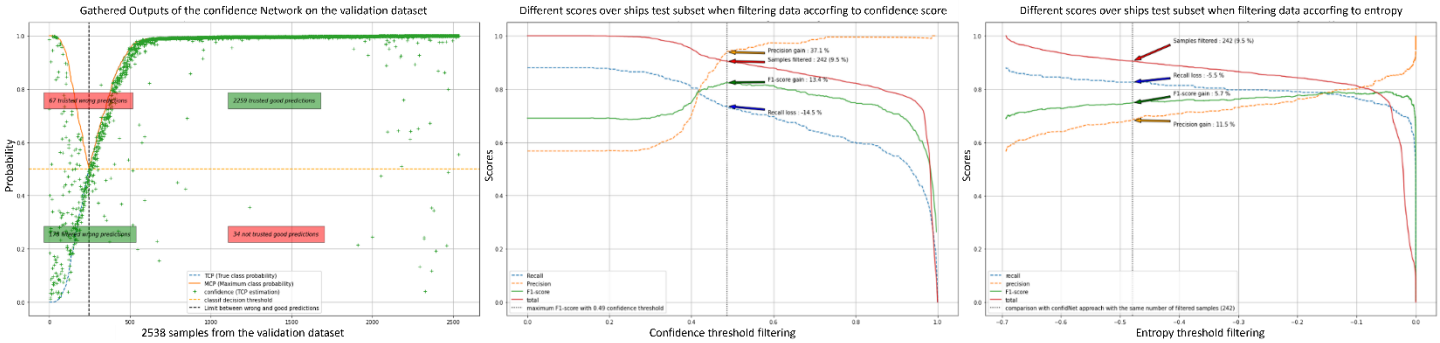

Figure 3: MCP, TCP and TCP estimation on test dataset (left) and scores evolution following confidence threshold filtering (middle), or entropy threshold filtering (right)

Results of the classifying DNN

We trained 2 DNNs with a VGG16 architecture on both the training databases, and get the following results.

Results on the ship detection test dataset : 97% accuracy, 87% good predictions for ships, 98% good predictions for negatives, with easily understandable wrong predictions (such has dense urban areas looking like harbours, off-shore windfarms looking like ships, …)

This network had some similar results when running the prediction over the entire Sentinel-2 products present in the test dataset.

Many false positives have similar patterns, such as windfarms, dense urban areas or coasts. These examples should be easily tagged as "not trustworthy" by the confidNet.

Need of enhancing the database

With 97% of accuracy over the test dataset (98.6% for Ocean Features usecase), predicting the True Class Probability (TCP) is not significatively different (when looking at the entire dataset) than predicting the Maximum Class Probability (MCP), that is why there is a need of enhancing the size of the database to train the confidNet. Indeed, MCP (already predicted by the classifier) does not differ from TCP when predicting the good class, this is why it is needed to add extra tiles with low TCP in both databases. The first database being processed by Agenium Space, we simply downloaded 4 extra tiles over Denmark seas and added 3183 with an overrepresentation (almost 100%) of tiles with low TCP.

For the second usecase, we used the dataset provided by ESA with other samples (3000) of the 10 different classes. As described above, transfer learning from training dataset to this test dataset is hard, and the accuracy on this dataset (70%) should enable the confidNet to learn some untrustworthy examples with this dataset.

Metrics

Although it is almost a theoretical metric, TCP would obviously be a good metric to assess the confidence one can have in a prediction.

$$TCP: (x, y^*) = P(Y = y^*|w, x)$$

We can assess the reliability of a method to estimate TCP by calculating RMSE between TCP and confidence estimation over the whole dataset:

$$RMSE = \sqrt{\sum_{(x,y)} (TCP(x, y^*) - \widehat{TCP}(x, y^*))^2}$$

To go further and extract more tangible metrics, we wanted to estimate if it would be possible to filter some of the predictions according to the predicted confidence. We then used classical $Recall$, $Precision$, $F1 - Score$ metrics to measure the impact of performances of the network when adding the possibility of an "unpredicted" output (ground truth unchanged, but the prediction is not taken into account, as if another column was added to a confusion matrix for example).

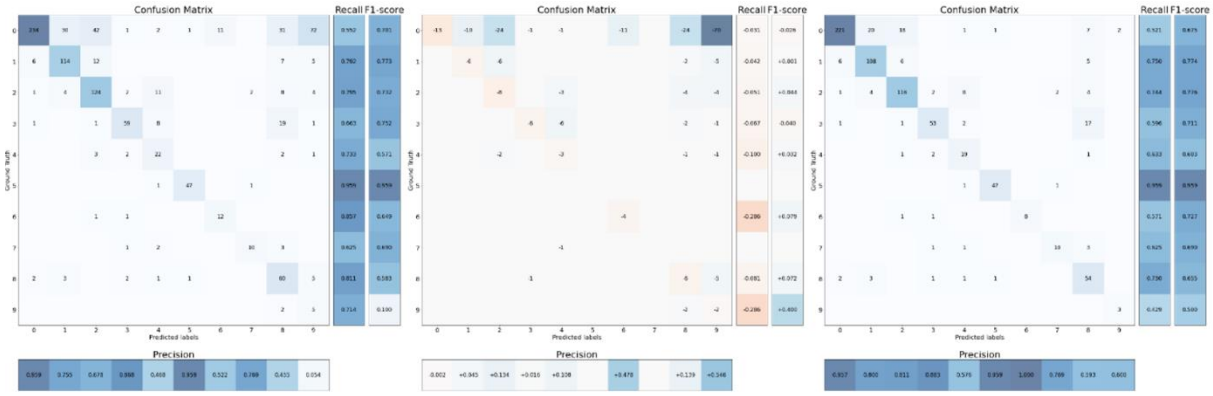

Figure 4: Visualization of the confidNet filtering via confusion matrixes (CM). CM before filtering (left), CM after filtering (right) and "difference CM" showing better the impact of filtering filtering 235 (23%) predictions (middle).

Results of the ConfidNet

A confidNet, as described by [Corbière et al 2019], was implemented and trained for a few epochs for both usecases. The confidNet could relatively well regress the TCP of the samples when trained on the second dataset (with many hard samples). Still, its best result appeared to be in a filtering scenario, with its capability to ignore not trusted predictions to improve the precision without too much recall loss. This results are displayed in **Erreur ! Source du renvoi introuvable.** for the first usecase. The results for the second usecase are displayed in Figure 4, it did not reach so high performances than for the ship detection usecase, but it could still enable an end-user to improve the chosen metrics over the test dataset.

One can also see in **Erreur ! Source du renvoi introuvable.** that confidNet approach outperformed entropy in a context of filtering not trusted predictions.

Scaling the ConfidNet

In an on-boarding scenario, it is important to estimate the scalability of the confidNet approach and how to design the model to get the best results without using an oversized architecture. First implementation [Corbière et al 2019] was made with 3 intermediate layers and 400 features by intermediate feature maps (around 2M parameters), so we tried to reduce the size of the model by reducing the number of parameters N_p.

$$\mathrm{N\_p} = ((\mathrm{N\_in} + 1) + (\mathrm{N\_i} + 1)\mathrm{N\_l} + 1) * \mathrm{N\_i} + 1$$

With N_l intermediate layers, N_i intermediate feature map dimension, and N_in input dimension.

1050 trainings were performed and although N_i was the most influent parameter for the size of the network, it appeared that N_l impacted more the performances of the network, which enabled the confidNet to have similar results with a 70x smaller architecture. Since the training time is short, it can be interesting to try several configurations to find the architectures that fits best with the project.

Conclusions

The study showed promising results in the context of trustworthy AI4EO. The confidNet approach was able to train for a 10-classes and a binary classification usecase, with SAR imagery and Sentinel-2 10m images. The main difficulty encountered is that the confidence network needs many wrong examples of predictions to regress the TCP. It can be a problem for a classification network with outstanding performances. It is also difficult to precisely assess how much the proposed confidence score is close to a good confidence estimation. However, we can see that it outperforms entropy in a filtering usecase for example.

This approach is easy to set up and trains in a reasonable time, so it is easy to test several configurations to get the best-fitted architecture. The depth of the confidNet seems to be the most decisive hyperparameter for the performances it can reach. In a follow-up, it would be interesting to try it on a network with even smaller latent space to reduce the size of the DNN and test the limits of the method. It would also be interesting to try this method on a segmentation usecase, and one can be confident it would work well.

Bibliography

[1] Randall Balestriero, Jerome Pesenti, Yann LeCun, 'Learning in High Dimension Always Amounts to Extrapolation", arXiv:2110.09485

[2] Corbière, Charles, et al. "Addressing failure prediction by learning model confidence." Advances in Neural Information Processing Systems 32 (2019)