

AGA: Attribute-Guided Augmentation



Mandar Dixit
UC San Diego



Roland Kwitt
University of Salzburg

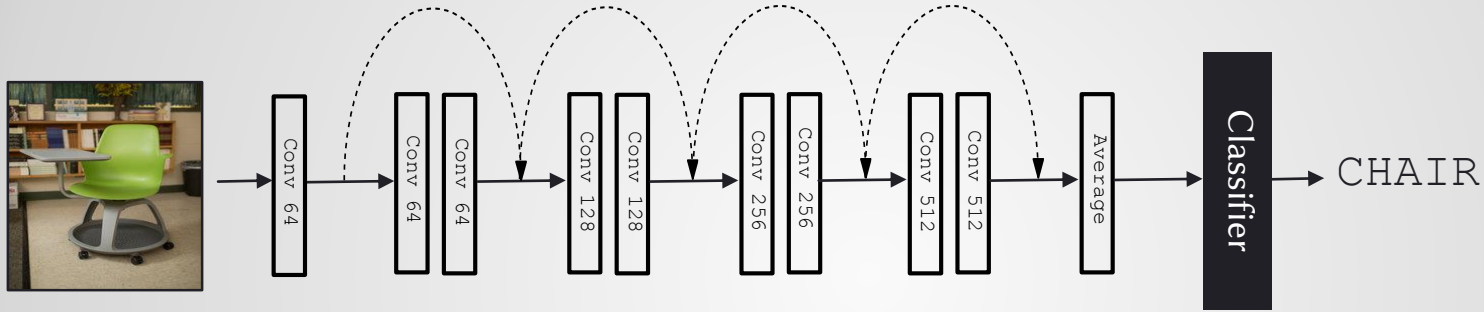


Marc Niethammer
UNC Chapel Hill



Nuno Vasconcelos
UC San Diego

Visual Recognition



Convolutional Neural Networks (CNNs) + Large-scale datasets



Challenge(s): appearance, pose, location invariance

Data Augmentation

Augmentation in **IMAGE** space



Original



Flip



(Random) Crops

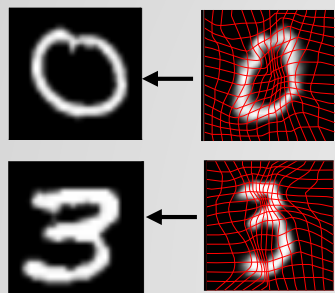


Crop + Flip

- Transformed copies of the original image
- Possible with simple **image processing**

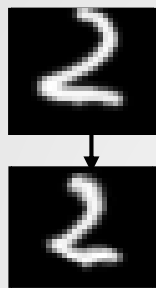
[Chatfield et al. 2014]
[Zeiler & Fergus 2014]
[Krishevsky et al. 2012]

Guided Data Augmentation

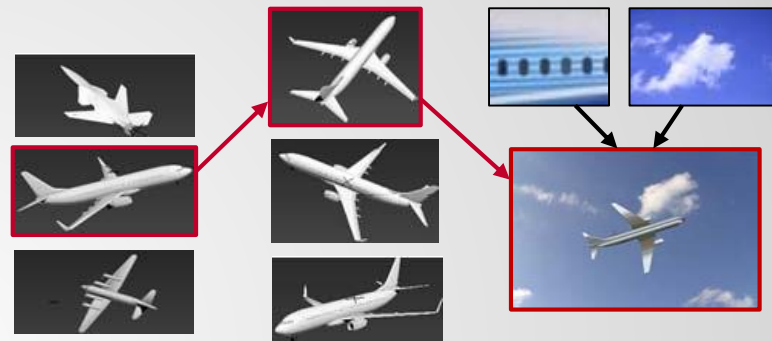


Learn

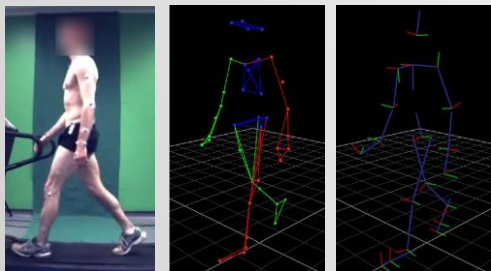
[Hauberg et al. 2016]



Apply



3D CAD based rendering
[Peng et al. 2015]



3D Motion Capture

[Charalambous et al. 2016], [Rogez & Schmid 2016]

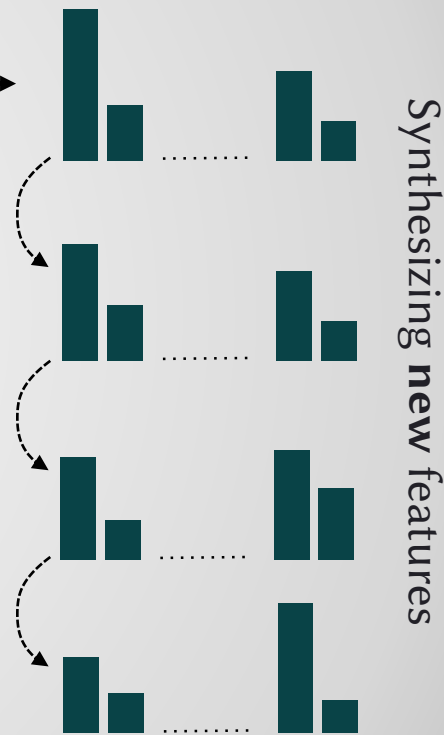


Synthesize

Synthesis of *non-trivial* variations

- Either **trivial in stimuli**, e.g., digits
- Or require **rendering** from 3D
- Image space - **high dimensional**
- Learning needs lot of data

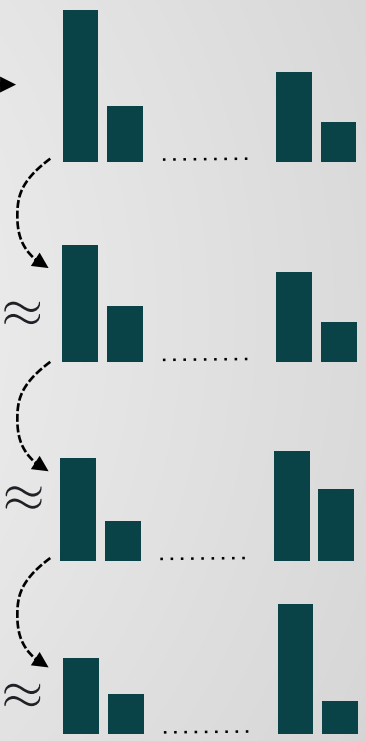
Objective



Desired variation
(e.g., **pose**)

Augmentation in space of **CNN activations**

Objective



Synthesizing **new** features

Equivalent images



Desired variation
(e.g., **pose**)

Augmentation in space of **CNN activations**

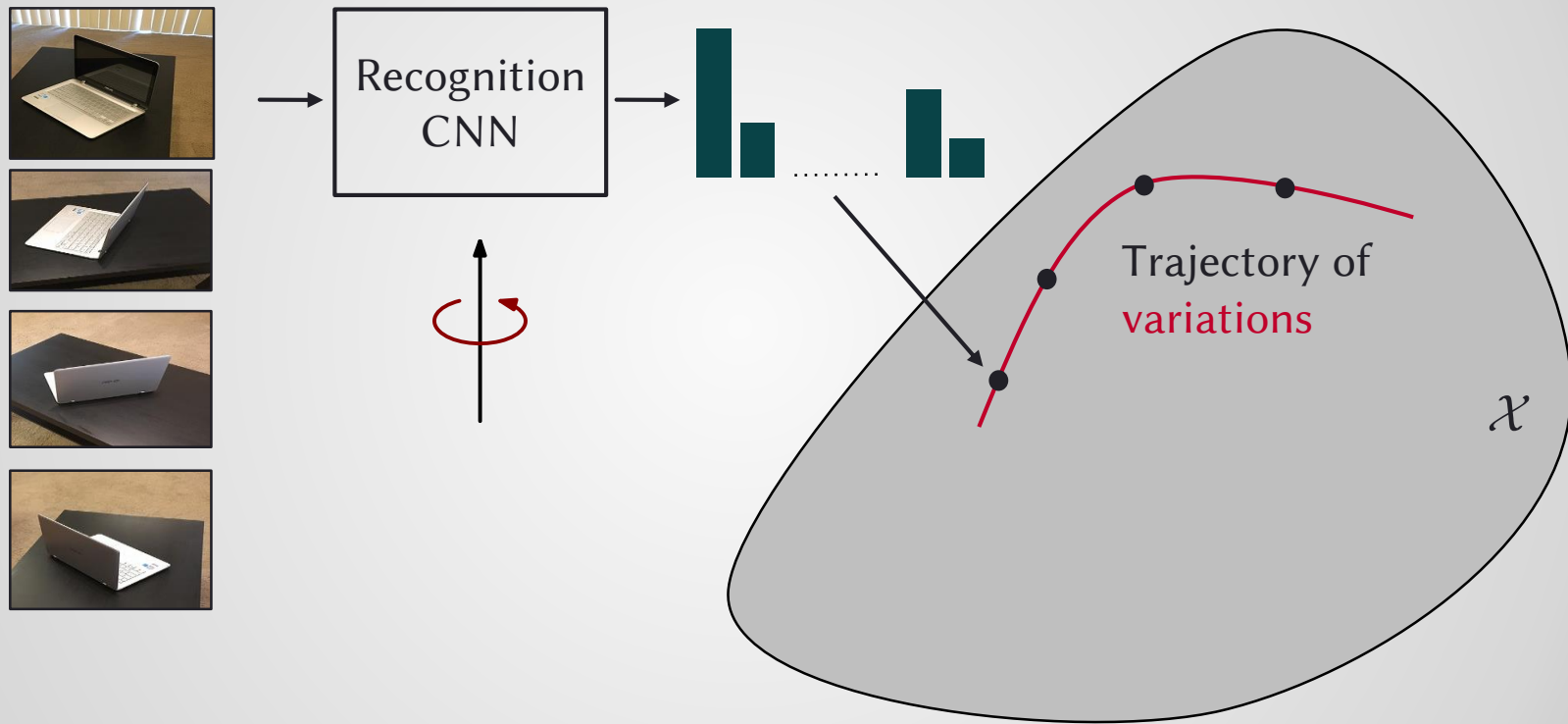
Data Augmentation

Augmentation in **FEATURE space**

- Inherits all the **invariance** of a CNN's representation
- Deep representations “unfold” the **manifold of images** [Bengio et al. 2012] (e.g., very recently leveraged in [Upchurch et al. 2016])
- Augmentation is performed in a space where trajectories of variation are “easier” to learn (e.g., with **less data**)

This is the strategy we advocate in this work!

A Regression / Synthesis Problem



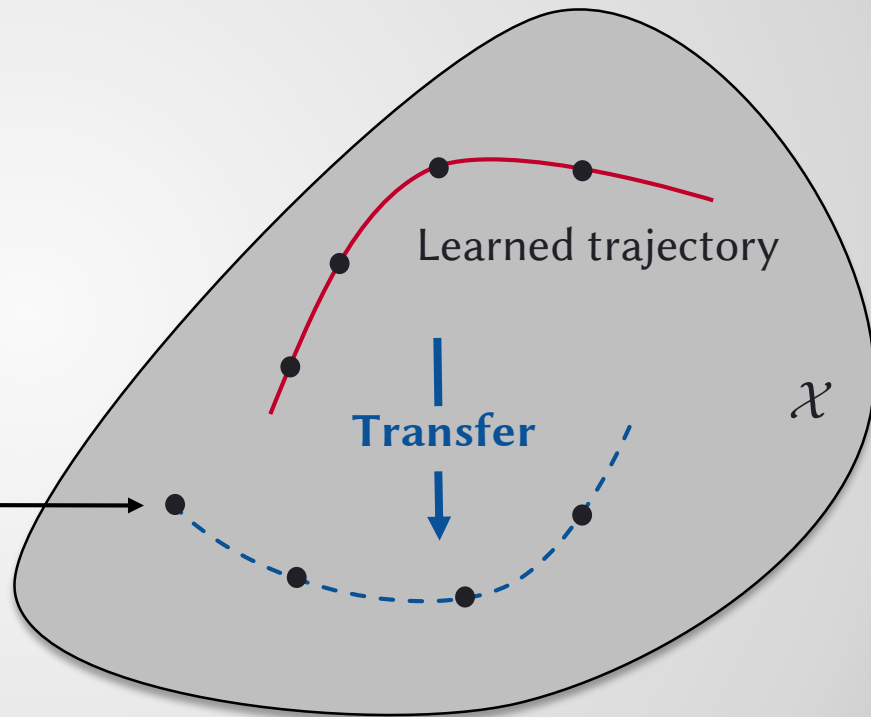
Learn trajectories in feature space

CNN activation space

A Regression / Synthesis Problem



Unseen Object

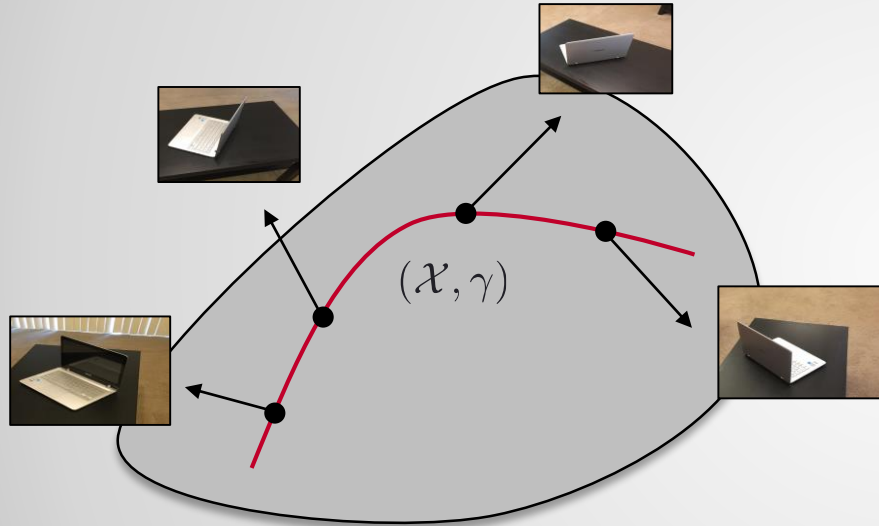


Transfer trajectories to unseen objects

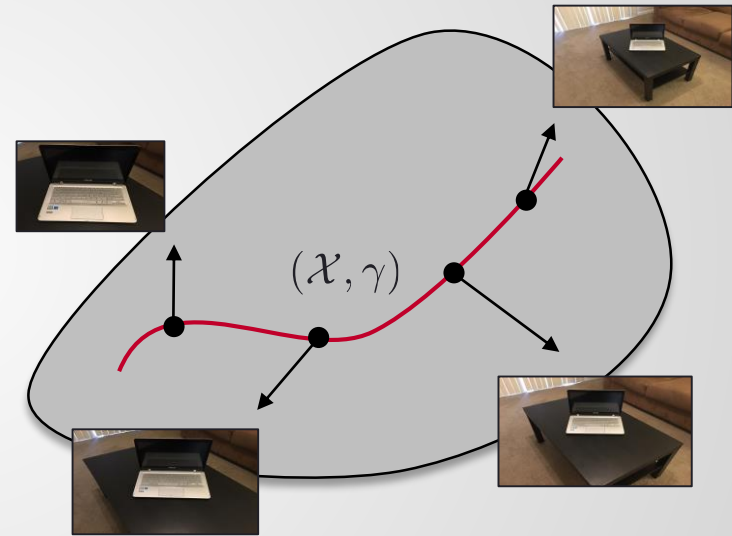
CNN activation space

Assumption

Object **pose**



Object **depth**



- Trajectories are parametrized by object **pose** and **depth**: **Attributes**
- Attributes assume a **scalar value** denoted by γ

Problem Formulation



$$\gamma : \mathcal{X} \rightarrow \mathbb{R}_+, \quad \mathbf{x} \mapsto \gamma(\mathbf{x})$$

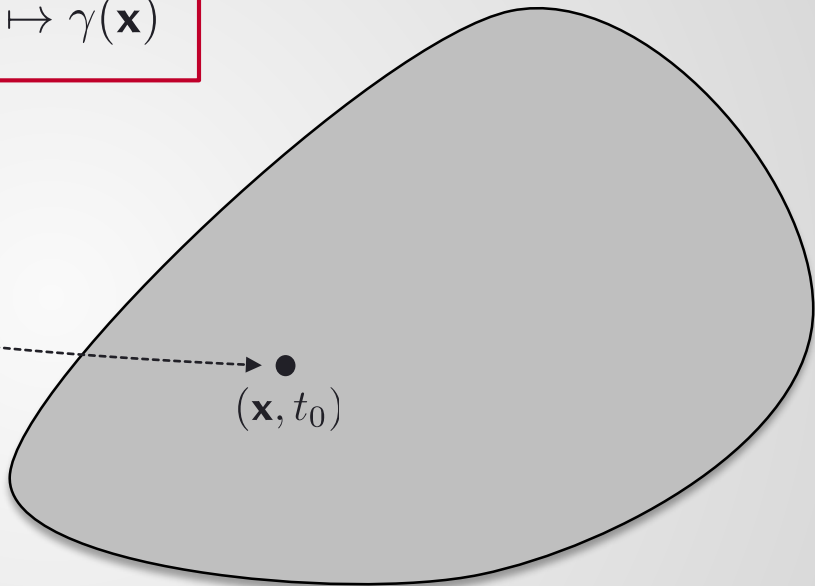
Object CNN

$$\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$$

Attribute predictor

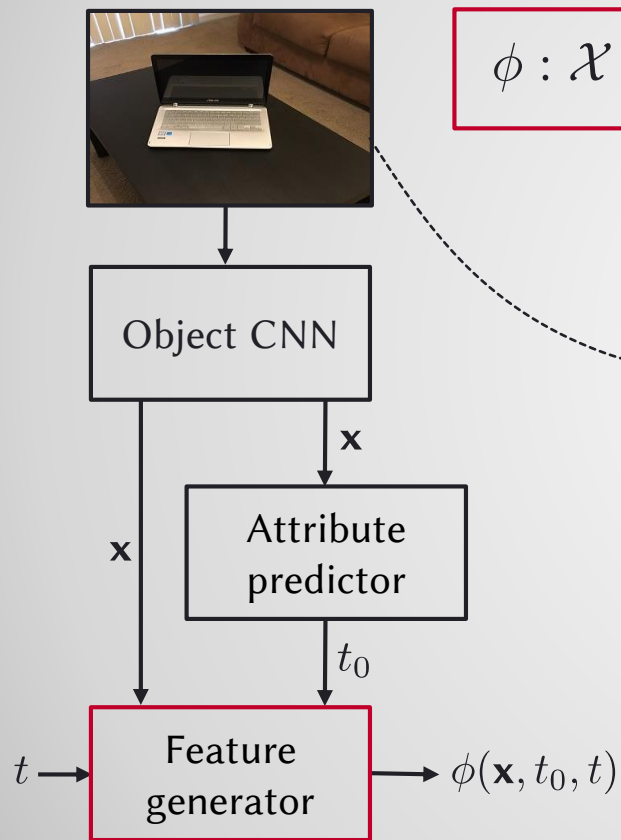
e.g., depth 2 [m]

$$\gamma(\mathbf{x}) = t_0 \in \mathbb{R}_+$$

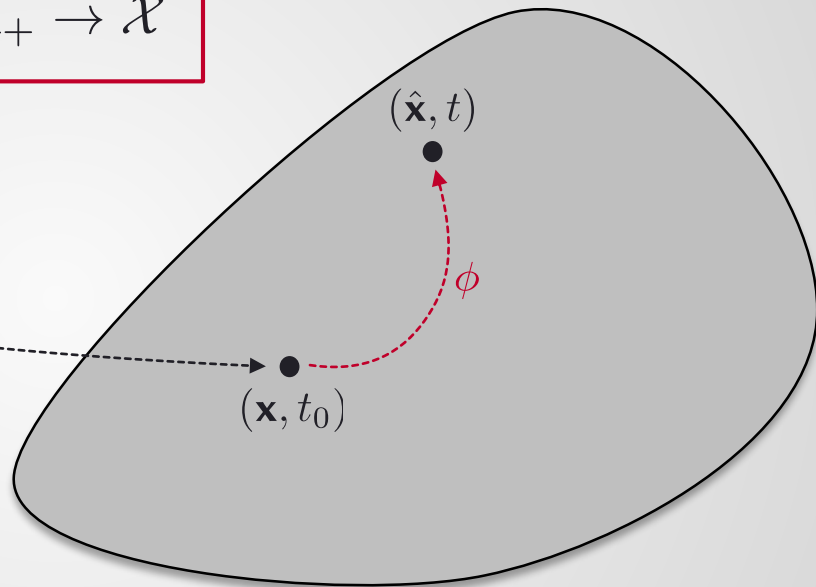


An **attribute predictor** γ trained on object CNN activations \mathbf{x}

Problem Formulation



$$\phi : \mathcal{X} \times \mathbb{R}_+ \times \mathbb{R}_+ \rightarrow \mathcal{X}$$



A **generator** ϕ that transforms \mathbf{x} s.t. the predicted attribute changes from t_0 to a **desired** value t

Problem Formulation



\mathbf{x}

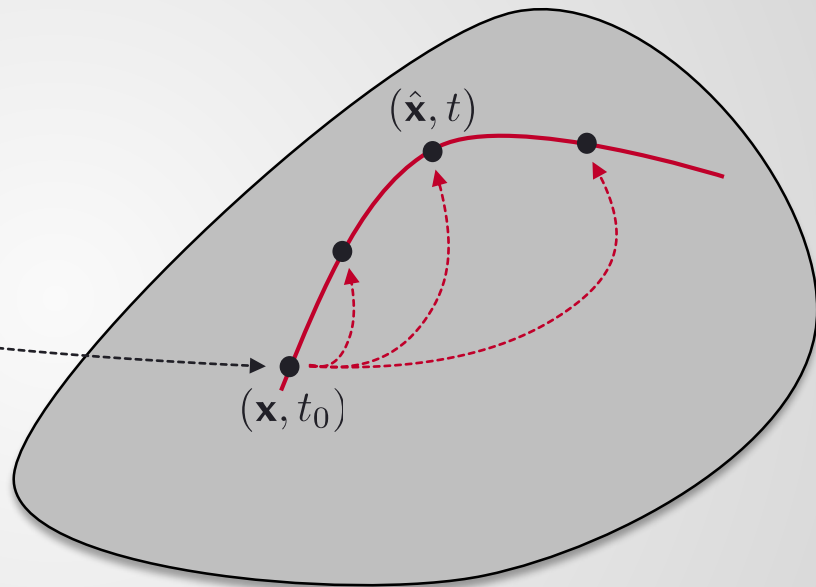
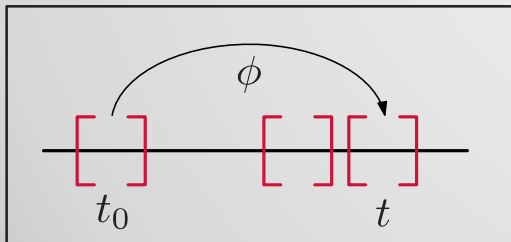
t_0

t

Feature
generator

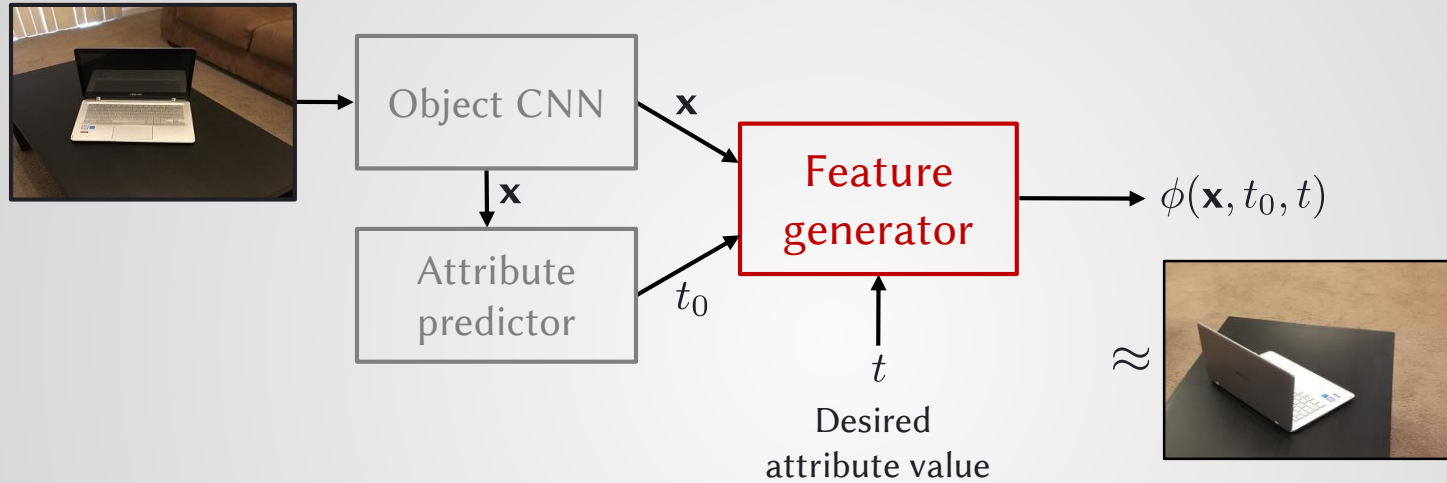
$\phi(\mathbf{x}, t_0, t)$

Attribute range



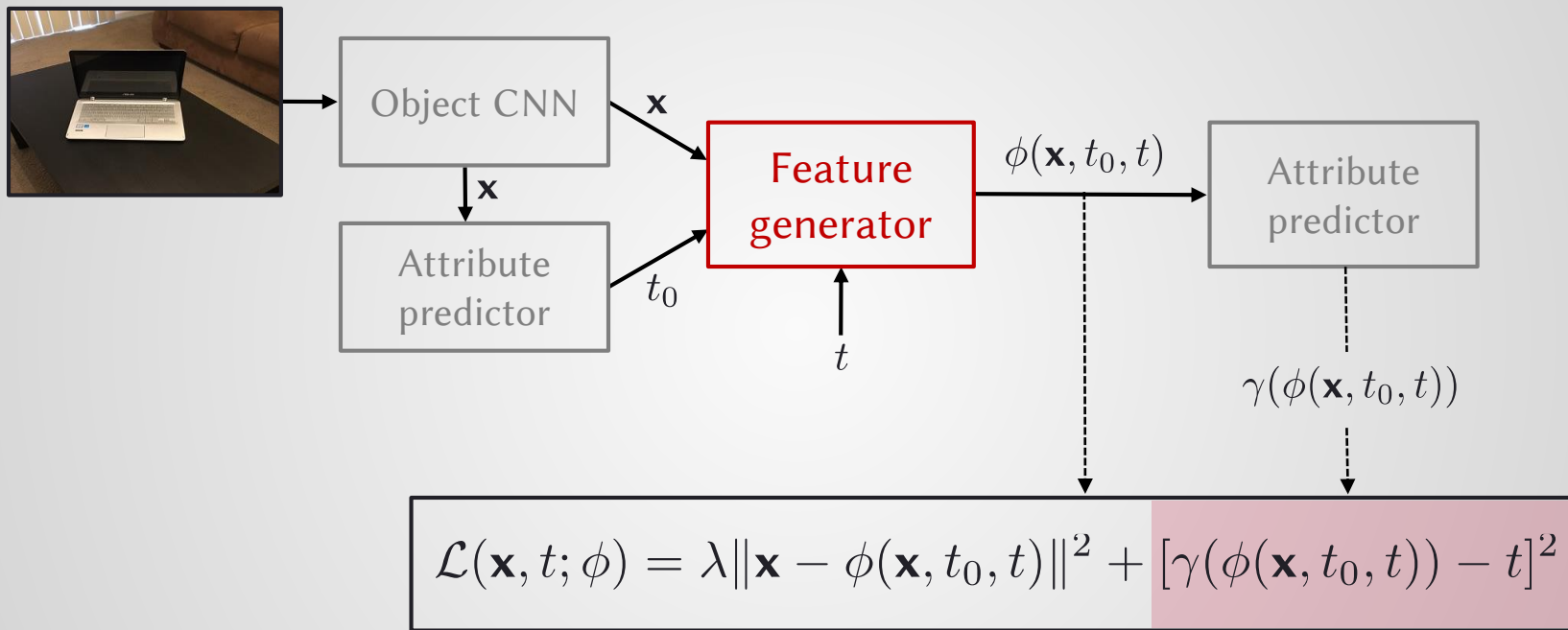
Generators trained for multiple pairs (t_0, t) can produce the **attribute trajectory**

Training for Synthesis



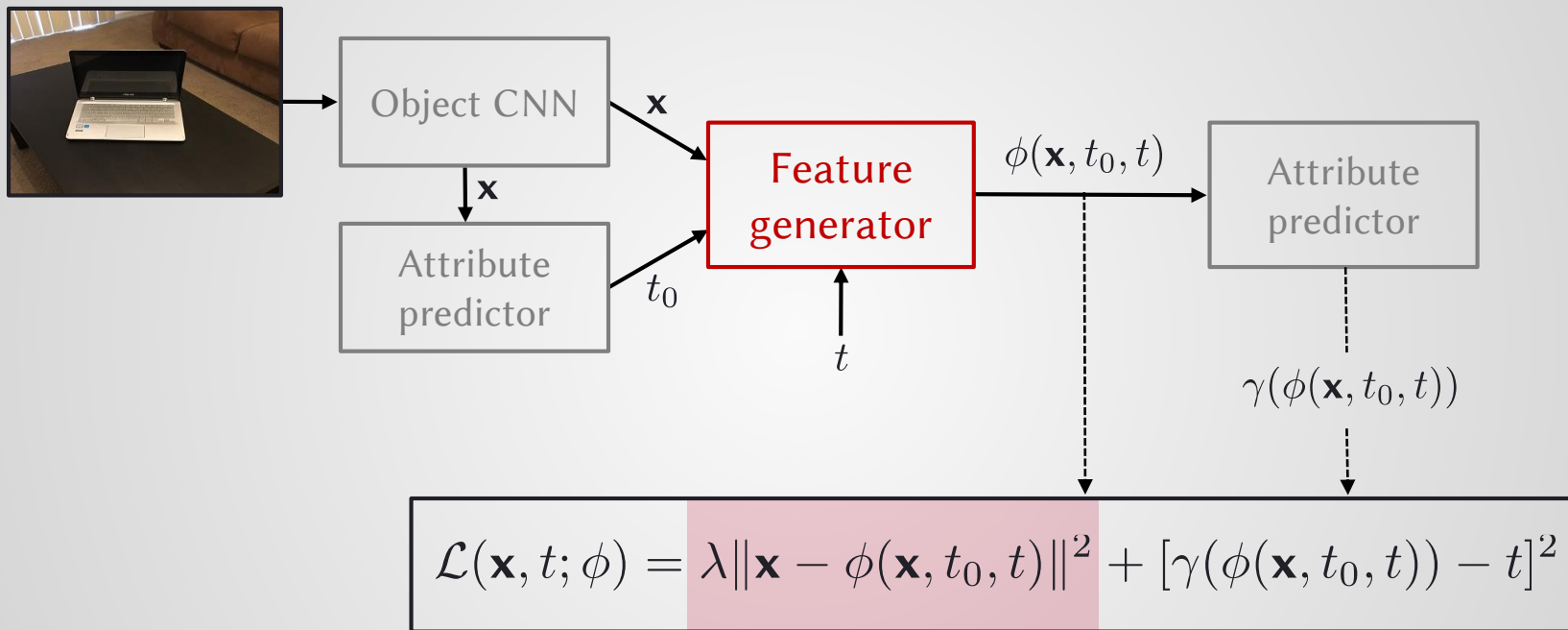
To **train** the generator for a given pair (t_0, t) we keep the **object CNN** and the **attribute predictor** frozen.

Training Loss for Synthesis



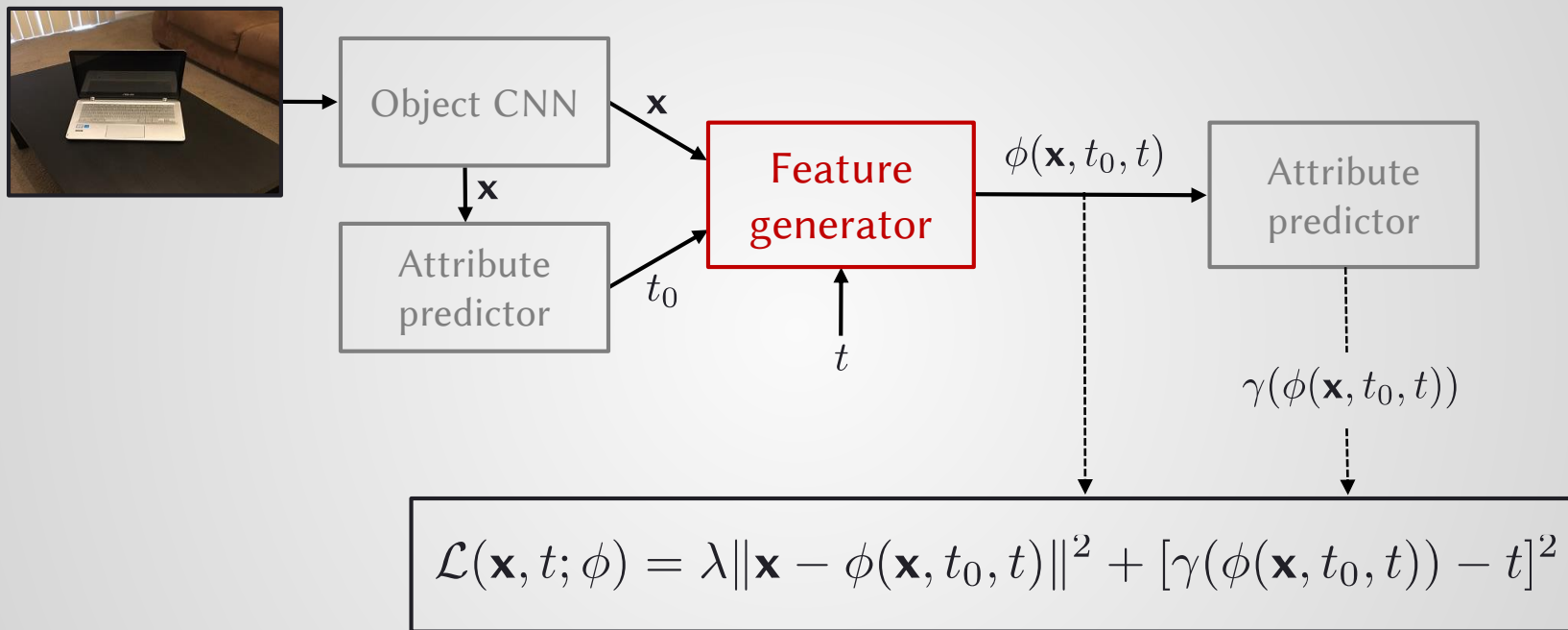
The loss minimizes **attribute mismatch** between the synthesized sample and the desired attribute value.

Training Loss for Synthesis



... and **restrains** the synthesized sample to lie in the neighborhood of the original sample to preserve **object identity**.

Training Loss for Synthesis



Note that learning only needs the given example \mathbf{x} and a scalar value t
No **paired data points**

Training for Synthesis: Implementation

SUN RGBD dataset [Song et al. 2015]



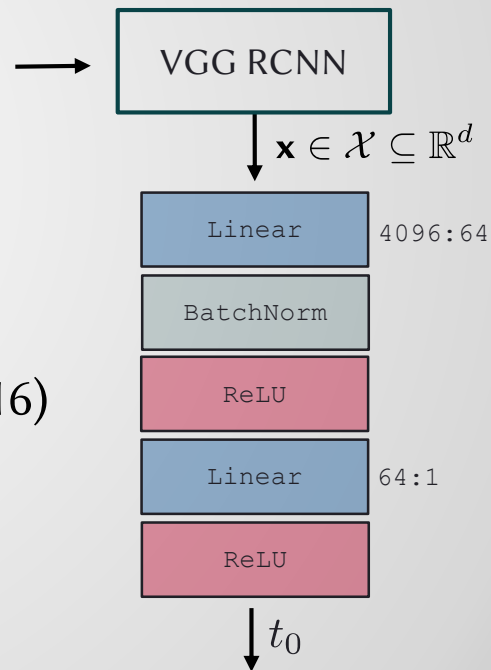
Images of 19 object classes along and their bounding boxes

Attributes: object **pose** and **depth**

- obtained from labeled 3D information

Training for Synthesis: Implementation

SUN RGBD dataset



Training:

Fast-RCNN trained for object detection (VGG 16)

Attribute predictor trained for depth & pose

MLP on RCNN “FC” activations

Training for Synthesis: Implementation

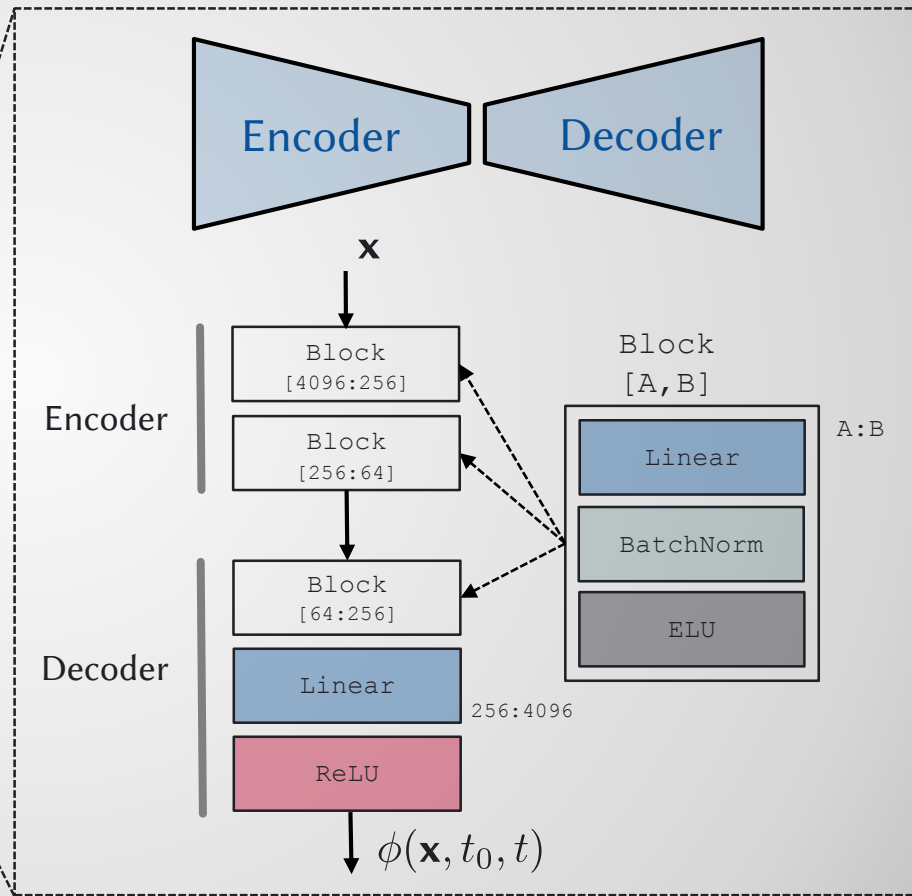
SUN RGBD dataset



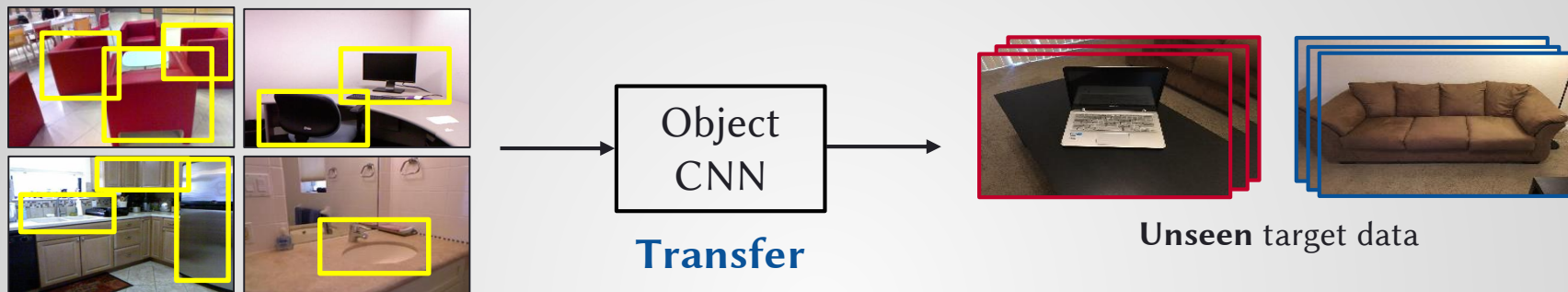
VGG RCNN

Attribute predictor

Feature generator



Application: Transfer-Learning



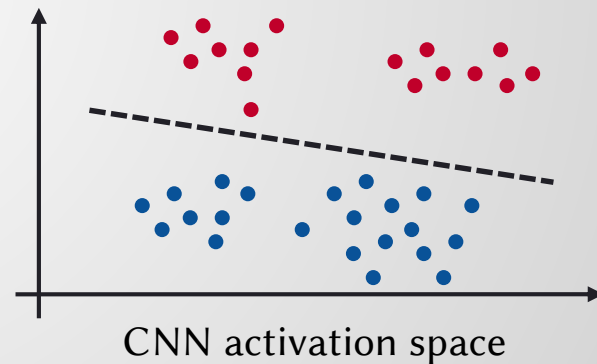
Source dataset

Source data : Object images with attribute labels

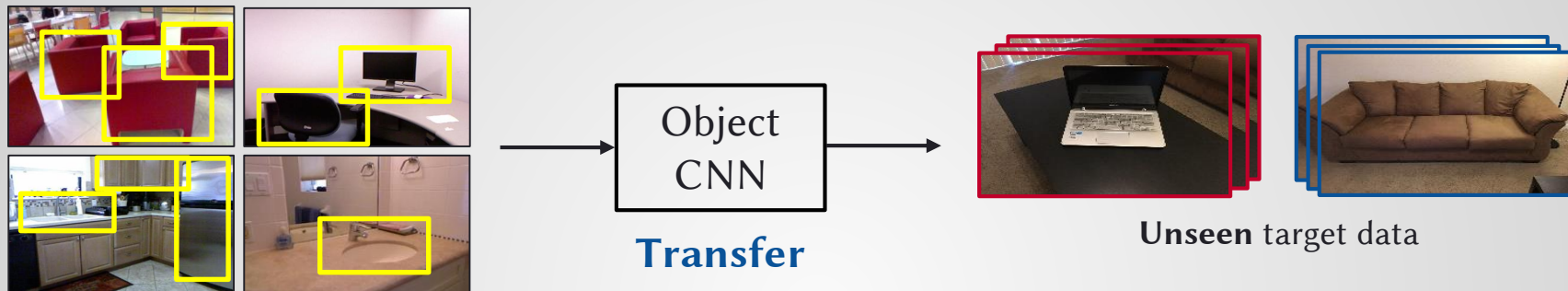
- 19 classes of SUN RGBD + **depth & pose**

Target data : **Unseen** object classes, no annotations

- **10 class subsets** (T1, T2) from SUN RGBD



Application: One / Few-Shot Transfer



Source dataset

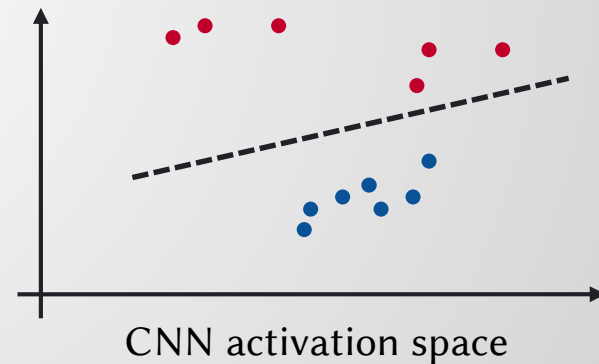
Unseen target data

Source data:

Training RCNN, Attribute predictor, Feature generator

Target data : Used for **one-shot / few-shot** object recognition

- Linear SVM trained with source RCNN activations



Application: One-Shot Object Recognition

For **every image** in **target dataset**, we

1. generate RCNN features: \mathbf{x}
2. predict it's **depth / pose**: t_0
3. synthesize features $\phi(\mathbf{x}, t_0, t)$ for a **range of desired depths / poses** t

Datasets	Baseline*
T1	33.74
T2	23.76
T1 and T2	22.84

*Baseline ... Linear SVM trained on one-shot instances only

Application: One-Shot Object Recognition

For **every image** in **target dataset**, we

1. generate RCNN features: \mathbf{x}
2. predict it's **depth / pose**: t_0
3. synthesize features $\phi(\mathbf{x}, t_0, t)$ for a **range of desired depths / poses** t

Datasets	Baseline*	Pose Aug.
T1	33.74	37.25
T2	23.76	27.15
T1 and T2	22.84	24.34

*Baseline ... Linear SVM trained on one-shot instances only

Application: One-Shot Object Recognition

For **every image** in **target dataset**, we

1. generate RCNN features: \mathbf{x}
2. predict it's **depth / pose**: t_0
3. synthesize features $\phi(\mathbf{x}, t_0, t)$ for a **range of desired depths / poses** t

Datasets	Baseline*	Pose Aug.	Depth Aug.
T1	33.74	37.25	38.32
T2	23.76	27.15	28.49
T1 and T2	22.84	24.34	25.52

*Baseline ... Linear SVM trained on one-shot instances only

Application: One-Shot Object Recognition

For **every image** in **target dataset**, we

1. generate RCNN features: \mathbf{x}
2. predict it's **depth / pose**: t_0
3. synthesize features $\phi(\mathbf{x}, t_0, t)$ for a **range of desired depths / poses** t

Datasets	Baseline*	Pose Aug.	Depth Aug.	D + P.	
T1	33.74	37.25	38.32	39.10	+5.36
T2	23.76	27.15	28.49	30.12	+6.36
T1 and T2	22.84	24.34	25.52	26.67	+3.83

*Baseline ... Linear SVM trained on one-shot instances only

Application: Few-Shot Object Recognition

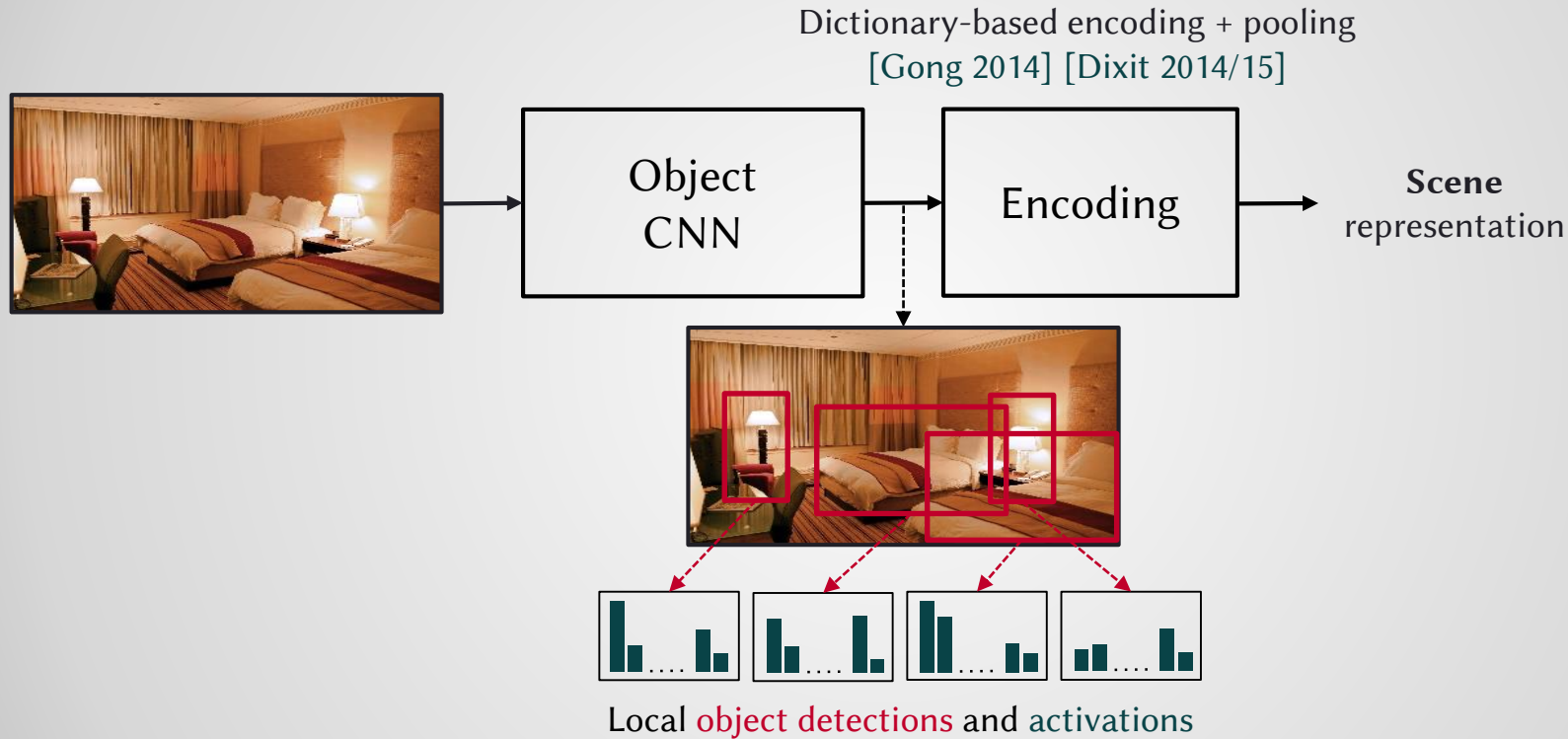
We repeat the previous experiment with **5 instances per target class**

Datasets	Baseline*	Pose Aug.	Depth Aug.	D + P.	
T1	50.03	55.04	53.83	56.92	+6.89
T2	36.76	44.57	42.68	47.04	+10.28
T1 and T2	37.37	40.46	39.36	42.87	+5.50

In summary, we observe gains similar to one-shot (in the range of **5-10 points**)

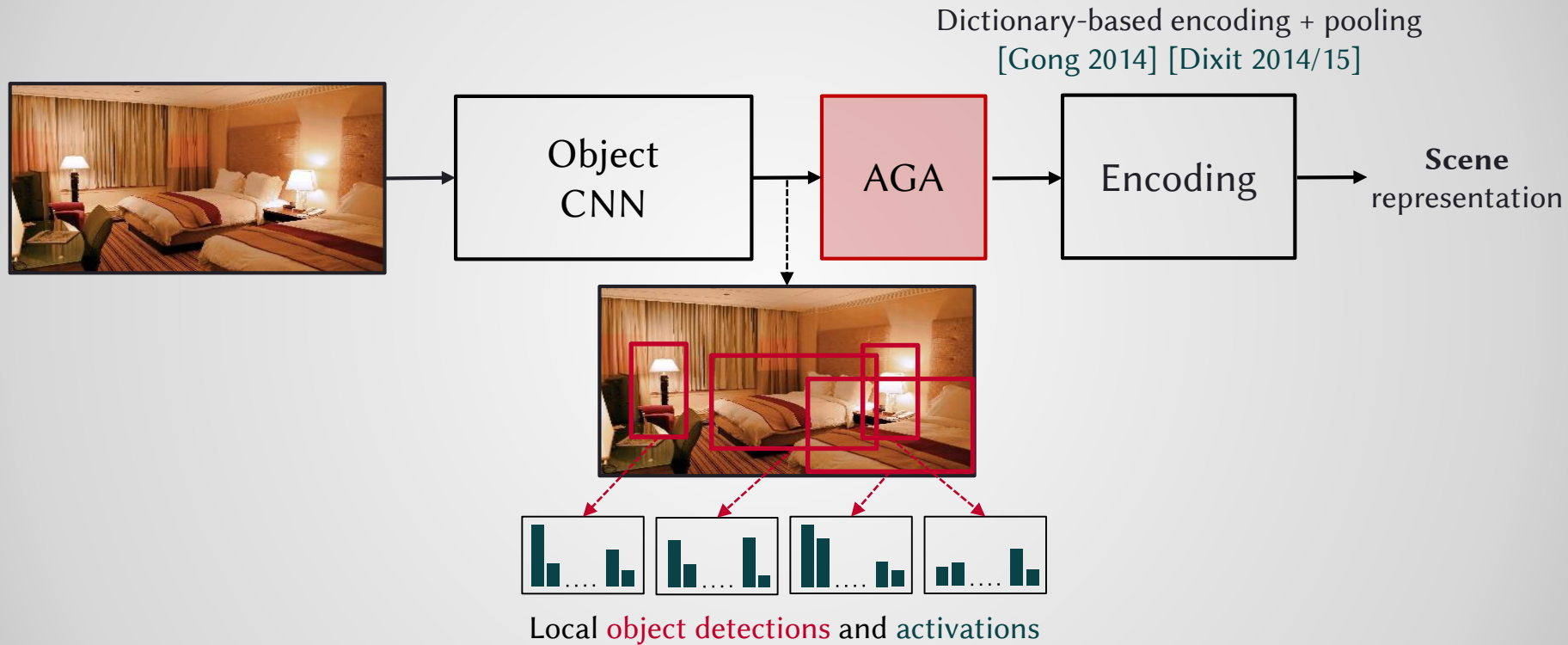
*Baseline ... Linear SVM trained on one-shot instances only

Object-based Scene Representation



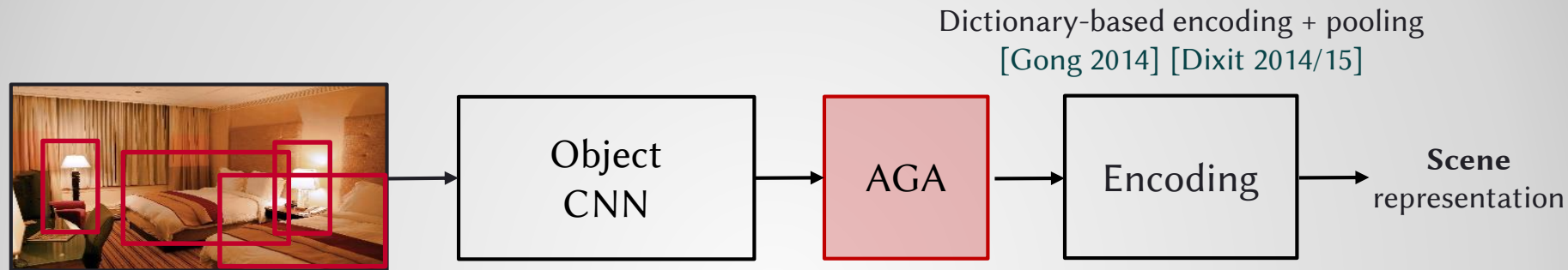
Problem: **weak encoding** due to few detections / image (more so in few-shot case)

Object-based Scene Representation



Deploy **pre-trained pose & depth guided synthesis** to multiply the features

Application: One-shot Scene Recognition



Source Data: SUN RGBD

- Train feature synthesis

Target Data: Subset of MIT Indoor Scenes [Quattoni et al. 2012]

- Local feature extraction + **depth / pose guided synthesis**
- Scene Representation (Fisher vector) [Perronnin et al. 2010]
- Combined with state-of-the-art rep.: **Places** [Zhou et al. 2014], **Sem-FV** [Dixit et al. 2015]

Application: One-shot Scene Recognition

Approaches

Accuracy (%)

Sem-FV [Dixit et al. 2015]

32.75

AGA-augmented Sem-FV

34.36

Application: One-shot Scene Recognition

Approaches	Accuracy (%)
Sem-FV [Dixit et al. 2015]	32.75
AGA-augmented Sem-FV	34.36
Places CNN [Zhou et.al. 2014]	51.28
AGA-augmented Places CNN	52.11

Accuracy of **Sem-FV** and **Places CNN** representations improves non-trivially!

Conclusion

- We propose a technique for **attribute-guided** data augmentation (**AGA**)
- Augmentation in feature space of a CNN
 - **Low complexity** learning
 - Inherits the **invariance** of a pre-trained CNN
- Augmentation: **Regression** along attribute trajectories
 - Trajectories once learned can be **transferred** to unseen objects
- Attribute trajectory learning
 - Attribute predictor MLP
 - Feature regressor MLP
 - **Training needs no paired examples**
- Augmentation improves one shot/few-shot recognition

Thank You!

Source code (+ trained models for pose / depth) is [publicly available!](#)



This work is supported by **NSF** awards IIS-1208522, CCF-0830535 and ECCS-1148870.
We also thank **NVIDIA** for the generous donation of TitanX GPUs.

Supplementary Material

Implementation

- **Framework:** Torch
- **Optimizer:** ADAM [Kingma & Ba, 2015]

	Attribute strength predictor	Encoder-Decoder* (per attribute interval [a,b] to desired target t)
Learning rate	0.001	0.001
Batch size	300	300
Epochs	50	50
Loss	MSE	Weighted MSE (0.7 regularizer + 0.3 mismatch)

*pre-trained using all available data

Quality of Synthesized Examples

Objects (T1)	Correlation coefficient	MAE (Depth [m])	Correlation coefficient	MAE (Pose [degrees])
Picture	0.67	0.08	0.65	5.13
Ottoman	0.70	0.09	0.70	4.41
Whiteboard	0.67	0.12	0.65	4.43
Fridge	0.69	0.10	0.68	4.48
Counter	0.76	0.08	0.77	3.98
Book	0.74	0.08	0.73	4.26
Stove	0.71	0.10	0.71	4.50
Cabinet	0.74	0.09	0.72	3.99
Printer	0.73	0.08	0.72	4.59
Computer	0.81	0.06	0.80	3.73
Average	0.72	0.09	0.71	4.35

Quality of Synthesized Examples

Retrieval experiment

- Using **synthesized samples** we retrieve the **nearest real sample** from target database.
- Results expressed in
 - **Top-1 retrieval accuracy** (object class)
 - **Coeff. of determination** (attribute strength)
- Reasonable predictability of attr. strength for most objects classes.
- In others, at least **class identity is retained**

Objects (T1)	Top-1	R ²
Picture	0.33	0.36
Ottoman	0.60	0.12
Whiteboard	0.12	0.30
Fridge	0.26	0.08
Counter	0.64	0.18
Books	0.52	0.07
Stove	0.20	0.13
Cabinet	0.57	0.27
Printer	0.31	0.02
Computer	0.94	0.26
Average	0.72	0.09