Unsupervised object discovery for instance recognition

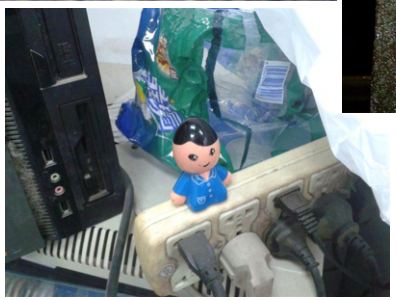
Oriane Siméoni *Ahmet Iscen* *Giorgos Tolias*

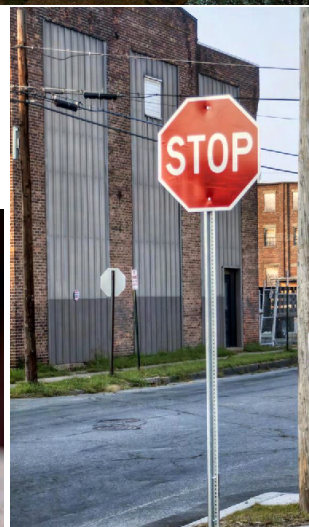
Yannis Avrithis *Ondřej Chum*

Retrieving the query

One query

Database



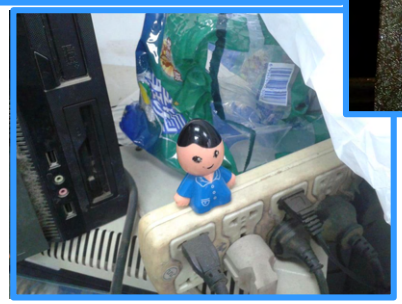


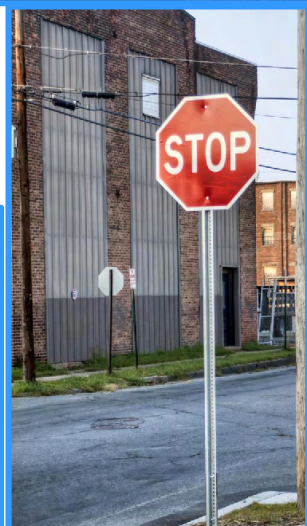
Retrieving the query

- One solution :
  - Index images using **one global descriptor** from the entire image

One query

Database



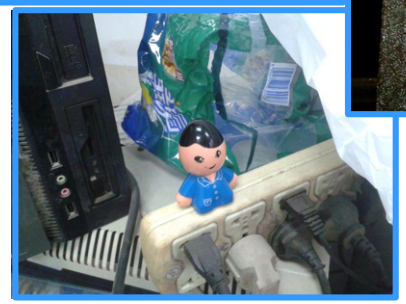


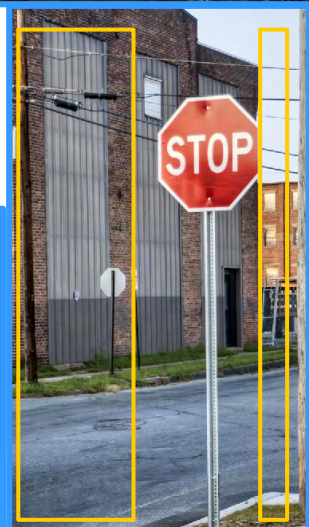
Retrieving the query

- One solution :
  - Index images using **one global descriptor** from the entire image
    - BUT : background

One query

Database



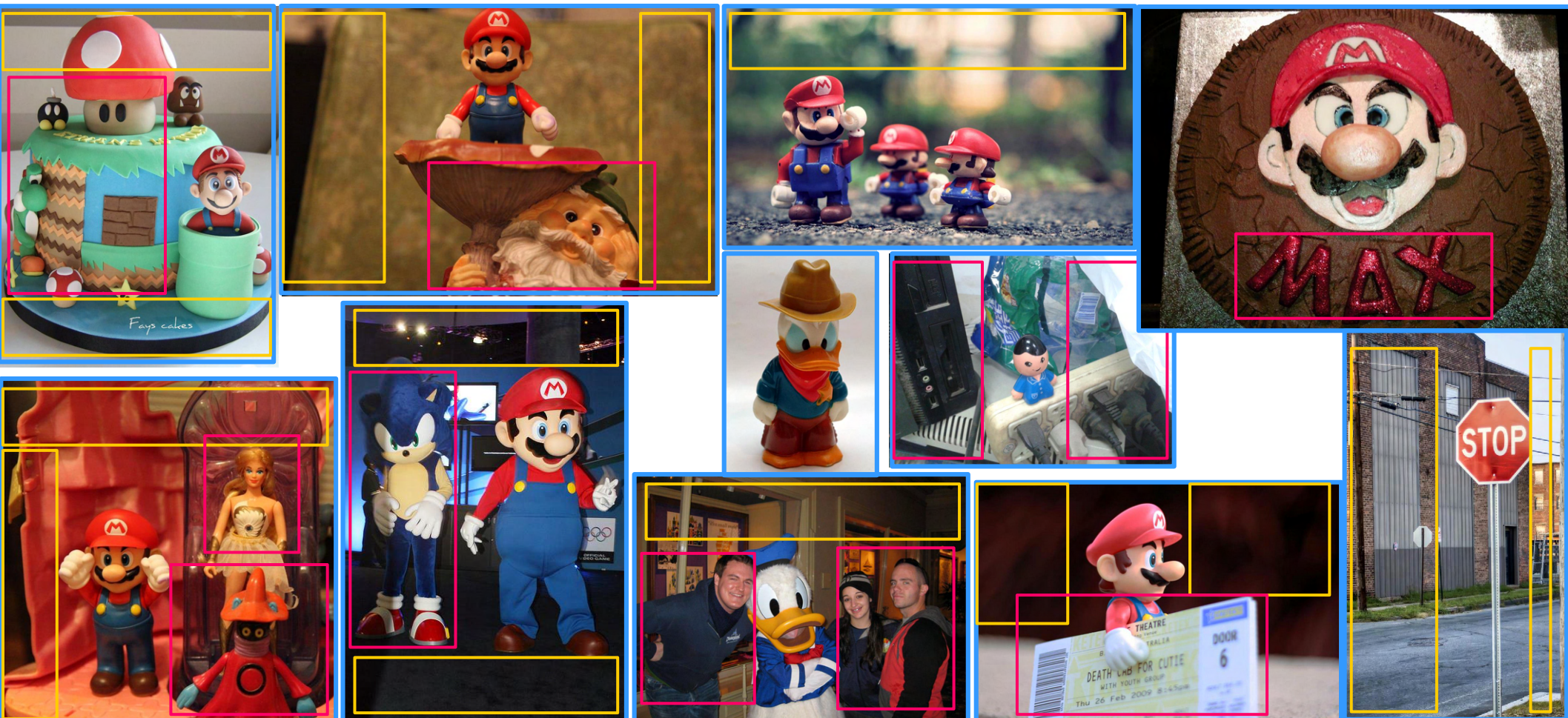


Retrieving the query

- One solution :
  - Index images using **one global descriptor** from the entire image
    - BUT : background, clutter and obstruction alter the quality of the descriptor

One query

Database

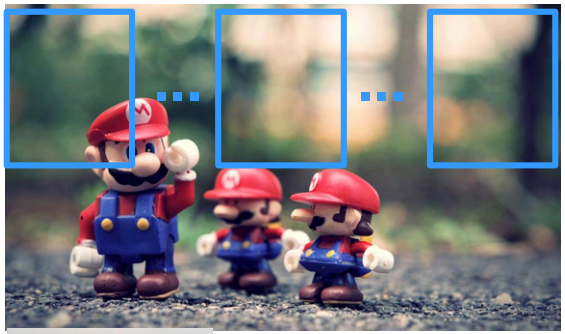


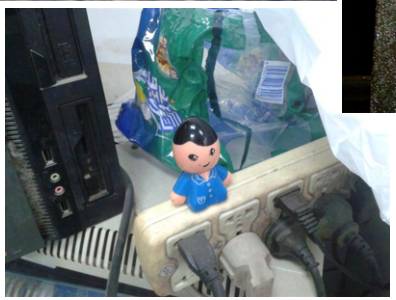
Retrieving the query

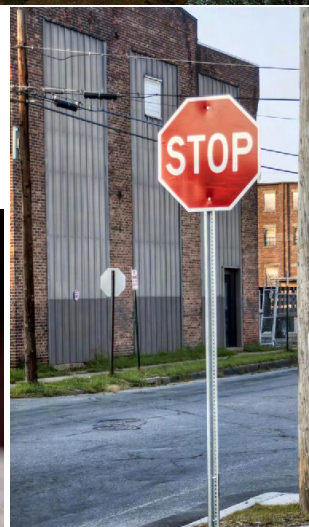
- Other solution :
  - Index images using **a regional descriptor** constructed from uniformly sampled regions

One query

Database







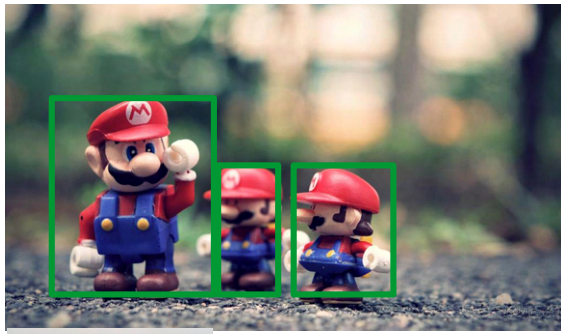
Retrieving the query

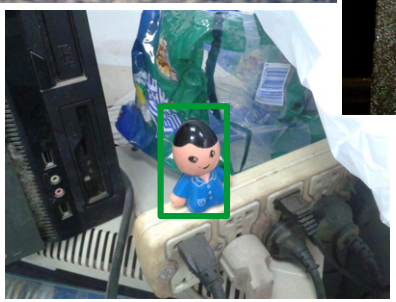
- **Our solution**:
  - Create one global descriptor using the most discriminative regions
    - → corresponding to repeating database objects

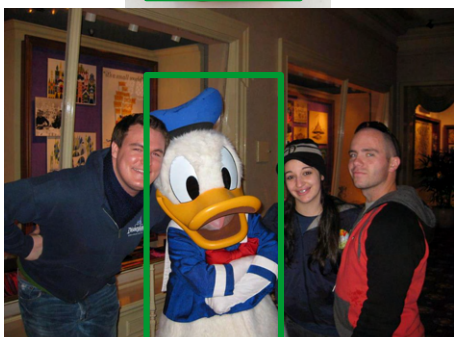
How to find them ?

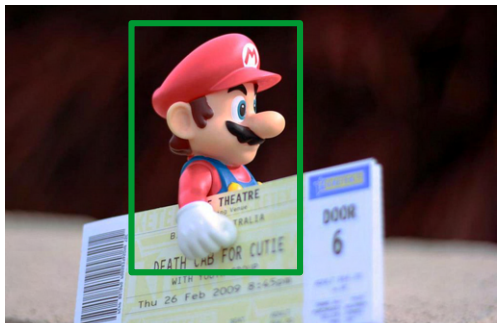
One query

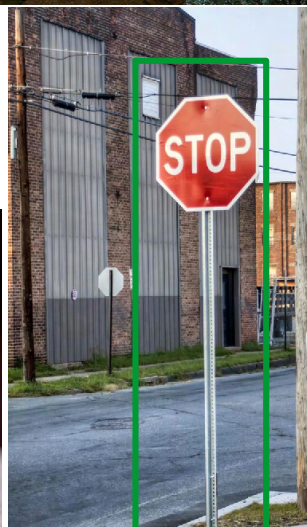
Database







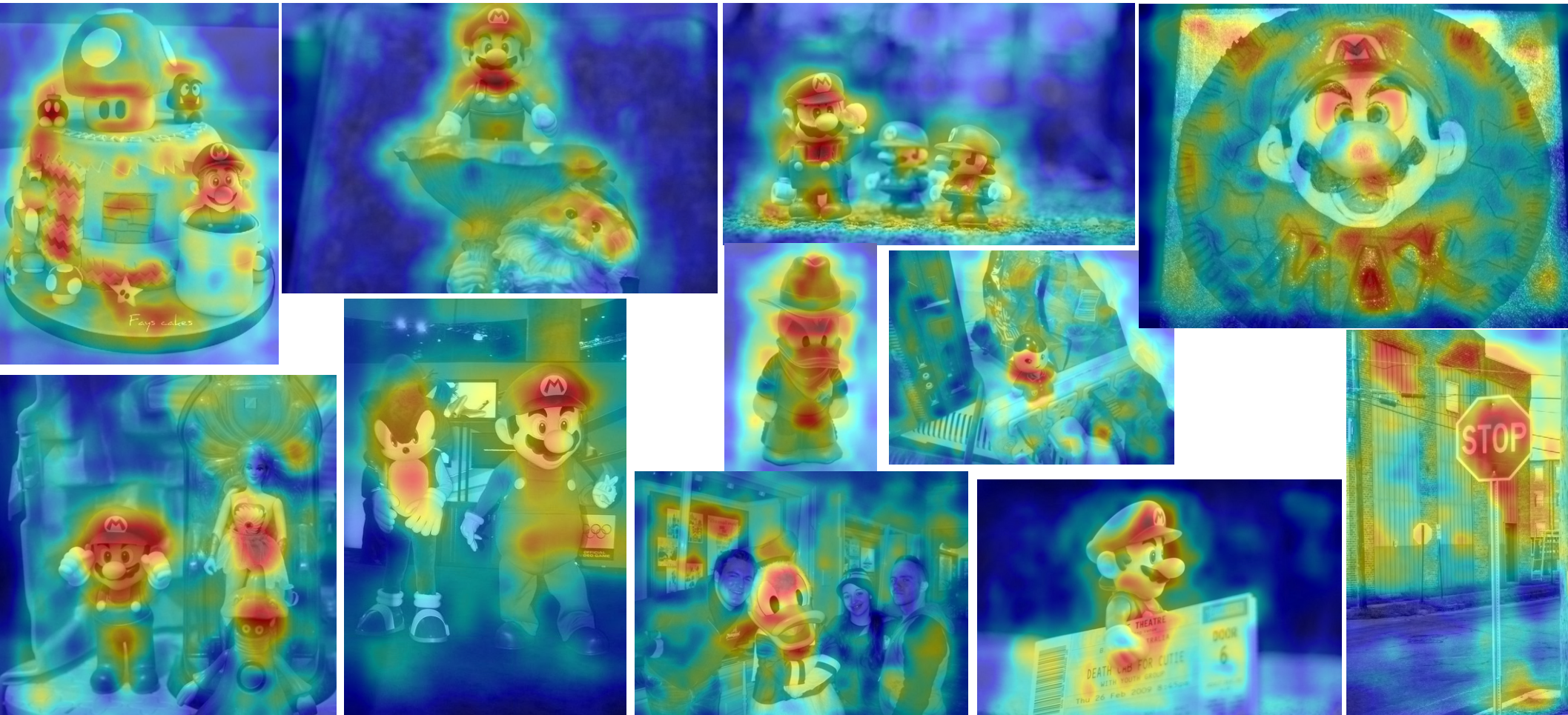




Create saliencies

- Computed using CNN activations
  - ➔ eg. CroW - *Kalantidis et al. ArXiv 2015*

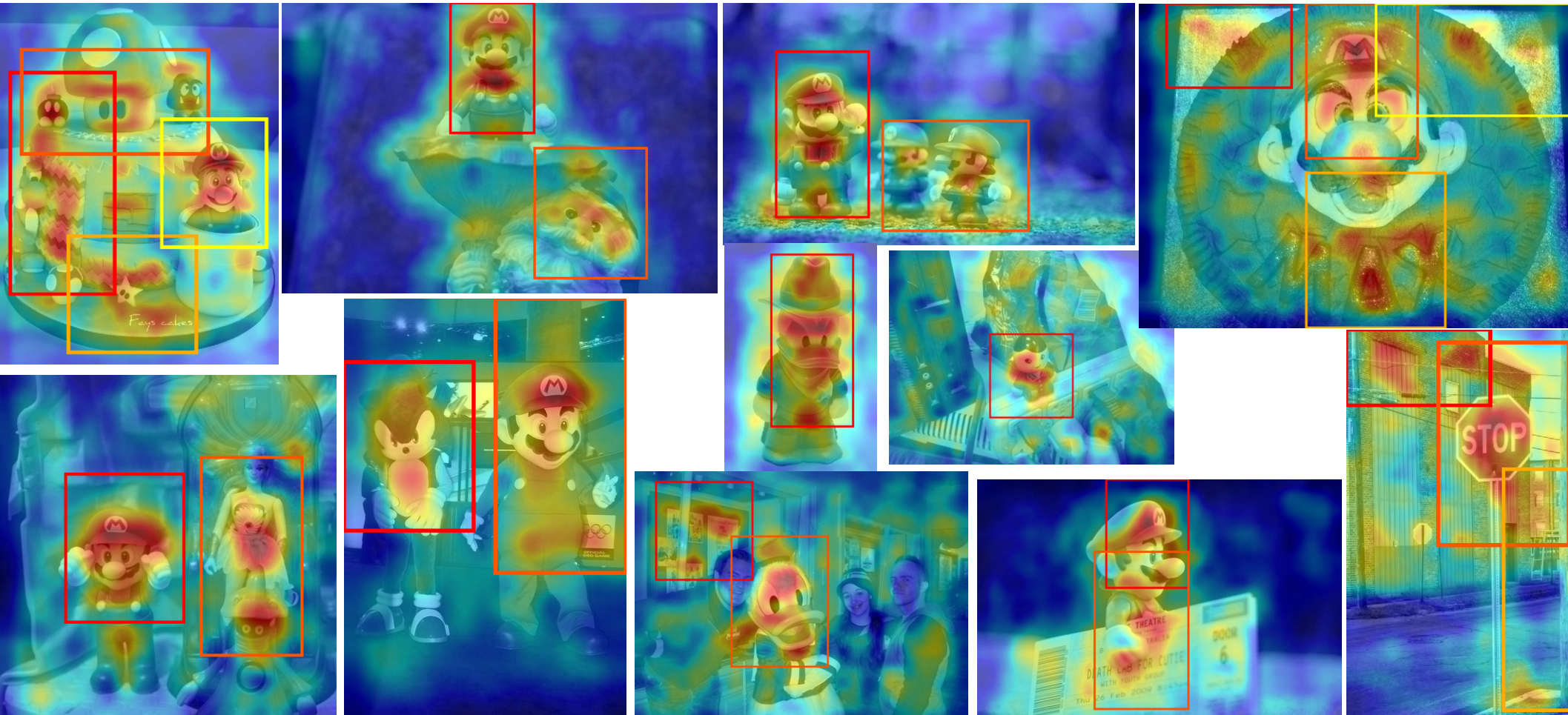
Database



Detect regions on saliency maps

- Use expanded version of Expanding Gaussian Mixture
  - *Avrithis et al. ECCV 2012* - to detect regions on saliency maps

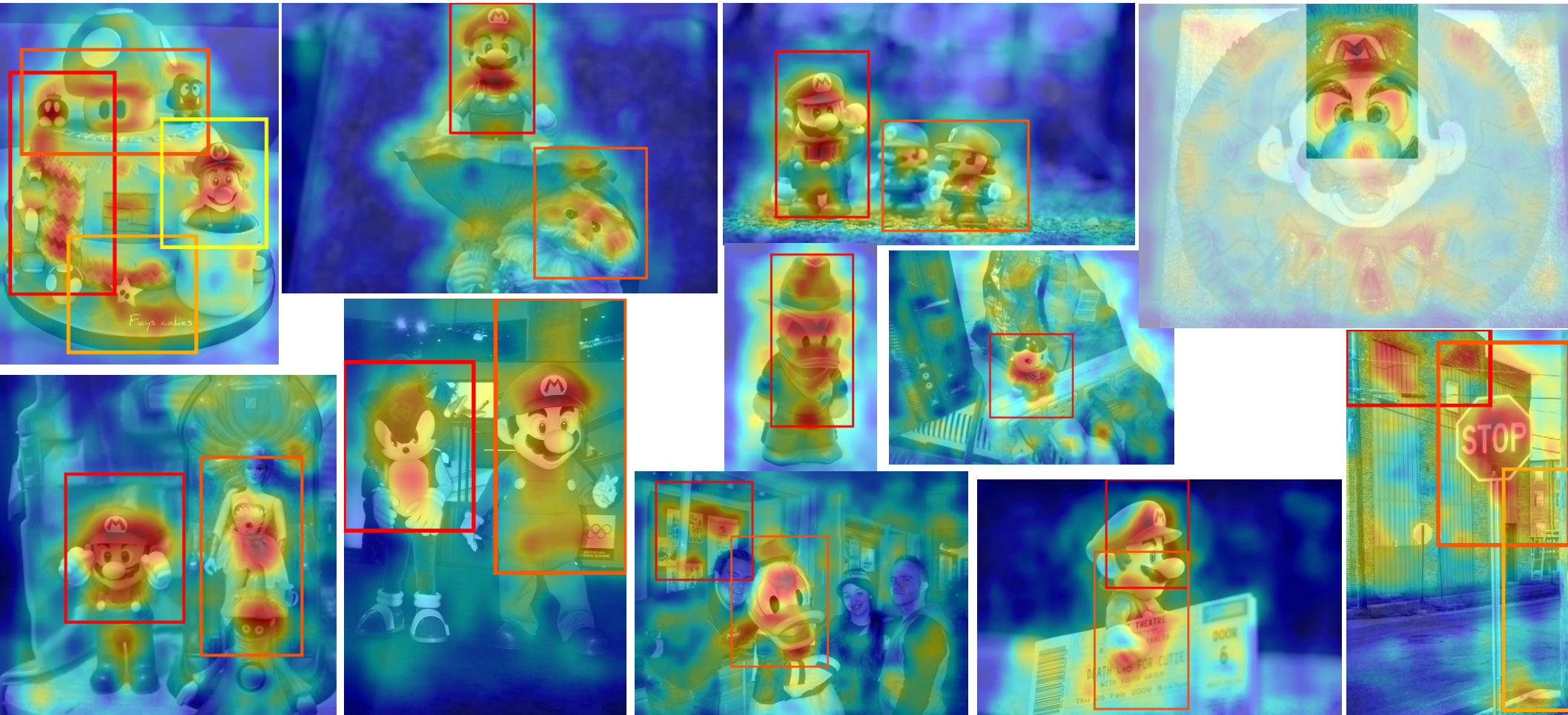
Database



Create a knn-graph

- Consider each detected region independently

Database



Create a knn-graph

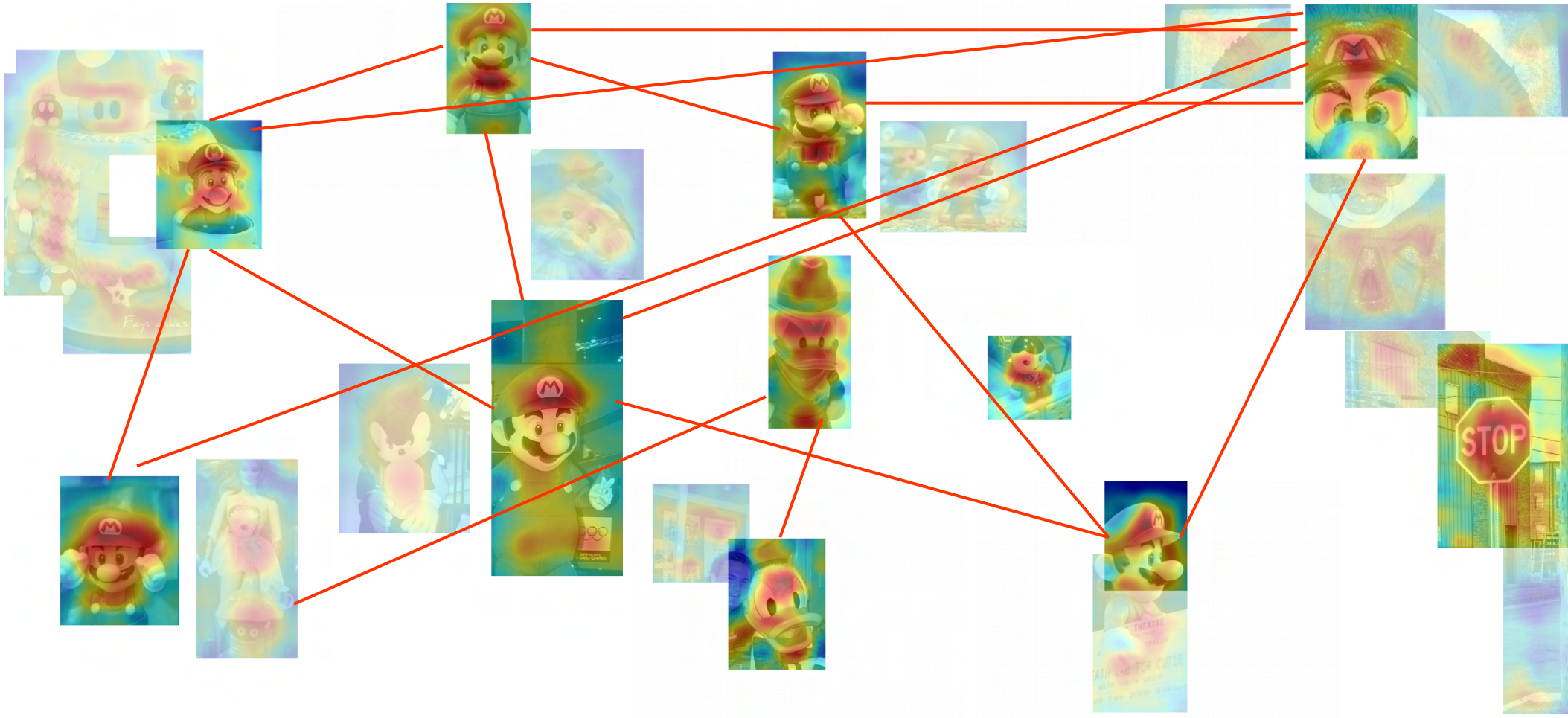
- Consider each detected region independently
  - Find its nearest neighbors

Database

Create a knn-graph

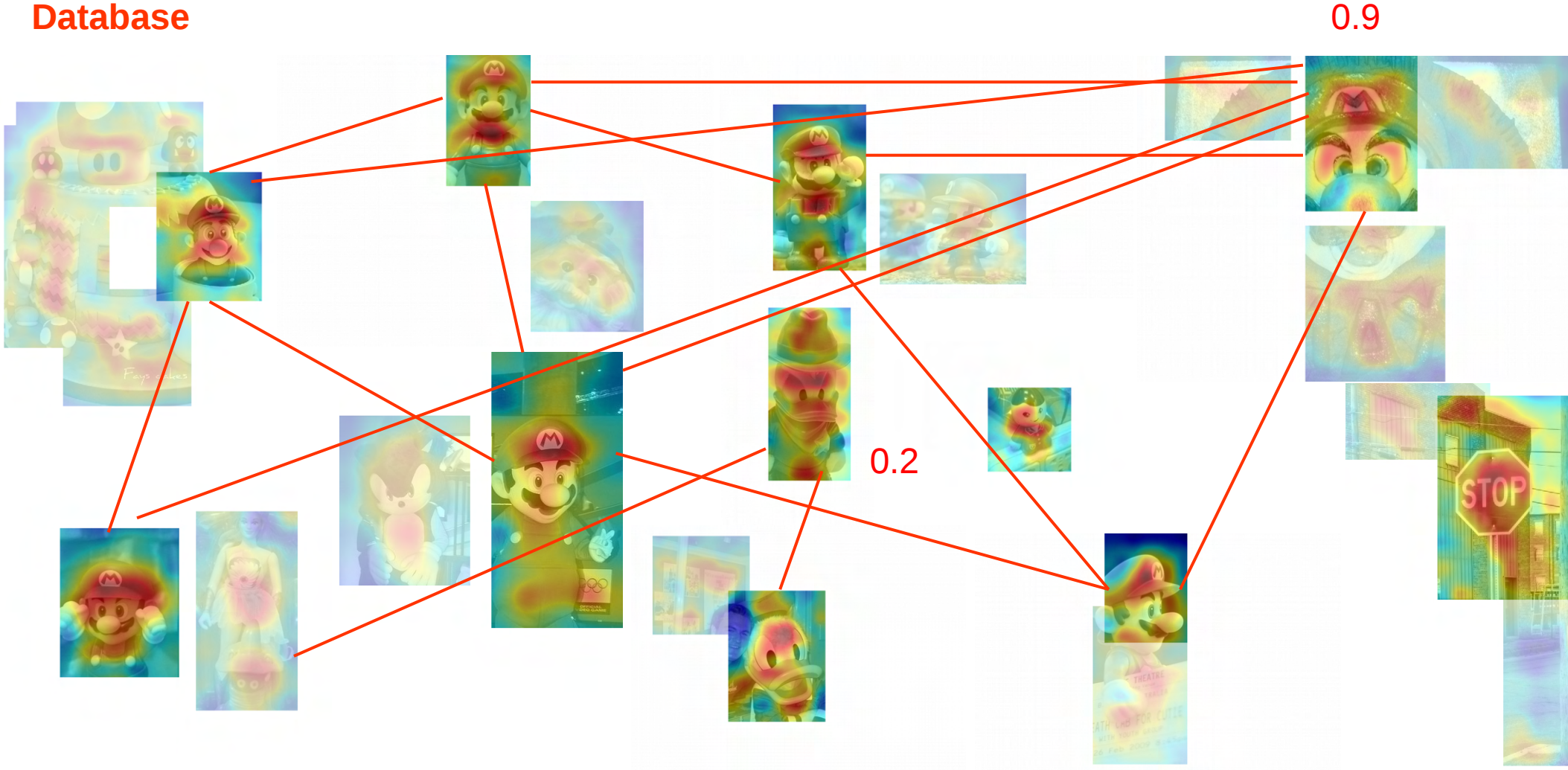
- Consider each detected region independently
  - Find its nearest neighbors
- Create the knn-graph

Database



Create a knn-graph

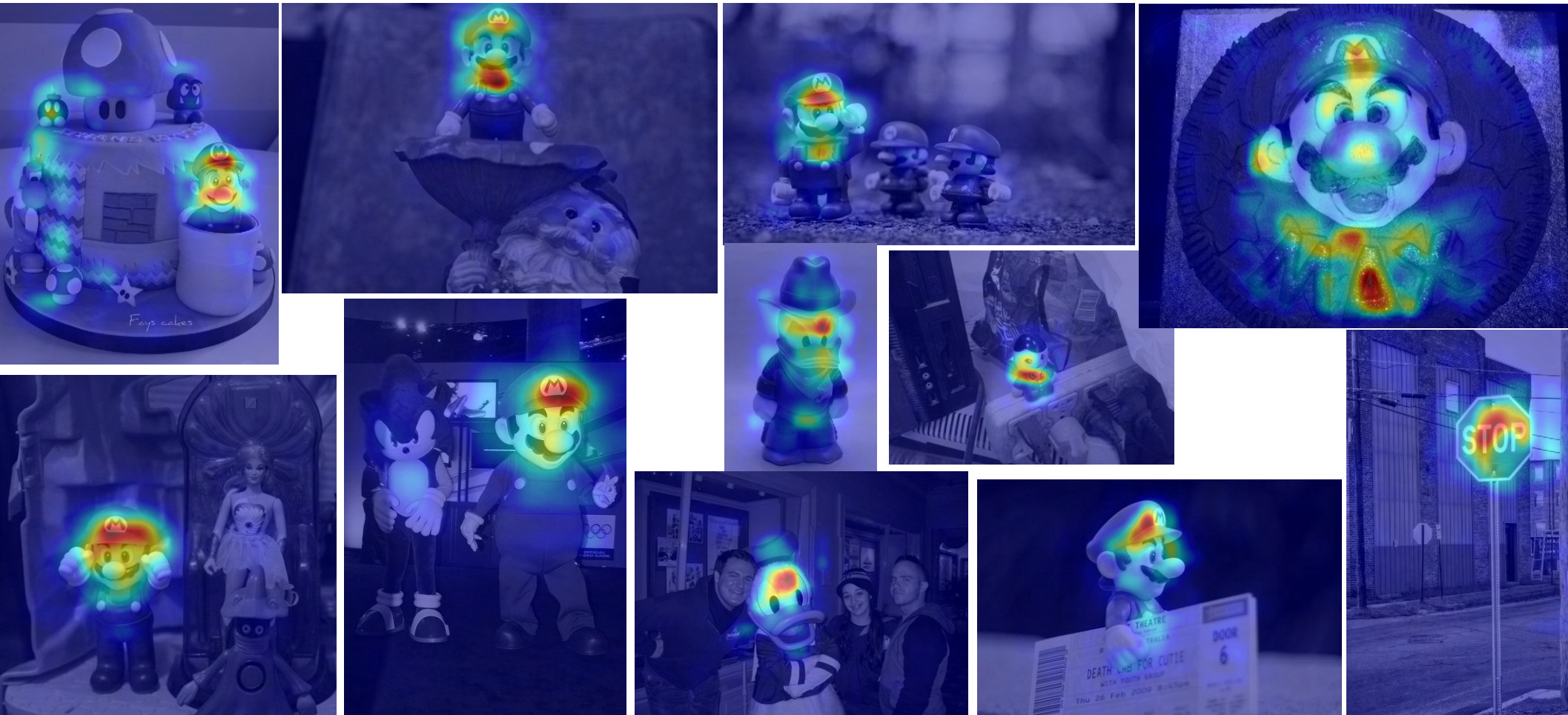
- Consider each detected region independently
  - Find its nearest neighbors
- Create the knn-graph
  - Compute katz centrality, it reflects a node importance



Compute Object Saliency

- For each patch of a sliding window :
  - Find nearest neighbors in the graph
  - Sum centrality of the neighbors

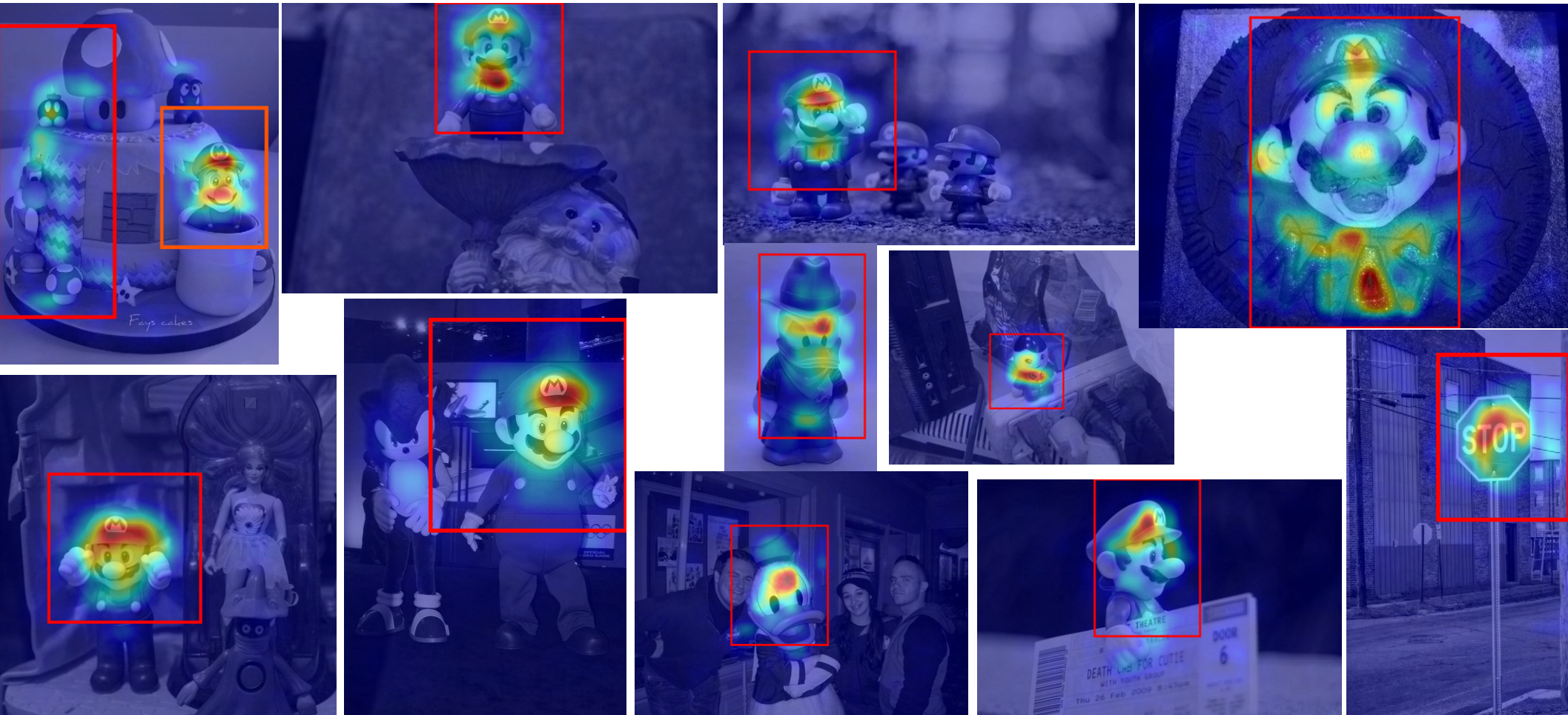
Database



Detect regions on the object saliency

- Use Expanding Gaussian Mixture on saliency maps to detect regions

Database

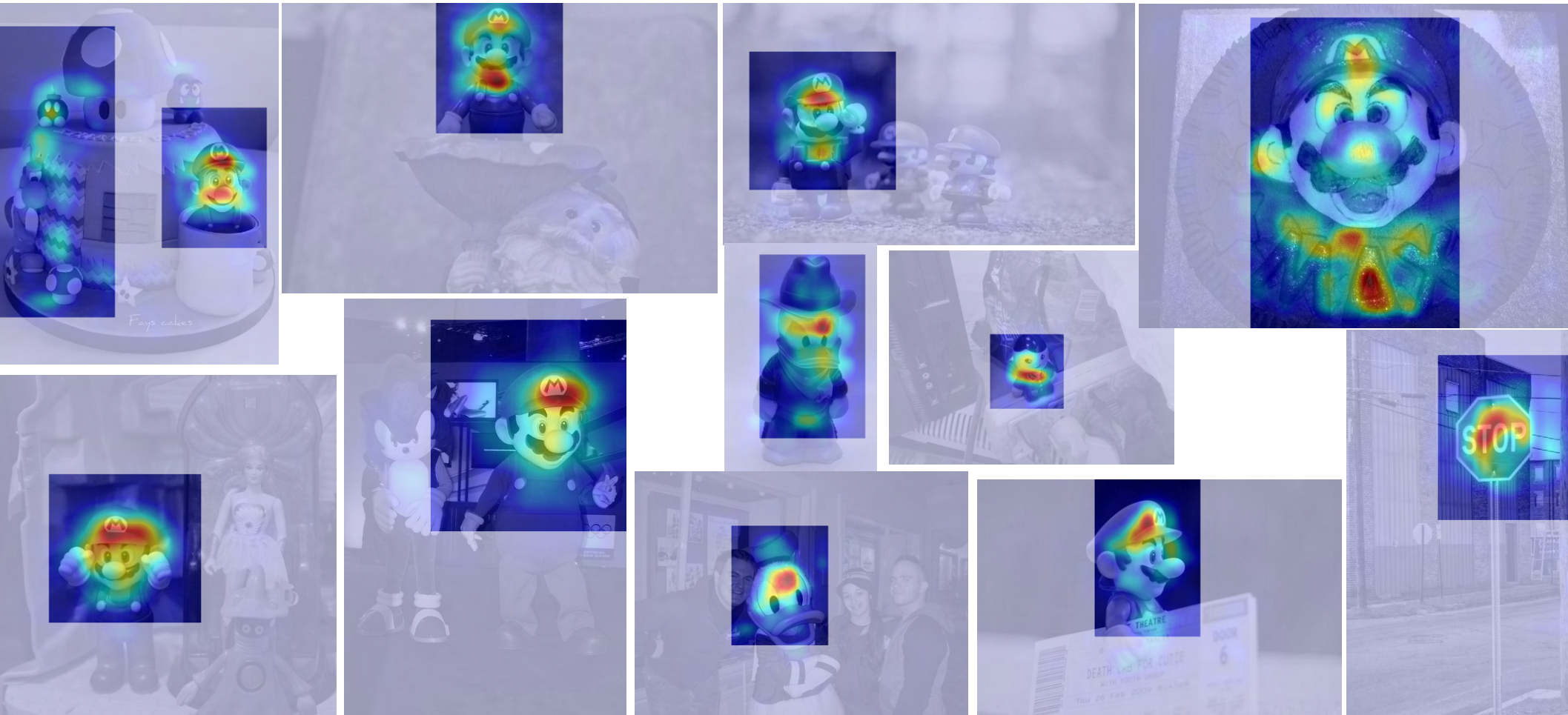


Detect regions on the object saliency

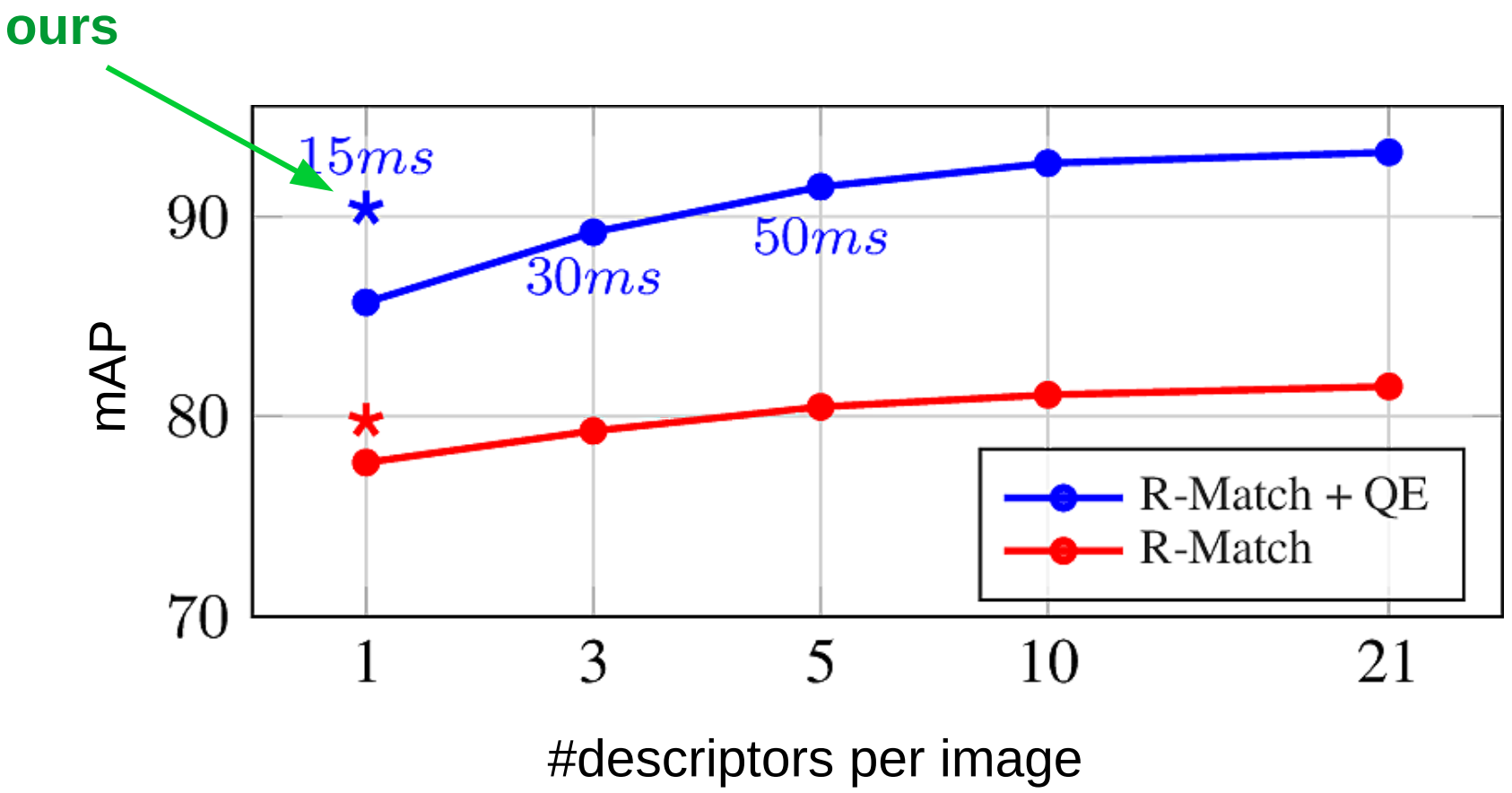
- Again, using Expanding Gaussian Mixture

Now we are able to find Mario everywhere !

Database

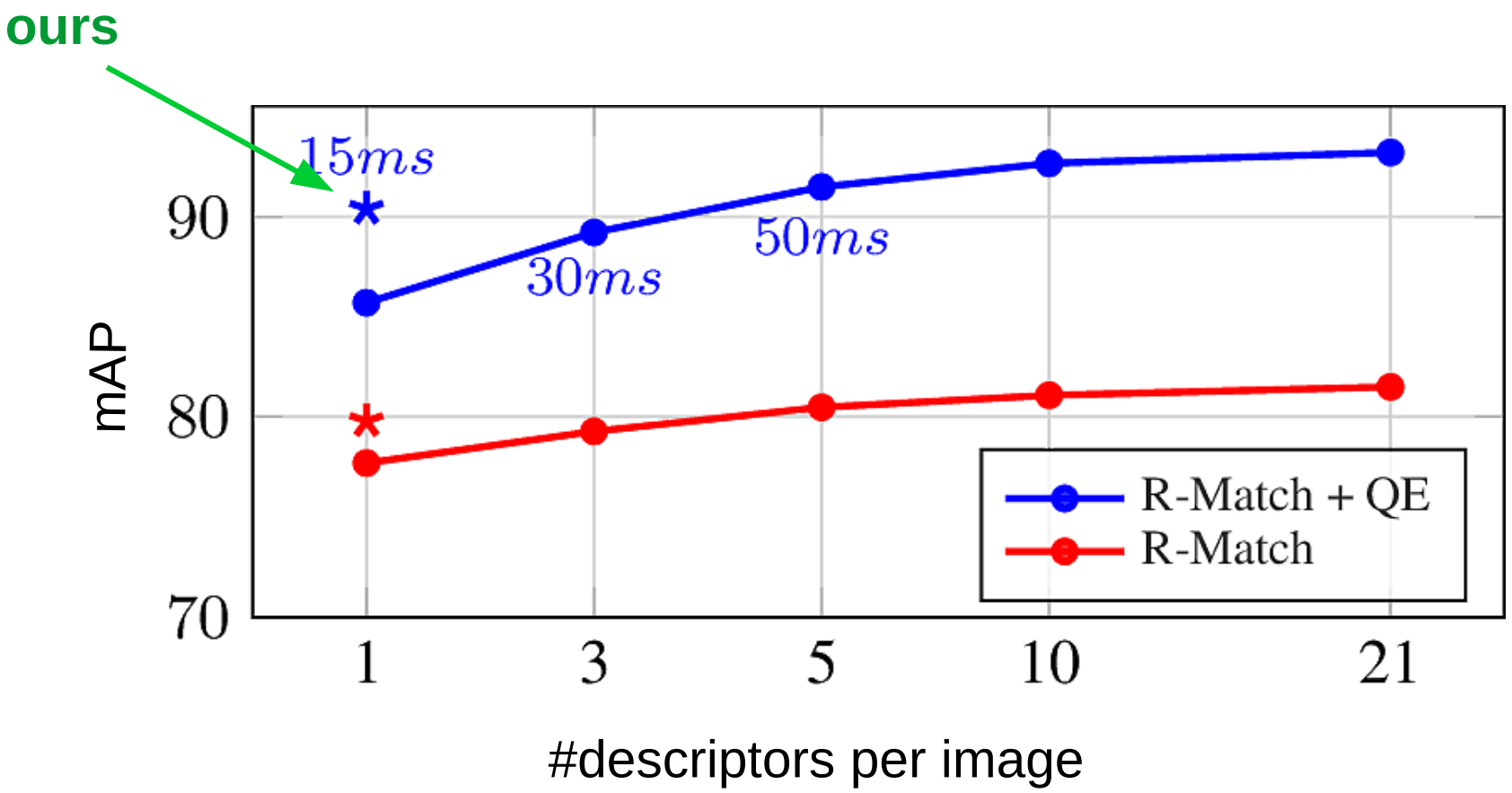


Results



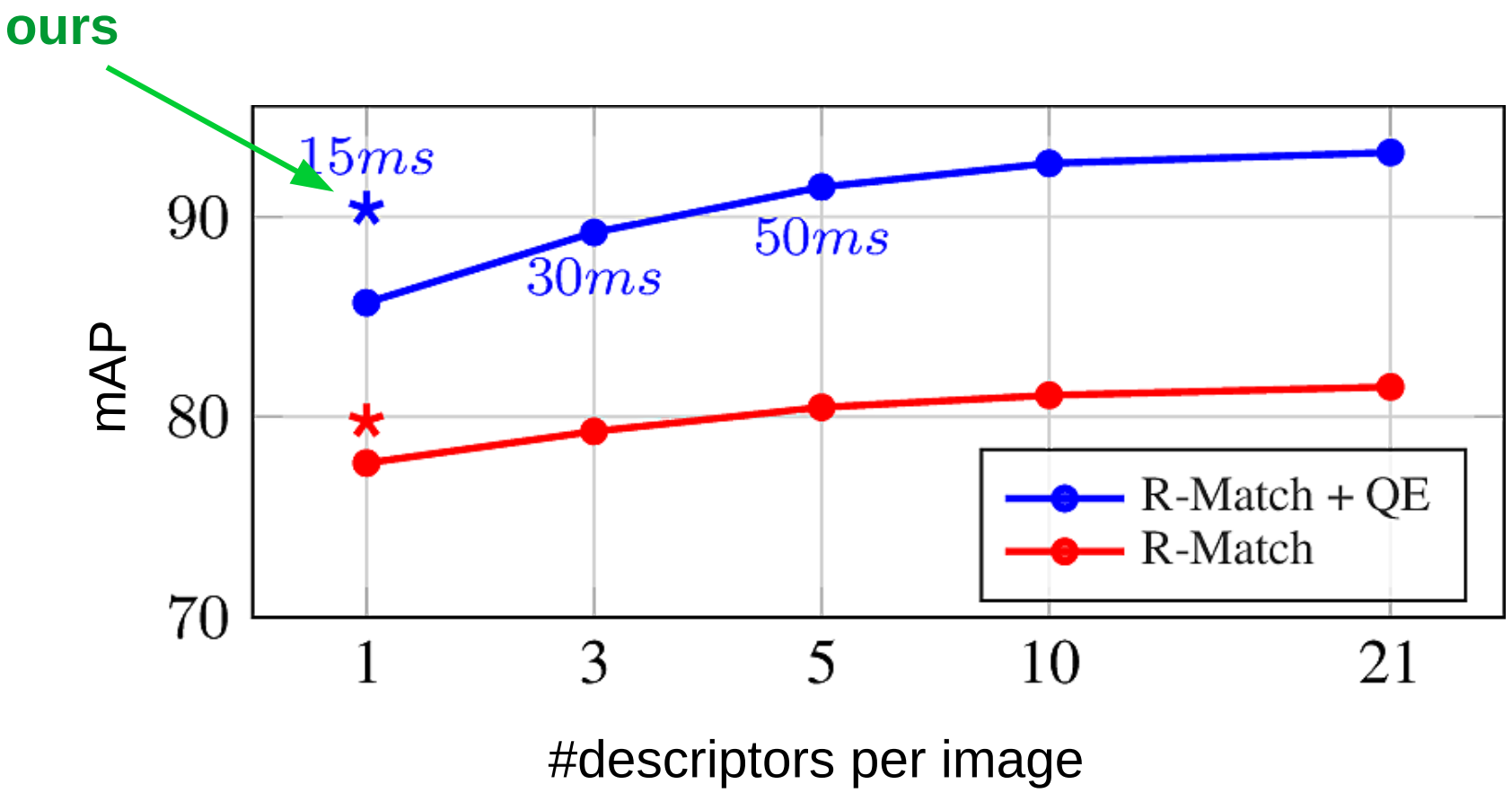
✔ **Same performance**

Results



- ✔ **Same performance**
- ✔ **3 time faster at query time**

Results



- ✔ **Same performance**
- ✔ **3 time faster at query time**
- ✔ **4 time less memory**

For details, please come to the **poster #XX**

Thank you for listening