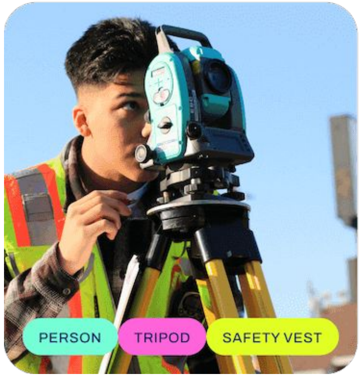
AICP: Image module Day 4: Deep Learning Applications

Guillaume Witz
Data Science Lab, University of Bern

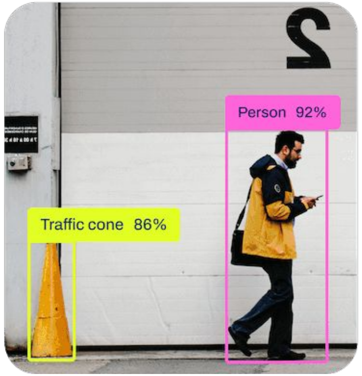
DSL

Common operations

Classify

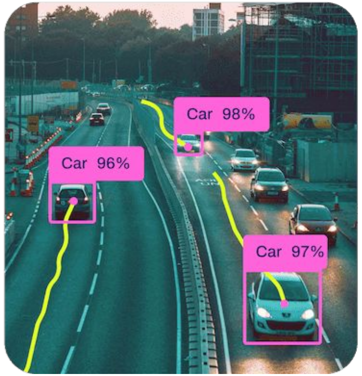


Detect



Segment

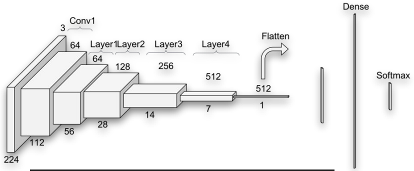
Track



Pose

https://github.com/ultralytics/ultralytics

How to run models for these applications?



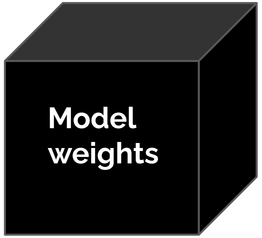

```
class Predictor(BasePredictor):
    def setup(self):
        """Load the model into memory to make running multiple predictions eff
        print("Loading pipeline...")

        # load the BLIP model
        self.model_blip, self.vis_processors, _ = load_model_and_preprocess(
            name="blip_caption",
            model_type="base_coco",
            is_eval=True,
            device=torch.device("cuda"),
        )

    def predict(
        self,
        image: Path = Input(
            description="Input image",
        ),
        task: str = Input(
            description="Describe how to edit the image", default="cat2dog"
        ),
        xa_guidance: float = Input(
            description="",
            default=0.1,
        ),
        negative_guidance_scale: float = Input(
            description="Number of images to output.",
            default=5.0,
        ),
        num_inference_steps: int = Input(
            description="Number of denoising steps", ge=1, le=500, default=50
```

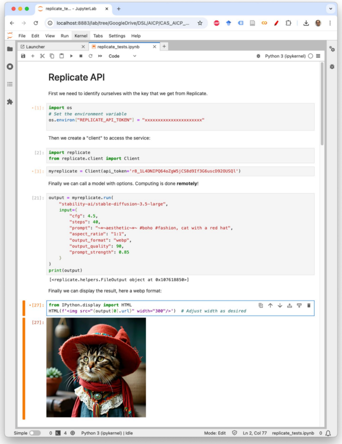
The model architecture and code to run it

The data to feed to the model (training or inference)



The model weights or parameters

The hardware to run the model



An interface

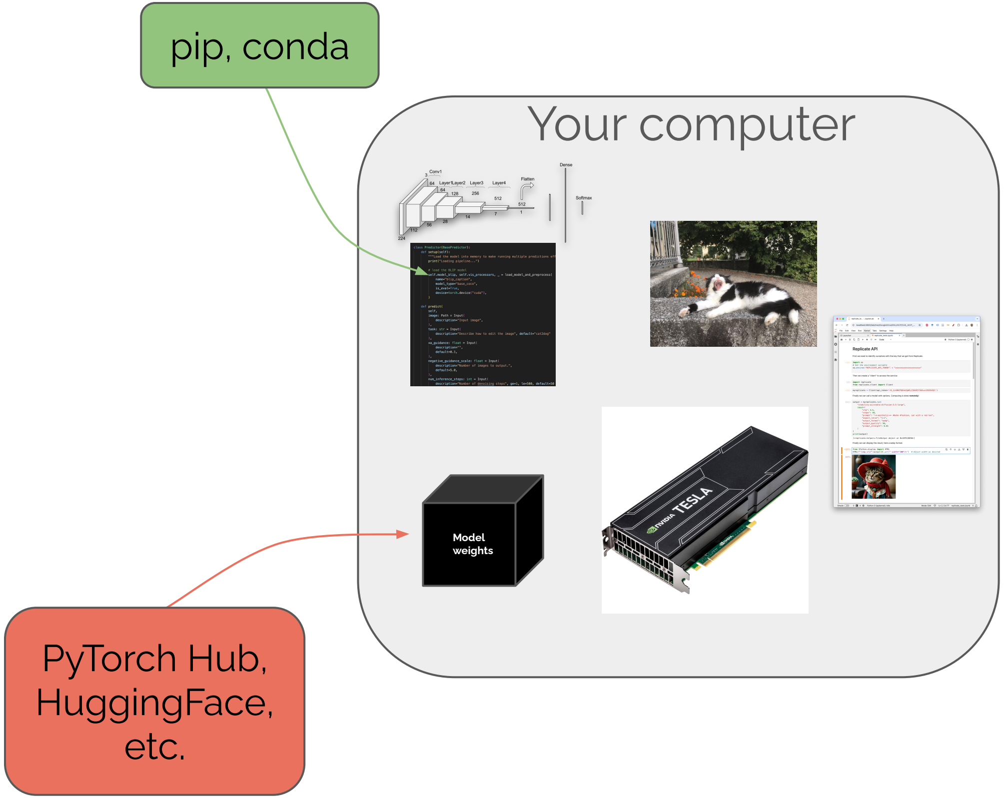
Local solution

On your computer you can:

- install the software (pip, conda)
- have your data ready
- download weights from online
- run your model
- choose your interface

Limitations:

- You will be limited by your hardware e.g. no diffusion models



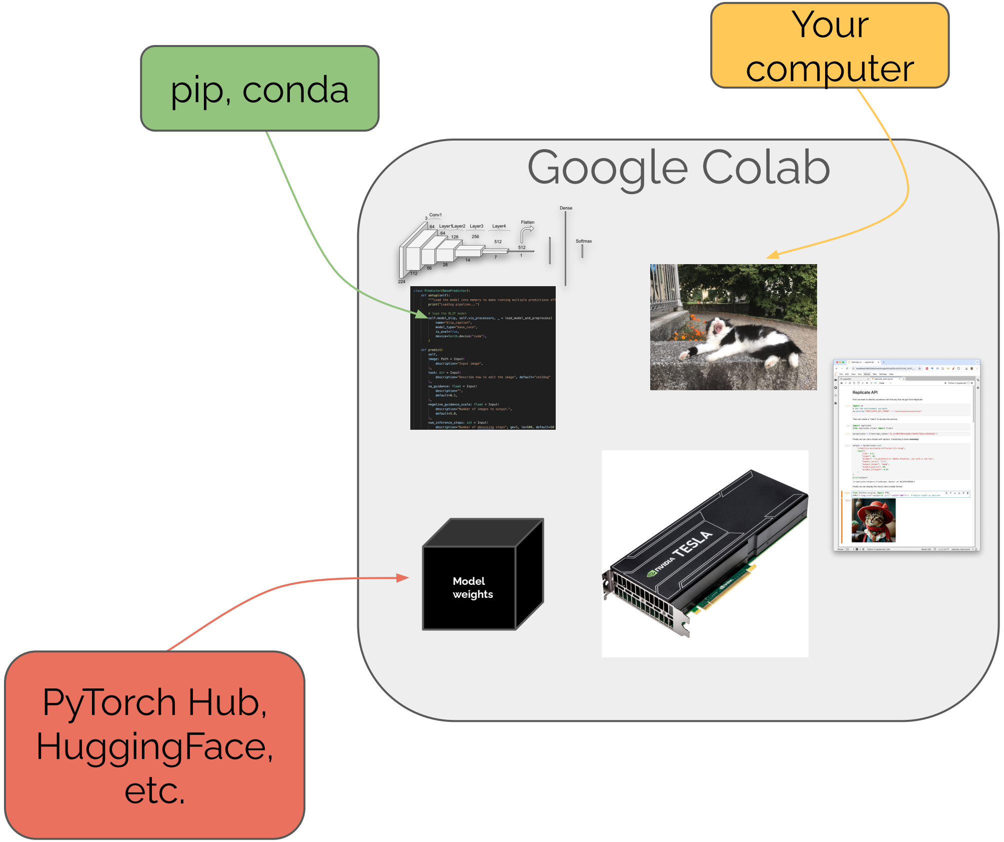
Online notebook e.g. Colab

Colab is an online notebook. All resources run remotely. You need to:

- install the software (pip, conda)
- upload your data
- download weights from online
- run your model

Limitations:

- Limited time and resources
- Continuous re-installation



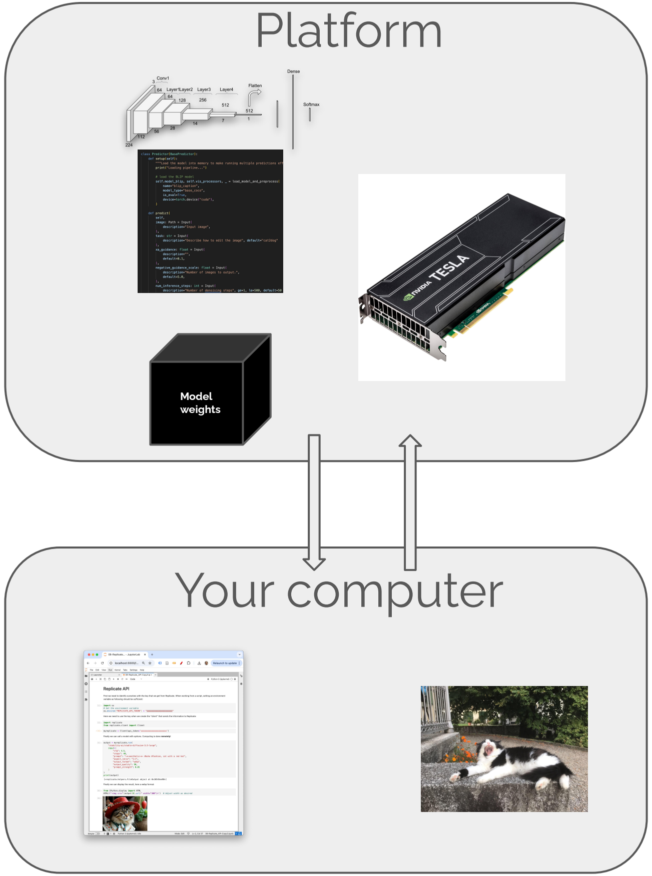
Local interface, remote model

You have insufficient local resources for a model but want to "automate" your workflow:

- Use a pre-installed model on a platform
- Send commands and data to the platform
- Use the platforms hardware
- Get the result locally

Limitations:

- models on platform
- price



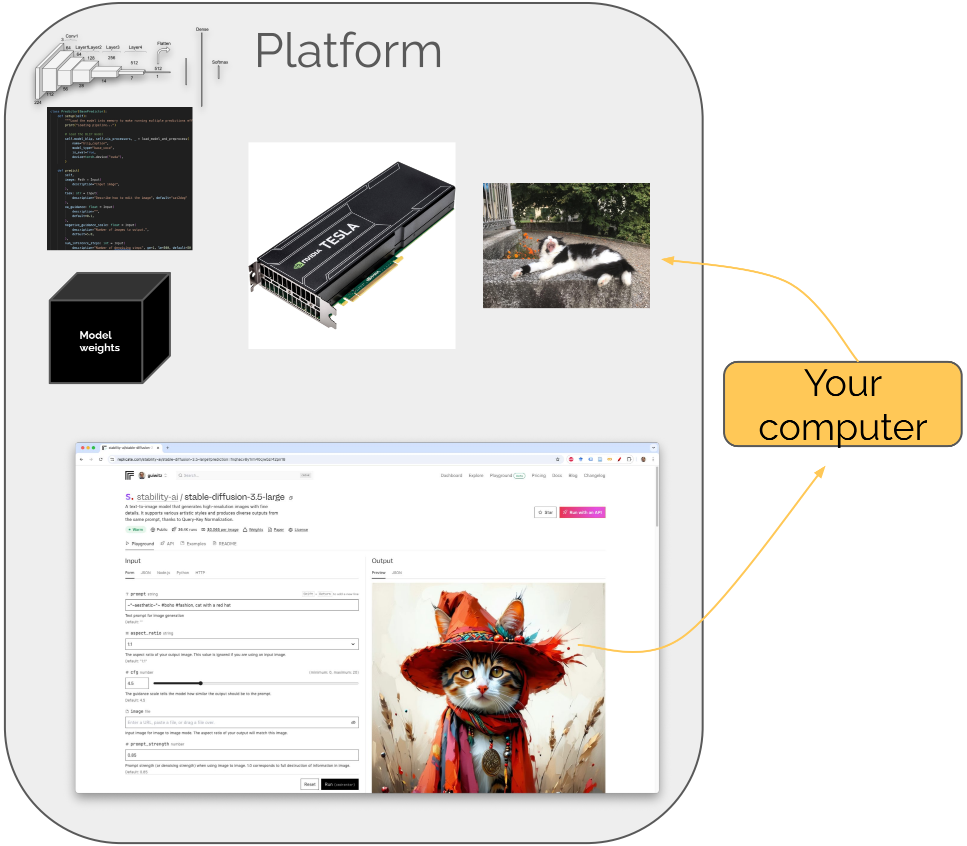
Remote interface and model

Fully remote solution:

- Use a pre-installed model on a platform
- Use interface: no code
- Use the platforms hardware

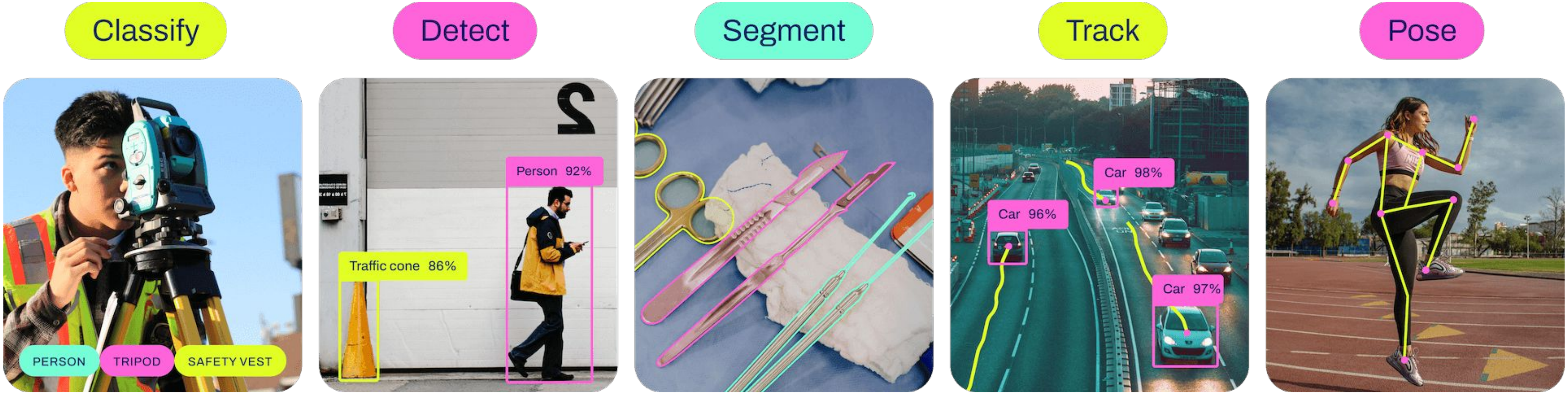
Limitations:

- models on platform
- price
- not integrated in workflow



Example 1: YOLO

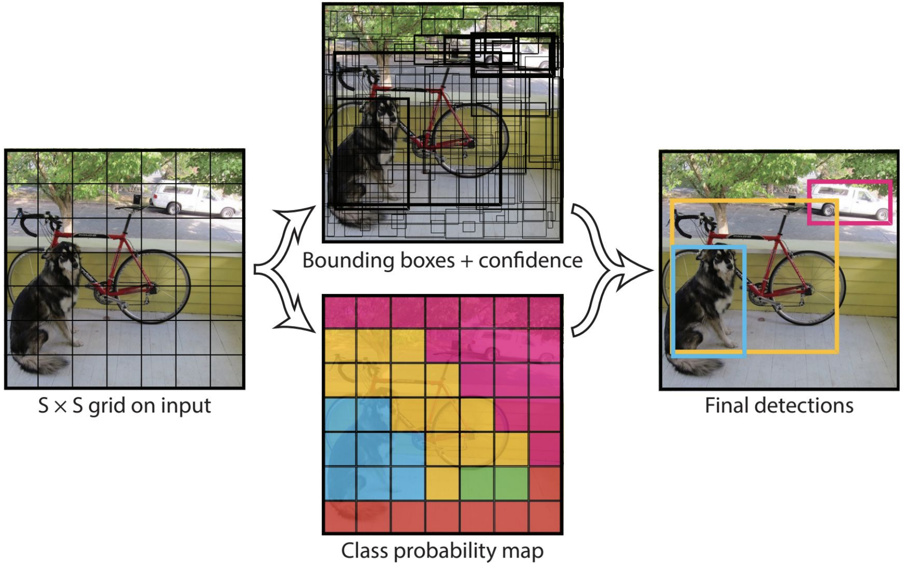
YOLO example



https://github.com/ultralytics/ultralytics

Object detection: YOLO

- The image is split into a grid
- Bounding boxes are predicted relative to each element of the grid
- The number of boxes and possible predictions per grid can be adjusted in the model
- The output of the CNN predicts box coordinates as well as the most probable class fort the grid elements



YOLO: generalization

Figure 5: Generalization results on Picasso and People-Art datasets.

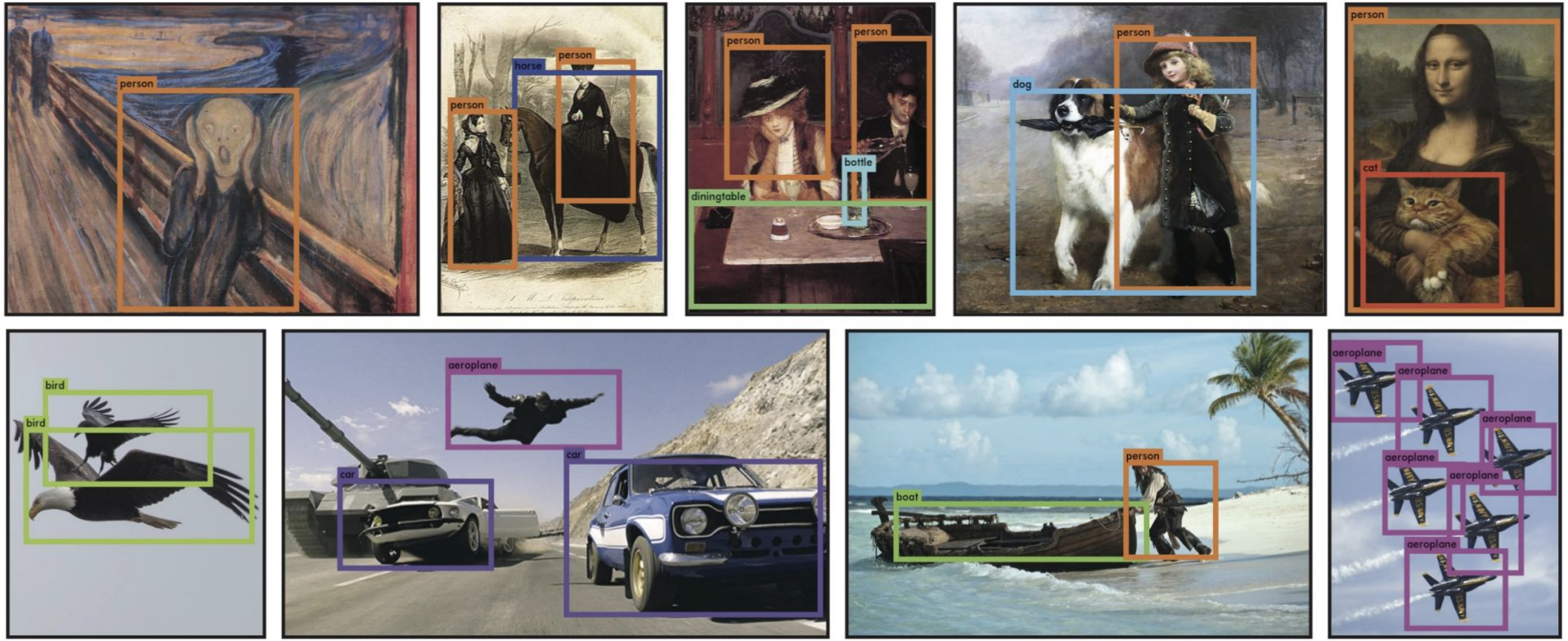
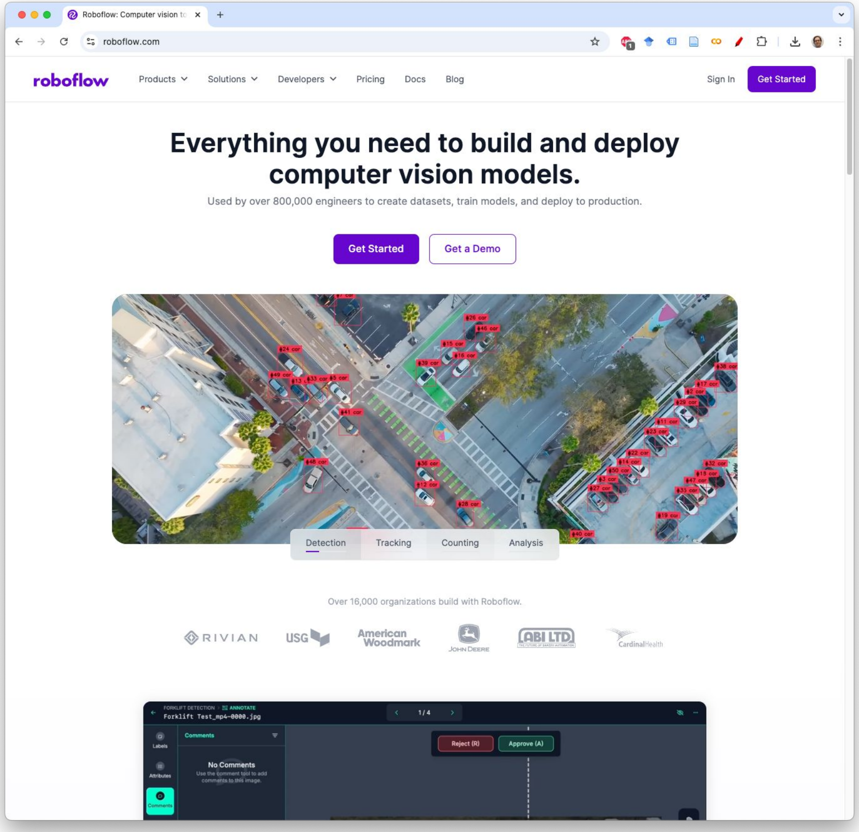


Figure 6: Qualitative Results. YOLO running on sample artwork and natural images from the internet. It is mostly accurate although it does think one person is an airplane.

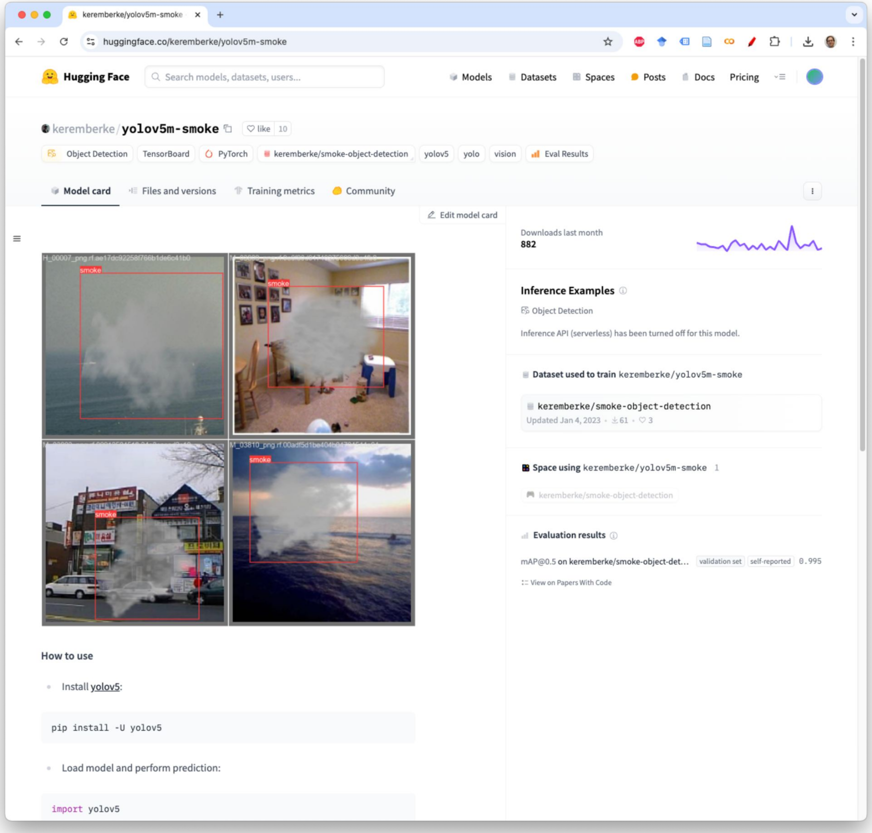
YOLO

- Available on platforms like Roboflow but also as package



YOLO

- Available on platforms like Roboflow but also as package
- Possibility to retrain model for special applications
- Good example of a model that is easy to run locally!



YOLO

YOLO is an efficient model that can even be run without a GPU

Create a new environment dedicated to it:

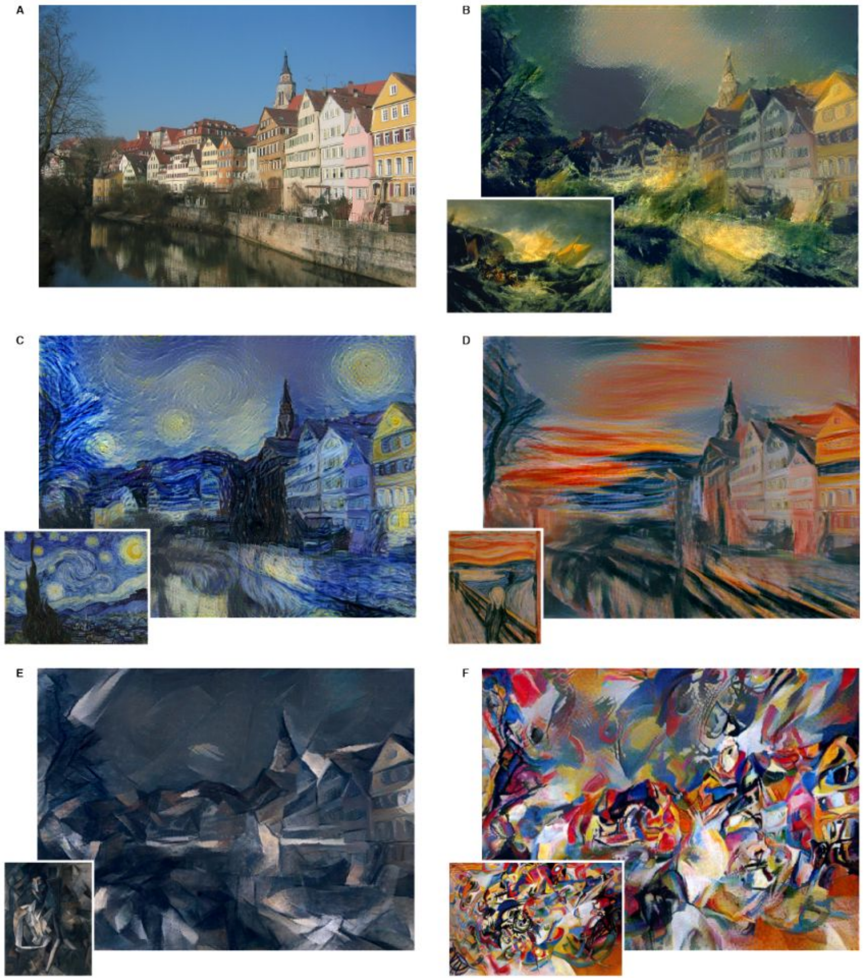
```
conda create -n yolo python=3.12 jupyterlab
conda activate yolo
pip install ultralytics replicate
```

Download the notebook:

https://github.com/guiwitz/CAS_AICP_M4/blob/main/05-YOLO.ipynb

Example 2: Style Transfer

Style Transfer



Gatys, L.A., Ecker, A.S., Bethge, M., 2015.A neural algorithm of artistic style.arXiv preprint arXiv:1508.06576

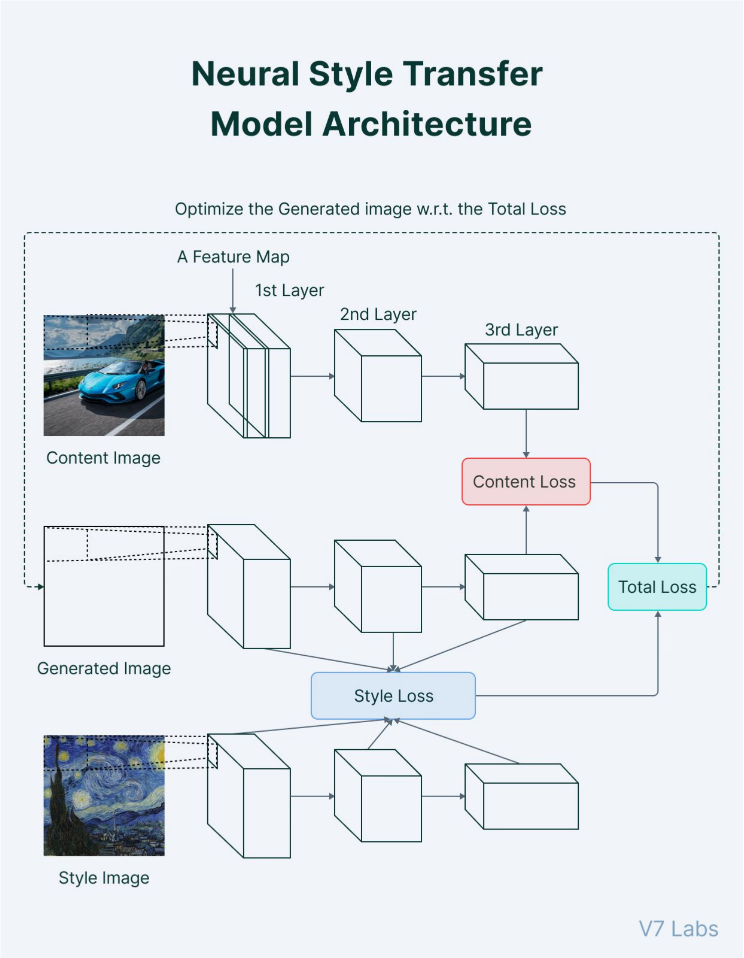
Style Transfer

In neural style transfer we exploit the fact that convolutional networks extract information at different scales.

Content without detail can be reconstructed via deep layers.

Style can be reconstructed via both large and small scale features.

We can train a network that aims at recovering both Content and Style



https://www.v7labs.com/blog/neural-style-transfer

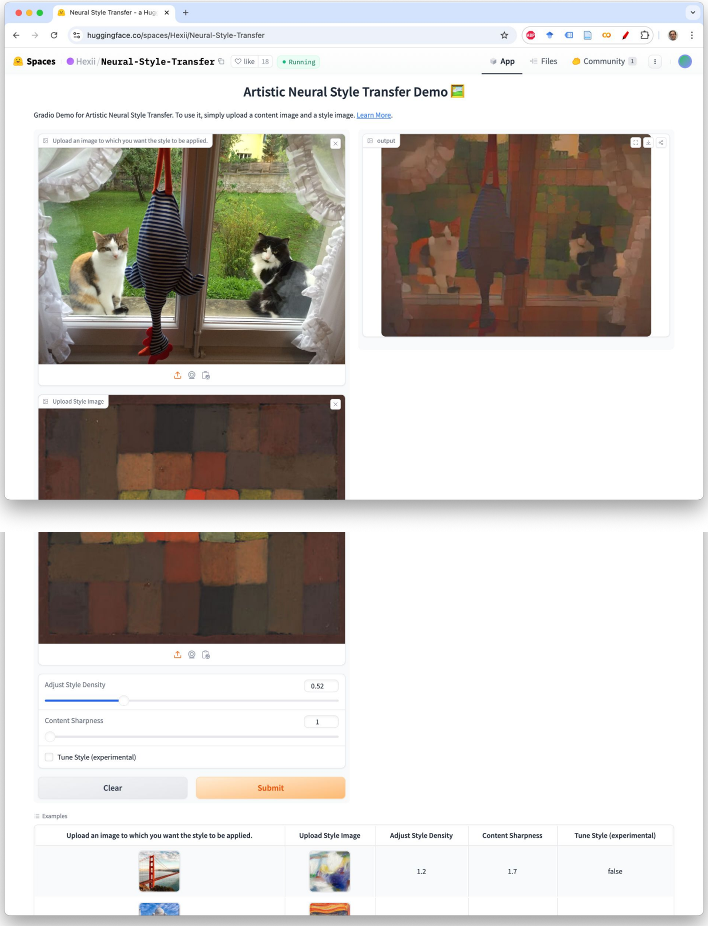
Style Transfer on HuggingFace

- Repository for models, datasets and demos
- Some free demos, some to run locally
- Models can be downloaded and used elsewhere

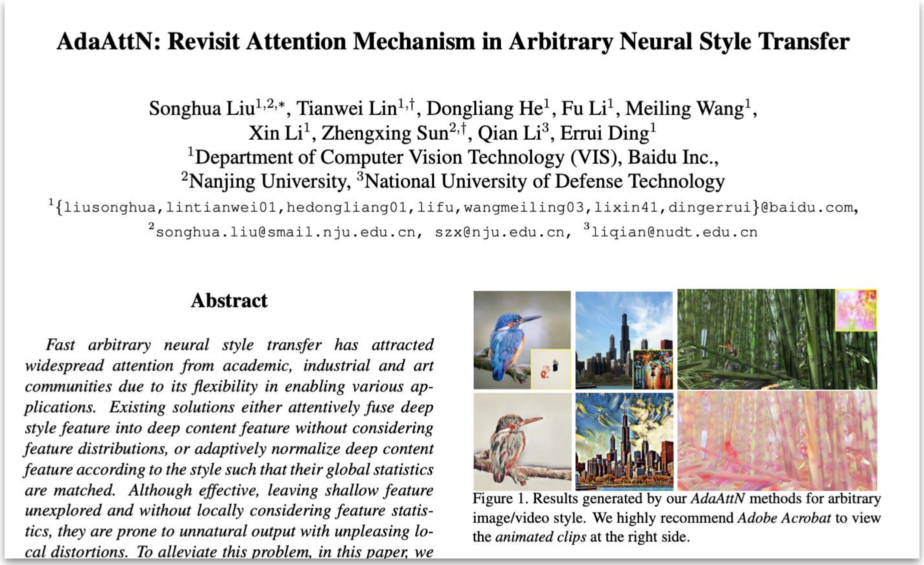
https://huggingface.co/spaces/Hexii/Neural-Style-Transfer

And let's look at the notebook and how it was made:

https://colab.research.google.com/github/guiwitz/CAS_AICP_M4/blob/main/06-Style_transfer.ipynb



Style Transfer on Replicate.com



AdaAttN: Revisit Attention Mechanism in Arbitrary Neural Style Transfer

Songhua Liu[1,2,*], Tianwei Lin[1,†], Dongliang He[1], Fu Li[1], Meiling Wang[1],
Xin Li[1], Zhengxing Sun[2,†], Qian Li[3], Errui Ding[1]
[1]Department of Computer Vision Technology (VIS), Baidu Inc.,
[2]Nanjing University, [3]National University of Defense Technology
[1]{liusonghua,lintianwei01,hedongliang01,lifu,wangmeiling03,lixin41,dingerrui}@baidu.com,
[2]songhua.liu@smail.nju.edu.cn, szx@nju.edu.cn, [3]liqian@nudt.edu.cn

Abstract

Fast arbitrary neural style transfer has attracted widespread attention from academic, industrial and art communities due to its flexibility in enabling various applications. Existing solutions either attentively fuse deep style feature into deep content feature without considering feature distributions, or adaptively normalize deep content feature according to the style such that their global statistics are matched. Although effective, leaving shallow feature unexplored and without locally considering feature statistics, they are prone to unnatural output with unpleasing local distortions. To alleviate this problem, in this paper, we

Figure 1. Results generated by our *AdaAttN* methods for arbitrary image/video style. We highly recommend *Adobe Acrobat* to view the *animated clips* at the right side.

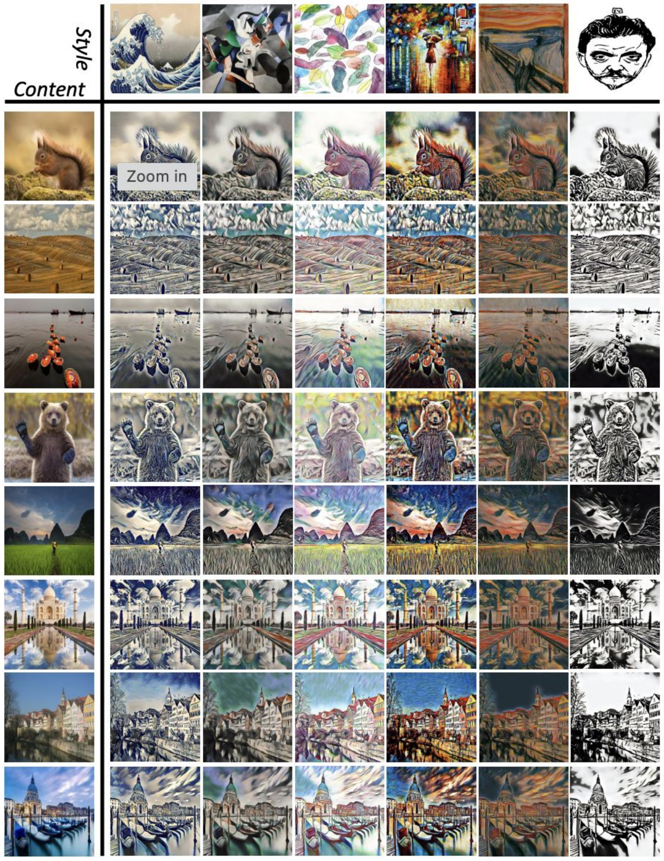
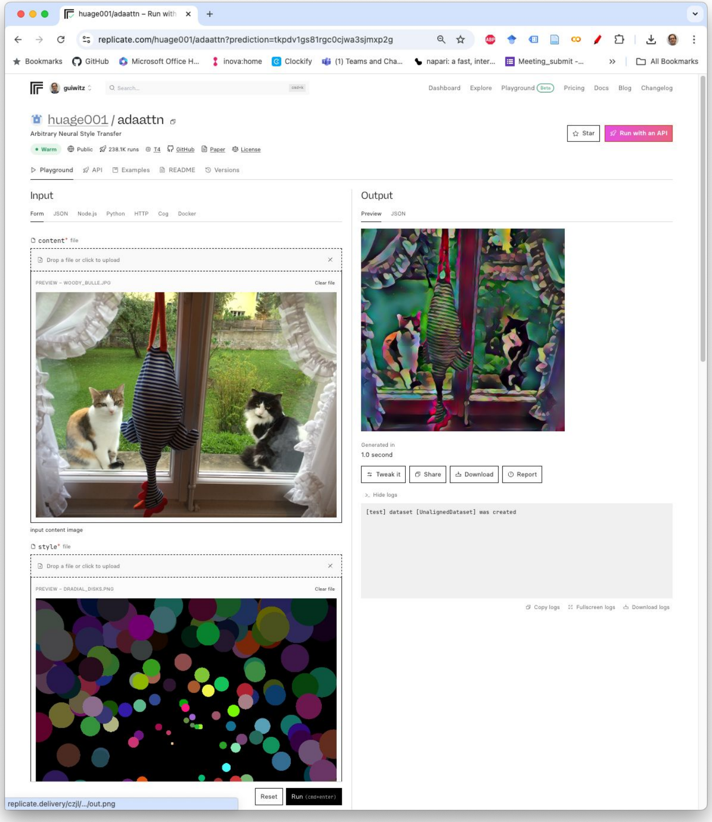
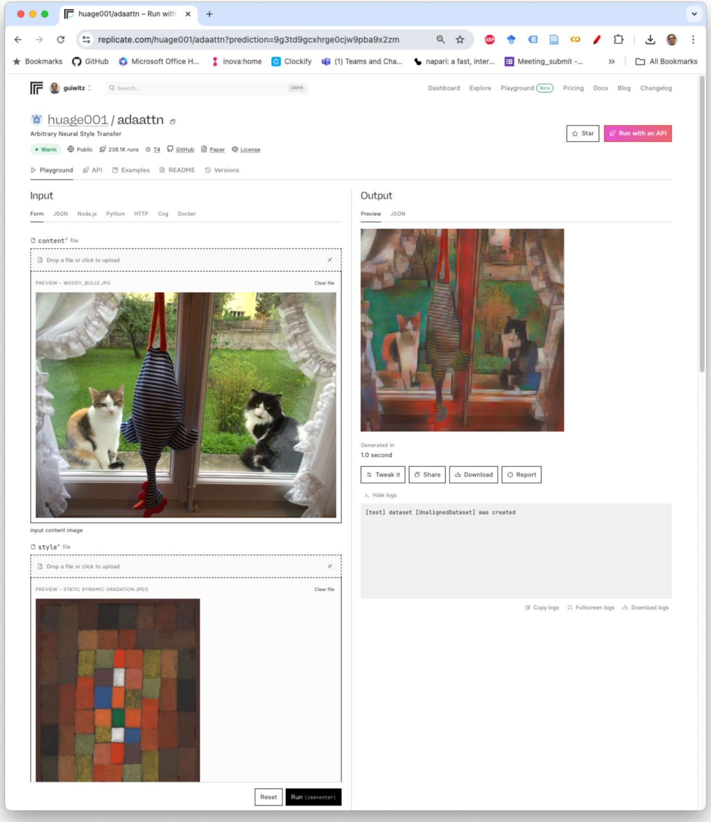


Figure 12. More image style transfer results.

Style Transfer on Replicate.com





https://replicate.com/huage001/adaattn

OnDemand UBELIX

https://ondemand.hpc.unibe.ch/