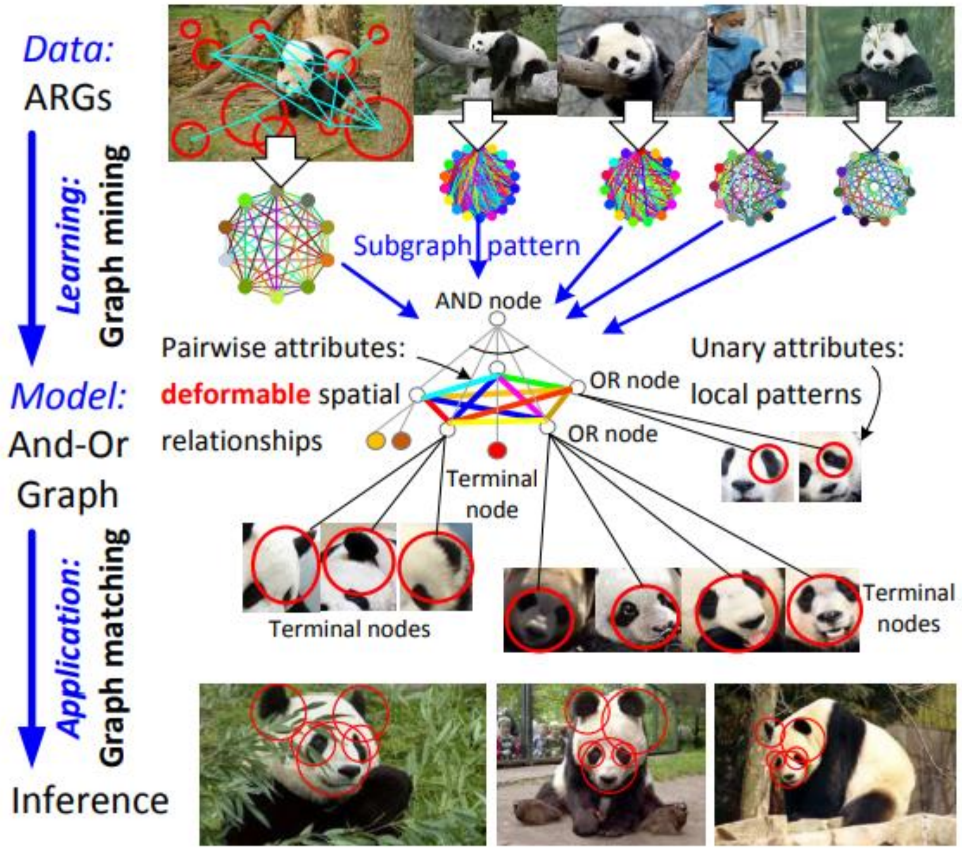
# Mining And-Or Graphs for Graph Matching and Object Discovery

Hogun Kee

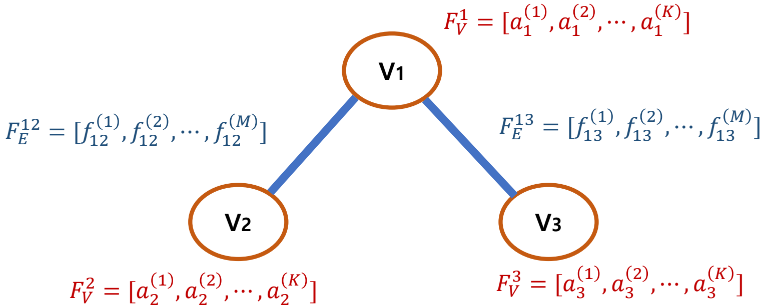
Seoul National University



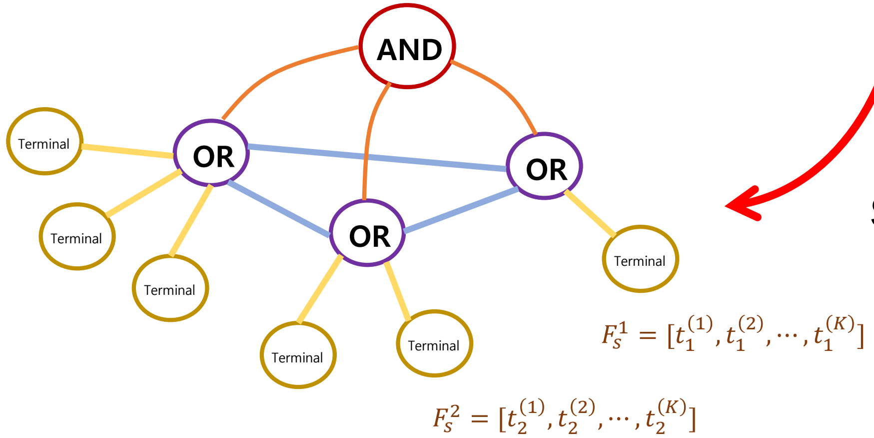

## Input Image

**Step 1.**

## ARG (Attributed Relational Graph)



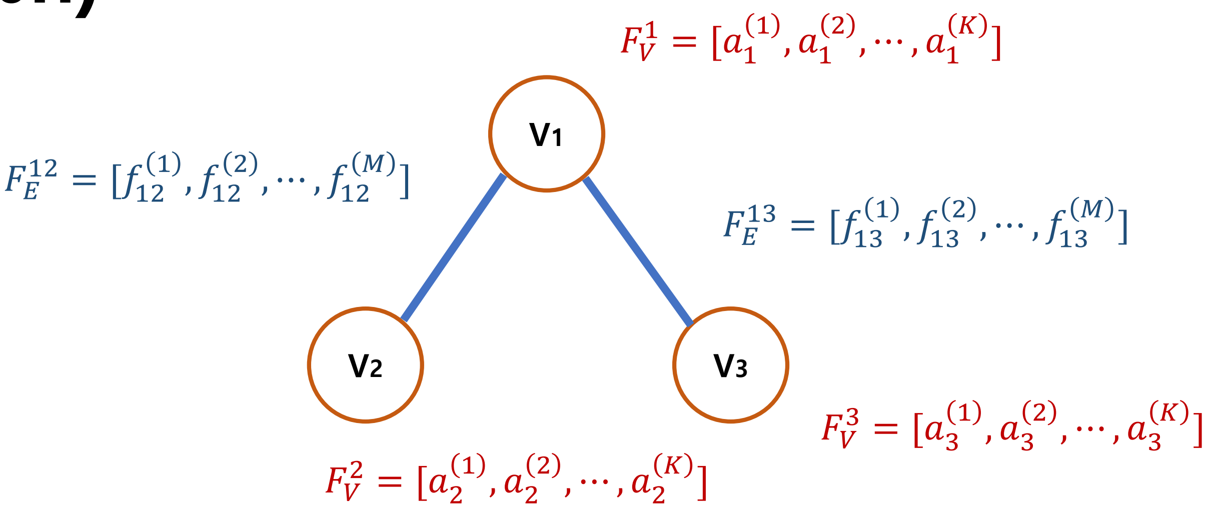

## AoG (And-Or Graph)




**Step 2.**

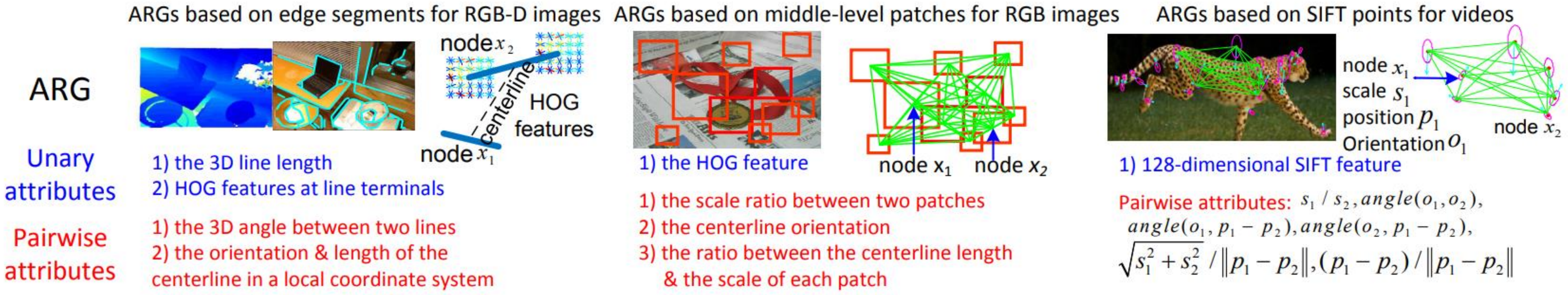
# ARG (Attributed Relational Graph)

- Each Node / Edge has a feature vector

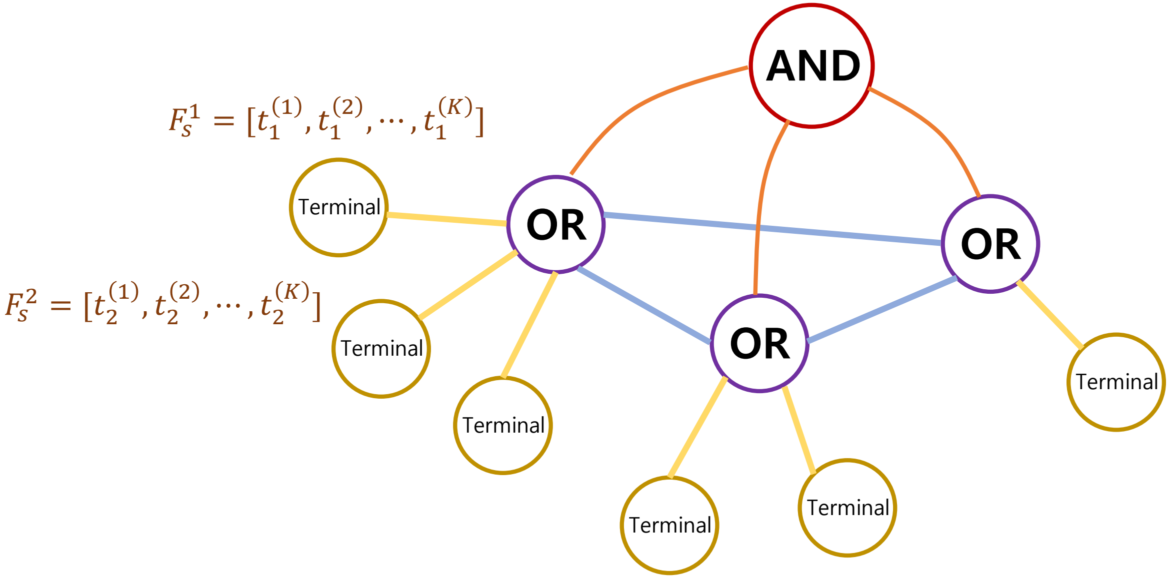


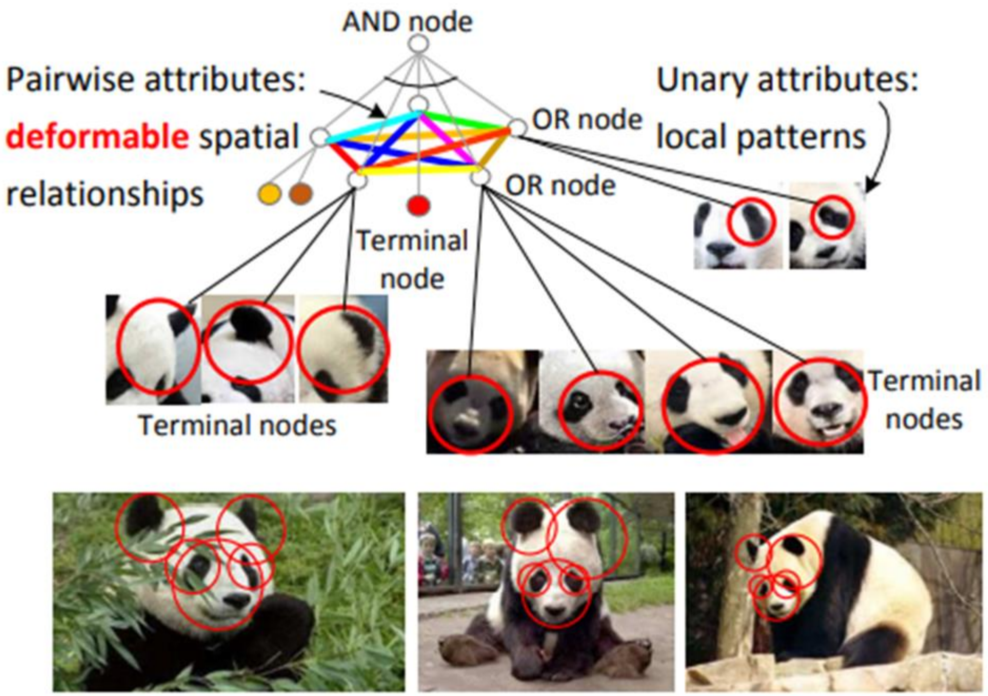

What is used as a NODE?

- RGB-D image: Line segment
- RGB image: Middle-level patch
- Video: SIFT points



# AoG (And-Or Graph)

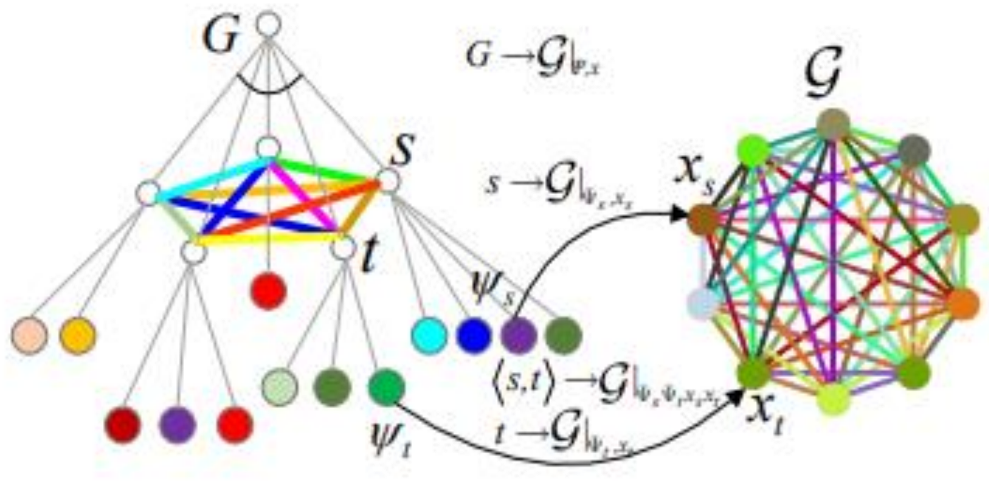



- OR node = Object part
- Terminal node = Local pattern of the part

# Graph Matching

**AoG (And-Or Graph)** **ARG**




- **Matching Probability / Energy**

$$P(G \mapsto \mathcal{G}) \propto \exp\left[-\mathcal{E}(G \mapsto \mathcal{G})\right]$$

$$\mathcal{E}(G \mapsto \mathcal{G}) = \sum\nolimits_{s \in V} \mathcal{E}(s \mapsto \mathcal{G}) + \sum\nolimits_{\langle s,t \rangle \in E} \mathcal{E}(\langle s,t \rangle \mapsto \mathcal{G})$$

- **Matching Energy per each Node / Edge**

$$\mathcal{E}(s \mapsto \mathcal{G}) = \begin{cases} \sum_{i=1}^{N_u} w_i^u \| F_i^{\psi_s} - \mathcal{F}_i^{x_s} \|^2, & x_s \in \mathcal{V} \\ u_{none}, & x_s = none \end{cases}$$

$$\mathcal{E}(\langle s,t \rangle \mapsto \mathcal{G}) = \begin{cases} \sum_{j=1}^{N_p} \frac{w_j^p \| F_j^{st} - \mathcal{F}_j^{x_s x_t} \|^2}{|V|-1}, & x_s \neq x_t \in \mathcal{V} \\ +\infty, & x_s = x_t \in \mathcal{V} \\ \frac{1}{|V|-1} p_{none}, & x_s \text{ or } x_t = none \end{cases}$$

# Graph Matching

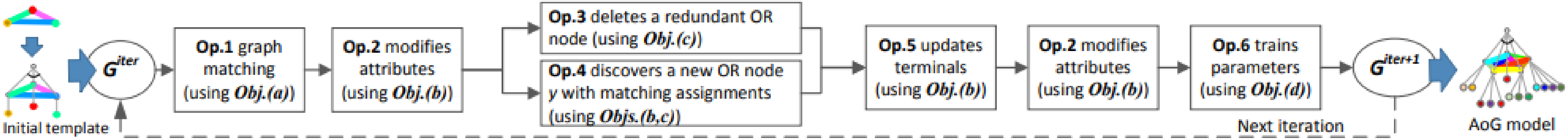



Figure 4. Flowchart of an approximate solution to graph mining

- **4 Objective functions**

**Obj.(a):** $\mathrm{argmin}_{\hat{\mathbf{x}}^k,\hat{\Psi}^k}\mathcal{E}(G\mapsto\mathcal{G}_k^+),\quad \mathrm{argmin}_{\tilde{\mathbf{x}}^l,\tilde{\Psi}^l}\mathcal{E}(G\mapsto\mathcal{G}_l^-)$

- Estimates the most probably object in each positive/negative image.

**Obj.(b):** $\mathrm{argmin}_{\mathbf{\Omega},\mathbf{F}_V,\mathbf{F}_E}\sum_{s\in V}(\mathcal{E}_s^+ - \mathcal{E}_s^- + \lambda|\Omega_s|)$

- Uniformly optimize its unary and pairwise attributes and the division of its terminal nodes.

**Obj.(c):** $\mathrm{argmax}|V| \quad \text{s.t. } \forall s\in V, \mathcal{E}_s^+ - \mathcal{E}_s^- + \lambda|\Omega_s| \le \tau$

- Extracting an AoG with the maximal number of OR nodes |V| by discovering new OR nodes and deleting redundant OR nodes.

**Obj.(d):** $\min_{\mathbf{w}}\|\mathbf{w}\|^2 + \frac{C}{N+}\sum_{k=1}^{N+}\xi_k^+ + \frac{C}{N-}\sum_{l=1}^{N^-}\xi_l^-,$

$\forall k = 1,2,...,N^+, -[\mathcal{E}(G\mapsto\mathcal{G}_k^+)+b]\ge 1-\xi_k^+,$

$\forall l = 1,2,...,N^-, \mathcal{E}(G\mapsto\mathcal{G}_l^-)+b\ge 1-\xi_l^-$

- Use a linear SVM to train the matching parameters W, which is an approximate solution to the minimization of the generative loss.

# Graph Matching

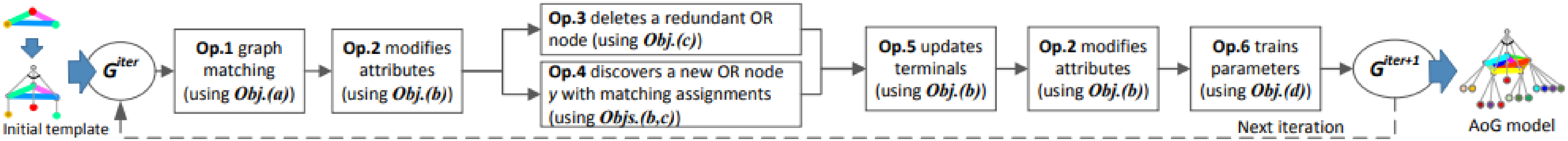



Figure 4. Flowchart of an approximate solution to graph mining

- **4 Objective functions**

$$\boxed{\textbf{Obj.(a):}\ \ \arg\min_{\hat{\mathbf{x}}^k,\hat{\Psi}^k}\mathcal{E}(G\mapsto\mathcal{G}_k^+),\ \ \arg\min_{\hat{\mathbf{x}}^l,\hat{\Psi}^l}\mathcal{E}(G\mapsto\mathcal{G}_l^-)}$$

- Estimates the most probably object in each positive/negative image.

$$\textbf{Obj.(b):}\ \ \arg\min_{\Omega,\mathbf{F}_V,\mathbf{F}_E}\sum_{s\in V}(\mathcal{E}_s^+ - \mathcal{E}_s^- + \lambda|\Omega_s|)$$

- Uniformly optimize its unary and pairwise attributes and the division of its terminal nodes.

$$\textbf{Obj.(c):}\ \ \arg\max|V|\quad \text{s.t. } \forall s\in V, \mathcal{E}_s^+ - \mathcal{E}_s^- + \lambda|\Omega_s| \le \tau$$

- Extracting an AoG with the maximal number of OR nodes |V| by discovering new OR nodes and deleting redundant OR nodes.

$$\textbf{Obj.(d):}\ \ \min_{\mathbf{w}}\|\mathbf{w}\|^2 + \frac{C}{N^+}\sum_{k=1}^{N^+}\xi_k^+ + \frac{C}{N^-}\sum_{l=1}^{N^-}\xi_l^-,$$
$$\forall k = 1,2,...,N^+, -[\mathcal{E}(G\mapsto\mathcal{G}_k^+) + b] \ge 1 - \xi_k^+,$$
$$\forall l = 1,2,...,N^-, \mathcal{E}(G\mapsto\mathcal{G}_l^-) + b \ge 1 - \xi_l^-$$

- Use a linear SVM to train the matching parameters W, which is an approximate solution to the minimization of the generative loss.

# Graph Matching

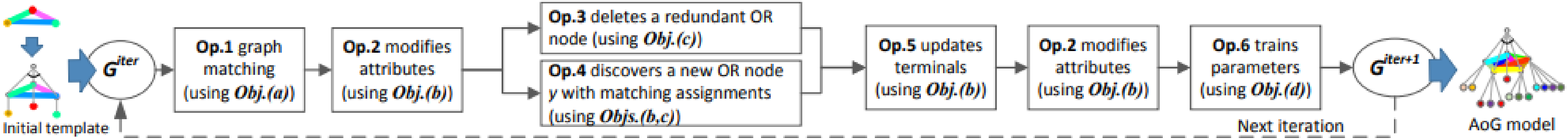



Figure 4. Flowchart of an approximate solution to graph mining

- **4 Objective functions**

**Obj.(a):** $\operatorname{argmin}_{\hat{\mathbf{x}}^k,\hat{\Psi}^k}\mathcal{E}(G\mapsto\mathcal{G}_k^+),\quad \operatorname{argmin}_{\tilde{\mathbf{x}}^l,\tilde{\Psi}^l}\mathcal{E}(G\mapsto\mathcal{G}_l^-)$

- Estimates the most probably object in each positive/negative image.

**Obj.(b):** $\operatorname{argmin}_{\mathbf{\Omega},\mathbf{F}_V,\mathbf{F}_E}\sum_{s\in V}(\mathcal{E}_s^+ - \mathcal{E}_s^- + \lambda|\Omega_s|)$

- Uniformly optimize its unary and pairwise attributes and the division of its terminal nodes.

**Obj.(c):** $\operatorname{argmax}|V| \quad \text{s.t. } \forall s\in V, \mathcal{E}_s^+ - \mathcal{E}_s^- + \lambda|\Omega_s| \le \tau$

- Extracting an AoG with the maximal number of OR nodes |V| by discovering new OR nodes and deleting redundant OR nodes.

**Obj.(d):**
$$\min_{\mathbf{w}}\|\mathbf{w}\|^2 + \frac{C}{N^+}\sum_{k=1}^{N^+}\xi_k^+ + \frac{C}{N^-}\sum_{l=1}^{N^-}\xi_l^-,$$
$$\forall k=1,2,...,N^+, -[\mathcal{E}(G\mapsto\mathcal{G}_k^+)+b]\ge 1-\xi_k^+,$$
$$\forall l=1,2,...,N^-, \mathcal{E}(G\mapsto\mathcal{G}_l^-)+b\ge 1-\xi_l^-$$

- Use a linear SVM to train the matching parameters W, which is an approximate solution to the minimization of the generative loss.

# Graph Matching

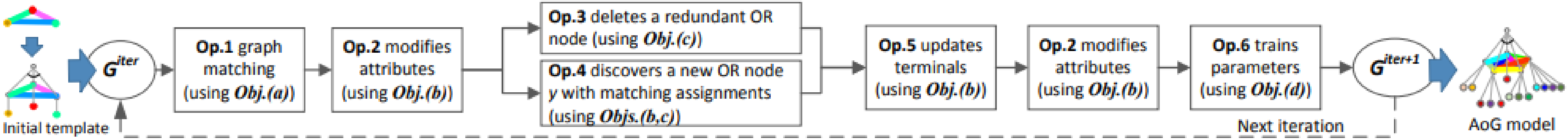



Figure 4. Flowchart of an approximate solution to graph mining

- **4 Objective functions**

**Obj.(a):** $\mathrm{argmin}_{\hat{\mathbf{x}}^k,\hat{\Psi}^k}\mathcal{E}(G\mapsto\mathcal{G}_k^+),\quad \mathrm{argmin}_{\tilde{\mathbf{x}}^l,\tilde{\Psi}^l}\mathcal{E}(G\mapsto\mathcal{G}_l^-)$

- Estimates the most probably object in each positive/negative image.

**Obj.(b):** $\mathrm{argmin}_{\mathbf{\Omega},\mathbf{F}_V,\mathbf{F}_E}\sum_{s\in V}(\mathcal{E}_s^+ - \mathcal{E}_s^- + \lambda|\Omega_s|)$

- Uniformly optimize its unary and pairwise attributes and the division of its terminal nodes.

$\boxed{\textbf{Obj.(c):}\quad \mathrm{argmax}|V| \quad \text{s.t. } \forall s\in V, \mathcal{E}_s^+ - \mathcal{E}_s^- + \lambda|\Omega_s| \le \tau}$

- Extracting an AoG with the maximal number of OR nodes |V| by discovering new OR nodes and deleting redundant OR nodes.

**Obj.(d):** $\min_{\mathbf{w}}\|\mathbf{w}\|^2 + \frac{C}{N^+}\sum_{k=1}^{N^+}\xi_k^+ + \frac{C}{N^-}\sum_{l=1}^{N^-}\xi_l^-,$

$\forall k = 1,2,...,N^+, -[\mathcal{E}(G\mapsto\mathcal{G}_k^+) + b] \ge 1 - \xi_k^+,$

$\forall l = 1,2,...,N^-, \mathcal{E}(G\mapsto\mathcal{G}_l^-) + b \ge 1 - \xi_l^-$

- Use a linear SVM to train the matching parameters W, which is an approximate solution to the minimization of the generative loss.

# Graph Matching

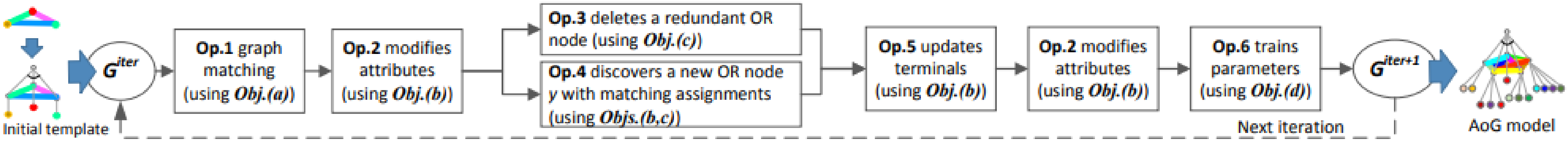



Figure 4. Flowchart of an approximate solution to graph mining

- **4 Objective functions**

**Obj.(a):** $\text{argmin}_{\hat{\mathbf{x}}^k,\hat{\Psi}^k}\mathcal{E}(G\mapsto\mathcal{G}_k^+)$, $\text{argmin}_{\tilde{\mathbf{x}}^l,\tilde{\Psi}^l}\mathcal{E}(G\mapsto\mathcal{G}_l^-)$

- Estimates the most probably object in each positive/negative image.

**Obj.(b):** $\text{argmin}_{\Omega,\mathbf{F}_V,\mathbf{F}_E}\sum_{s\in V}(\mathcal{E}_s^+ - \mathcal{E}_s^- + \lambda|\Omega_s|)$

- Uniformly optimize its unary and pairwise attributes and the division of its terminal nodes.

**Obj.(c):** $\text{argmax}|V| \quad \text{s.t.}\ \forall s\in V, \mathcal{E}_s^+ - \mathcal{E}_s^- + \lambda|\Omega_s| \le \tau$

- Extracting an AoG with the maximal number of OR nodes |V| by discovering new OR nodes and deleting redundant OR nodes.

**Obj.(d):** $\min_{\mathbf{w}}\|\mathbf{w}\|^2 + \frac{C}{N^+}\sum_{k=1}^{N^+}\xi_k^+ + \frac{C}{N^-}\sum_{l=1}^{N^-}\xi_l^-,$
$\forall k = 1,2,...,N^+, -[\mathcal{E}(G\mapsto\mathcal{G}_k^+) + b] \ge 1-\xi_k^+,$
$\forall l = 1,2,...,N^-, \mathcal{E}(G\mapsto\mathcal{G}_l^-) + b \ge 1-\xi_l^-$

- Use a linear SVM to train the matching parameters W, which is an approximate solution to the minimization of the generative loss.

# Graph Matching

**Operation 1, graph matching**

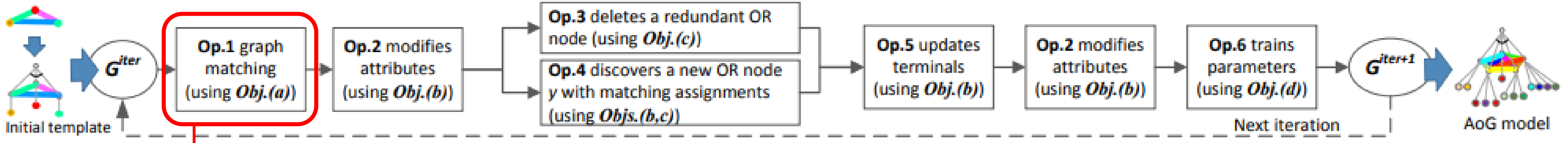



Figure 4. Flowchart of an approximate solution to graph mining

**Obj.(a):** $\mathrm{argmin}_{\hat{\mathbf{x}}^k,\hat{\Psi}^k}\mathcal{E}(G\rightarrow\mathcal{G}_k^+),\quad \mathrm{argmin}_{\check{\mathbf{x}}^l,\check{\Psi}^l}\mathcal{E}(G\rightarrow\mathcal{G}_l^-)$

# Graph Matching

**Operation 2, attribute estimation**

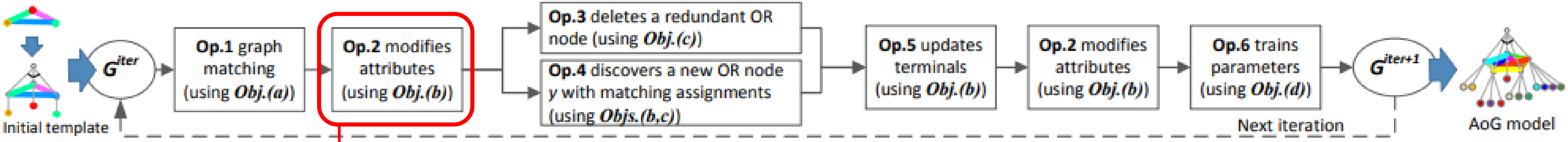



Figure 4. Flowchart of an approximate solution to graph mining

**Obj.(b):** $\operatorname{argmin}_{\mathbf{\Omega},\mathbf{F}_V,\mathbf{F}_E} \sum_{s \in V} (\mathcal{E}_s^+ - \mathcal{E}_s^- + \lambda |\Omega_s|)$

# Graph Matching

**Operation 3, node elimination**

**Operation 4, node discovery**

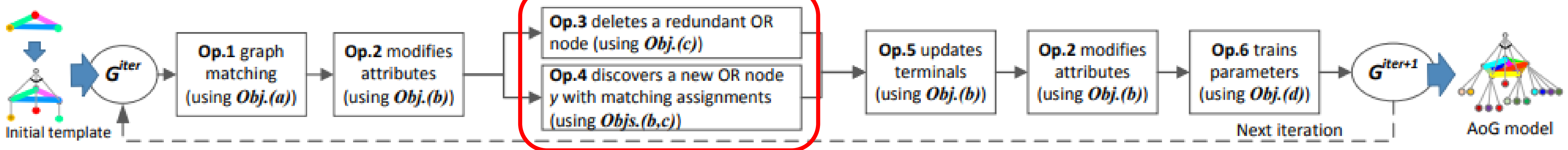



Figure 4. Flowchart of an approximate solution to graph mining

**Obj.(b):** $\operatorname{argmin}_{\mathbf{\Omega},\mathbf{F}_V,\mathbf{F}_E}\sum_{s\in V}(\mathcal{E}_s^+ - \mathcal{E}_s^- + \lambda|\Omega_s|)$

**Obj.(c):** $\operatorname{argmax}|V| \quad \text{s.t. } \forall s\in V, \mathcal{E}_s^+ - \mathcal{E}_s^- + \lambda|\Omega_s| \le \tau$

# Graph Matching

Operation 5, terminal determination:

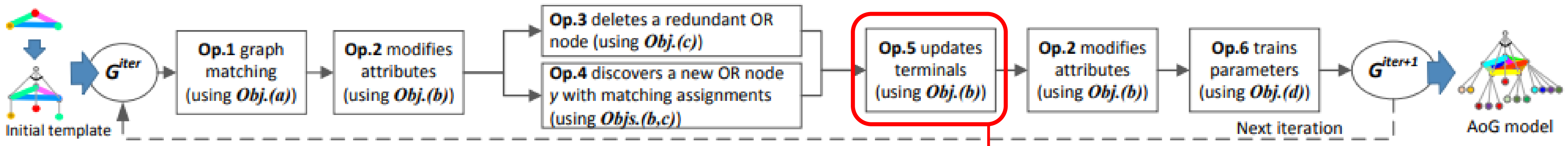



Figure 4. Flowchart of an approximate solution to graph mining

**Obj.(b):** $\text{argmin}_{\mathbf{\Omega},\mathbf{F}_V,\mathbf{F}_E} \sum_{s \in V} (\mathcal{E}_s^+ - \mathcal{E}_s^- + \lambda|\Omega_s|)$

# Graph Matching

Operation 2, attribute estimation

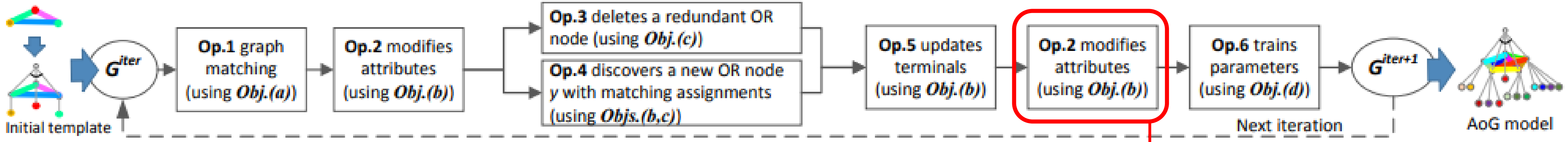



Figure 4. Flowchart of an approximate solution to graph mining

**Obj.(b):** $\mathrm{argmin}_{\mathbf{\Omega},\mathbf{F}_V,\mathbf{F}_E} \sum_{s \in V} (\mathcal{E}_s^+ - \mathcal{E}_s^- + \lambda |\Omega_s|)$

# Graph Matching

Operation 6, parameter training

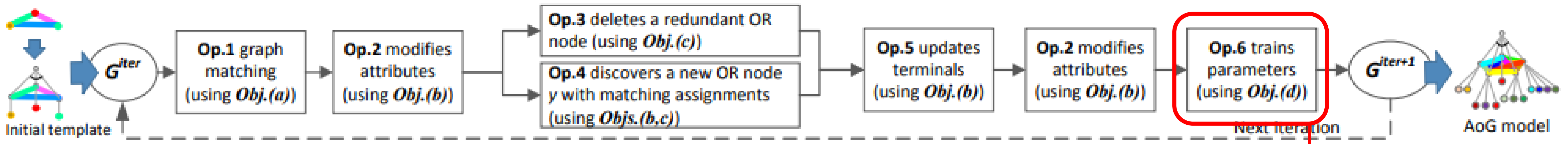



Figure 4. Flowchart of an approximate solution to graph mining

**Obj.(d):** $\min_{\mathbf{w}} \|\mathbf{w}\|^2 + \frac{C}{N+} \sum_{k=1}^{N^+} \xi_k^+ + \frac{C}{N^-} \sum_{l=1}^{N^-} \xi_l^-,$

$\forall k = 1, 2, ..., N^+, -[\mathcal{E}(G \mapsto \mathcal{G}_k^+) + b] \geq 1 - \xi_k^+,$

$\forall l = 1, 2, ..., N^-, \mathcal{E}(G \mapsto \mathcal{G}_l^-) + b \geq 1 - \xi_l^-$

# Graph Mining

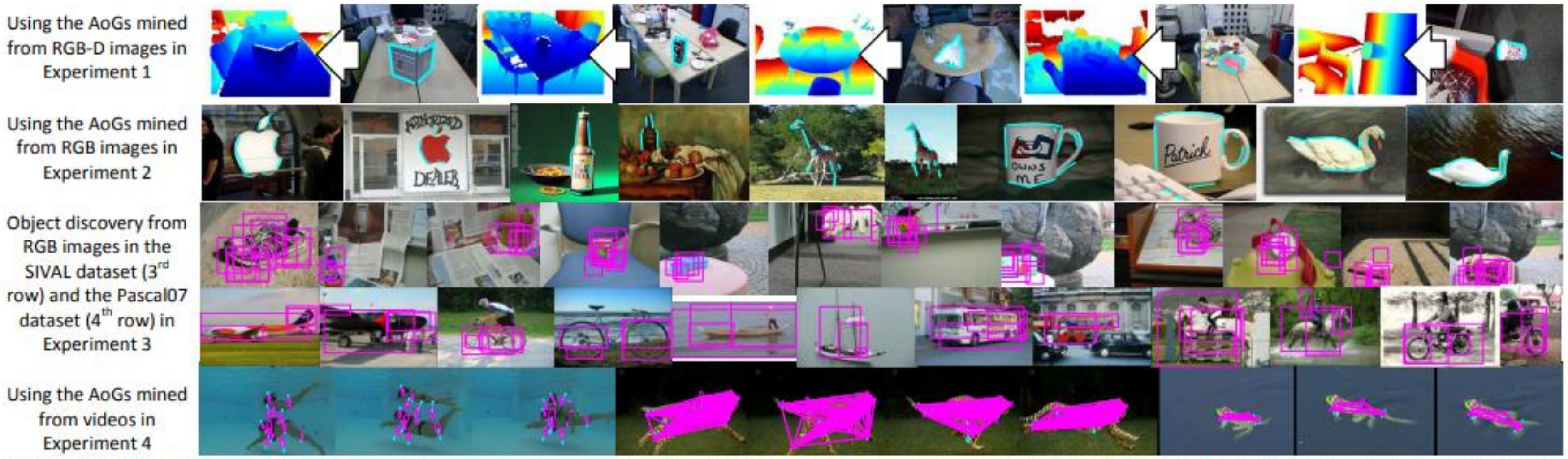



Figure 9. Matching results based on AoGs. We draw edge connections of the frog and cheetah models to show their structure deformation.

| | Ours | bMCL | SD | $M^3IC$ | BAMIC | UnSL | MFC |
|---|---|---|---|---|---|---|---|
| SIVAL1 | 89.0 | **95.3** | 80.4 | 39.3 | 38.0 | 27.0 | 45.0 |
| SIVAL2 | **93.2** | 84.0 | 71.7 | 40.0 | 33.3 | 35.3 | 33.3 |
| SIVAL3 | **88.4** | 74.7 | 62.7 | 37.3 | 38.7 | 26.7 | 41.3 |
| SIVAL4 | 87.8 | **94.0** | 86.0 | 33.0 | 37.7 | 27.3 | 53.0 |
| SIVAL5 | **92.7** | 75.3 | 70.3 | 35.3 | 37.7 | 25.0 | 48.3 |
| **Average** | **90.2** | 84.7 | 74.2 | 37.0 | 37.1 | 28.3 | 44.2 |

Table 1. Average purity of object discovery in the SIVAL dataset

| Pose | | aero. | bicy. | boat | bus | horse | moto. | **Avg.** |
|---|---|---|---|---|---|---|---|---|
| Left | MA | 16.1 | 18.5 | 10.6 | 42.9 | 11.3 | 27.7 | 21.3 |
| | Ours | 73.2 | 64.6 | 29.8 | 71.4 | 58.1 | 80.9 | 63.2 |
| Right | MA | 13.5 | 10.7 | 17.3 | 57.9 | 8.20 | 20.5 | |
| | Ours | 75.0 | 66.1 | 42.3 | 73.7 | 45.9 | 77.3 | |

Table 2. Average localization rate in PASCAL07-6×2 dataset

# Conclusion

## Advantages

- The AoG represents a hierarchical deformable template that has strong expressive power in modeling objects.
- The AoG can be mined without labeling object positions.
- We do not use sliding windows to enumerate object candidates for model mining.

## Critique

Rule-based ARG construction → Can't we switch it to another learning technique?

# Code Implementation - 구현된 코드 사용

- 작동순서

1. Code compile
2. Feature extraction
3. Model producing
4. Mining AoG

**compile**();

- Matlab 코드들을 compile한다.

**featureExtraction**(카테고리이름);

- Data path에 저장된 해당 카테고리의 이미지들에 대해 feature extraction을 진행한다.
- 실행결과로 mat/폴더에 RGs_category.mat, ARGs_net_category.mat, Clu_category.mat, FinalPatches_category.mat, FinalPatches_neg_category.mat, Final_PatchSet_category.mat, Kmeans_category.mat, PatchSet_category.mat, PatchSet_neg_category.mat 파일이 생성된다.

**produce_model**(카테고리이름);

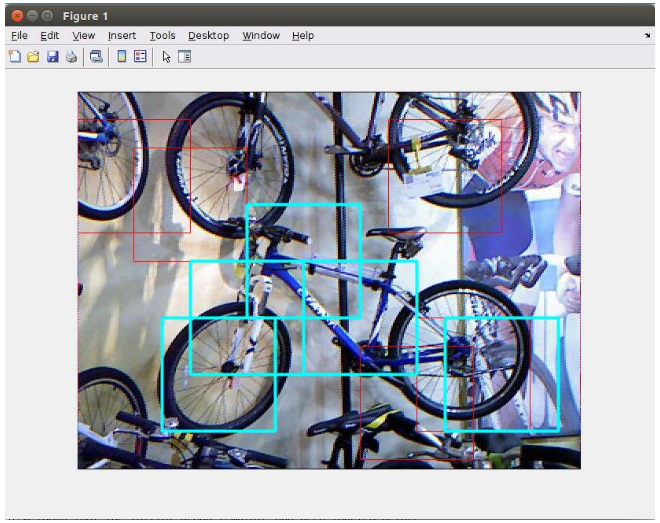
- 초기 graph template을 라벨링해준다. 프로그램이 사진들을 띄워주면, 라벨링할 사진에서 'y'를 입력하고 마우스 드래그로 정사각형 box를 그려 object의 서로 다른 부위를 3~5개 정도 잡아준다.
- 지정해준 initial template과 feature extraction의 결과들을 바탕으로 ARG 그래프를 생성한다.

**main_aog**(카테고리이름);

- Graph matching을 통해 위에서 생성한 ARG들에 대한 AoG를 생성하고, 이 AoG를 바탕으로 Test data에 대해 Object discovery를 시행한다.

# Experiment Results – Object discovery

## Category: Bicycle




Initial template

# Code Links

- My github page

https://github.com/hogunkee/AndOrGraph.git

- Original project page

https://sites.google.com/site/quanshizhang/code_aog

- Original code download link

https://onedrive.live.com/?authkey=%21APR5TLc1r5vnFFk&cid=BBDCC5334B0B1B33&id=BBDCC5334B0B1B33%21203&parId=BBDCC5334B0B1B33%21174&action=locate