Learning Graphs to Match

Inria Paris – WILLOW

Minsu Cho

Karteek Alahari

Jean Ponce

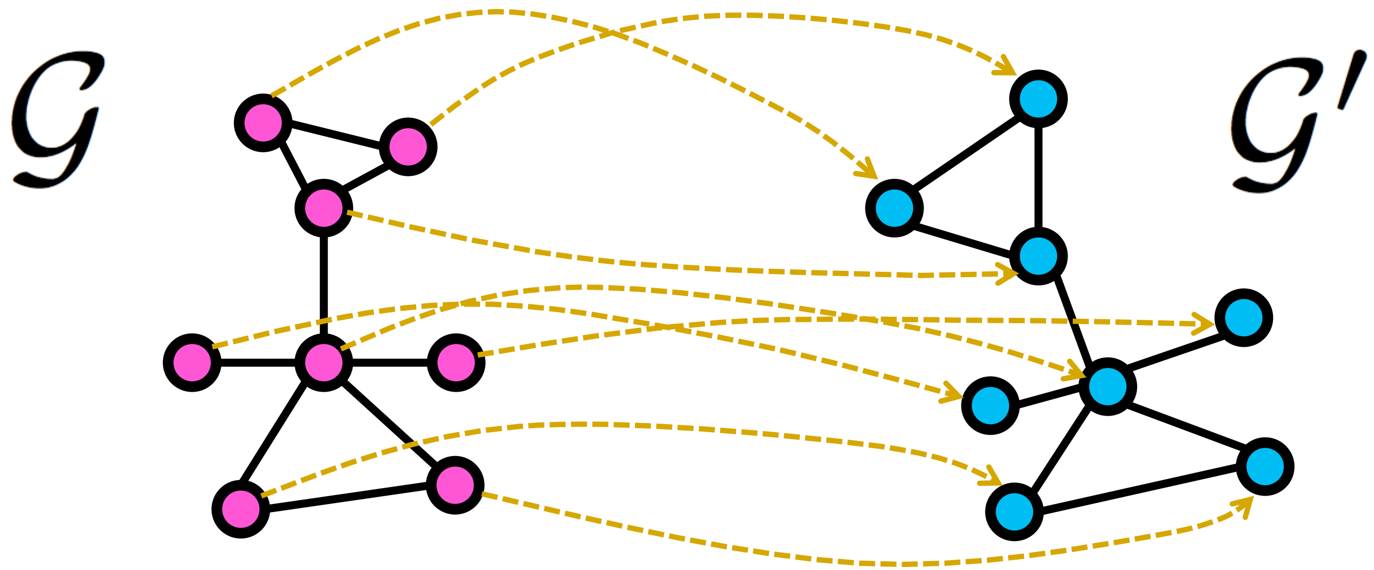
Graph Matching in Vision

- **Finding matches between two IMAGES**

- Non-rigid or deformable objects
- Feature matching by minimizing distortion

Graph Matching

- **Finding matches between two GRAPHS**

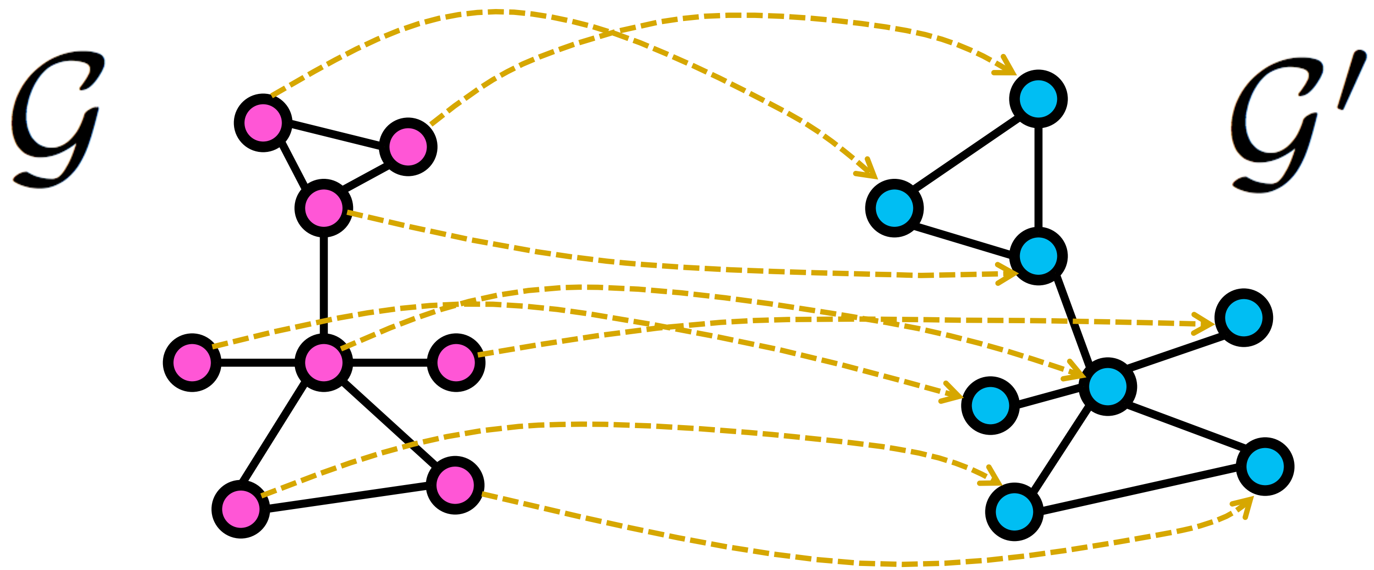


$$\mathbf{y} \in \{0,1\}^{nn'}$$

- $\mathbf{y}_{ia}$= 1 if node ***i*** in $\mathcal{G}$ corresponds to node ***a*** in $\mathcal{G}$'
- $\mathbf{y}_{ia}$= 0 otherwise

Graph Matching

- **Maximizing the matching score S**

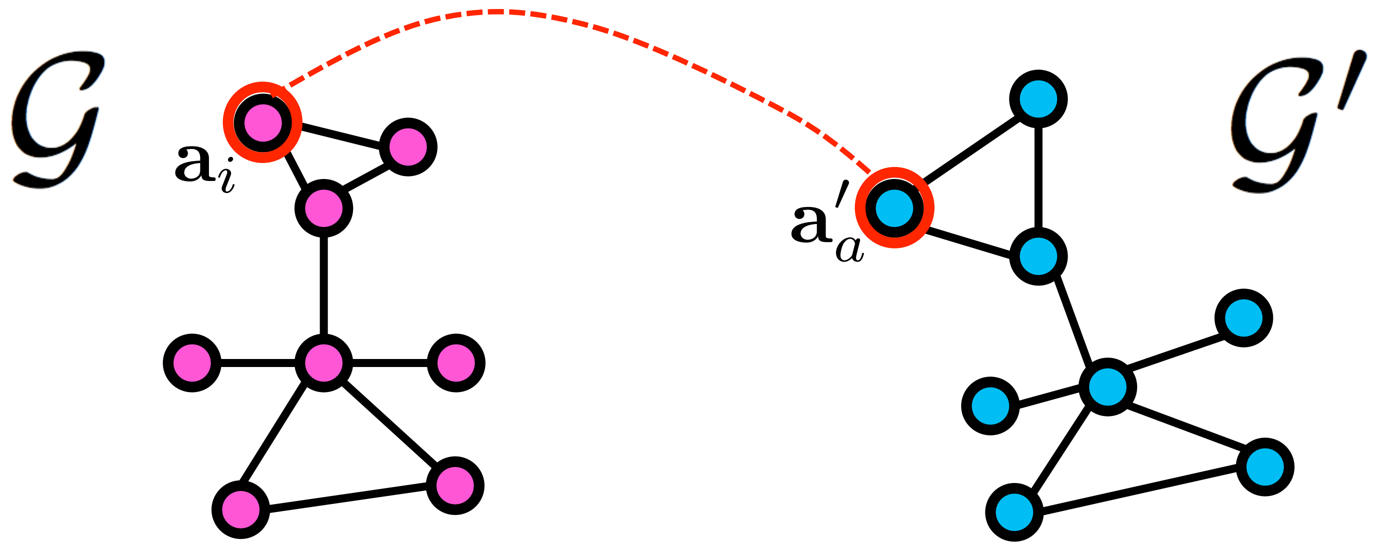


$$\mathbf{y}^* = \arg\max_{\mathbf{y}} S(\mathcal{G}, \mathcal{G}', \mathbf{y}),$$

$$s.t. \begin{cases} \mathbf{y} \in \{0,1\}^{nn'}, \\ \sum_{i=1}^{n} \mathbf{y}_{ia} \leq 1, \;\; \sum_{a=1}^{n'} \mathbf{y}_{ia} \leq 1 \end{cases}$$

Graph Matching

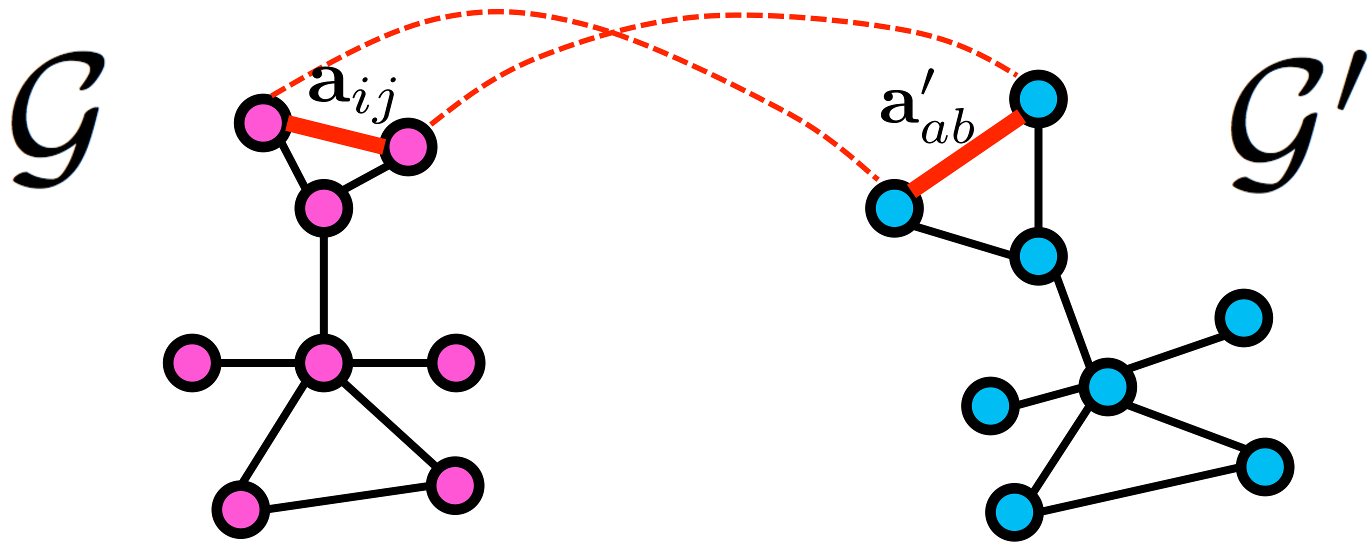
- **How to measure the matching score S ?**



- Each node & each edge has its own attribute
- Node similarity function $\mathbf{s}_V(\mathbf{a}_i, \mathbf{a}'_a)$

Graph Matching

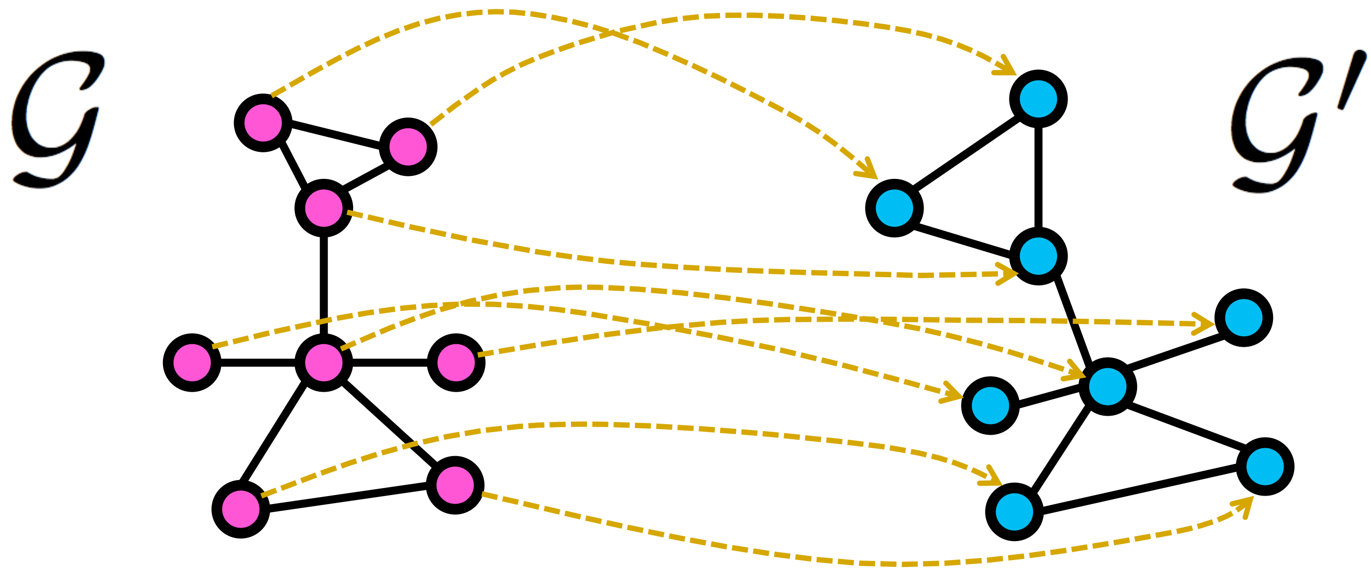
- **How to measure the matching score S ?**



- Each node & each edge has its own attribute.
- Node similarity function $\mathbf{s}_V(\mathbf{a}_i, \mathbf{a}'_a)$
- Edge similarity function $\mathbf{s}_E(\mathbf{a}_{ij}, \mathbf{a}'_{ab})$

Graph Matching

- **How to measure the matching score S ?**



$$S(\mathcal{G},\mathcal{G}',\mathbf{y}) = \sum_{\mathbf{y}_{ia}=1} \mathbf{s}_V(\mathbf{a}_i,\mathbf{a}'_a) + \sum_{\substack{\mathbf{y}_{ia}=1\\ \mathbf{y}_{jb}=1}} \mathbf{s}_E(\mathbf{a}_{ij},\mathbf{a}'_{ab})$$

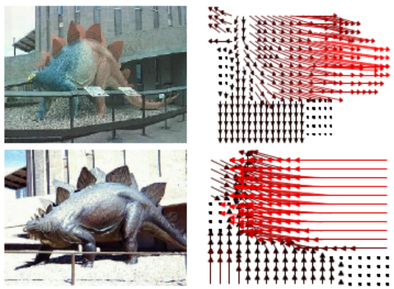
- Sum of $\mathbf{S}_\mathrm{V}$ and $\mathbf{S}_\mathrm{E}$ values for the assignment $\mathbf{y}$

Advances in Graph Matching

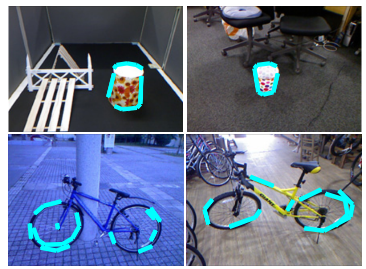
- **Quadratic assignment problem**
  - NP-hard, thus exact solution is infeasible
- **Advances in approximate algorithms**
  - Relaxation and Projection — Cour et al. '07, Leordeanu et al. '09, Zaslavskiy et al. '09
- **Hyper-graph extensions**
  - High-order potentials — Zass & Shashua '08, Duchenne et al. '10
  - Generalized formulation — Lee et al. '11, Leordeanu et al. '12
- **Boosting techniques**
  - Online-update of GM — Cho & Lee '12
  - Factorization of GM — Zhu & Torre '12

Recent applications in Vision

Object Recognition

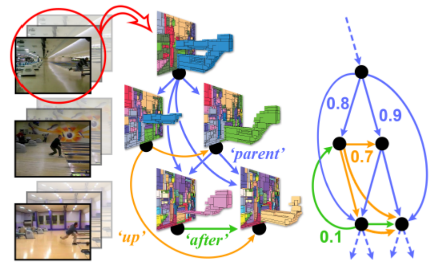


Duchenne et al.
ICCV 2011

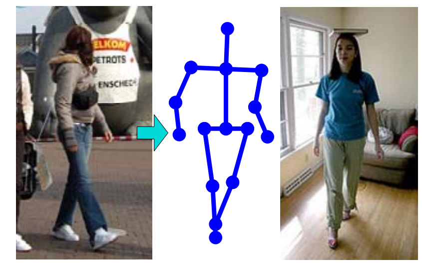


Zhang et al.
CVPR 2013

Action Recognition

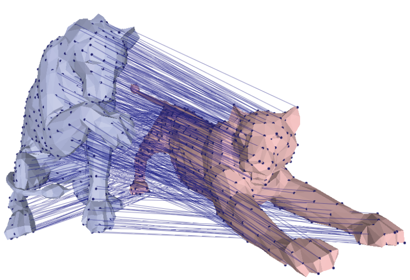


Brendel & Todorovic
ICCV 2011

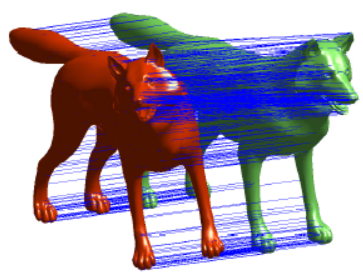


Yao & Fei-Fei
ECCV 2012

Shape Matching

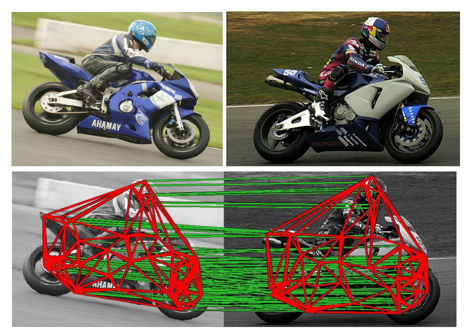


Zheng et al.
CVPR 2010



Smeets et al.
CVPR 2011

Image Matching



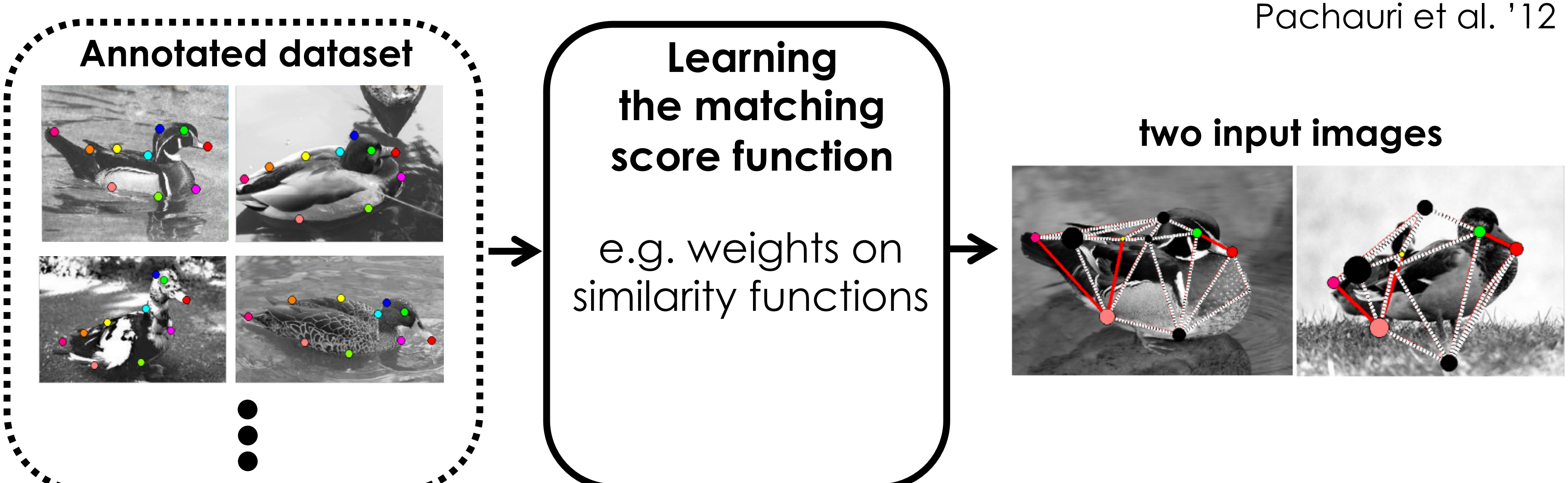
Cho & Lee
CVPR 2012

Leordeanu et al.
IJCV 2012

Motivation

- **How to improve matching by learning?**
  - A hand-crafted matching score function performs poor in many practical problems
  - Learn parameters of the matching score function to better match two instances

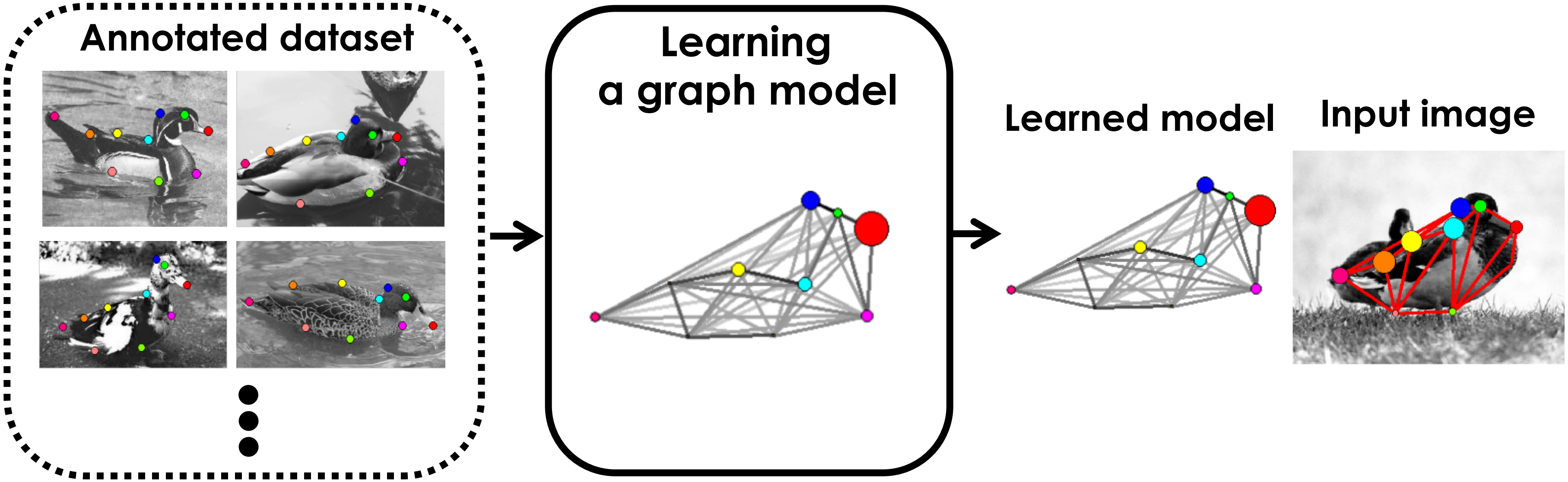
Caetano et al. '07
Torresani et al. '08
Leordeanu et al. '12
Pachauri et al. '12



Our Approach

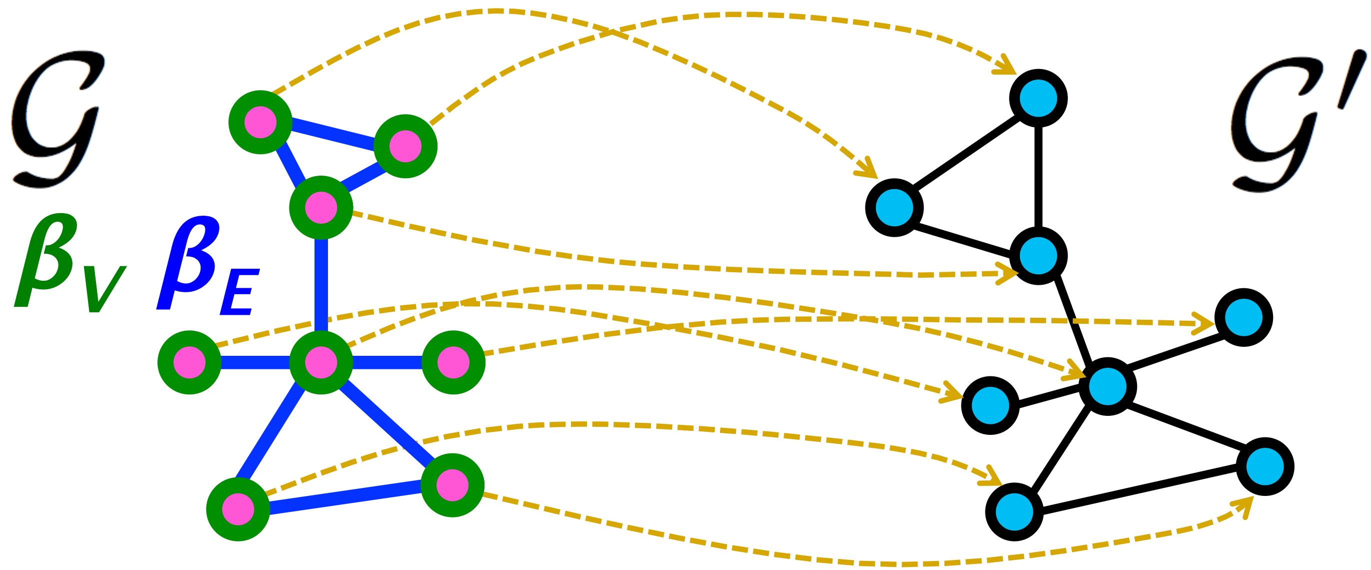
- **How to obtain a graph model for matching?**
  - Learn the class-specific graph model from training data, and use it to match to instances of the class
  - Related to generic graph learning

Lee et al. '06
Hofling & Tibshirani '09
Nowozin et al. '10



What to Learn? : Previous Approaches

- **Shared weights on nodes & on edges**



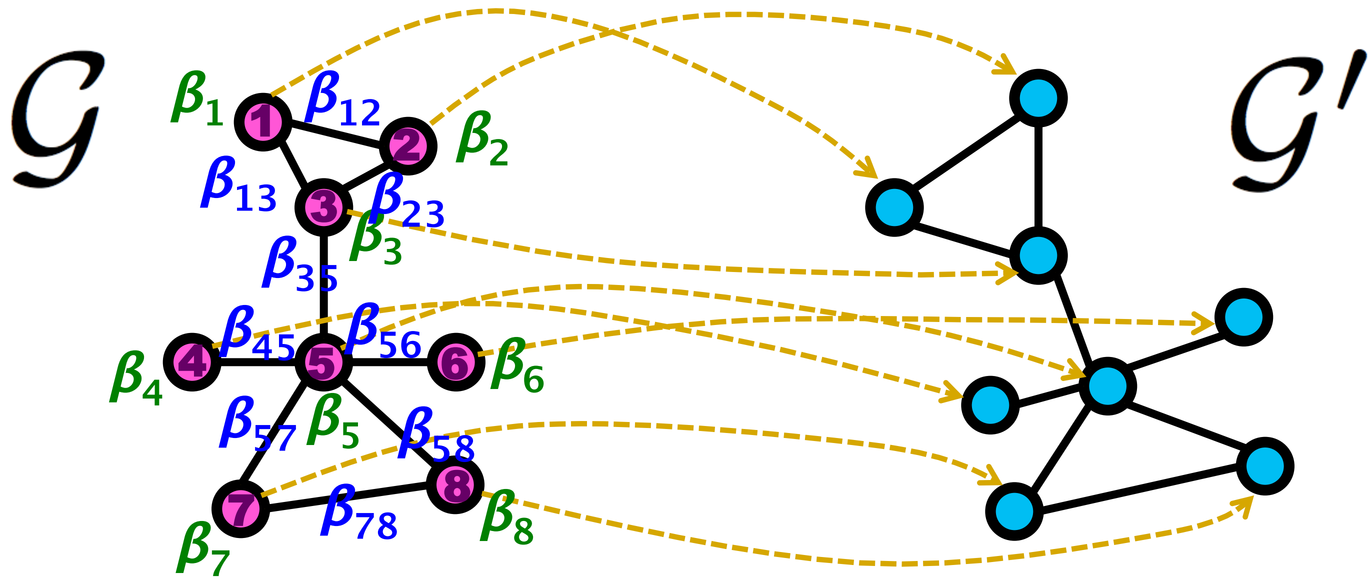
$$S(\mathcal{G}, \mathcal{G}', \mathbf{y}; \boldsymbol{\beta}) = \sum_{\mathbf{y}_{ia}=1} \boldsymbol{\beta}_V \cdot \mathbf{s}_V(\mathbf{a}_i, \mathbf{a}'_a) + \sum_{\substack{\mathbf{y}_{ia}=1 \\ \mathbf{y}_{jb}=1}} \boldsymbol{\beta}_E \cdot \mathbf{s}_E(\mathbf{a}_{ij}, \mathbf{a}'_{ab})$$

- All nodes share the same weight $\boldsymbol{\beta}_V$
- All edges share the same weight $\boldsymbol{\beta}_E$

Caetano et al. '07
Torresani et al. '08
Leordeanu et al. '12
Pachauri et al. '12

What to Learn? : Generalization

- **Discriminative weights**

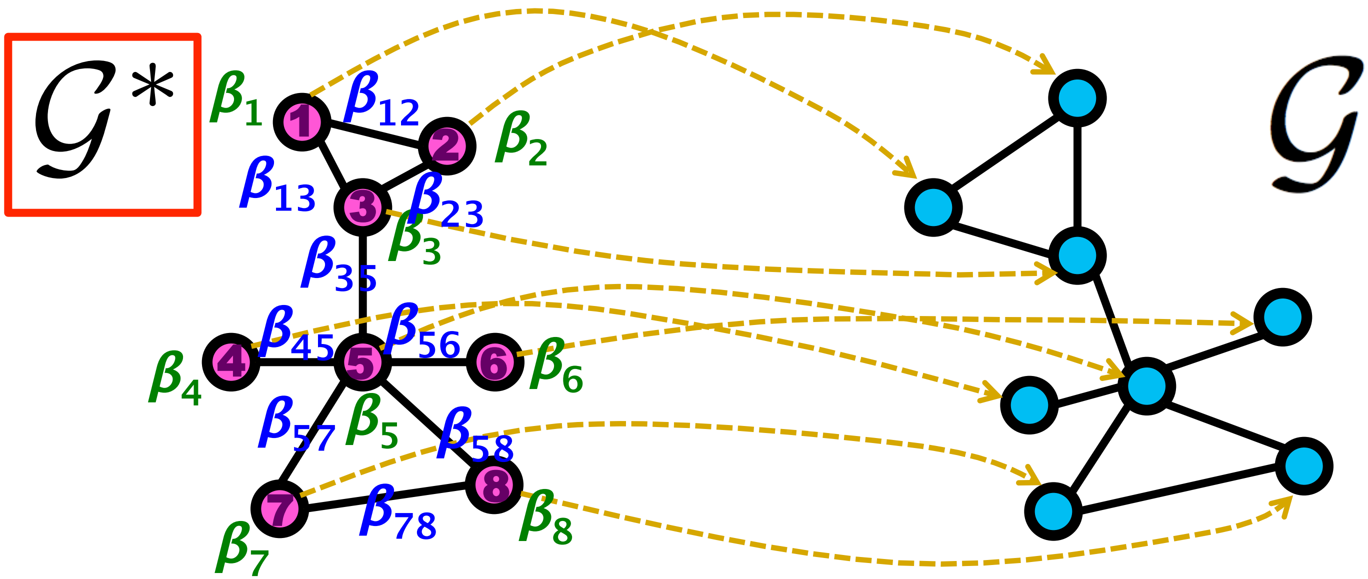


$$S(\mathcal{G}, \mathcal{G}', \mathbf{y}; \boldsymbol{\beta}) = \sum_{\mathbf{y}_{ia}=1} \boldsymbol{\beta}_i \cdot \mathbf{s}_V(\mathbf{a}_i, \mathbf{a}'_a) + \sum_{\substack{\mathbf{y}_{ia}=1 \\ \mathbf{y}_{jb}=1}} \boldsymbol{\beta}_{ij} \cdot \mathbf{s}_E(\mathbf{a}_{ij}, \mathbf{a}'_{ab})$$

- Each node and edge has its own weight
- This generalizes the previous learning approaches

What to Learn? : Graph Model

- **Model and weights**



$$S(\boxed{\mathcal{G}^*}, \mathcal{G}, \mathbf{y}; \boldsymbol{\beta}) = \sum_{\mathbf{y}_{ia}=1} \boldsymbol{\beta}_i \bullet \mathbf{s}_V(\boxed{\mathbf{a}_i^*}, \mathbf{a}_a) + \sum_{\substack{\mathbf{y}_{ia}=1 \\ \mathbf{y}_{jb}=1}} \boldsymbol{\beta}_{ij} \bullet \mathbf{s}_E(\boxed{\mathbf{a}_{ij}^*}, \mathbf{a}_{ab})$$

- Goal: learn model graph $\mathcal{G}^*$ and weights $\boldsymbol{\beta}$
- **How to parameterize $\mathcal{G}^*$ and $\boldsymbol{\beta}$?**

Parameterization

- **Assume the similarity function is the dot product of two attributes:**

$$\mathbf{s}_V(\mathbf{a}_i^*, \mathbf{a}_a) = \mathbf{a}_i^* \cdot \mathbf{a}_a, \quad \mathbf{s}_E(\mathbf{a}_{ij}^*, \mathbf{a}_{ab}) = \mathbf{a}_{ij}^* \cdot \mathbf{a}_{ab}$$

  - Then, the attributes of the model graph can be factored out and combined with the weights:

$$\begin{aligned} S(\mathcal{G}^*, \mathcal{G}, \mathbf{y}; \beta) &= \sum_{\mathbf{y}_{ia}=1} \beta_i \mathbf{s}_V(\mathbf{a}_i^*, \mathbf{a}_a) + \sum_{\substack{\mathbf{y}_{ia}=1 \\ \mathbf{y}_{jb}=1}} \beta_{ij} \mathbf{s}_E(\mathbf{a}_{ij}^*, \mathbf{a}_{ab}) \\ &= \sum_{\mathbf{y}_{ia}=1} (\beta_i \mathbf{a}_i^*) \cdot \mathbf{a}_a + \sum_{\substack{\mathbf{y}_{ia}=1 \\ \mathbf{y}_{jb}=1}} (\beta_{ij} \mathbf{a}_{ij}^*) \cdot \mathbf{a}_{ab} \\ &= \underbrace{(\beta \odot \Theta(\mathcal{G}^*))}_{\text{Model and weights}} \cdot \underbrace{\Psi(\mathcal{G}, \mathbf{y})}_{\text{Feature map}} \\ &= \underline{\mathbf{w}} \cdot \Psi(\mathcal{G}, \mathbf{y}) \end{aligned}$$

Max-Margin Learning

- **Learned in the standard SSVM framework**
  - Given training data $D = (\langle \mathcal{G}_1, \mathbf{y}_1 \rangle, \ldots, \langle \mathcal{G}_n, \mathbf{y}_n \rangle)$,

    Minimize

$$L_D(\mathcal{G}^*, \beta) = \underline{r(\mathcal{G}^*, \beta)} + \frac{C}{n} \sum_{i=1}^{n} \underline{\Delta(\mathbf{y}_i, \underline{\hat{\mathbf{y}}(\mathcal{G}_i; \mathcal{G}^*, \beta)})}$$

    - Predictor: $\underline{\hat{\mathbf{y}}(\mathcal{G}; \mathcal{G}^*, \beta)} = \hat{\mathbf{y}}(\mathcal{G}; \mathbf{w}) = \arg\max_{\mathbf{y} \in \mathcal{Y}(\mathcal{G})} \mathbf{w} \cdot \Psi(\mathcal{G}, \mathbf{y})$
    - Loss function: $\underline{\Delta(\mathbf{y}, \hat{\mathbf{y}})} = 1 - \frac{1}{||\mathbf{y}||_F^2} \mathbf{y} \cdot \hat{\mathbf{y}}$ **(Normalized Hamming loss)** Caetano et al. '09
    - Regularization: $\underline{r(\mathcal{G}^*, \beta)} = \frac{1}{2} ||\mathbf{w}||^2$
  - Optimization by the cutting plane method

Joachims et al. '09

Graph Representation

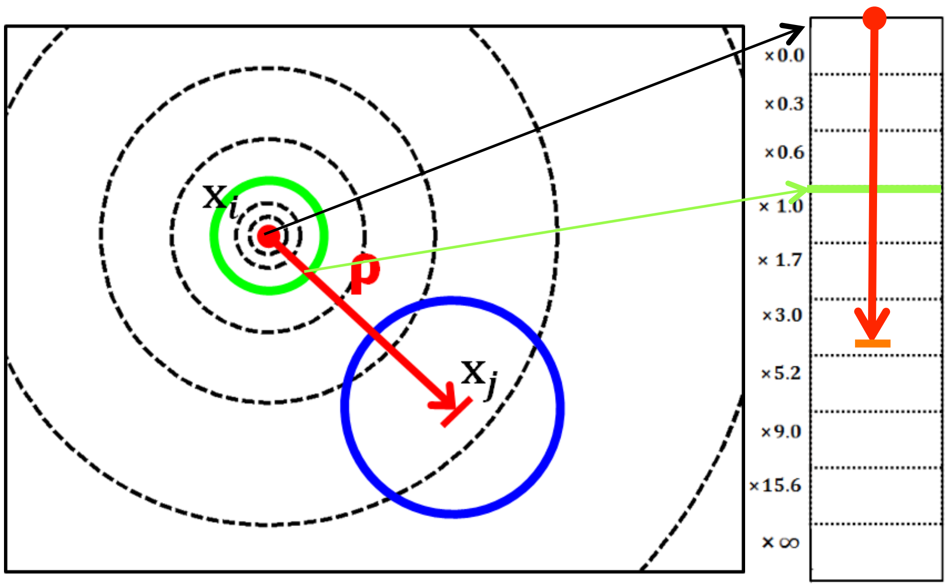
- **Proposition**
  - For **any graph representation** where **dot product between two attributes** is defined as their **similarity**, both of the model graph attributes and their weights can be jointly learned as a single vector.

- **Our proposal for visual matching problems**
  - **Histogram-Attributed Relational Graph (HARG)**
    - Node attribute: histograms of gradient bins (SIFT in this work)
    - Edge attribute: histograms of log-polar bins (as follows)

Lowe '04
Dalal & Triggs '05

Graph Representation

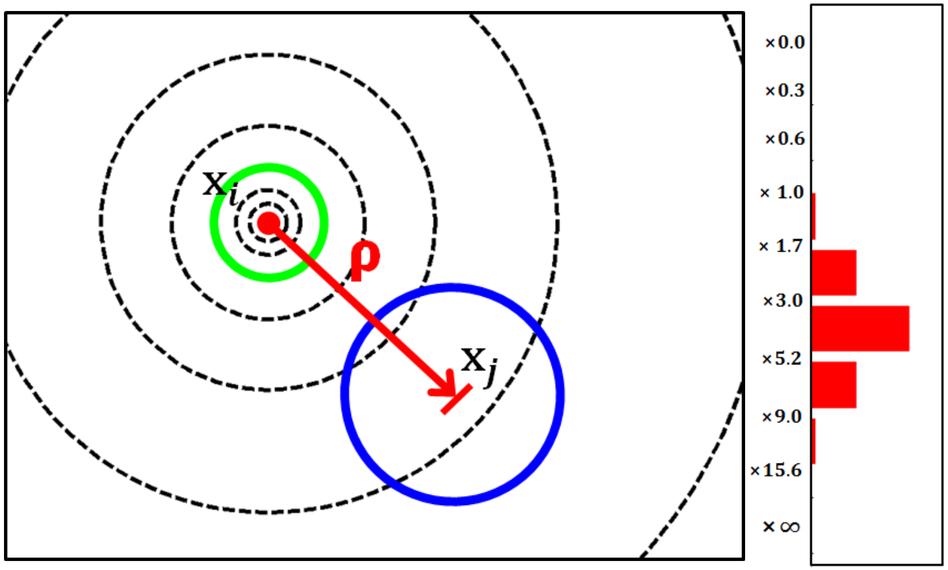
- **Edge attribute**
  - **histograms of log-polar bins**
    - **Concatenation of length and angle histograms**



length of edge e_{ij} and its histogram attribute

Graph Representation

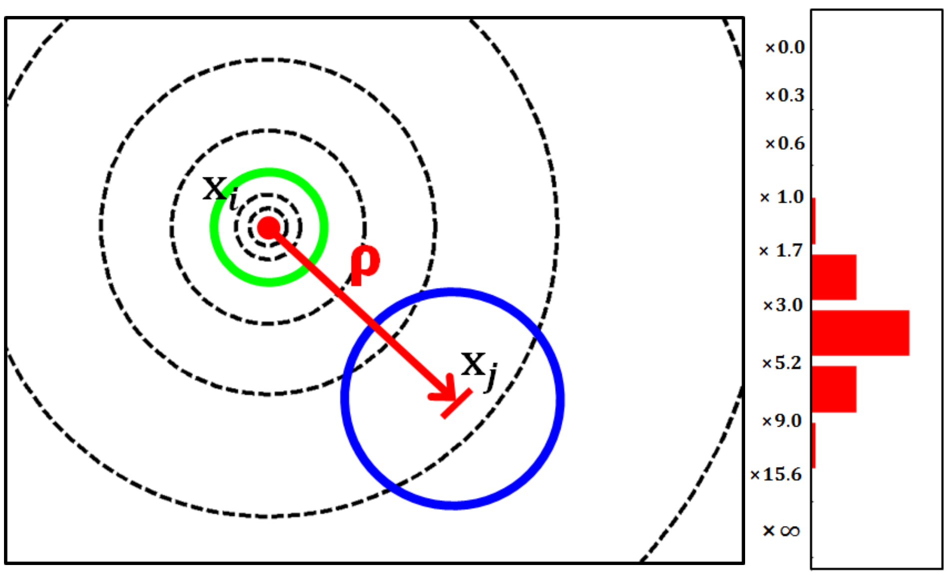
- Edge attribute
  - histograms of log-polar bins
    - Concatenation of length and angle histograms



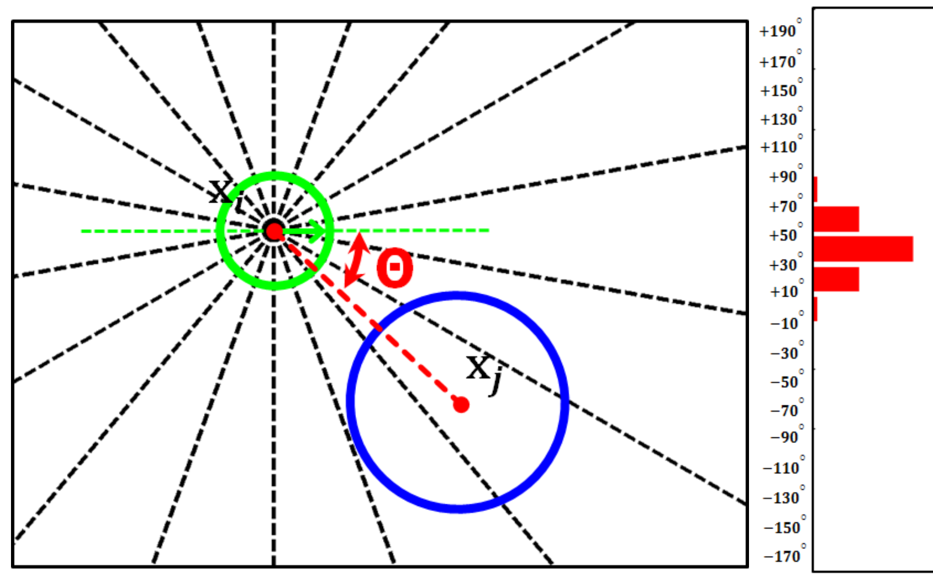
length of edge e_{ij} and its histogram attribute

Graph Representation

- **Edge attribute**
  - **histograms of log-polar bins**
    - **Concatenation of length and angle histograms**



length of edge e_{ij} and its histogram attribute



angle of edge e_{ij} and its histogram attribute

  - **Non-parametric** length and angle distribution
  - Robust to variation, and effective in learning

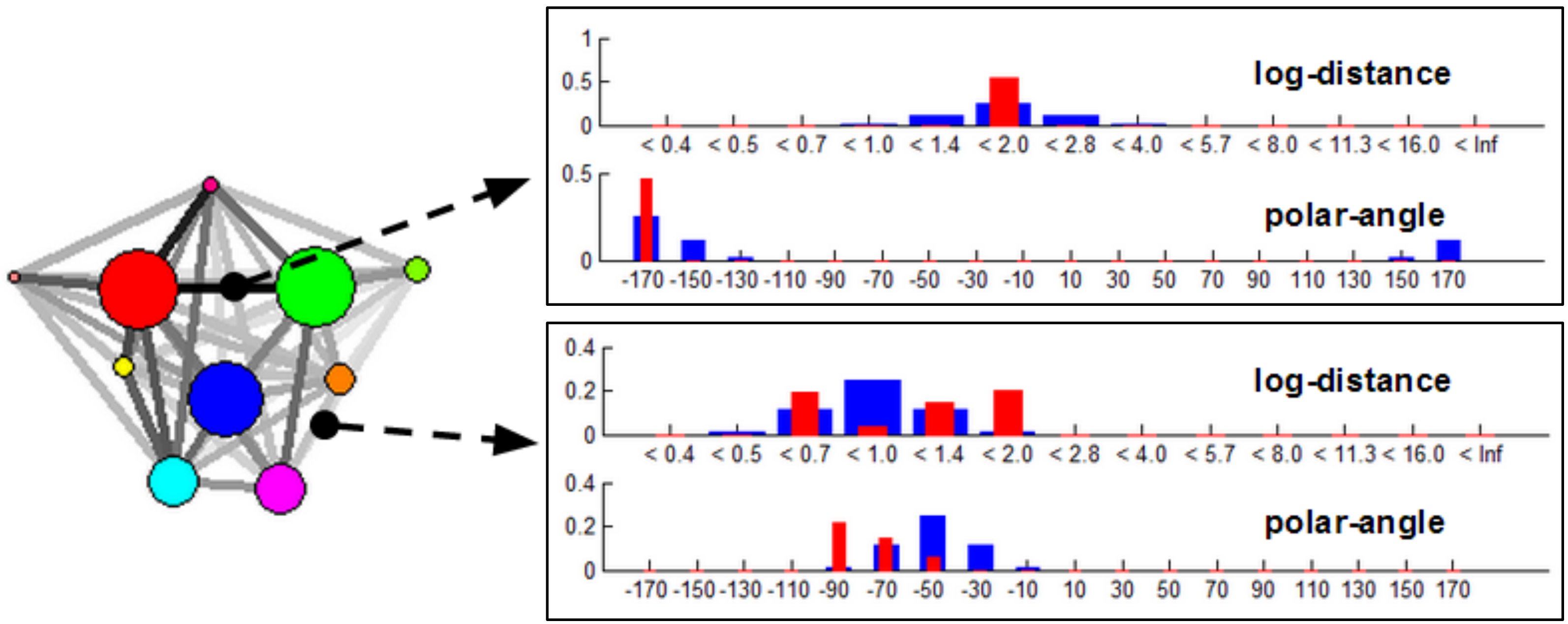
Max-Margin Learning

- **Example of a learned graph model**
  - Face images with 10 point annotations

• • •

Max-Margin Learning

- **Example of a learned face model**



- **Larger** weights on **darker** edges & **bigger** nodes
- Examples of learned edge attributes
  - Blue histograms: attributes from a training image
  - Red histograms: attributes from the learned model

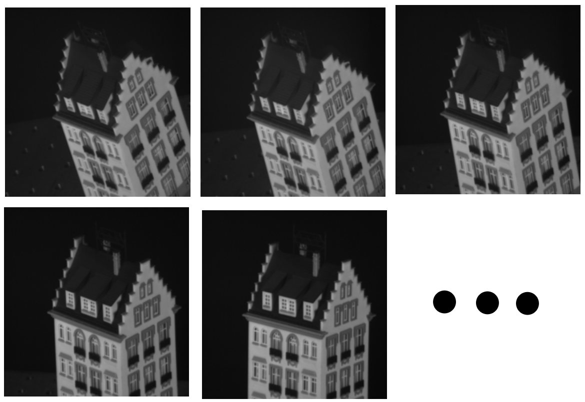
Experimental Evaluation

- **On synthetic and real image datasets**
  - Synthetic point sets
  - CMU House/Hotel
  - Object classes (5 classes from Caltech-256 & PASCAL VOC 2007)

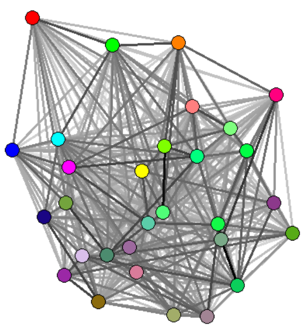
- **Graph construction and matching**
  - Fully-connected graph as an initial graph
  - Graph matching module: RRWM Cho et al. '10

Experiments: CMU House/Hotel

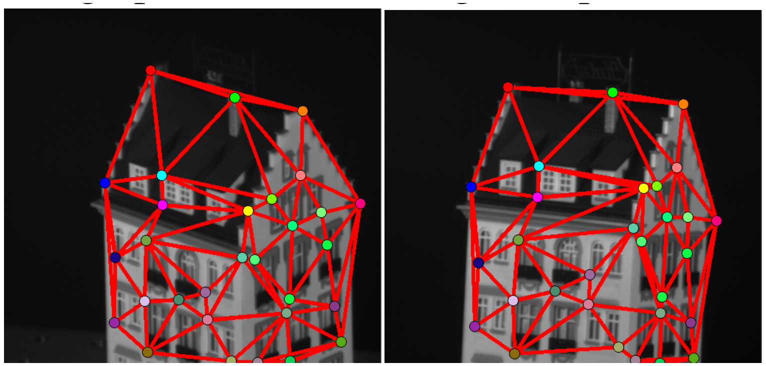
- **Image sequence with varying viewpoints**
  - 111 images for House, 101 images for Hotel
  - 30 annotated points for each frame Caetano et al., 2007



Hotel sequence



Learned model



Matching results

		House	Hotel
Method	Training size	Accuracy (%)	Accuracy (%)
Caetano *et al.*'09	5	84	87
Caetano *et al.*'09	106	96	90
Leordeanu *et al.*'12	5	99.8	94.8
HARG-SSVM (ours)	3	**100.0**	**100.0**

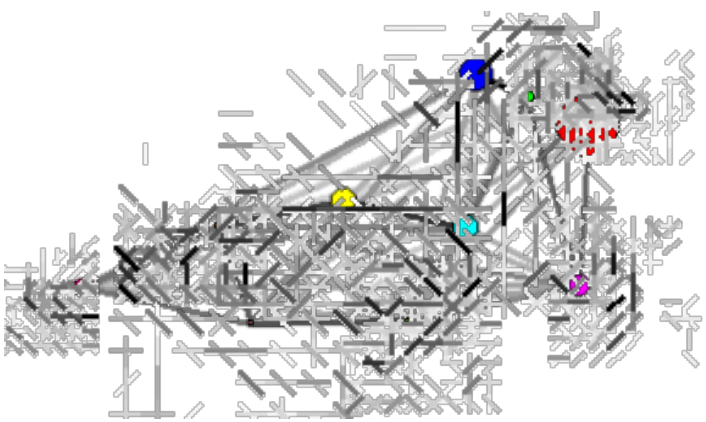
Experiments: Object Classes

- **Annotated object class dataset**
  - 5 object classes constructed using images from Caltech-256 and PASCAL VOC datasets
    (Face:109, Duck: 50, Wine bottle: 66, Motorbike: 40, Car: 40)
  - 10 distinctive points annotated for each image

- **Quantitative evaluation**
  - 20 images for training and the rest for testing
  - Endpoint error for each match w.r.t object size
    - True match if the error < 0.15
  - Average performance over the 20 random splits

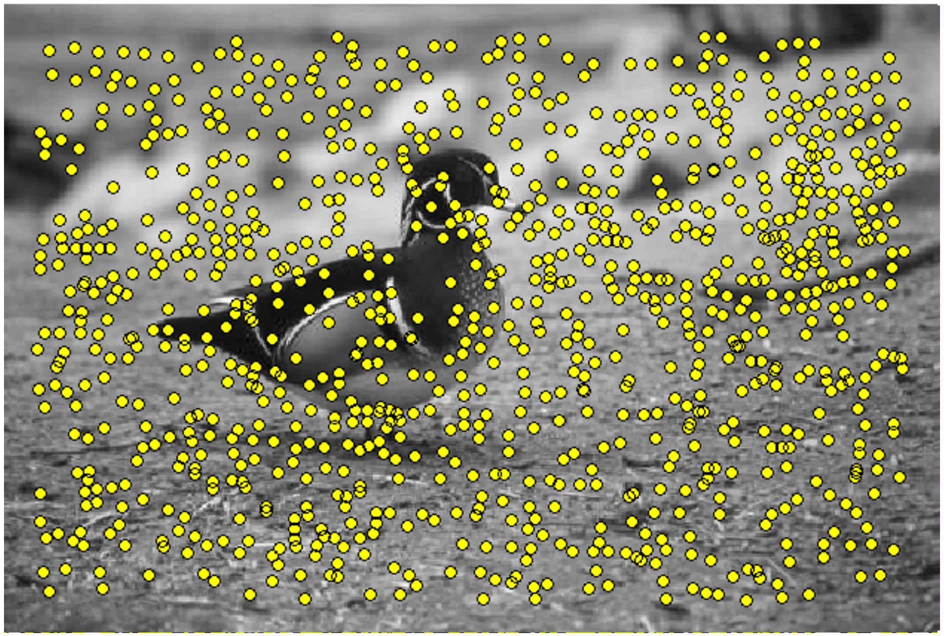
Experiments: Object Classes

- **Duck**



Learned model

- Node color: feature identity
- Bigger nodes: larger weights
- Darker edges: larger weights

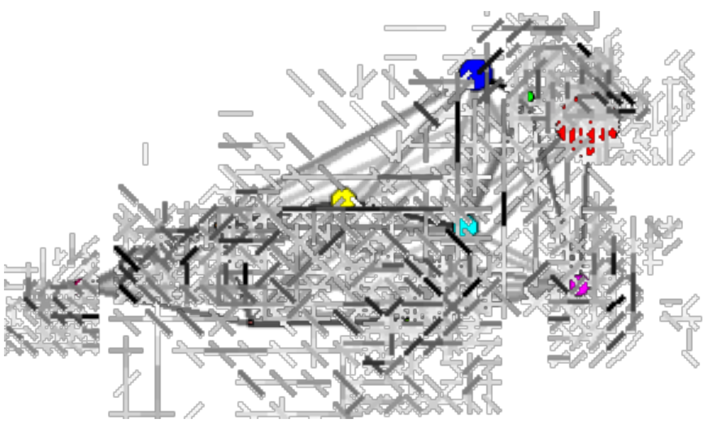


Input image

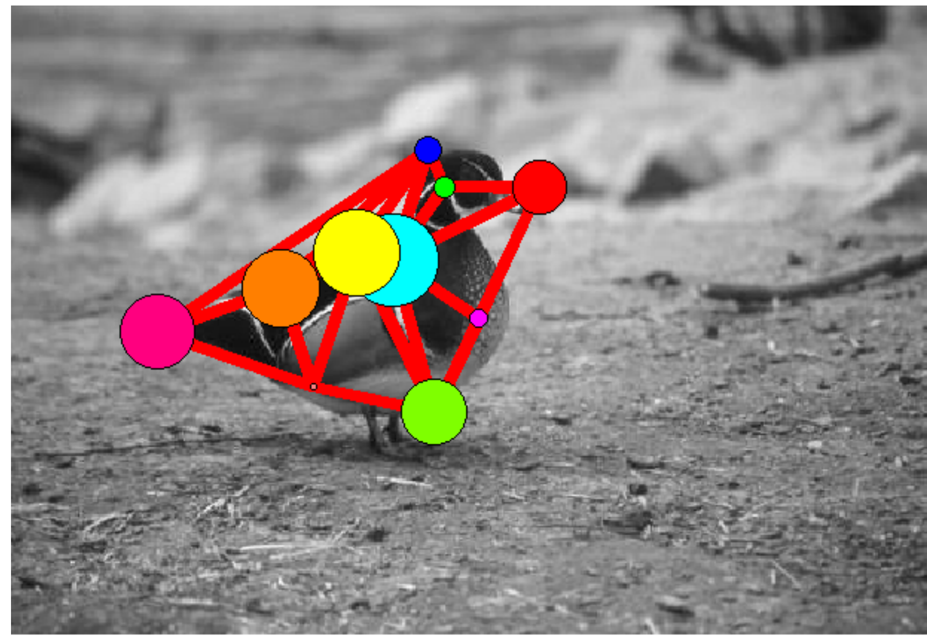
- Node color: matching feature
- Bigger nodes: higher similarity
- Red edges: connecting true ones

Experiments: Object Classes

- **Duck**



Learned model



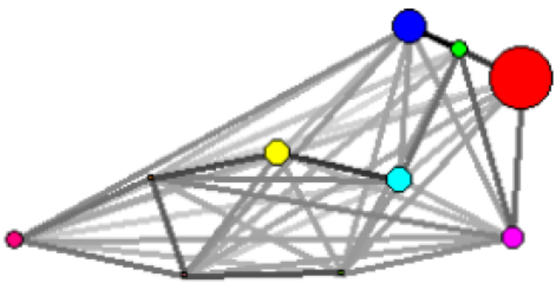
Input image

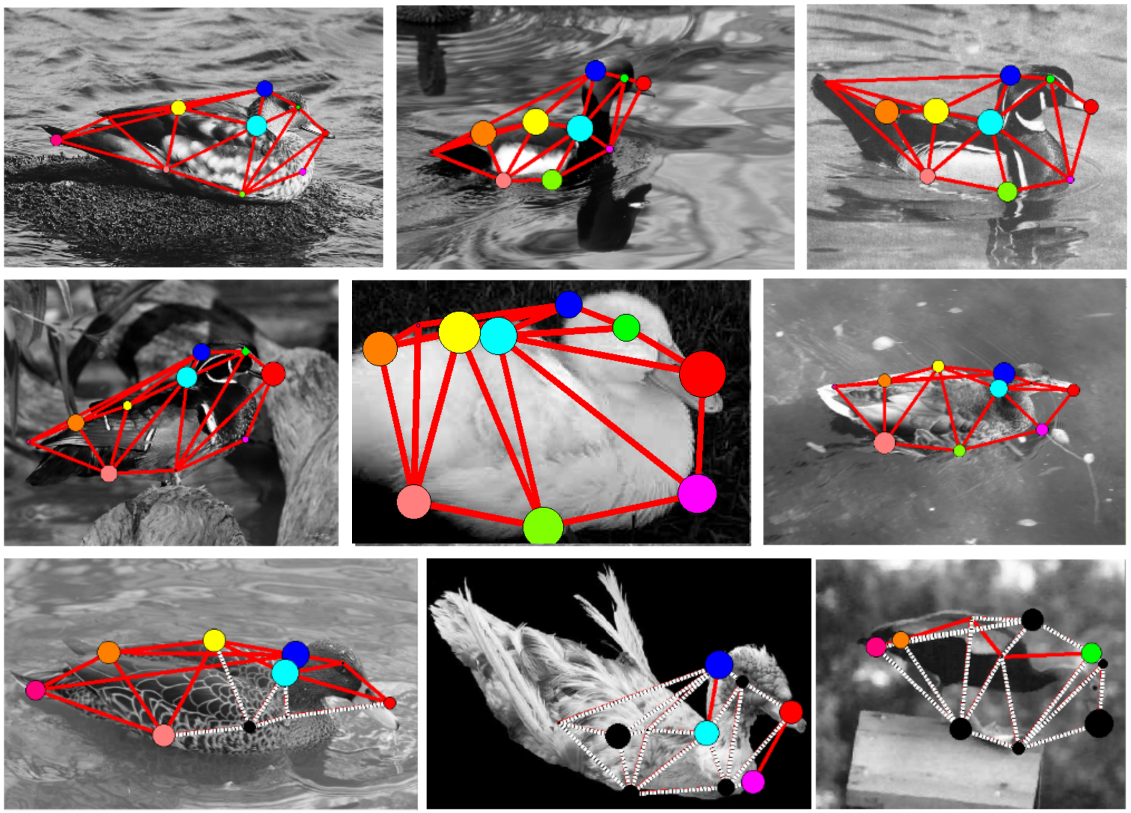
- Node color: feature identity
- Bigger nodes: larger weights
- Darker edges: larger weights

- Node color: matching feature
- Bigger nodes: higher similarity
- Red edges: connecting true ones

Experiments: Object Classes

- Duck

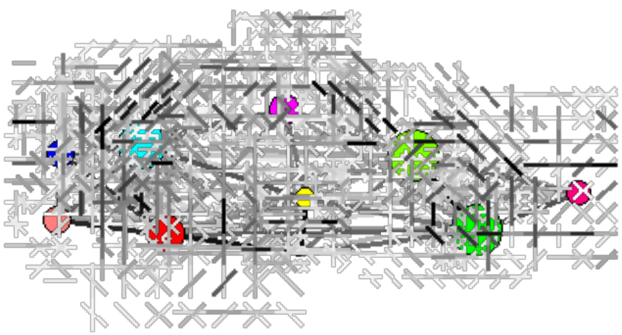




- Black nodes: false matches

Experiments: Object Classes

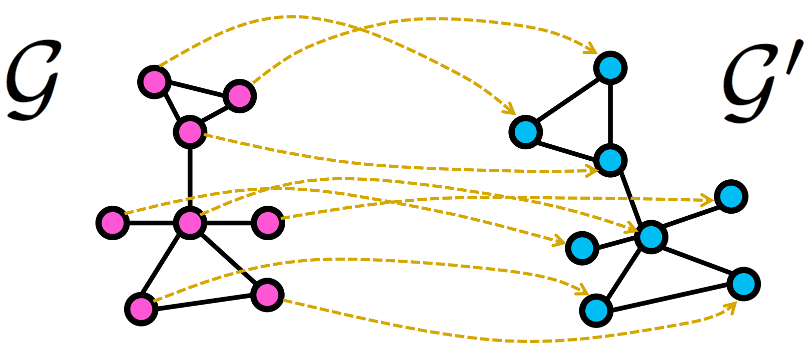
- **Car**



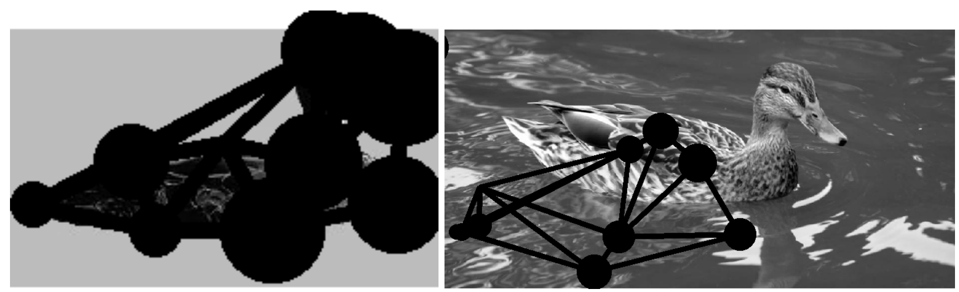
- Black nodes: false matches

Experiments: Object Classes

- **Comparison**
  - **w/o learning**: uniform weights without learning

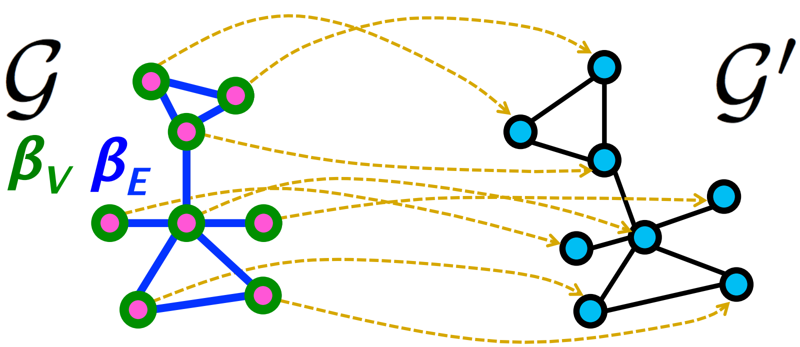


$$S(\mathcal{G}, \mathcal{G}', \mathbf{y}) = \sum_{\mathbf{y}_{ia}=1} \mathbf{s}_V(\mathbf{a}_i, \mathbf{a}'_a) + \sum_{\substack{\mathbf{y}_{ia}=1 \\ \mathbf{y}_{jb}=1}} \mathbf{s}_E(\mathbf{a}_{ij}, \mathbf{a}'_{ab})$$

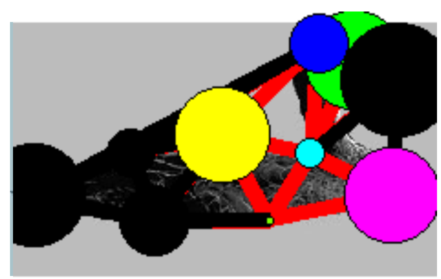


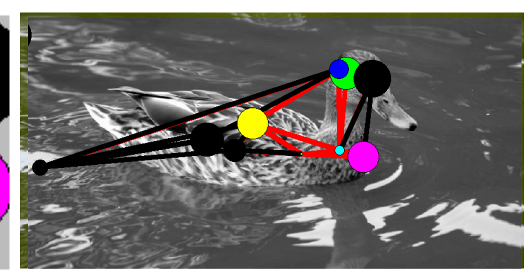
Experiments: Object Classes

- **Comparison**
  - w/o learning: uniform weights without learning
  - **SW-SSVM**: shared weights, learned in SSVM Caetano et al., 2007



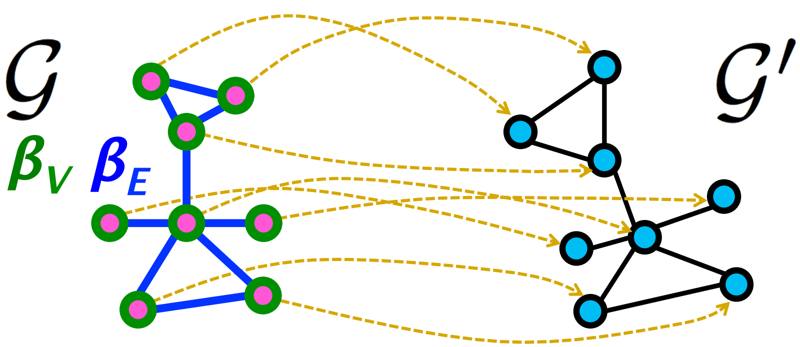
$$S(\mathcal{G},\mathcal{G}',\mathbf{y};\boldsymbol{\beta}) = \sum_{\mathbf{y}_{ia}=1} \boldsymbol{\beta}_V \cdot \mathbf{s}_V(\mathbf{a}_i,\mathbf{a}'_a) + \sum_{\substack{\mathbf{y}_{ia}=1\\ \mathbf{y}_{jb}=1}} \boldsymbol{\beta}_E \cdot \mathbf{s}_E(\mathbf{a}_{ij},\mathbf{a}'_{ab})$$



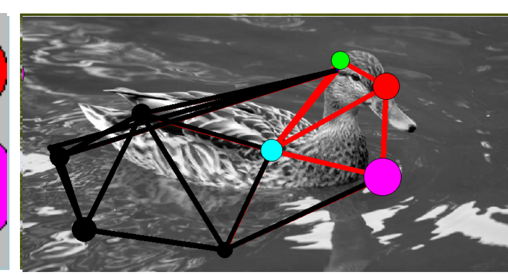


Experiments: Object Classes

- **Comparison**
  - w/o learning: uniform weights without learning
  - SW-SSVM: shared weights, learned in SSVM Caetano et al., 2007
  - **SW-SPEC**: shared weights, learned by Leordeanu et al., 2012

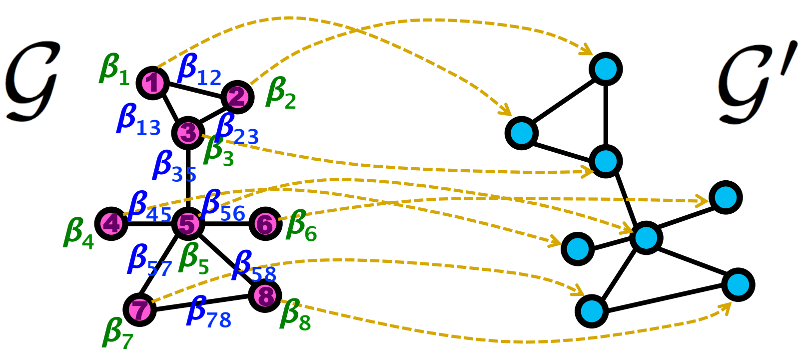


$$S(\mathcal{G},\mathcal{G}',\mathbf{y};\boldsymbol{\beta}) = \sum_{\mathbf{y}_{ia}=1} \boldsymbol{\beta}_V \cdot \mathbf{s}_V(\mathbf{a}_i,\mathbf{a}'_a) + \sum_{\substack{\mathbf{y}_{ia}=1\\ \mathbf{y}_{jb}=1}} \boldsymbol{\beta}_E \cdot \mathbf{s}_E(\mathbf{a}_{ij},\mathbf{a}'_{ab})$$

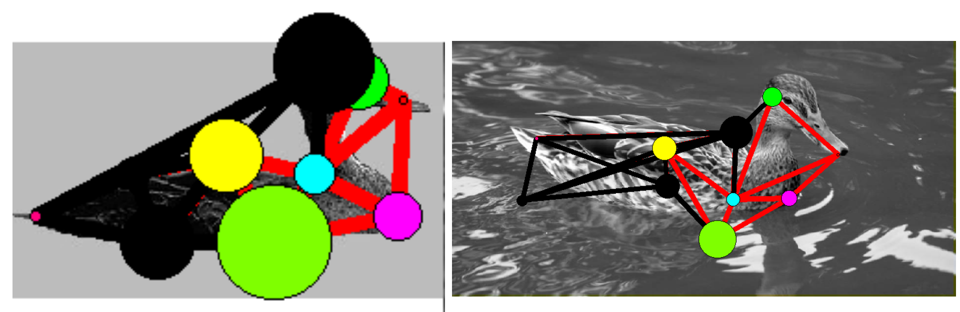


Experiments: Object Classes

- **Comparison**
  - w/o learning: uniform weights without learning
  - SW-SSVM: shared weights, learned in SSVM Caetano et al., 2007
  - SW-SPEC: shared weights, learned by Leordeanu et al., 2012
  - **DW-SSVM**: individual weights, learned in SSVM

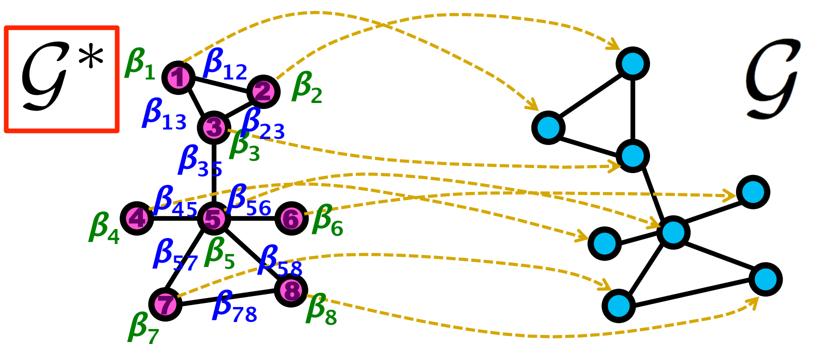


$$S(\mathcal{G},\mathcal{G}',\mathbf{y};\boldsymbol{\beta}) = \sum_{\mathbf{y}_{ia}=1} \beta_i \bullet \mathbf{s}_V(\mathbf{a}_i,\mathbf{a}'_a) + \sum_{\substack{\mathbf{y}_{ia}=1\\ \mathbf{y}_{jb}=1}} \beta_{ij} \bullet \mathbf{s}_E(\mathbf{a}_{ij},\mathbf{a}'_{ab})$$

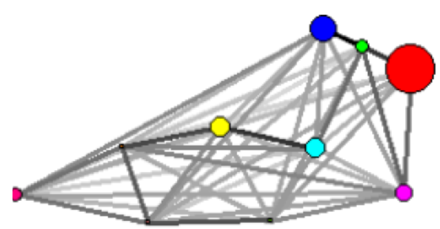


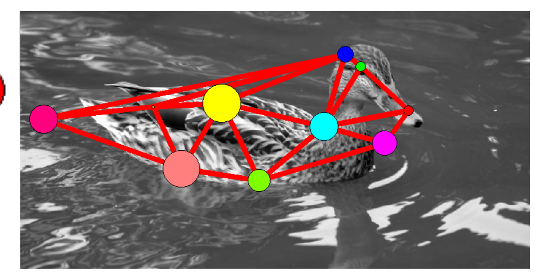
Experiments: Object Classes

- **Comparison**
  - w/o learning: uniform weights without learning
  - SW-SSVM: shared weights, learned in SSVM Caetano et al., 2007
  - SW-SPEC: shared weights, learned by Leordeanu et al., 2012
  - DW-SSVM: individual weights, learned in SSVM
  - **HARG-SSVM**: the proposed method



$$S(\boxed{\mathcal{G}^*}, \mathcal{G}, \mathbf{y}; \boldsymbol{\beta}) = \sum_{\mathbf{y}_{ia}=1} \beta_i \cdot \mathbf{s}_V(\boxed{\mathbf{a}_i^*}, \mathbf{a}_a) + \sum_{\substack{\mathbf{y}_{ia}=1 \\ \mathbf{y}_{jb}=1}} \beta_{ij} \cdot \mathbf{s}_E(\boxed{\mathbf{a}_{ij}^*}, \mathbf{a}_{ab})$$





Experiments: Object Classes

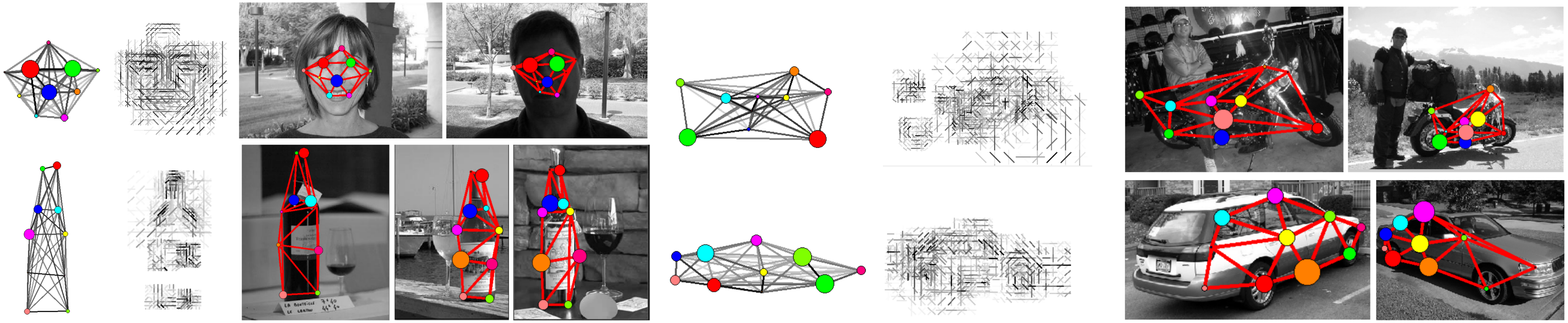
- **Quantitative comparison**
  - The average performance on 20 random splits

	Face		Motorbike		Car		Duck		Wine bottle	
Method	Acc. (%)	error	Acc. (%)	error	Acc. (%)	error	Acc. (%)	error	Acc. (%)	error
w/o learning	66.6	0.205	44.1	0.226	34.1	0.301	39.0	0.228	70.5	0.129
SW-SSVM	75.3	0.142	48.6	0.211	40.3	0.259	42.2	0.216	73.3	0.122
SW-SPEC	78.7	0.133	47.2	0.212	42.1	0.253	44.2	0.211	72.4	0.124
DW-SSVM	84.3	0.102	54.2	0.189	50.8	0.244	52.1	0.186	75.5	0.120
HARG-SSVM	**93.9**	**0.070**	**71.4**	**0.134**	**71.9**	**0.158**	**72.2**	**0.126**	**86.1**	**0.090**

  - w/o learning: uniform weights without learning
  - SW-SSVM: shared weights, learned in SSVM Caetano et al., 2007
  - SW-SPEC: shared weights, learned by Leordeanu et al., 2012
  - DW-SSVM: individual weights learned in SSVM
  - **HARG-SSVM**: the proposed method

Summary

- **Effective learning of class-specific models**
  - Useful for a variety of practical matching problems



- Annotated datasets & code soon available:
  http://www.di.ens.fr/willow/research/graphlearning/

❖ This research was supported in part by the ERC advanced grant VideoWorld, Institut Universitaire de France, and the Quaero programme funded by the OSEO.