Active Object Localization

Reinforcement Learning Final Project 2021

Meet Us

- Mohammed Deifallah: Mohammed.Deifallah@skoltech.ru

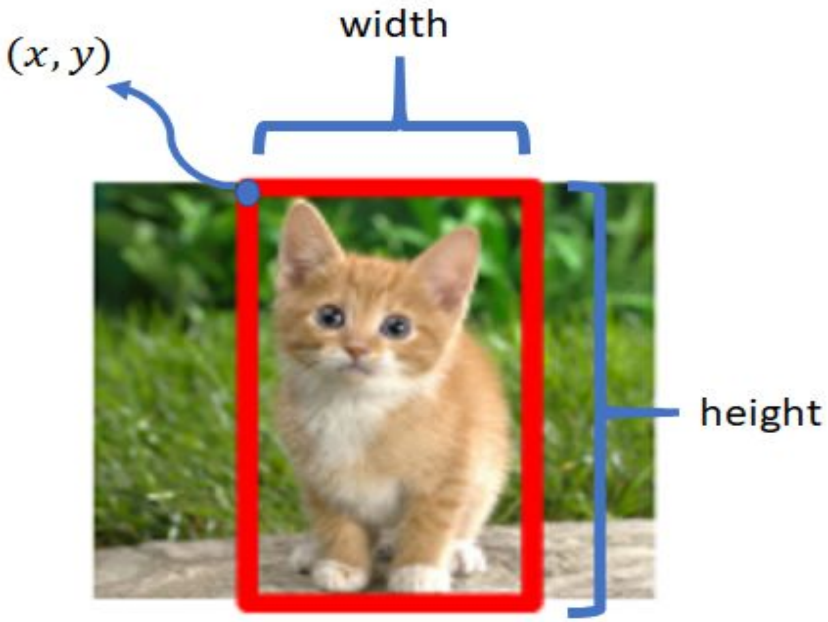
- Alexander Vikhorev: Alexander.Vikhorev@skoltech.ru

Agenda

- Motivation
- Related Work
- Dataset
- Model Architecture
- Training
- Evaluation
- Results
- Conclusion

Motivation

- Collection of CV tasks.
- RCNN Family
- YOLO Family
- New Level of RL



Source: https://tinyurl.com/czuvhrrc

Agenda

- Motivation
- Related Work
- Dataset
- Model Architecture
- Training
- Evaluation
- Results
- Conclusion

Related Work

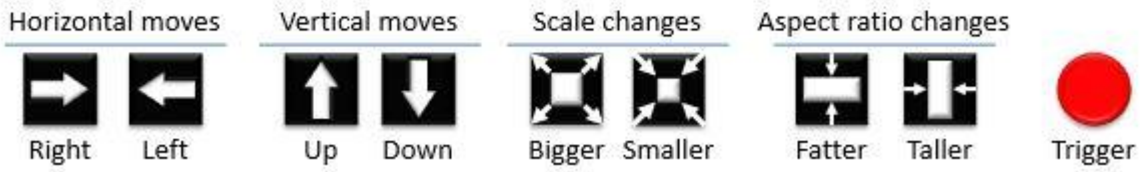
- Based on Active Object Localization with Deep Reinforcement Learning
- Per-class RL agents
- **MDP** (Markov Decision Process): **A** (actions), **S** (state), **R** (reward function)
- ε-greedy policy

Related Work (Method)

State

o (4096) | h (9 x 10)

Actions



Reward

$$R_a(s, s') = sign\,(IoU(b', g) - IoU(b, g))$$

Agenda

- Motivation
- Related Work
- Dataset
- Model Architecture
- Training
- Evaluation
- Results
- Conclusion

Dataset

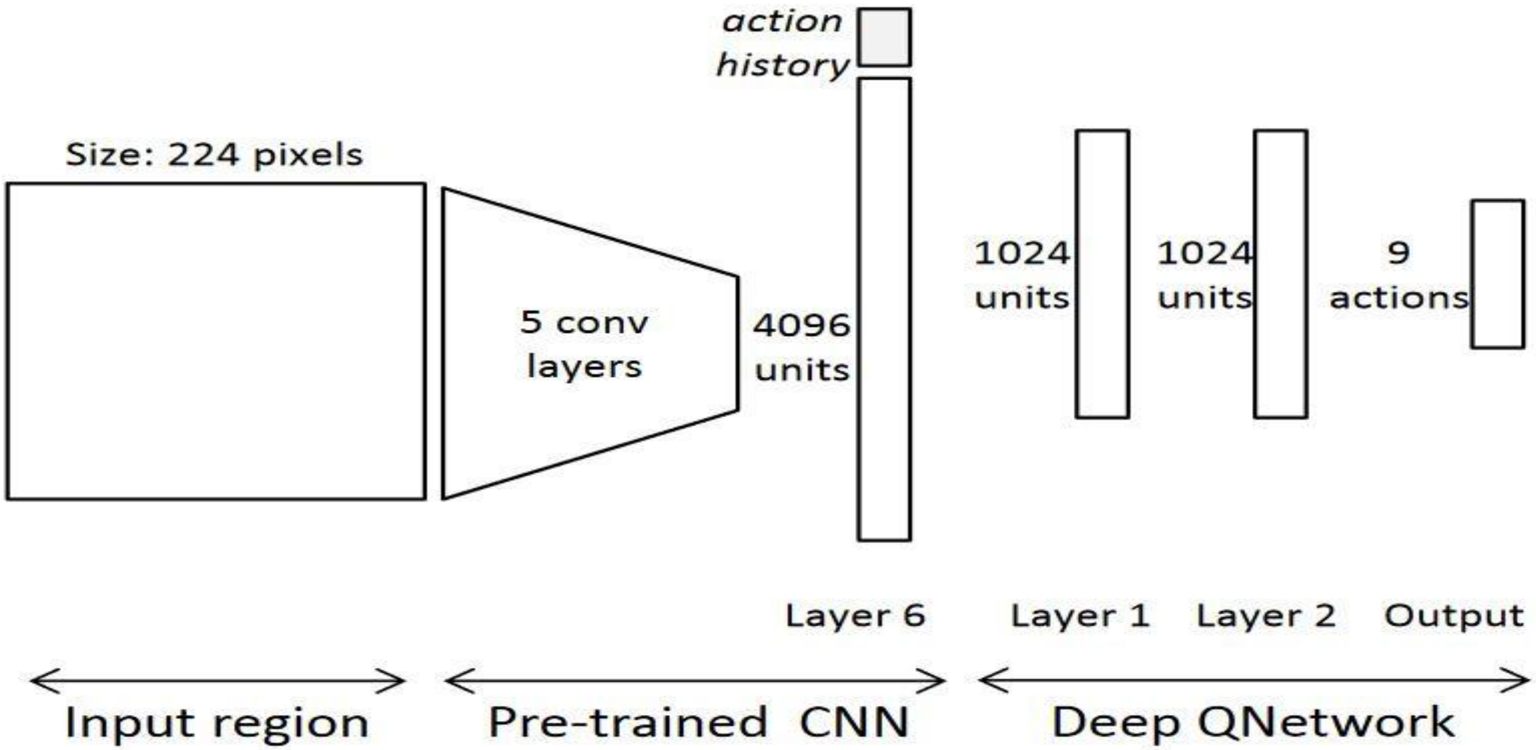
- PASCAL VOC 2007-2012 (only 2007 😉)
- 20 classes (only 5 😉)
- PyTorch (only offline 😉)

Source: http://host.robots.ox.ac.uk/pascal/VOC/

Agenda

- Motivation
- Related Work
- Dataset
- Model Architecture
- Training
- Evaluation
- Results
- Conclusion

Model Architecture



Agenda

- Motivation
- Related Work
- Dataset
- Model Architecture
- Training
- Evaluation
- Results
- Conclusion

Training

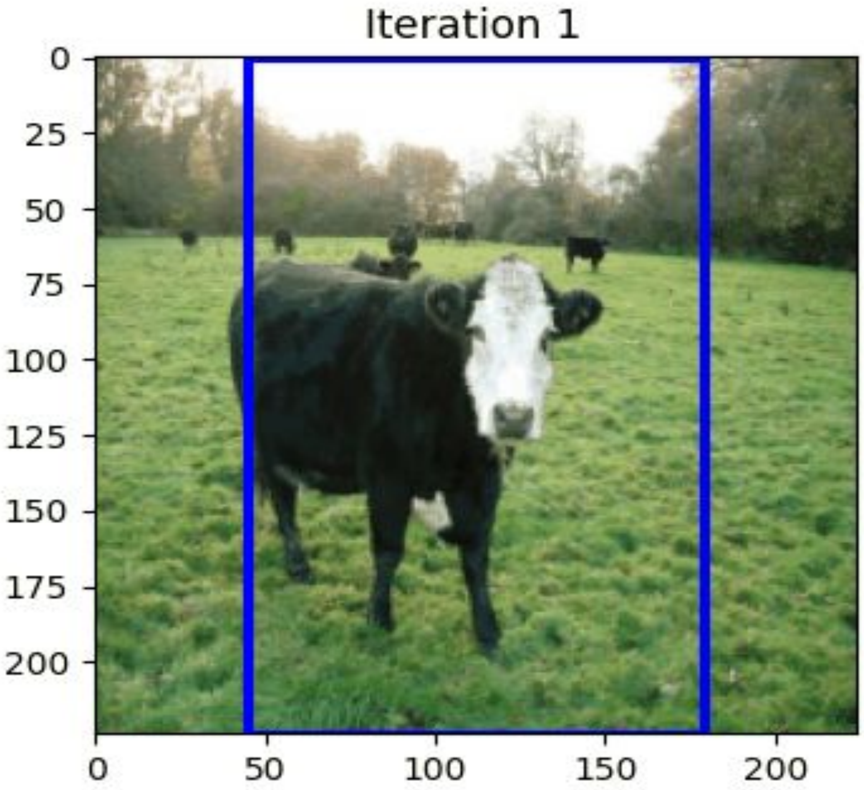
- Pascal VOC 2007 training set
- Input: 224 x 224 image
- $\alpha = 0.2$
- $\varepsilon = 1.0 \longrightarrow 0.1$
- $\eta = \text{sign}(3.0)$
- Replay Memory
- 15 episodes

Agenda

- Motivation
- Related Work
- Dataset
- Model Architecture
- Training
- Evaluation
- Results
- Conclusion

Evaluation

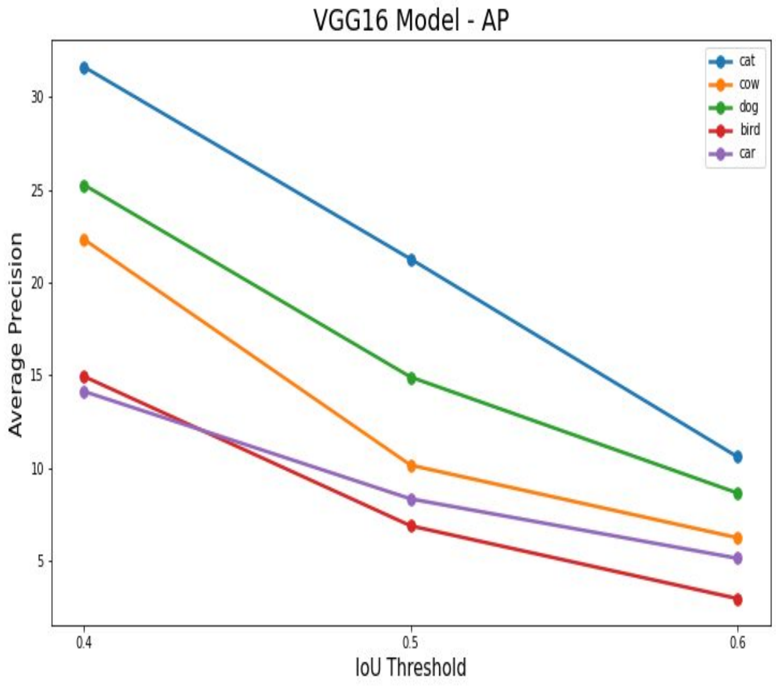
- PASCAL VOC 2007 validation set
- AP (Average Precision) & Recall
- Max. 40 iterations

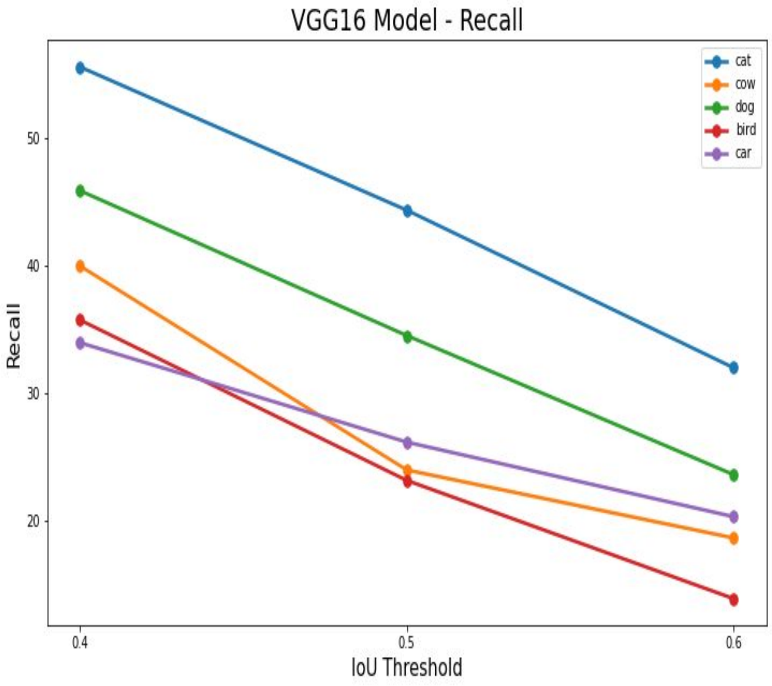


Agenda

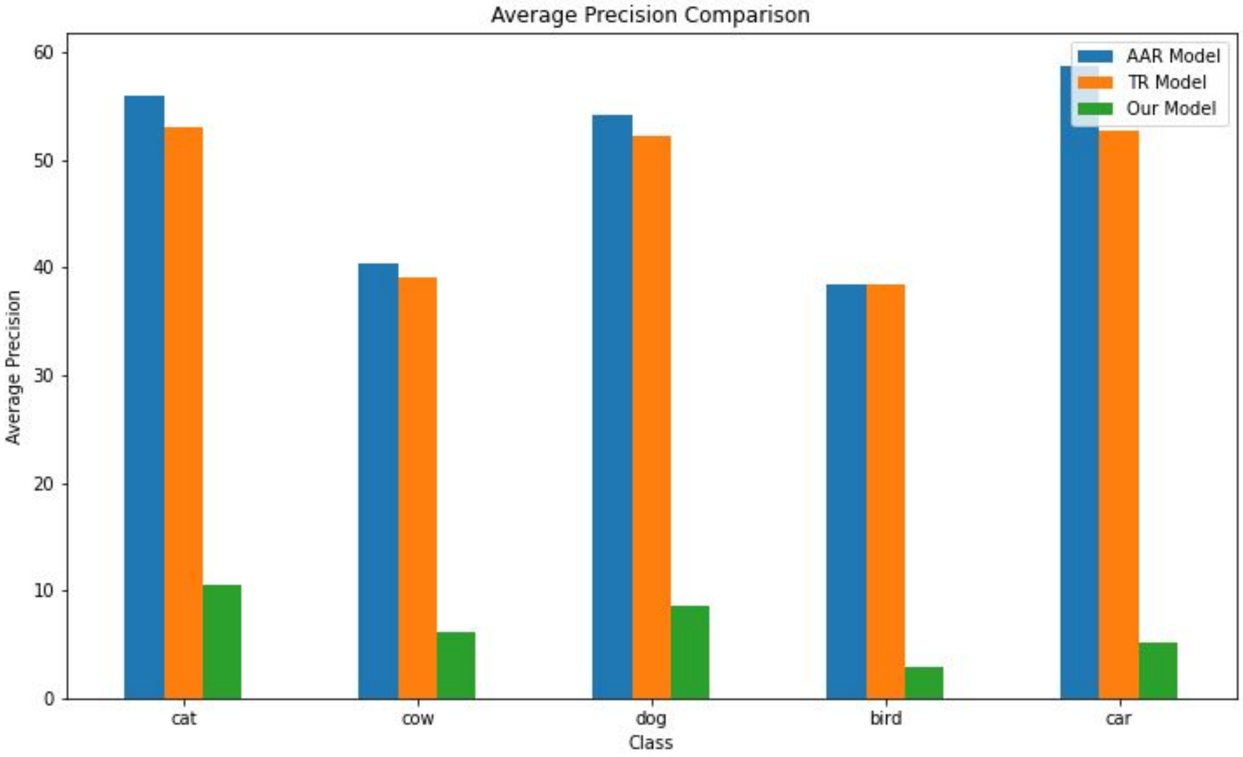
- Motivation
- Related Work
- Dataset
- Model Architecture
- Training
- Evaluation
- Results
- Conclusion

Results


VGG16 Model - AP
Average Precision
IoU Threshold
cat
cow
dog
bird
car
5
10
15
20
25
30
0.4
0.5
0.6


VGG16 Model - Recall
Recall
IoU Threshold
cat
cow
dog
bird
car
20
30
40
50
0.4
0.5
0.6

Results


Average Precision Comparison
AAR Model
TR Model
Our Model
Average Precision
0
10
20
30
40
50
60
cat
cow
dog
bird
car
Class

Results (GitHub Repo)

Code Overview

- `Training.ipynb` is used to reproduce the training process of the model.
- `Testing.ipynb` is used to reproduce the testing process of the model and visualize some examples of localization.
- `Plotting.ipynb` is used to plot all graphs and charts shown above using `media` folder.
- `media` is a folder to save all examples of localization and graphs.
- `models` is a directory which is needed to keep the saved models weights after training. This is an example of the so-called folder.
- `utils` is a directory contaning the following files:
    - `agent.py`: a wrapper for the per-class agent that contains the whole components of RL (ϵ-greedy policy, reward, ... etc).
    - `models.py`: a wrapper for the two main modules of the network: *Feature Extractor* and *DQN*.
    - `dataset.py`: a separate file for reading the dataset (train and val).
    - `tools.py`: a collection of useful functions, such as computing the metrics or extracting a specified class from the dataset.

Source: https://github.com/Mohammed-Deifallah/Active-Object-Localization

Agenda

- Motivation
- Related Work
- Dataset
- Model Architecture
- Training
- Evaluation
- Results
- Conclusion

Conclusion

- Linear SVM should be added to the model.
- IoU threshold > 0.6 has negative effect.
- Different pre-trained networks (e.g. ResNet50)

Bonus slides : replacing DQN with PPO

- All environment related logic has been factored out and wrapped by gym.Env interface
- Allows to plug-in any available out-of-box algorithm implementation which works with Open AI Gym (Stable Baselines, Tianshou, Elegant RL, Lightning Bolts, ...)
- PPO implementation from Stable Baselines 3 has been chosen for its ability to handle both discrete and continuous inputs as well as learning performance
    - Actor-Critic scheme - learns both the policy and value function
    - On-policy algorithm - updates the model episode by episode based on the current exploration of the agent

Bonus slides : PPO (objective function)

$$L_t^{CLIP+VF+S}(\theta) = \hat{\mathbb{E}}_t\left[L_t^{CLIP}(\theta) - c_1 L_t^{VF}(\theta) + c_2 S[\pi_\theta](s_t)\right]$$

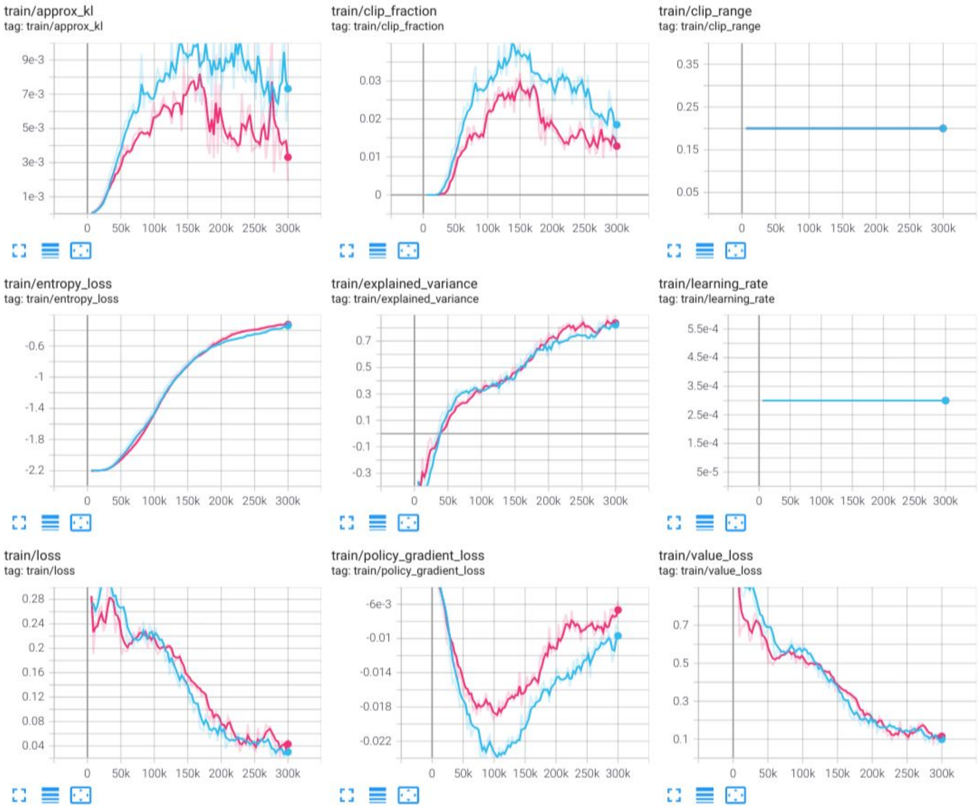
Add an entropy bonus **to ensure sufficient exploitation.**

c1 and c2 are coefficients.

Squared-error value loss: ($V_\theta(s_t) - V_t^{targ})^2$

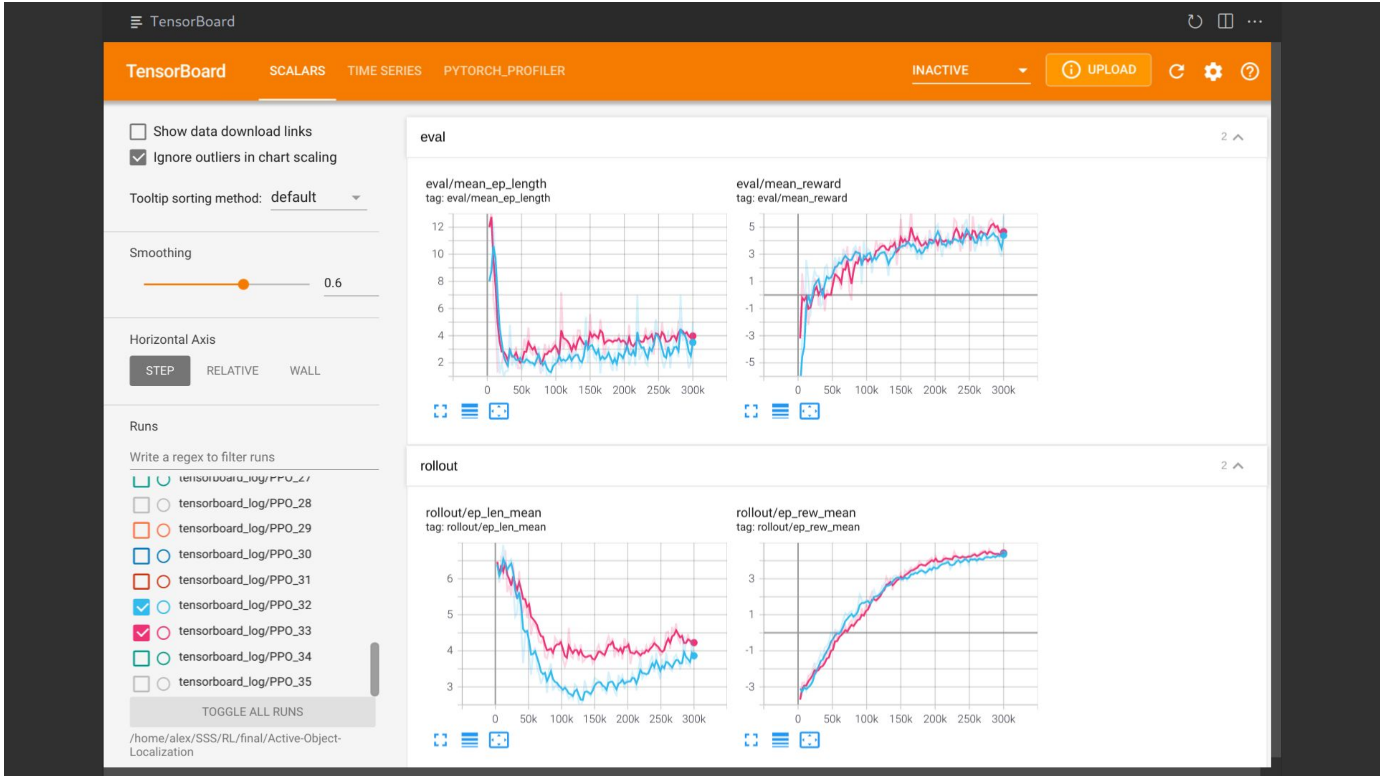
- 3 parts : PG loss , VF loss, Entropy loss
- Figure source : https://spraphul.github.io/blog/RL5

Bonus slides : PPO (training metrics)



KL-divergence - how different is the updated policy from the previous policy (we don't want this metric to be high since we want gradual change of the policy)
Clip fraction - Fraction of times the clip range hyperparam is used. PPO clips the new policy within the clip range of the old policy(stable learning).
Clip range - defined by hyper-param (usually 0.1 - 0.2)
Entropy loss increases (in the objective function we reward high entropy and entropy decreases over time (in the beginning favoring exploration over exploitation and vise versa in the end)
General train loss, policy gradient loss and value function loss decreases over time

Bonus slides : PPO (mean episode reward)



Bonus slides : PPO (results)

Thanks!

Questions?