Applied Machine Learning: Examples in the ICT domain
Lecture 2

Prof. Andrea Zanella

zanella@dei.unipd.it

- office: +39 049 8277770
  fax : +39 049 8277699
- email: zanella@dei.unipd.it
- web : http://www.dei.unipd.it/~zanella

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Cognition-based networks: applying cognitive science to wireless networking

Outline

- Cognition-based networks
  - Evolution of cognitive communications and networks
  - New holistic concept of cognition-based networks
  - Use of machine learning tools toward this vision
- Examples of application
  - QoE-driven video streaming control

Motivation

- Communication systems are becoming very complex, and call for intelligent solutions
- New technologies make such evolutions real
- For example:
  - White spaces in licensed spectrum (e.g., TV bands), spectrum re-used by a secondary user
  - Self-organizing networks in ad hoc (e.g., disaster) scenarios
  - Advanced paradigms in HetNets
- All of these are specific cases of a more general approach based on learning and context-awareness

Cognition applied to wireless

- Applying cognition is a way to deal with the complexity and challenges of future systems
- Cognition is already in use today in several cases
  - Cognitive radios, biologically inspired networks, node adaptation by learning, etc
- However, in order to draw the most benefit from this approach, one needs to
  - Consider all players in a more coherent manner
  - Apply cognition at all network layers and end-to-end
  - Apply the most advanced paradigms taken from cognitive science

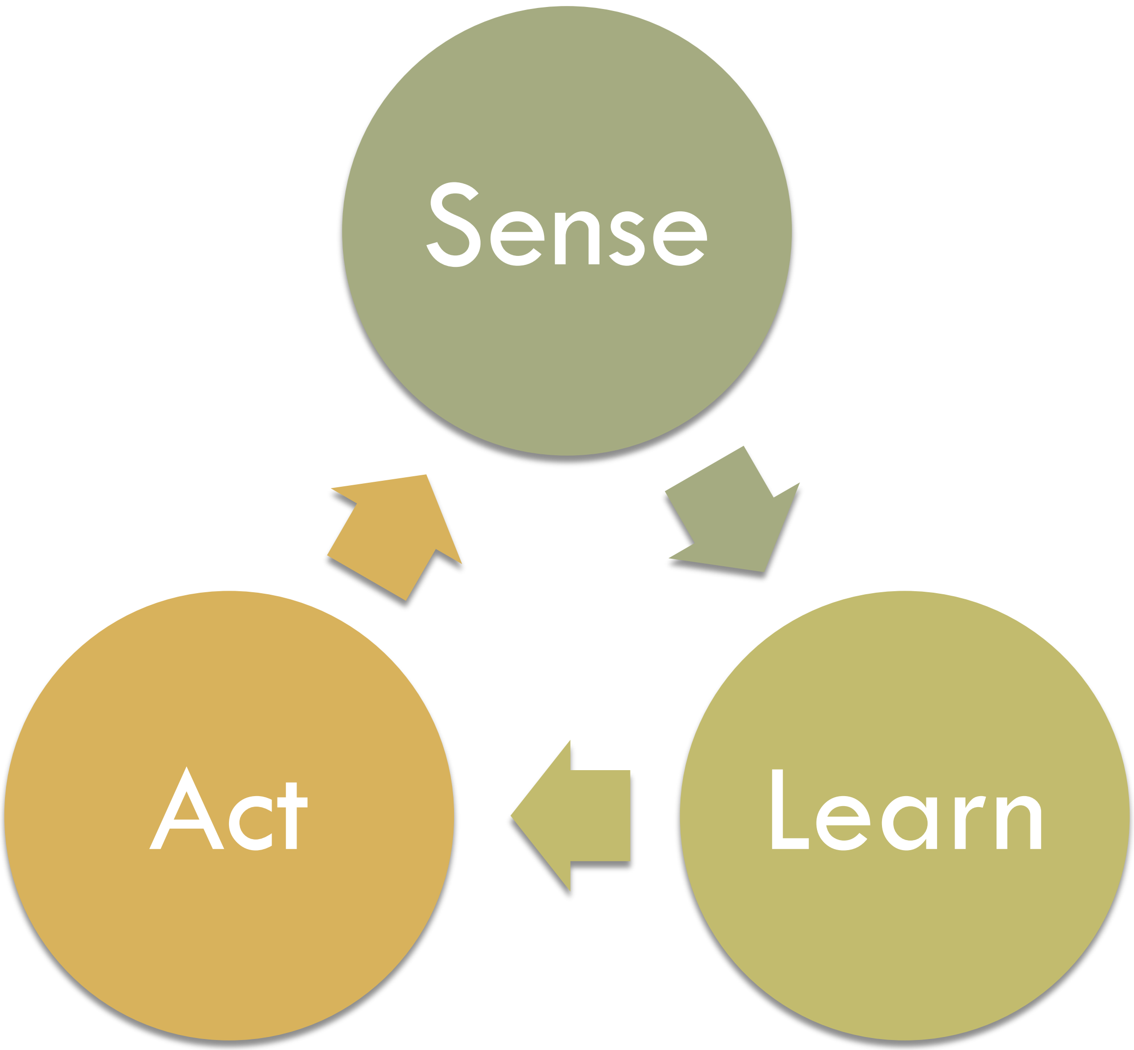
Cognition applied to wireless

- Mitola (2000) and Haykin (2005) actually gave a very general definition of the cognitive paradigm
- They spoke about the essence of cognition, including
  - Intelligent observation, learning, decision-making, emergent and collaborative behaviors
- It is clear that what has happened in this field since then has only scratched the surface
  - Cognitive networking in a broad sense remains an exciting and largely unexplored research field

Evolution of the cognitive paradigm

- Cognitive radios: frequency-agile devices for opportunistic access
- Cognitive radio networks: networks of CRs
- Cognitive networks: the cognitive approach applied to networking layers and end-to-end
  - Learning, decision-making, information sharing
- **Cognition-based networks:** drawing from the most recent results in cognitive science
  - Advanced unsupervised learning, emergent behaviors, evolutionary computation, model building, collective intelligence, combined with reconfigurable, software-defined communication techniques
  - Including out-stack information (related to the environment & the user)

The cognition cycle (simplified)



True for humans, true for networks

- **Sense**: nowadays devices are crammed with transducers / sensing apparatuses
  - needs efficient data handling
- **Learn**: optimization algorithms can be run at each node individually
  - needs (i) efficient algos (ii) harmonization
- **Act**: network modifies the environment
  - requires convergence of multiple devices

Supervised vs unsupervised learning

- Supervised learning requires a training set and/or explicit feedback about outcomes
  - Suitable when this is available, and when the goal is well-defined and known a priori
  - However, it has been shown to fail in some cases
- Unsupervised has no prior knowledge
  - No pre-existing model, more general
  - Reveals features that are not predefined, leading to the development of a data-driven worldview
  - Can be used in distributed optimization

Unsupervised learning

- The way we learn without prior information
- Cognitive stimuli are processed
  - We build a view of the world based on data
  - Generative model: probabilistic view of the world
  - Background for all our cognitive activities
- Has the potential to give an agent the ability to face and react to situations never seen before
- Can use the huge amount of data available
- The worldview provided will be an extremely valuable starting point for goal-specific learning

Learning in networks

- So far, we have discussed learning in nodes
- When dealing with populations of agents, we can refer to more complex phenomena, e.g.:
  - Evolutionary approaches mimic natural evolution
  - Emergent behaviors in agent based systems
  - Intelligent behaviors may emerge from simple entities (swarm intelligence)
- Network optimization as emergent property
- Nature has solved the scalability problem

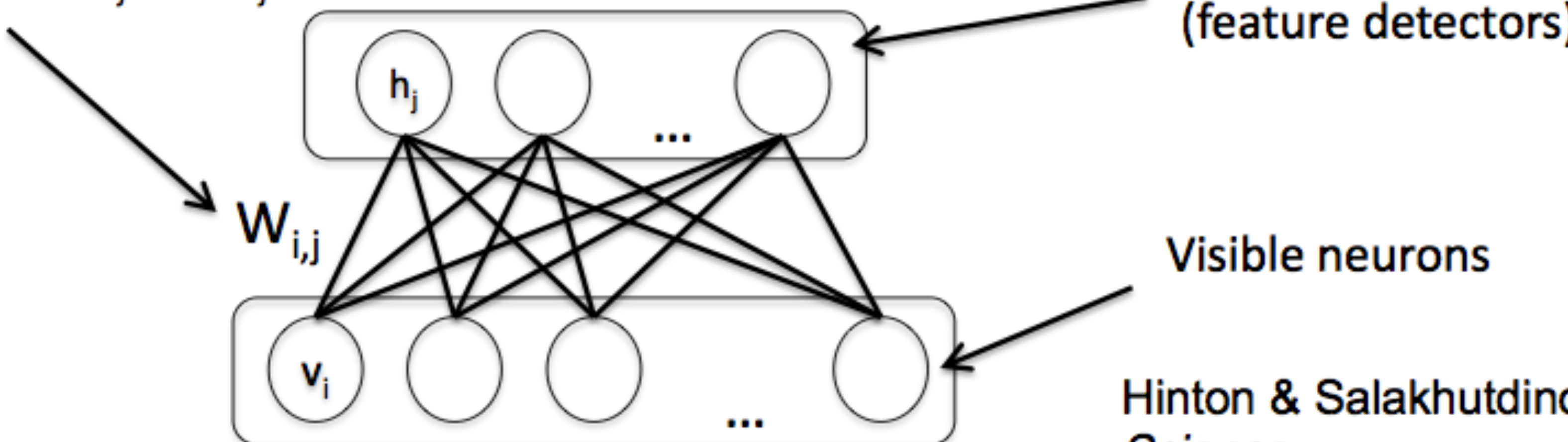
Cognition-based Network

- Each node of the network:
  - exploits local information to achieve its goal
  - shares (part of) it with its neighbors
- Self-adaptation to the environment to achieve network wide goals
- Cognition applied to the entire network (not just at the PHY and MAC layers)
  - Both vertically and horizontally

Restricted Boltzmann Machine (RBM)

- Stochastic neural network with input layer symmetrically connected with a hidden layer of feature-detectors
  - (probabilistic graphical model: undirected graph)
- Unsupervised learning of an internal model of the data (features or latent causes).
- Objective function: minimize contrastive divergence (Kullback-Liebler) between input data and (top-down) reconstruction of the data

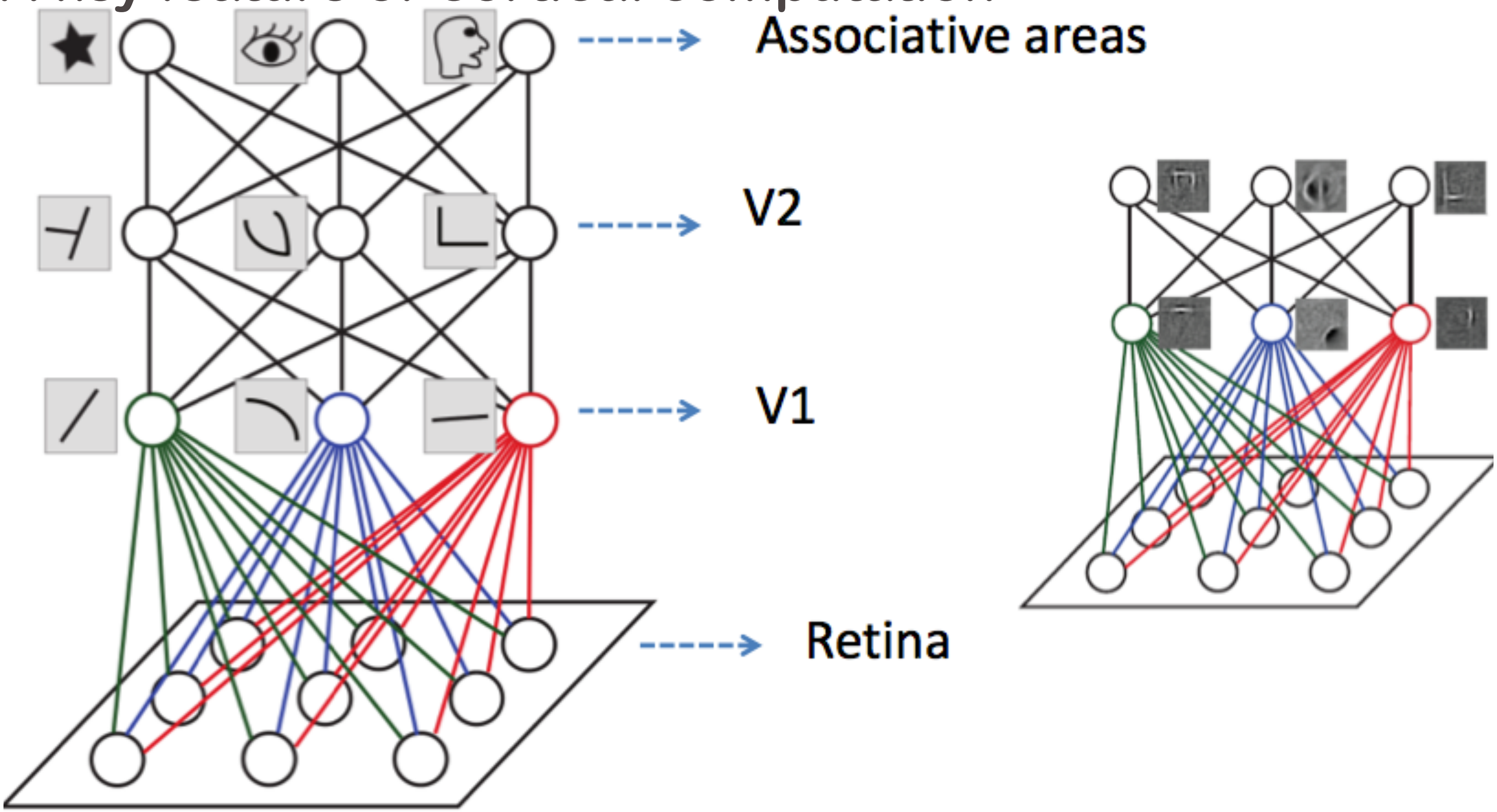
$$W_{i,j} = (v_i^+ h_j^+ \quad v_i h_j)$$



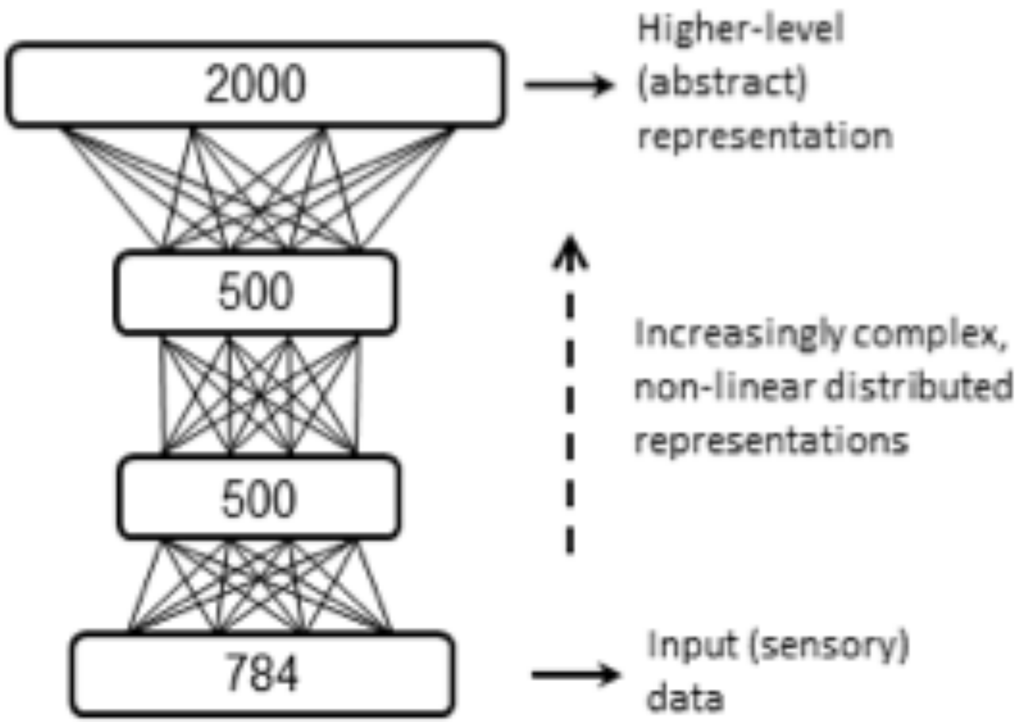
Hinton & Salakhutdinov, 2006, *Science*

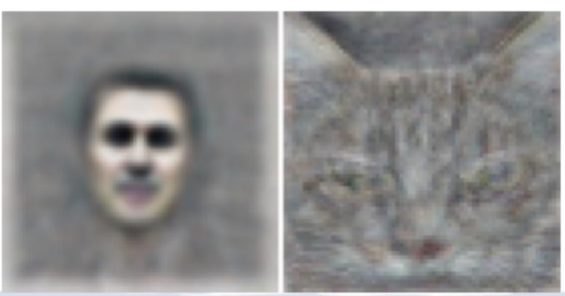
Hierarchical processing

A key feature of cortical computation



Deep learning





Large scale simulation
Deep network 9 layers (1 billion connections)
10 million (200x200) unlabeled images

(Le et al., 2012)

- Bottom-up:
  - Higher layer neurons "activate" when input presents some specific features
  - Higher layer provides an abstract representation of input features
- Top-down:
  - Activating hidden layer neurons according to their weight and propagating back toward inputs we can generate signals with similar features of training signal

Advantages of deep learning

- Classification objective: divide inputs according to certain criteria
- Standard approach: supervised training of classifier
  - e.g., linear classifier, or neural network
  - representative set of input signals with associated classes
  - Apply classifier to new signals and look at outputs
- Classification is (often) better if classifier is trained by using higher-layer deep network neurons as inputs in place of original signal

Examples of application

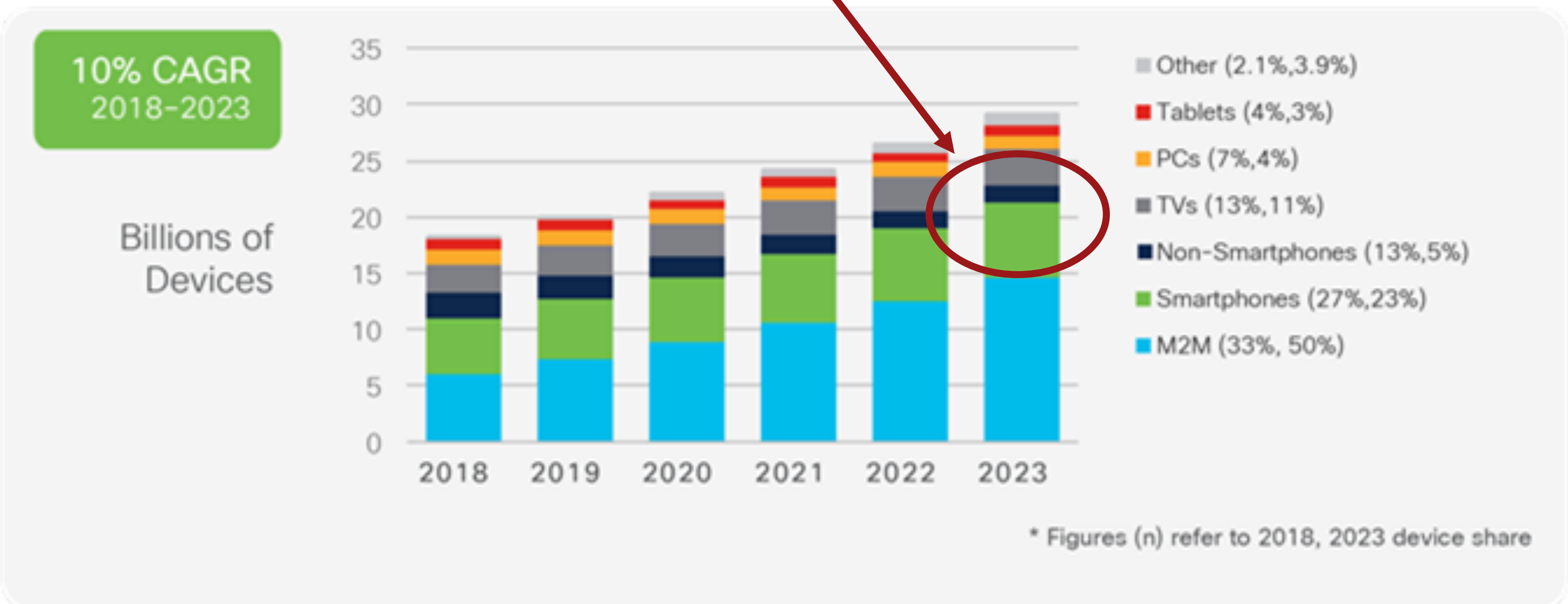
- **Content-based video management**
  - Learn video features and apply this knowledge to some useful networking task
- **Context-dependent handover in HetNets**
  - Learn environmental features and make context-based handover decisions accordingly
- **Prediction of Mobile Devices Discharging Time**
  - Learn the usage pattern and predict the battery charge duration of a smartphone
- …

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Deep Learning for Video Streaming Characterization

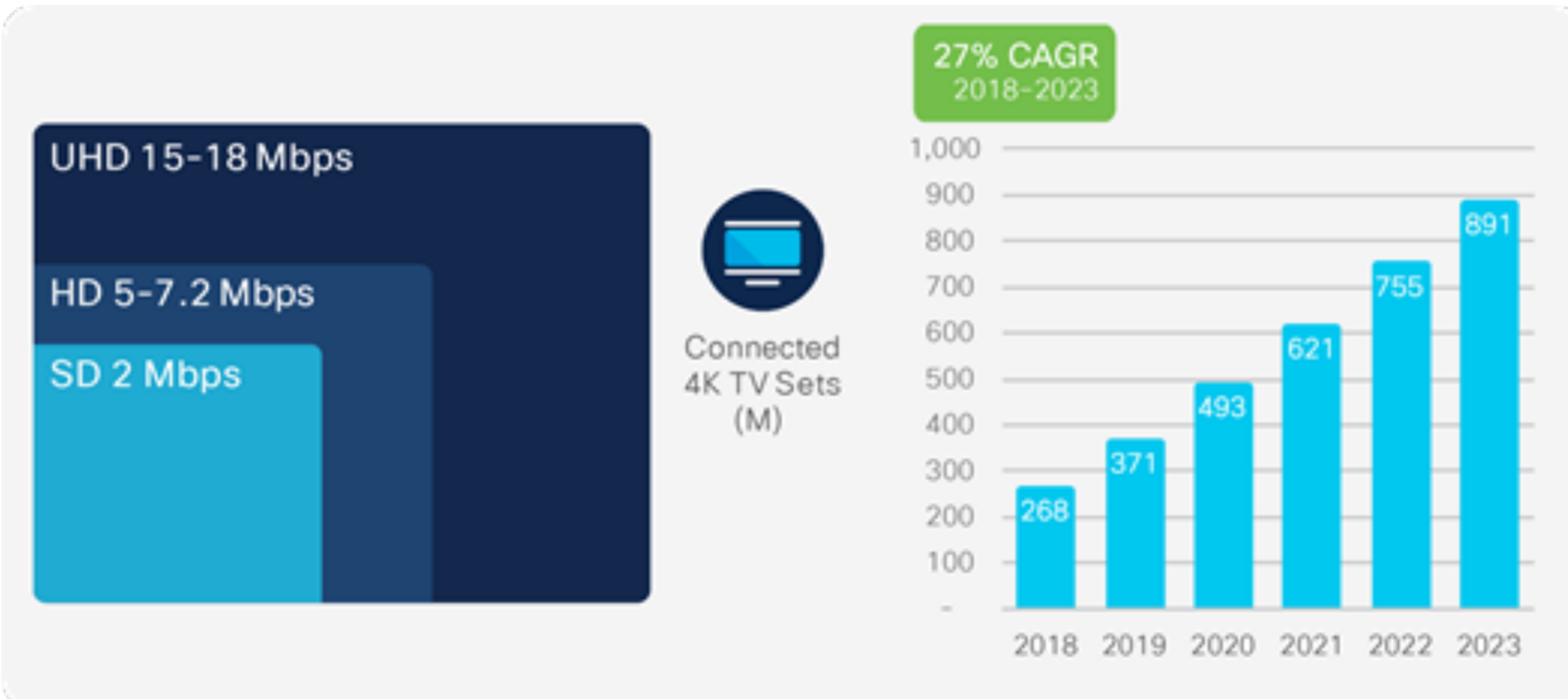
Type of devices

Smartphone & TV counts for about half of connected devices but generate most of the traffic!



Source: Cisco Annual Internet Report (2018–2023) White Paper

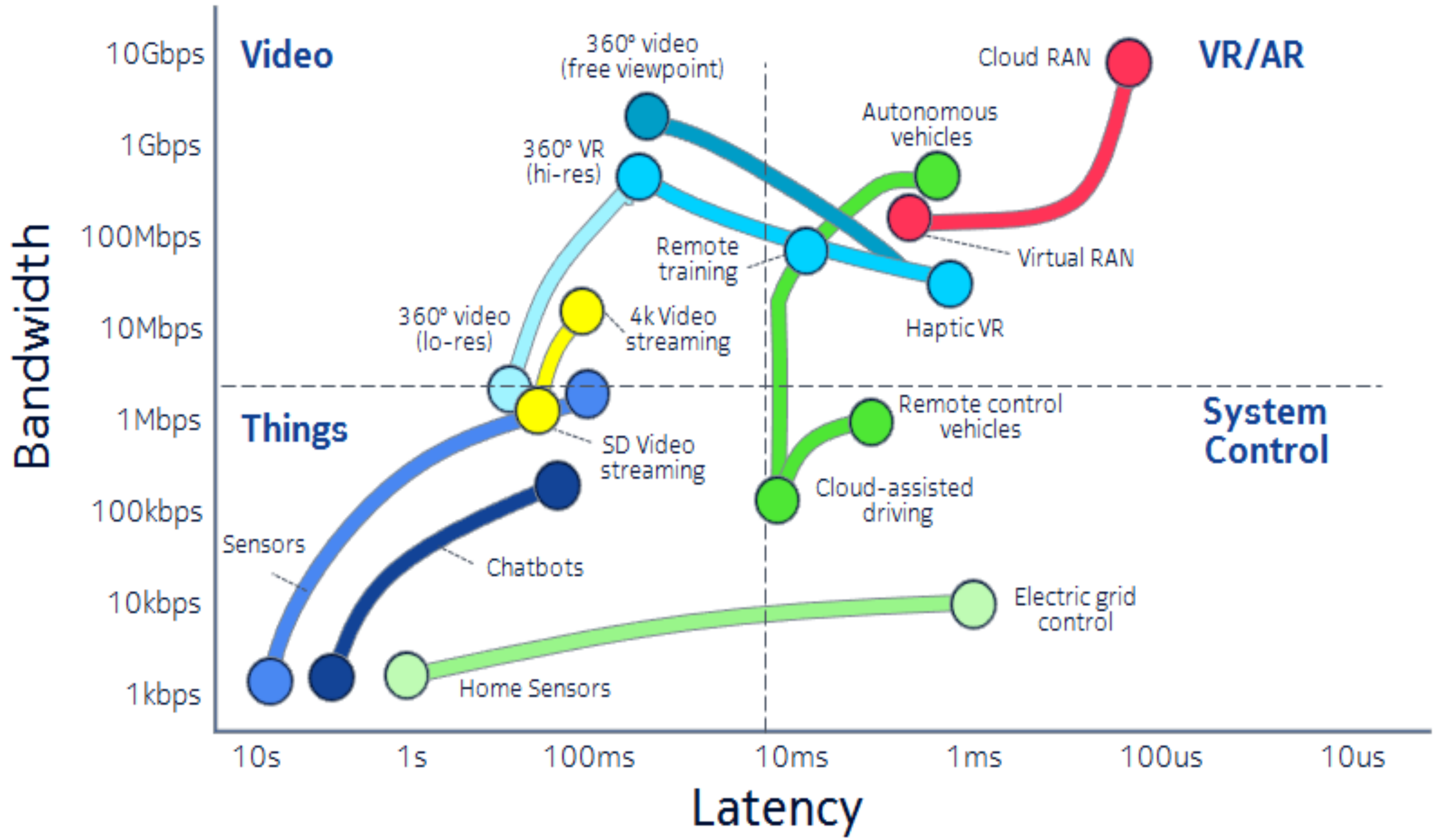
Connected TV sets



By 2023, 66 percent of connected flat-panel TV sets will be 4K

Source: Cisco Annual Internet Report (2018–2023) White Paper

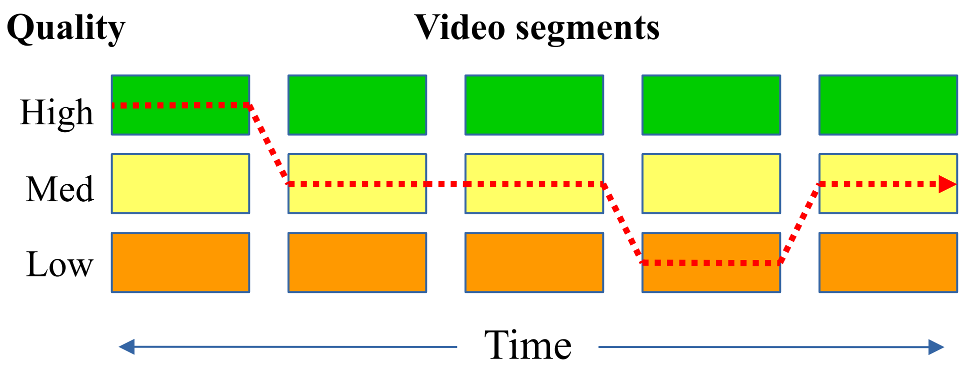
Bandwidth vs latency

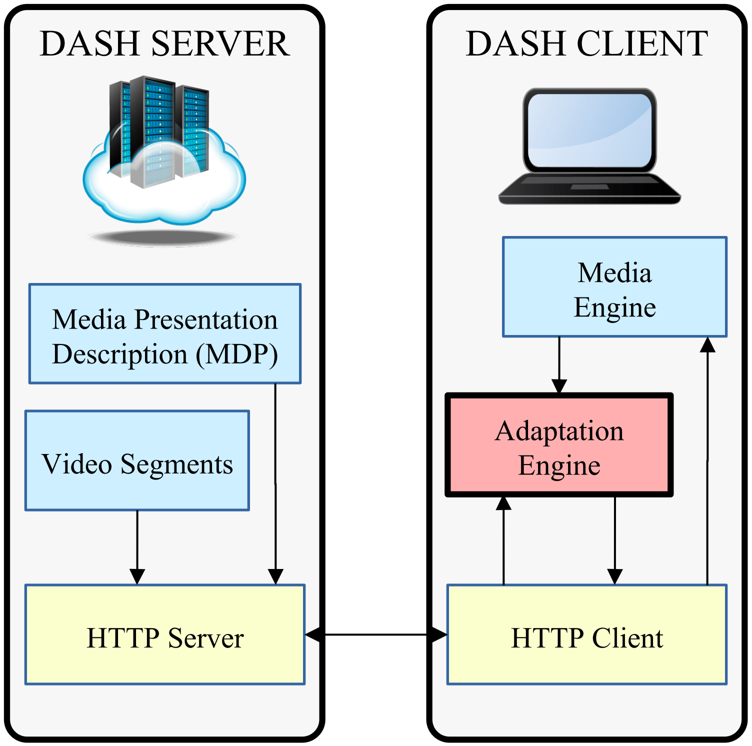


Challenges

- Mobile video streaming is very demanding:
  - High bitrates
    - Compressed HD videos still requires 2-10 Mbit/s
  - Low delay
    - Less than 10 ms for interactive videos, less than 250 ms for real-time streaming
  - Stable links
    - Link fluctuations are counteracted by a (small) playout buffer → if the buffer empties then the play freezes!
  - Content-dependent requirements
    - Quality-rate characteristic depends on the video content

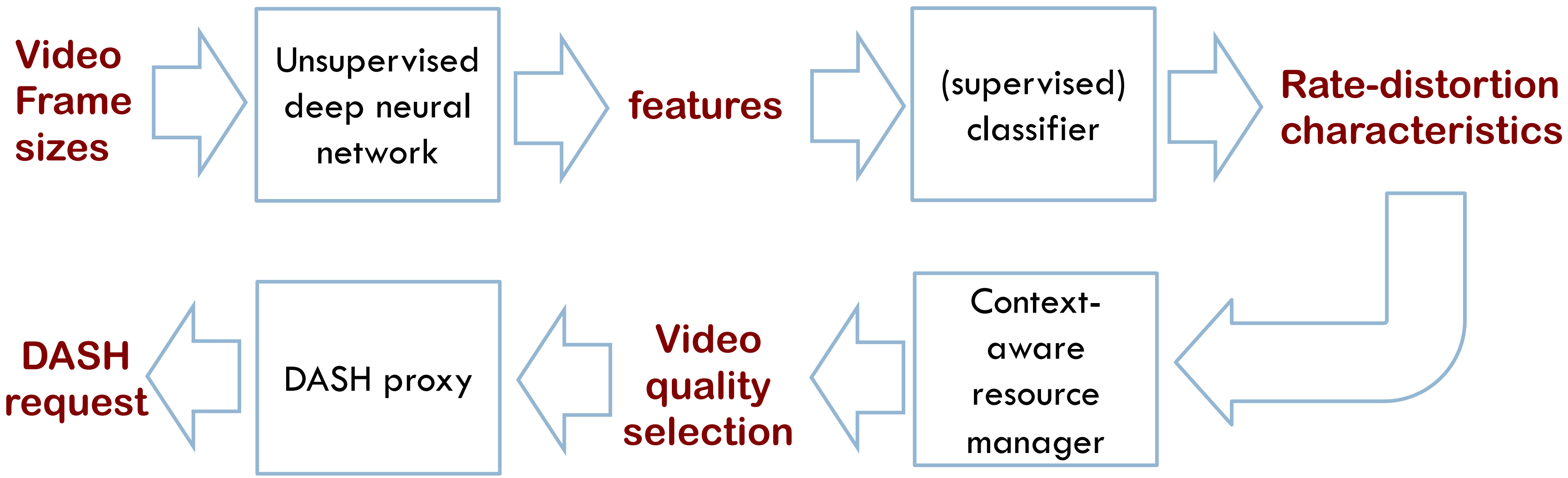
DASH - Dynamic Adaptive Streaming





- HTTP Standard Server
- Client-side Adaptation Engine
  - The client chooses the proper segments
  - Maximize Quality of Experience (QoE) e.g. avoid rebuffering...

Our approach



Analysis

- We consider a test set of 38 video clips, all encoded in an H.264-AVC format
- All the videos are encoded with a 16-frame structure (1 I-frame, 15 P-frames) and compressed with 18 different quantization strategies
  - Transmit rate [bit/s] of video v at compression level c: $r_v(c)$
  - Rate Scaling Factor (RSF): $\rho_v(c)=\log(r_v(c)/r_v(1))$

QoE characterization

- Depending on the content, the perceived quality of a given compression level changes
- There are several metrics to measure quality of a video signal
- Here, video quality is expressed in terms of Structural Similarity (SSIM)

Structural Similarity (SSIM)

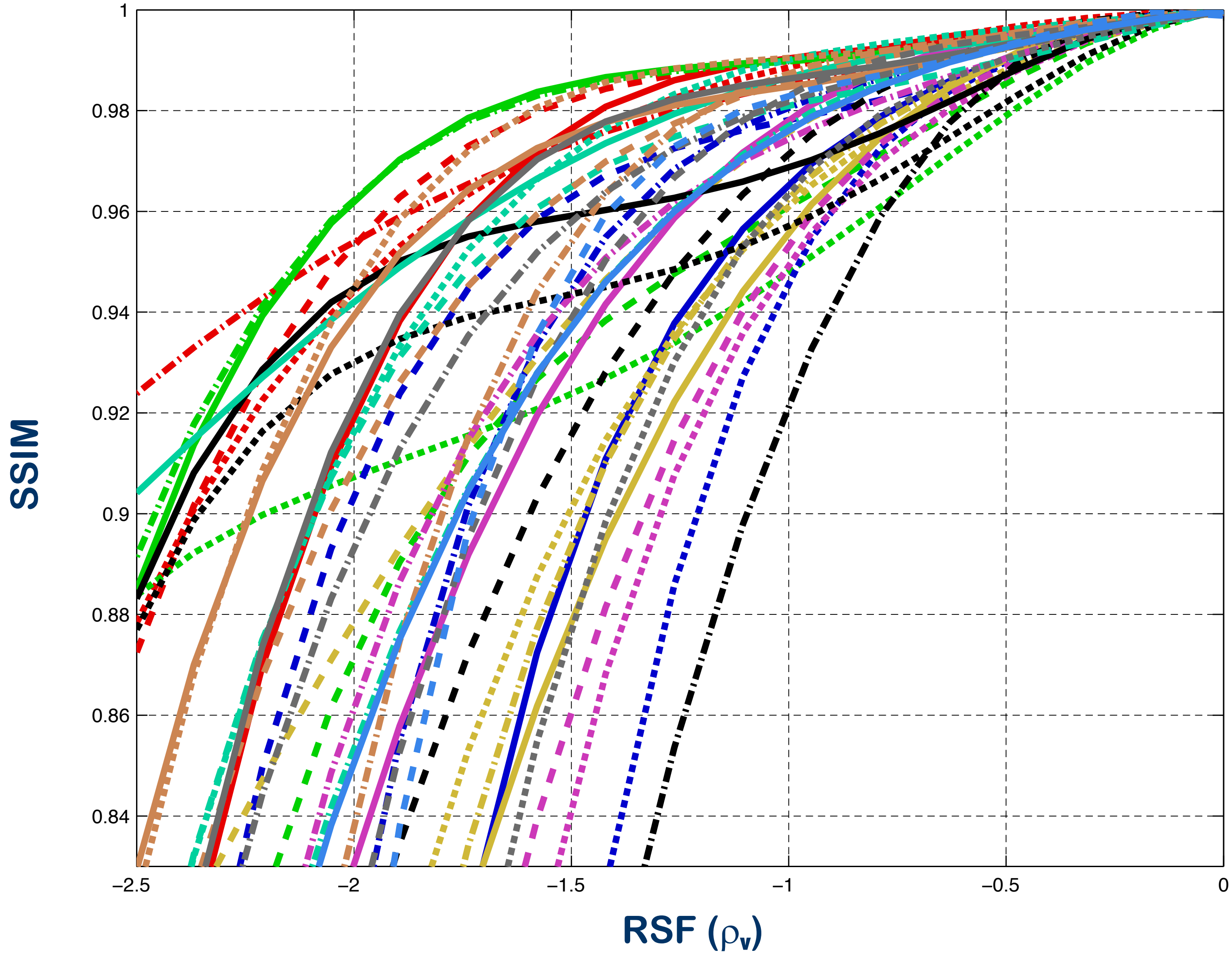
- SSIM measures the closeness of square sets of pixels, and is computed as

$$SSIM(X,Y) = \frac{(2\mu_X\mu_Y + c_1)(2\sigma_{XY} + c_2)}{(\mu_X^2 + \mu_Y^2 + c_1)(\sigma_X^2 + \sigma_Y^2 + c_2)} \tag{1}$$

with μ and σ^2 denoting the mean and variance of the luminance value in the corresponding window, and c_1 and c_2 being variables to stabilize the division with weak denominator

  - Measures image degradation in terms of perceived structural information change
  - Represents quality as seen by the human eye

SSIM versus RSF



Remarks

- All the videos exhibit similar trends
  - monotonic descent
  - a steep “fall” after a threshold
- However, there are quantitative differences
  - different perceived end quality
  - different resource requirements
- These characteristics are roughly consistent within the same (homogeneous) video

SSIM polynomial approximation

- We introduce a polynomial approximation to express SSIM behavior
  - This provides a compact representation for use in VAC and RM

$$F_v^{(n)}(\rho) \simeq 1 + a_{v,1}\rho + a_{v,2}\rho^2 + a_{v,3}\rho^3 + \ldots + a_{v,n}\rho^n$$

- A 4-degree polynomial provides a quite accurate approximation of the SSIM vs RSF curve

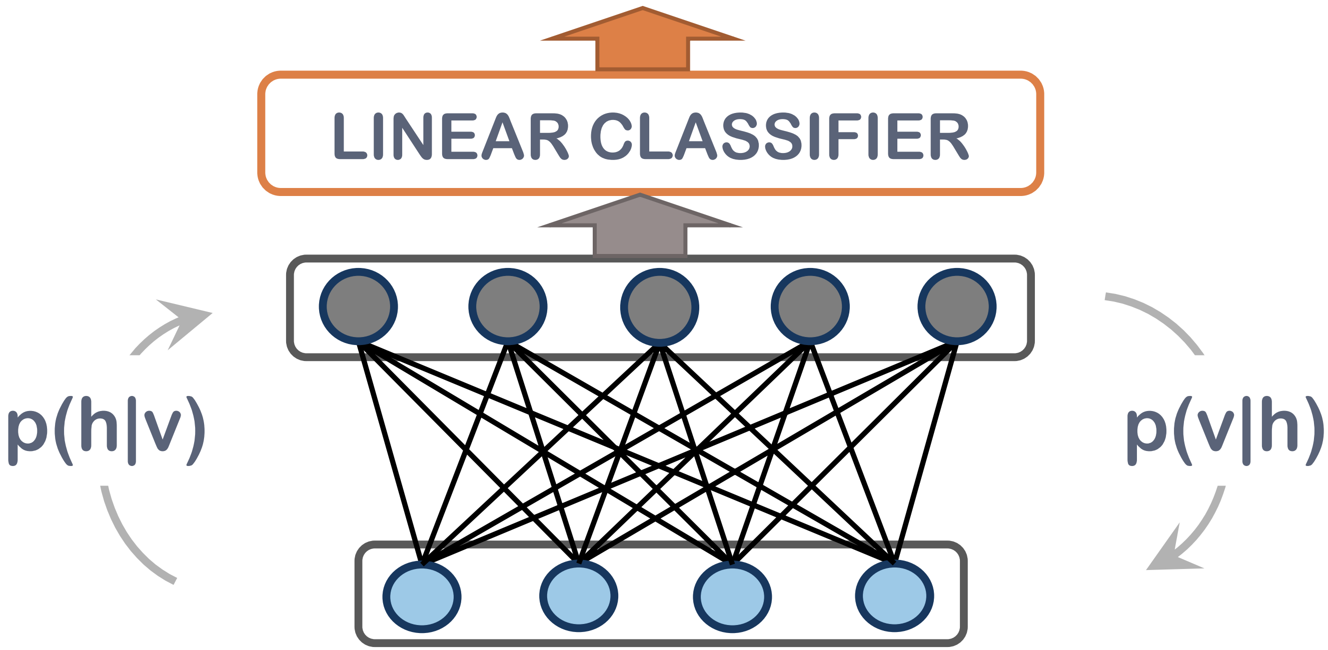
Possible video applications

- Knowledge of these characteristics of the video may be useful for
  - (i) QoE-aware admission/congestion control
  - (ii) determining popular content
  - (iii) inferring user behavioral characteristics

Proposed approach

Video recognition/classification/SSIM estimation

estimation



visible layer:

GoP frames' size

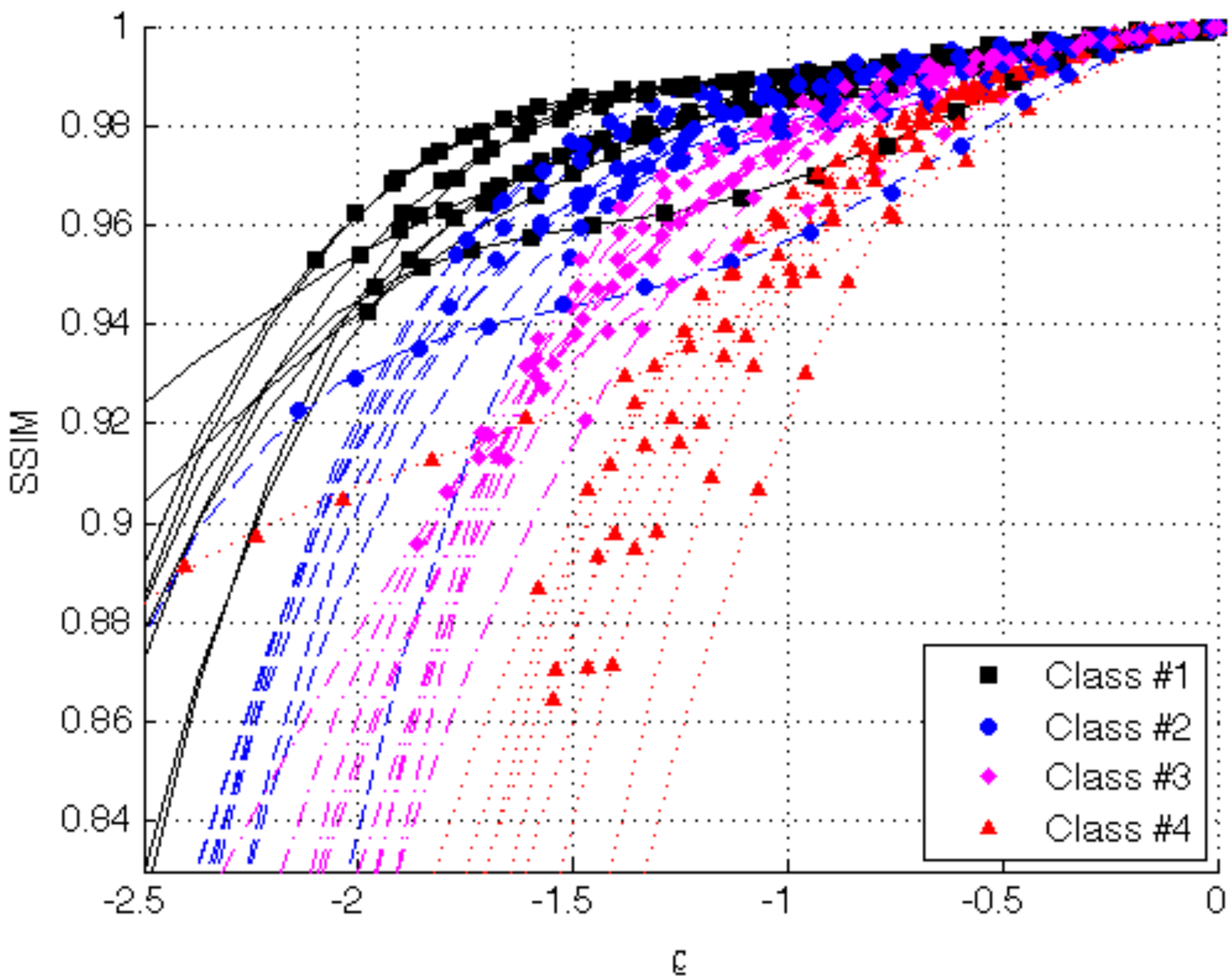
Proposed approach

- Input to the RBM: **frame size only**
  - This is done for a whole GoP (isolated)
  - The RBM “learns” by creating certain patterns in the hidden layer
  - This enables a sparser representation of the input in the hidden layer
- After that, we apply a linear classifier for recognition / classification / SSIM estimation
  - Note: we must train a different linear classifier for each case, while the RBM is trained just once for all

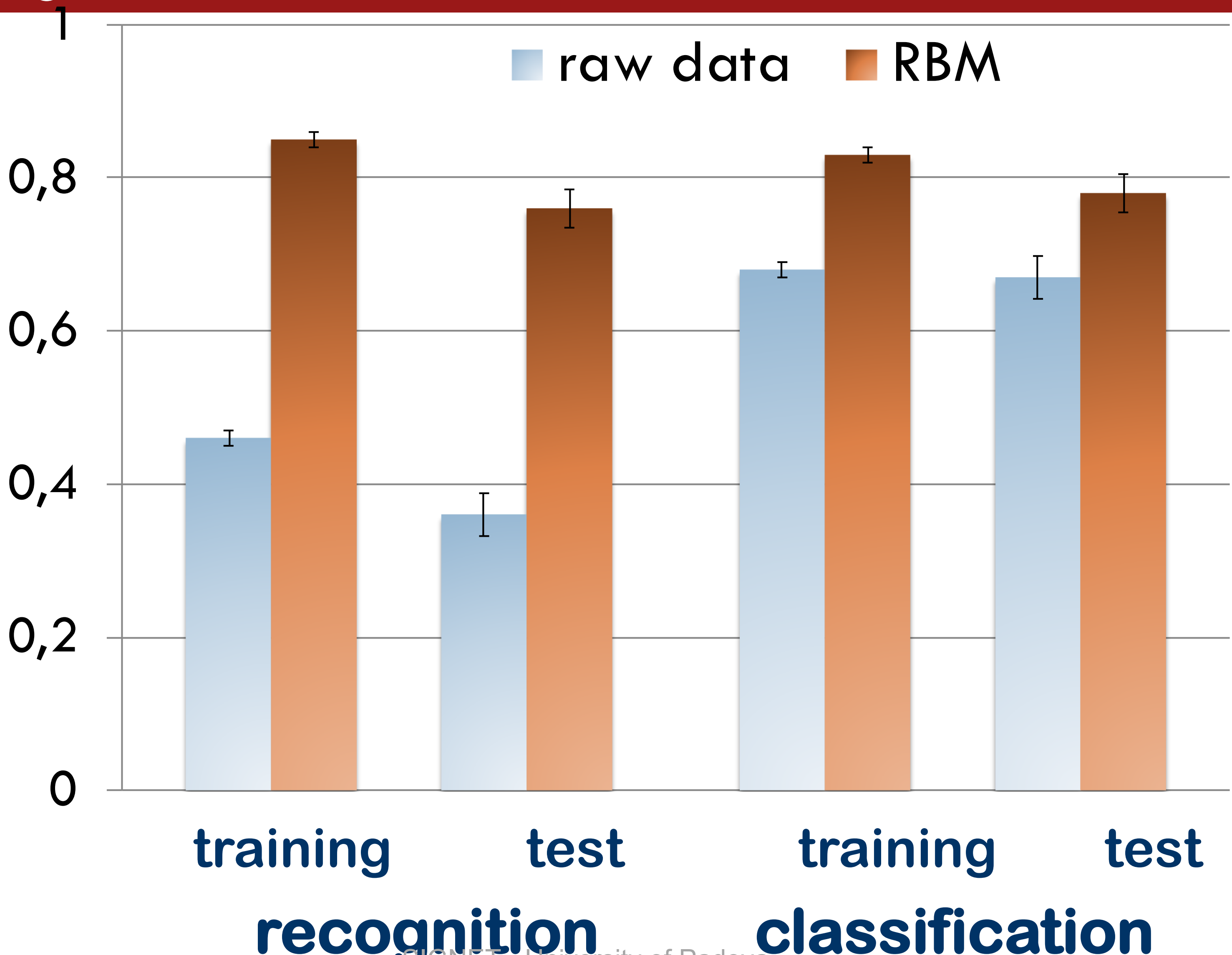
Setting the learning machine

- Dataset is split: Training Set - Test Set
- Input of the RBM (visible layer): 32 units
  - for each of the 16 frames, size of the uncompressed version and of the version compressed at intermediate rate 9
- Hidden layer set (empirically) to 70 units
- The result is compared to just using the raw data as the input of the classifier

SSIM-based Video Classification

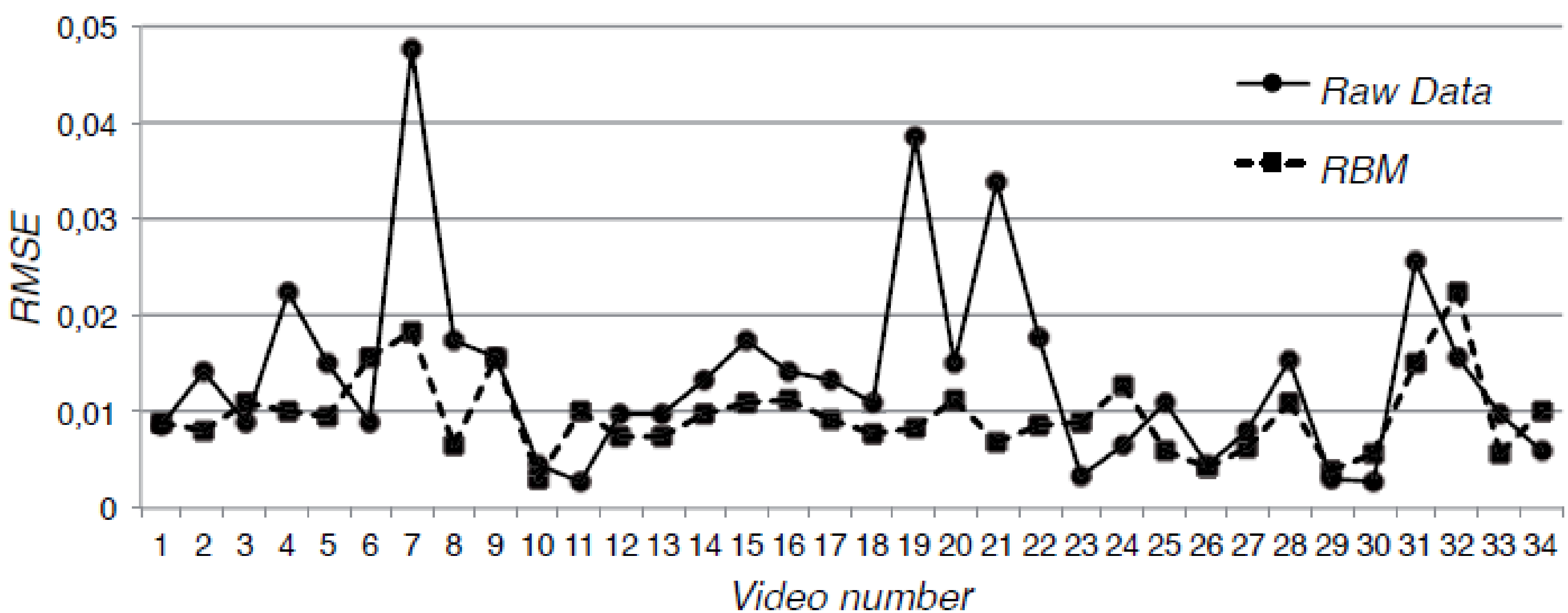


Preliminary results

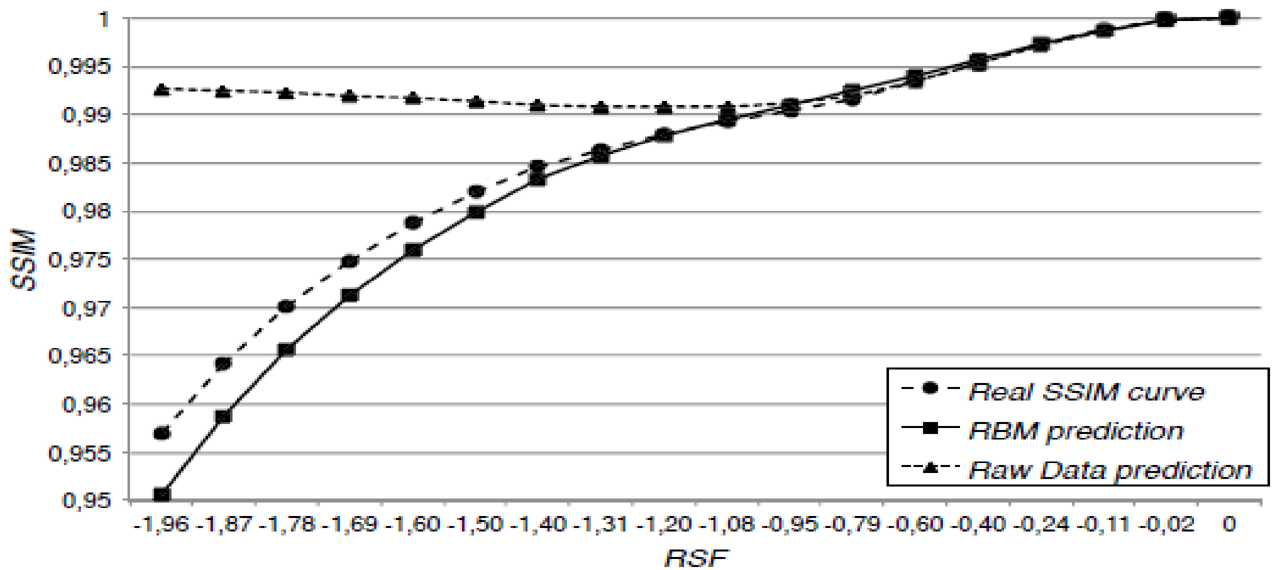


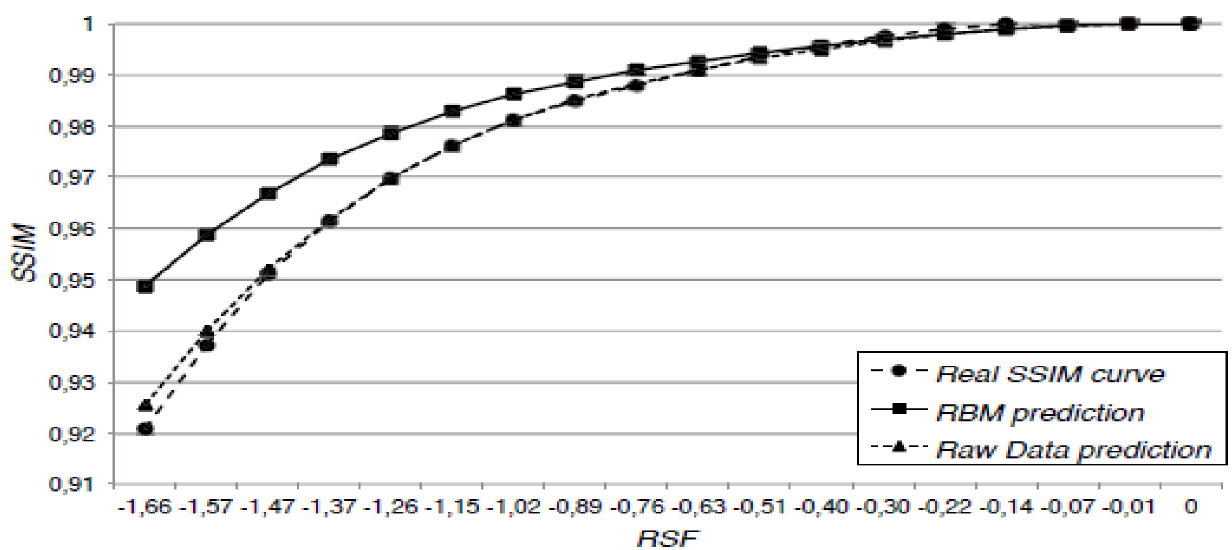
Learning via RBM: accuracy

- Root Mean Square Error (RMSE) between exact SSIM and polynomial approximation with estimated coefficients



Exact vs. estimated SSIM curves for two random videos





Cognitive Video Admission Control & Resource Management

- Videos multiplexed into a shared link of capacity R
- **Resource Manager (RM)**: detects changes and triggers optimization to adapt video rates to maximize QoE utility function
- **Video Admission Controller (VAC)**: determines whether a new video request can be accepted without decreasing QoE of any video below a threshold F*

Cognitive VAC

- At each new video flow request, VAC invokes RM to get the "best" resource allocation according to a specified policy
- RM returns the resource that can be assigned to each video
- VAC computes the SSIM of each video with the best compression level, compatible with the allotted resources
  - If estimated SSIM of all active videos is above the quality threshold the video request is accepted

- The optimization problem addressed by RM is as follows

Resource share allotted to video “v”

$$\Gamma_{\text{opt}} = \arg\max_{\Gamma} \overbrace{U(\Gamma, R, \{F_v\})}^{\text{Utility function}} \quad \text{s.t.} \quad \sum_v \gamma_v \leq 1$$

Rate allocation vector

Channel rate

SSIM videos’ characteristics

Cognitive RM algorithms

Possible utility functions

Rate fairness (RF)

$$U(\Gamma, R, \{F_v\}) = \left[\sum_v \left| \log \frac{\gamma_v R}{r_v(1)} \right| \right]^{-1}$$

SSIM fairness (SF)

$$U(\Gamma, R, \{F_v\}) = \min_v F_v(\log(\gamma_v R / r_v(1)))$$

Resource share to be allotted:

RF:

$$\gamma_v = \frac{r_v(1)}{\sum_j r_j(1)}$$

SF:

$$\gamma_v = r_v(1) 10^{F_v^{-1}(\phi)}$$

where

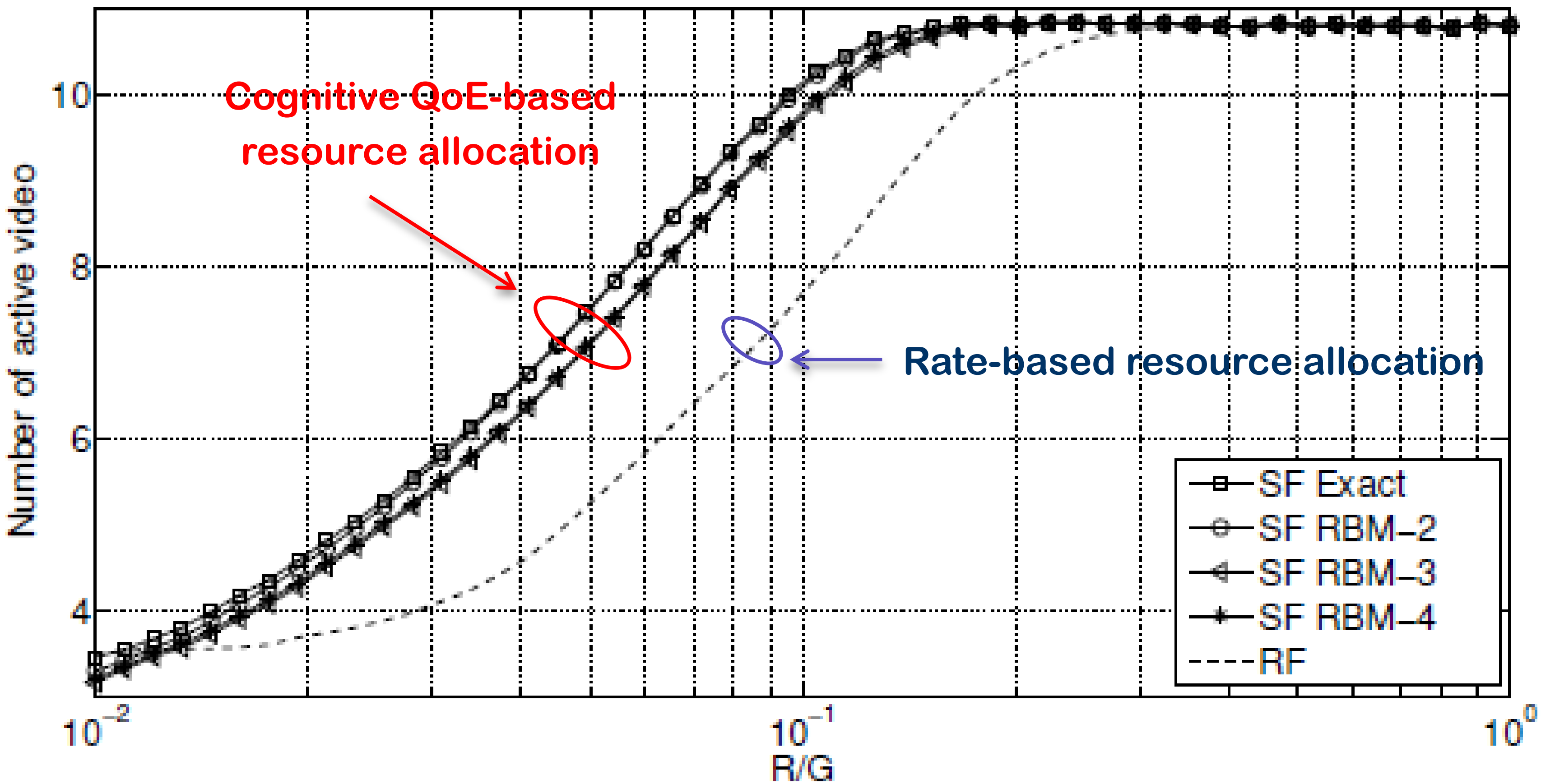
$$\phi = \arg\max_\phi \left\{ \sum_v r_v(1) 10^{F_v^{-1}(\phi)} \leq R \right\}$$

Simulation setup

- Poisson video requests (0.66 req./s)
- average offered load of 11 videos
- Aggregate max rate of G = 161 Mbit/s
- F* = 0.95 SSIM value to reach MOS 4

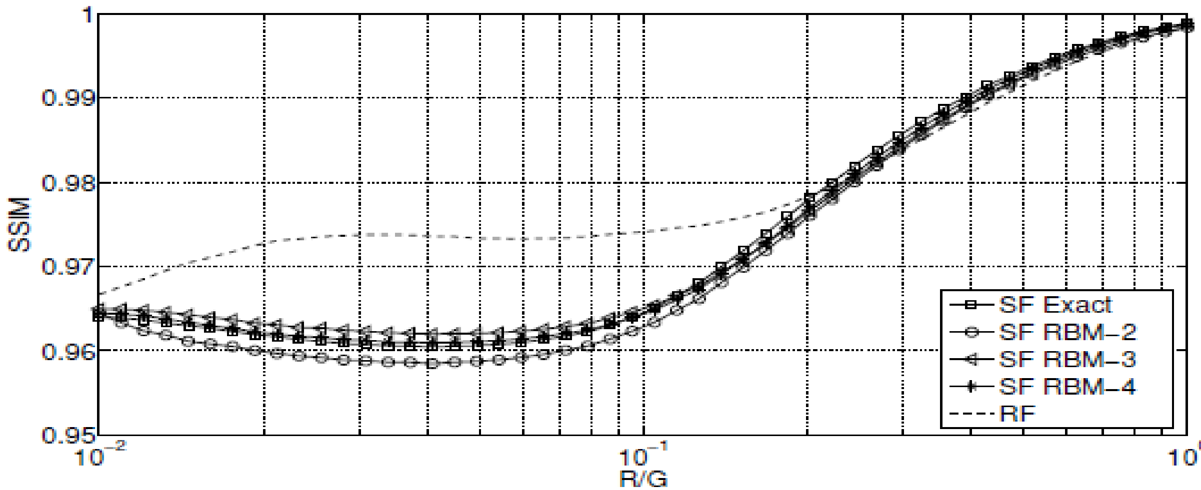
SSIM	MOS	Quality	Impairment
≥ 0.99	5	Excellent	Imperceptible
$[0.95, 0.99)$	4	Good	Perceptible but not annoying
$[0.88, 0.95)$	3	Fair	Slightly annoying
$[0.5, 0.88)$	2	Poor	Annoying
< 0.5	1	Bad	Very annoying

Cognitive video admission control

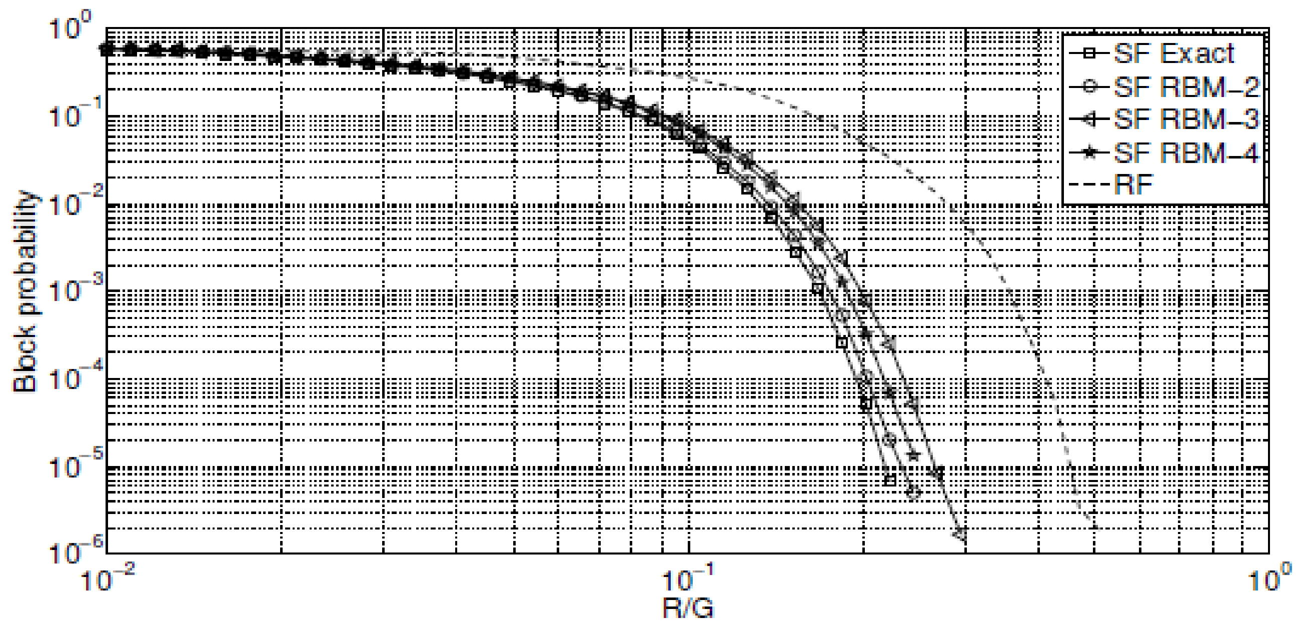


Link rate over aggregate full-quality video rate

Effect of SSIM approximation



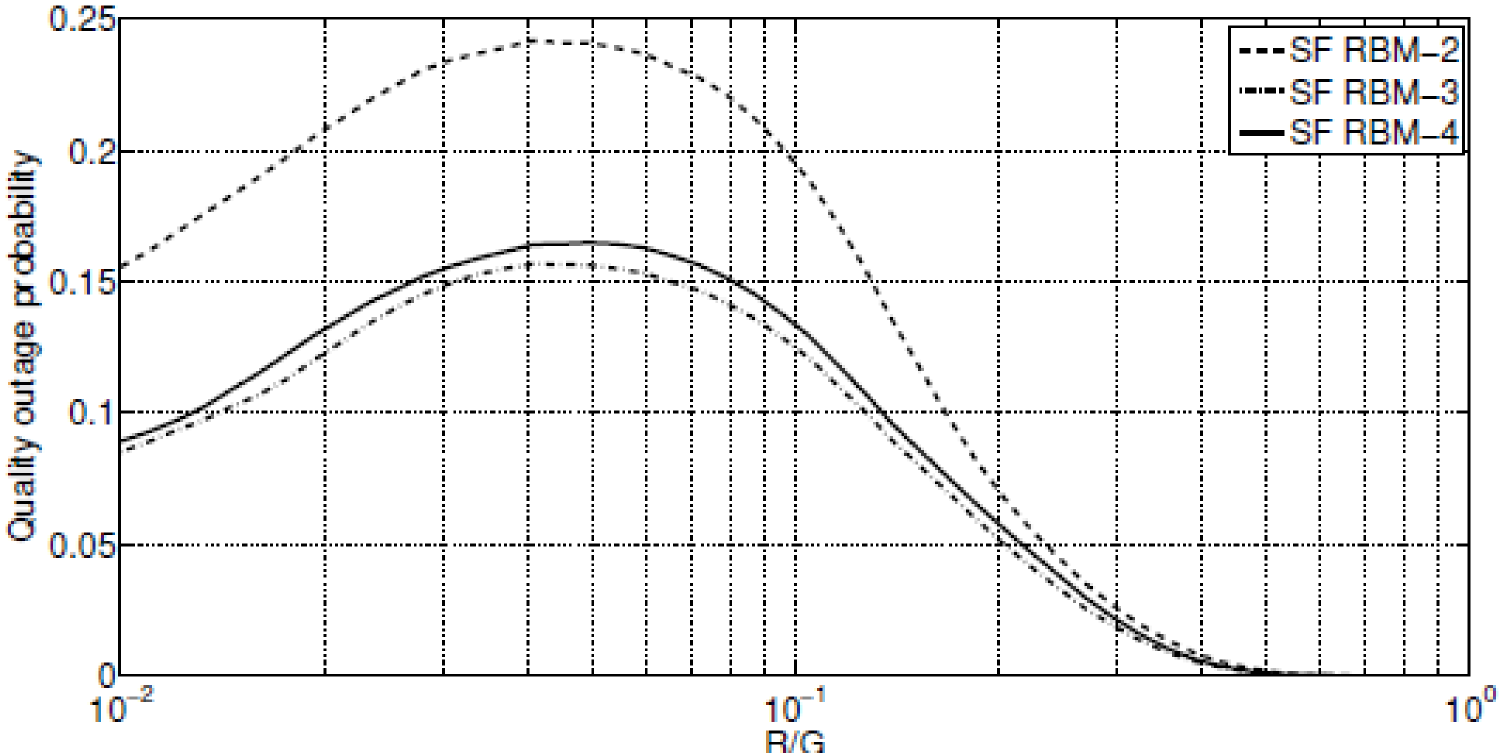
SF RBM-n: SF algorithm, with n-degree polyn. approx. of SSIM curve obtained by RBM approach



Quality outage probability

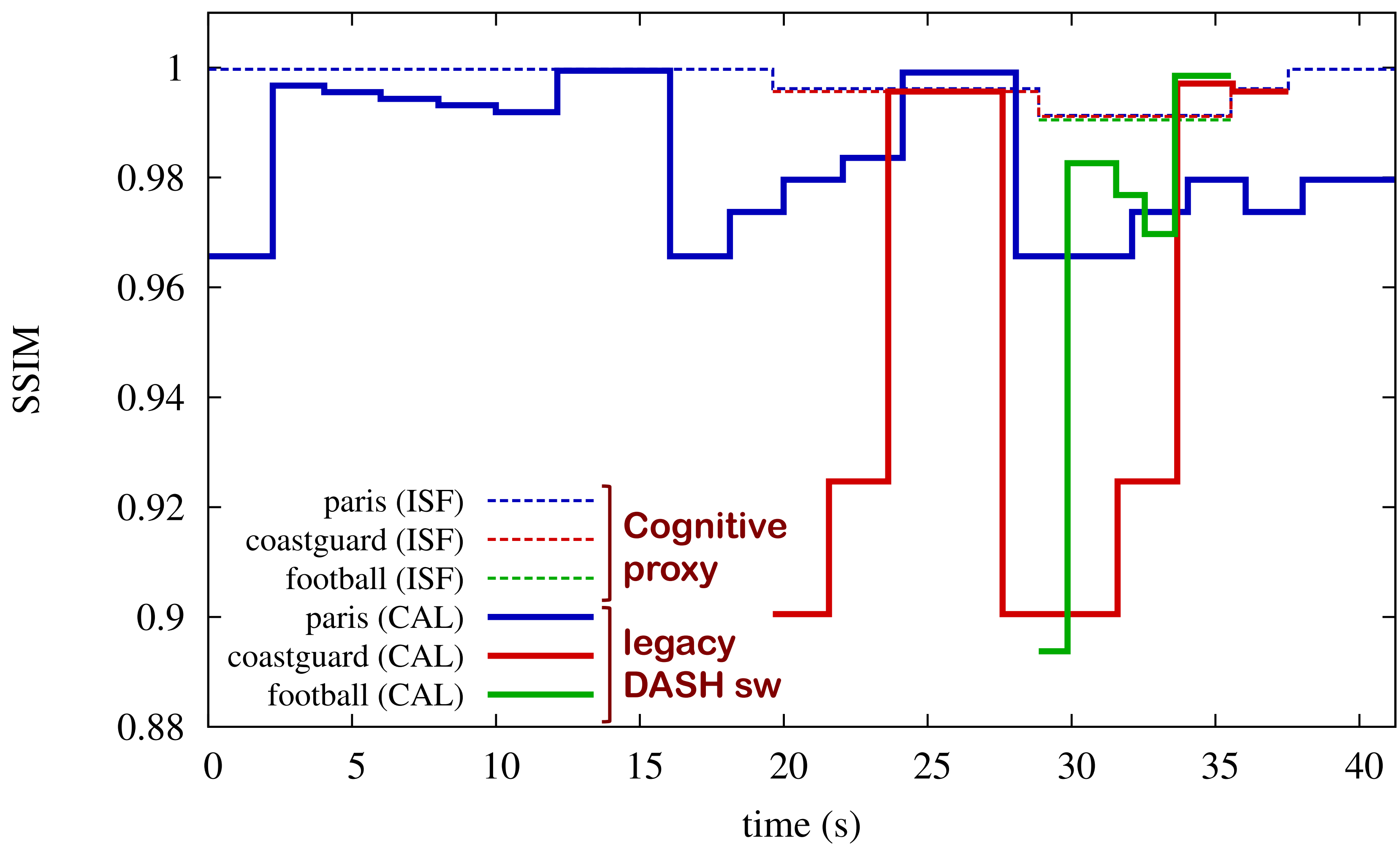
- Because of SSIM approximation errors, videos may occasionally be accepted even if quality threshold is not met
- Tradeoff on SSIM polynomial approximation: the fewer the coefficients, the coarser the approximation, but the better the RBM coefficient estimate
- Question: is it better to have a well-estimated low degree or a coarsely estimated high-degree polynomial?

Quality outage probability



- n=2 is too small, n=3 or 4 give very similar results: n=3 is the best choice in this case

QoE-aware proxy vs legacy video clients



DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Deep Reinforcement Learning for DASH Video Streaming

Video streaming problem

- A user is watching a movie through her smartphone
- Connection quality may change over time due to mobility and network congestions
- Video quality can also change over time to adapt to connection variations
  - Video quality reduces when video buffer is almost empty to speed up downloading of the next segment
  - If buffer empties, video playout freezes while next few segments are downloaded from the server
- Quality of Experience is heavily affected by freezing events, low quality, or continuous quality variatons

Problem setting

- The problem is hence to determine the quality of the next video segment to download from the server in order to maximize the QoE
- Suppose video is split in segments of 1 second each
- Each segment can be downloaded with different quality levels
  - The higher the quality, the larger the segment size
- Every second, the video client chooses the quality of the next segment to download

Defining the problem as MDP

- Can you set the problem as a MDP?
- State?
- Action space?
- Reward?

DASH-DQN

- State

$$s_t = [q_{t-1}, \boldsymbol{C}_t, D_t, B_t]$$

- Buffer size
- Video segment complexity
- Wireless channel capacity vector
- Previous quality (SSIM)

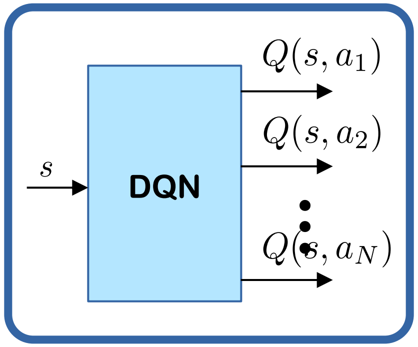
- Reward: quality of Experience (QoE)

Freezing time

$$r(q_t, q_{t-1}, f_t) = q_t - \beta\|q_t - q_{t-1}\| - \lambda f_t$$

- Actions (q): choose among 8 segment qualities levels

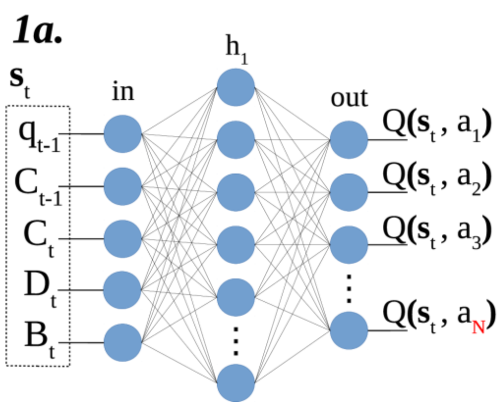
Deep Q-Network (DQN)

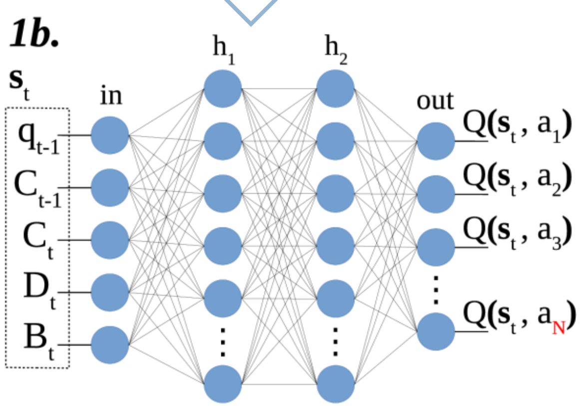


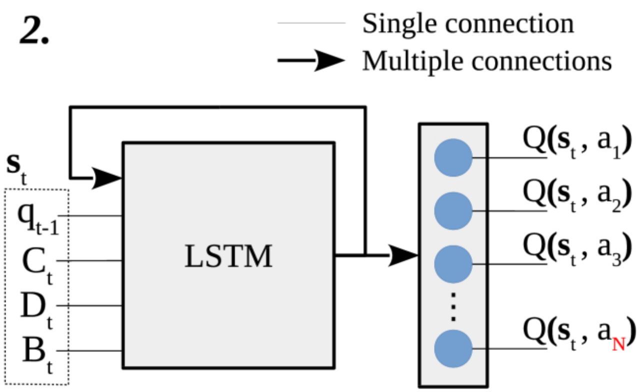
Multi-layer Perceptron with 1 hidden layer: MLP1

Multi-layer Perceptron with 2 hidden layer: MLP2

Long-Short Term Memory: LSTM







Neural Network Training

- Loss function for segment t

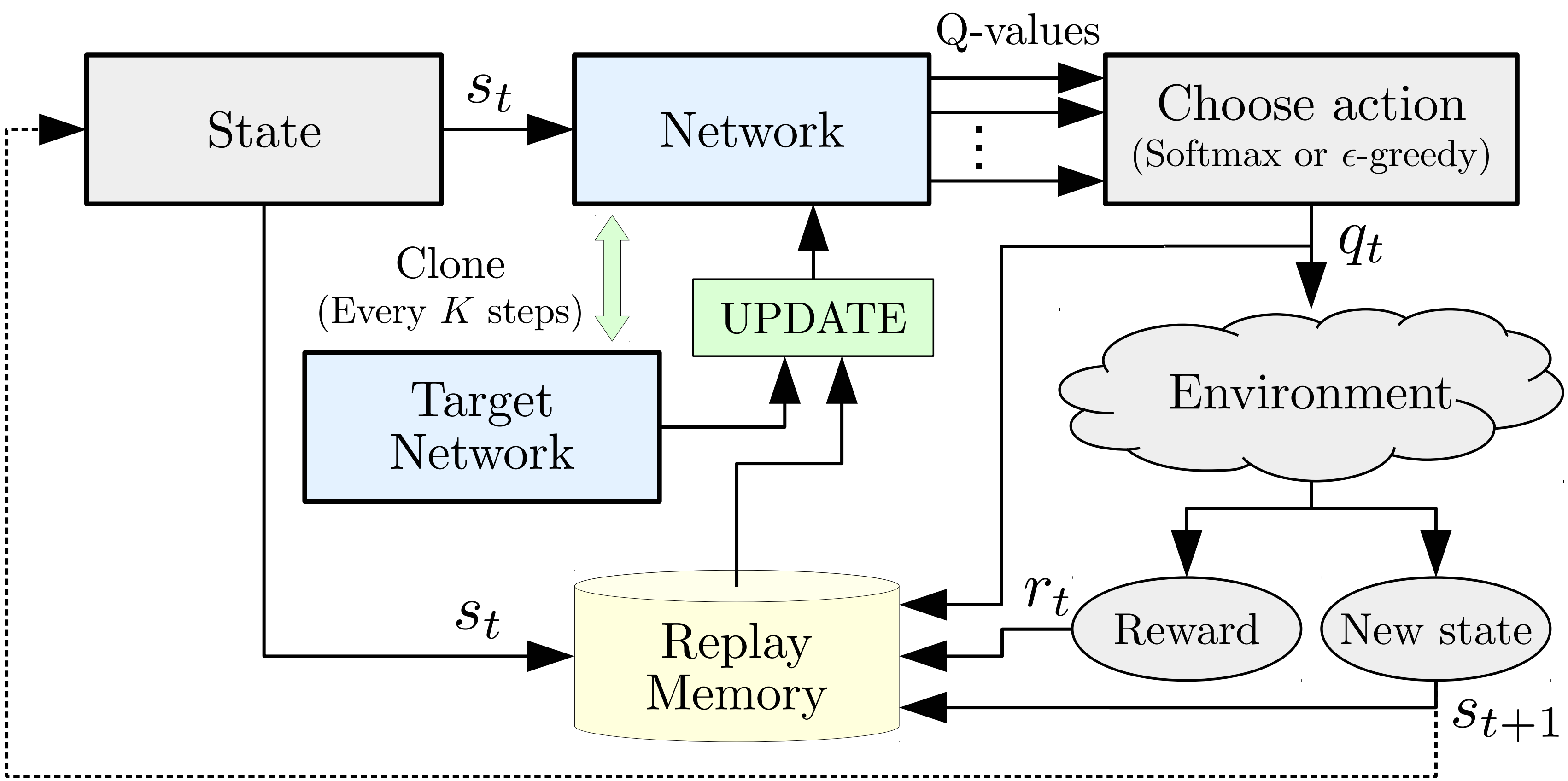
long-term reward estimate
through Bellman equation

$$L_t = \left(r_t + \lambda \max_q \hat{Q}(s_{t+1}, q | \overline{\boldsymbol{w}}_t) - Q(s_t, q_t | \boldsymbol{w}_t) \right)^2$$

- r_t : reward accrued for segment t
- Two neural networks are used [Mnih15]
  - **First neural network**
    - **updated for each new segment (every t)**
    - **weight vector $\boldsymbol{w}_t$**
  - **Second "target network"**
  - **Updated every K segments (learning steps)**
    - **Setting** $\overline{\boldsymbol{w}}_t = \boldsymbol{w}_t \text{ if } (t \mod K) == 0$

[Mnih15] V. Mnih *et al.*, Human Level Control through Deep Reinforcement Learning, Vol. 518, No. 7540, *Nature*, 2015.

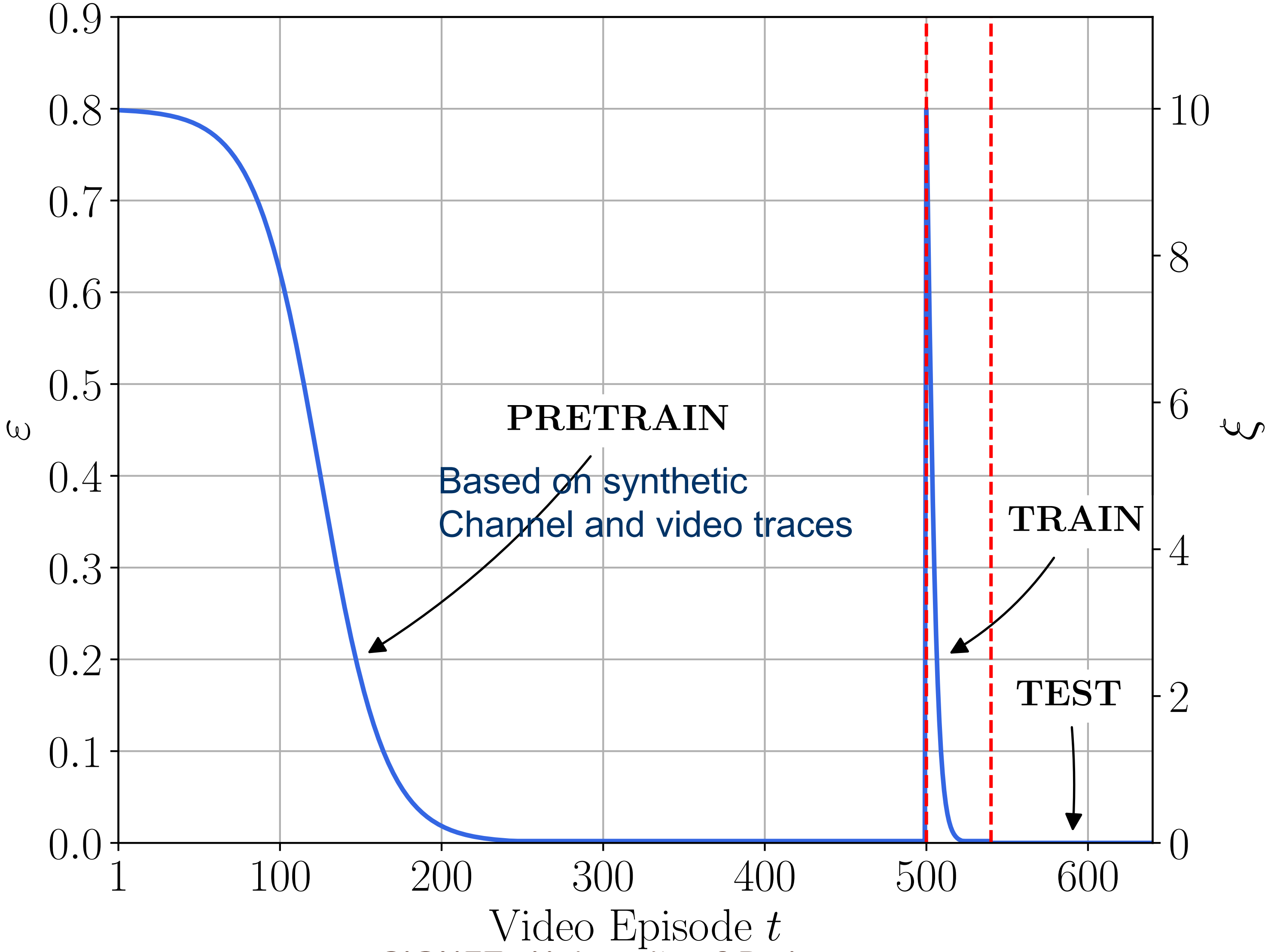
Update iteration



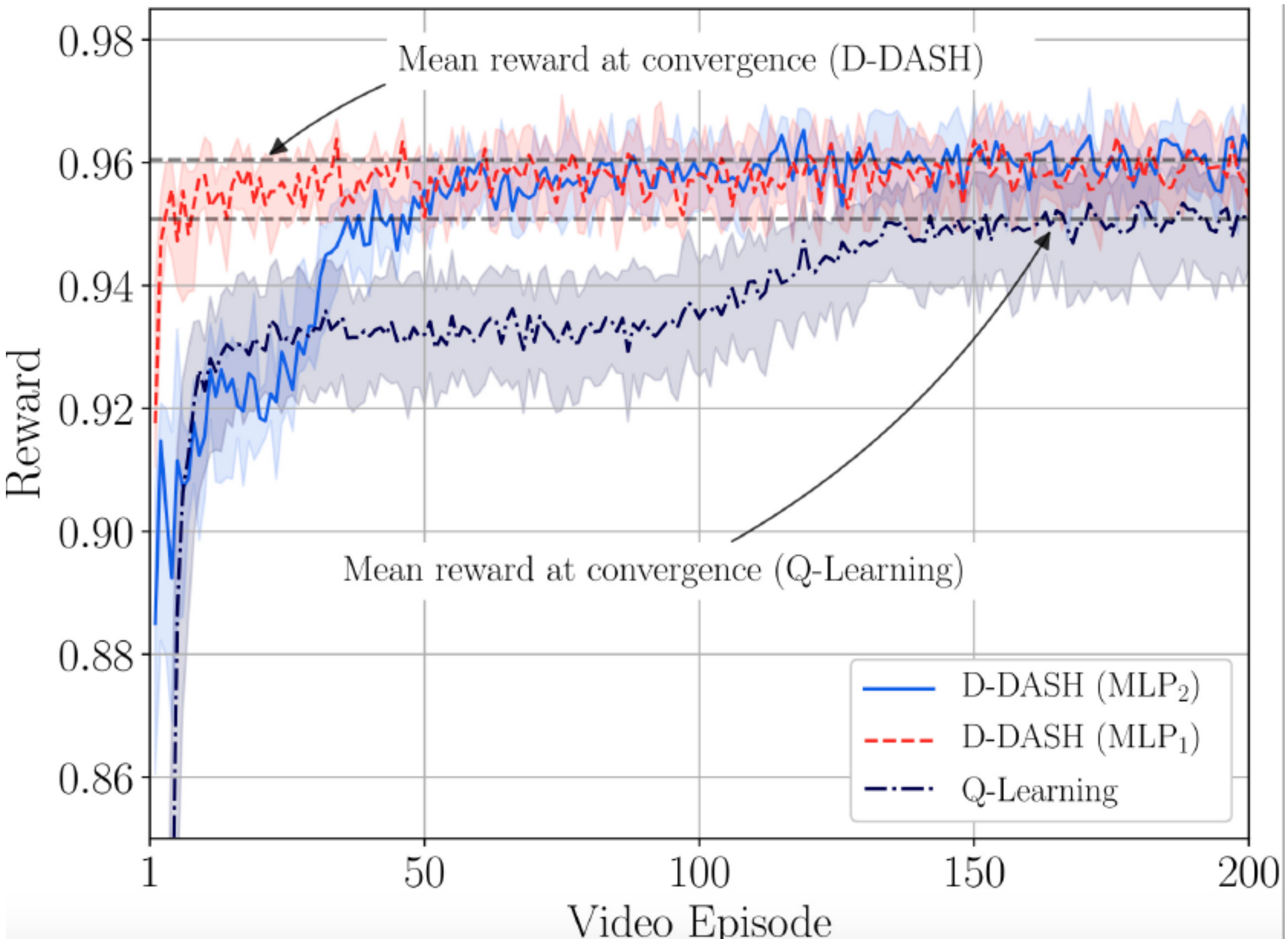
Simulation

- Video model: 5 complexities, exponential scene duration
- Capacity model for the wireless channel
  - Markov - 500 pre-training videos
  - Real - 40 training videos + 100 test videos
  - NS3 - 40 training videos + 100 test videos
- Comparison
  - Rate-based
  - FESTIVE
  - Q-Learning

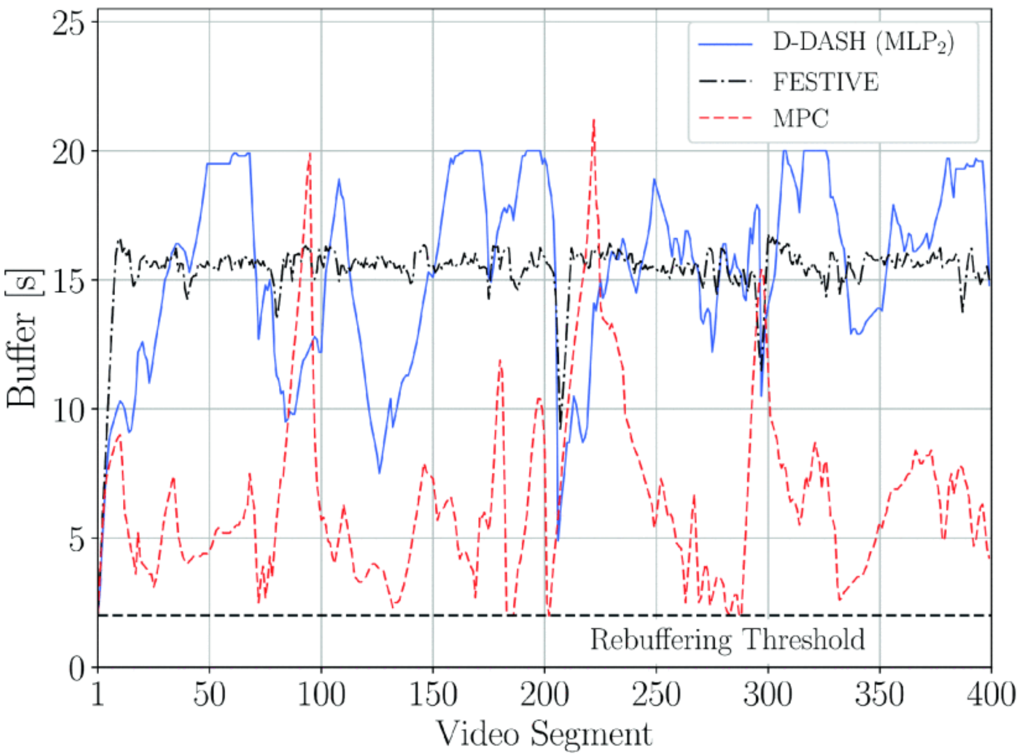
Exploration/training/test



Convergence time



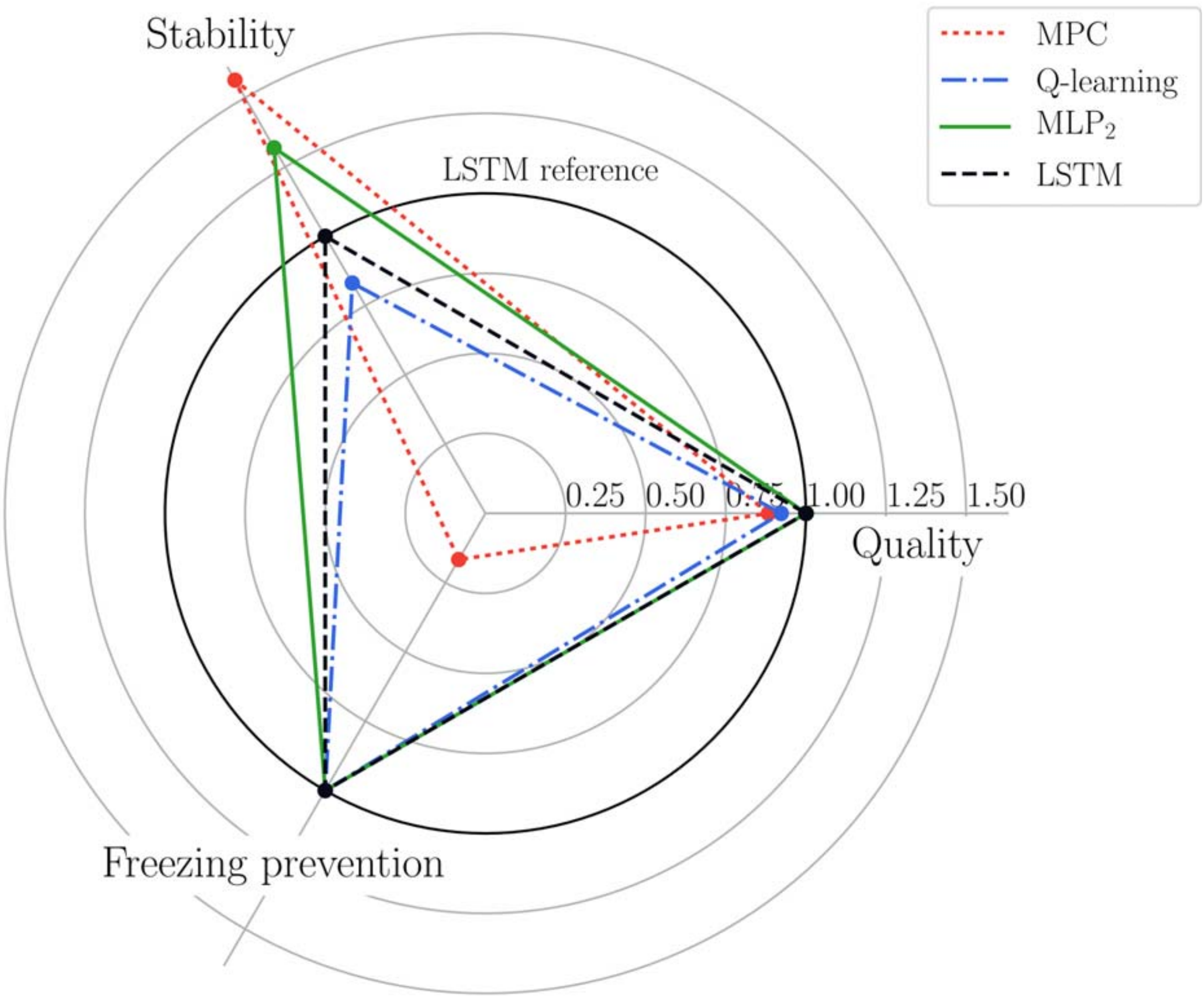
Buffer



- D-Dash fully exploits the buffer to survive connection fluctuations while keeping constant video quality
- Legacy DASH clients try to keep buffer level constant, adapting video quality (worse QoE)

Comparison along three axes

- Quality variations
  - Stability
- Video quality
  - Quality
- Playout buffer
  - Freezing events



Conclusions

- Optimizing resource allocation for video transmission is challenging
  - many numerical parameters involved
  - subjective QoE issues
  - high signaling exchange

- Learning-based approaches are useful to
  - obtain a compact representation
  - extrapolate the most significant data
  - Provide a framework with no need for prior models

Publications

- M. De Filippo De Grazia, D. Zucchetto, A. Testolin, A. Zanella, and Ma. Zorzi, and M. Zorzi, "QoE Multi-Stage Machine Learning for Dynamic Video Streaming," IEEE Transactions on Cognitive Communications and Networking (TCCN) vol. 4, n. 1, pp. 146 - 161, March 2018.
- M. Gadaleta, F. Chiariotti, M. Rossi, and A. Zanella, "**D-DASH: a Deep Q-learning Framework for DASH Video Streaming**," IEEE Transactions on Cognitive Communications and Networking in IEEE Transactions on Cognitive Communications and Networking, vol. 3, no. 4, pp. 703-718, Dec. 2017.
  DOI: 10.1109/TCCN.2017.2755007
- Daniele Munaretto, Andrea Zanella, Daniel Zucchetto, Michele Zorzi, "Data-driven QoE optimization techniques for multi-user wireless networks" in the Proceedings of the 2015 International Conference on Computing, Networking and Communications, February 16-19, 2015, Anaheim, California, USA
- Alberto Testolin, Marco Zanforlin, Michele De Filippo De Grazia, Daniele Munaretto, Andrea Zanella, Marco Zorzi, Michele Zorzi, "A Machine Learning Approach to QoE-based Video Admission Control and Resource Allocation in Wireless Systems" in the Proceedings of IEEE IFIP Annual Mediterranean Ad Hoc Networking Workshop, Med-Hoc-Net 2014, June 2-4, 2014, Piran, Slovenia.
- Leonardo Badia, Daniele Munaretto, Alberto Testolin, Andrea Zanella, Marco Zorzi, Michele Zorzi , "Cognition-based networks: applying cognitive science to multimedia wireless networking" in the Proceedings of Video Everywhere (VidEv) Workshop of IEEE WoWMoM'14, 16 June, 2014, Sydney, Australia.
- Marco Zanforlin, Daniele Munaretto, Andrea Zanella, Michele Zorzi, "SSIM-based video admission control and resource allocation algorithms" in the Proceedings of WiOpt workshop WiVid'14, May 12-16, 2014, Hammamet, Tunisia.

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

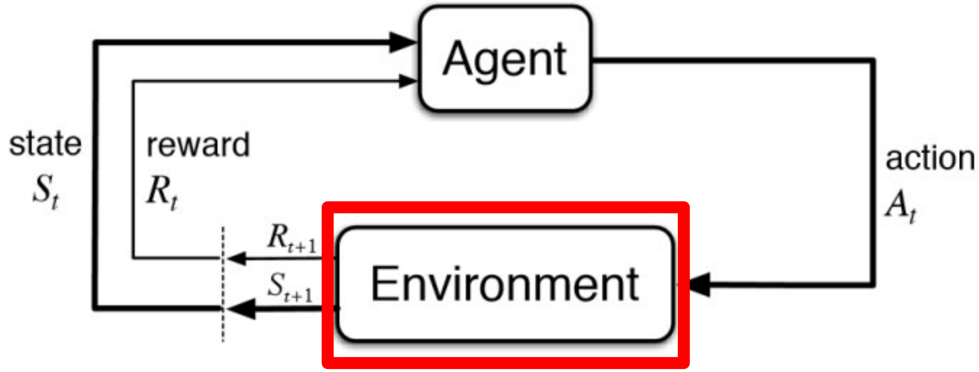
DISTRIBUTED DEEP REINFORCEMENT LEARNING FOR DRONE SWARM CONTROL

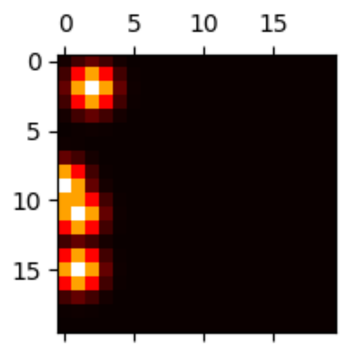
Credits to Federico Venturini, Federico Mason, Federico Chiariotti, Alberto Testolin

DRONE SWARM

- Drone swarm can be used for:
  - Rescue operations
  - Environmental monitoring
  - Delivery services
  - Defence
  - Aerial filming
  - Maintenance

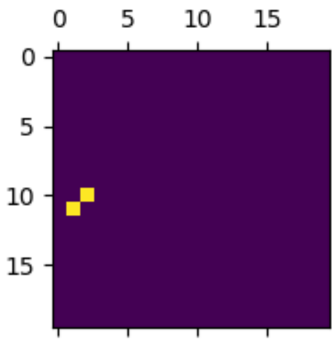
REINFORCEMENT LEARNING

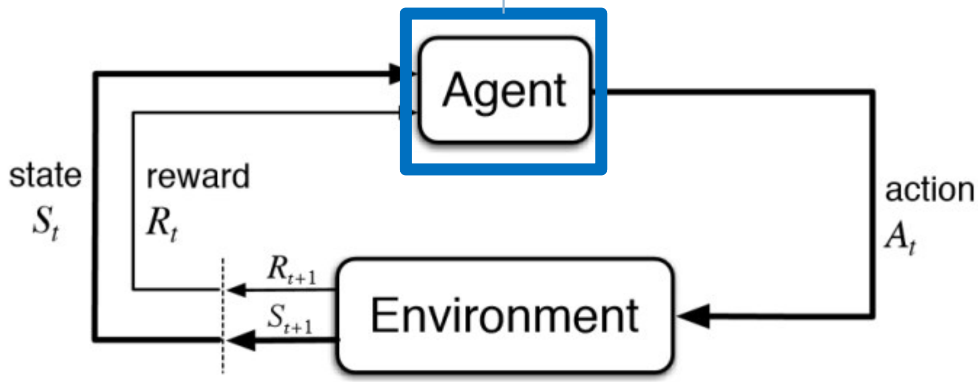




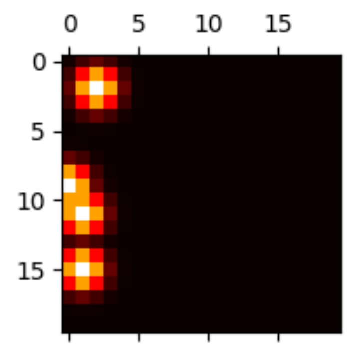
The environment is a map with some targets (the white pixels).

Agent



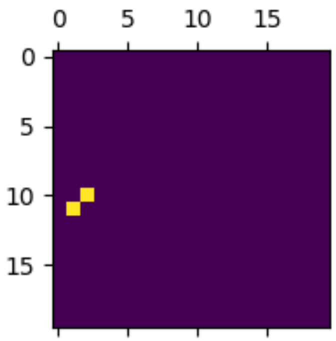


Each drone (yellow pixel) is an agent

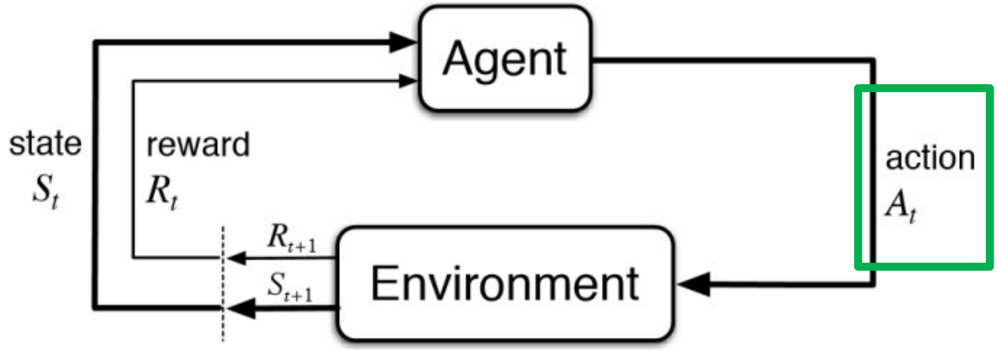


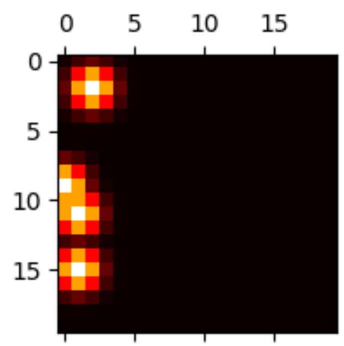
The environment is a map with some targets (the white pixels).

Action



Each drone (yellow pixel) is an agent



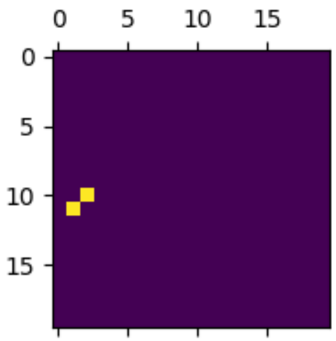


The environment is a map with some targets (the white pixels).

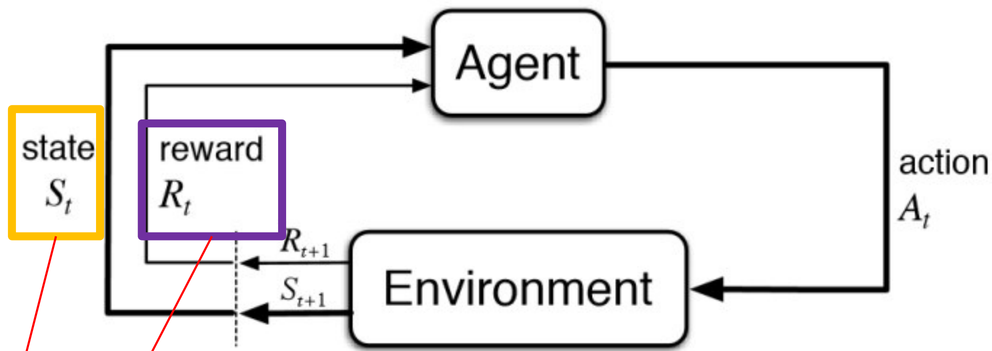
Each agent can choose one of the following 5 actions:

- Stand
- Left
- Right
- Up
- Down

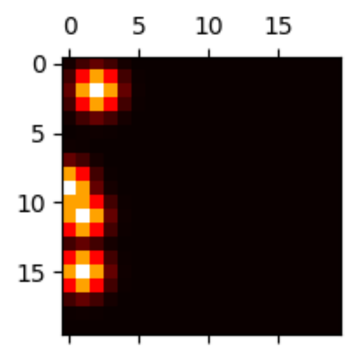
State and rewards



Each drone (yellow pixel) is an agent



Ok, but what are states and rewards?

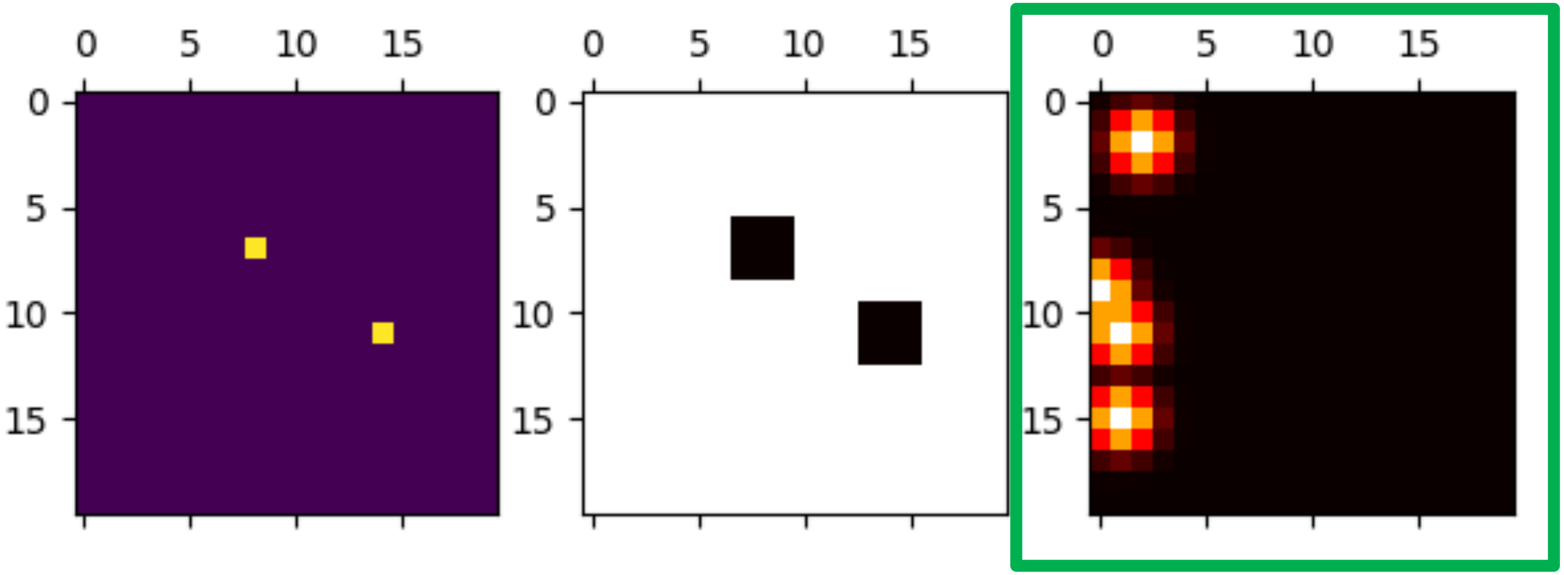


The environment is a map with some targets (the white pixels).

Each agent can choose one of the following 5 actions:

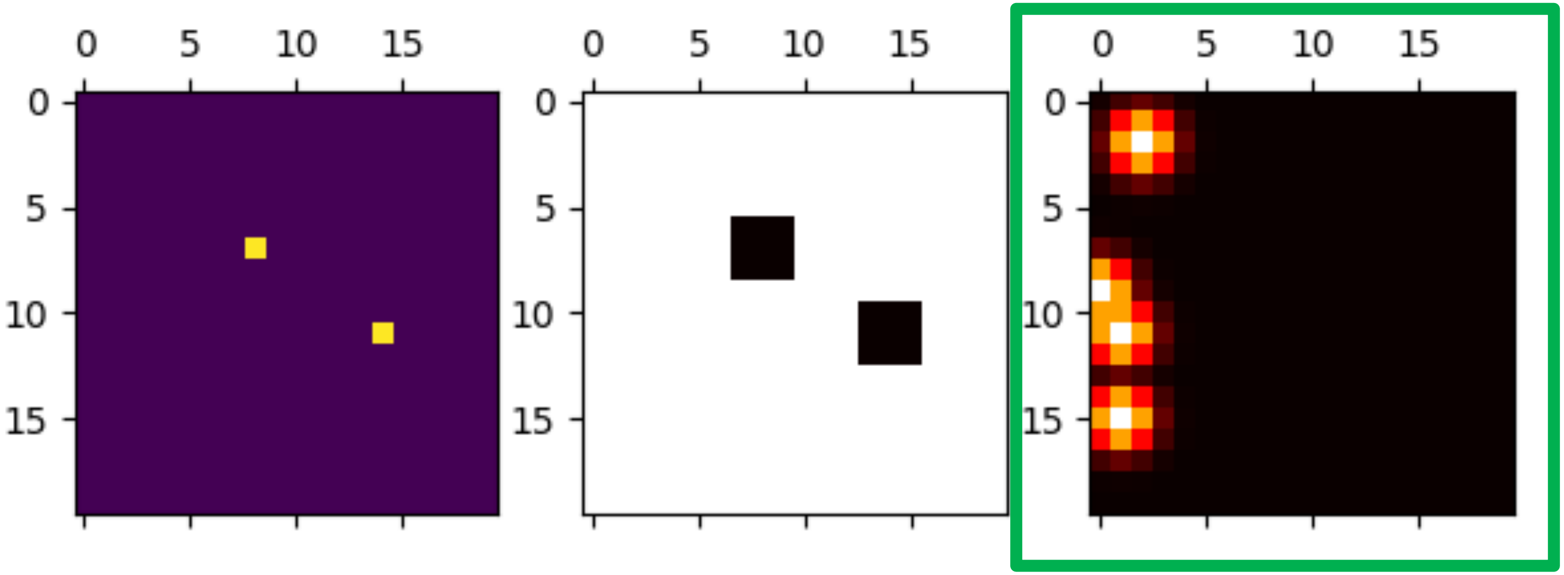
- Stand
- Left
- Right
- Up
- Down

Environment



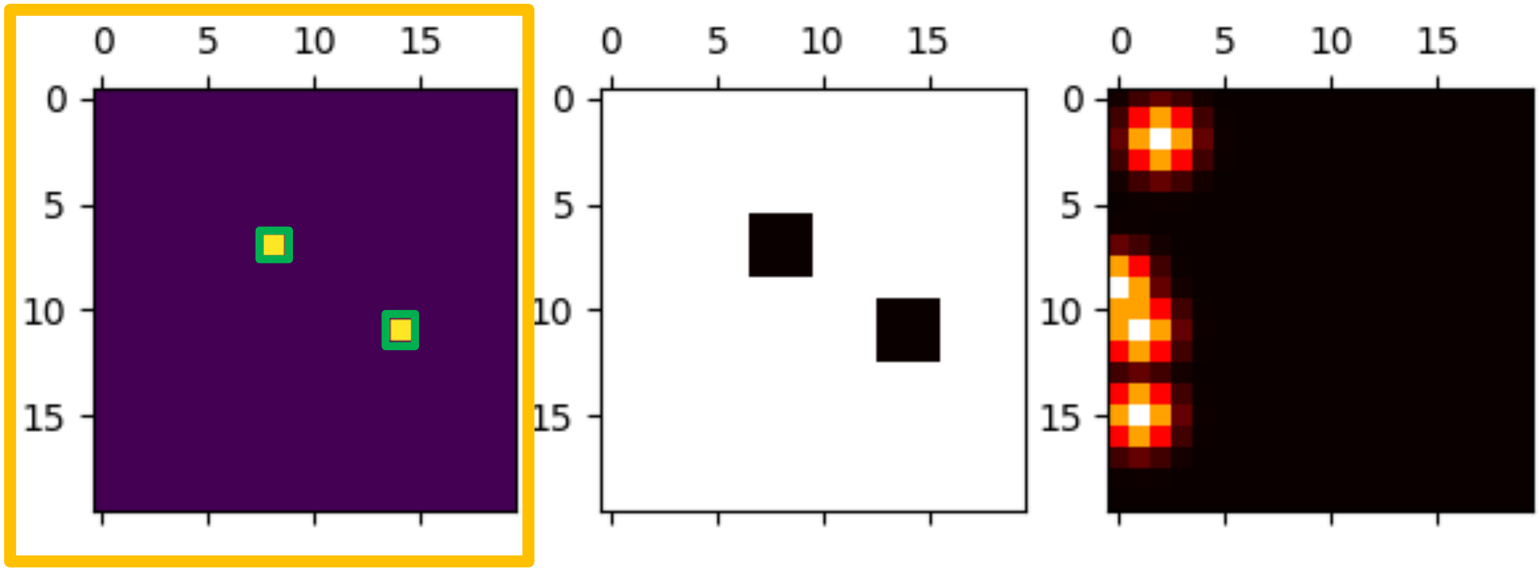
Real map: 4 targets (white pixels)

Environment



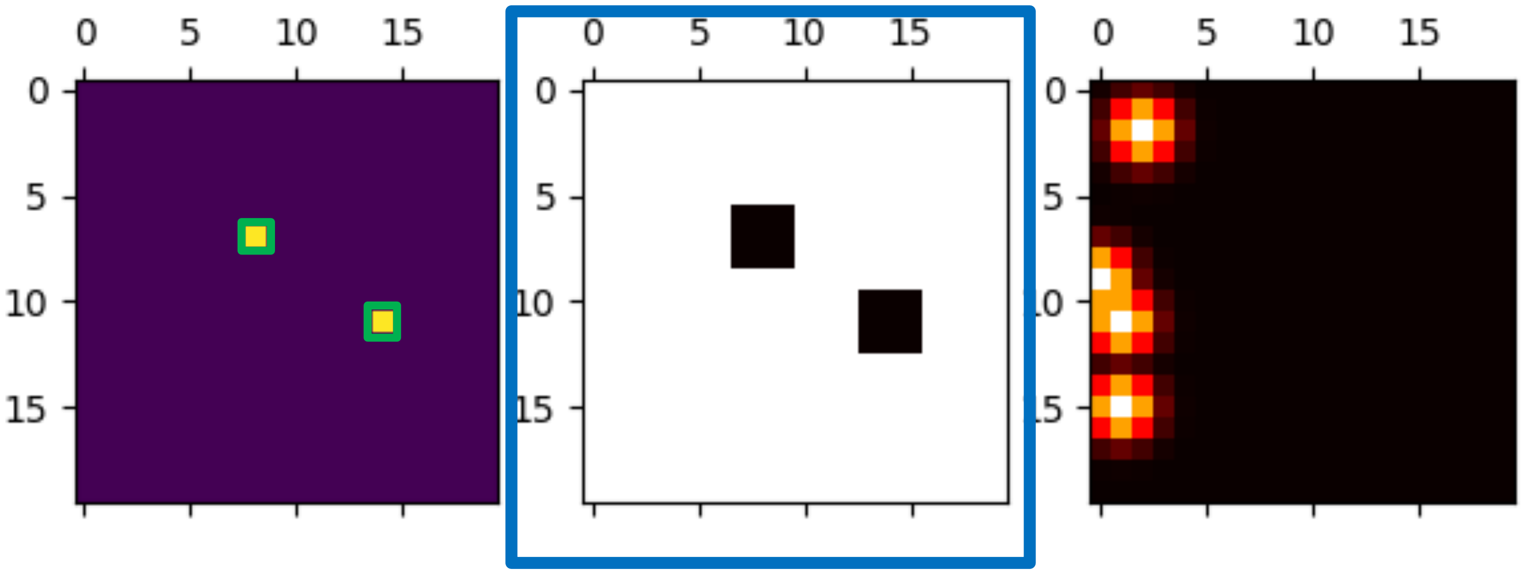
The values of the pixels are between 0 (black pixels) and 1 (white pixels)

Drones current position



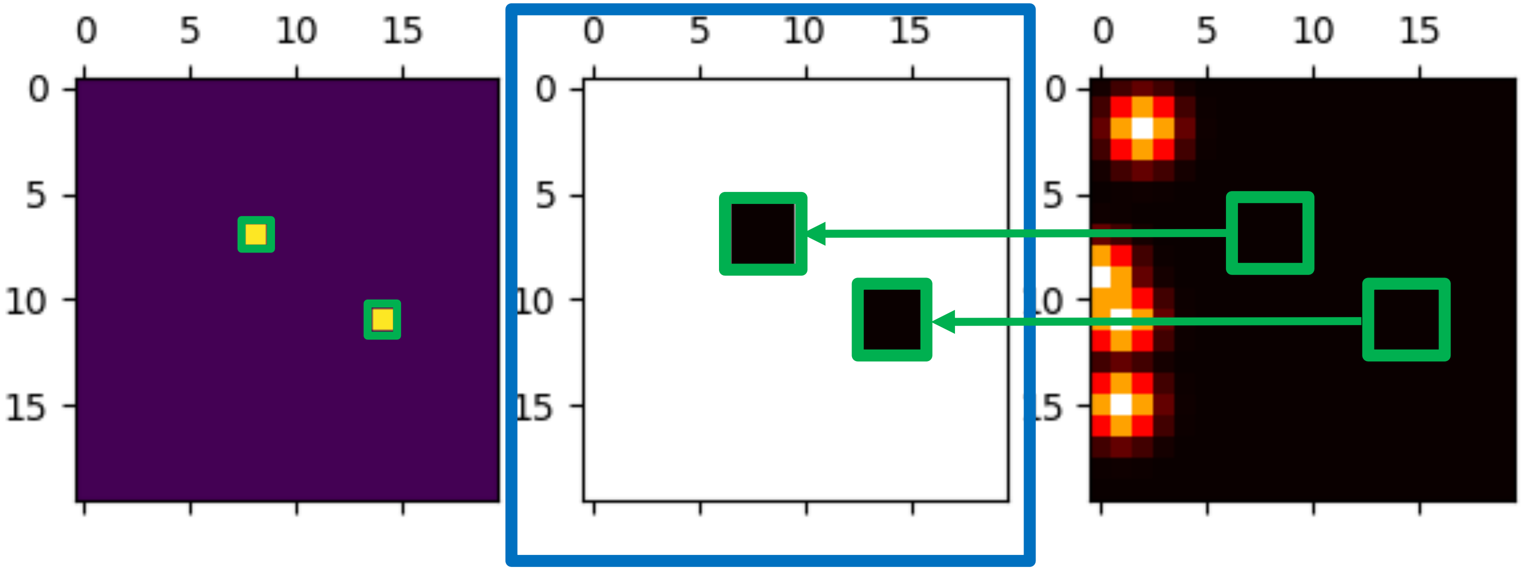
Some drones in the map

Discovered map



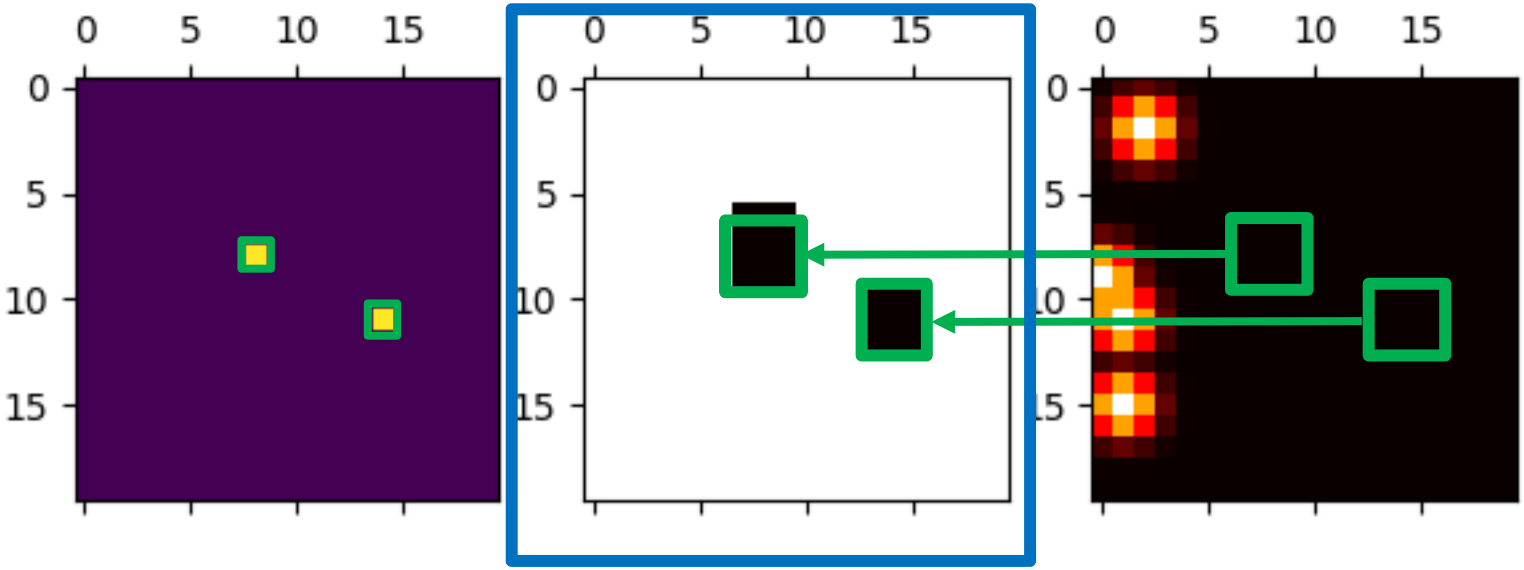
That don't know the real map but share a known map

Discovered map (cont)



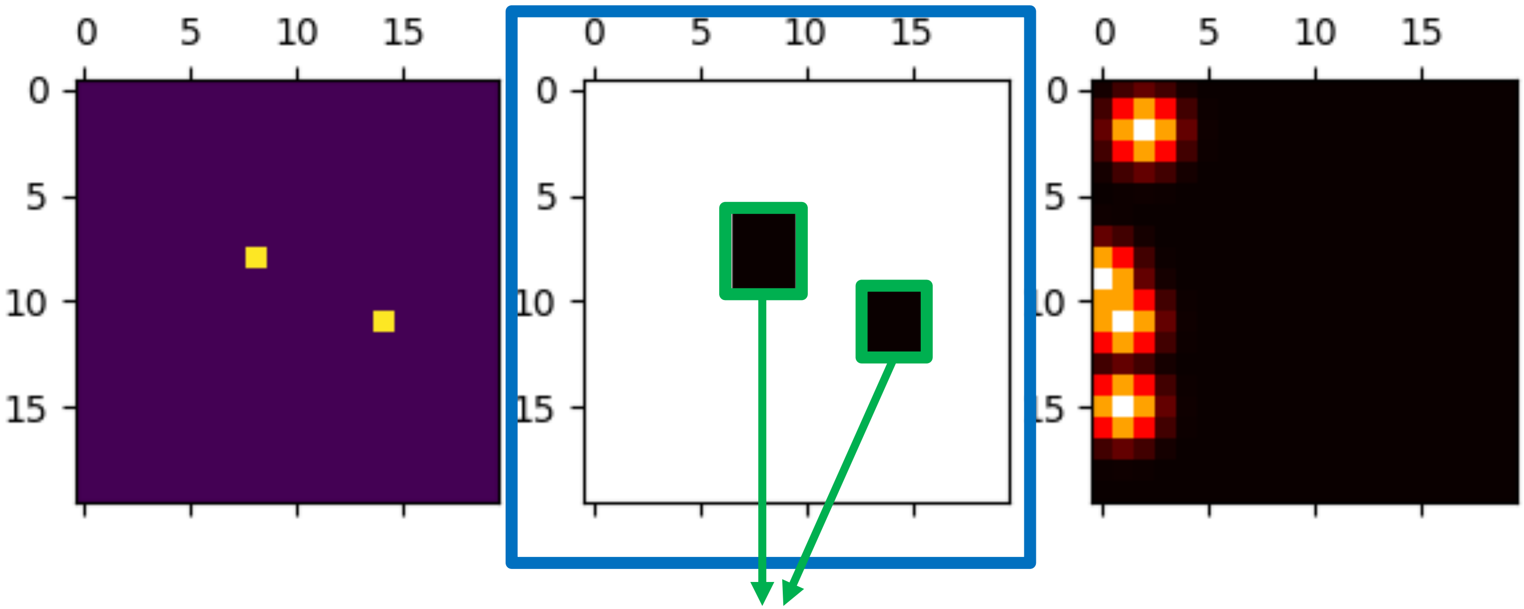
They have a Field of View (FOV) of 3x3 pixels

Discovered map (cont)



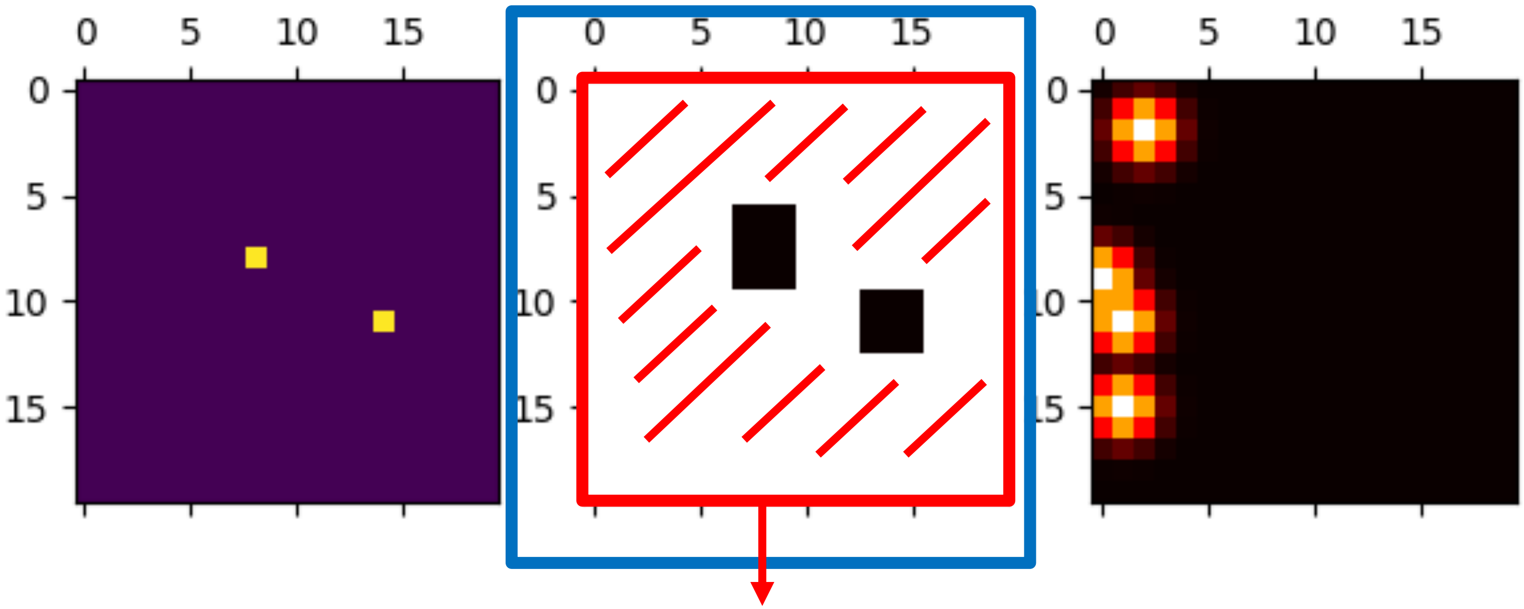
And when they move they discover the real values of the pixels inside the FOV

Discovered map (cont)



So these are the explored cells

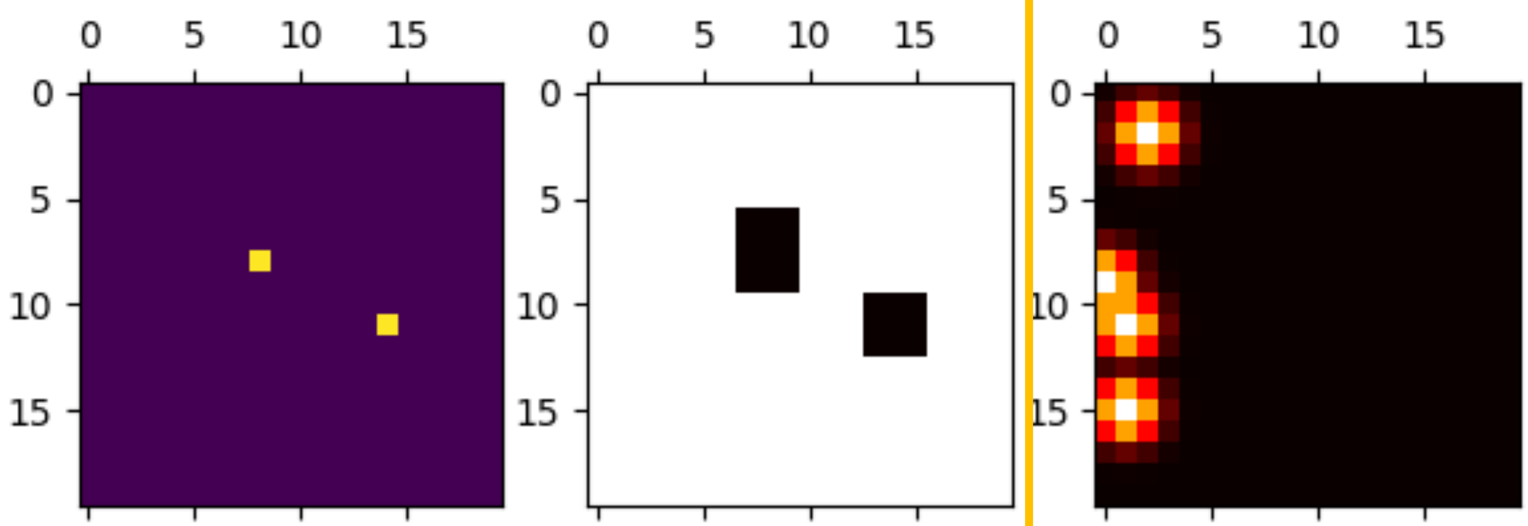
Discovered map (cont)



While these are the unexplored cells

State definition

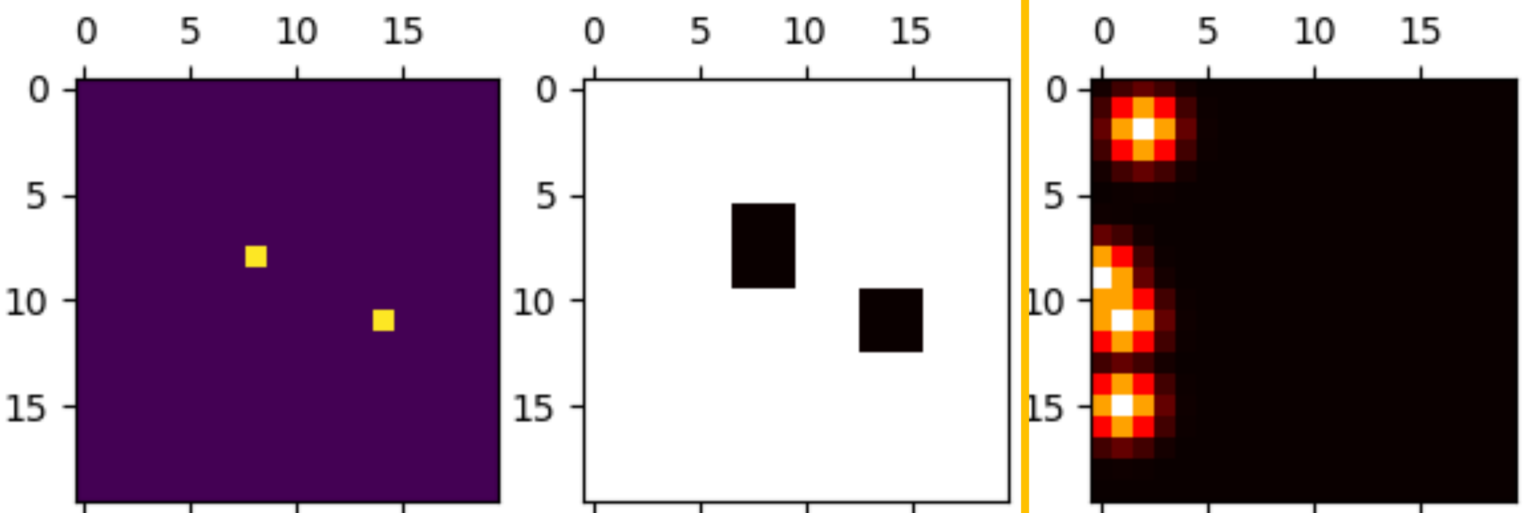
State



The input of the model (the state) is the combination of these 2 matrices: the one containing the position of the drones and the *known* map.

Goal

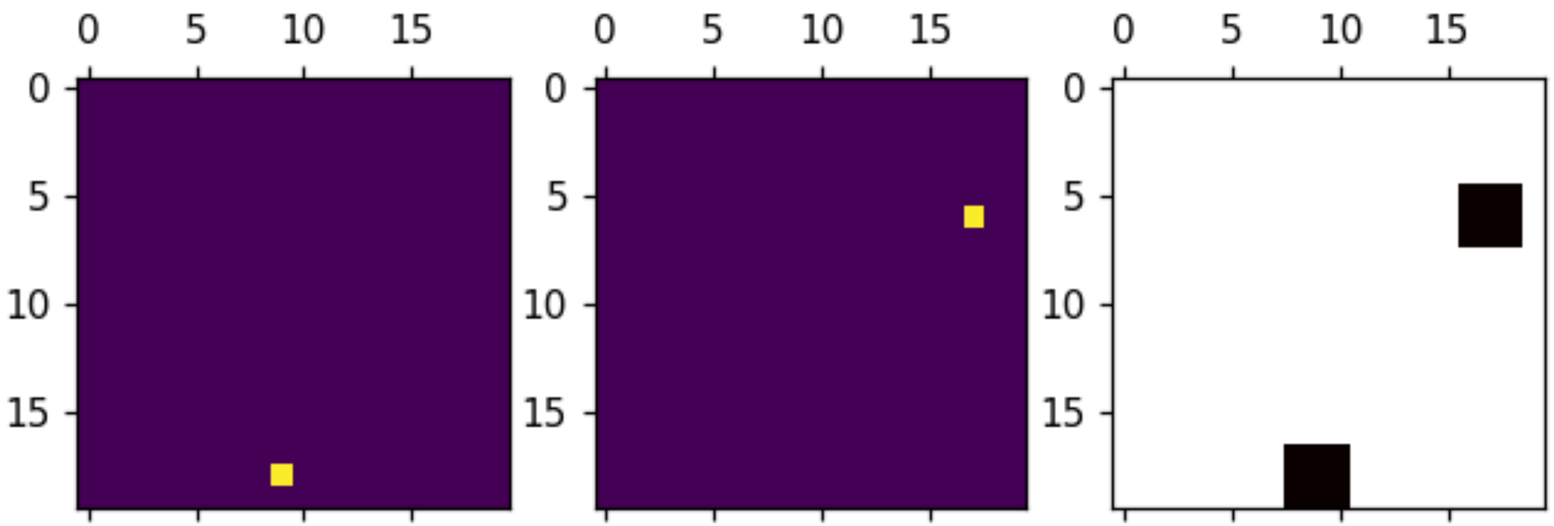
State



Our goal is to discover the map, that is unknown at the beginnning of each episode.

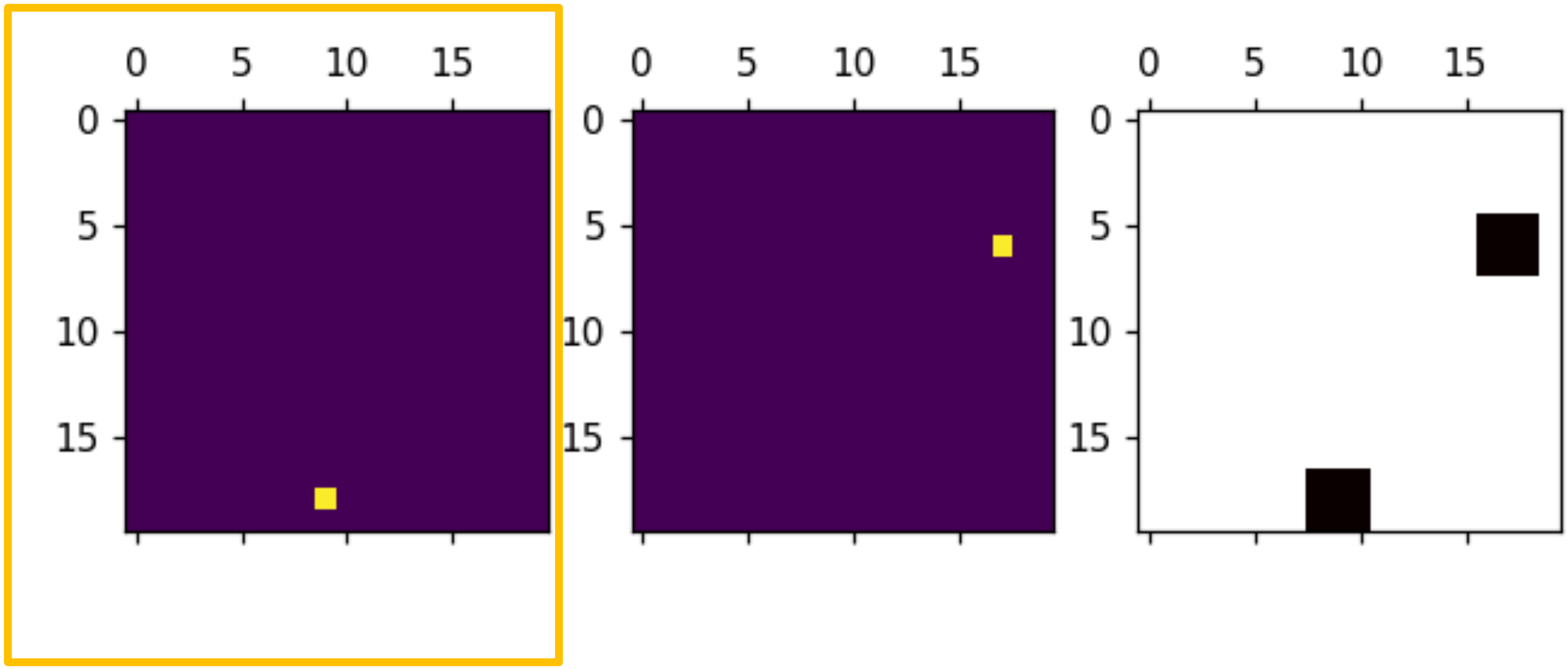
Actual state representation

State



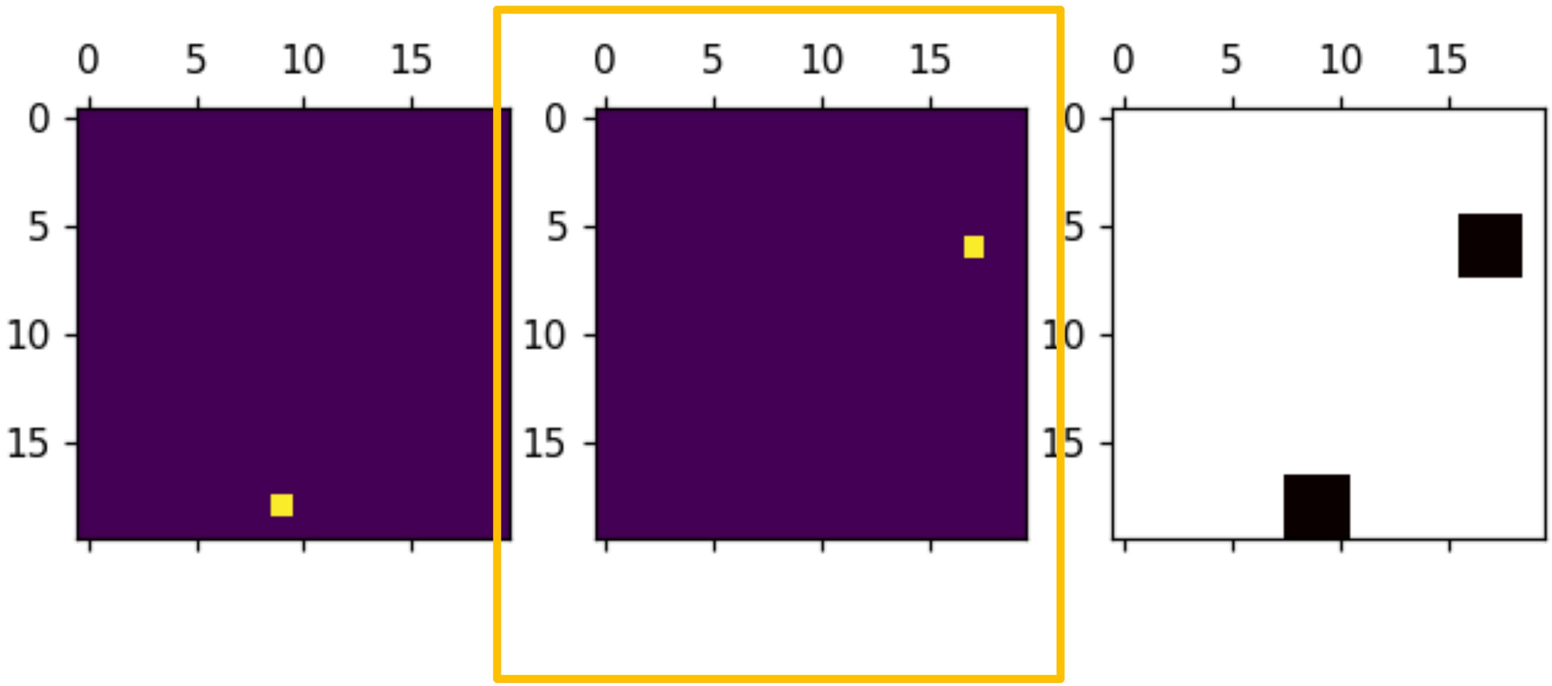
In detail, the state is composed by 3 matrices, 2 for the positions of the drones and one for the *known* map. This is due to the fact that the algorithm is distributed and applied to each drone. Therefore, each of them must be able to distinguish between its own position, and the position of other drones.

Actual state representation



So, the first matrix contains my own position.

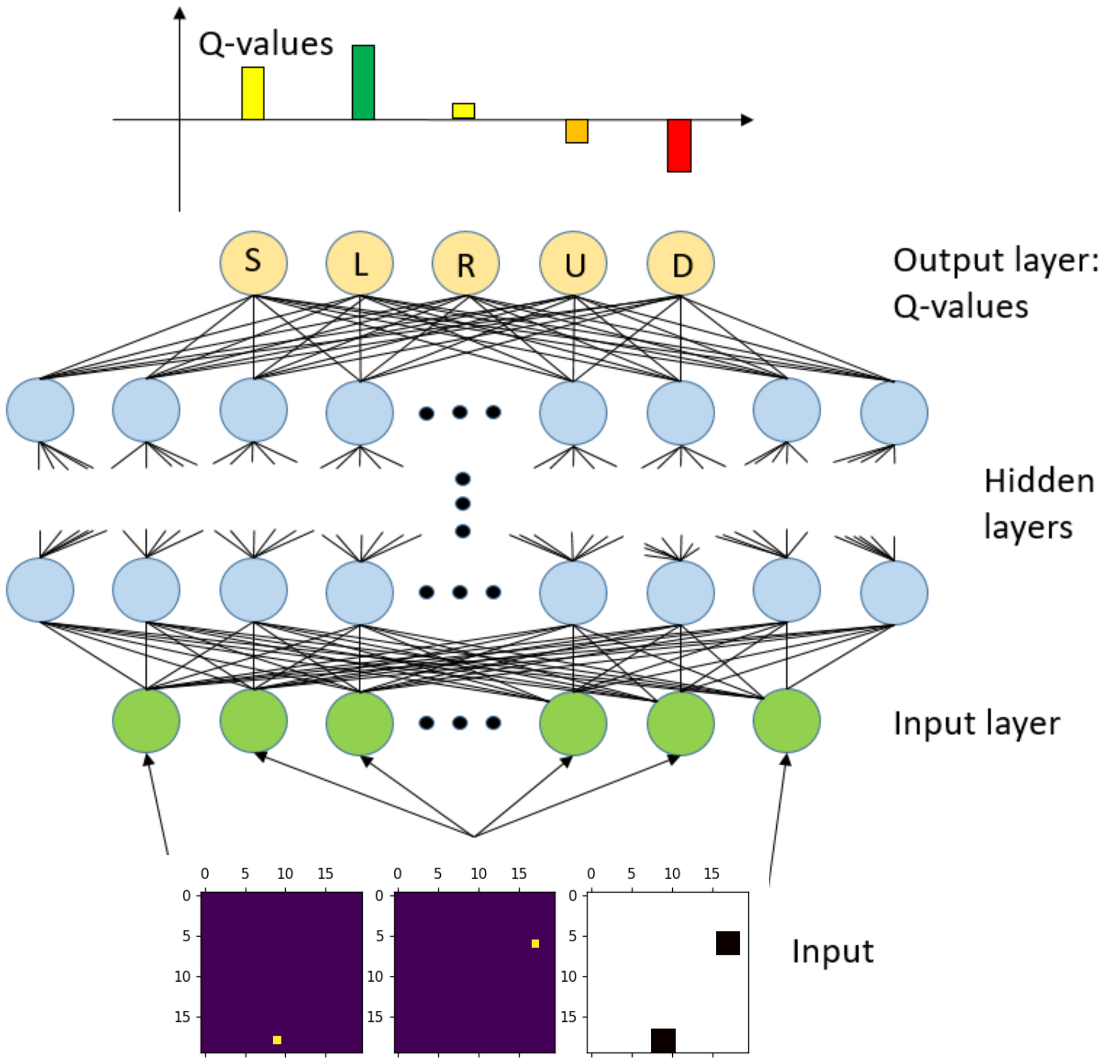
Actual state representation



The second one contains the positions of the other drones. In the case of more than 2 drones, this second matrix will contain more than 1 yellow pixels.

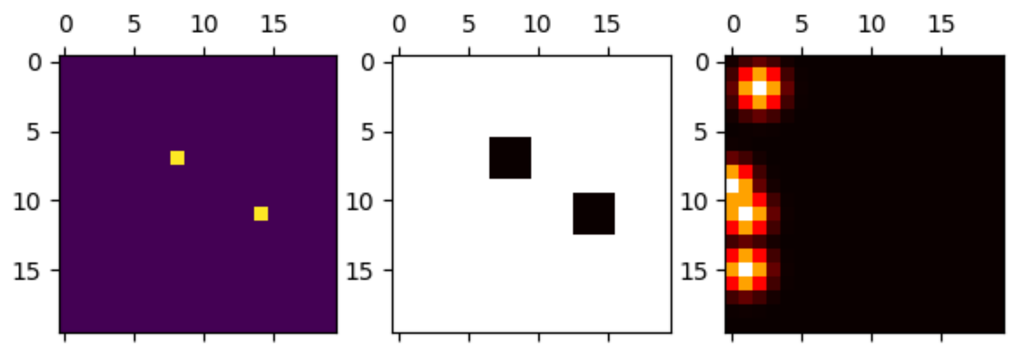
NEURAL NETWORK'S I/O

The state (the 3 matrices of the previous slide) is the input of a neural network, the agent, that has to learn which is the best action to take, given the own position of the drone, the position of the other drones, and the map discovered at the actual time step. In output there are as many outputs as the number of possible actions. These outputs represent the Q-values, the long-term reward associated to each possible action, given that state in input.

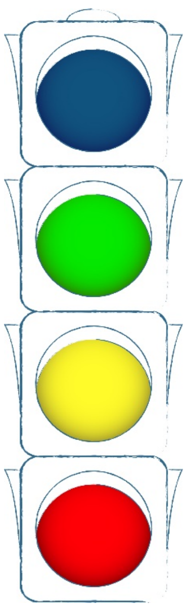


Rewards

At each time step, 1 drone takes an action (the drones move aynchronously) and receives a reward based on that action



Rewards:



0

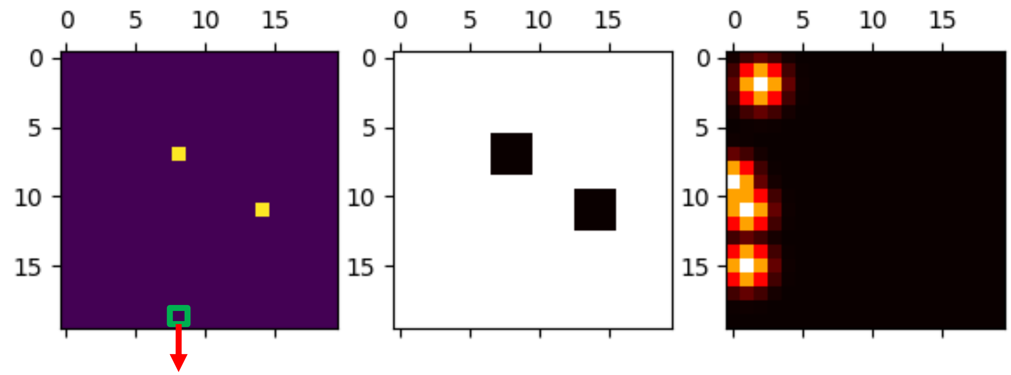
+1

-0.4

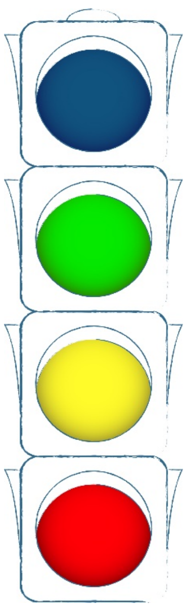
-1

Rewards

At each time step, 1 drone takes an action (the drones move aynchronously) and receives a reward based on that action



Rewards:



0

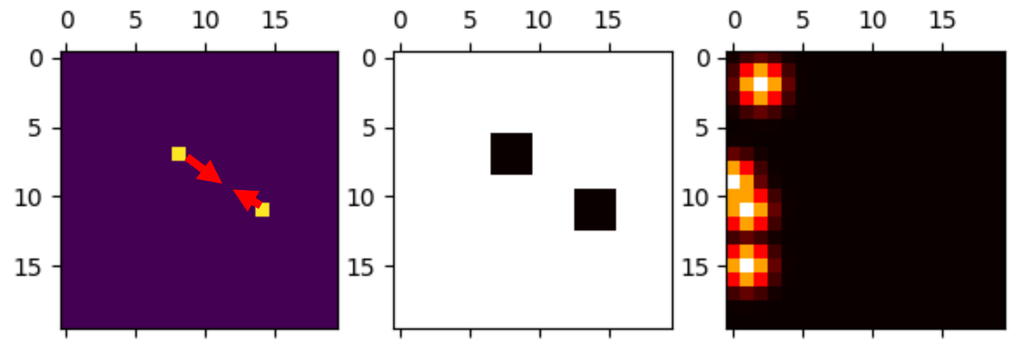
+ 1

- 0.4

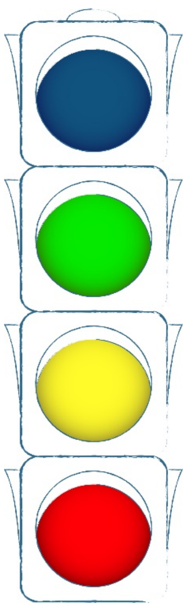
-1

When the drone is on the border of the map and tries to exit from the map

At each time step, 1 drone takes an action (the drones move aynchronously) and receives a reward based on that action



Rewards:



0

+ 1

- 0.4

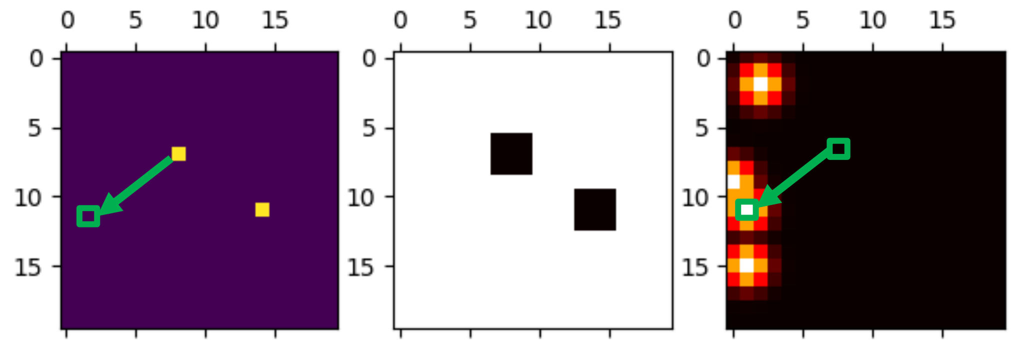
-1

When we have a collision between the drones: they are in the same position

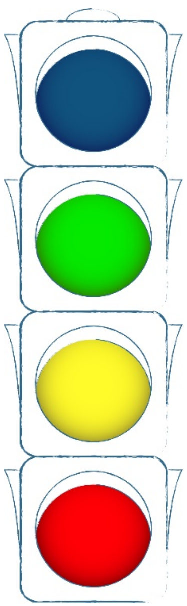
When the drone is on the border of the map and tries to exit from the map

Rewards

At each time step, 1 drone takes an action (the drones move aynchronously) and receives a reward based on that action



Rewards:



0

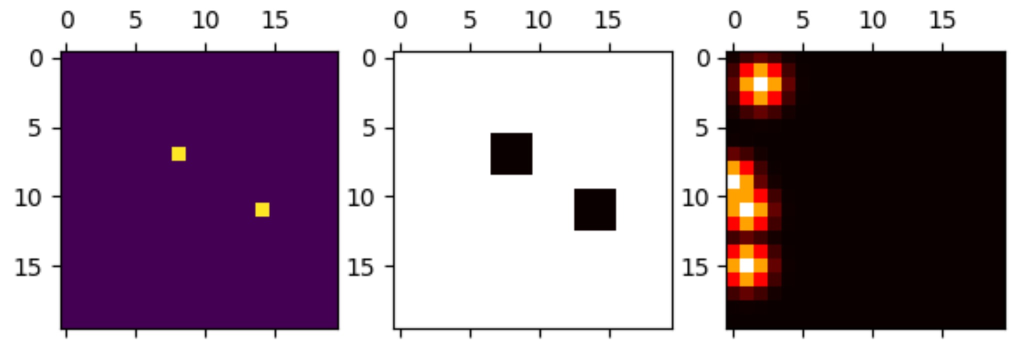
+1 When the drone reaches one of the targets (white pixels)

-0.4 When we have a collision between the drones: they are in the same position

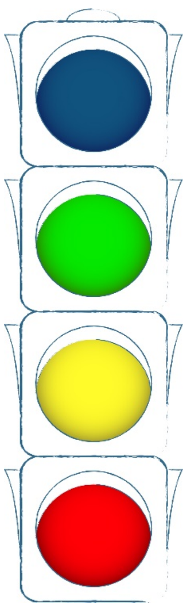
-1 When the drone is on the border of the map and tries to exit from the map

Rewards

At each time step, 1 drone takes an action (the drones move aynchronously) and receives a reward based on that action



Rewards:



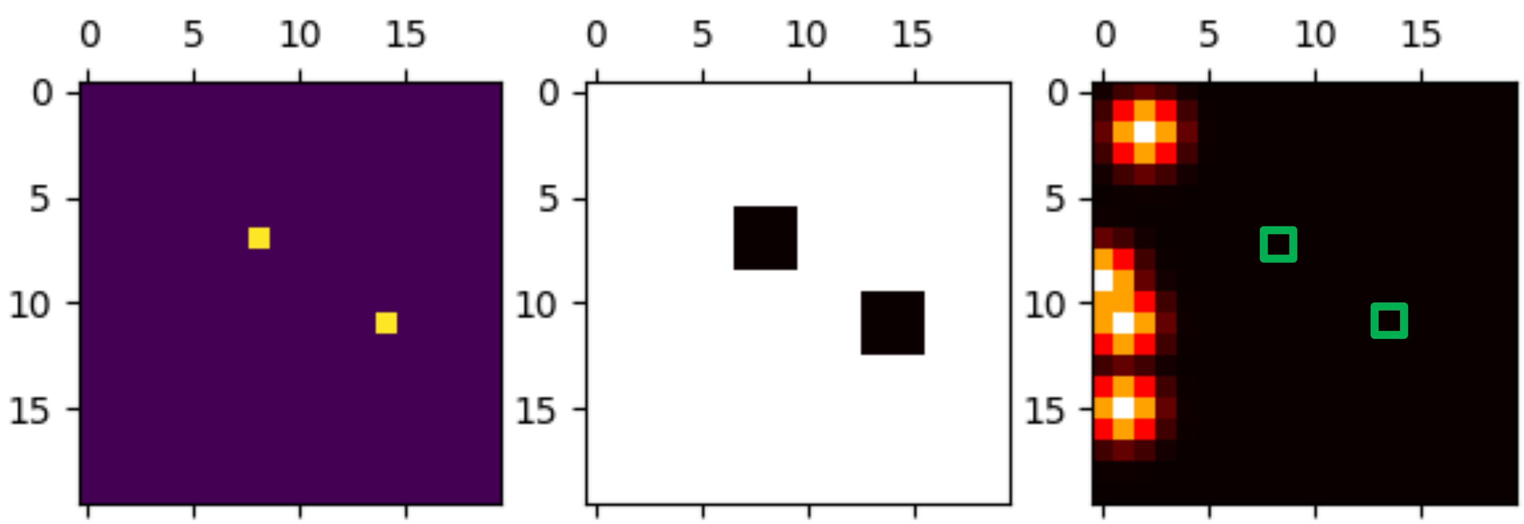
0 In all the other cases

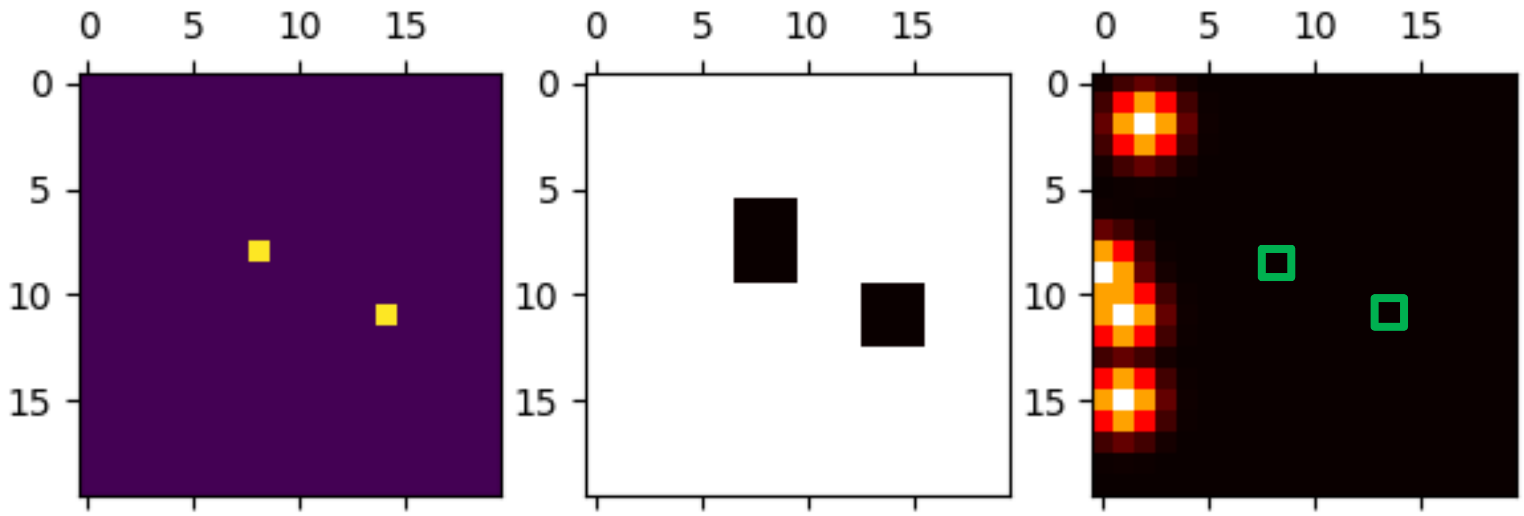
+1 When the drone reaches one of the targets (white pixels)

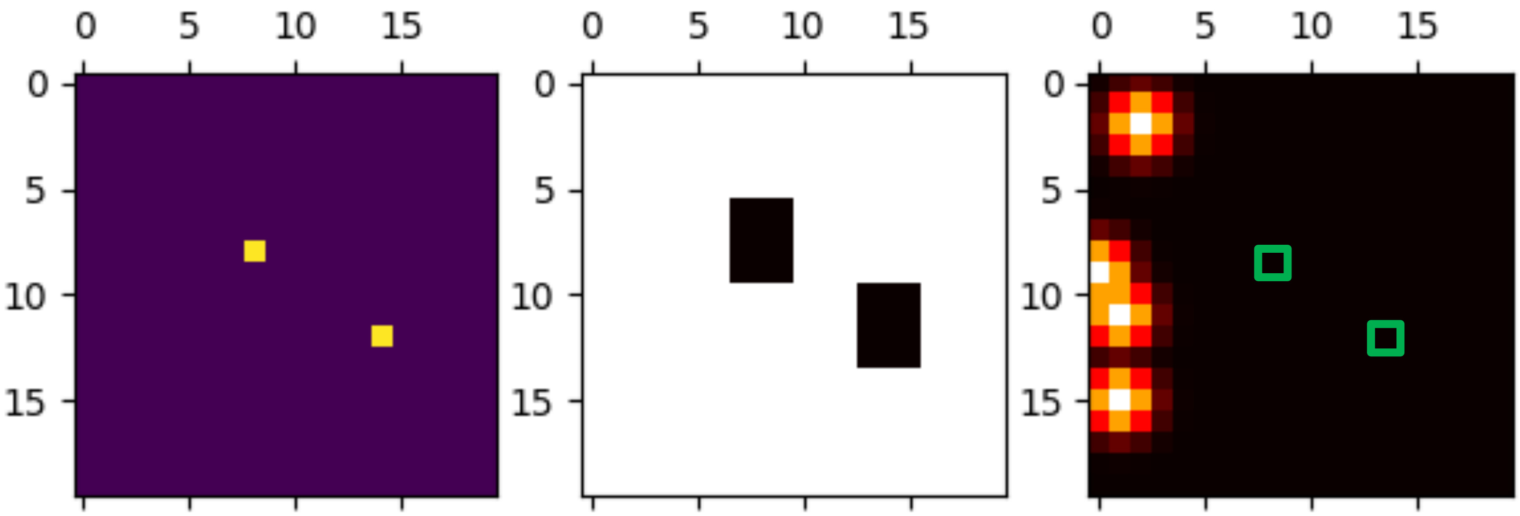
-0.4 When we have a collision between the drones: they are in the same position

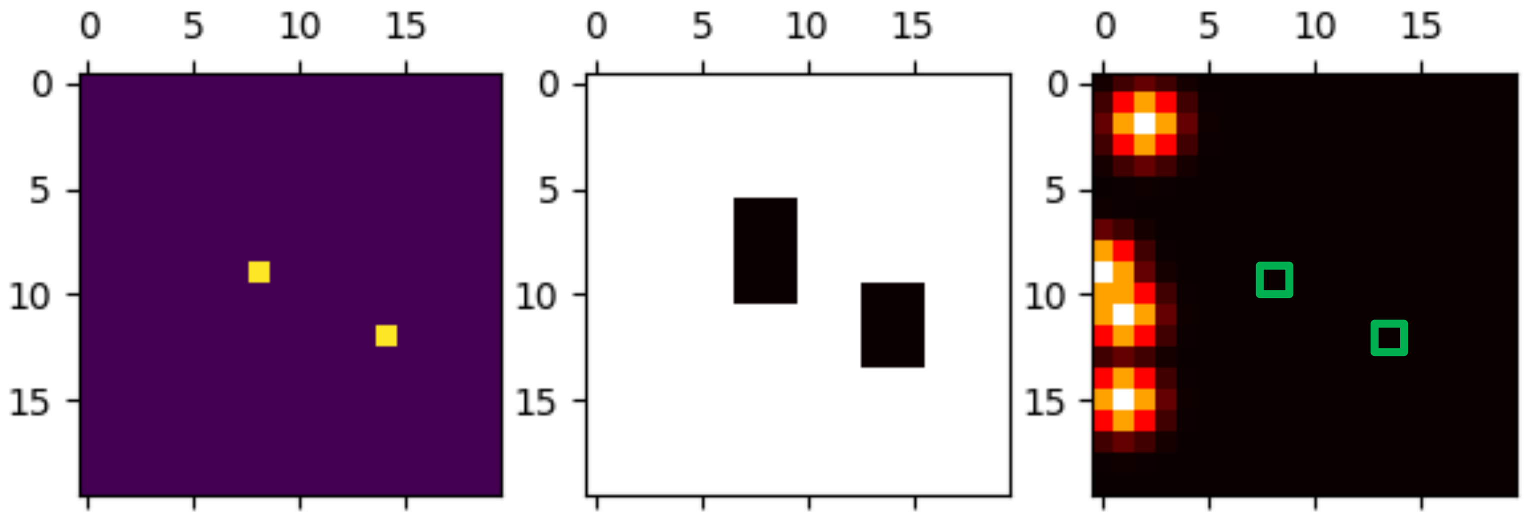
-1 When the drone is on the border of the map and tries to exit from the map

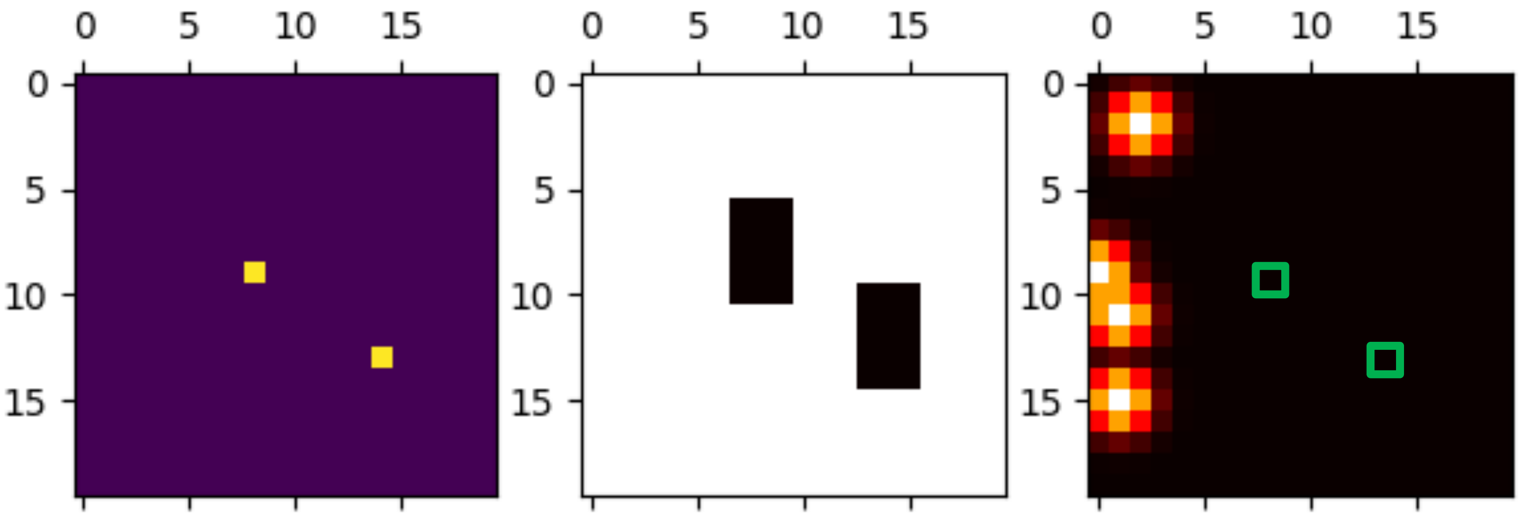
SIMULATION



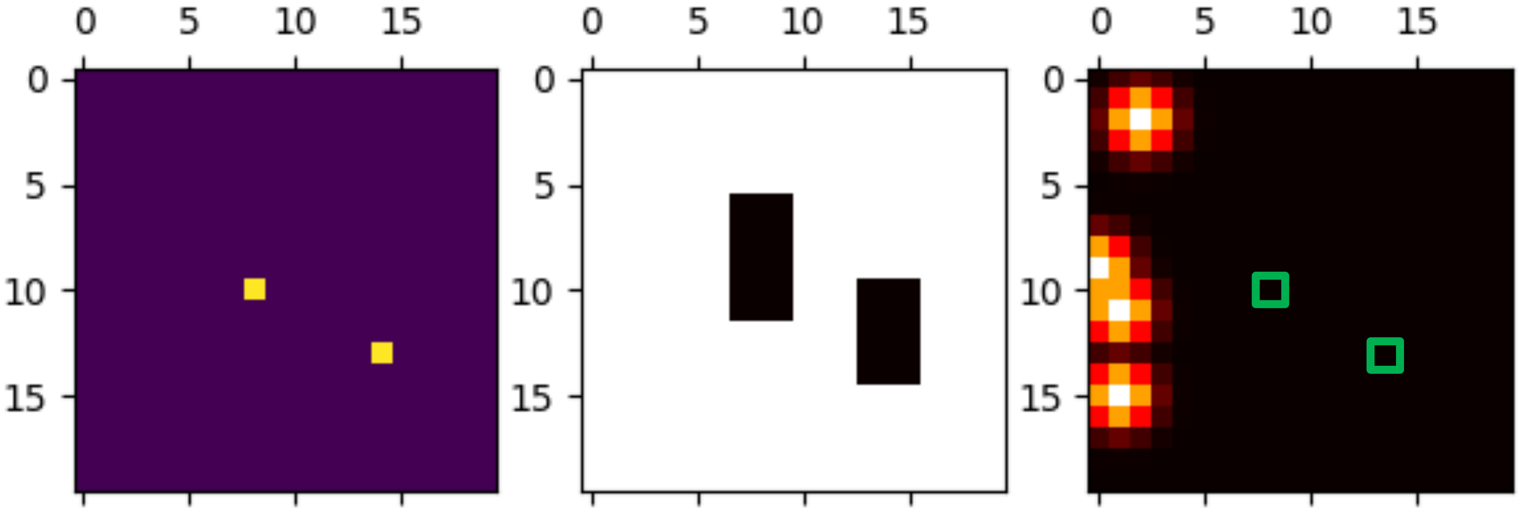

0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15


0
5
10
15
0
5
10
15

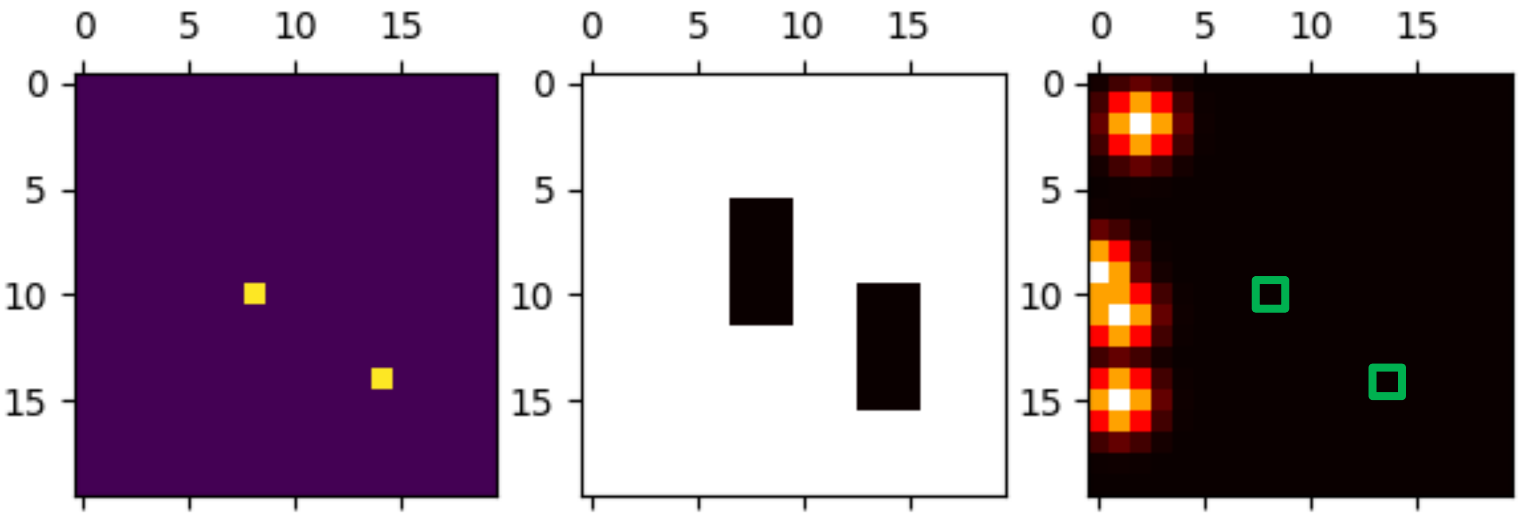

0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15



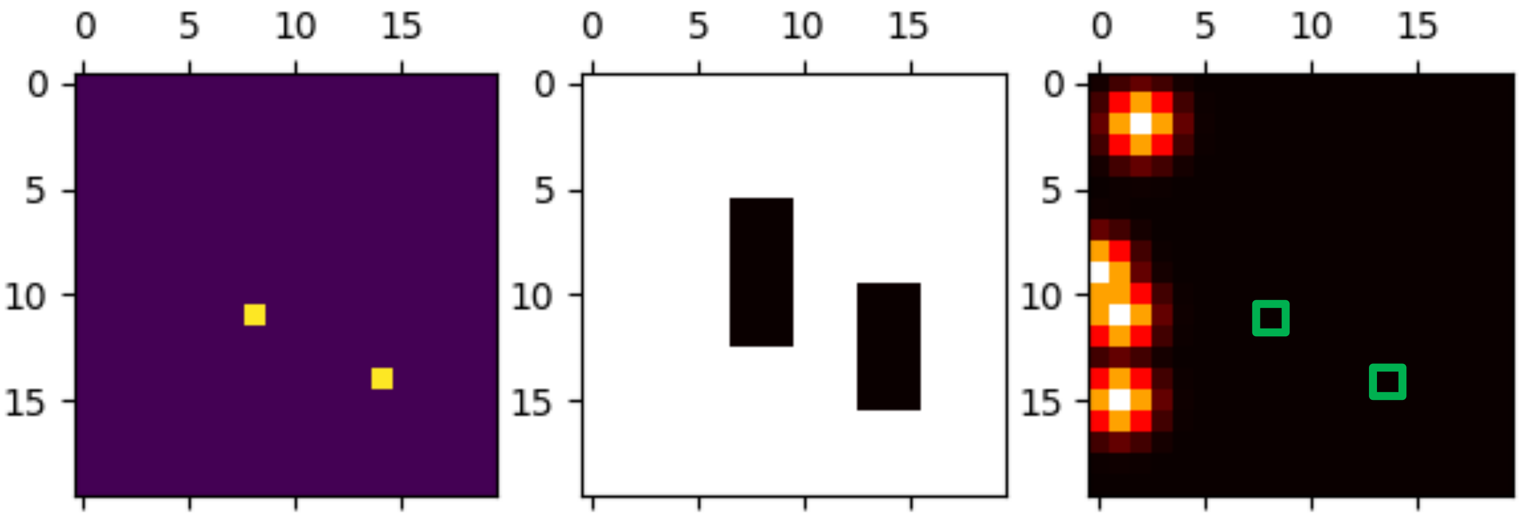
0
5
10
15



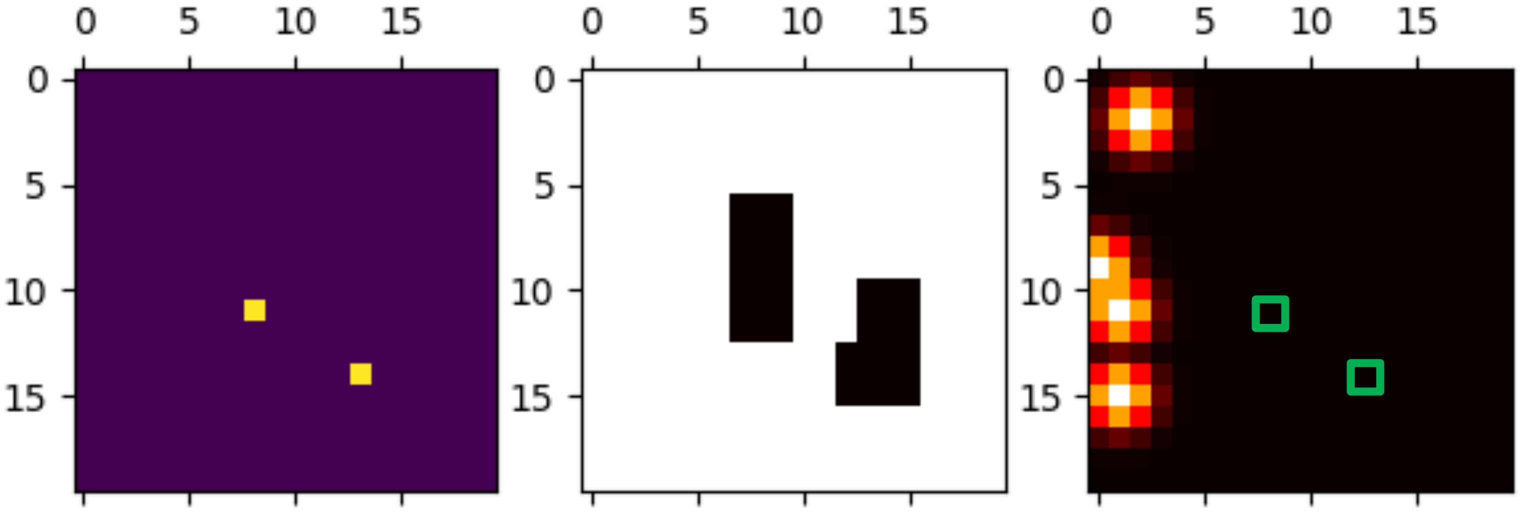
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15



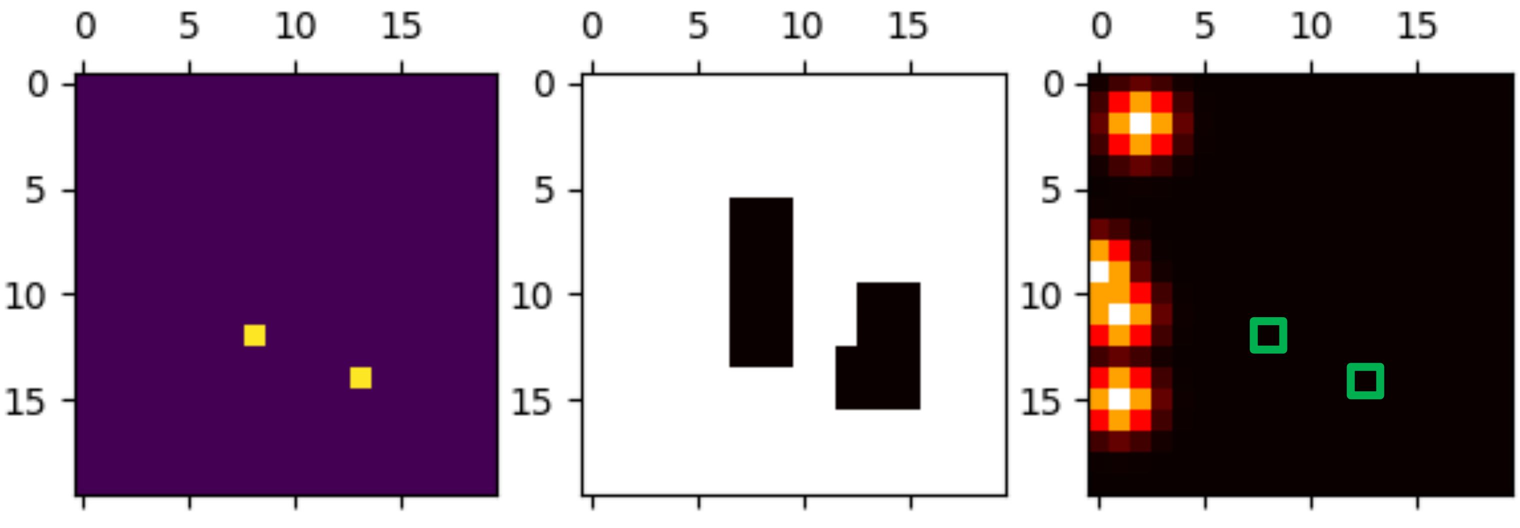
0
5
10
15



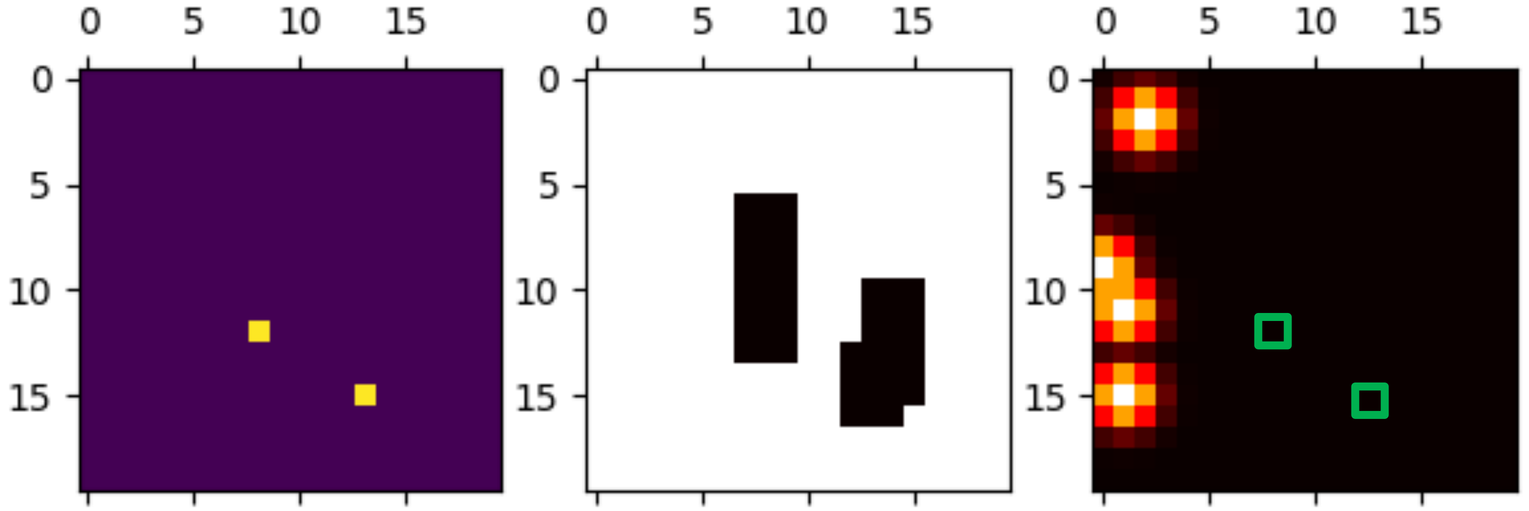
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15



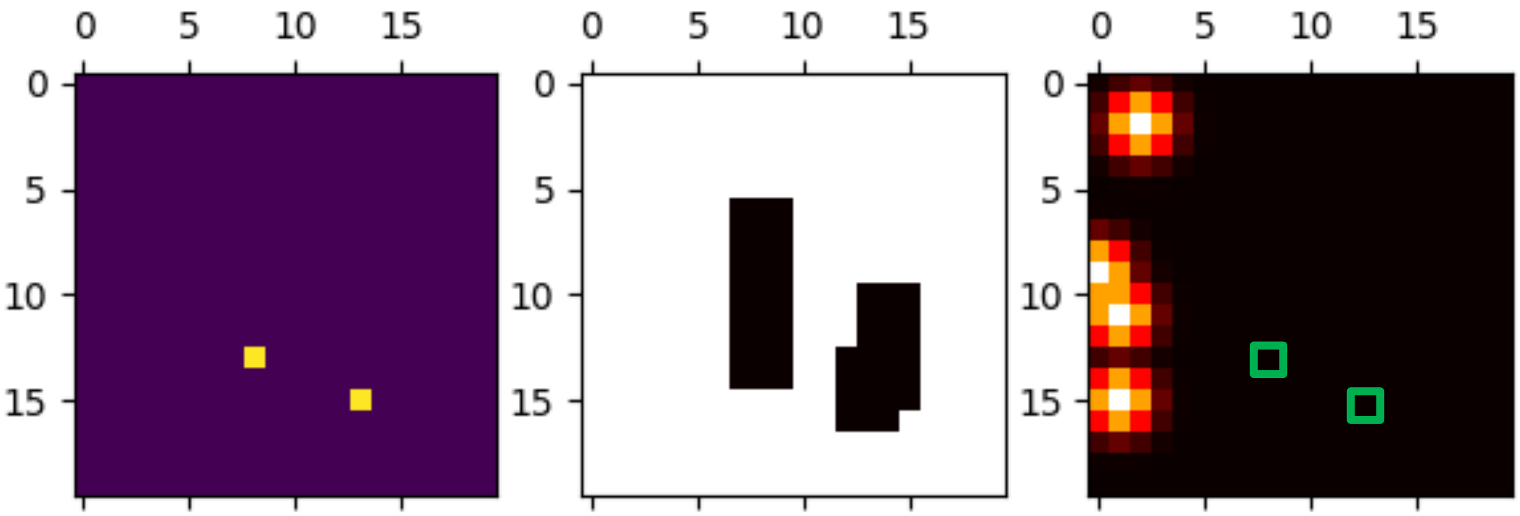
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15



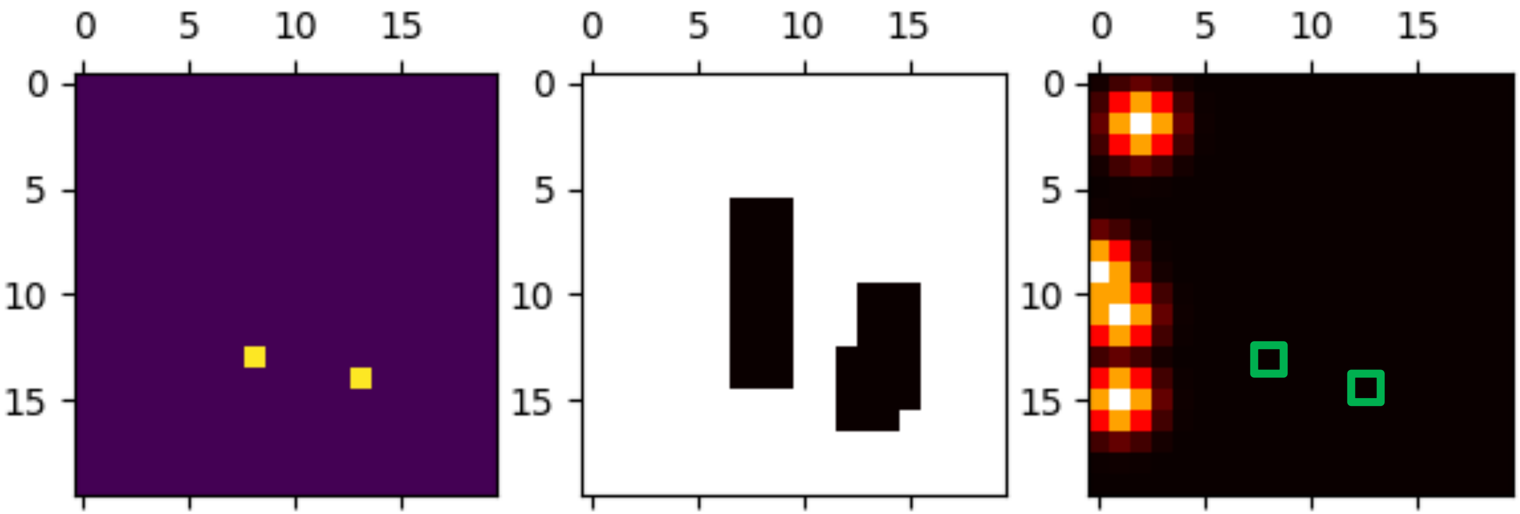
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15

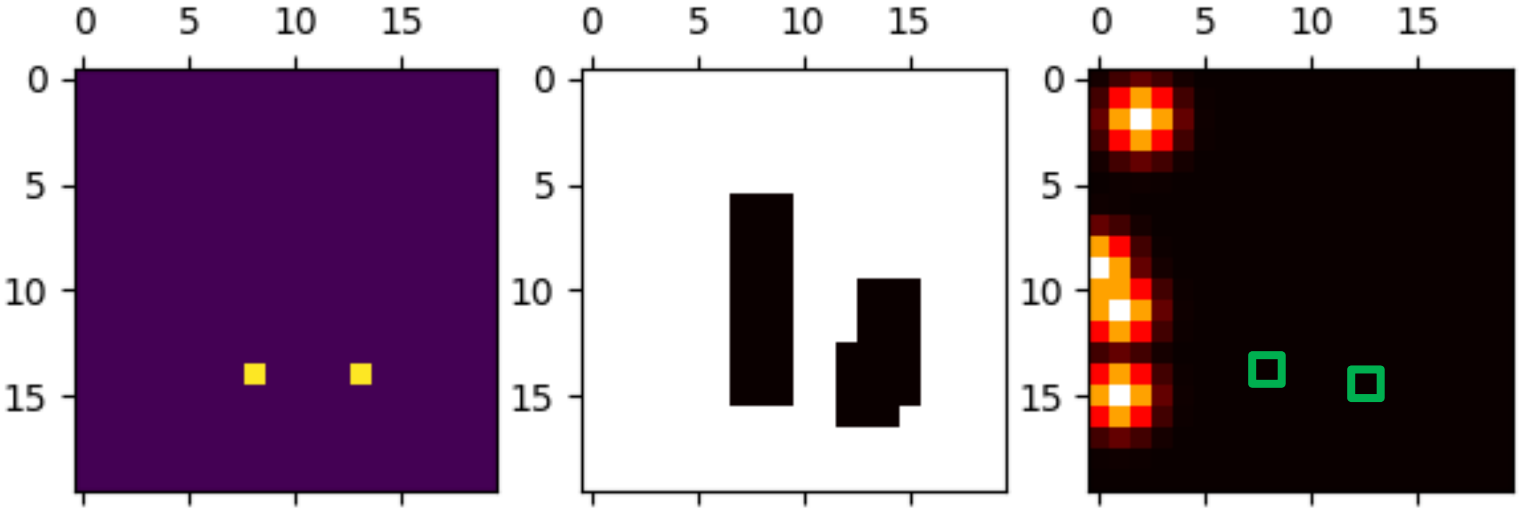


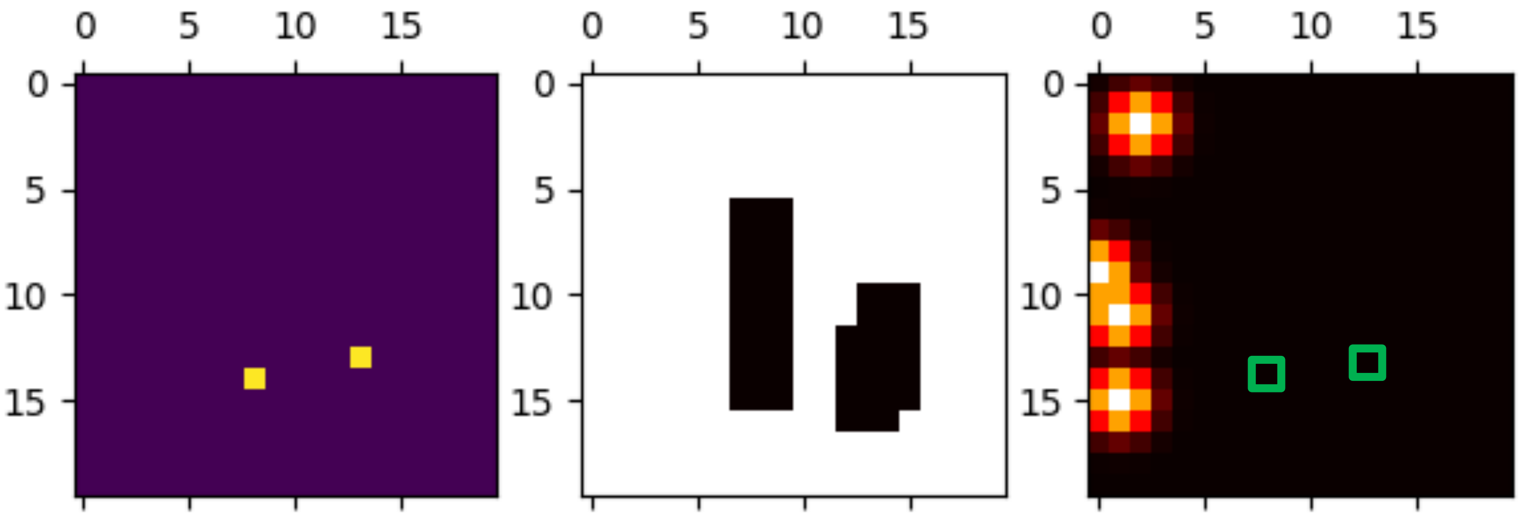
0
5
10
15



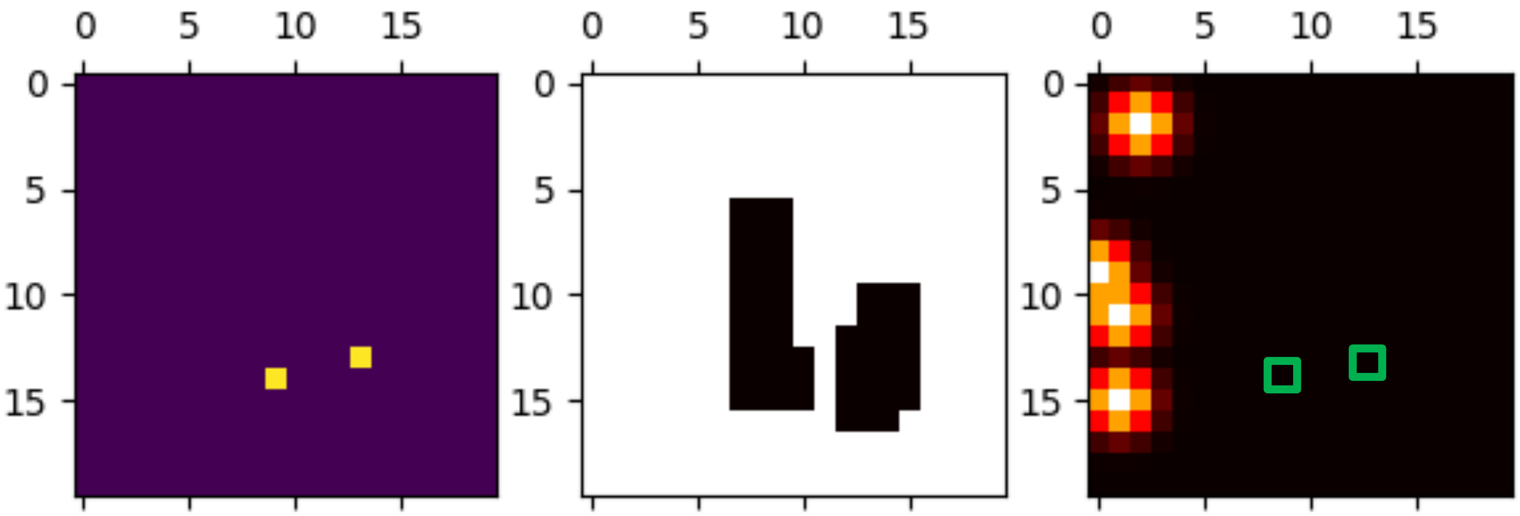
0
5
10
15
0
5
10
15



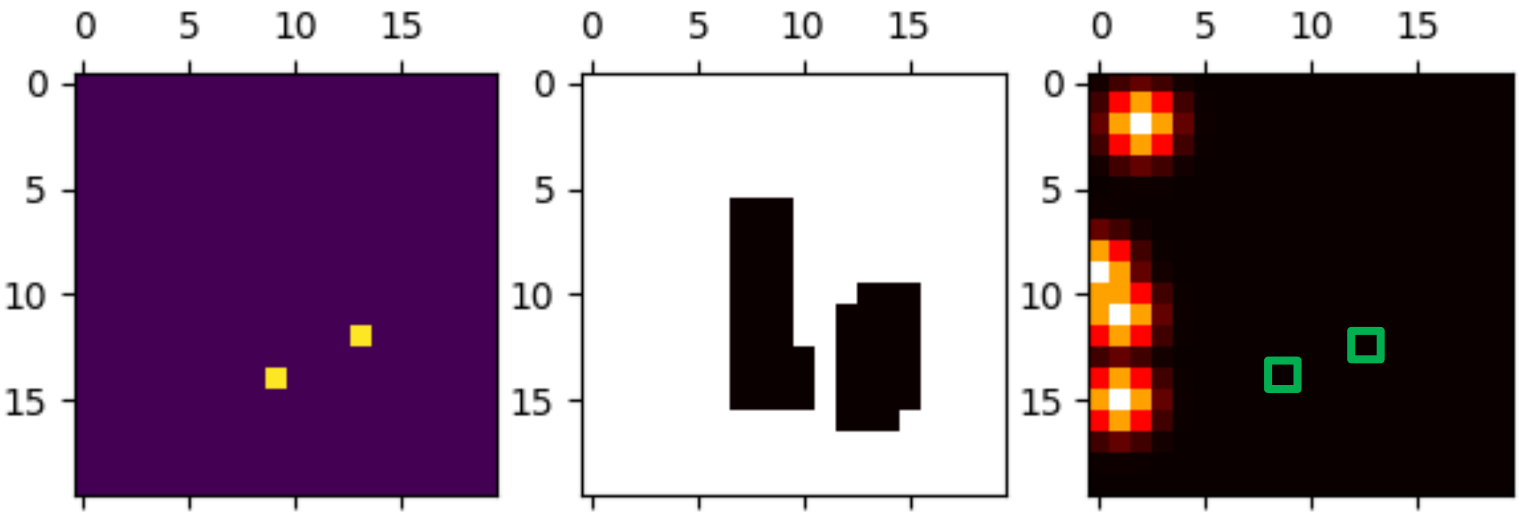

0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15



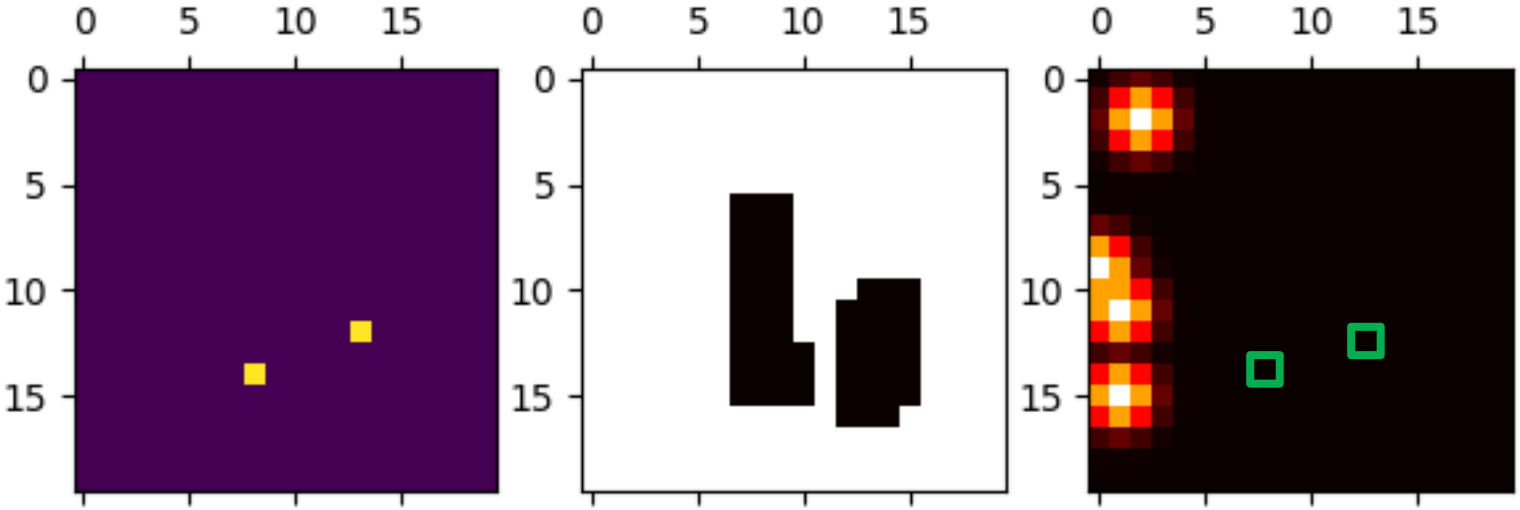
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15



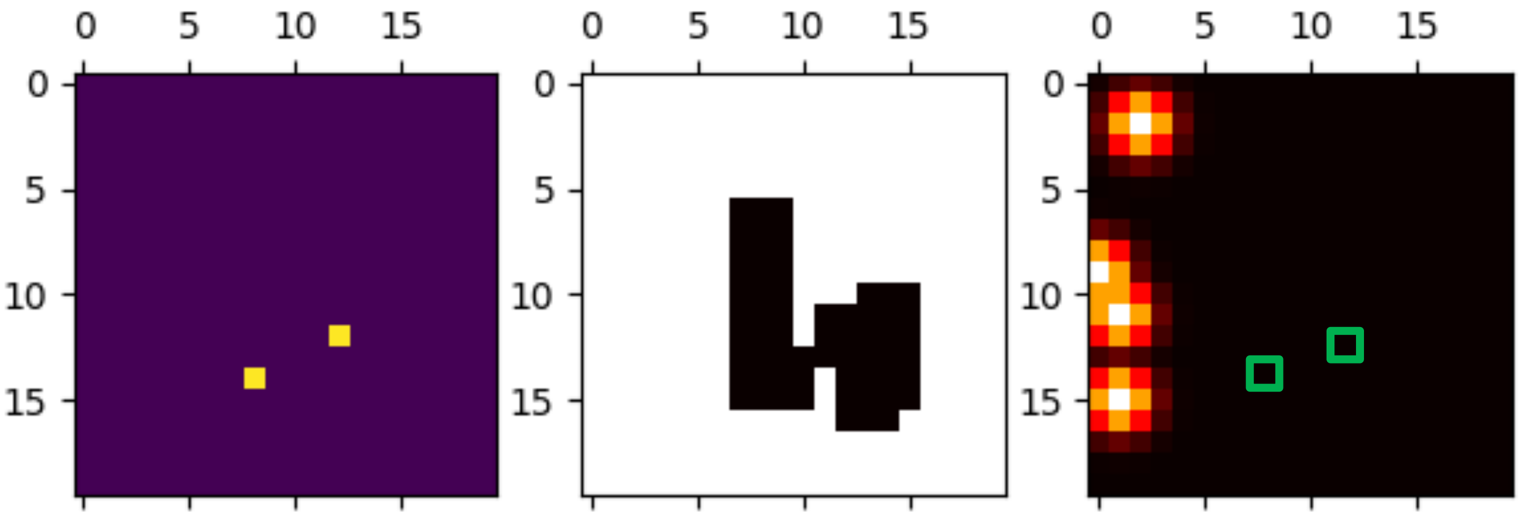
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15

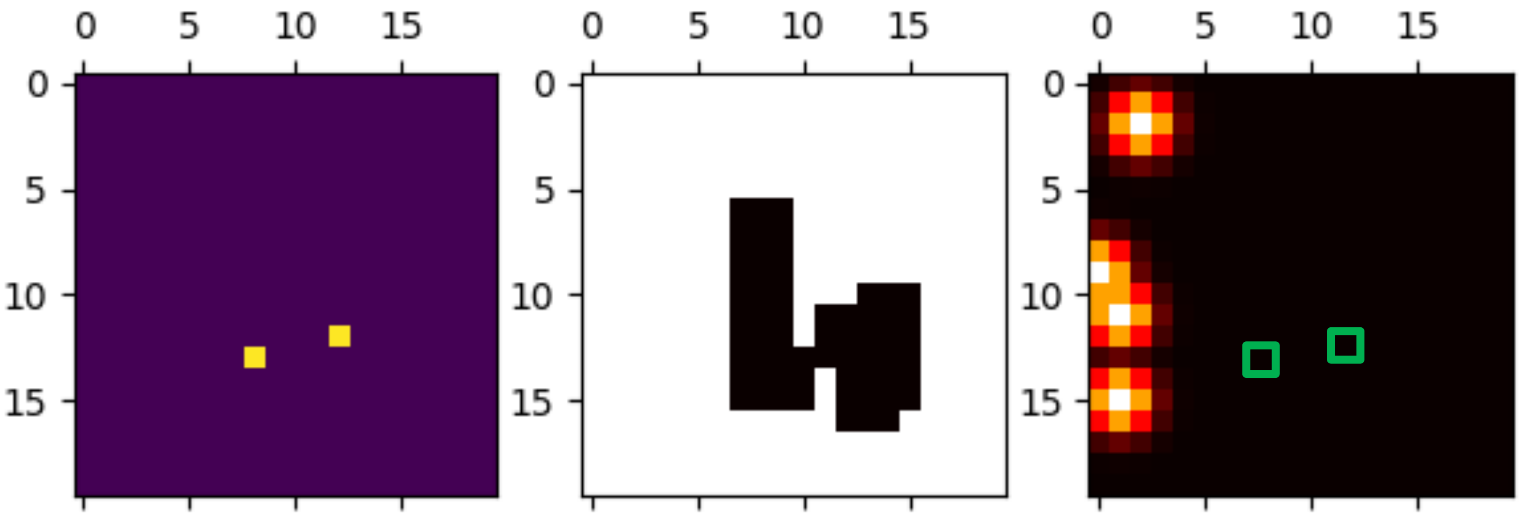


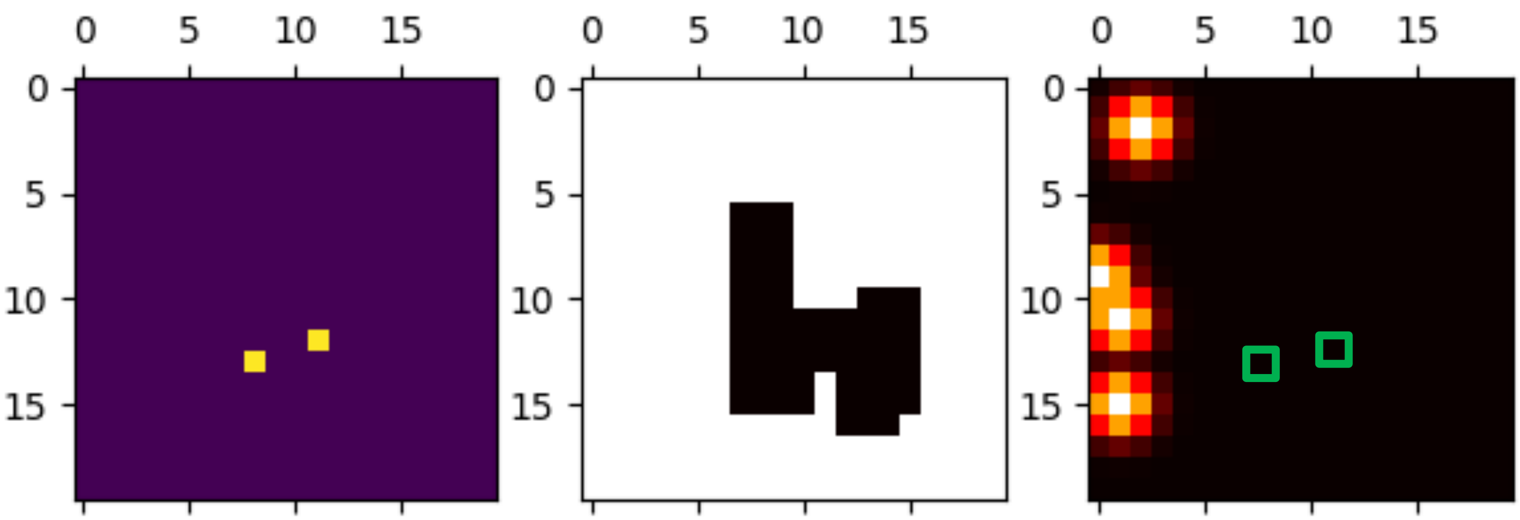
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15



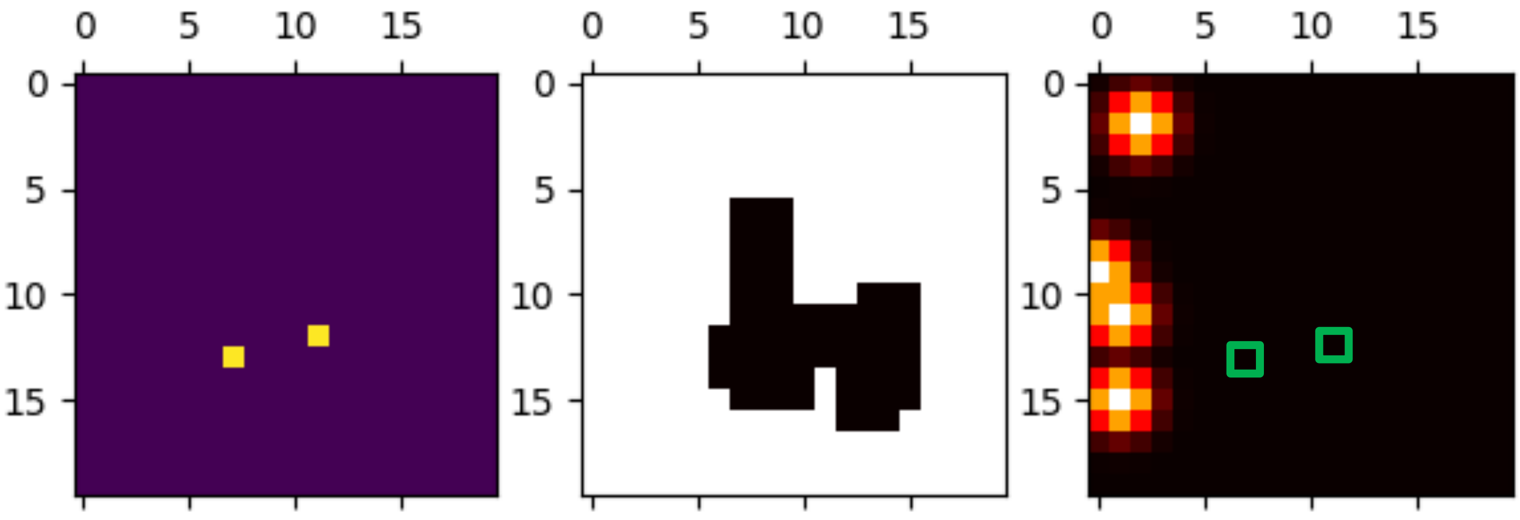
0
5
10
15
0
5
10
15


0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15

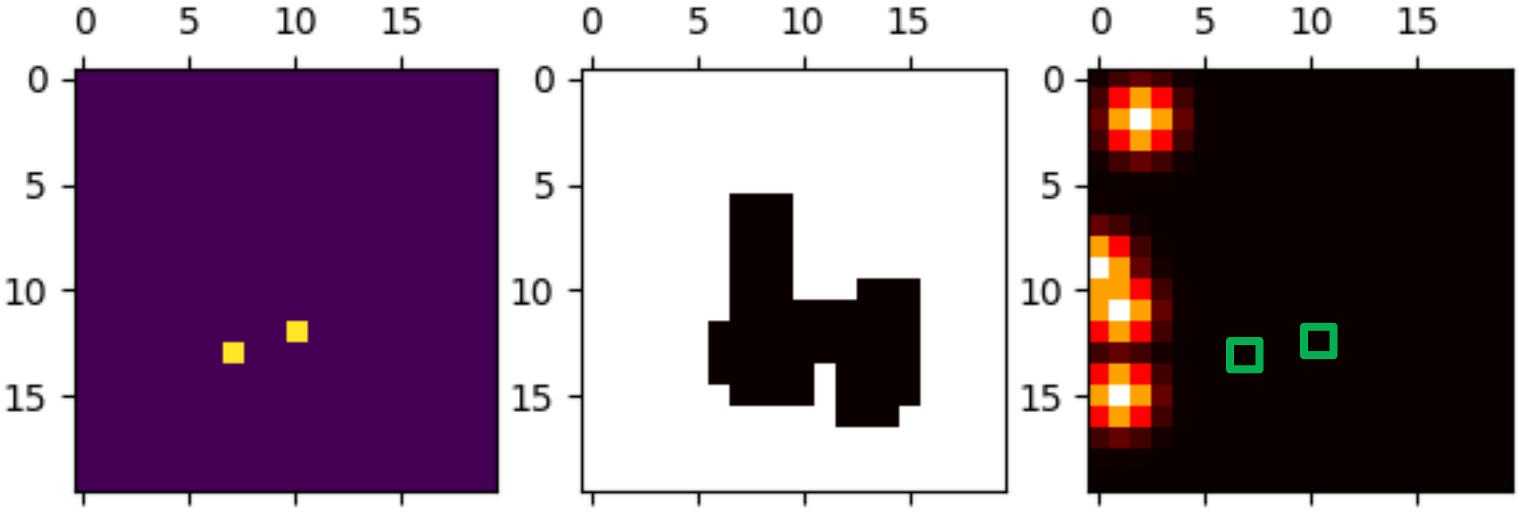




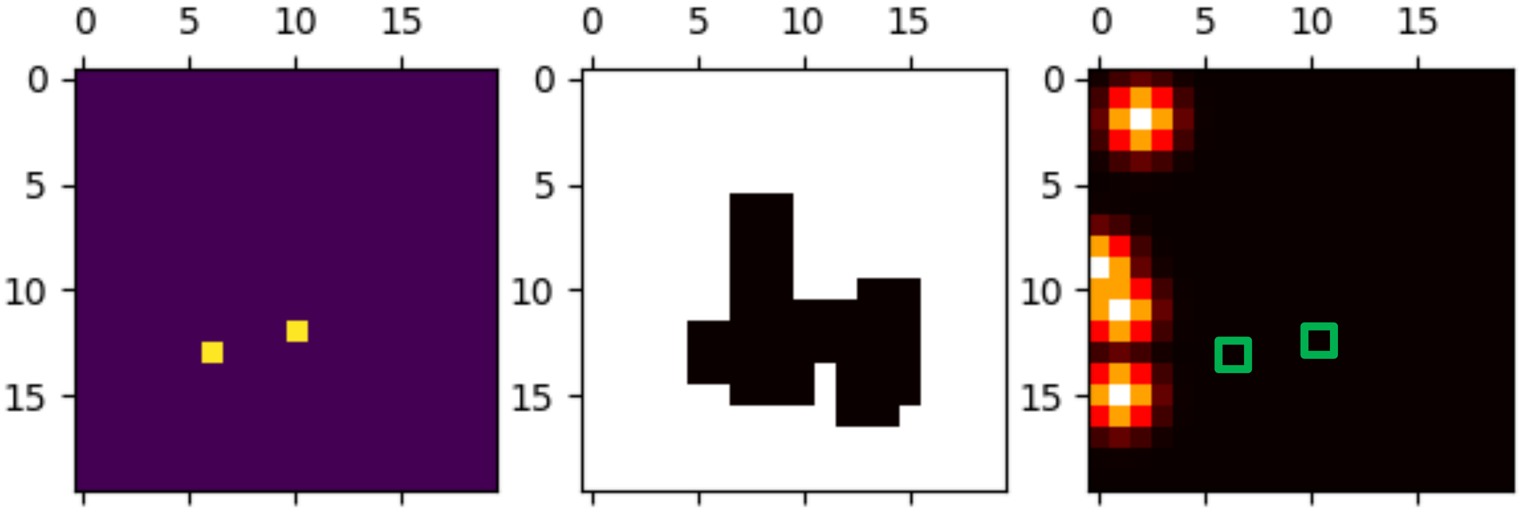
0
5
10
15

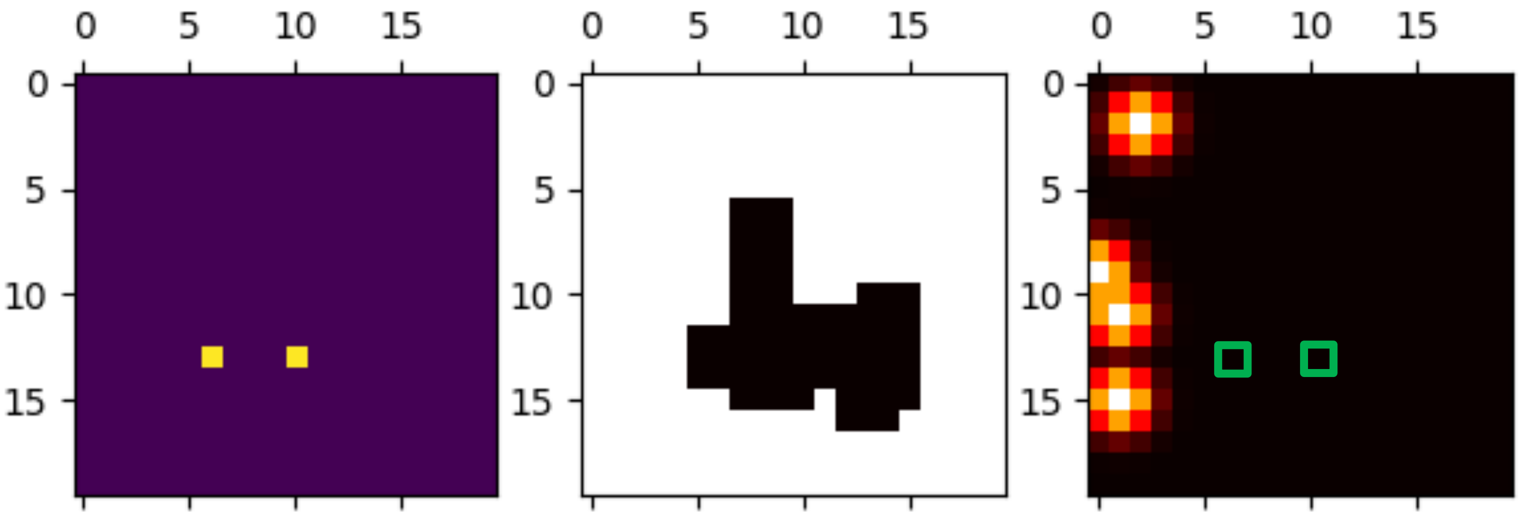


0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15

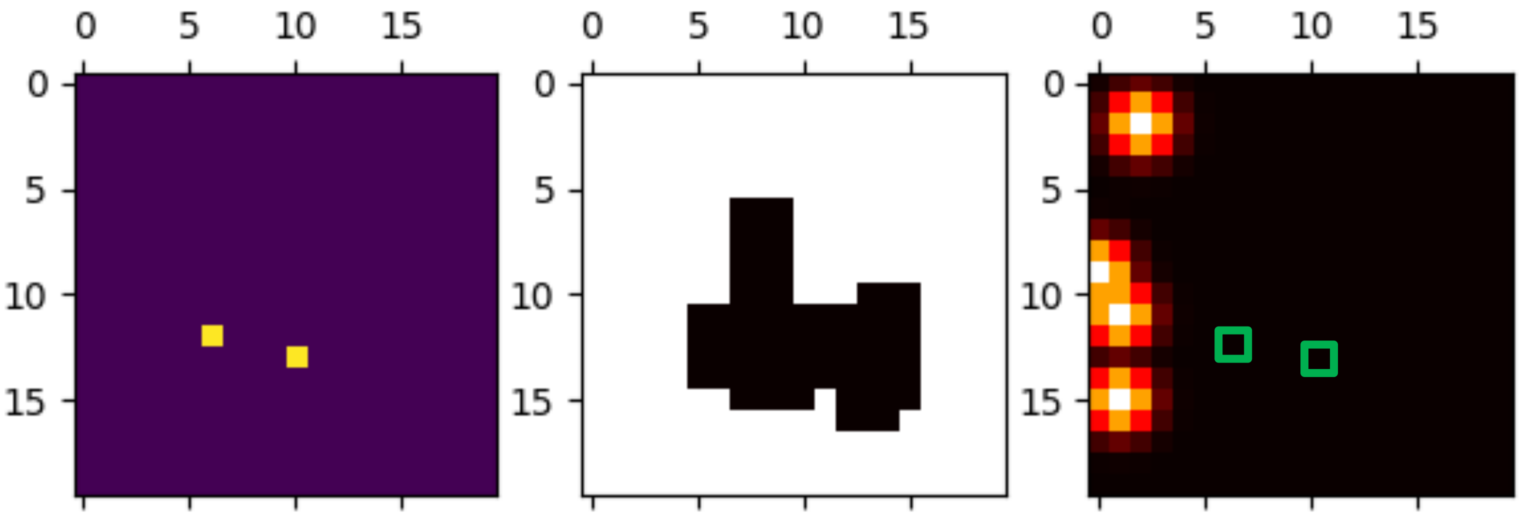


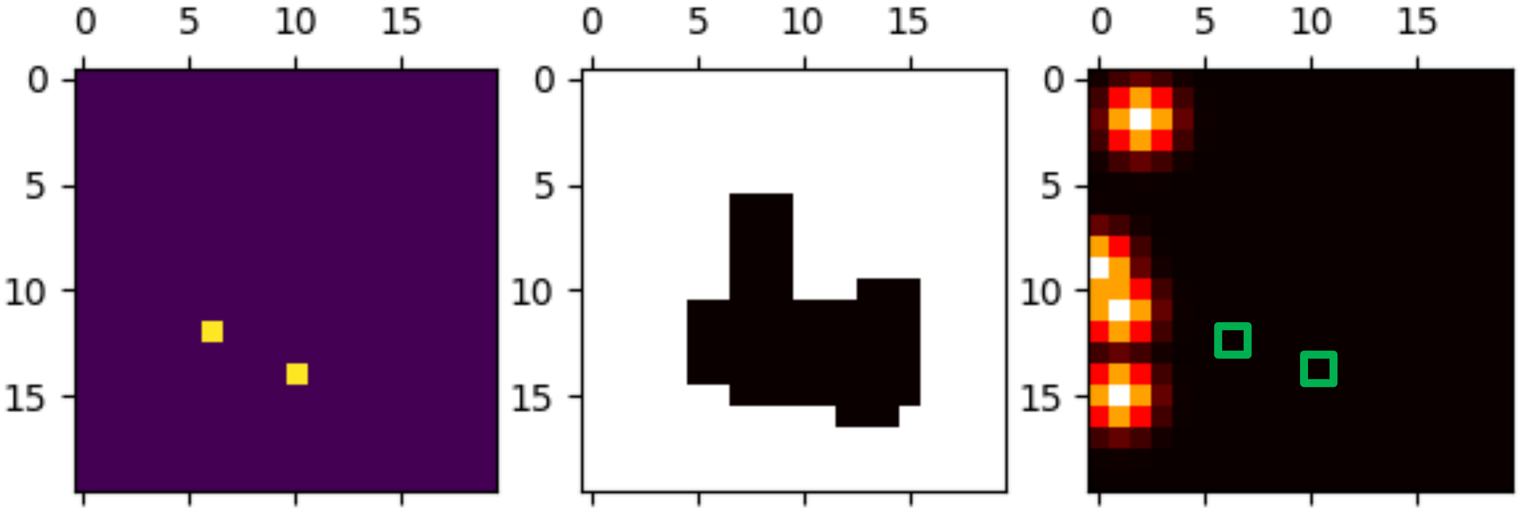
0
5
10
15


0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15

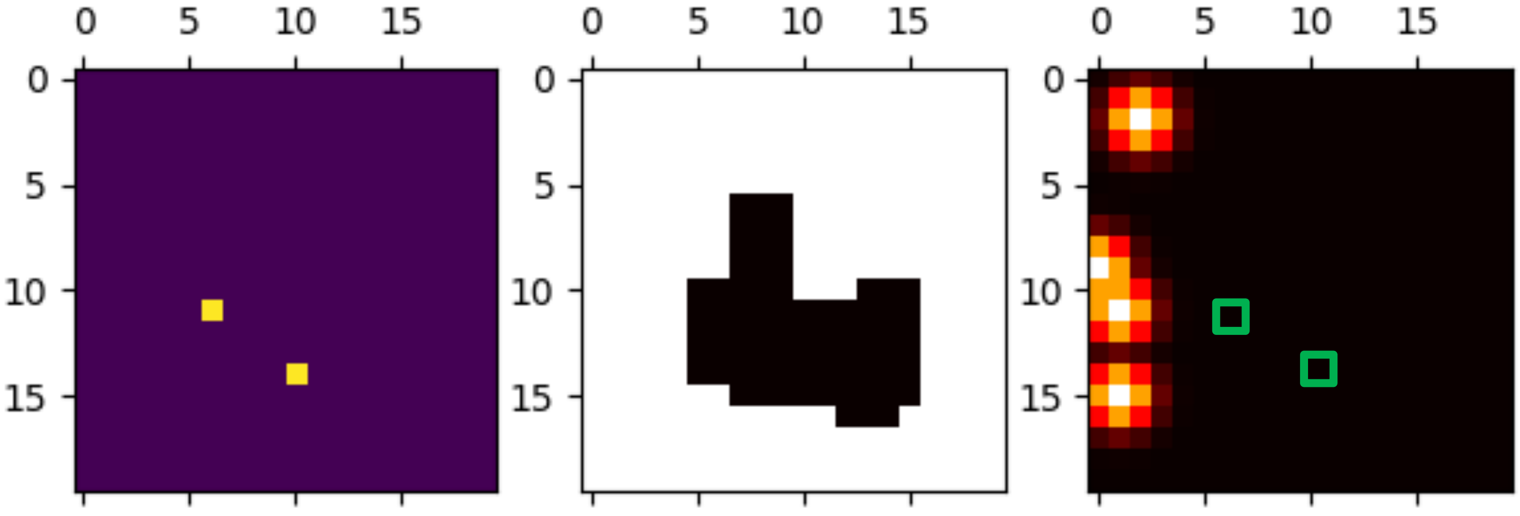


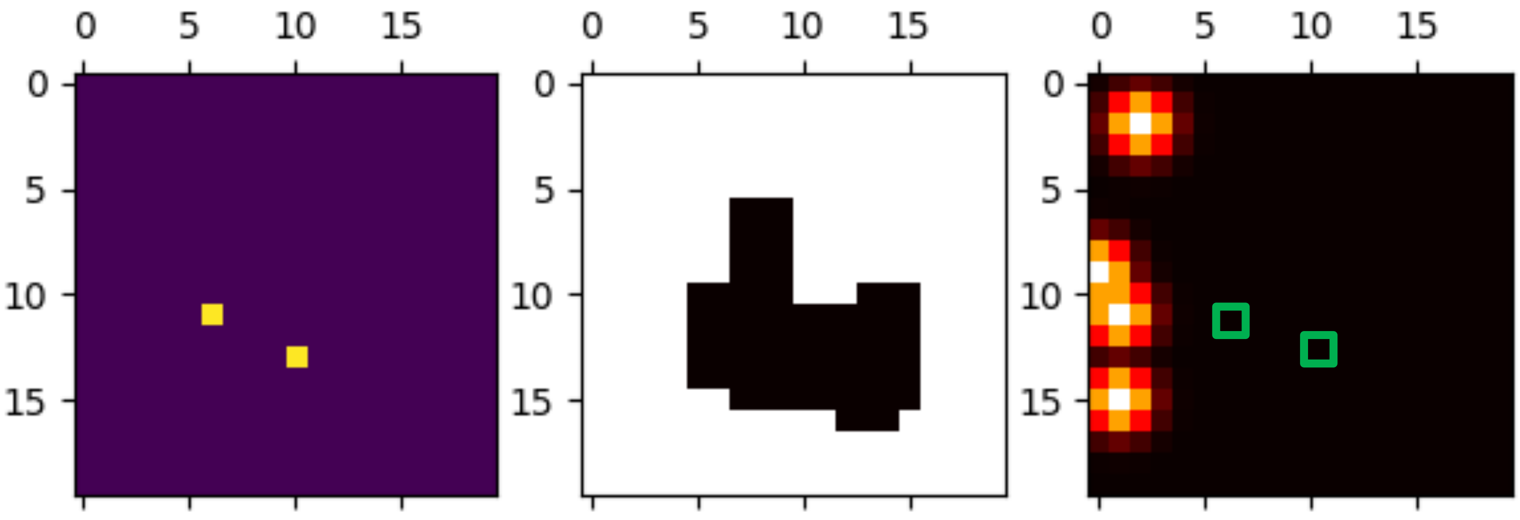
0
5
10
15

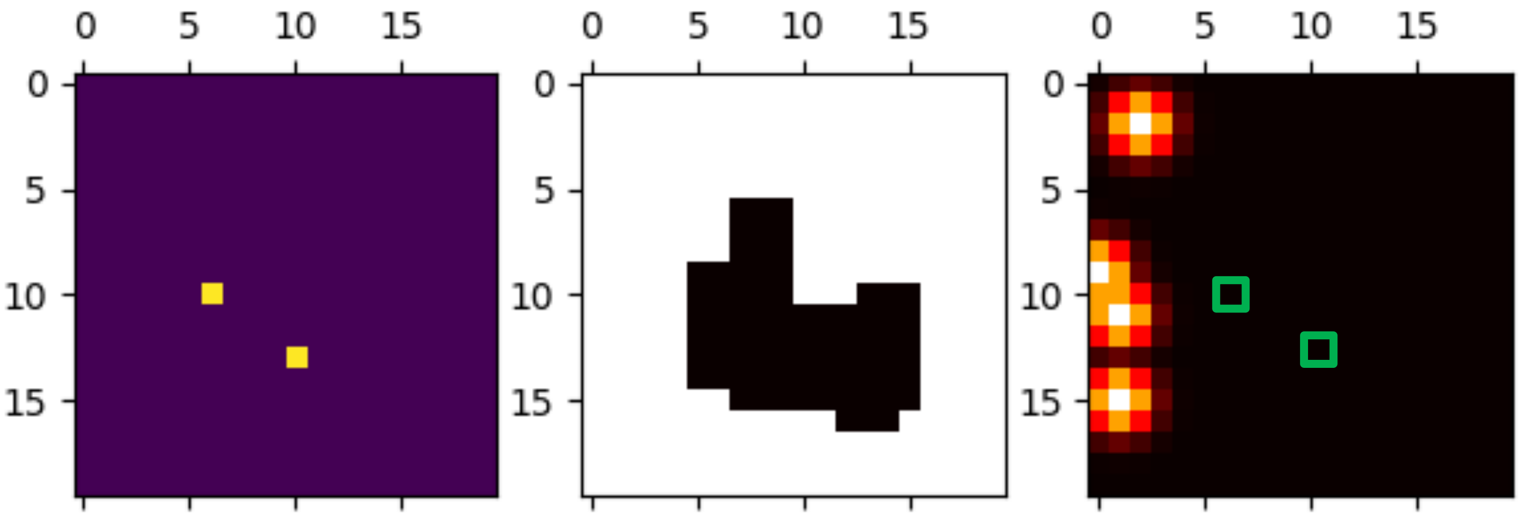

0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15

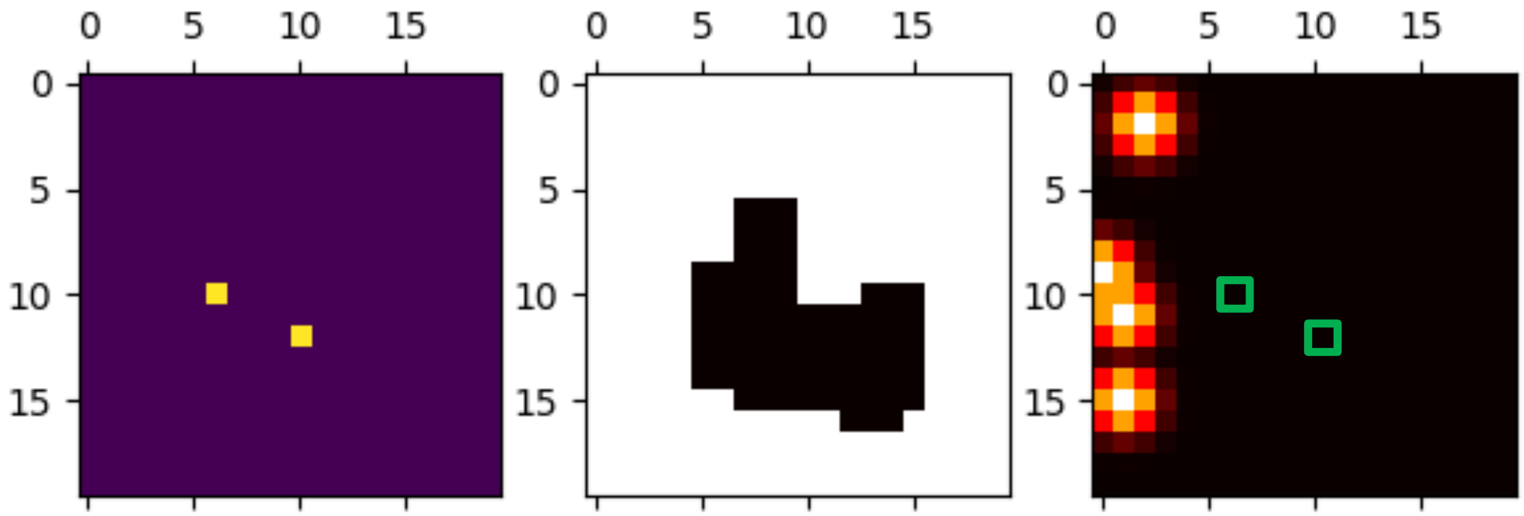


0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15

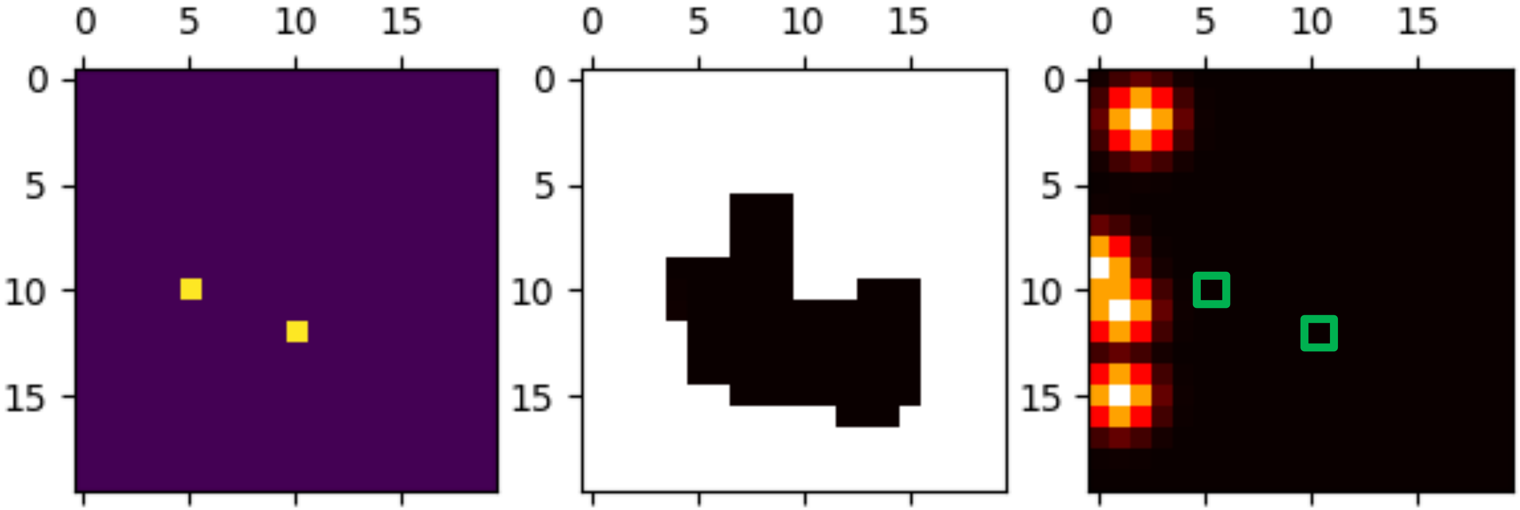

0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15


0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15

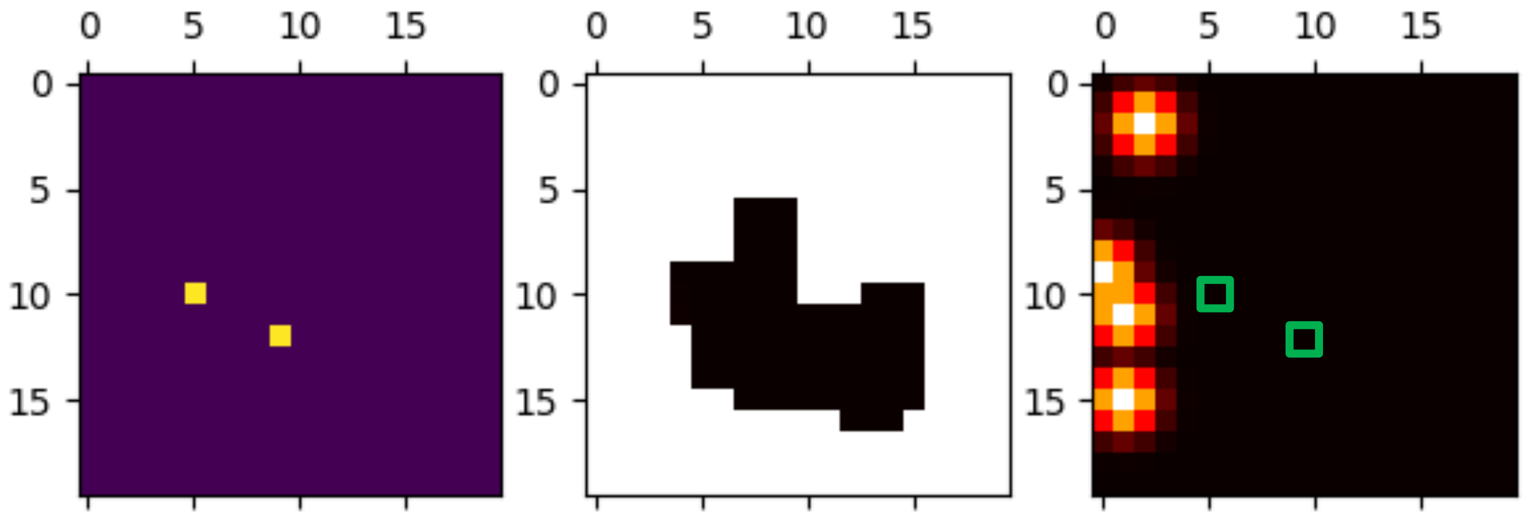

0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15

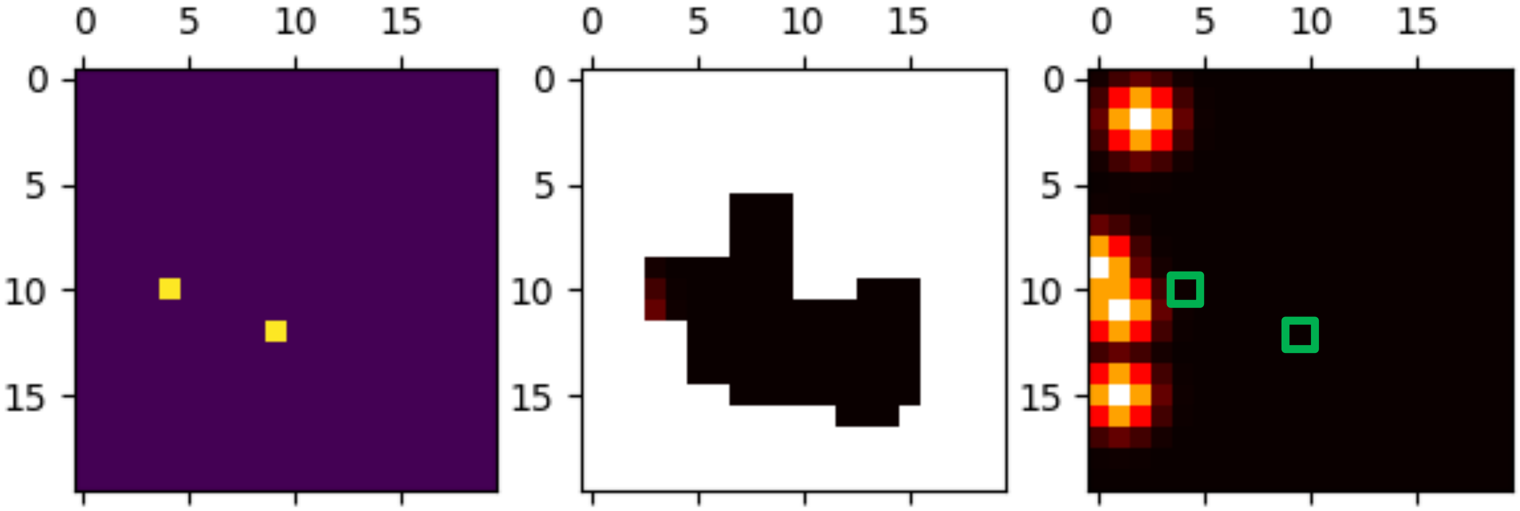


0
5
10
15

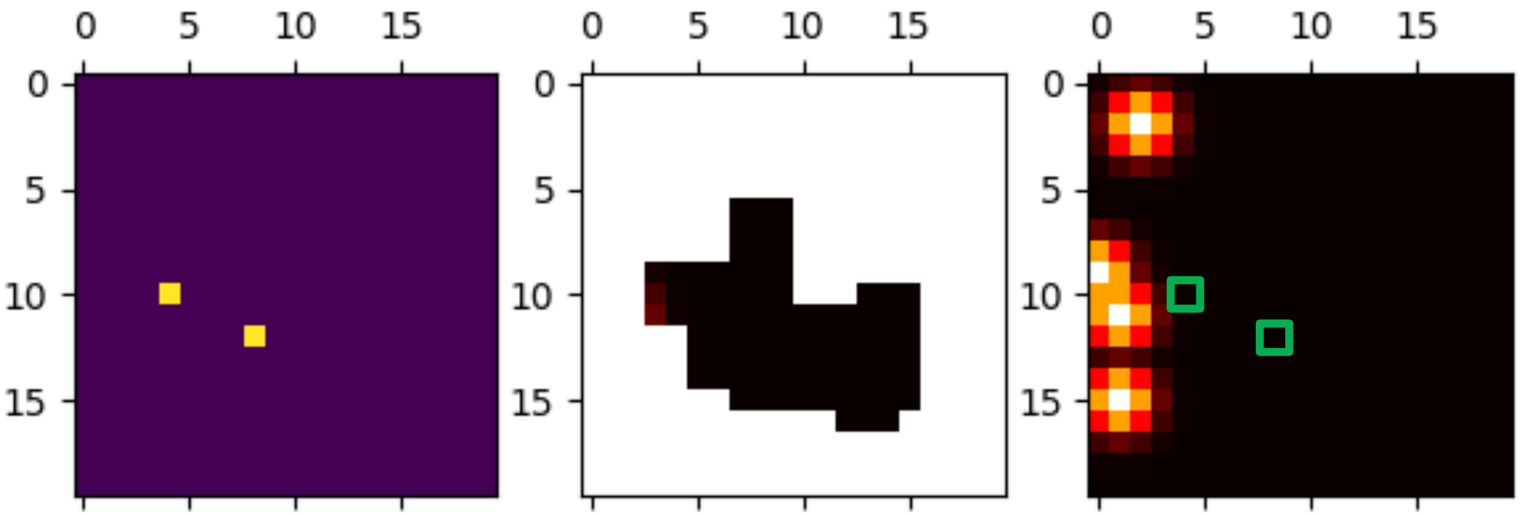


0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15

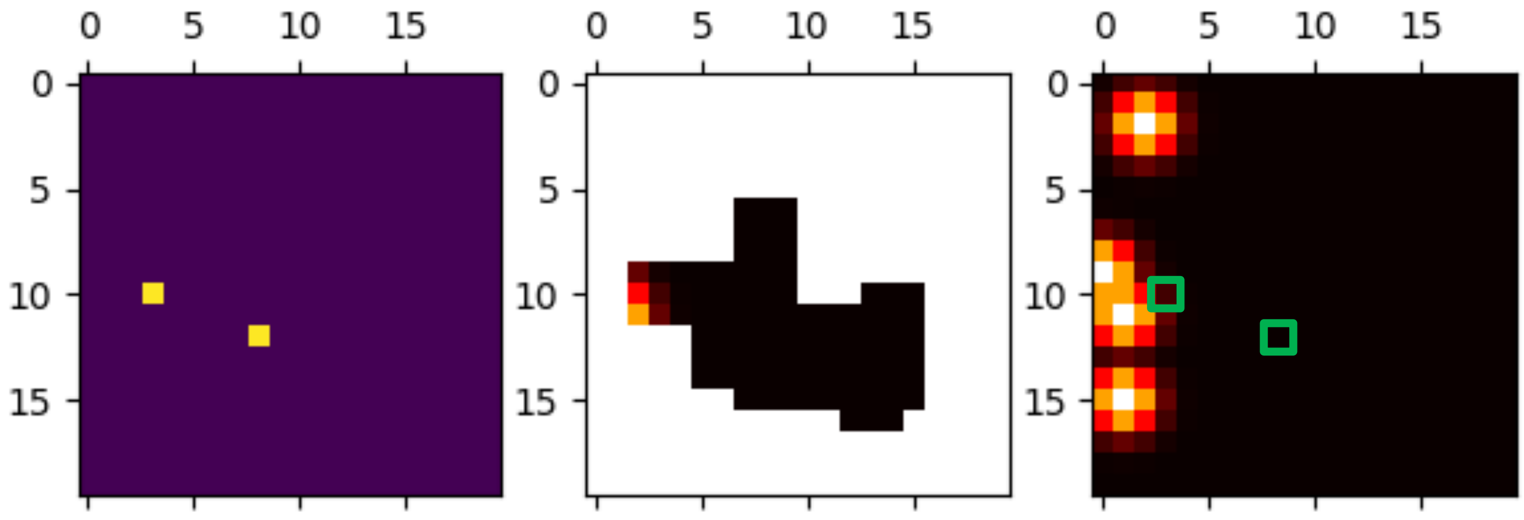

0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15

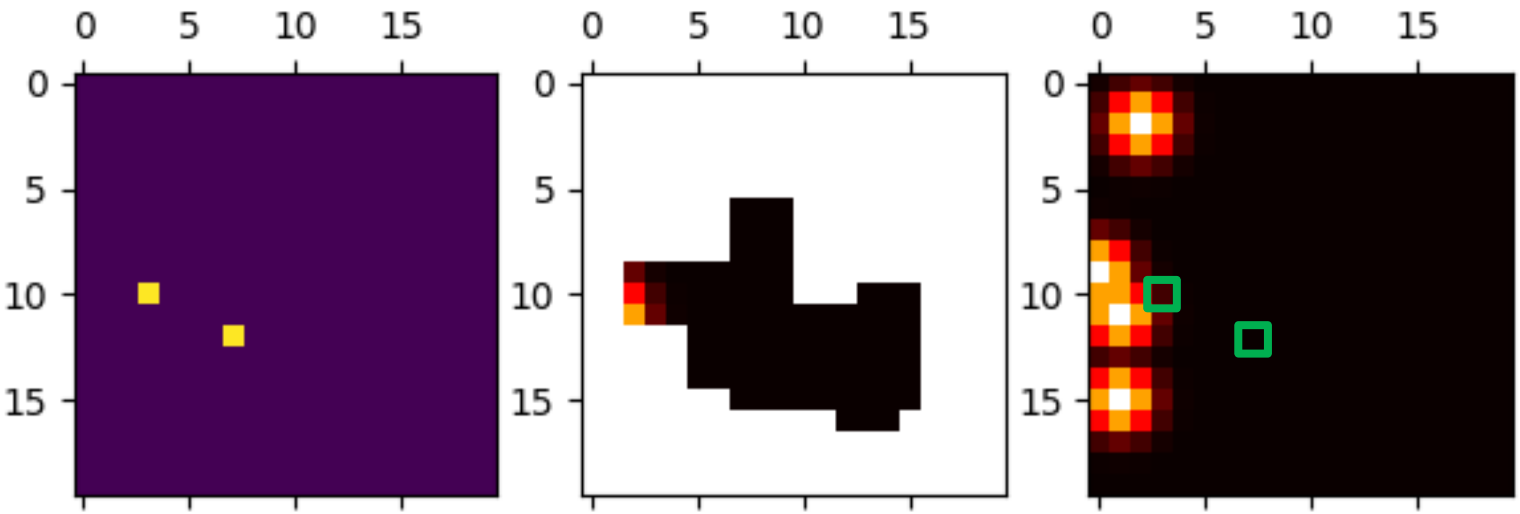


0
5
10
15
0
5
10
15
0
5
10
15

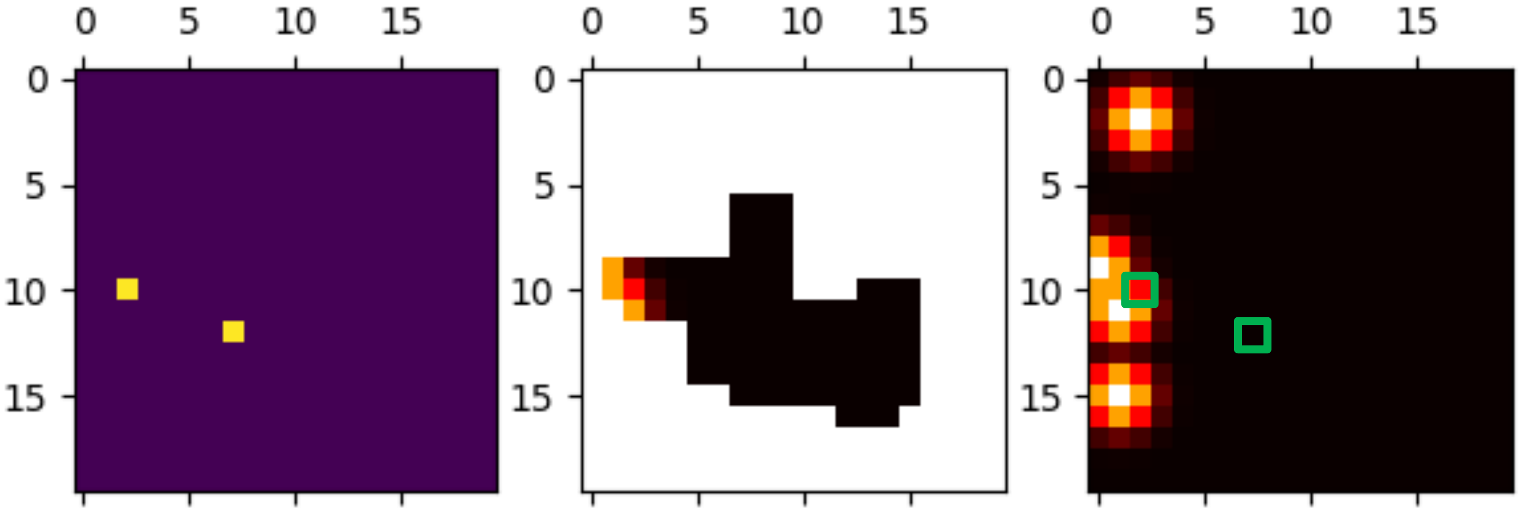


0
5
10
15

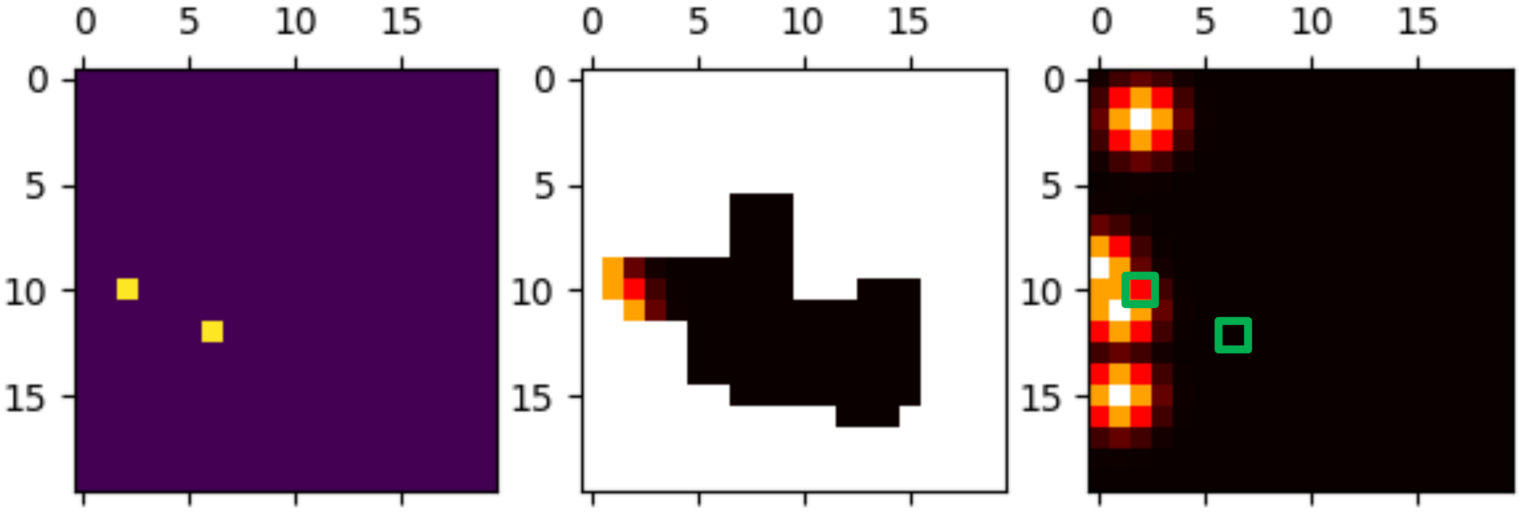

0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15

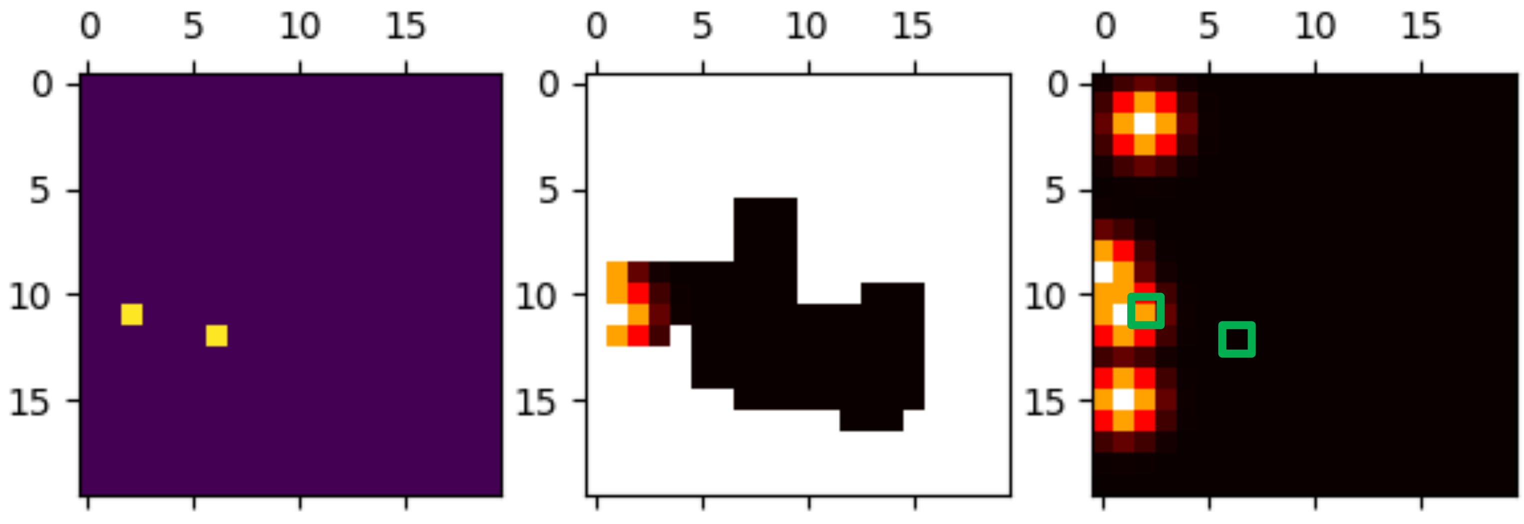


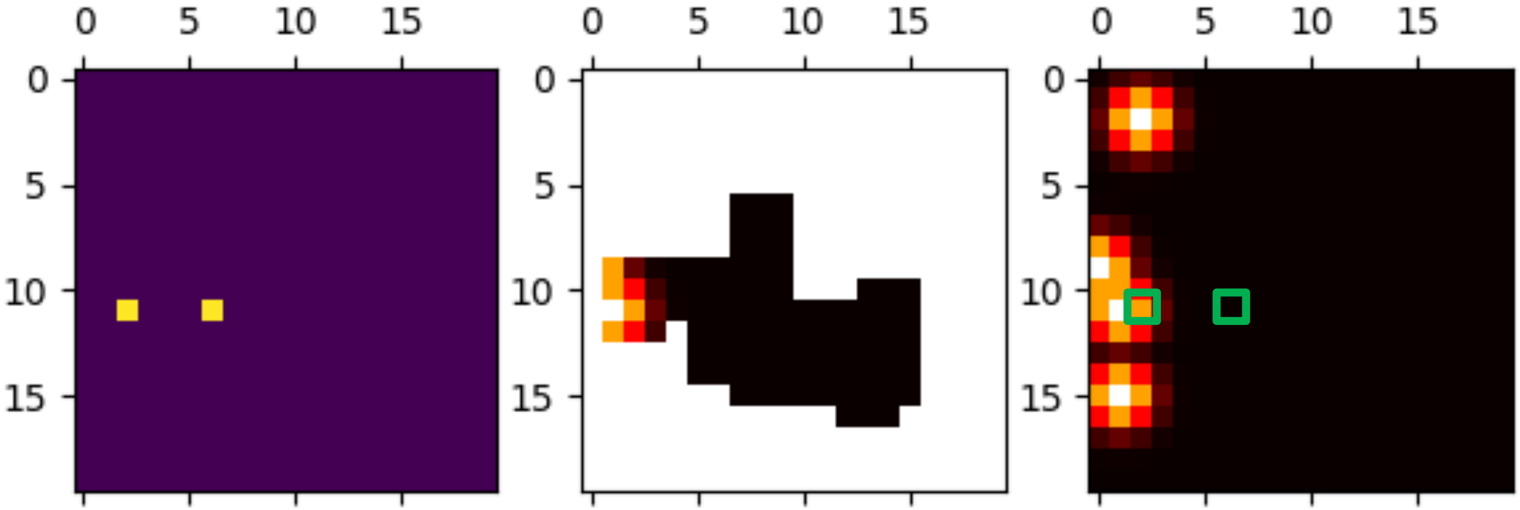
0
5
10
15

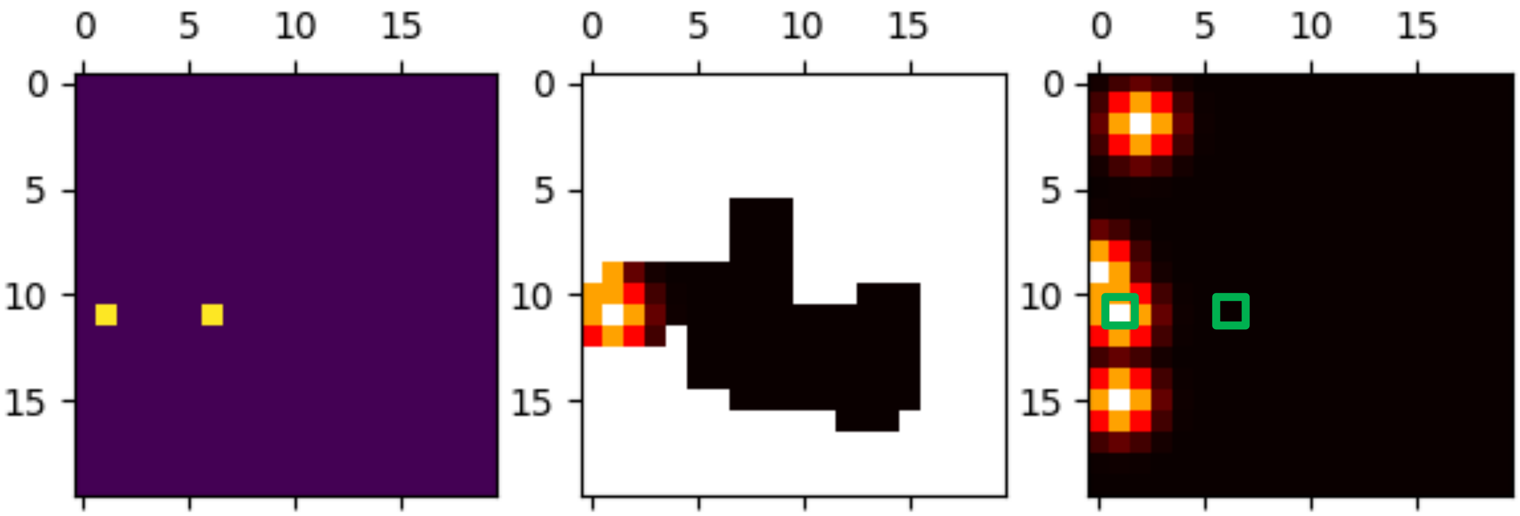


0
5
10
15

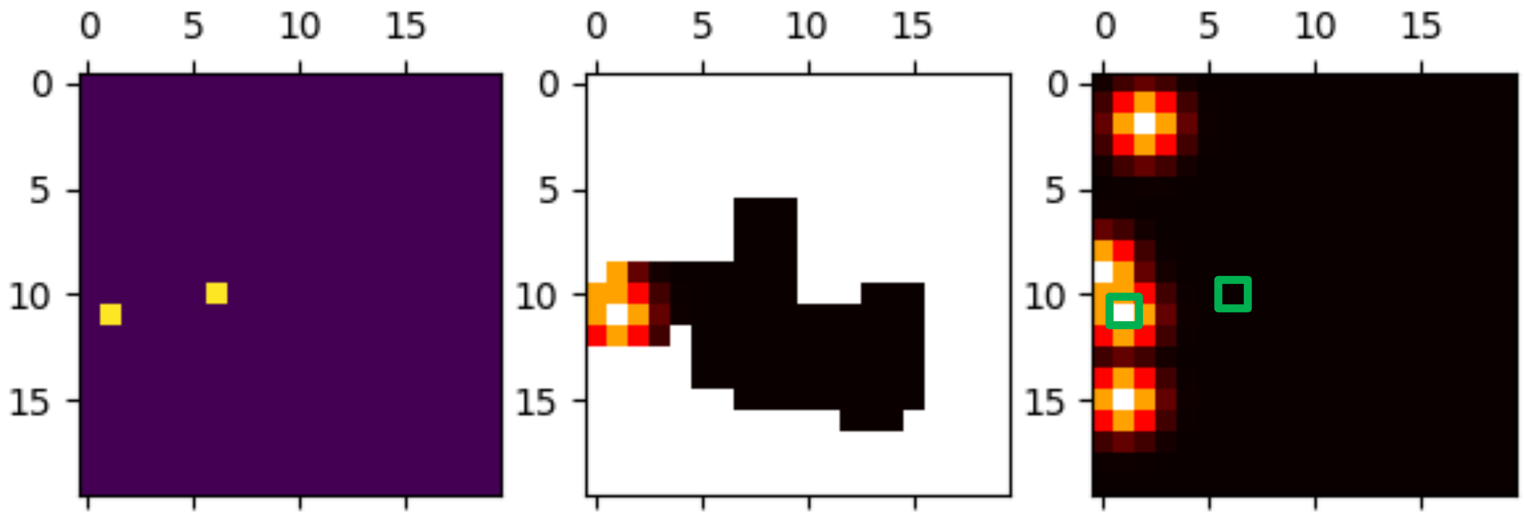

0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15


0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15

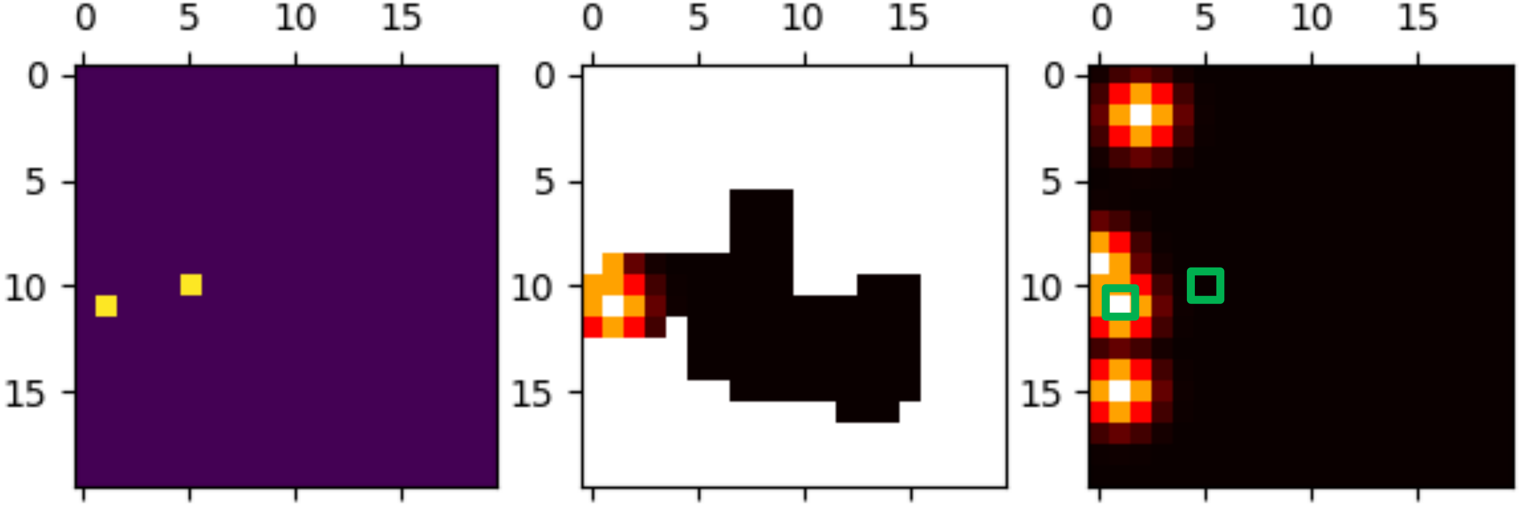

0 5 10 15
0 5 10 15
0 5 10 15

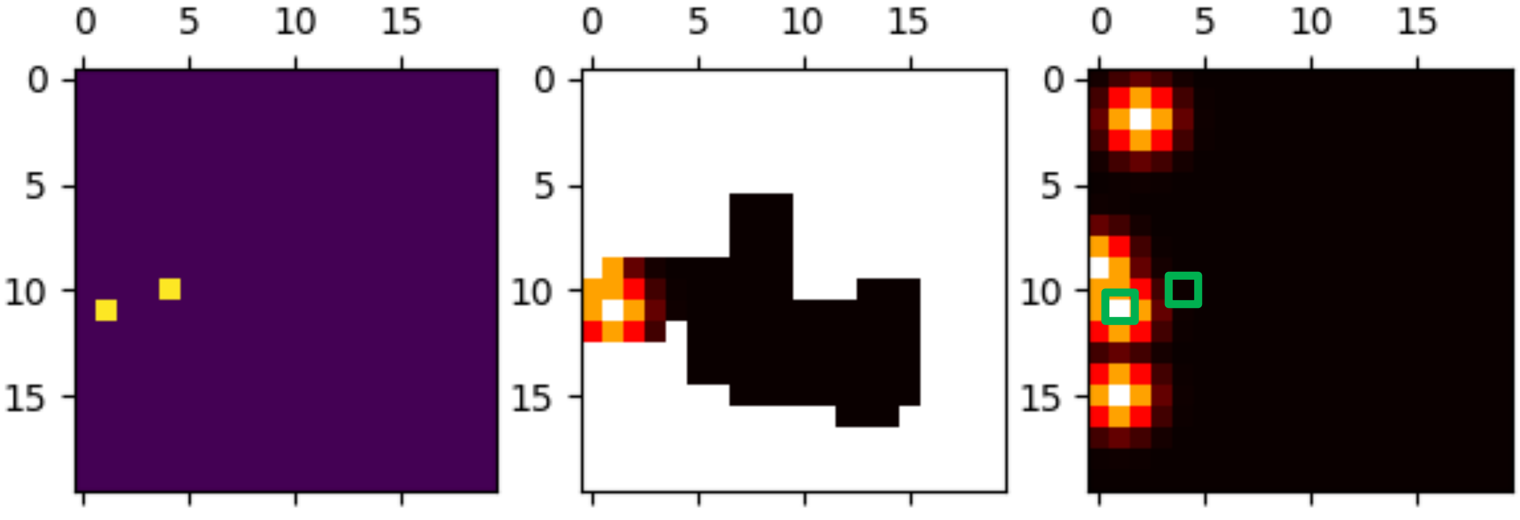


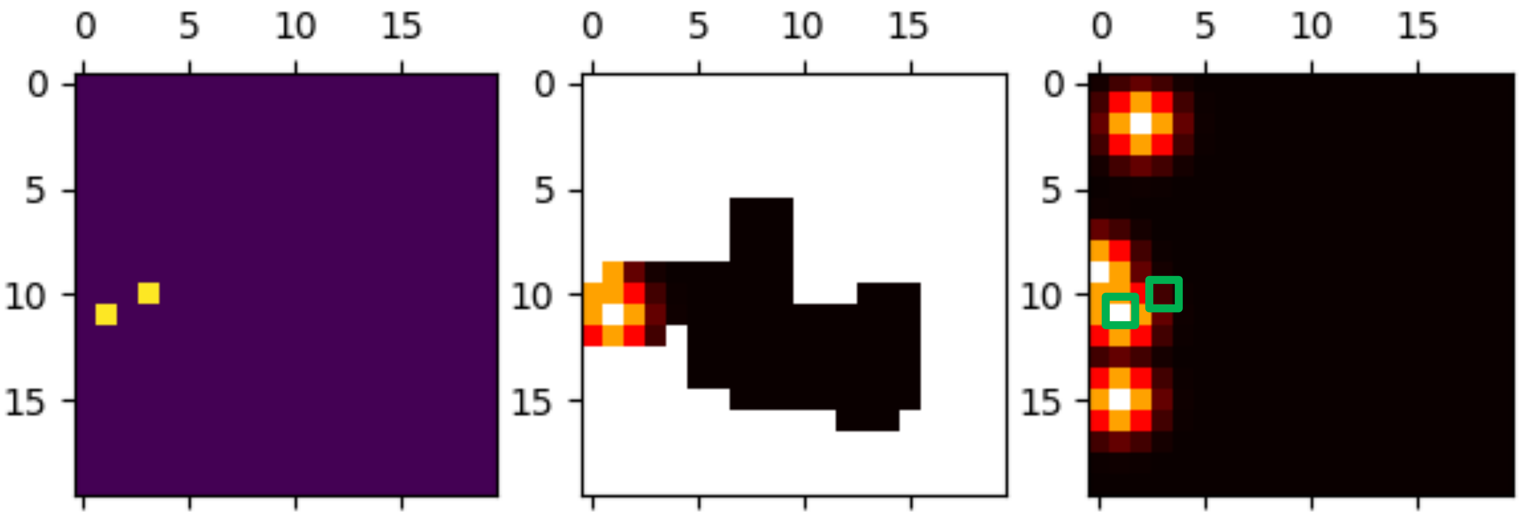
0
5
10
15

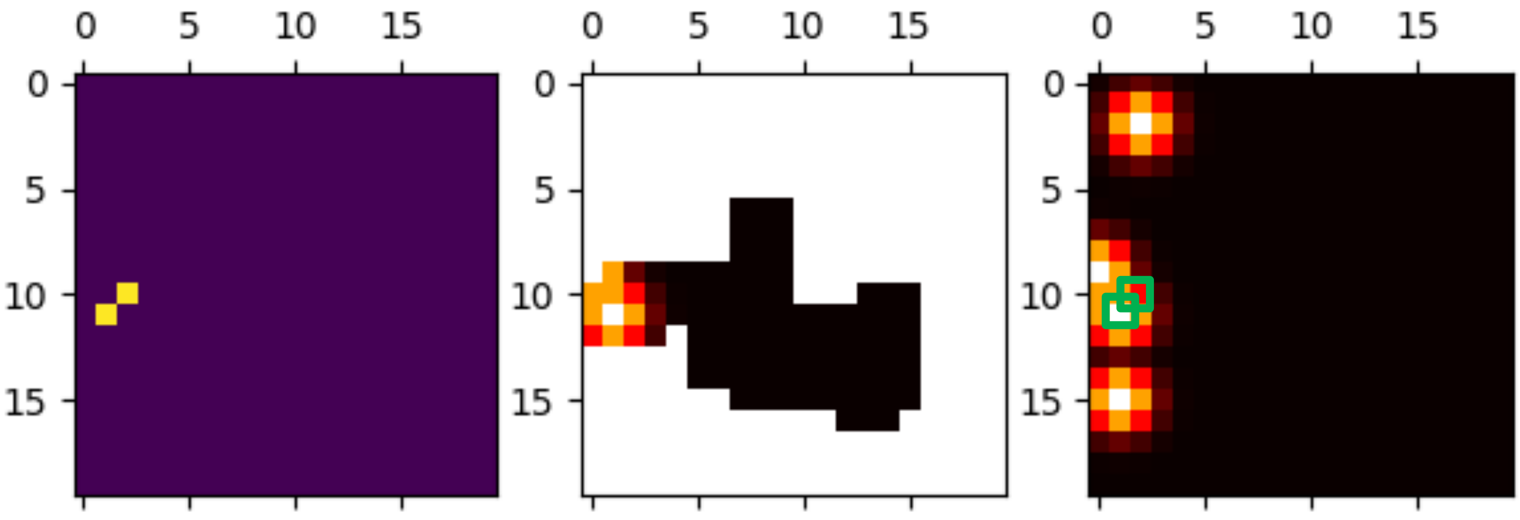


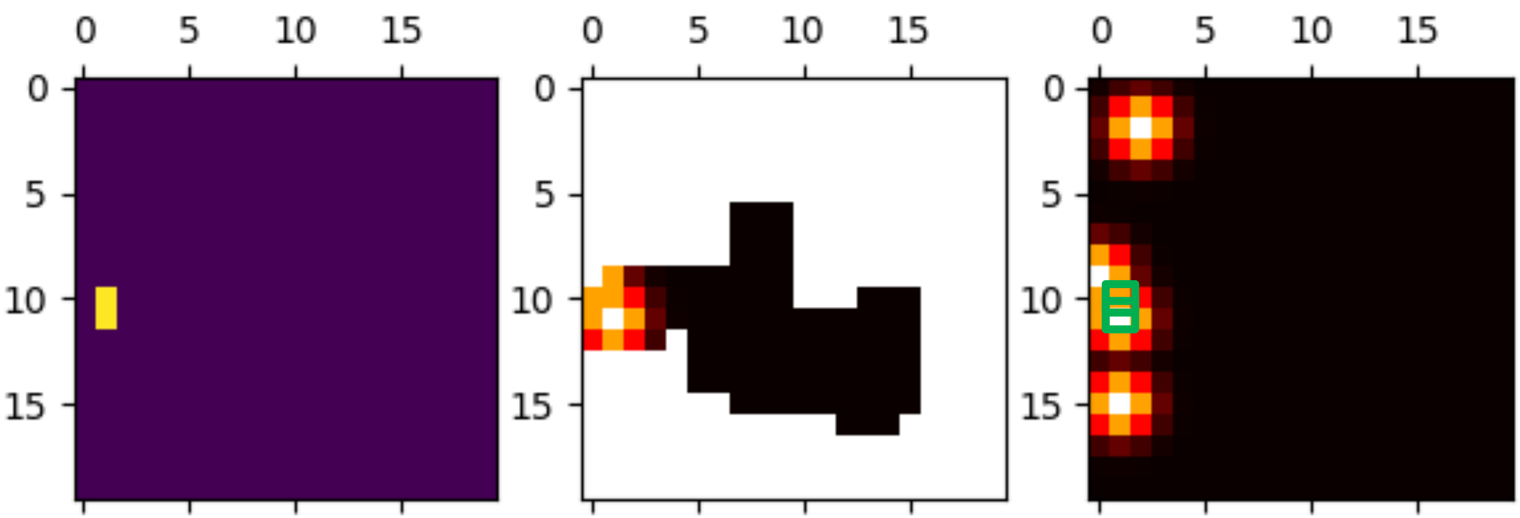
0
5
10
15

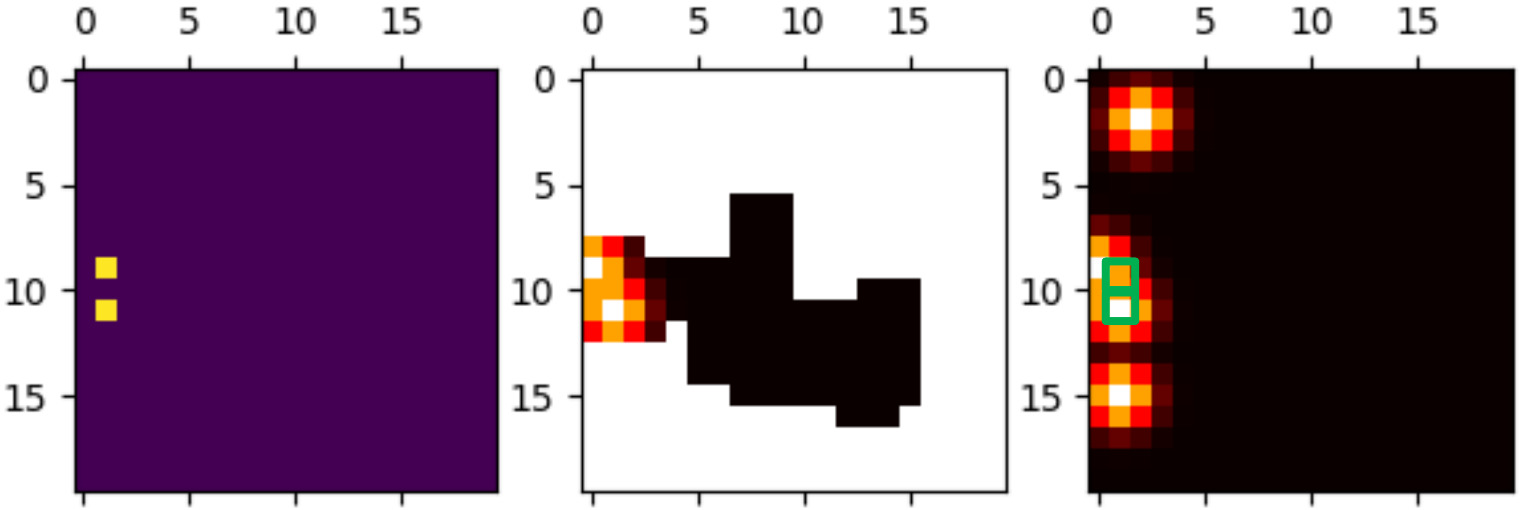

0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15

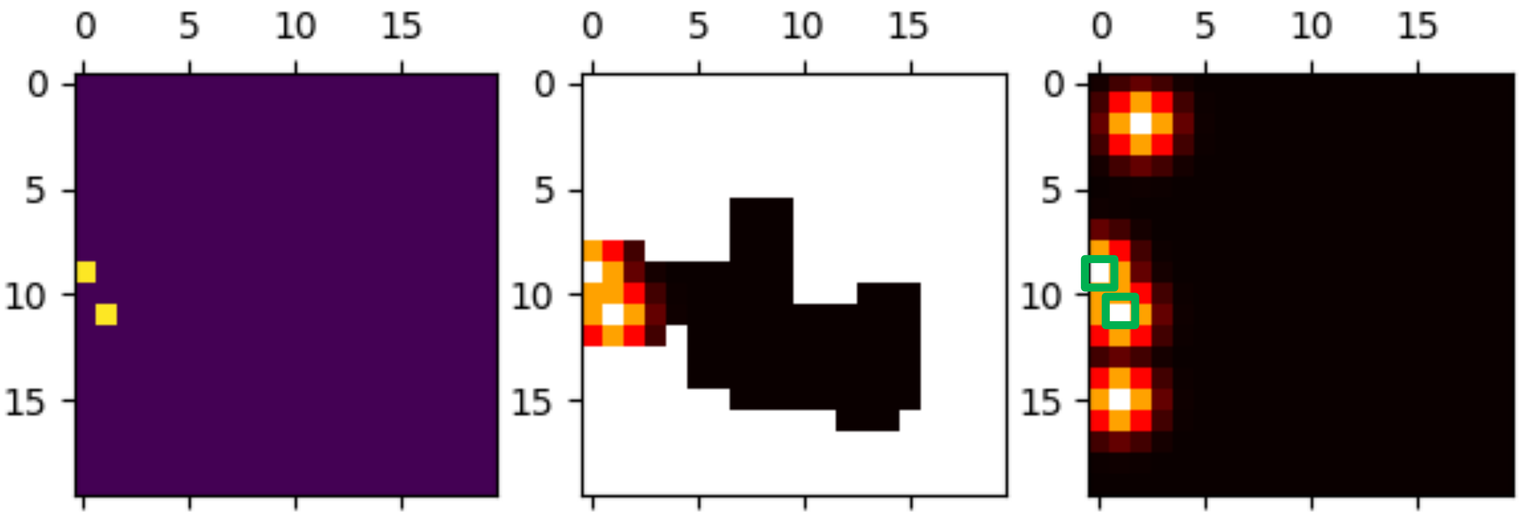

0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15


0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15
0 5 10 15


0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15

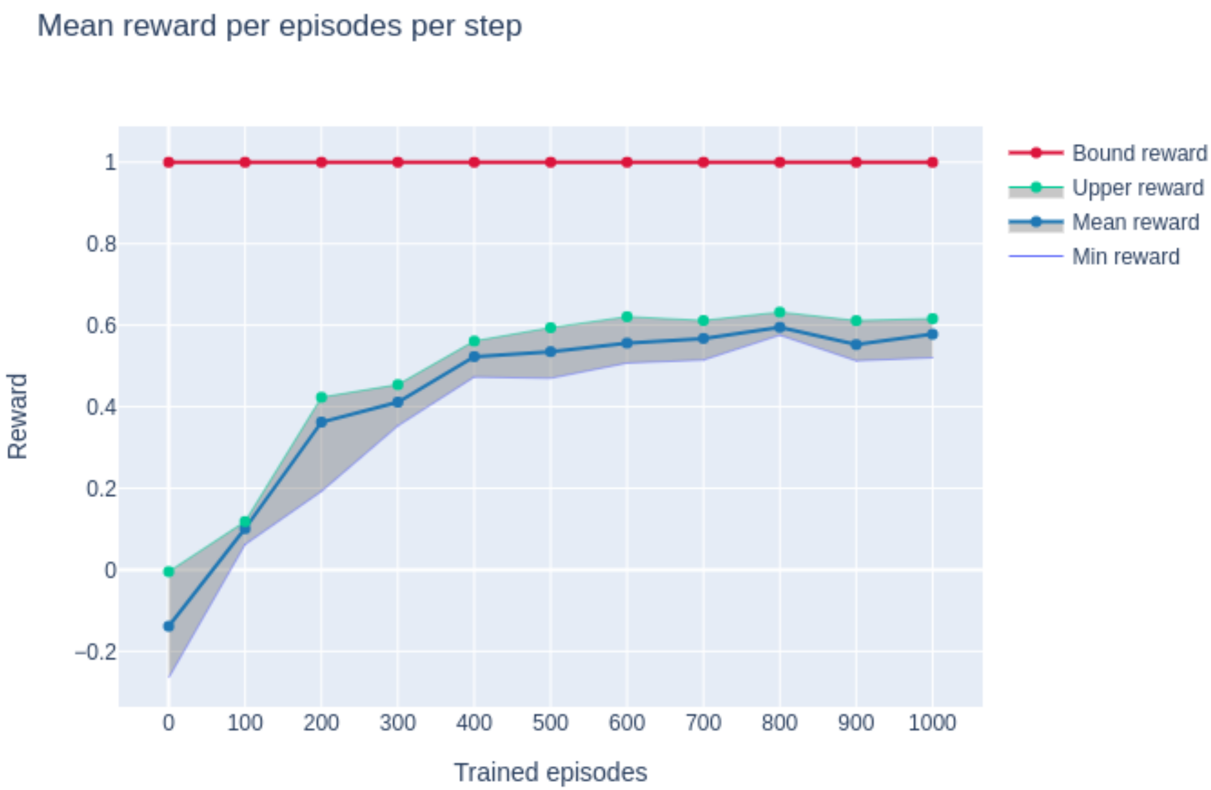

0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15
0
5
10
15


0 5 10 15
0
5
10
15



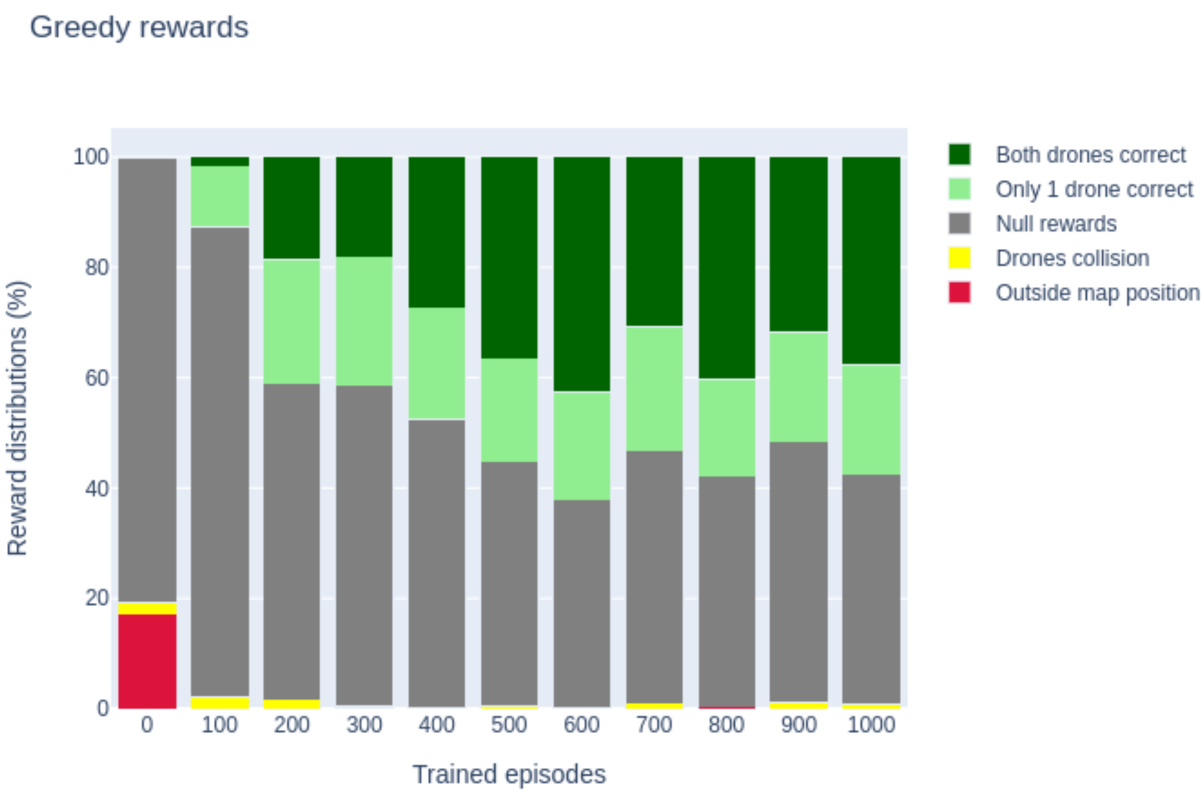
0
5
10
15

Training



We trained the algorithm 5 times.
These are the performances during learning of the 5 trainings.

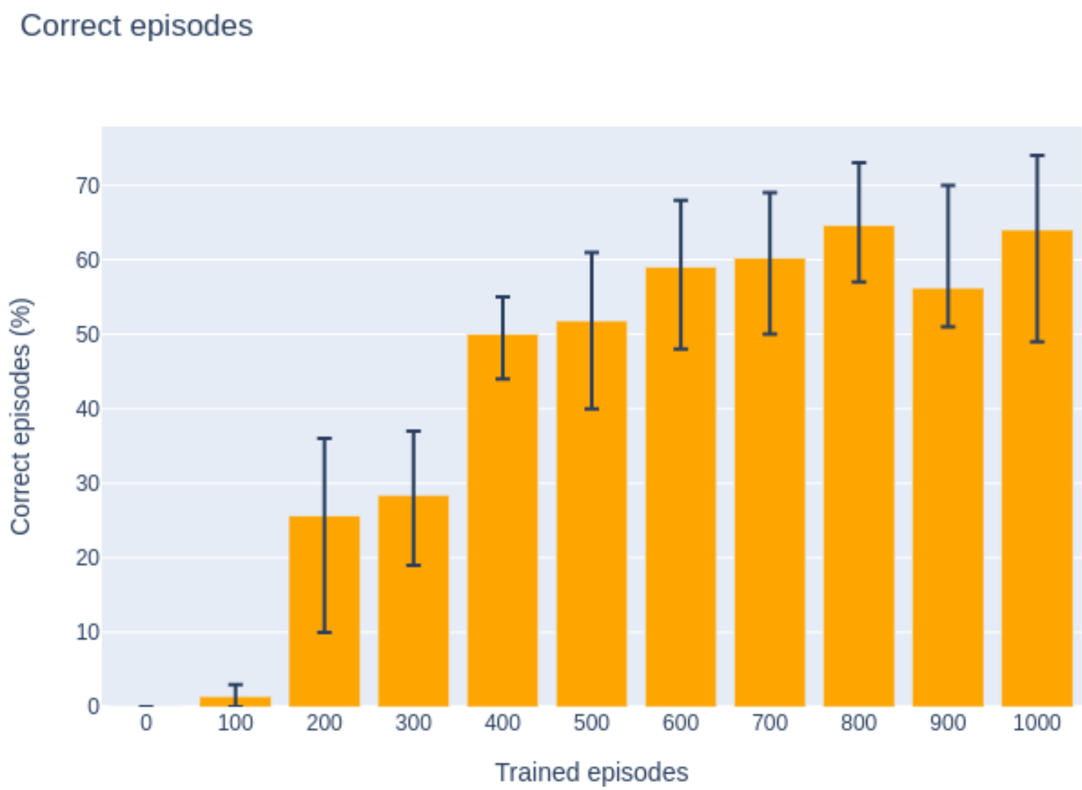
Training (cont)



Every 100 episodes of training, we tested the learned strategy on 100 further episodes, and we plotted the distribution of the rewards accumulated during the testing phase

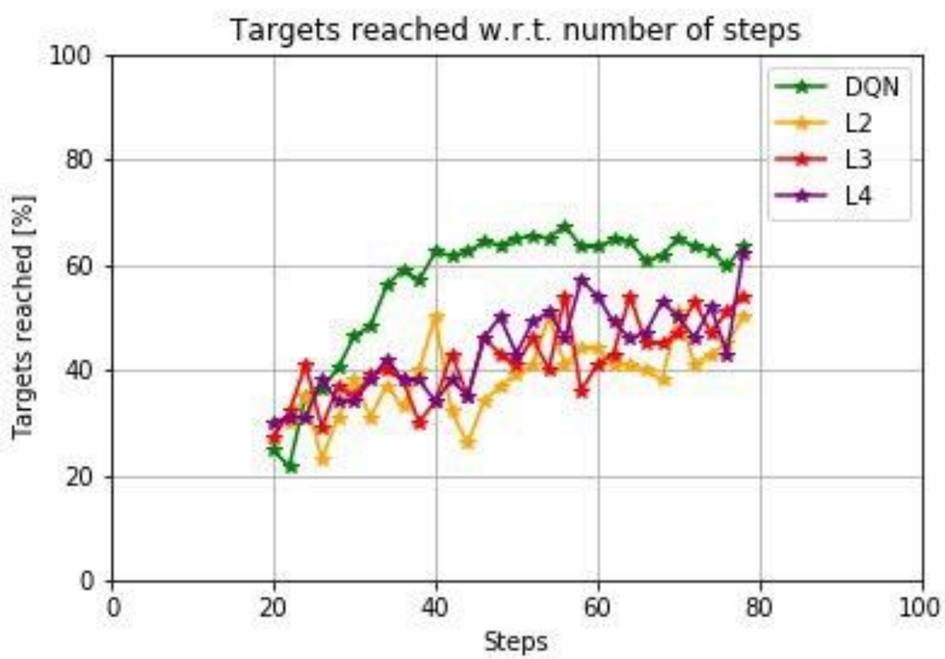
You can see that the drones learn quite soon to remain inside the map. The real challenge is to make both of them able to reach the targets.

Training (cont)



Our algorithm is able to place both the drones on top of 2 targets in at least the 60% of the simulated episodes

Results



We also compared our algorithm (DQN) with some look-ahead heuristic, named L2,L3, and L4, where the number indicates the look-ahed capability, i.e., the number of step forward that the policy can anticipate to evaluate the next action.
DQN requires less steps to reach the targets, showing better coordination among the drones.

Work in progress

- Transfer Learning to generalize to an arbitrary number of drones
- Dynamic map
- Application of Reinforcement Learning to the routing scenarios
- Hierarchial Reinforcement Learning to coordinate drones with different tasks (exploration, exploitation…)