Analyzing and Improving Normalization Techniques of Embedding-Based Zero-Shot Learning Classifiers

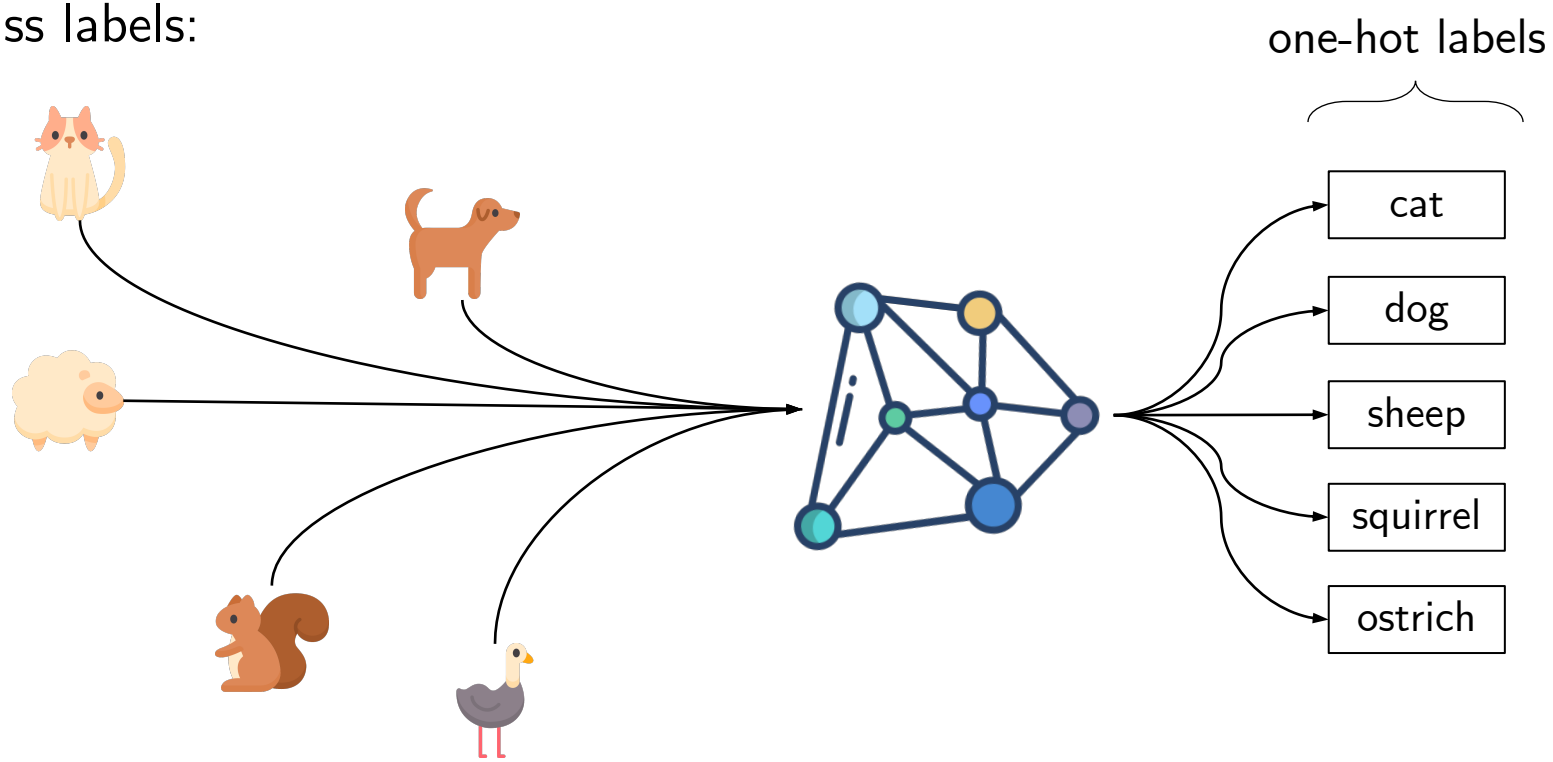
Ivan Skorokhodov, Mohamed Elhoseiny

Overview

- In the work, we (mainly) explore *zero-shot learning* models: models which perform classification based on *semantic class attributes* instead of one-hot class labels
- ZSL models (typically) use 2 normalization tricks: (*scaled cosine similarity* and *attributes normalization)*
  - They were shown to be very important in practice
  - But there is no good understanding of their influence
- In our work, we tried to "explain" these tricks by analyzing the variance of logits and activations
  - and found them working worse when the model depth increases
- This motivated us to develop *class normalization*: a BatchNorm-like mechanism for some specific component of a ZSL model
- It allowed to achieve state-of-the-art performance with a very simple model
- We also formulated a ``generalized'' version of ZSL: continual zero-shot learning
  - + formulated some metrics to evaluate a CZSL model

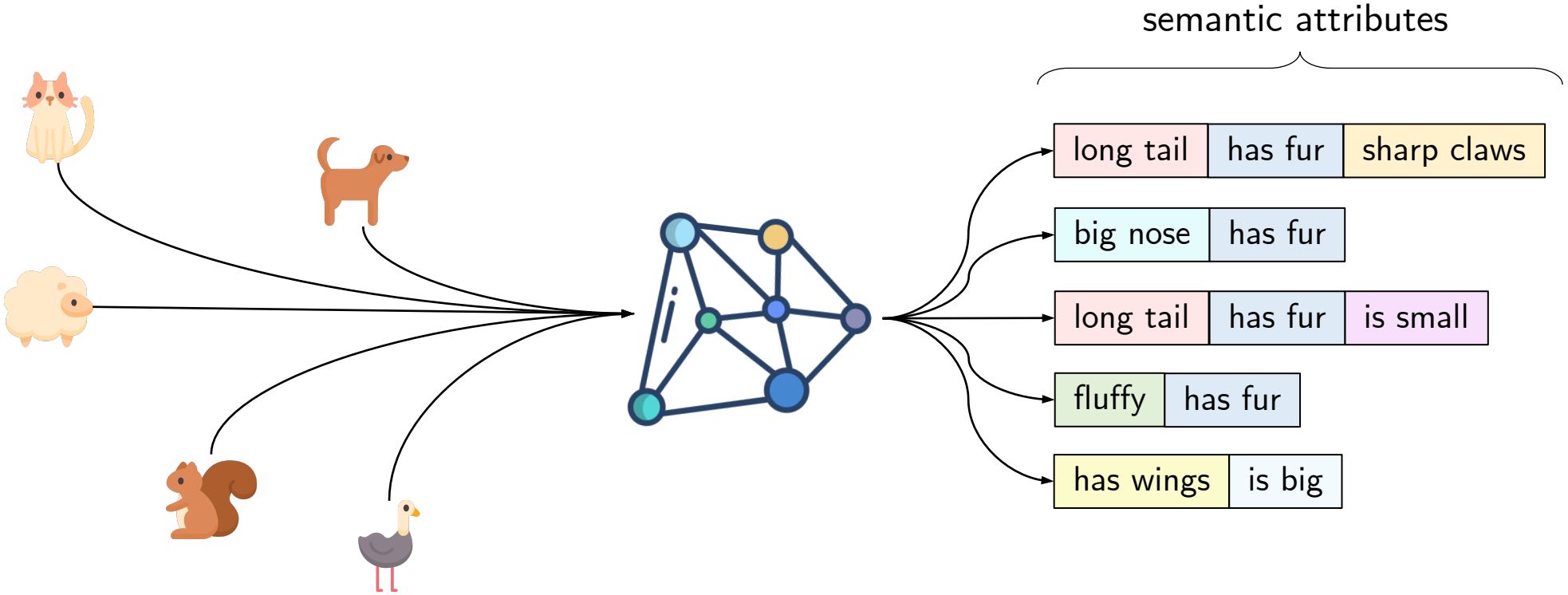
What is zero-shot learning (ZSL)?

Traditional (non-ZSL) classification matches images with class labels:



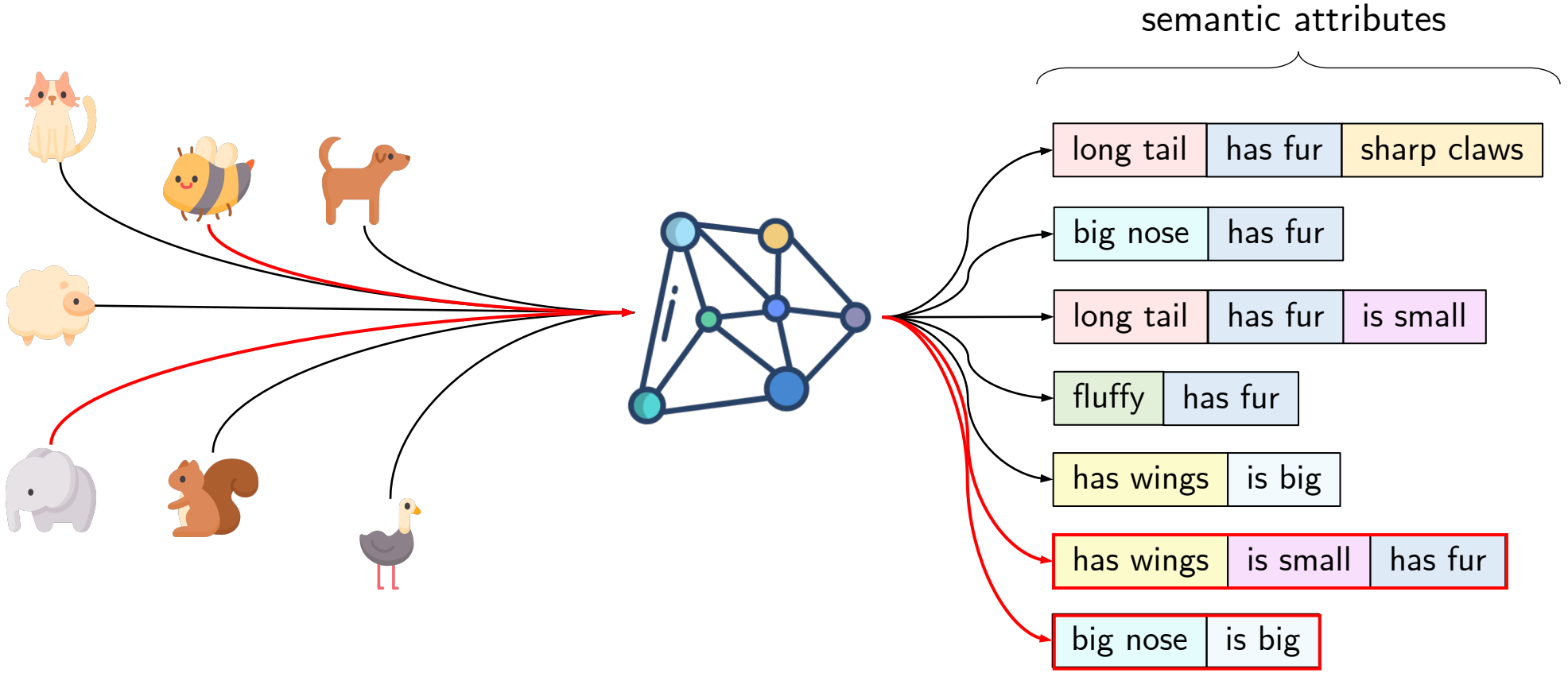
What is zero-shot learning (ZSL)?

ZSL classification matches images with semantic class attributes:



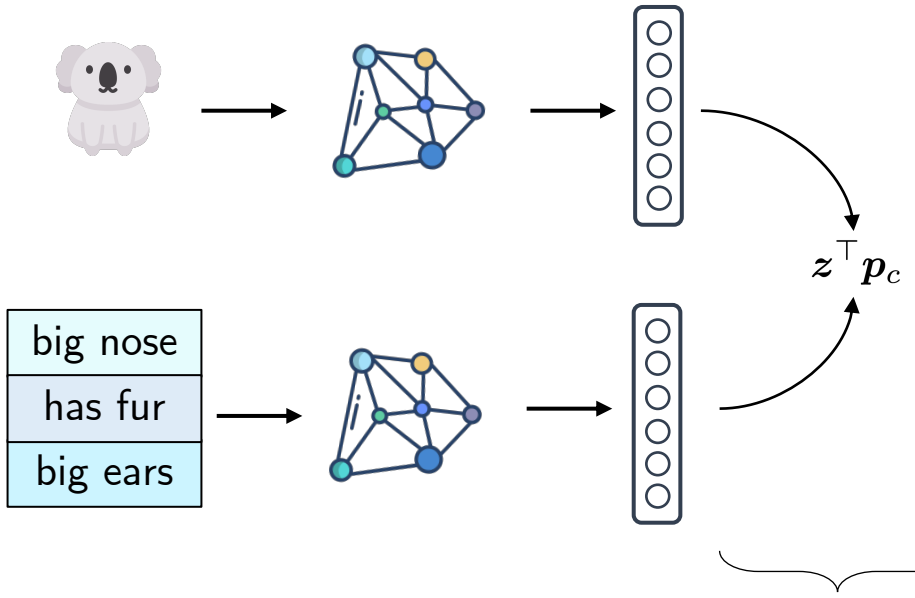
What is zero-shot learning (ZSL)?

...and deals with novel classes at test time:



Two paradigms in ZSL

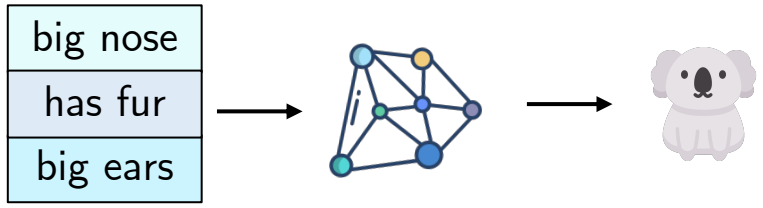
Embedding-based



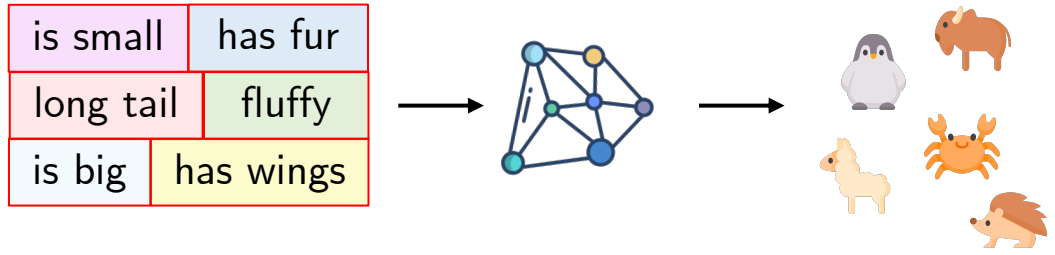
maximize similarity
for matching pairs

In our work, we follow the embedding-based paradigm

Generative-based



train a generator to synthesize images from attributes



generate images of unseen classes
at test time and train a classifer

Embedding-based ZSL

- Embedding-based ZSL model has two components:
  - image embedder $\boldsymbol{z} = E_{\mathfrak{b}}\boldsymbol{x}_{\mathfrak{c}}$
  - class attribute embedder $\boldsymbol{p}_{\mathrm{c}} = P(\boldsymbol{a}_c)$
- Typically, E(x) is pretrained on ImageNet for traditional classification task and then *kept fixed*
  - In this way, only $P(\boldsymbol{a}_c)$ is trained
- The model is optimized with the vanilla cross-entropy loss:

$$\mathcal{L}(\theta) = \sum_{i=1}^{n}\sum_{c=1}^{K}[y^{(n)} = c] \cdot \ln \hat{y}_c^{(n)}$$

But in contrast to a traditional classifier, which predicts a logit as $\hat{y} = f(x)$, the embedding-based ZSL model uses a semantic attribute vector:

$$\hat{y}_c^{(n)} = (\boldsymbol{E}(\boldsymbol{x}^{(n)}))^{\top}\boldsymbol{P}(\boldsymbol{a}_c)$$

This formulation allows to use at test time such $\boldsymbol{a}_c$ that have not been seen during training (and classify images of unseen classes)

Normalization techniques in the embedding-based ZSL

1. Scaled cosine similarity (when computing logits):

$$\hat{y}_c = \boldsymbol{z}^\top \boldsymbol{p}_c \Longrightarrow \hat{y}_c = \gamma^2 \frac{\boldsymbol{z}^\top \boldsymbol{p}_c}{\|\boldsymbol{z}\| \|\boldsymbol{p}_c\|}$$

⇒ Stabilizes logits' variance:

$$\mathrm{Var}\left[\hat{y}_c\right] \approx \gamma^4 \frac{d_z}{(d_z - 2)^2}$$

2. Attributes normalization:

$$\boldsymbol{a}_c \longmapsto \boldsymbol{a}_c / \|\boldsymbol{a}_c\|_2$$

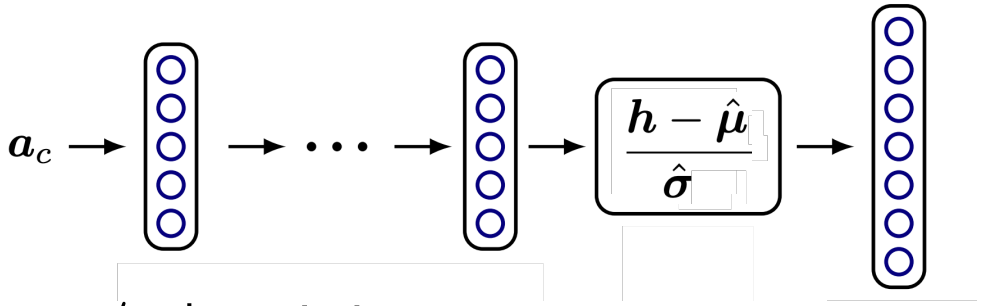
⇒ Stabilizes representations' variance:

$$\mathrm{Var}[\tilde{y}_c] = \mathrm{Var}[z^\top \boldsymbol{p}_c] = \mathrm{Var}[z_i]$$

But only for a linear attribute embedder $P(\boldsymbol{a}_c)$

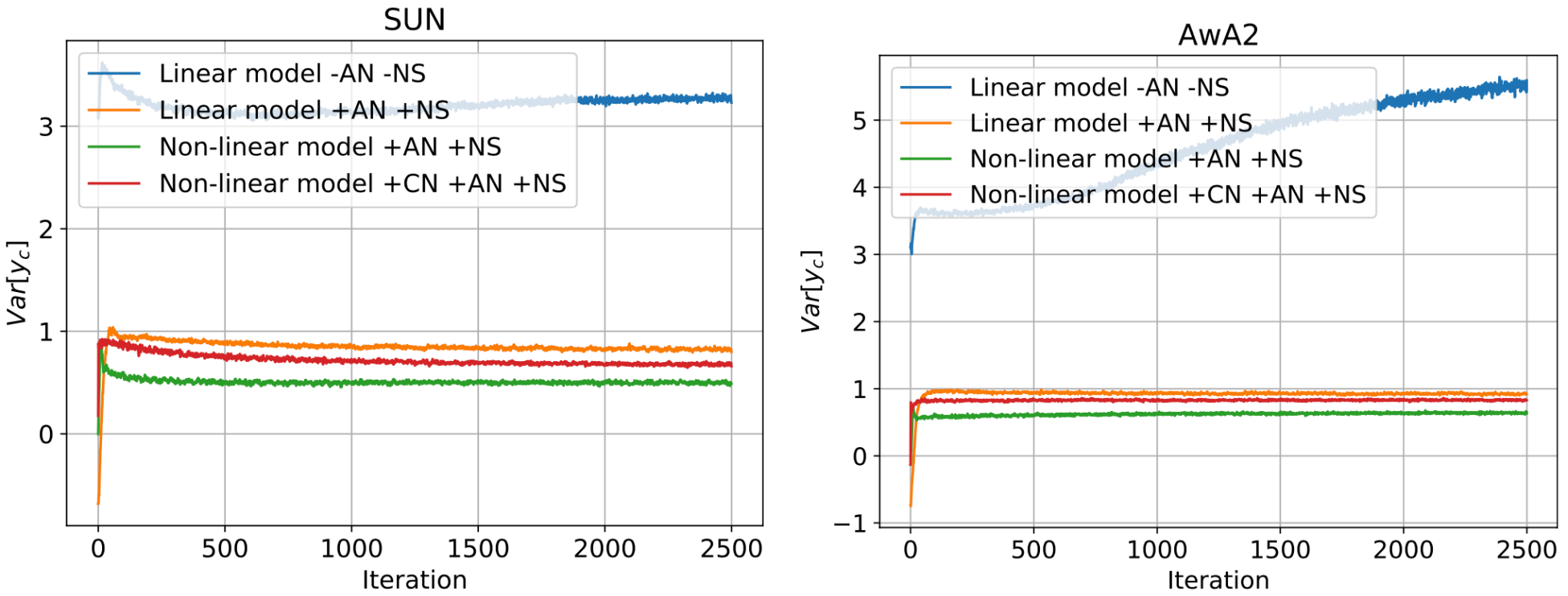
Class Normalization (CN)

- to stabilize variance in a *deep* attribute embedder, we propose *class normalization:*



- here $\hat{\mu}, \hat{\sigma}$ are minibatch mean/std statistics
- it is like batch normalization, but:
  - is applied in the attribute embedder module (i.e. class-wise)
  - does not use scaling or shifting

Empirical analysis of the variance



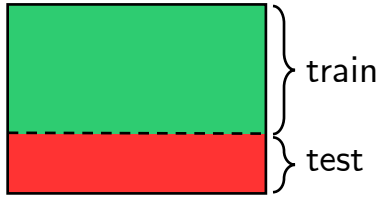
Logits variance across a minibatch for different models on SUN and AwA2 datasets

GZSL results

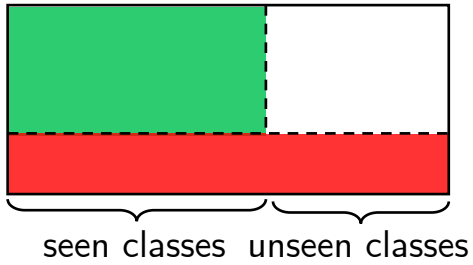
	SUN			CUB			AwA1			AwA2			Avg training time
	U	S	H	U	S	H	U	S	H	U	S	H	
DCN (Liu et al., 2018)	25.5	37.0	30.2	28.4	60.7	38.7	-	-	-	25.5	84.2	39.1	50 min
RN (Sung et al., 2018)	-	-	-	38.1	61.4	47.0	31.4	91.3	46.7	30.9	93.4	45.3	35 min
f-CLSWGAN (Xian et al., 2018b)	42.6	36.6	39.4	57.7	43.7	49.7	57.9	61.4	59.6	-	-	-	-
CIZSL (Elhoseiny & Elfeki, 2019)	-	-	27.8	-	-	-	-	-	-	-	-	24.6	2 hours
CVC-ZSL (Li et al., 2019)	36.3	42.8	39.3	47.4	47.6	47.5	62.7	77.0	69.1	56.4	81.4	66.7	3 hours
SGMA (Zhu et al., 2019)	-	-	-	36.7	71.3	48.5	-	-	-	37.6	87.1	52.5	-
SGAL (Yu & Lee, 2019)	42.9	31.2	36.1	47.1	44.7	45.9	52.7	75.7	62.2	55.1	81.2	65.6	50 min
DASCN (Ni et al., 2019)	42.4	38.5	40.3	45.9	59.0	51.6	59.3	68.0	63.4	-	-	-	-
F-VAEGAN-D2 (Xian et al., 2019)	45.1	38.0	41.3	48.4	60.1	53.6	-	-	-	57.6	70.6	63.5	-
TF-VAEGAN (Narayan et al., 2020)	45.6	40.7	43.0	52.8	64.7	**58.1**	-	-	-	59.8	75.1	66.6	1.75 hours
EPGN (Yu et al., 2020)	-	-	-	52.0	61.1	56.2	62.1	83.4	**71.2**	52.6	83.5	64.6	-
DVBE (Min et al., 2020)	45.0	37.2	40.7	53.2	60.2	56.5	-	-	-	63.6	70.8	67.0	-
LsrGAN (Vyas et al., 2020)	44.8	37.7	40.9	48.1	59.1	53.0	-	-	-	54.6	74.6	63.0	1.25 hours
ZSML (Verma et al., 2020)	-	-	-	60.0	52.1	55.7	57.4	71.1	63.5	58.9	74.6	65.8	-
3-layer MLP	31.4	40.4	35.3	45.2	48.4	46.7	57.0	69.9	62.8	54.5	72.2	62.1	**30 seconds**
3-layer MLP + Eq. (9)	41.5	41.3	41.4	49.4	48.6	49.0	60.1	73.0	65.9	60.3	75.6	67.1	
3-layer MLP + Eq. (10)	24.1	37.9	29.5	45.3	44.5	44.9	58.4	70.7	64.0	52.1	72.0	60.5	
3-layer MLP + **CN** (i.e. (9) + (10))	44.7	41.6	**43.1**	49.9	50.7	50.3	63.1	73.4	67.8	60.2	77.1	**67.6**	

Continual Zero-Shot Learning

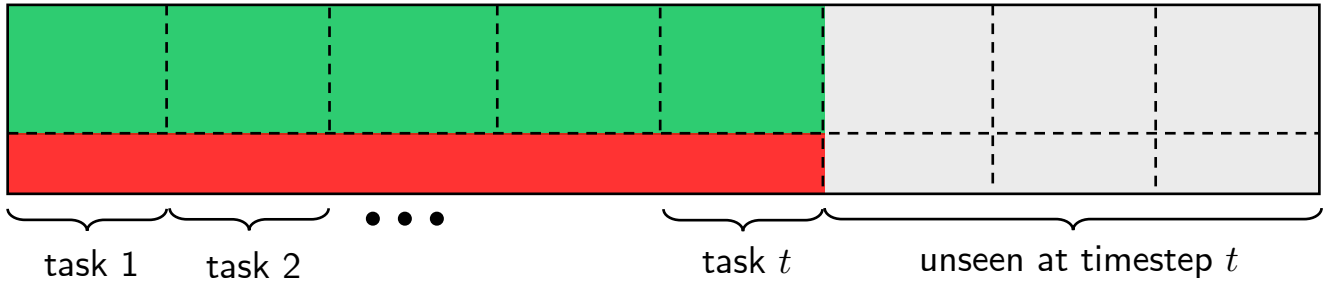
Traditional learning



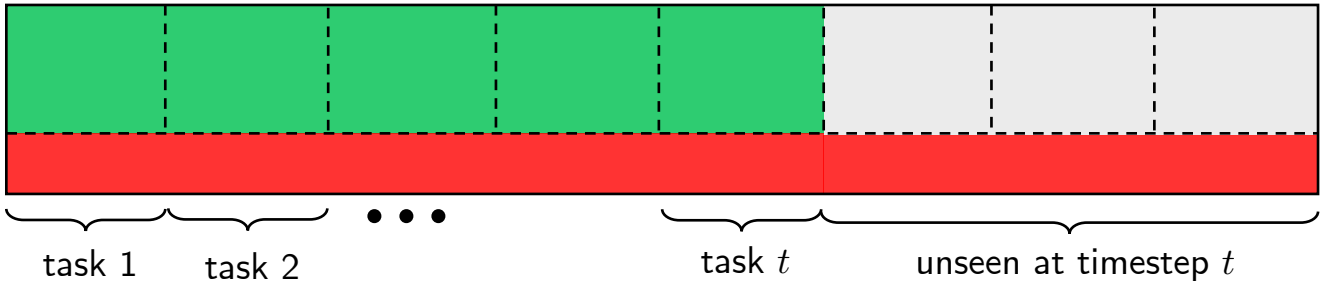
Zero-shot learning
(generalized setting)



Traditional continual learning



Continual ZSL
(our formulation)



CZSL results

	CUB				SUN			
	mAUC ↑	mH ↑	mJA ↑	Forgetting ↓	mAUC↑	mH↑	mJA↑	Forgetting ↓
EWC-online (Schwarz et al., 2018)	11.6	18.0	25.4	0.08	2.7	9.6	11.4	**0.02**
EWC-online + ClassNorm	**14.1**$^{+22\%}$	23.3$^{+29\%}$	**28.6**$^{+13\%}$	**0.04**$^{-50\%}$	**4.8**$^{+78\%}$	**14.3**$^{+49\%}$	**15.8**$^{+39\%}$	0.03$^{+50\%}$
MAS-online (Aljundi et al., 2017)	11.4	17.7	25.1	0.08	2.5	9.4	11.0	**0.02**
MAS-online + ClassNorm	14.0$^{+23\%}$	**23.8**$^{+34\%}$	28.5$^{+14\%}$	0.05$^{-37\%}$	**4.8**$^{+92\%}$	14.2$^{+51\%}$	**15.8**$^{+44\%}$	0.03$^{+50\%}$
A-GEM (Chaudhry et al., 2019)	10.4	17.3	23.6	0.16	2.4	9.6	10.8	0.05
A-GEM + ClassNorm	13.8$^{+33\%}$	**23.8**$^{+38\%}$	28.2$^{+19\%}$	0.06$^{-62\%}$	4.6$^{+92\%}$	14.2$^{+48\%}$	15.4$^{+43\%}$	0.04$^{-20\%}$
Sequential	9.7	17.2	22.6	0.17	2.3	9.3	10.4	0.05
Sequential + ClassNorm	13.5$^{+39\%}$	23.0$^{+34\%}$	27.9$^{+23\%}$	0.05$^{-71\%}$	4.6$^{+99\%}$	14.0$^{+51\%}$	15.3$^{+47\%}$	0.03$^{-40\%}$
Multi-task	23.4	24.3	39.6	0.00	4.2	12.5	14.9	0.00
Multi-task + ClassNorm	26.5$^{+13\%}$	30.0$^{+23\%}$	42.6$^{+8\%}$	0.01	6.2$^{+48\%}$	14.8$^{+18\%}$	18.5$^{+24\%}$	0.01

Thank you!