Supplementary Material:
Two-Level Adversarial Visual-Semantic Coupling for Generalized Zero-shot Learning

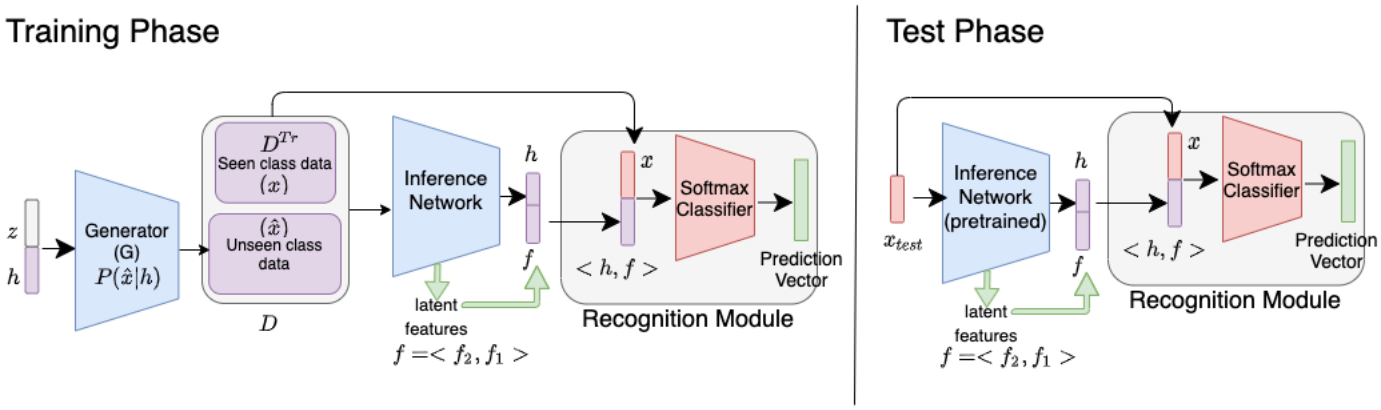


Figure A: Training and testing phase pipelines as explained in Section 3.4 for the proposed framework that uses representation from inference network, along with synthesized features for training recognition module.

In this supplementary section, we discuss the following details, which could not be included in the main paper owing to space constraints:

- Implementation details of our experiments (in continuation to Sec 4)
- Results on the standard ZSL setting for CUB, AWA2, SUN and FLO datasets (in continuation to results in Sec 4)
- Details describing the Alignment loss (in continuation to Sec 3.2)
- Related work on Generative Models (in continuation to Sec 2)
- Results on the GZSL setting for SUN dataset
- Show results of our proposed approach using a weighted classifier instead of concatenation of all features

We have also added a complete figure, Figure A, with both training and testing phases of the recognition module for clarity of understanding.

A. Implementation Details

In this section, we describe the implementation details for our methodology. The generator (G), inference network (I) and discriminators (D_1, D_2, D_3) are all implemented using fully connected neural networks. In order to ensure fair comparison, we follow the architecture used in [A14] for all our components. Formally, the generator and inference network both consist of 2 dense layers of size 4096 with leaky ReLU activation except at the output layer which has ReLU activation. These layers in the inference network form the latent features $\mathbf{f}_1, \mathbf{f}_2$ in our recognition module as shown in Figure A. The dimension of output layer is 2048 in case of the generator and d_h in case of the inference network where d_h denotes the dimension of semantic attributes. The three discriminators consist of 2 fully connected hidden layers of size 4096 with leaky ReLU activation. The noise vector z is sampled from a unit Gaussian (zero mean, unit variance). We find that taking the dimension of noise vector same as that of semantic embeddings works well as in [A14]. The generator is updated every 5 discriminator iterations as suggested in [A8]. We use an Adam optimizer. We use a single-layered softmax classifier in our recognition module for fair comparison (most earlier work use this) and

simplicity. Table A presents the details of hyperparameters for generalized zero-shot learning for each of the considered benchmark datasets.

Dataset	β	λ	γ	α_1	α_2
CUB	0.01	10	3	1	2
FLO	0.01	10	0.01	1	1
AWA1	0.01	10	0.001	10	2
AWA2	0.01	10	0.01	5	4

Table A: Hyperparameters used for different datasets

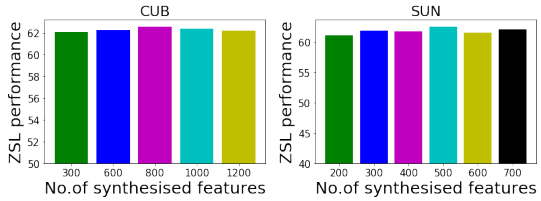
Dataset	**CUB**	**AWA2**	**SUN**	**FLO**
Methods	PS	PS	PS	PS
CONSE(ICLR 2014)	34.3	44.5	38.8	-
SSE(ICCV 2015)	43.9	61.0	51.5	-
LATEM(CVPR 2016)	49.3	55.8	55.3	40.4
ALE(TPAMI 2016)	54.9	62.5	58.1	48.5
DEVISE(NIPS 2013)	52.0	59.7	56.5	45.9
SJE(CVPR 2015)	53.9	61.9	53.7	53.4
ESZSL(ICML 2015)	53.9.	58.6	54.5	51.0
SYNC(CVPR 2016)	55.6	46.6	56.3	-
SAE(CVPR 2017)	33.3	54.1	40.3	-
GFZSL(ECML 2017)	49.2	67.0	62.6	-
CVAE-ZSL(CVPRW 2018)	52.1	65.8	61.7	-
SE-ZSL(CVPR 2018)	59.6	69.2	63.4	-
DCN(NIPS 2018)	56.2	-	61.8	-
JGM-ZSL(ECCV 2018)	54.9	69.5	59.0	-
RAS+cGAN(NC 2019)	52.6	-	61.7	-
DEM(CVPR 2017)	51.7	67.1	61.9	-
SP-AEN(CVPR 2018)	55.4	58.5	59.2	-
f-clsWGAN(CVPR 2018)	57.3	68.2	60.8	67.2
CADA-VAE(CVPR 2019)	60.4	64	61.8	-
f-VAEGAN(CVPR 2019)	61.0	71.1	64.7	67.7
GZLOCD(CVPR 2020)	60.3	**71.3**	63.5	-
TACO-ZSL (312)	**63.0**	**71.3**	63.0	**68.5**
TACO-ZSL	**68.8**	**71.3**	63.0	**68.5**
TACO-ZSL (312) (using Φ_2)	**66.4**	**72.4**	**65.4**	-
TACO-ZSL(using Φ_2)	**72.3**	**72.4**	**65.4**	-

Table B: ZSL performance comparison with several baseline and state-of-the-art methods. We measure Top-1 accuracy for conventional zero-shot setting on proposed splits (**PS**), following the protocol in [28]. Best results are highlighted in bold. **TACO-ZSL (312)** indicates result on CUB dataset with only 312 dimensional attributes (included for fair comparison with other work that use this setting)

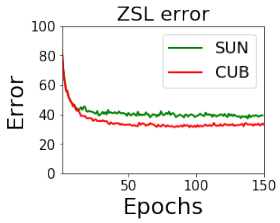
B. Performance on ZSL

This work is focused on addressing the more practical and challenging generalized zero-shot learning problem, similar to [11][16]. However, to further demonstrate the effectiveness of our proposed method, we also evaluate our proposed methodology on the standard ZSL setting. For fair comparison, we follow the standard training/validation/testing splits and evaluation protocols for ZSL setting, as in [28].

For an exhaustive performance comparison, we compare with all state-of-the art ZSL methods, including recent ones,



(a) Variation in ZSL performance with number of synthesised features for unseen classes



(b) TACO-ZSL error trajectory over epochs for proposed method

Figure B: Analysis of performance of TACO on standard ZSL

as mentioned in the very recent work [12]. In addition, we also compare with some other important ZSL approaches like f-VAEGAN [30], CADA-VAE [21], f-clsWGAN [29], SP-AEN [A2], DEM [32]. Table B shows the results for our method on the ZSL setting. For fair comparison, all results reported are without fine-tuning the backbone ResNet-101 network. It can be clearly seen that even on the standard ZSL setting, our method outperforms other methods (including ones specifically designed explicitly for this setting) on CUB, AWA2, FLO datasets and achieves competitive performance on SUN dataset. Also, our method provides state-of the art performance on SUN dataset as well as all other datasets when using Φ_2 as the feature extractor backbone.

To further study the ZSL performance of TACO, we analyzed the results with varying number of synthesized features for unseen classes and also the error trajectory over epochs for standard ZSL setting. Figure B shows these results. It can be clearly seen that the trend for ZSL accuracy is fairly stable with variation in number of synthesized features for both CUB and SUN datasets, supporting the robustness of our algorithm to such hyperparameter choices. Also, the ZSL error smoothly decreases over epochs with a stable trend, and reaches convergence quite early in the trend, after which the performance stays nearly constant.

C. Alignment Loss

In this section, we provide more details on the $L_{wasserstein}$ term in Eqn 11 of the main paper. We use the sinkhorn distance-based lightspeed computation proposed by Cuturi in [A4] for computing our alignment loss. The sinkhorn distance metric has also been used for approximat-

ing the Wasserstein distance in [A13] for a very different projection-based (non-generative) ZSL method for alignment in visual space. We instead use the metric to provide distributional alignment in the semantic space which helps us to preserve high-level semantics better and reduce semantic loss. To the best of our knowledge, this has not been done before in ZSL literature. We use the Wasserstein distance to model the joint probability of visual-semantic features better by combining it with adversarial loss in a generative GZSL setting. Formally, the sinkhorn distance can be written as:

$$\mathcal{L}_{Wasserstein} = \min_{X} \sum_{i,j} \text{dis}_{ij} x_{ij} - \epsilon H(X) \qquad (1)$$

where $H(X)$ is an entropy-based regularization term and $\text{dis}_{ij}(\cdot)$ is as defined in [A13]. We compute $\text{dis}_{ij}(\cdot)$ however on an assignment matrix X with entries given by x_{ij}, which defines the matching relationship between the class centers of output semantic attributes, $\hat{\mathbf{h}}$, and ground truth semantic centers $\mathbf{h}$. Here, $i \in A$ and $j \in B$ where A and B are sets of class centres of output semantic attributes and ground truth semantic attributes respectively. This helps compute the term $P(\mathbf{h}|\hat{\mathbf{h}})$ in our methodology.

D. Related Work: Generative Models

Since ours is a generative approach to GZSL, we briefly present the earlier work in generative models underlying our methodology, for completeness of our discussion on related work. Generative modeling aims to learn the probability distribution of data points such that we can randomly sample data from it. The idea behind Generative Adversarial Networks (GANs) is to learn a generative model to capture an arbitrary data distribution via a min-max training procedure which consists of a generator that synthesizes fake data and a discriminator that distinguishes fake and real data. These models have been used in many interesting computer vision applications especially for image generation [A7, A12, A3] and have achieved compelling results. However, GANs are also known for their instability in training and are known to suffer from the mode collapse problem. In order to mitigate these problems and improve the quality of synthetic samples, methods like WGAN [A1] and WGAN-GP [A8] have been proposed, which we leverage in this work. GANs have also been used for tasks like multi-view generation and learning cross-modal representations for downstream tasks like retrieval or semi-supervised classification [A12] [A6]. Such generative models can be trained explicitly to model conditional/joint distributions of random variables, which we leverage in this work. For example, [A10] shows how such generative models can be used to generate data for a specific class by conditioning them on the label. [A6, A11, A9, A5] show how adversarial training can be used to model joint and utilize the trained model for semi-supervised learning.

E. GZSL Performance on SUN Dataset

In Table C, we show the results of TACO in the GZSL setting on the SUN dataset, which could not be included in the main paper due to space constraints. Note that our method outperforms recent methods even on this dataset.

Dataset	**SUN**		
Methods	U	S	H
DEM(CVPR'17)[32]	20.5	34.3	25.6
ZSKL(CVPR'18)[10]	21.0	31.0	25.1
DCN(NIPS'18)[14]	25.5	37.0	30.2
ALE(TPAMI'13)[1]	21.8	33.1	26.3
DEVISE(NIPS'13)[9]	16.9	27.4	20.9
ESZSL(ICML'15)[20]	11.0	27.9	15.8
SYNC(CVPR'16)[5]	7.9	43.3	13.4
LATEM(CVPR'16)[27]	14.7	28.8	19.5
SJE(CVPR'15)[2]	14.7	30.5	19.8
CLSWGAN(CVPR'18)[29]	42.6	36.6	39.4
CADA-VAE(CVPR'19)[21]	47.2	35.7	40.6
VSE(CVPR'19)[19]	-	-	-
DASCN(NIPS'19)[16]	42.4	38.5	40.3
SGAL(NIPS'19)[31]	40.9	30.5	34.9
SE-GZSL(CVPR'18) [23]	40.9	30.5	34.9
CycWGAN(ECCV'18)[8]	47.2	33.8	39.4
f-VAEGAN(CVPR'19)[30]	45.1	38.0	41.3
ZSML(AAAI'20)[24]	-	-	-
TACO-GZSL	44	39	41.3
TACO-GZSL(weighted classifier)	46.5	39.1	**42.4**
TACO-GZSL(using Φ_2)(weighted classifier)	51.7	36.8	**43**

Table C: GZSL performance comparison with several baseline methods on SUN dataset. For fair comparison, all results reported here are *without fine-tuning* the backbone ResNet101 feature extractor. We measure Top-1 accuracy on Unseen(U), Seen(S) classes and their Harmonic mean(H).

F. Weighted Classifier

As discussed in Sec-3.4 and Figure A, we train the softmax classifier of our recognition module on concatenated features $< \mathbf{x}, \mathbf{h}, \mathbf{f} >$. In order to study this choice further, we studied the performance of our method using a weighted softmax classifier instead. We use the weighted classification results from $< \mathbf{x}, \mathbf{h} >$ and $< \mathbf{f}_1, \mathbf{f}_2 >$ i.e. r_{vs}, r_{latent} respectively to get the final predictions of our method. The final classification results (r_{cls}) are given by:

$$r_{cls} = r_{vs} + w * r_{latent} \qquad (2)$$

These results are shown in Table D.

References

[A1] M. Arjovsky, S. Chintala, and L. Bottou. "Wasserstein gan." In: *ICML* (2017).

Dataset	CUB			FLO			AWA1			AWA2		
TACO-GZSL(312)(using weighted classifier)	53.7*	58.7*	**56.1***	61.8	76.5	68.4	61.3	71.8	**66.19**	59.7	72.0	65.3

Table D: GZSL performance using weighted classifier as discussed in supplementary.We follow the same protocol as followed for Table 1 in main paper. Note that * indicates result on CUB dataset with only 312 dim attributes

[A2] Long Chen et al. "Zero-Shot Visual Recognition using Semantics-Preserving Adversarial Embedding Networks". In: *CVPR* (2018).

[A3] X. Chen et al. "Infogan: Interpretable representation learning by information maximizing generative adversarial nets." In: *NIPS* (2016).

[A4] M. Cuturi. "Sinkhorn distances: Lightspeed computation of optimal transport". In: *NIPS* (2013).

[A5] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. "Adversarial feature learning". In: *ICLR* (2017).

[A6] V. Dumoulin et al. "Adversarially learned inference." In: *ICLR* (2017).

[A7] I. Goodfellow et al. "Generative adversarial nets". In: *NIPS* (2014).

[A8] I. Gulrajani et al. "Improved training of wasserstein gans." In: *NIPS* (2017).

[A9] M.-Y. Liu and O. Tuzel. "Coupled generative adversarial networks". In: *NIPS* (2016).

[A10] M. Mirza and S. Osindero. "Conditional generative adversarial nets." In: *CoRR* (2014).

[A11] Yunchen Pu et al. "Jointgan: Multi-domain joint distribution learning with generative adversarial nets." In: *ICML* (2018).

[A12] A. Radford, L. Metz, and S. Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks". In: *ICLR* (2016).

[A13] Ziyu Wan et al. "Transductive Zero-Shot Learning with Visual Structure Constraint". In: *NIPS* (2019).

[A14] Y. Xian et al. "A feature generating framework for any-shot learning". In: *CVPR* (2019).