Stacked Capsule Autoencoders (SCAE)

A. R. Kosiorek, S. Sabour, Y. W. Teh, and G. E. Hinton
University of Oxford, Google Brain, and DeepMind

Compiled by Hongming Shan
May 20, 2020

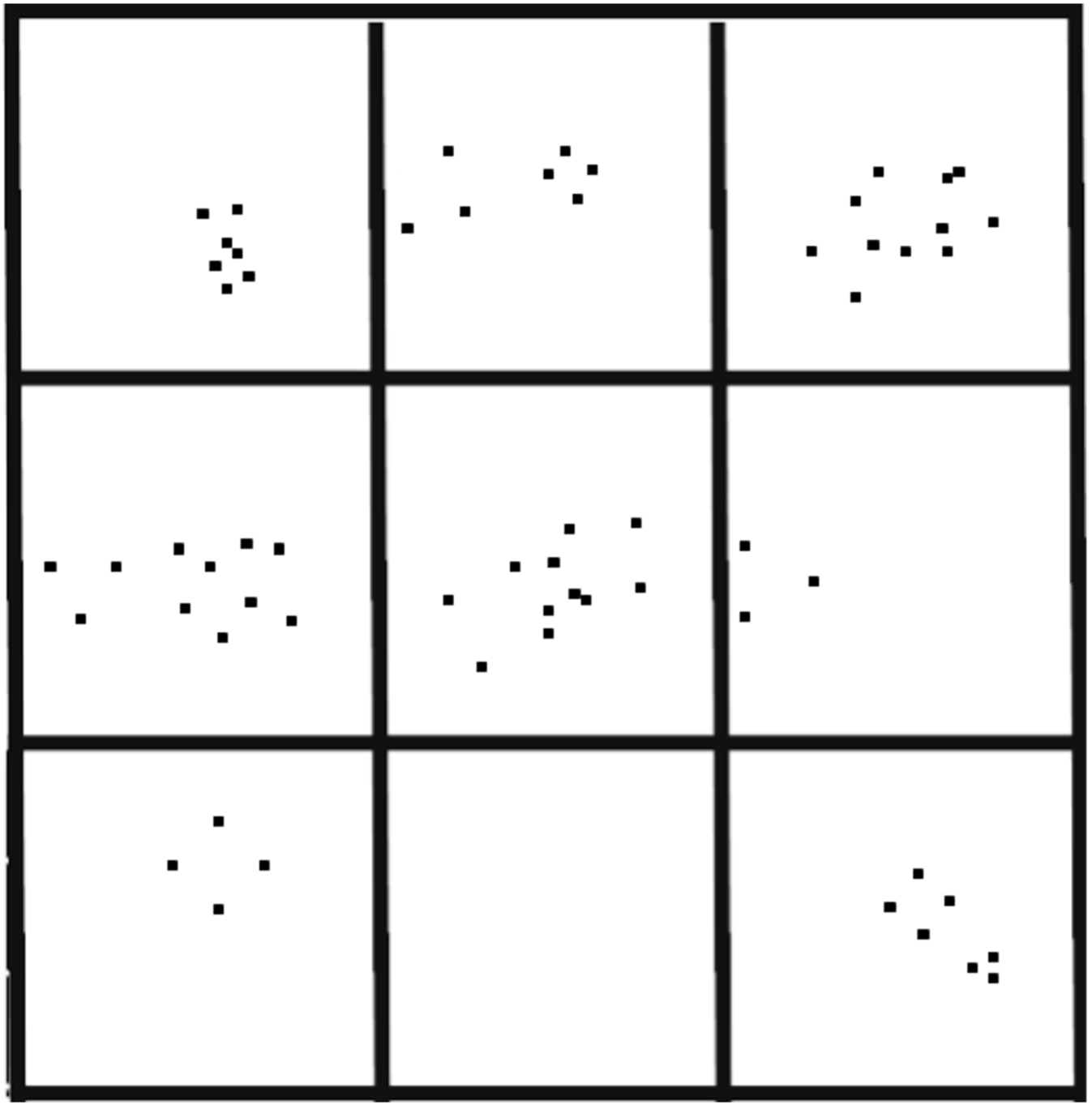
Contributions

- Unsupervised stacked capsule autoencoder (SCAE) can **explicitly** use geometric relationships between parts to reason about objects.
- Since these relationship do not depend on the viewpoint, so the model is **robust** to viewpoint changes.
- SCAE has two stages:
  1. Predicts **presences and poses** of part templates directly from the image and tries to **reconstruct the image** by appropriately arranging the templates.
  2. Predicts **parameters of a few object capsules**, which are then used to reconstruct part poses.
- Object capsule presences are highly informative of the object class, achieving SOTA SVHN (55%) and MNIST (98.7%).

Draw

From

Memory

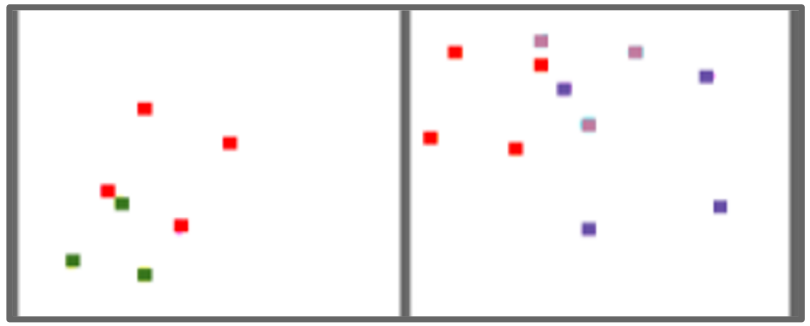


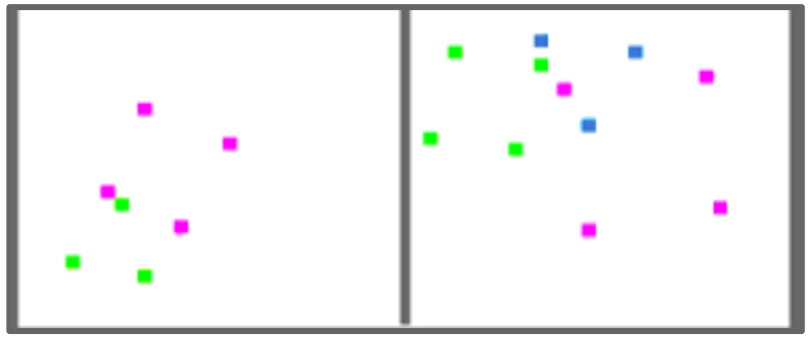
Some slides from https://www.crcv.ucf.edu/cvpr2019-tutorial/slides/intro_sara.pptx

Sparse

Pattern

Recognition





Occlusion

Train

(5,9) (4,9) (8,7) (8,4)

Test

(?,?) (?,?) (?,?) (?,?)

(6,0) (7,1) (8,6) (2,7)

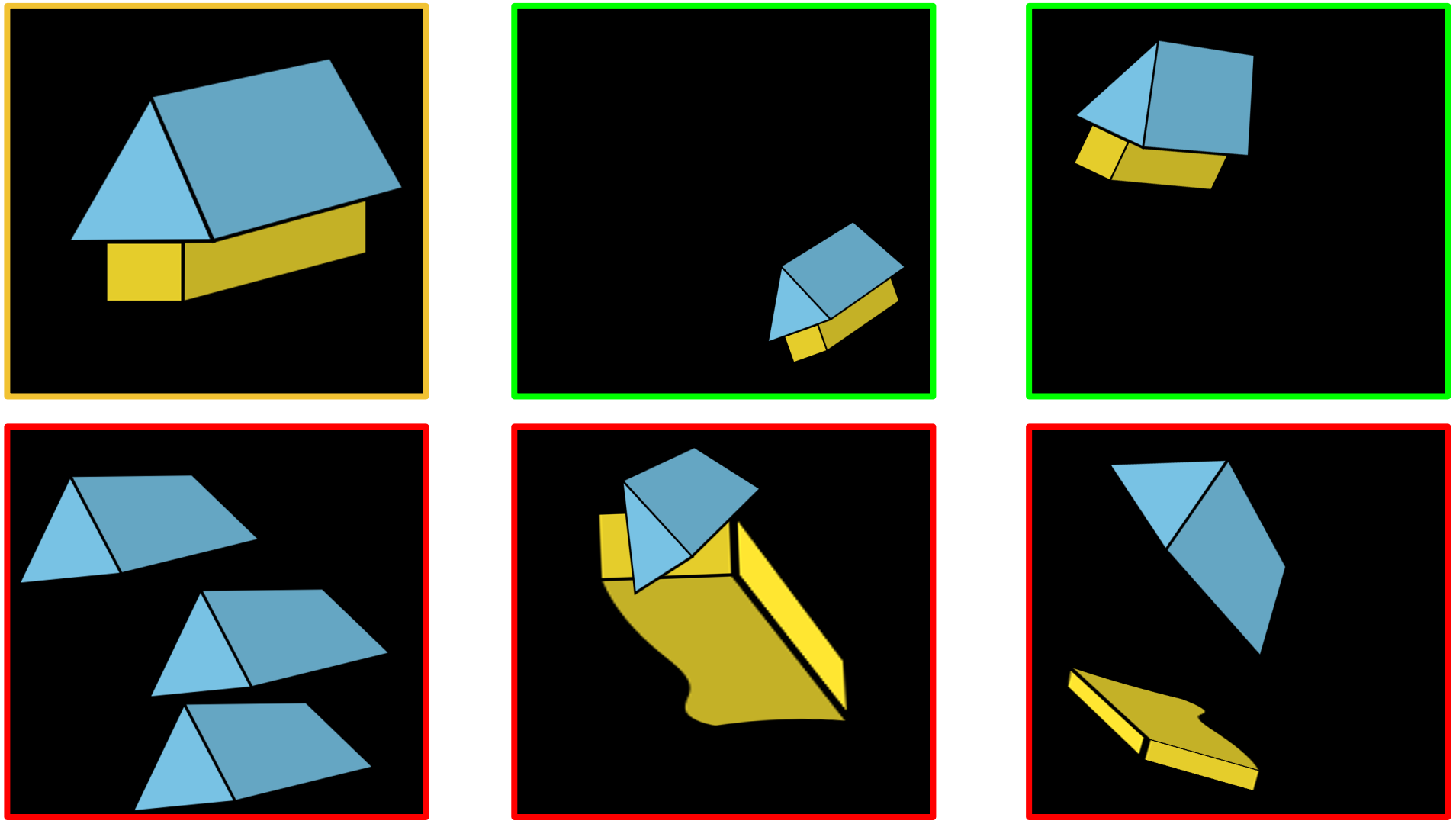
Viewpoint

Train

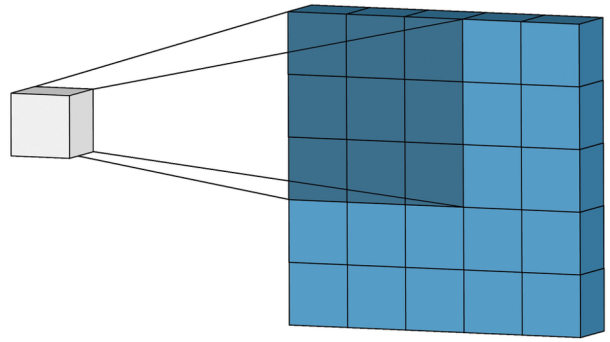
Test

Bag of Features?

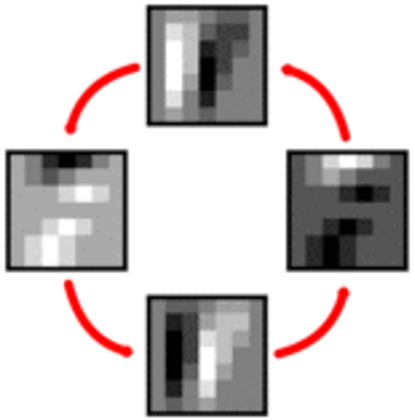
How can we handle different viewpoints?



Viewpoint Equivariance

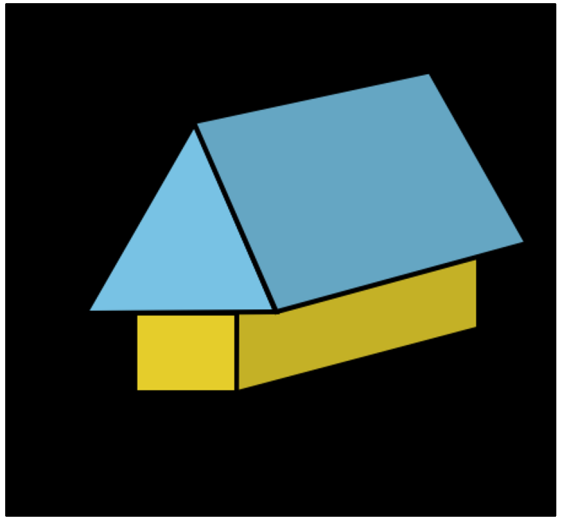


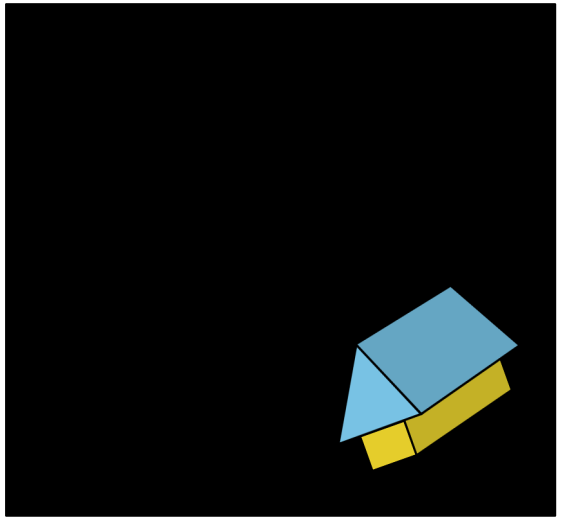
- Convolutional Neural Networks are translation equivariant by gridding.

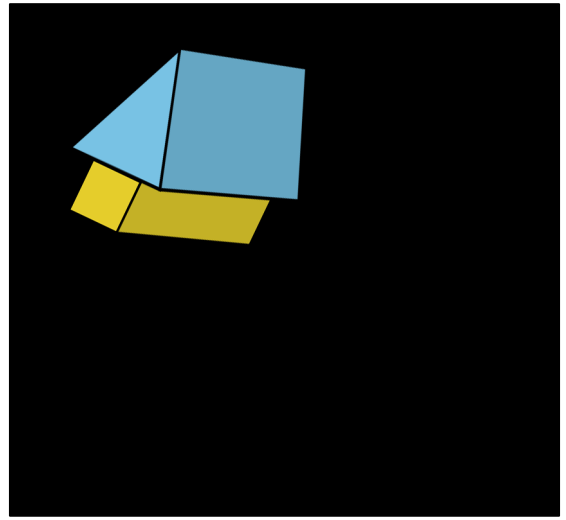


- If we use camera to take a picture from any viewpoints, should we grid the rotation, scale, and other dimensions as well?

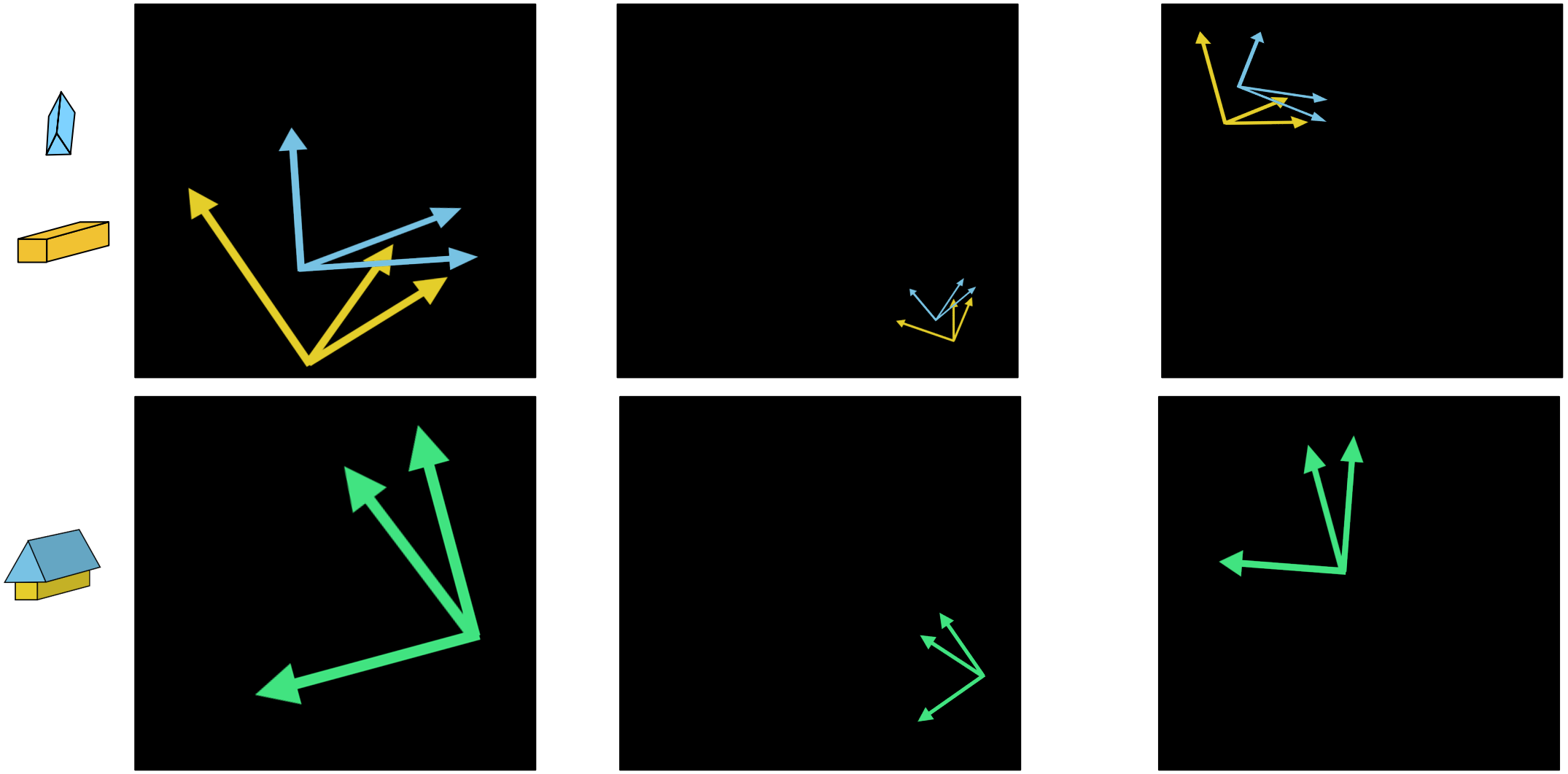
What stays constant?







What stays constant?

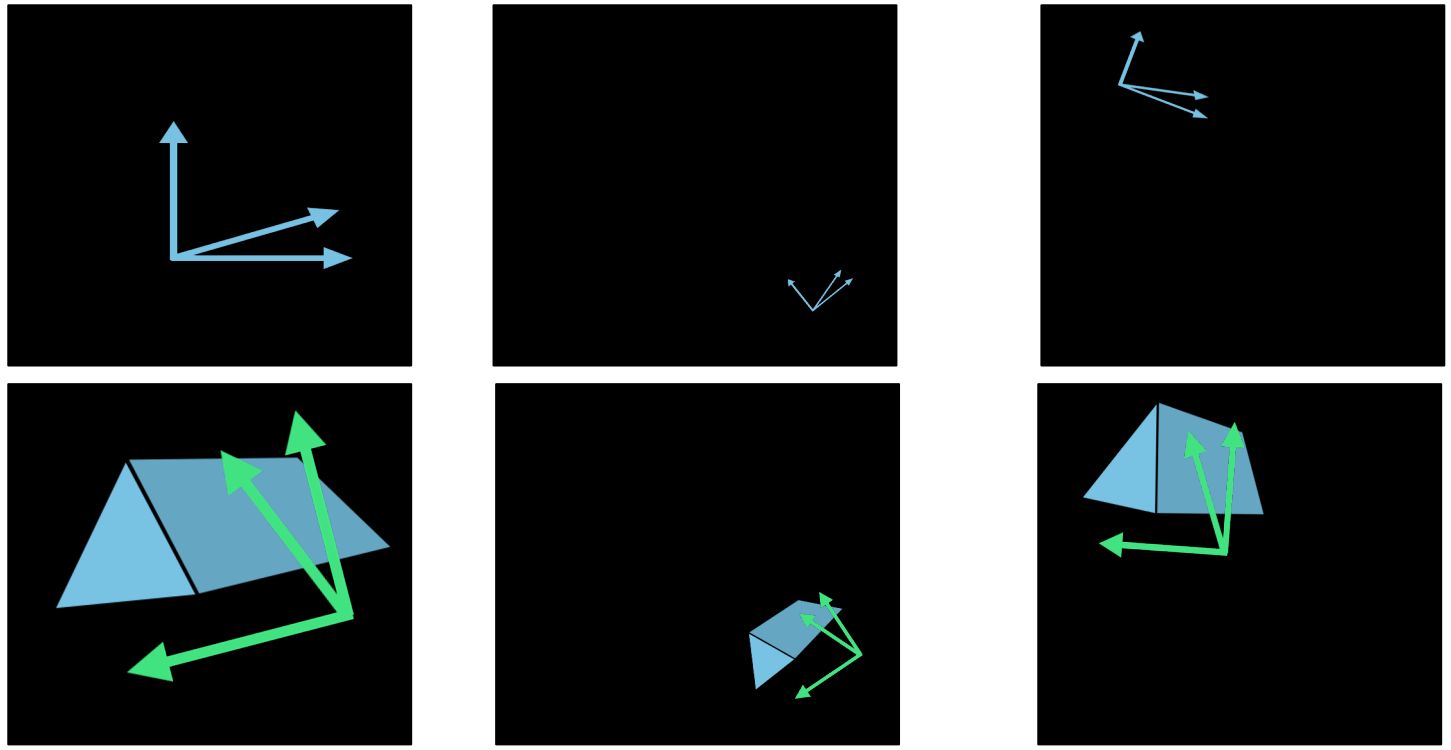


What stays constant?

The relation between coordinate frame of an object and its parts.

Tilt by 90°.

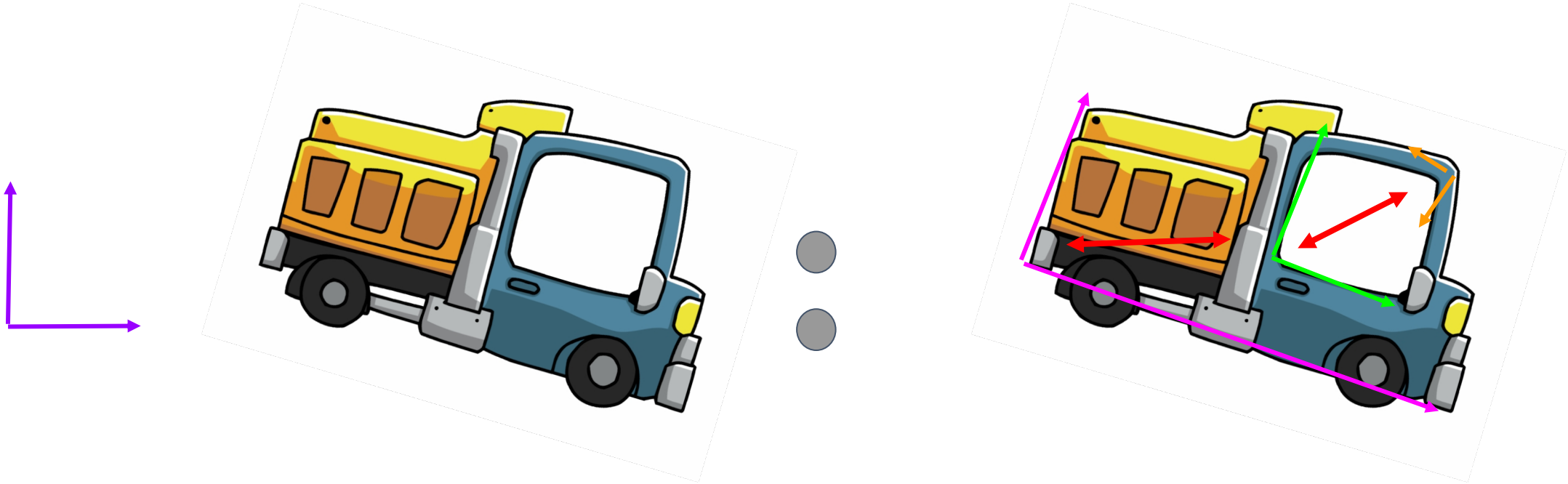
Scale 1.2x.



Coordinate frame

Learn how to transform coordinate frames.

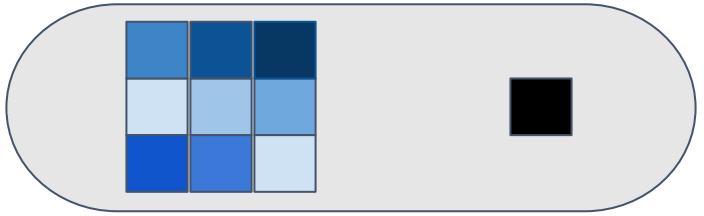
- Coordinate frame: capture scale, position, rotation: Geometrical aspects.
- **Math terms**: Coordinate frame representation is a vector or a matrix (a point in $\mathbb{R}^d$).
- **Math terms**: Coordinate frame relation is a Transformation Matrix.



Capsule

- A group of neurons
- Encapsulating properties of single entity
  1. How the entity exists: Instantiation parameters
  2. Whether the entity exists

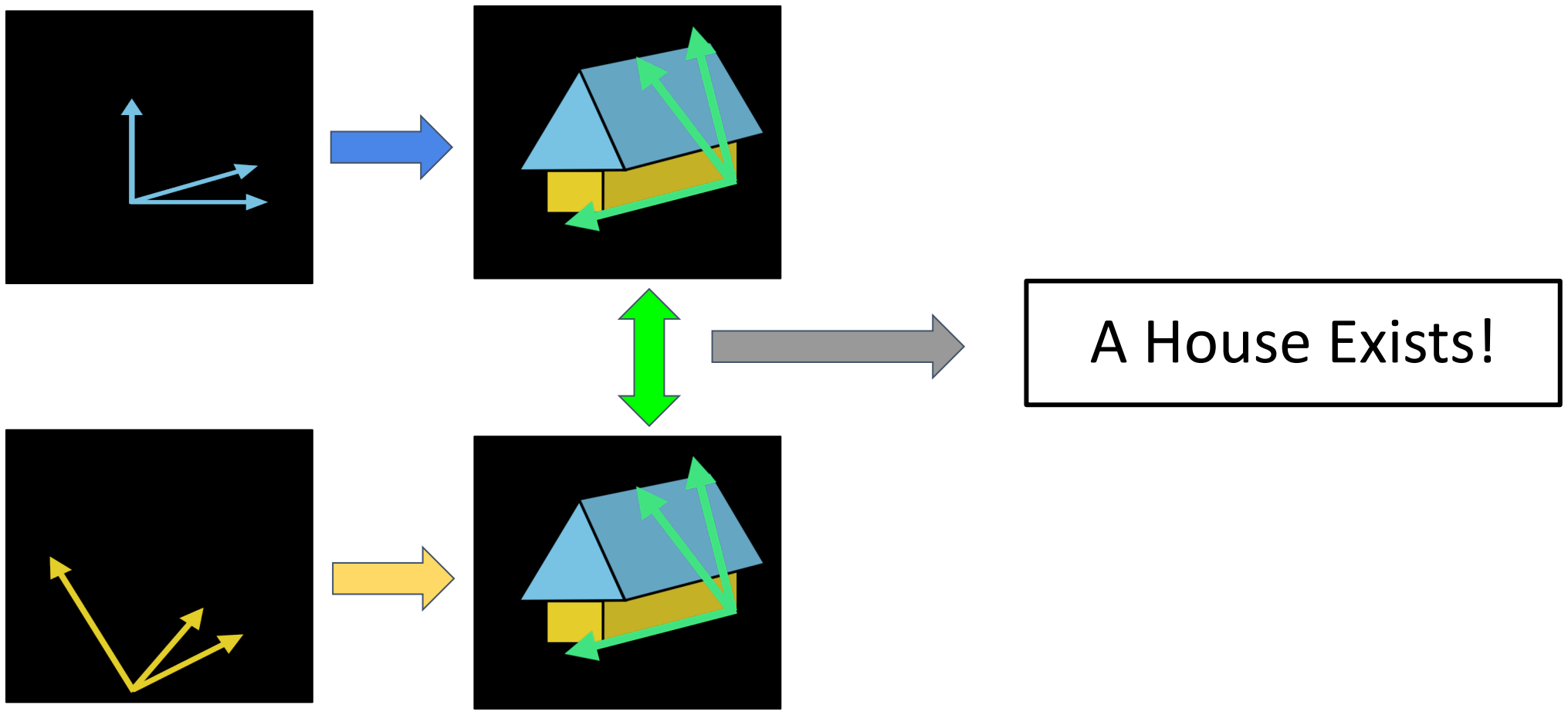
Vector format



Matrix format

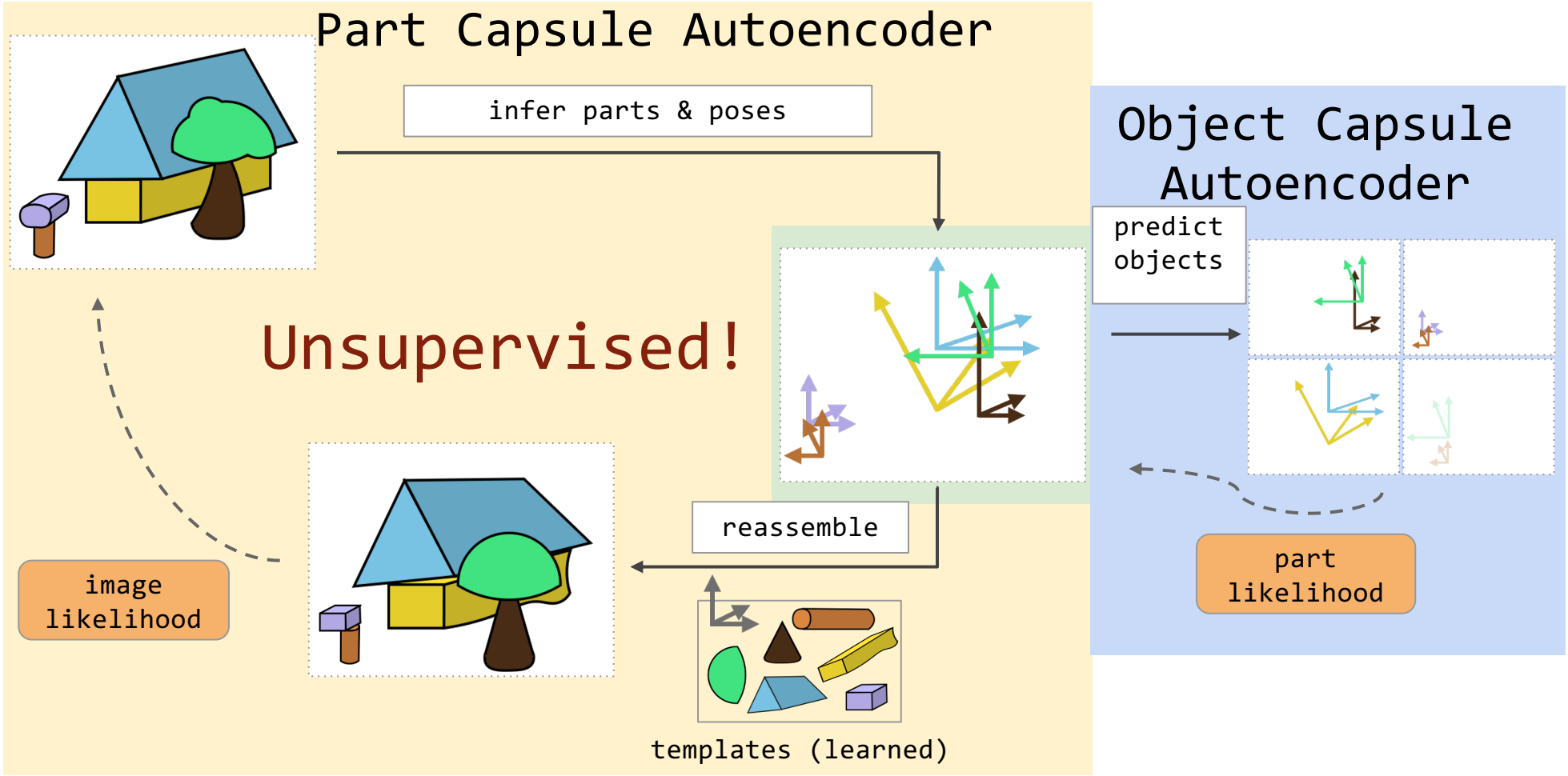
How to detect objects?

- An object exists if there is agreement between **multiple part** predictions.

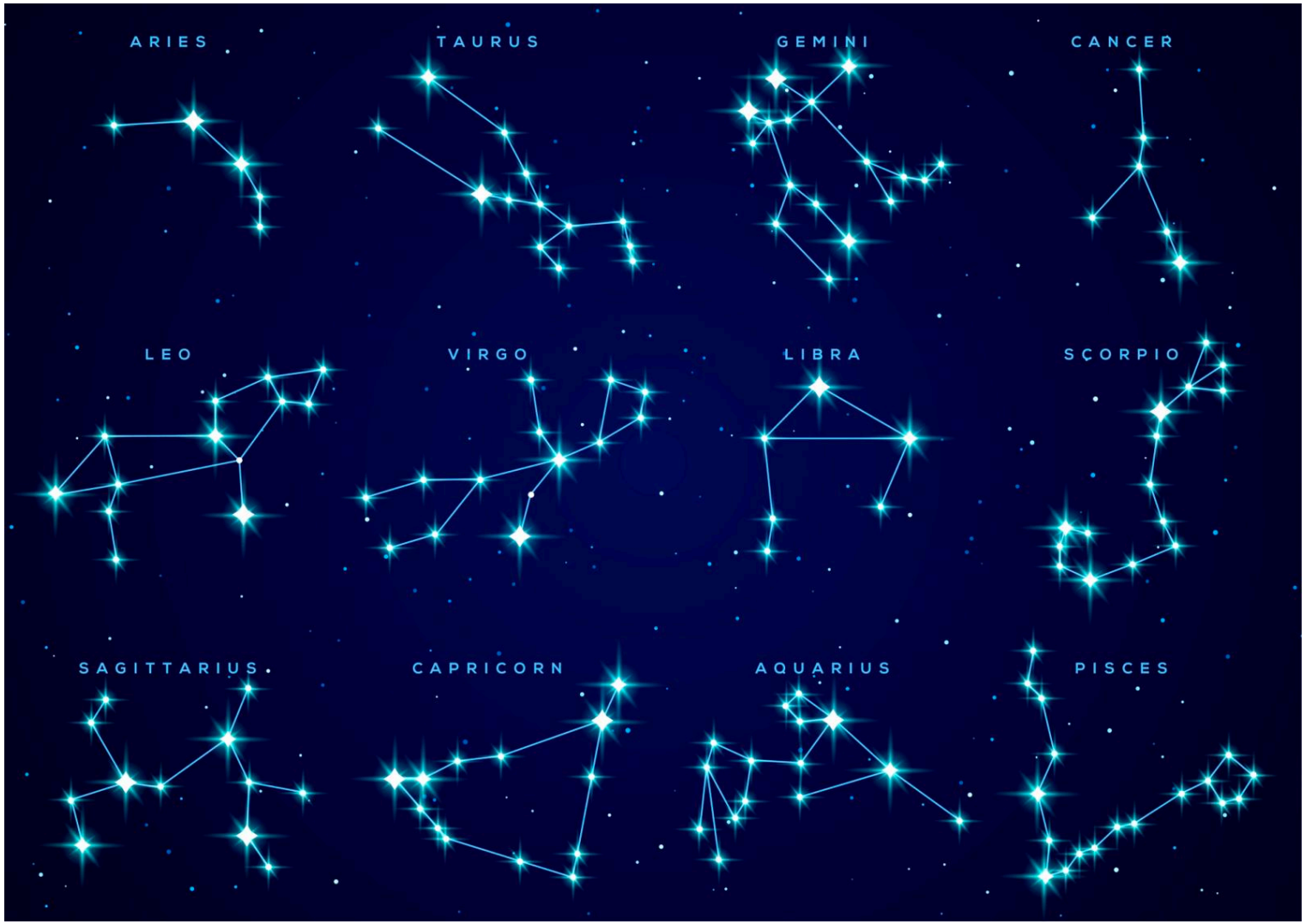


Stacked Capsule Autoencoder

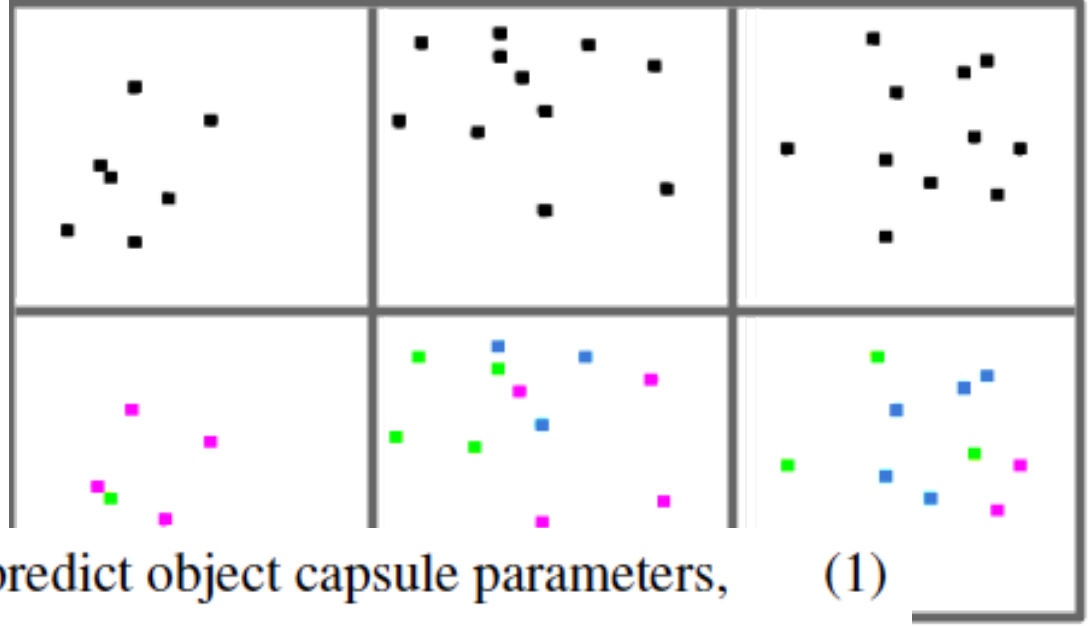
Adam Kosiorek
Yee Whye Teh
Geoff Hinton



Constellation Autoencoder



Constellation Autoencoder

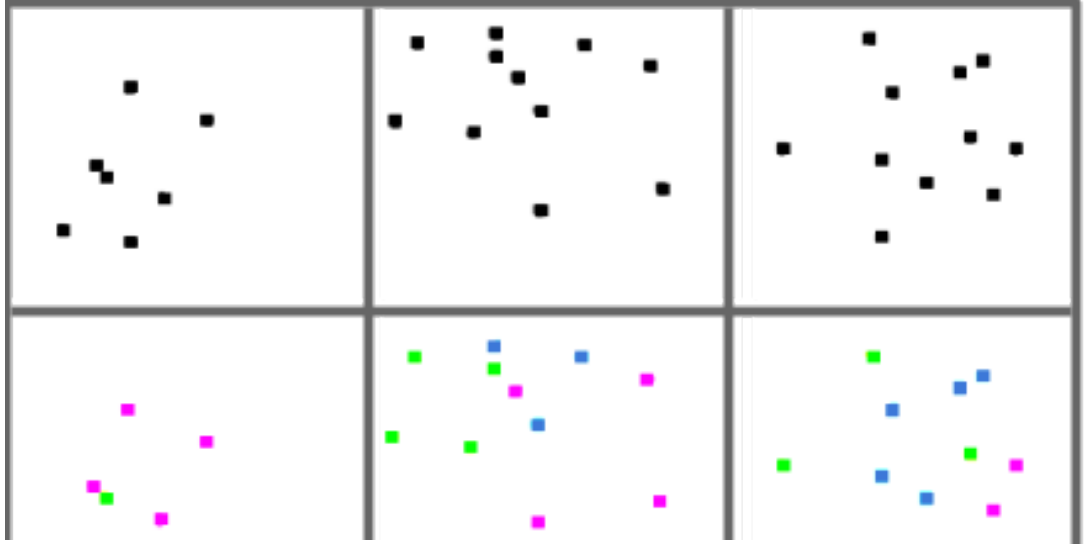


$$\mathrm{OV}_{1:K}, \mathbf{c}_{1:K}, a_{1:K} = \mathrm{h}^{\mathrm{caps}}(\mathbf{x}_{1:M})$$

predict object capsule parameters, (1)

- Let {x_m | m = 1, . . . ,M} be a set of two-dimensional input points, where every point belongs to a constellation
- We first encode all input points (which take the role of part capsules) with Set Transformer—a permutation-invariant encoder h^caps based on attention mechanisms—into K object capsules.
- An object capsule k consists of a capsule feature vector c_k , its presence probability a_k and a 3x3 object-viewer-relationship (OV) matrix, which represents the affine transformation between the object (constellation) and the viewer

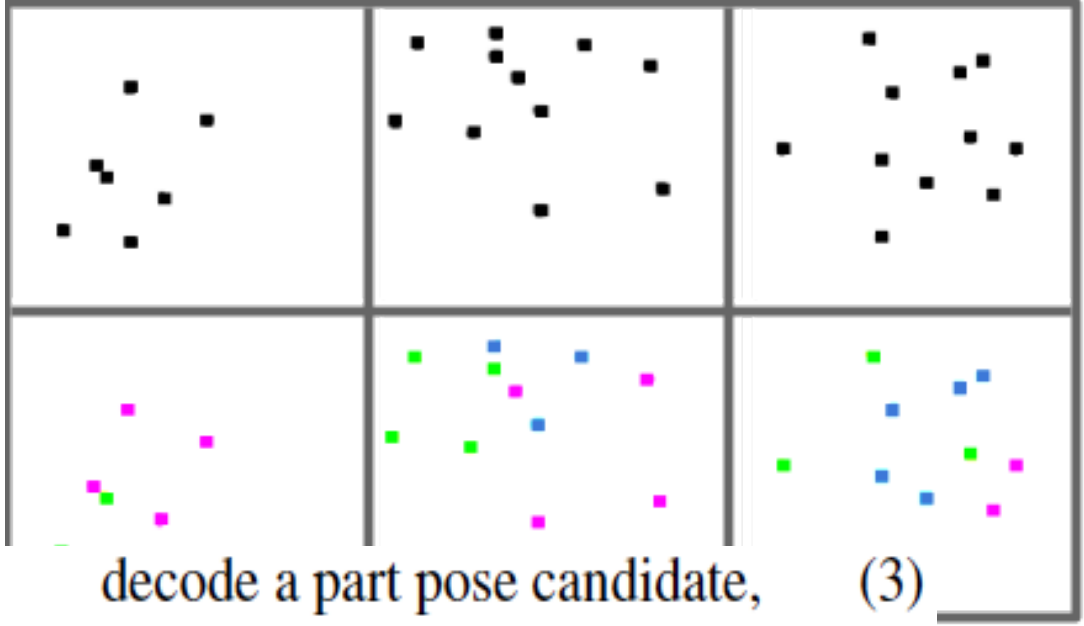
Constellation Autoencoder



$$\mathrm{OP}_{k,1:N}, a_{k,1:N}, \lambda_{k,1:N} = \mathrm{h}_{\mathrm{k}}^{\mathrm{part}}(\mathbf{c}_k) \qquad \text{decode candidate parameters from } c_k\text{'s,} \qquad (2)$$

- Every object capsule uses a separate multilayer perceptron (MLP) h^part_k to predict N <=M part candidates from the capsule feature vector c_k .
- Each candidate consists of the conditional probability a_k,n that a given candidate part exists, an associated scalar standard deviation \lambda_k,n , and a 3x3 object-part-relationship (OP) matrix, which represents the affine transformation between the object capsule and the candidate part3

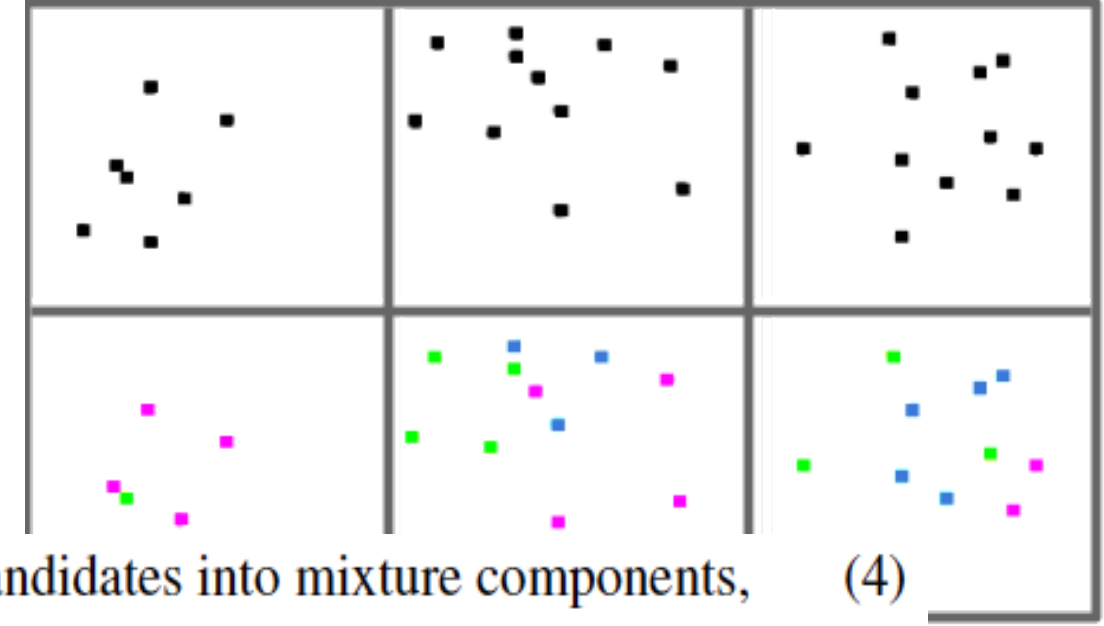
Constellation Autoencoder



$$V_{k,n} = \mathrm{OV}_k \mathrm{OP}_{k,n}$$ decode a part pose candidate, (3)

- Candidate predictions μ_k,n are given by the product of the object capsule OV and the candidate OP matrices

Constellation Autoencoder

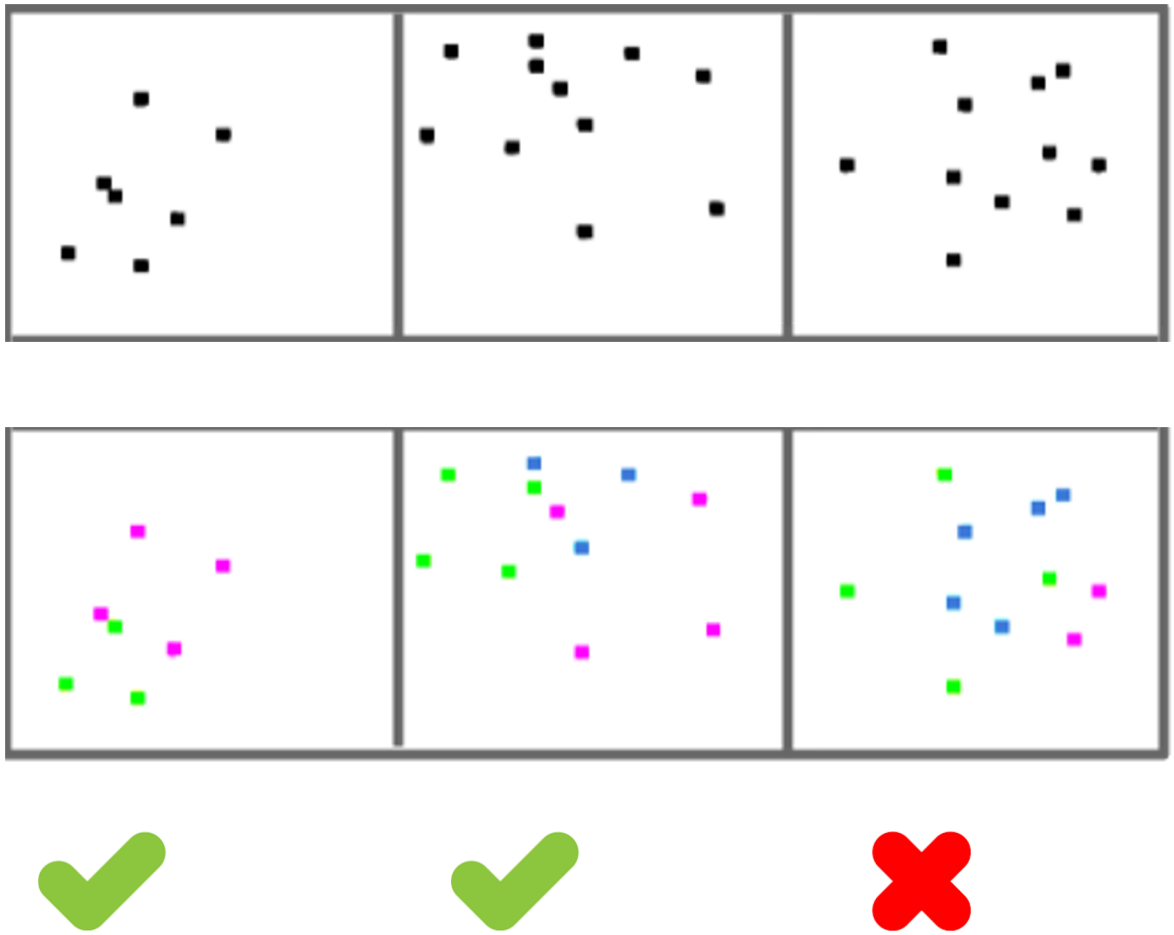


$$p(\mathbf{x}_m \mid k, n) = \mathcal{N}(\mathbf{x}_m \mid \mu_{k,n}, \lambda_{k,n})$$ turn candidates into mixture components, (4)

- We model all input points as a single Gaussian mixture, where μ_k,n and \lambda_k,n are the centres and standard deviations of the isotropic Gaussian components.

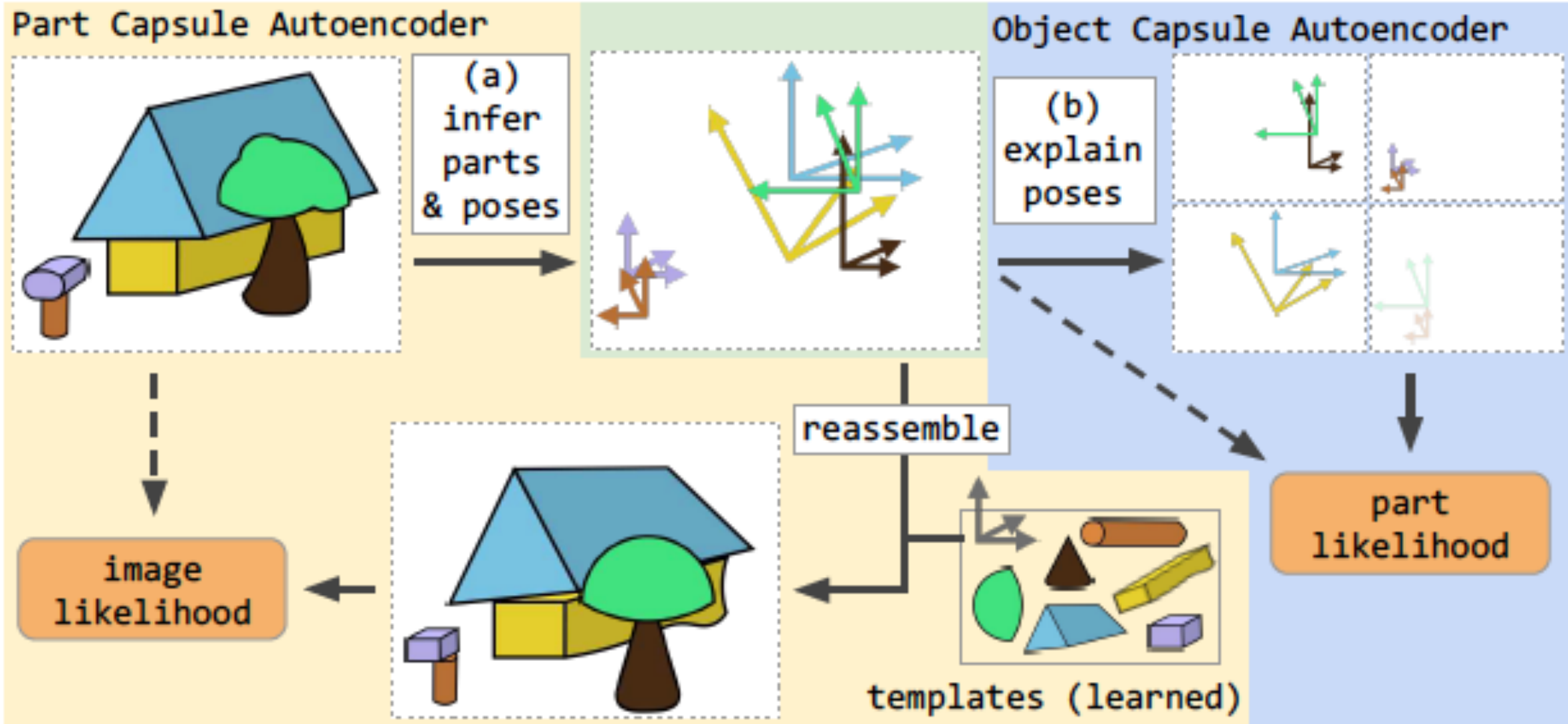
$$p(\mathbf{x}_{1:M}) = \prod_{m=1}^{M} \sum_{k=1}^{K} \sum_{n=1}^{N} \frac{a_k a_{k,n}}{\sum_i a_i \sum_j a_{i,j}} p(\mathbf{x}_m \mid k, n)\,. \tag{5}$$

Constellations



Error:

- Best: 2.8%
- Average: 4.0%
- Baseline: 26.0%
- Affine-aware decoder wins!


Part Capsule Autoencoder
(a) infer parts & poses
Object Capsule Autoencoder
(b) explain poses
reassemble
image likelihood
part likelihood
templates (learned)

Part Capsule Autoencoder

$$\mathbf{x}_{1:M}, d_{1:M}, \mathbf{z}_{1:M} = \mathrm{h}^{\mathrm{enc}}(\mathbf{y}) \qquad \text{predict part capsule parameters,} \qquad (6)$$

- Let y \in [0, 1]^{h x w x c} be the image.
- We limit the maximum number of part capsules to M and use an encoder to infer their poses x_m , presence probabilities d_m , and special features z_m \in R^{c_z} , one per part capsule.

Part Capsule Autoencoder

$$c_m = \mathrm{MLP}(\mathbf{z}_m) \qquad \text{predict the color of the m}^{\text{th}} \text{ template,} \qquad (7)$$

$$\widehat{T}_m = \mathrm{TransformImage}(T_m, \mathbf{x}_m) \qquad \text{apply affine transforms to image templates,} \qquad (8)$$

- Special features can be used to alter the templates in an input-dependent manner (we use them to predict colour, but more complicated mappings are possible). The special features also inform the OCAE about unique aspects of the corresponding part (e. g., occlusion or relation to other parts).
- Templates T_m \in [0, 1]^{h_t x w_t x (c+1)} are smaller than the image y, but have an additional alpha channel which allows occlusion by other templates.

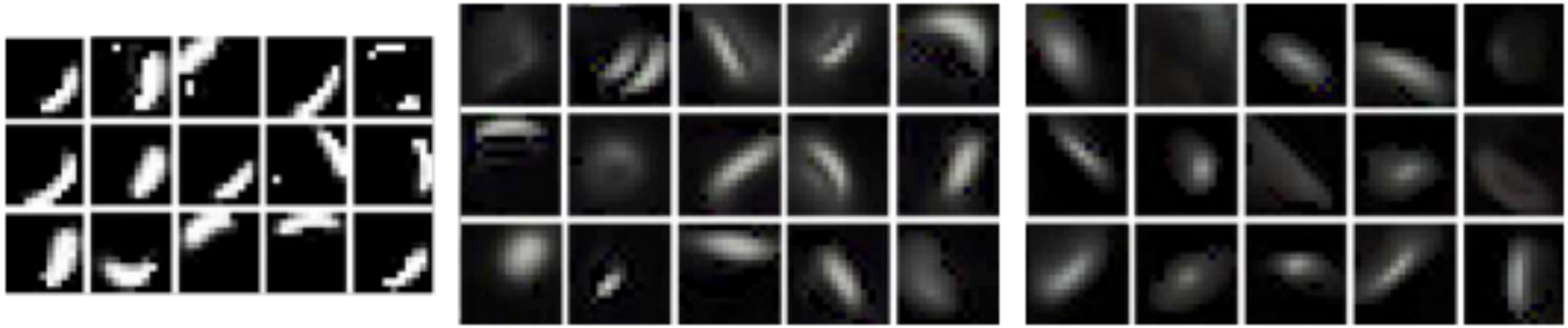
Part Capsule Autoencoder

$$p^y_{m,i,j} \propto d_m \widehat{T}^a_{m,i,j} \qquad \text{compute mixing probabilities,} \tag{9}$$

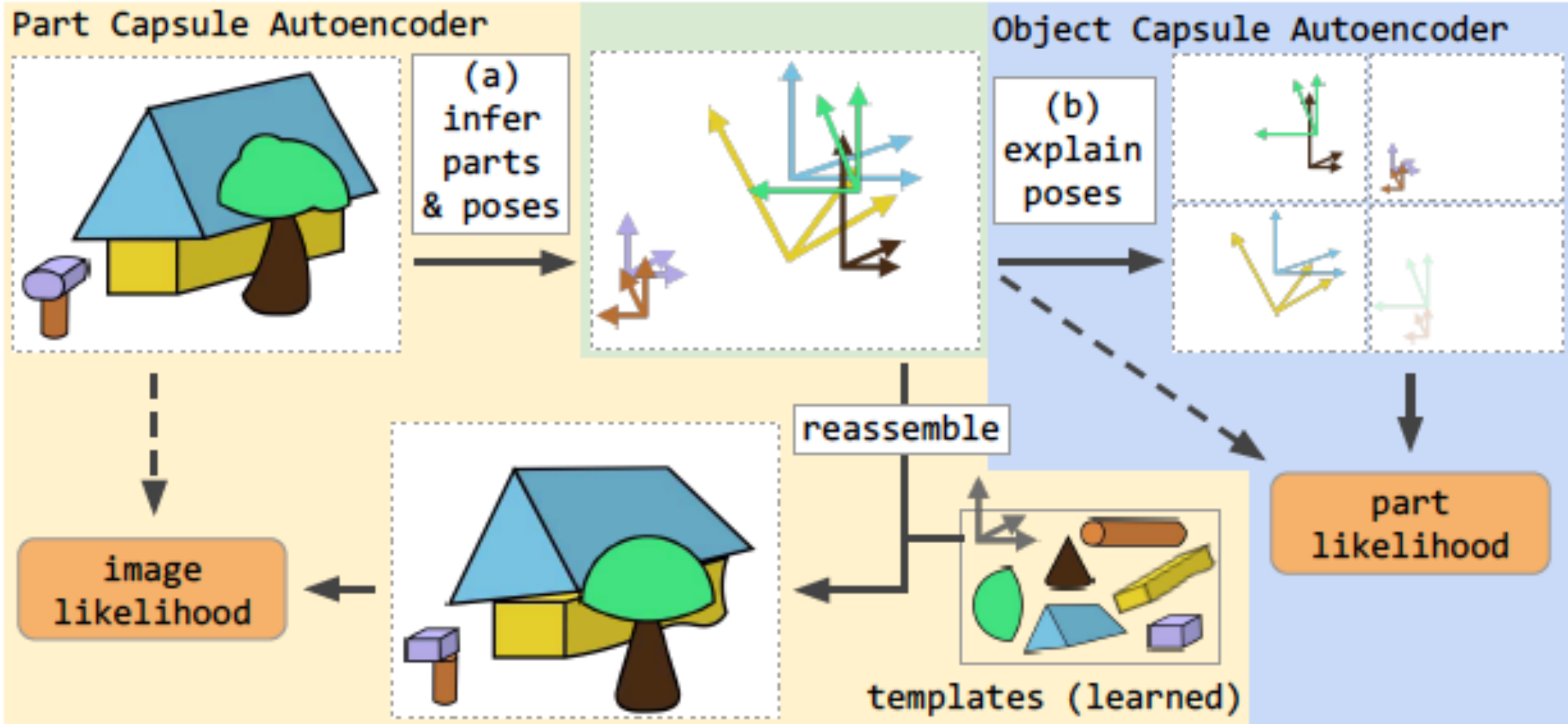
$$p(\mathbf{y}) = \prod_{i,j} \sum_{m=1}^{M} p^y_{m,i,j} \mathcal{N}\left(y_{i,j} \mid \boldsymbol{c}_m \cdot \widehat{T}^c_{m,i,j}; \sigma_y^2\right) \qquad \text{calculate the image likelihood.} \tag{10}$$

- We use T_m^a to refer to the alpha channel and T_m^c to refer to its colours.
- The image is modelled as a spatial Gaussian mixture

Stroke-like templates



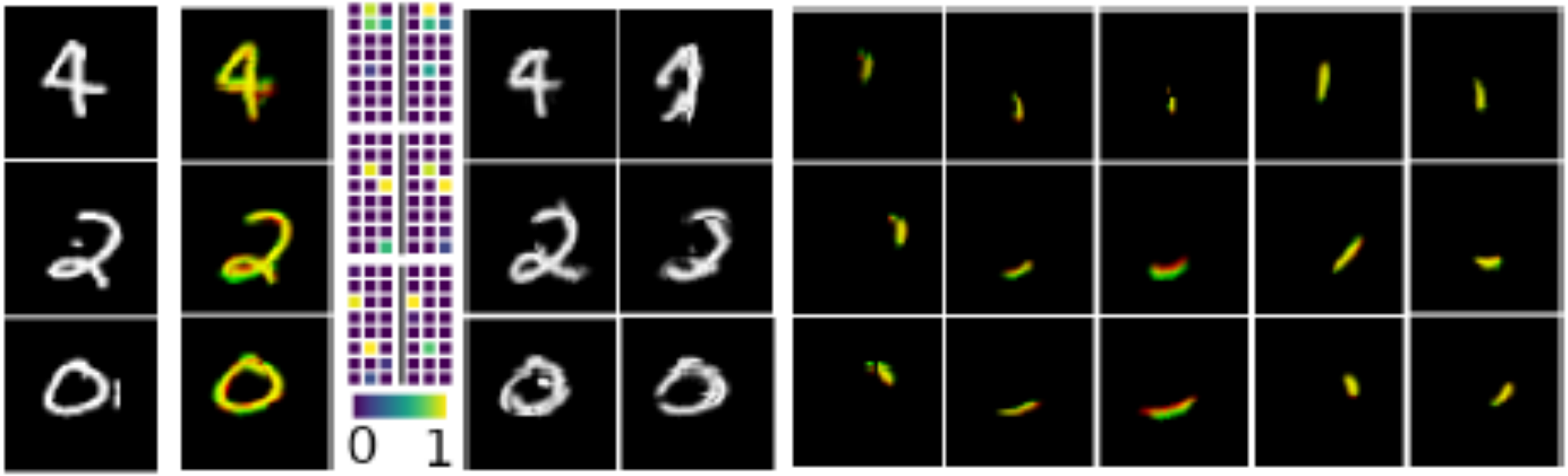
- Figure 4: Stroke-like templates learned on MNIST (left) as well as sobel-filtered SVHN (middle) and CIFAR10 (right). For SVHN they often take the form of double strokes due to sobel filtering


Part Capsule Autoencoder
(a) infer parts & poses
Object Capsule Autoencoder
(b) explain poses
reassemble
image likelihood
part likelihood
templates (learned)

Object Capsule Autoencoder

- Finally, parts discovered by the PCAE have independent identities (templates and special features rather than 2D points). Therefore, every part-pose is explained as an independent mixture of predictions from object-capsules—where every object capsule makes exactly M candidate predictions V_k,1:M , or exactly one candidate prediction per part. Consequently, the part-capsule likelihood is given by,

$$p(\mathbf{x}_{1:M}, d_{1:M}) = \prod_{m=1}^{M} \left[\sum_{k=1}^{K} \frac{a_k a_{k,m}}{\sum_i a_i \sum_j a_{i,j}} p(\mathbf{x}_m \mid k, m) \right]^{d_m} . \qquad (11)$$



- MNIST (a) images,
- (b) reconstructions from part capsules in red and object capsules in green, with overlapping regions in yellow.
- Only a few object capsules are activated for every input (c) a priori (left) and even fewer are needed to reconstruct it (right).
- The most active capsules (d) capture object identity and its appearance;
- (e) shows a few o f the affine-transformed templates used for reconstruction.

Sparse and Diverse Capsule Presences

- Prior sparsity
  - We assume that training examples contain objects from different classes **uniformly** at random and we would like to assign the same number of object capsules to every class
  - We assume that **only one object** is present in every image
- Posterior Sparsity
  - minimizing the **within-example entropy** of capsule posterior presence
  - maximizing its **between-example entropy**

Architecture

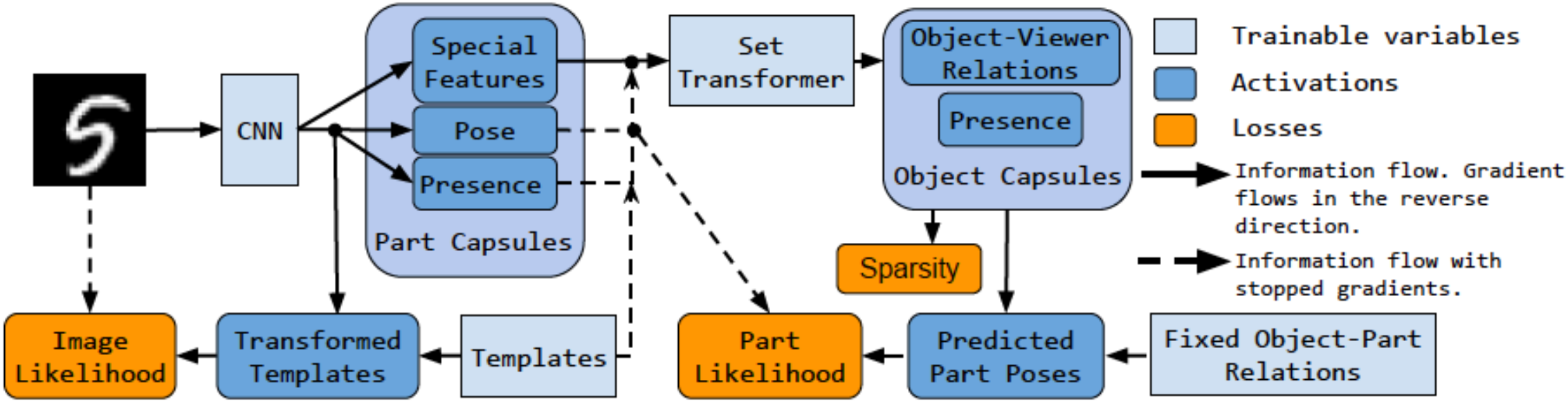


Figure 6: SCAE architecture.

Unsupervised class discovery in Images

Table 1: Unsupervised classification results in % with (standard deviation) are averaged over 5 runs. Methods based on mutual information are shaded. Results marked with † use data augmentation, ∇ use IMAGENET-pretrained features instead of images, while § are taken from Ji et al., 2018. We highlight the best results and those that are are within its 98% confidence interval according to a two-sided t test.

Method	MNIST	CIFAR10	SVHN
KMEANS (Haeusser et al., 2018)	53.49	20.8	12.5
AE (Bengio et al., 2007)§	81.2	31.4	-
GAN (Radford et al., 2016)§	82.8	31.5	-
IMSAT (Hu et al., 2017)†,∇	**98.4** (0.4)	45.6 (0.8)	**57.3** (3.9)
IIC (Ji et al., 2018)§,†	**98.4** (0.6)	**57.6** (5.0)	-
ADC (Haeusser et al., 2018)†	**98.7** (0.6)	29.3 (1.5)	38.6 (4.1)
SCAE (LIN-MATCH)	**98.7 (0.35)**	25.01 (1.0)	**55.33 (3.4)**
SCAE (LIN-PRED)	**99.0 (0.07)**	33.48 (0.3)	**67.27 (4.5)**

Ablation Study

Table 2: Ablation study on MNIST. All used model components contribute to its final performance. AFFNIST results show out-of-distribution generalization properties and come from a model trained on 40 × 40 MNIST. Numbers represent average % and (standard deviation) over 10 runs. We highlight the best results and those that are are within its 98% confidence interval according to a two-sided t test.

	Method	MNIST	40 × 40 MNIST	AFFNIST
	full model	**95.3 (4.65)**	**98.7 (0.35)**	**92.2 (0.59)**
a)	no posterior sparsity	**97.5 (1.55)**	**95.0 (7.20)**	**85.3 (11.67)**
	no prior sparsity	72.4 (22.39)	88.2 (6.98)	71.3 (5.46)
	no prior/posterior sparsity	84.7 (3.01)	82.0 (5.46)	59.0 (5.66)
b)	no noise in object caps	**96.7 (2.30)**	**98.5 (0.12)**	**93.5 (0.38)**
	no noise in any caps	**93.1 (5.09)**	78.5 (22.69)	64.1 (26.74)
	no noise in part caps	**93.9 (7.16)**	82.8 (24.83)	70.7 (25.96)
c)	similarity transforms	**97.5 (1.55)**	95.9 (1.59)	88.9 (1.58)
	no deformations	87.3 (21.48)	87.2 (18.54)	79.0 (22.44)
d)	LINEAR part enc	**98.0 (0.52)**	63.2 (31.47)	50.8 (26.46)
	CONV part enc	**97.6 (1.22)**	**97.8 (.98)**	81.6 (1.66)
e)	MLP enc for object caps	27.1 (9.03)	36.3 (3.70)	25.29 (3.69)
f)	no special features	90.7 (2.25)	58.7 (31.60)	44.5 (21.71)

Conclusion

- A novel method for representation learning, in which highly structured decoder networks are used to train one encoder network that can segment an image into parts and their poses and another encoder network that can compose the parts into coherent wholes
- Even though our training objective is not concerned with classification or clustering, SCAE is the only method that achieves competitive results in unsupervised object classification without relying on mutual information (MI).
- SCAE under-performs on CIFAR 10, which could be because of using fixed templates, which are not expressive enough to model real data. This might be fixed by building deeper hierarchies of capsule autoencoders.