Mastering Transformers: From Building Blocks to Real-World Applications

Deployment

Prof. Alptekin Temizel
Middle East Technical University
13 Sept 2023

Computer Vision Tasks

Image Classification

Object Detection

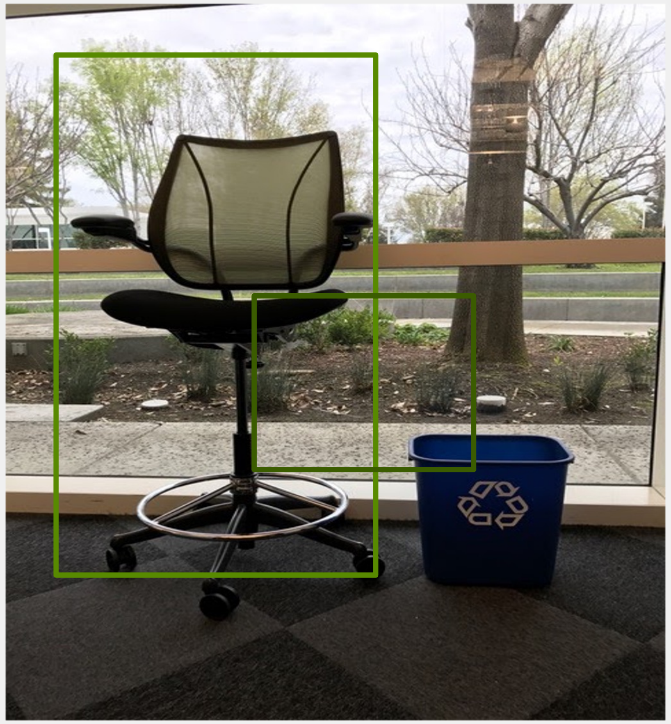
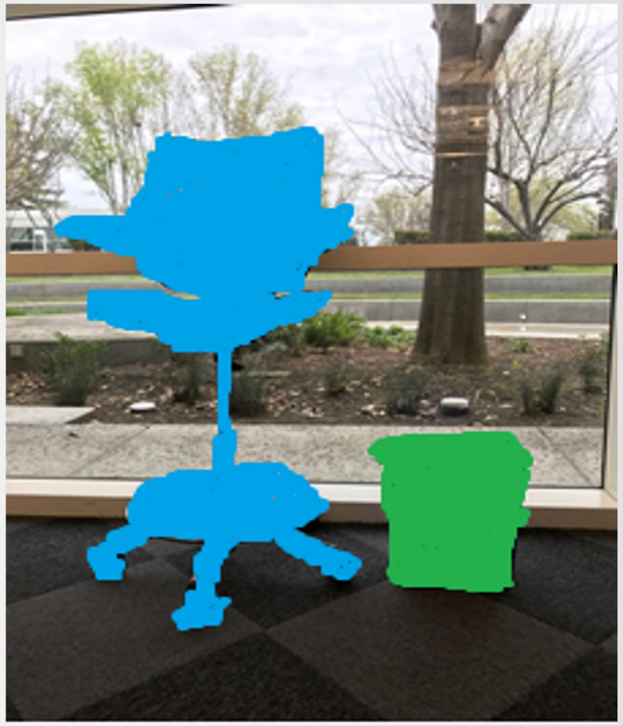


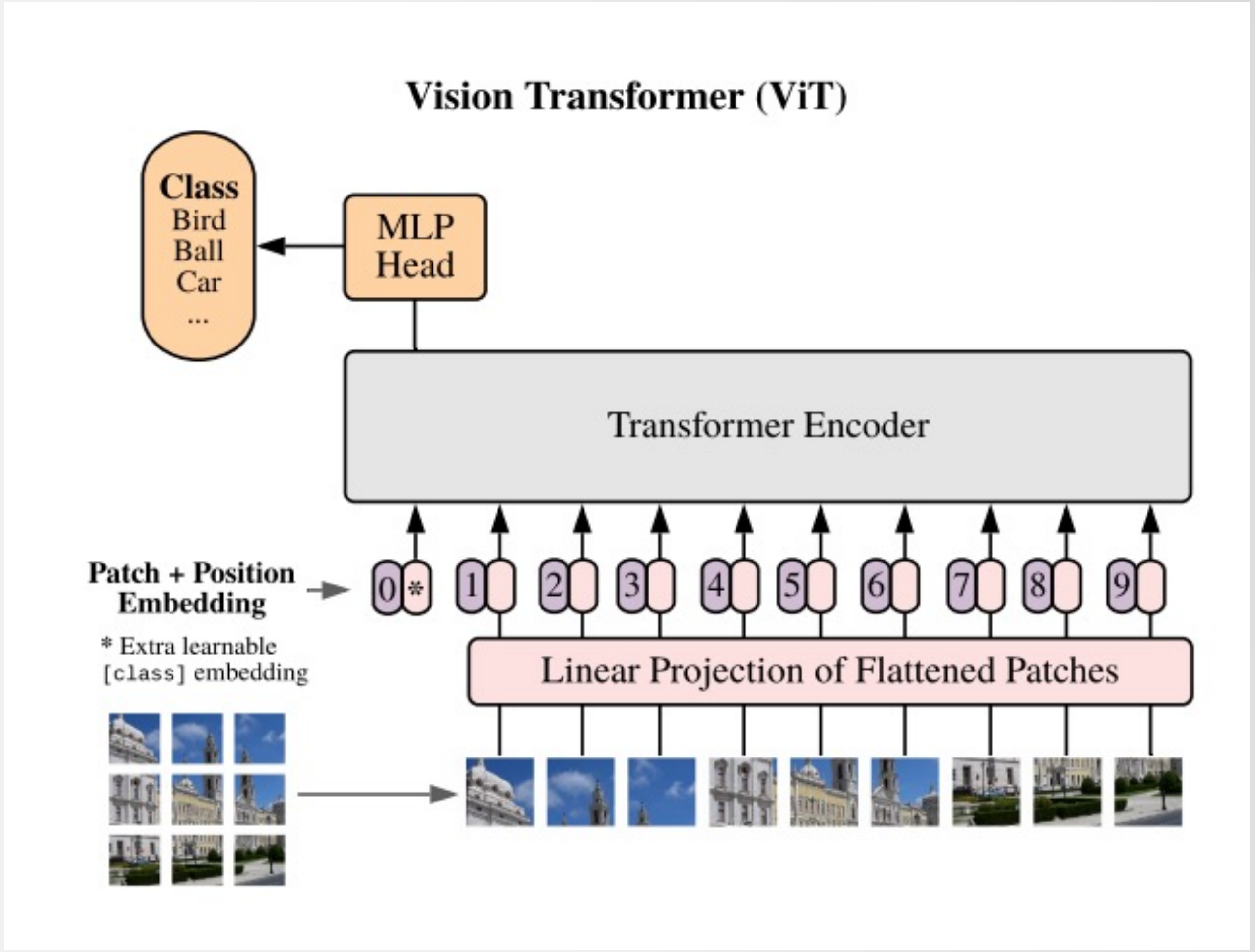
Image Segmentation



(examples are from cs231n lecture from Stanford University)

The Vision Transformer - ViT

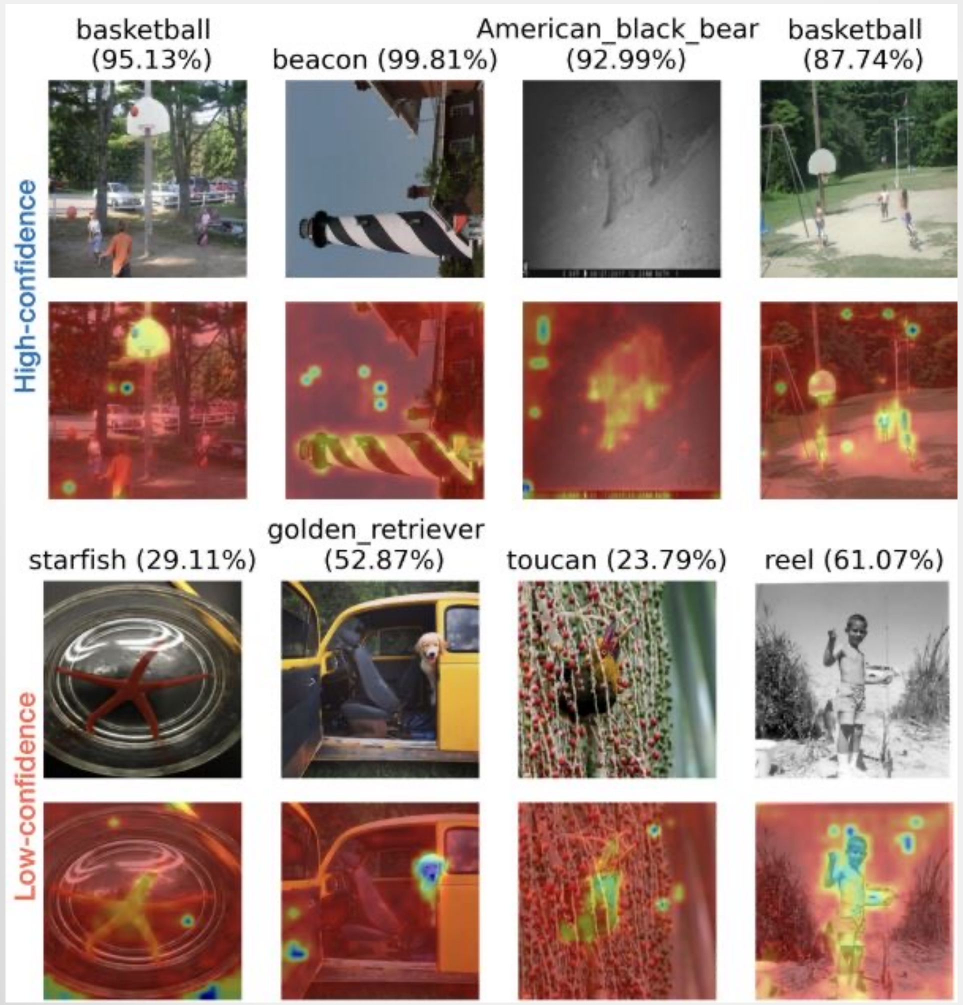
- ViT, is a model for image classification that employs a Transformer-like architecture over patches of the image.
- An image is split into fixed-size patches, each of them are then linearly embedded, position embeddings are added, and the resulting sequence of vectors is fed to a standard Transformer encoder.
- In order to perform classification, the standard approach of adding an extra learnable “classification token” to the sequence is used.



Vision Transformer (ViT)
Class
Bird
Ball
Car
...
MLP Head
Transformer Encoder
Patch + Position Embedding
* Extra learnable [class] embedding
0
*
1
2
3
4
5
6
7
8
9
Linear Projection of Flattened Patches

Dosovitskiy et al., An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale

The Vision Transformer - ViT



Vision Transformers vs. CNNs

Inductive biases allow the learning solution to prioritise one solution over another, independent of the observed data.

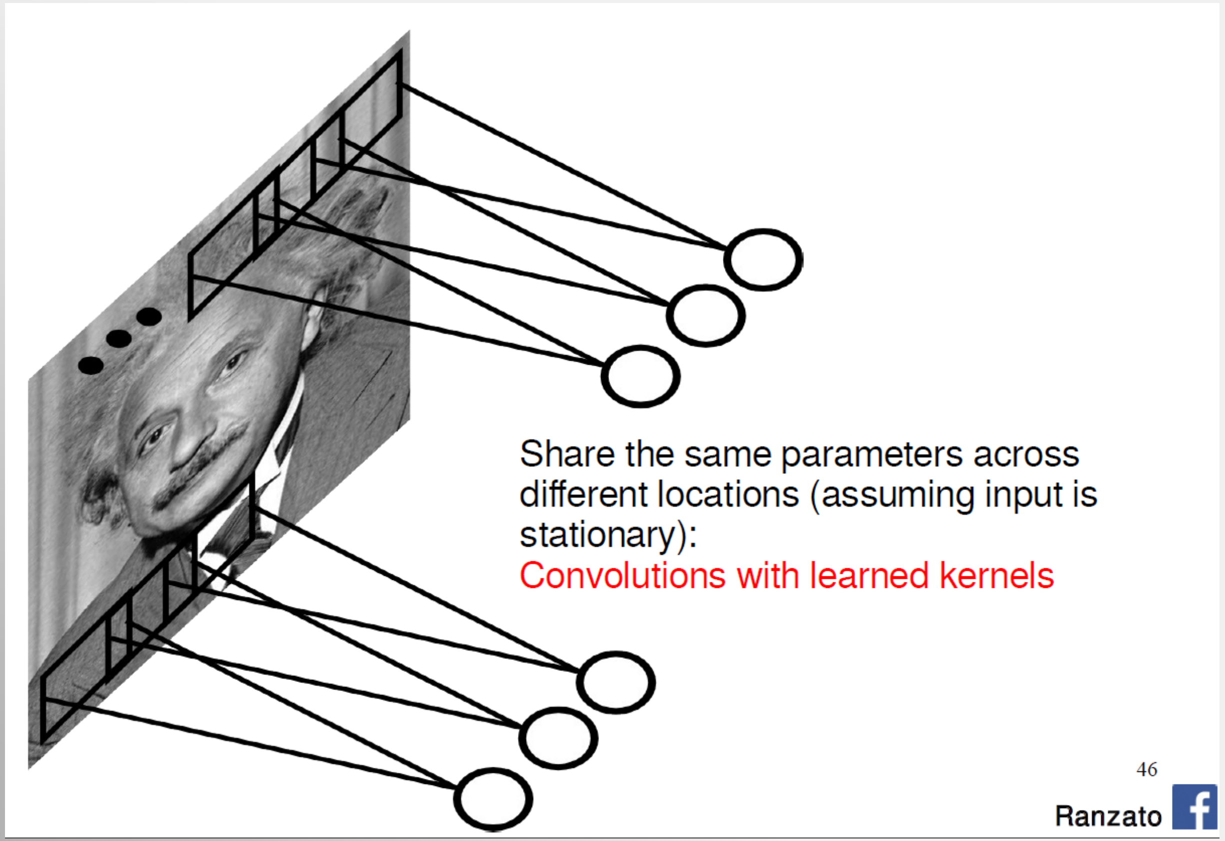
- By constraining the models, it reduces the space of solutions without sacrificing key metrics thus improving performance.
- It allows us to encode our understanding of the world to search for viable solutions. This lets us steers the model generalization the way we want.

https://medium.com/mlearning-ai/everything-you-need-to-know-about-inductive-bias-4850b77048f2

Vision Transformers vs. CNNs

In convolutional neural networks : The inductive bias is on **locality** i.e the features are generated using local pixels and then combined **hierarchically.**

Translational equivariance is another inductive bias for CNNs



Share the same parameters across different locations (assuming input is stationary):

Convolutions with learned kernels

Idea: statistics are similar at different locations - stationarity (Lecun 1998)

Connect each hidden unit to a small input patch and share the weight across space: convolution layer.

https://medium.com/mlearning-ai/everything-you-need-to-know-about-inductive-bias-4850b77048f2

Vision Transformers vs. CNNs

- CNNs achieve excellent results even with training based on data volumes that are not as large as those required by Vision Transformers.
- This different behaviour seems to derive from the presence in the CNNs of some **inductive biases** that can be somehow exploited by these networks to grasp more quickly the particularities of the analysed images even if, on the other hand, they end up limiting them making it more complex to grasp global relations. On the other hand, the Vision Transformers are free from these biases which leads them to be able to capture also global and wider range relations but at the cost of a more onerous training in terms of data.
- Vision Transformers also proved to be much more robust to input image distortions such as adversarial patches or permutations.
- Hybrid architectures combining convolutional layers with Vision Transformers have shown to provide excellent results in Computer Vision.
- On the other hand, weaker inductive bias results in increased reliance on model regularization or data augmentation when training on smaller datasets.

Coccomini, Davide, 2021, "Vision Transformers or Convolutional Neural Networks? Both!". *Towards Data Science*.

ViT Model Sizes

ViT-B (Base): 12 layers, hidden size of 768, 86M parameters.

ViT-L (Large): 24 layers, hidden size of 1024, 307M parameters.

ViT-H (Huge): 32 layers, hidden size of 1280, 632M parameters.

ViT Variants
Data-efficient Image Transformer - DeiT

The original paper concluded that transformers “do not generalize well when trained on insufficient amounts of data”, and the training of these models involve extensive computing resources.

DeiT (data-efficient image transformers) are more efficiently trained transformers for image classification, requiring far less data and far less computing.

DeiT is trained using a teacher-student strategy specific to transformers.

It relies on a distillation token ensuring that the student learns from the teacher through attention.

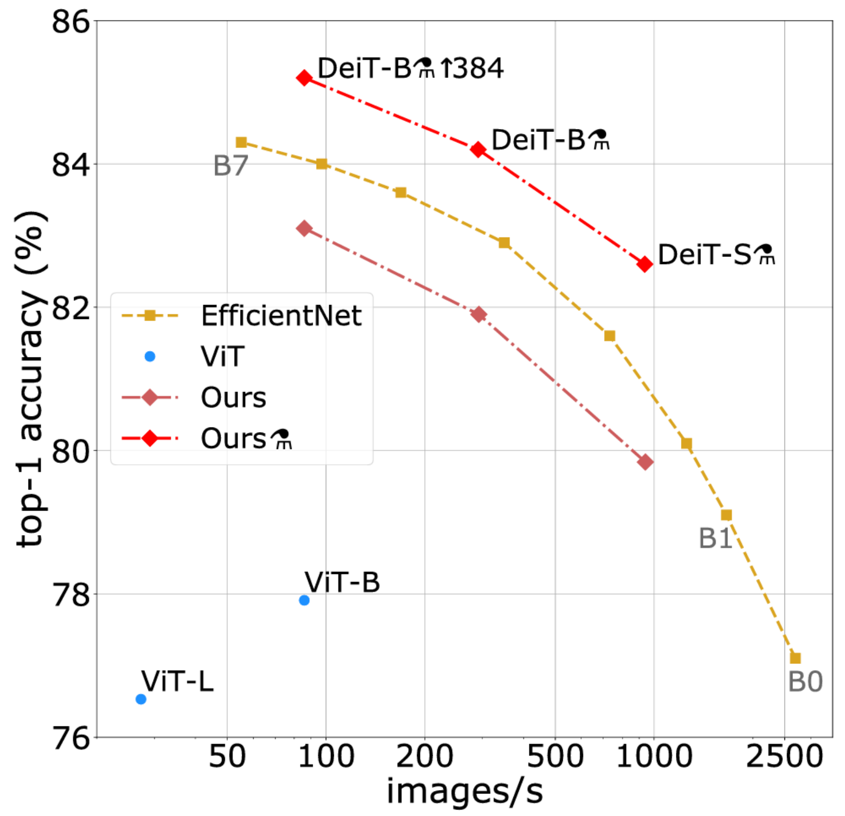


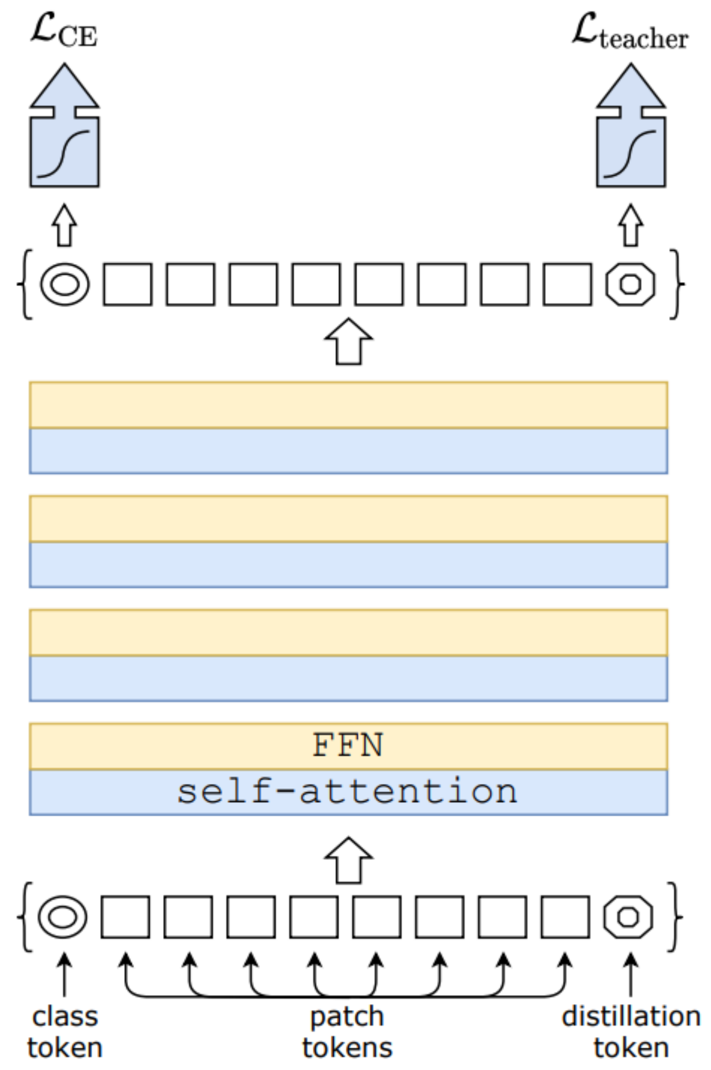
Figure 1: Throughput and accuracy on Imagenet of our methods compared to EfficientNets, trained on Imagenet1k only. The throughput is measured as the number of images processed per second on a V100 GPU. DeiT-B is identical to VIT-B, but the training is more adapted to a data-starving regime. It is learned in a few days on one machine. The symbol ⚗ refers to models trained with our transformer-specific distillation. See Table 5 for details and more models.

ViTVariants
Data-efficient Image Transformer - DeiT

Distillation token interacts with the class token and patch tokens through the self-attention layers.

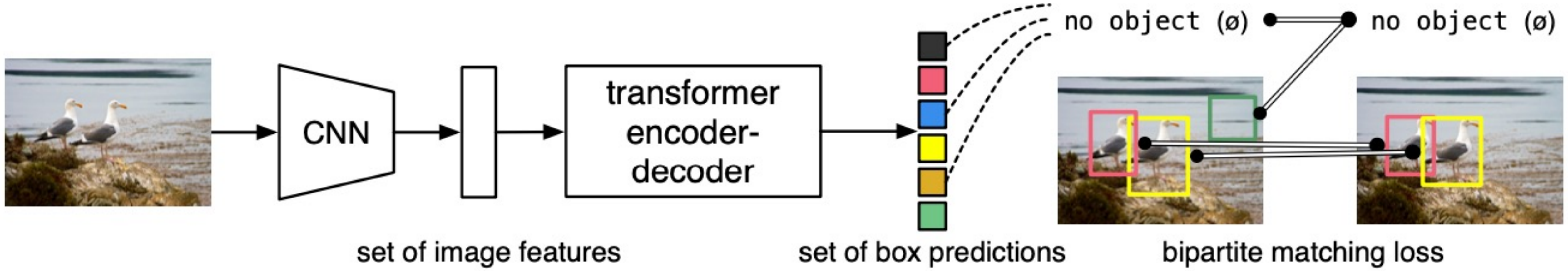
It is employed in a similar fashion as the class token, except that on output of the network its objective is to reproduce the (hard)label predicted by the teacher, instead of true label.

Both the class and distillation tokens input to the transformers are learned by back-propagation.



Object Detection
DEtection TRansformer (DETR)

- DETR introduces transformers to object detection tasks by reframing detection as a set prediction problem, eliminating the need for proposal generation and post-processing steps.
- Initially, despite competitive performance, DETR suffered from slow training convergence and ineffective detection of smaller objects.



Object Detection
DEtection TRansformer (DETR)

- Initially, despite competitive performance, DETR suffered from slow training convergence and ineffective detection of smaller objects.

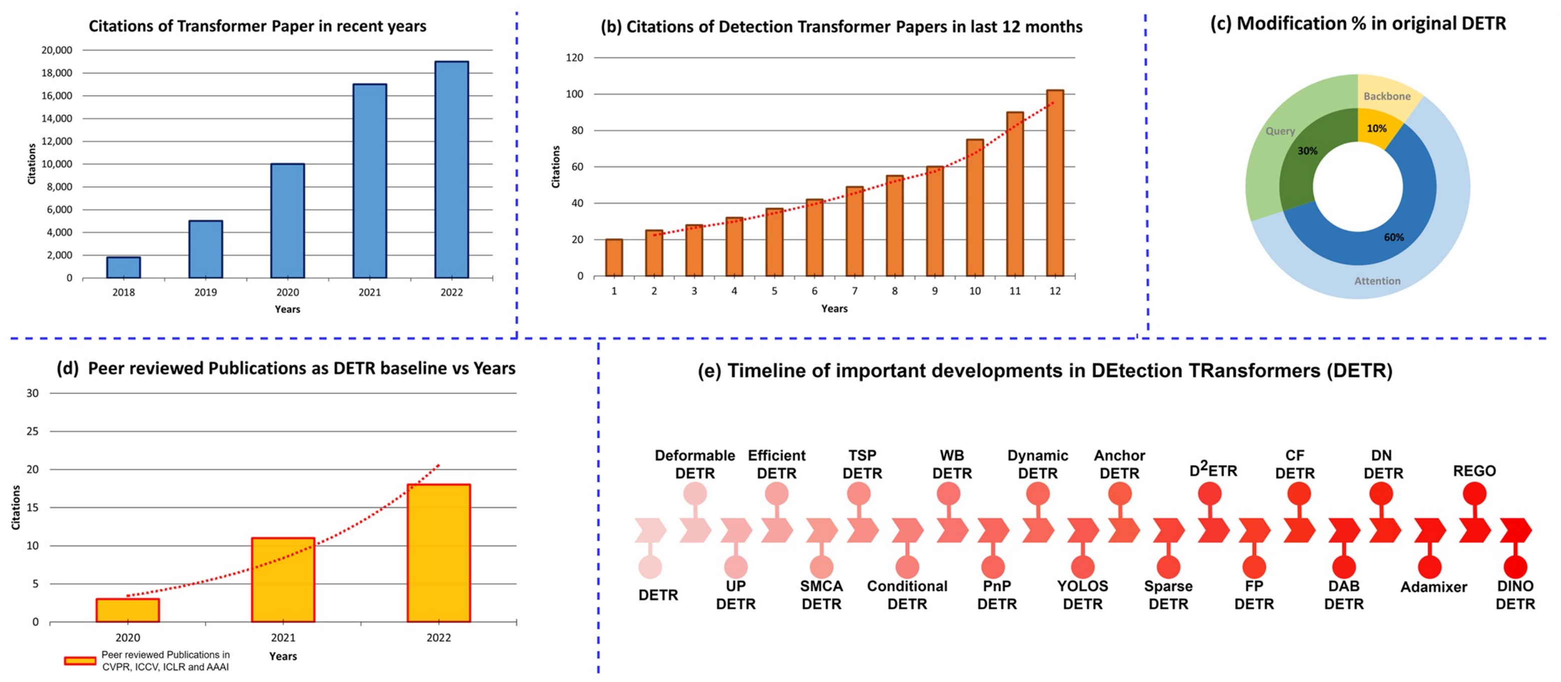


Fig. 1: Statistics overview of the literature on Transformers. (a) Number of citations per year of Transformer papers. (b) Citations in the last 12 months on Detection Transformer papers. (c) Modification percentage in original DEtection TRansformer (DETR) to improve performance and training convergence (d) Number of peer-reviewed publications per year that used DETR as a baseline. (e) A non-exhaustive timeline overview of important developments in DETR for detection tasks.

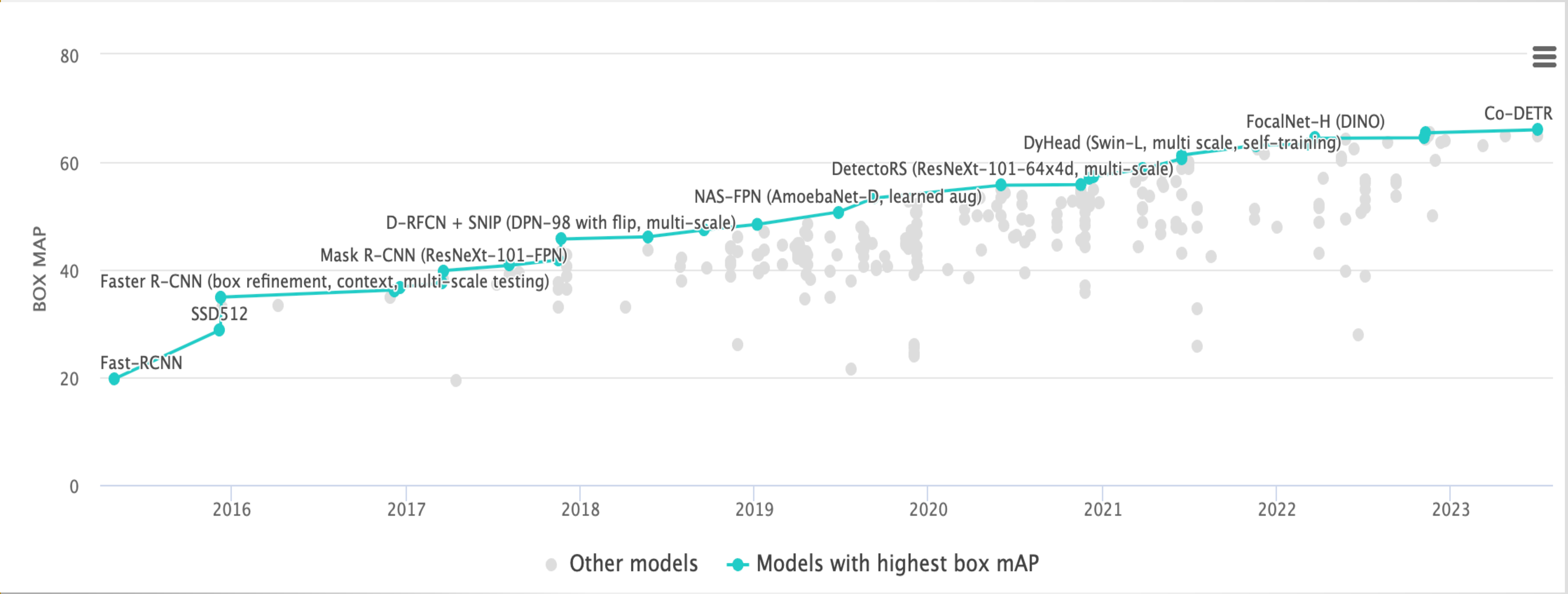
Object Detection
DINO-DETR

DETR with **I**mproved de**N**oising anch**O**r boxes (DINO) improves over previous DETR-like models in performance and efficiency by

- using a contrastive way for denoising training
- a mixed query selection method for anchor initialization
- a look forward twice scheme for box prediction.

Zhang et al., DINO: DETR with Improved DeNoising Anchor Boxes for End-to-End Object Detection, ICLR 2023

Object Detection State-of-the-Art



https://paperswithcode.com/sota/object-detection-on-coco

Object Detection
Co-Deformable-DETR

“Too few queries assigned as positive samples in DETR with one-to-one set matching leads to sparse supervision on the encoder's output which considerably hurt the discriminative feature learning of the encoder and vice visa for attention learning in the decoder.

To alleviate this, we present a novel collaborative hybrid assignments training scheme, namely □o-DETR, to learn more efficient and effective DETR-based detectors from versatile label assignment manners. This new training scheme can easily enhance the encoder's learning ability in end-to-end detectors by training the multiple parallel auxiliary heads supervised by one-to-many label assignments such as ATSS and Faster RCNN. In addition, we conduct extra customized positive queries by extracting the positive coordinates from these auxiliary heads to improve the training efficiency of positive samples in the decoder. In inference, these auxiliary heads are discarded and thus our method introduces no additional parameters and computational cost to the original detector while requiring no hand-crafted non-maximum suppression (NMS).

Zong et al., DETRs with collaborative hybrid assignments training, Oct. 2023, *ICCV*

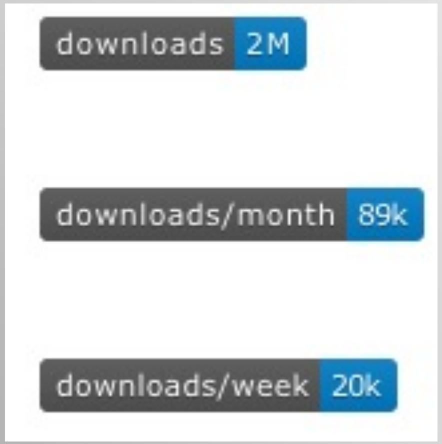
Slicing Aided Hyper Inference (SAHI)

Large scale object detection & instance segmentation

Over 3000 stars on Github

PyPI link

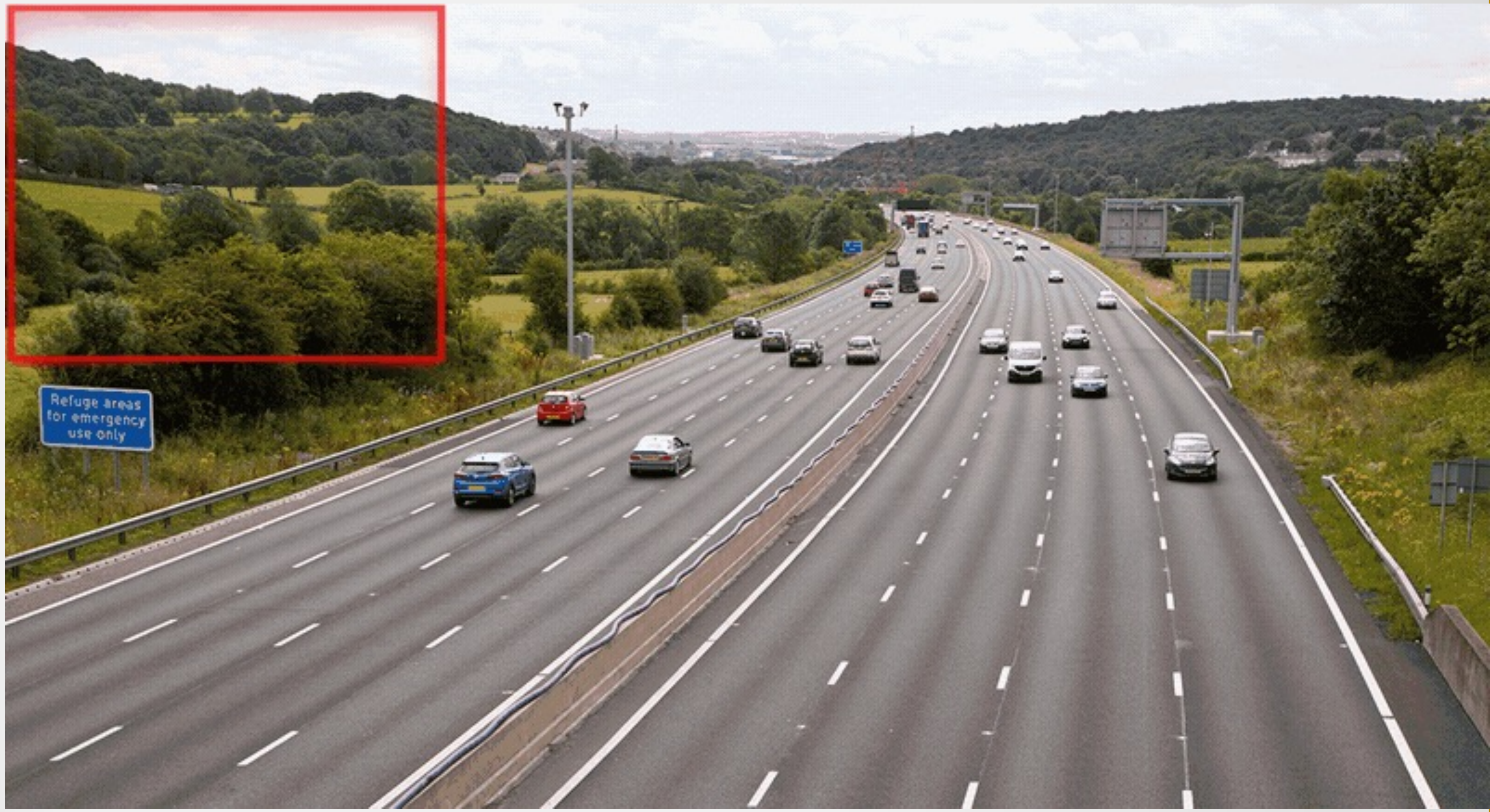
https://pypi.org/project/sahi



downloads 2M

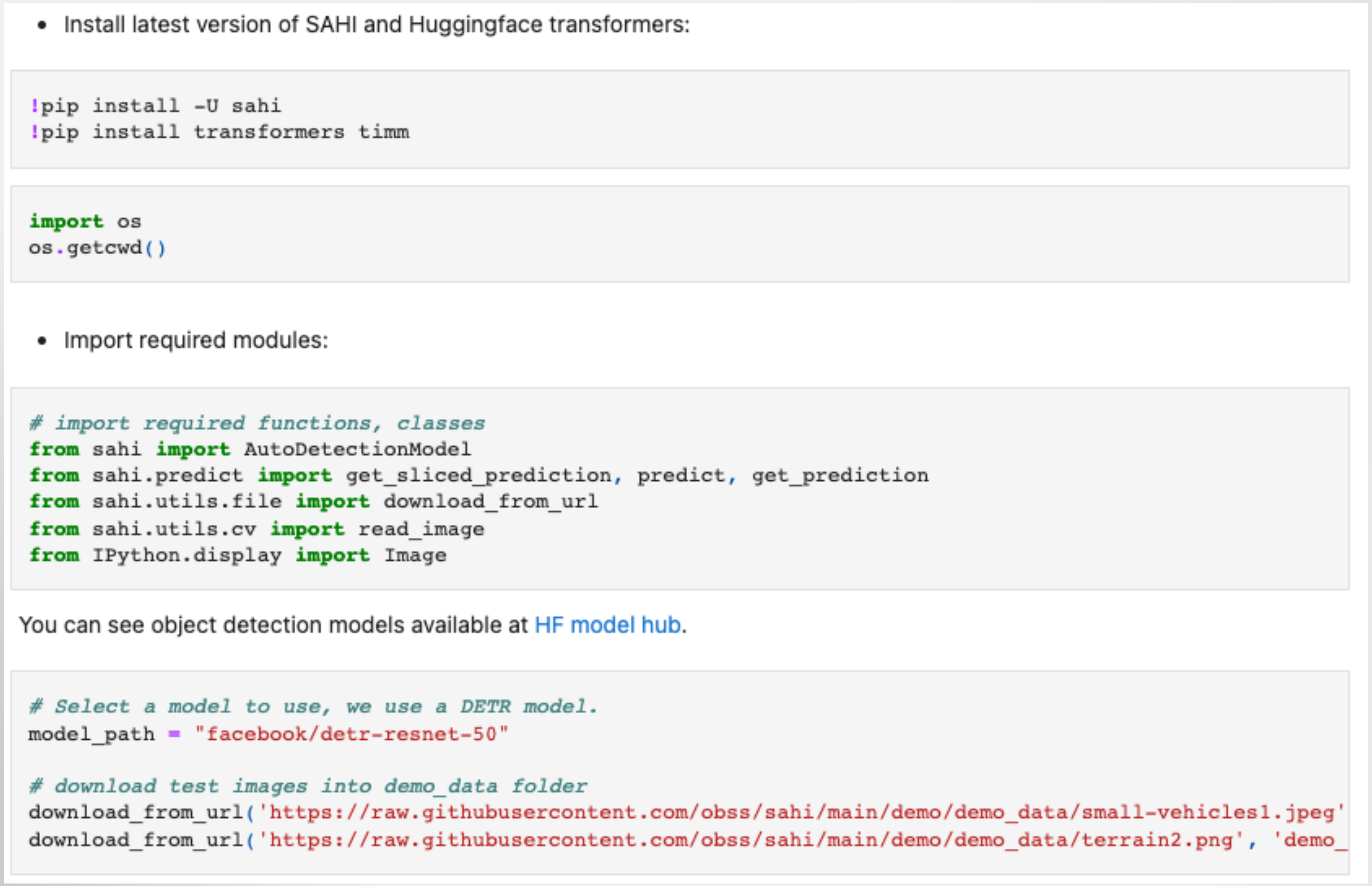
downloads/month 89k

downloads/week 20k



F.C. Akyon, S.O. Altinuc, A. Temizel, "Slicing Aided Hyper Inference and Fine-tuning for Small Object Detection", ICIP, Oct. 2022.

Slicing Aided Hyper Inference (SAHI) DETR Integration



- Install latest version of SAHI and Huggingface transformers:

```
!pip install -U sahi
!pip install transformers timm
```

```
import os
os.getcwd()
```

- Import required modules:

```
# import required functions, classes
from sahi import AutoDetectionModel
from sahi.predict import get_sliced_prediction, predict, get_prediction
from sahi.utils.file import download_from_url
from sahi.utils.cv import read_image
from IPython.display import Image
```

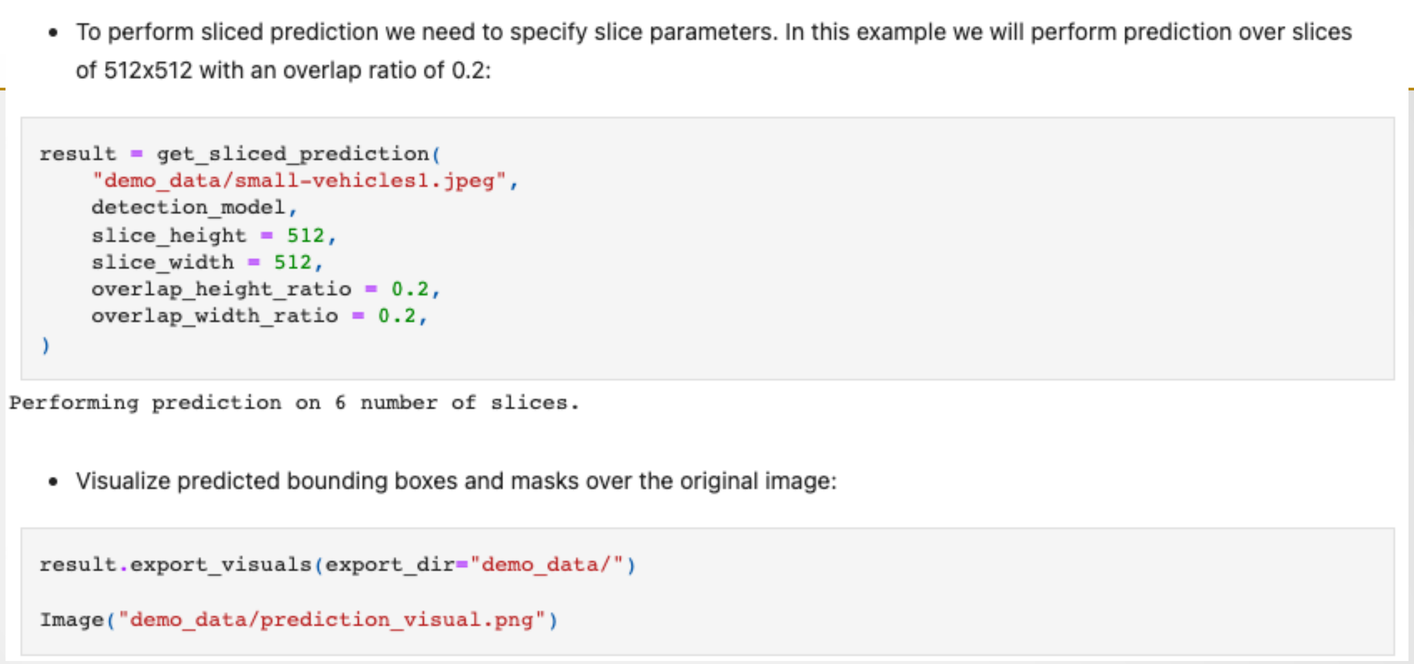
You can see object detection models available at HF model hub.

```
# Select a model to use, we use a DETR model.
model_path = "facebook/detr-resnet-50"

# download test images into demo_data folder
download_from_url('https://raw.githubusercontent.com/obss/sahi/main/demo/demo_data/small-vehicles1.jpeg'
download_from_url('https://raw.githubusercontent.com/obss/sahi/main/demo/demo_data/terrain2.png', 'demo_
```

SAHI Notebook with Huggingface (HF) Transformers
https://github.com/obss/sahi/blob/main/demo/inference_for_huggingface.ipynb

Slicing Aided Hyper Inference (SAHI) DETR Integration



- To perform sliced prediction we need to specify slice parameters. In this example we will perform prediction over slices of 512x512 with an overlap ratio of 0.2:

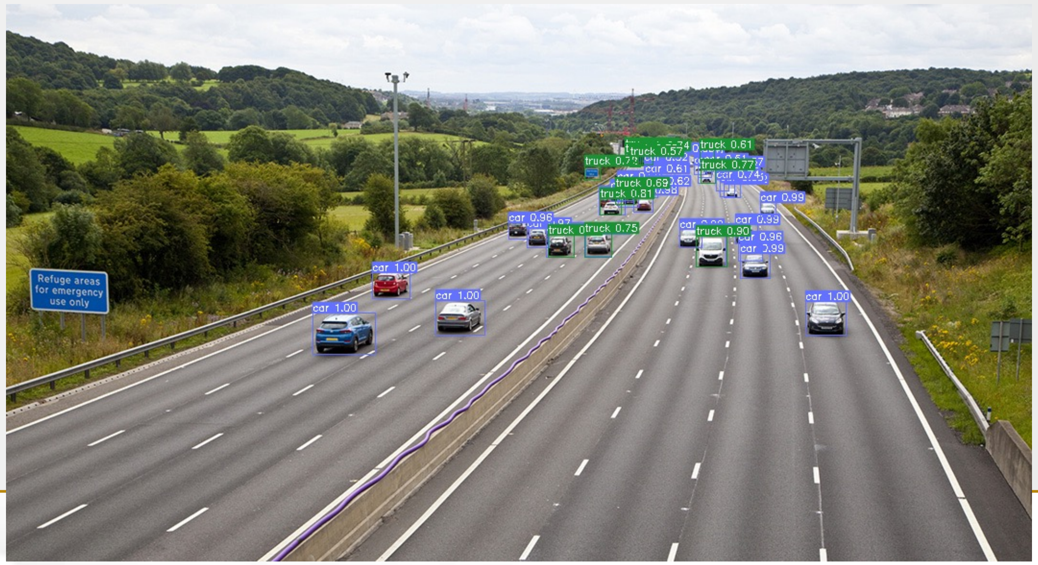
```
result = get_sliced_prediction(
    "demo_data/small-vehicles1.jpeg",
    detection_model,
    slice_height = 512,
    slice_width = 512,
    overlap_height_ratio = 0.2,
    overlap_width_ratio = 0.2,
)
```

```
Performing prediction on 6 number of slices.
```

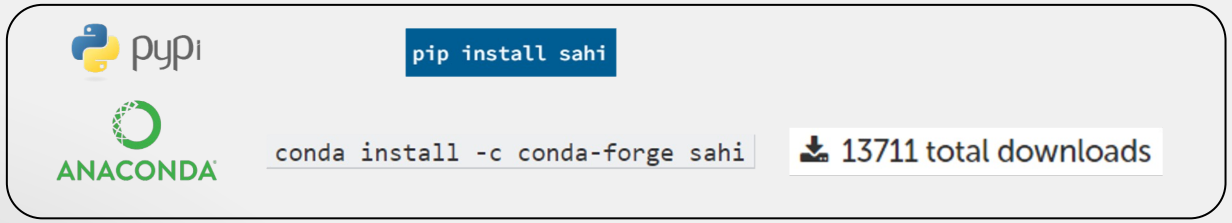
- Visualize predicted bounding boxes and masks over the original image:

```
result.export_visuals(export_dir="demo_data/")

Image("demo_data/prediction_visual.png")
```



SAHI Resources



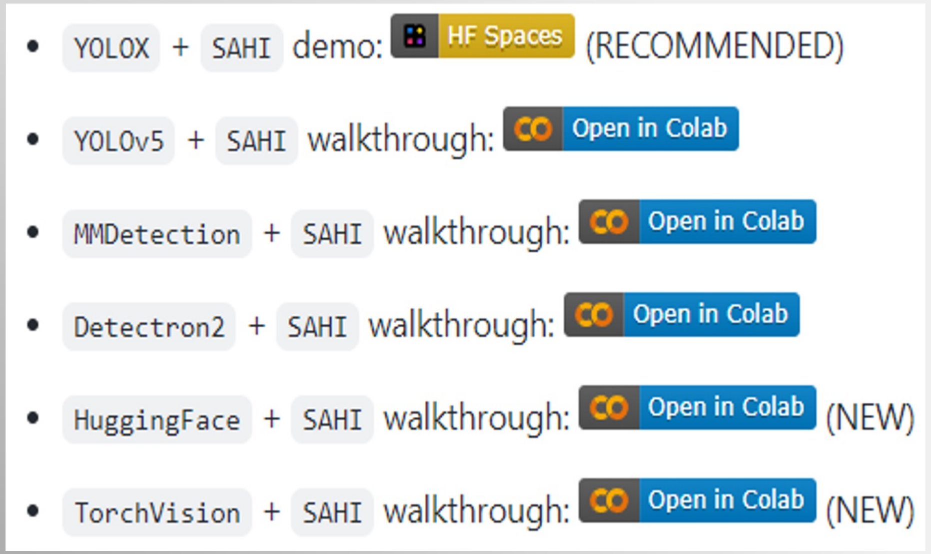
```
pip install sahi
```

```
conda install -c conda-forge sahi
```

13711 total downloads

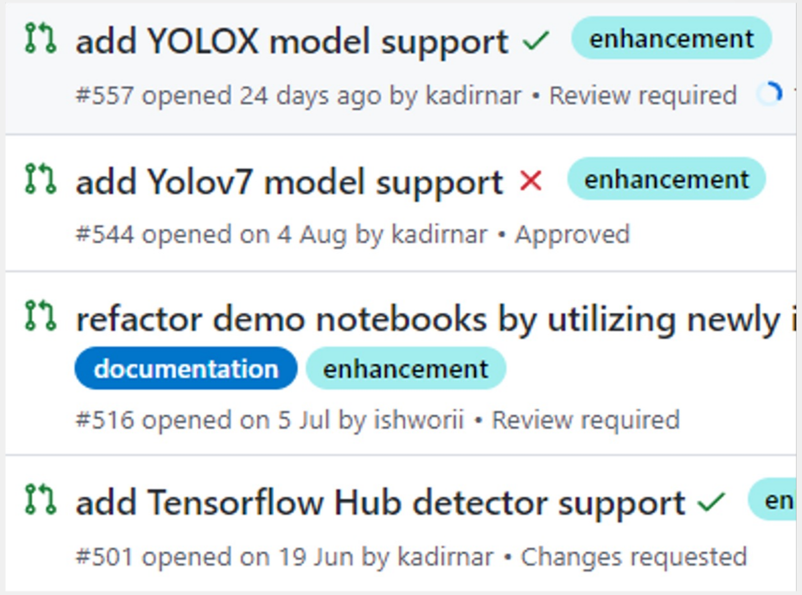
https://github.com/obss/sahi

Supporting Most Trending Detectors:



- `YOLOX` + `SAHI` demo: HF Spaces (RECOMMENDED)
- `YOLOv5` + `SAHI` walkthrough: Open in Colab
- `MMDetection` + `SAHI` walkthrough: Open in Colab
- `Detectron2` + `SAHI` walkthrough: Open in Colab
- `HuggingFace` + `SAHI` walkthrough: Open in Colab (NEW)
- `TorchVision` + `SAHI` walkthrough: Open in Colab (NEW)

More Detector Support In-progress:



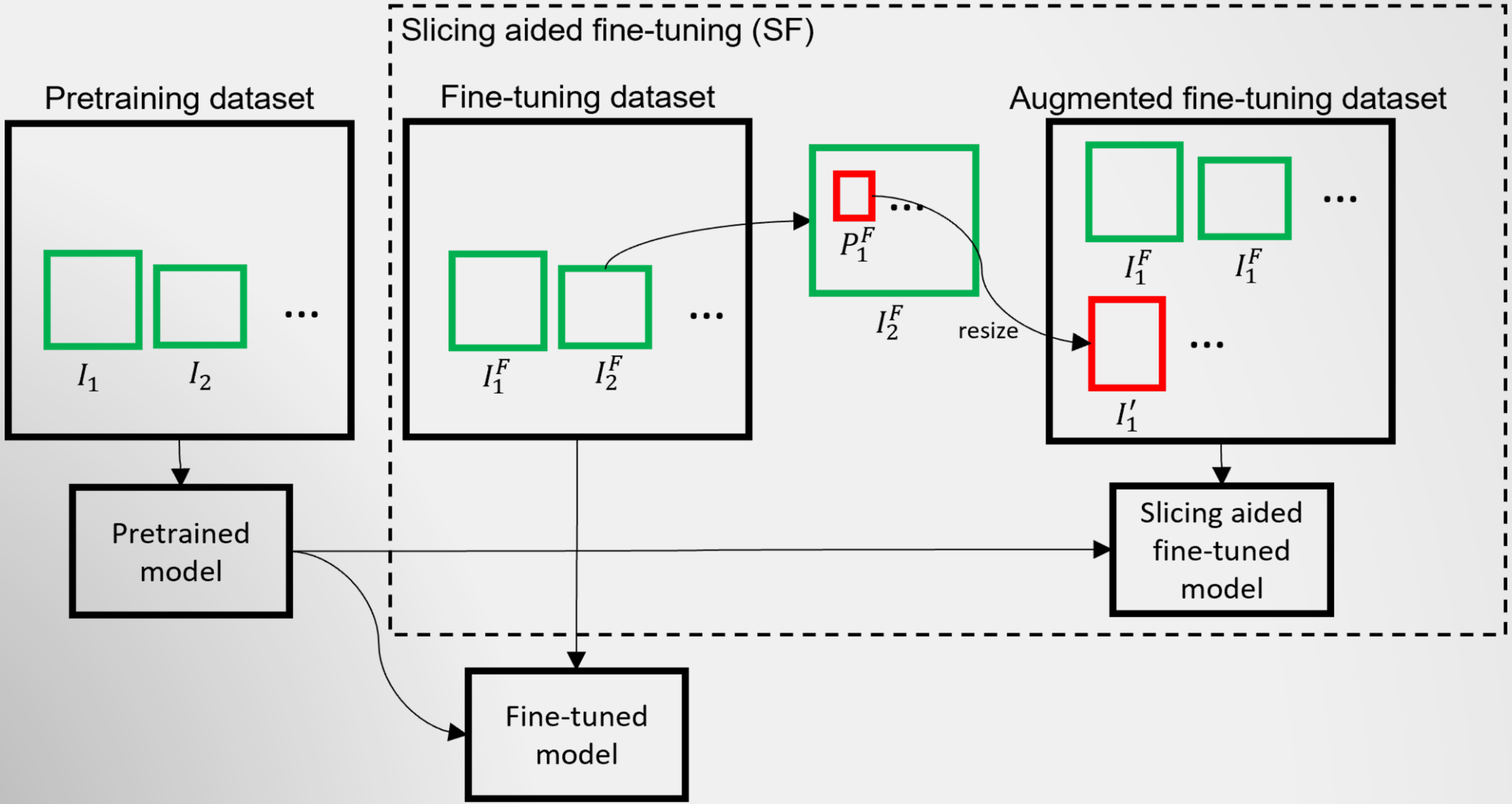
add YOLOX model support ✓ enhancement
#557 opened 24 days ago by kadirnar • Review required

add Yolov7 model support ✗ enhancement
#544 opened on 4 Aug by kadirnar • Approved

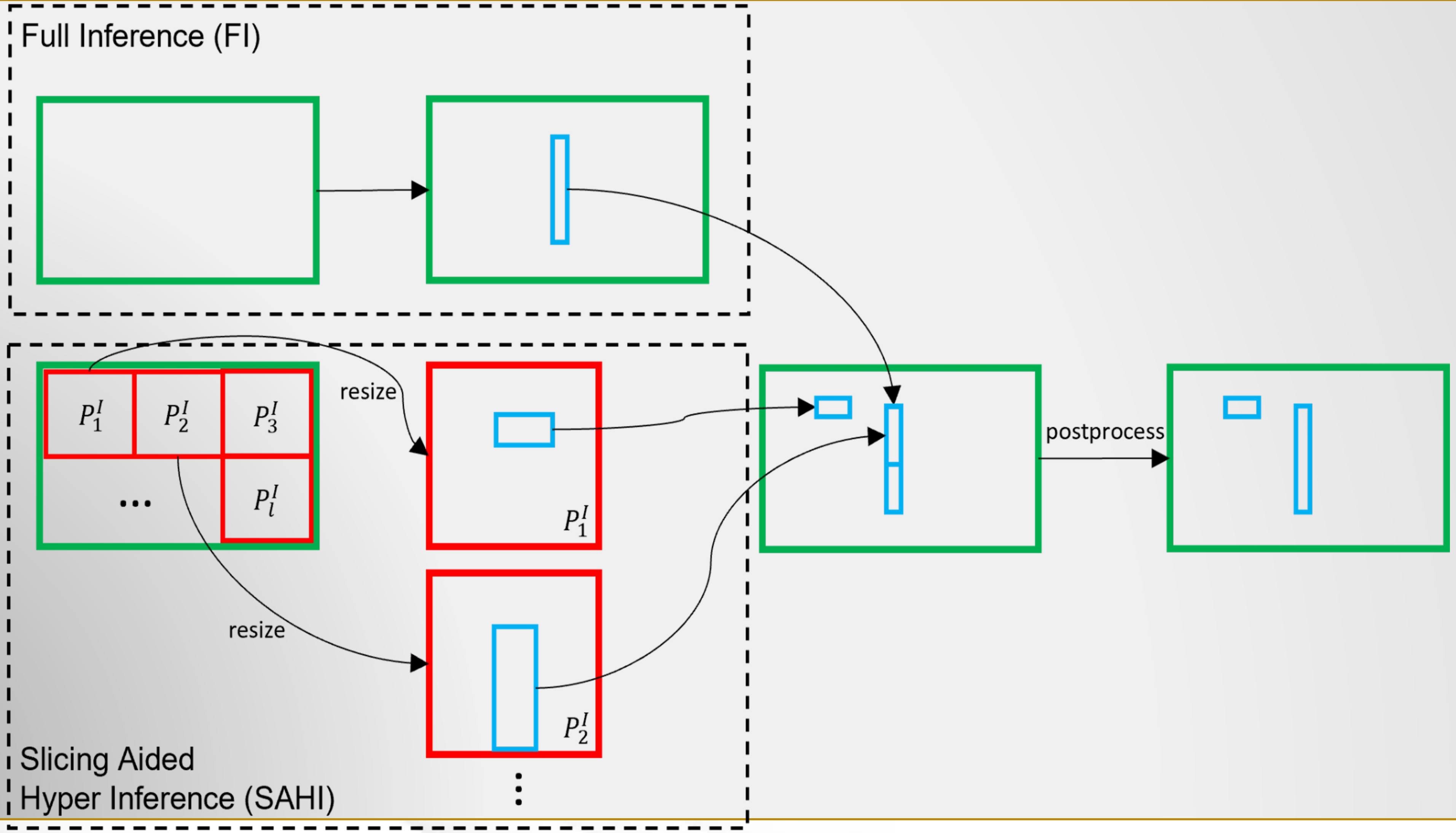
refactor demo notebooks by utilizing newly i
documentation enhancement
#516 opened on 5 Jul by ishworii • Review required

add Tensorflow Hub detector support ✓ en
#501 opened on 19 Jun by kadirnar • Changes requested

Slicing Aided Fine-tuning



Slicing Aided Hyper Inference (SAHI)



Vision Transformers + CNNs

- ViT performance depends highly on dataset-specific hyperparameters, and network depth. CNNs are much easier to optimize.
- A variation on a pure transformer is to marry a transformer to a CNN stem/front end. A typical ViT stem uses a 16x16 convolution with a 16 stride. By contrast a 3x3 convolution with stride 2, increases stability and also improves accuracy*.
- The CNN translates from the basic pixel level to a feature map. A tokenizer translates the feature map into a series of tokens that are then fed into the transformer, which applies the attention mechanism to produce a series of output tokens.
- Finally, a projector reconnects the output tokens to the feature map. The latter allows the analysis to exploit potentially significant pixel-level details. This drastically reduces the number of tokens that need to be analyzed, reducing costs accordingly.

Xiao, Tete et al., 2021, "Early Convolutions Help Transformers See Better". arXiv:2106.14881