Coronal jets identification using Deep Learning as Image and Video Object Detection

Adrien Joliat
École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland - Machine Learning and Optimization (MLO)
Supervised by Martin Jaggi and Sophie Musset

***Abstract*—This report presents a study on the development and application of a Region-based Convolutionnal Neural Network, Faster RCNN and a more complex one, TransVOD, to locate solar coronal jets using data from the Solar Dynamic Observatory (SDO). The study focus on jet detection on image and video.**

***Keywords*: Solar Jets, Faster RCNN, TransVOD, Computer Vision, Video Object Detection, Solar Flares, Solar Physics.**

I. Introduction

In recent years, there has been an exponential increase in the volume of data available for research in solar physics. This surge is largely due to the launch of the Solar Dynamic Observatory (SDO) in 2010, equipped with the Advanced Imaging Assembly (AIA) instrument. The SDO/AIA enables the acquisition of full-disk images of the Sun with a temporal resolution of 12 seconds, resulting in an immense data stream.[1]

In the field of coronal jets, creating a consistent and reliable database for these events is crucial for advancing research in various solar phenomena.

These events were being manually reported in the Heliophysics Events Knowledgebase (HEK).[2] However, given the overwhelming volume of data, manual inspection by experts is neither feasible nor efficient.

A citizen initiative, Solar Jet Hunter was launched in 2021 in the platform Zooniverse.[3] In this project, volunteers were shown a series of movie strips of different regions of the Sun, with the goal of identifying solar jets. This approach, requiring minimal training for participants, significantly scaled up the analysis of solar data. By September 2023, approximately 21% of the data from 2011 to 2016 had been processed, leading to the identification of 881 solar jets by volunteers. However, a substantial portion of the HEK catalogue still remains unclassified.

While some algorithms have been developed to detect and track some of the features in this data, these have not been very successful at detecting solar jets. This is believed to be due to the fact that jets have various shapes, sizes, duration, and brightness. In particular, when seen on the solar disk, they sometimes are not brighter than the background.[4]

This report investigates the potential of a novel machine learning algorithm designed to capture and locate the presence of coronal jets in videos of solar images. This model, trained by data obtained from the citizen-driven Solar Jet Hunter initiative, aims to automate and enhance the classification process for the vast array of data within the HEK catalogue.

Coronal jets are easily detectable by humans due to their dynamic motion, but present a challenge for algorithmic detection as a single image may not contain sufficient information to detect solar jets. Faster RCNN is an image based model and for this reason, the following report won't go in depth about this model as it performed poorly on video data.[5] [6] [7] The fundamental problem with frame detection models applied to video is that they do not capture the temporal changes between frames and so will not be able to capture jet events effectively. Some new models such as $Bal - R^2CNN$ attempt to develop this aspect of recurrent detection with Faster RCNN notably, but they are still under development and video detection models seem more adequate for this task.[8][9][10]

This report will first go through the implementation of Faster RCNN and will follow with the whole development of a model,TransVOD Lite (Transformers for Video Object Detection) applied with Solar data.[11] This model has been selected for it's capacity to give fast outputs and showed amazing results for other task such as object identifications for common objects.[12][13]

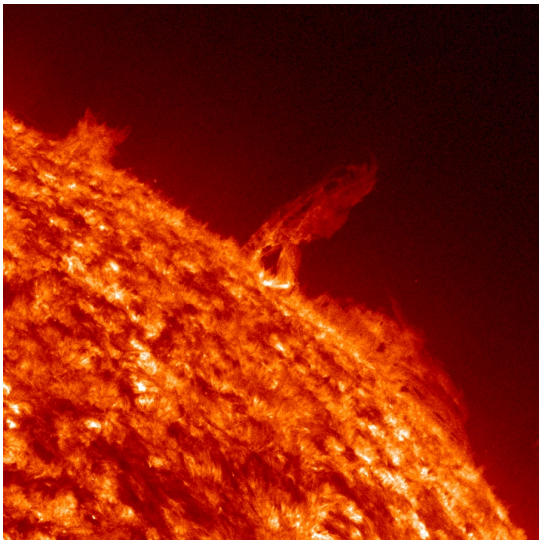


Fig. 1: Solar coronal jet event in the HEK catalogue

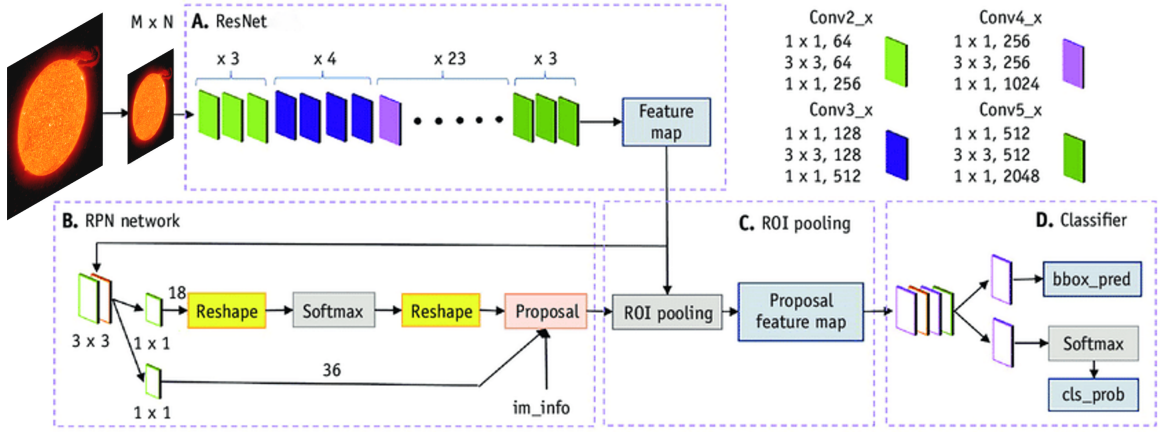


Fig. 2: Faster RCNN architecture

II. Faster RCNN

This first section covers the fine-tuning of a Faster RCNN to detect solar jets. This popular model is based on a single frame and can consequently be applied independently to each frame of a video to extract the overall output for that video. The architecture of the model is shown in Figure 2 and is quite simple:

A. ResNet-50 Backbone
Specific architecture of Residual Networks (ResNets) which is based on deep convolutionnal neural network (CNN). This backbone consist of 50 consecutive layers.

B. RPN network
Region Proposal Network that generate candidates object bounding boxes (region proposal) in an image.

C. ROI pooling
Region Of Interest pooling mix both previous information layer to extract finals regions proposal.

D. Classifier
The final component returns the bounding boxes and their class probability by applying the Softmax function.

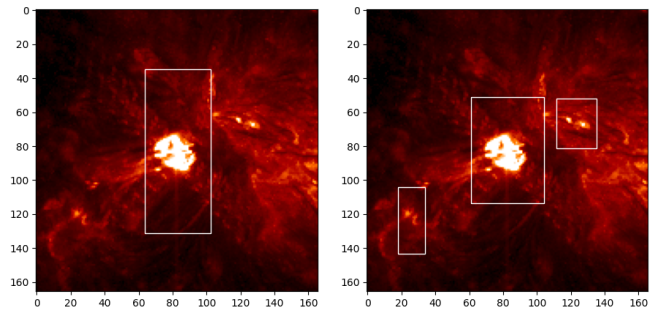
Using the weights of this model trained on the COCO (Common Object in Context) dataset, which contains 200,000 images labelled for 80 classes, this model base is reused to fine-tune the target model.[14] [6]

Next, the Faster RCNN model architecture is modified by changing the last two features (ROI clustering and classifier) to design the wished output. The ROI clustering has to be trained again and the classifier now contains only two classes (one for the solar jet and one for the background).

Finally, the model is trained on a dataset created during a previous project with two other colleagues and adapted to new bounding boxes.[15] Because the dataset was small, the quality was low and the bounding boxes were large, it was not optimal for this task, but it was the only dataset in existence at the time (see section III for a new and significantly enhanced dataset).

The results were weak and unsatisfactory, as there was no temporal link between the different images in a video, the desired bounding box was incorrectly calculated on a validation dataset and the detection of active regions and solar prominences was much too excessive and unintended. Even the application of an NMS (Non-Maximum Suppression) method to better filter the output of a model by post-processing the results was not sufficient.

This Faster RCNN model revealed that it was not feasible to use an image model to detect coronal jets as they are difficult to identify in a single static image shown in Figure 3. The next steps were to find or develop a new model capable of using temporal information to accurately detect jets and provide temporally consistent results.
These requirements were subsequently demonstrated with a new complex model developed in 2023 called TransVod Lite.[11]



(a) True Dataset (b) Results of the Model

Fig. 3: Faster RCNN Results

III. Data Collection and Preprocessing

The new dataset created for the TransVOD model is more advanced, complete and ideal for training such a complex model. Still based on the Zooniverse volunteer set, it includes dates and locations of 21% of solar data from 2011 to 2016, with the duration and locations of solar jets. The methods for handling this dataset are explained here : [16], [17] and aggregations of citizens notations : [18]. All the events are downloaded again from the JSOC (Joint Science Operations Center) server, but this time they are complete, which means that an event can range from 15 to 450 images and the quality of each image is maintained at maximum.

Sunpy [19] [20] is used for this task, a Python library for solar physics data, to retrieve images from the JSOC server based on specific locations and dates [21]. The images were converted and standardized to JPEG formats for use in the model.

Using this data, we can convert the satellite spatial data into a dataset that can be used for machine learning by transforming the Helioprojective latitude/longitude values into the pixel value corresponding to the corners of the bounding box in each image. As shown in Figure 4, the conversion of spatial coordinates into pixel values is accurate and allows the extraction of new data and precise bounding boxes that are updated on average every 15 images. It should be noted that in the initial dataset the bounding boxes could have an angle of rotation, which is not possible with the model and so must also be recalculated by taking a wider box that covers the entire initial box and colours may change due to the normalisation of pixel values while creating JPEG images.

By downloading all the data from the Citizen project, the total number of events are equal to 781 and consist of a total number of images of 41'054. A total memory of 13.2 GB is necessary to store this data. This is a important increase of data compared of the previous dataset. As it is mentioned in [17], the data provided has been annotated by different people all over the world and can differ from one another. The aggregation prevent this kind of problem but doesn't resolve everything. Also, this dataset has been made to contain a single bounding box per image which may be restrictive as one frame may contain multiple coronal jets. Overall, the dataset is the first one to be able to be applied to vision models.

A JSON file contains information about the image path, the bounding box coordinates and more detailed information such as the id, height, width, area and video id for each image and is needed to load the data. One file is created for training, which contains 90% of the total data, and another for validation, which contains the remaining 10%.

To enhance model robustness and realism, data augmentation techniques are implemented, particularly random rotations and flips. It was crucial to apply the same transformation to every frame within a sequence and their corresponding bounding boxes. This consistency is vital because the primary identifiable characteristic of a jet is its motion, and altering the orientation of images between frames could impede the model's ability to detect it.

Finally, when the model's input data is loaded, it is normalised once again to the mean and standard deviation of the values calculated for the red, green and blue bands on the training dataset and resized for a square frame of size 600×600 pixels.

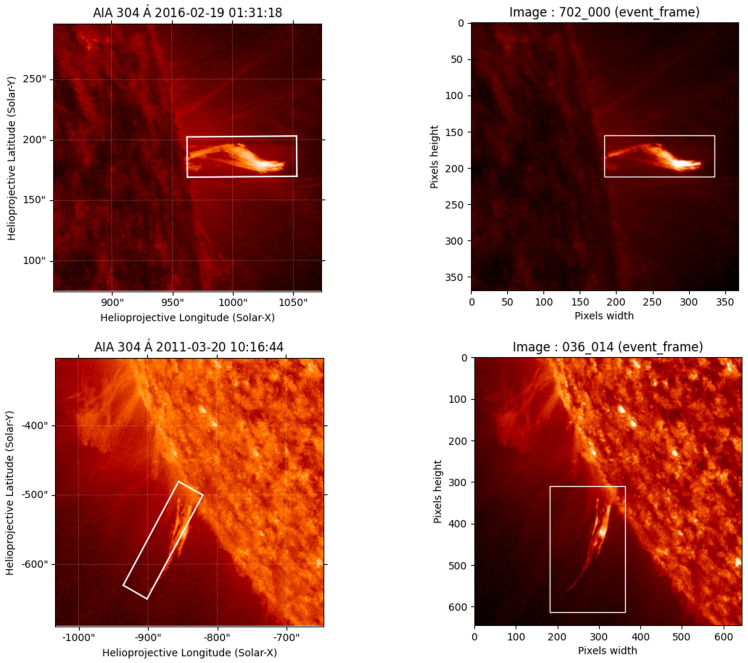


Fig. 4: Examples of modification from the citizen catalogue (left) to a vision model training dataset (right)

IV. Deep-Learning Framework

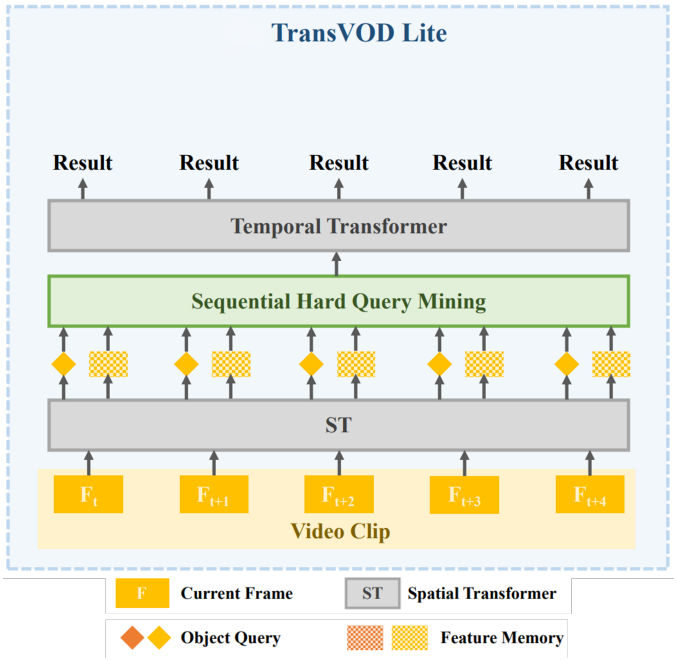


Fig. 5: Structure of TransVOD Lite Model

A. Model Architecture

TransVOD Lite framework is a new end-to-end video object detection based on Transformer architecture. TransVOD adapts the Transformer architecture to video object detection by using a spatio-temporal transformer to learn both spatial and temporal dependencies between objects in a video clip.

The TransVOD Lite framework is based on the input of a video containing a number of consecutive frames. According to the article creating this model, TransVOD Lite receives several images of video clips as input and provides detection results for each image received.[11] It contains four main components: Spatial transformers for object detection for object detection on a single frame, extraction of object queries, and compact representation of the objects. and compact feature representation (memory for each frame), the temporal deformable transformer encoder deformable transformer (TDTE) encoder to merge the memory of spatial memory of the spatial transformers, the temporal query encoder (TQE) to merge the memory outputs of the spatial transformers. (TQE) to link the objects in each image along the temporal dimension and the temporal deformable transform decoder (TDTD) to obtain the final outputs for the current images. The main components are represented in Figure 5. For more details on this complex model, the full model framework is displayed in Figure 7.

Since jet events can be quite short, with only 15 frames in a complete event, the number of frames needed to train the temporal aspect of the model should be set between 12 and 15 frames. This will be detailed in Hyperparameter Tuning T_w.

The model employed in this project is a lighter version of the older TransVOD model [12], because it is much easier to train due to its reduced parameter count (106.3 M parameters) and because its performance is better overall with a shorter inference time. One of the advantages of this approach is that it can be used continuously with satellite imagery and real-time data. The frames per second (FPS) returned by the model are close to 14 FPS and the satellite gives an image every 24 seconds, which means that we are well within the limit of continuous scanning.

B. Hyperparameter Tuning

The selection of hyperparameters such as batch size, learning rate, number of epochs, learning rate scheduler, optimizers, criterion and temporal window size is vital in deep learning optimisation. In addition, the detection threshold parameter also needs to be adjusted, but this will be done when evaluating the model in section VI.

1) Batch size: count of data samples used to compute the gradient and update the model's weights at each iteration. An ideal batch size is problem-specific, and typically depends on experimentation. Smaller batch sizes can introduce a form of generalization due to the fact that it update weights more frequently and may converge faster, but are prone to noisier gradient estimates. Due to the large requirement of the model, the best batch size is set of 1.

2) Learning rate (scheduler): magnitude of weight updates during model training, essentially controlling the step size in the gradient descent process. The learning rate is typically set experimentally and often decreases over time (based on a scheduler), starting with a broader search to approach the area of minimum loss, and then slow down to stabilize and refine the training. A MultiStep learning rate scheduler is used to reduce the learning rate by a factor 10^{-1} each time on some predetermined epochs while the model is being trained.

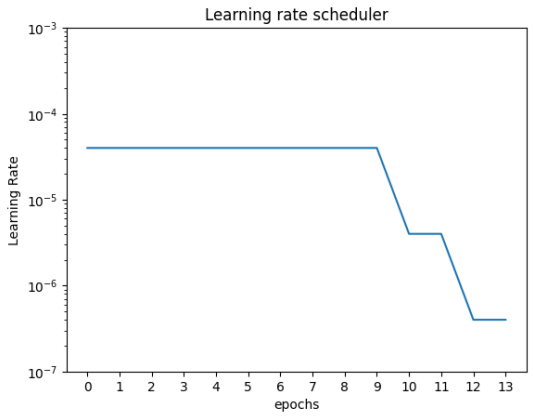


Fig. 6: MultiStep Learning Rate Scheduler

3) Epochs: number of times model's weights are updated by passing through the entire training dataset. The optimal

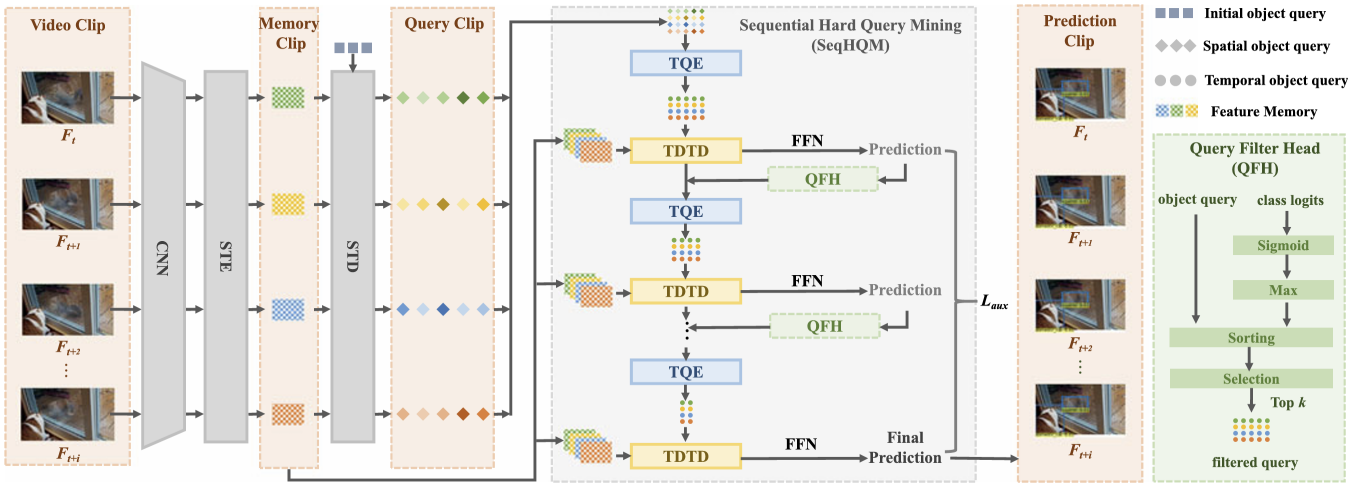


Fig. 7: Detailed TransVOD Lite Architecture

number of epochs depends on various factors, such as the model's complexity, the task's nature, the size and quality of the training data, and the available computational resources. This training does not require a large number of epochs because of the large volume of data, which seems to converge after 14 epochs.

4) Optimizer: determines the update of model parameters during training. The simplest is gradient descent, which adjusts weights against the loss gradient. The selected optimizer is AdamW which is the most widely used in the literature. Combines momentum, adapting LR, and regularization through weight decay to avoid overfitting.[22]

5) Loss functions (criterion): difference between the predicted output and actual target values.
It should be initialized with different losses because it produces distinct things (class prediction and bounding boxes):

- Loss classes : Focal loss for classification
- L1 loss
- Loss giou : Loss of generalized Intersect of Union

and the total loss is equal to :

$$\mathcal{L}_{aux} = \lambda_{classes} * \mathcal{L}_{classes} + \lambda_{L1} * \mathcal{L}_{L1} + \lambda_{giou} * \mathcal{L}_{giou} \quad (1)$$

λ are the coefficients for each loss.

6) Temporal Window size, T_w: This parameter, which determines the number of frames of a single video used as a single input, establishes the temporal parameters of the model. As previously indicated, this value must be low because of the brevity of the jet episodes. After training different models, the results for this hyperparameter are revealed to be Figure 8 and therefore set at 14, which seems consistent for this project. The speed of inference and the memory increases when T_w is larger. In this method, it may effectively exploit the GPU's memory to accelerate the inference time.

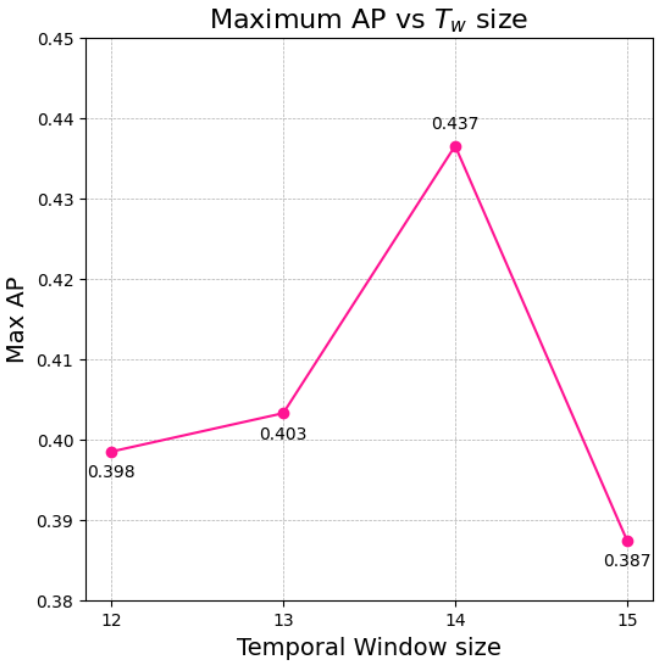


Fig. 8: Temporal Window size against maximum Average Precision

7) Backbone: The backbone is the structure for image detection and classification that can be used with or without pretrained weights. This is the core feature extraction network to produce high-level feature map. Lately, the majority of the new models were using the ResNet-50 network for object detection, image classification or even for segmentation (as seen for Faster RCNN). The Swin backbone, which has just surfaced as a new backbone for these activities, exhibits overall superior performance than the preceding ones. Here, three alternative sizes can be used: Swin Base, Swin Tiny, and Swin Small. The Swin Base one was chosen because of

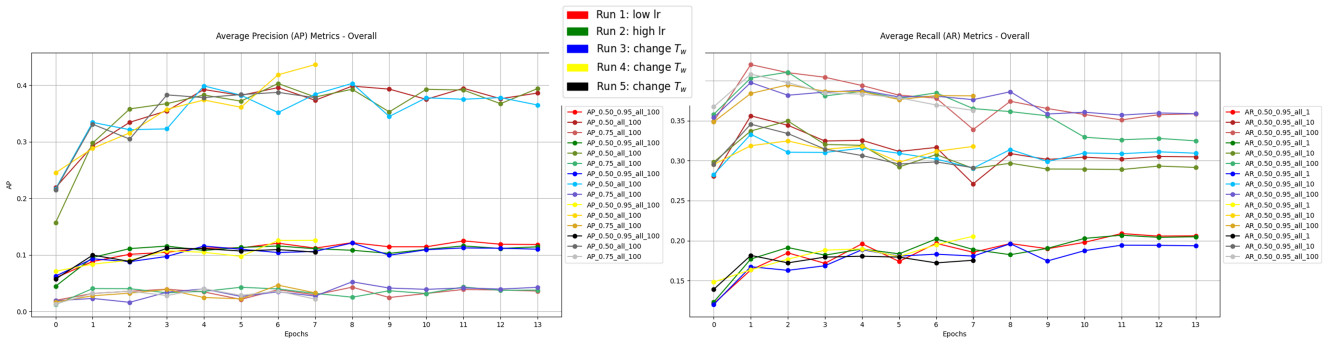


Fig. 9: Cross validation on some hyperparameters

its even higher precision. [23][24][25]

In Figure 9 we can see the results of various runs with different hyperparameters, note that some runs stopped early due to high GPU demand in the EPFL cluster. The overview of the selected hyperparameters can be found in Table I.

Initial LR	4×10^{-5}
Scheduler	MultiStep Learning rate
Optimizer	AdamW
Weight Decay	10^{-4}
Criterion	$\mathcal{L}_{aux}$
Batch size	1
Total epochs	14
Temporal Window, T_w	14
Backbone	Swin Base

TABLE I: Overview of the hyperparameters

V. Model Training

A. General notes

As the training requirements are quite expensive, it needs some preparation and setup. An efficient GPU (graphic processing unit) can compute one epoch in about 4 hours on average, therefore a standard computer cannot complete the training of several epochs. This sophisticated model is trained on the EPFL cluster using multiple GPUs over several days.

The training combine image and video identification. It means that the inputs are shuffled images from the same video with or without transformations applied on all images of the same film. This makes the training more efficient, robust and need less computations.

What's more, the training doesn't start from scratch, we train from a pre-trained model that was created for 30 classes and achieved amazing results (the first class is changed from a plane to a solar coronal jet) which was based on COCO's pre-trained weights. [26]

B. Data from training

As we can see in Figure 10, in the course of learning, different losses are assessed and minimised over time. We will see in the section VI that reducing these values does not necessarily mean obtaining a better model.

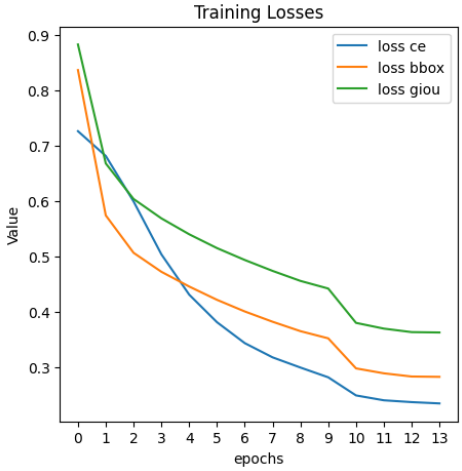
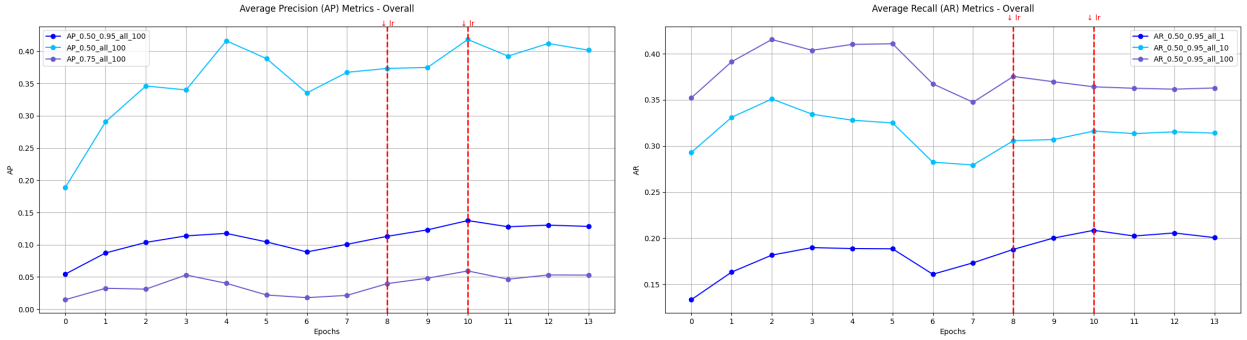


Fig. 10: Training losses over the epochs

VI. Results and Discussion

The results of the final model are displayed in Figure 11 and show a good overall improvement. However, the recall curve does not really benefit from a large number of bounding boxes, which is not really important because a video of coronal jet event often contains one to five jets at the same time, but not a hundred different boxes per image.

As we know, precision assesses the accuracy of positive predictions and the recall curve focuses on capturing all event instances. For this specific project, we would like to maximize the detection of coronal jet events even if it decrease the precision curve. We want to have the minimum false negative classification.



(a) Average precision for a threshold of 0.5 to 0.95

(b) Average recall for different number of boxes (1,10,100)

Fig. 11: TransVOD Lite Results

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{2}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{3}$$

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}} \tag{4}$$

Here, the evaluation metric is based on the IoU (intersection of Union). To match the predictions with the ground truth, the IoU must be calculated for each predicted delimitation box with each ground truth delimitation box. On the basis of the IoU threshold, we classify each prediction as True Positives or False Positives. Any ground truth box that has no corresponding predicted box (based on the IoU threshold) is a False Negative.

These values should also be taken with caution as the dataset does not provide a precise bounding box and has only one box, as mentioned above. As a consequence, we don't want the result to be identical to the initial dataset but it gives an approximation of how well the model fits the data. In the validation dataset it may contains multiples coronal jets but only one is bounded with a box.
In addition, the minimum threshold evaluated here is 50%, i.e. 0.5, and it is therefore possible to lower it even further in order to increase the model's detection capacity, even though this will undoubtedly lead to a net increase in false positives, but it does reduce the number of false negatives. These methods effectively improve recall metrics.

The final model selected is the one of the tenth epoch. It offers a good compromise between precision and recall, with 0.42 precision and 0.37 recall for a threshold between 0.5 and 0.95. These measurements can be seen on the upper curve of Figure 11a and on the lower curve of Figure 11b.

After evaluating the model with new data and different field of view of the sun (notably more far away), the final threshold is set at 0.4 and can provide the desired minimisation and not having too much unwanted false positive while the sun is quiet.

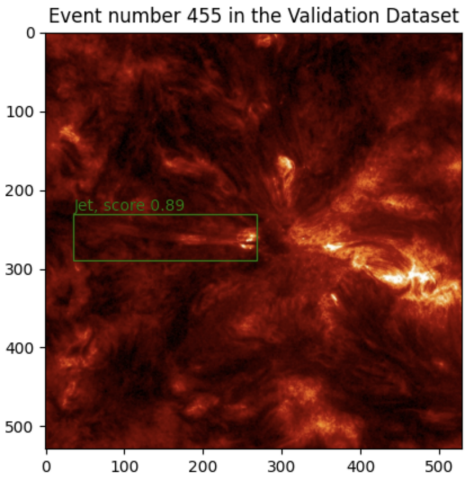


Fig. 12: Output of an on-disk coronal jet event. Click the image for an animated version.

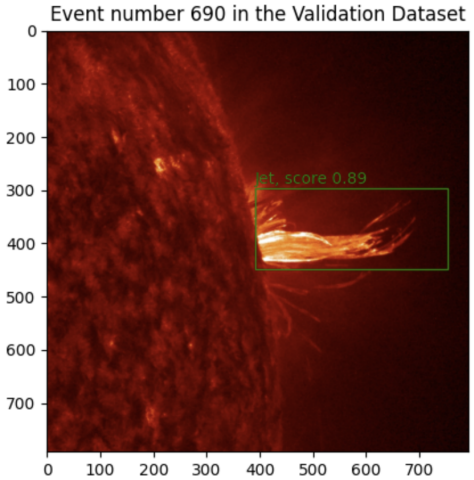


Fig. 13: Output of an off-limbs coronal jet event. Click the image for an animated version.

A variety of animated model outputs are displayed in Figure 12, Figure 13 and here (online). It is interesting to note that in Figure 12, it would have been very difficult to detect this coronal jet on a single image, as is the case there.

VII. Conclusion

In this paper, I have implemented and developed two different models for detecting and locating coronal jets using data from the Solar Dynamic Observatory with AIA instrument at 304 Angstrom.

We saw that the Faster RCNN gave mixed results, but that TransVOD gave very good and reliable scores.

The first dataset for vision models based on the Citizen Solar Jet Hunter project has been created and obtained promising results for future applications. This dataset can be improved by adding ”zoomed out” data, for example images of the whole sun and perhaps data that does not contain jets so that the model is better able to avoid detection, which is slightly lacking because of the need to minimise false negatives. Other improvements can also be made, such as using multiple boxes for a single image and more accurate bounding boxes. In this way, the robustness of the model could be improved and it would be easier to interpret the final precision and recall results instead of using the model on the validation dataset to truly see how it performs.

This model can be used on real-time data and automatically detects and locates the coronal jet all around the solar globe for any size of sequential image input, but performs even more accurately for sharper images around coronal jets.

Generally speaking, it is a development of the new possibilities offered by advances in machine learning models adapted to the field of space research and the understanding of our star, along with space weather. To my knowledge, this project is the first to have been carried out, and the results are more than encouraging.

VIII. Acknowledgment

Many thanks to Sophie Musset and Martin Jaggi, who actively supervised this project !

IX. Appendix

A. Code Repository

For detailed information on my project’s code structure, the libraries utilized, and the implementation of the model in predicting new data, please visit my GitHub repository at https://github.com/adrienjoliat/Jet-detection-and-localisation.git.

B. Selection of model

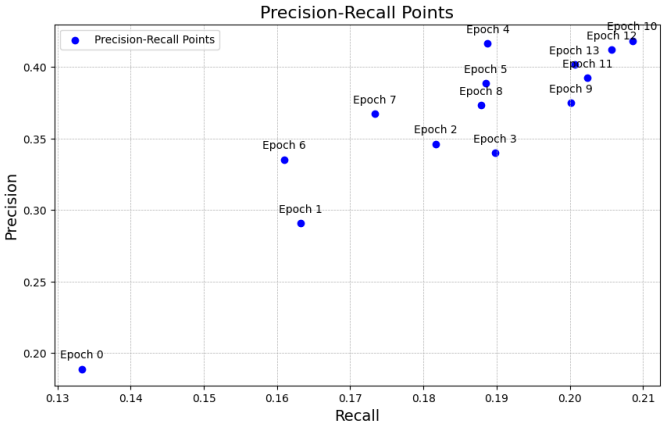


Fig. 14: precision-recall, the more upper right, the best

REFERENCES

[1] J. R. Lemen, A. M. Title, D. J. Akin, P. F. Boerner, C. Chou, J. F. Drake, D. W. Duncan, C. G. Edwards, F. M. Friedlaender, G. F. Heyman *et al.*, "The atmospheric imaging assembly (aia) on the solar dynamics observatory (sdo)," *Solar Physics*, vol. 275, pp. 17–40, 2012.

[2] C. S. N. Hurlburt, M. Cheung. (2010) Heliophysics event knowledgebase and datasearch. [Online]. Available: https://www.lmsal.com/heksearch/

[3] S. Musset, P. Jol, R. Sankar, S. Alnahari, C. Kapsiak, E. Ostlund, K. Lasko, L. Glesener, L. Fortson, G. D. Fleishman, N. K. Panesar, Y. Zhang, M. Jeunon, and N. Hurlburt, "Solar jet hunter: a citizen science initiative to identify coronal jets in euv data sets," 2023.

[4] Y. K. M. D. N. Nishizuka, K. Sugiura and M. Ishii1, "Deep flare net (defn) model for solar flare prediction," 2018. [Online]. Available: https://iopscience.iop.org/article/10.3847/1538-4357/aab9a7/meta

[5] R. T. K. H. Prabu Mohandas, Jerline Sheebha Anni and M. M. Azizan, "Object detection and movement tracking using tubelets and faster rcnn algorithm with anchor generation," 2021.

[6] Y. Varun. (2020) Fine-tuning faster-rcnn using pytorch. [Online]. Available: https://www.kaggle.com/code/yerramvarun/fine-tuning-faster-rcnn-using-pytorch/notebook

[7] J. Schmidt, "Train your own object detector with faster-rcnn pytorch," 2021. [Online]. Available: https://johschmidt42.medium.com/train-your-own-object-detector-with-faster-rcnn-pytorch-8d3c759cfc70

[8] T. G. K. C. X. W. Q. C. F. Z. D. L. N. Y. H. Feng, "Temporal roi align for video object recognition," 2021. [Online]. Available: https://arxiv.org/pdf/2109.03495v2

[9] X. C. H. L. Q. W. F. M. H. Qiu, "Bal-r2cnn: High quality recurrent object detection with balance optimization," 2021. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9382102/citations#citations

[10] B. S. D. A. R. J. D. R. S. Yu-Wei Chao1, Sudheendra Vijayanarasimhan, "Rethinking the faster r-cnn architecture for temporal action localization," 2023.

[11] L. H. Y. Y. L. M. G. C. Y. T. Qianyu Zhou, Xiangtai Li and D. Tao, "Transvod: End-to-end video object detection with spatial-temporal transformers," 2023. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9960850

[12] L. H. Y. Y. G. C. Y. T. L. M. D. T. Qianyu Zhou, Xiangtai Li, "Transvod: End-to-end video object detection with spatial-temporal transformers," 2022. [Online]. Available: https://arxiv.org/pdf/2201.05047

[13] H. Hettiarachchi, "Transvod: The next big thing in video object detection?" *Medium*, 2023. [Online]. Available: https://medium.com/@hansahettiarachchi/transvod-the-next-big-thing-in-video-object-detection-966bb2bdc21f

[14] Pytorch. (2017) Faster r-cnn. [Online]. Available: https://pytorch.org/vision/main/models/faster_rcnn.html

[15] C. C. C. Julie Charlet and A. Joliat, "Reccurent convolutionnal neural network for coronal jet identification," 2023.

[16] S. Musset. (2023) Solar jet hunter blog. [Online]. Available: https://solarjethunter.wordpress.com/author/sophiemusset/

[17] E. Ostlund, S. Alnahari, Y. Zhang, L. Glesener, N. Panesar, S. Musset, M. Jeunon, C. Kapsiak, G. D. Fleishman, L. Fortson, P. Jol, R. Sankar, K. Lasko, and L. Clemmer. (2023) Results solar jet hunter. [Online]. Available: https://www.zooniverse.org/projects/sophiemu/solar-jet-hunter/about/results

[18] S. Musset. (2024) Solar jet hunter catalogue. [Online]. Available: https://github.com/somusset/SolarJetHunter_catalogue

[19] (2024) Sunpy. [Online]. Available: https://sunpy.org/about/index.html

[20] (2024) Astropy. [Online]. Available: https://www.astropy.org/

[21] R. L. K. Musset, Sophie; Sankar. (2023) Solar jet hunter: Jet catalog from hek events 2011-2016. [Online]. Available: https://hdl.handle.net/11299/257209

[22] Pytorch. (2023) Adamw. [Online]. Available: https://pytorch.org/docs/stable/generated/torch.optim.AdamW.html

[23] Y. Liu, "Swin transformer and resnet-50 for object detection and segmentation," *medium*, 2023. [Online]. Available: https://medium.com/@yl9539/swin-transformer-and-resnet-50-for-object-detection-and-segmentation-ee68e3a1493a

[24] Pytorch. (2023) Swin transformer. [Online]. Available: https://pytorch.org/vision/main/models/swin_transformer.html

[25] ——. (2023) Resnet. [Online]. Available: https://pytorch.org/hub/pytorch_vision_resnet/

[26] (2017) Models and pre-trained weights. [Online]. Available: https://pytorch.org/vision/stable/models.html