# Supplementary Material for Learning Robust Visual-Semantic Embeddings

Yao-Hung Hubert Tsai Liang-Kang Huang Ruslan Salakhutdinov
School of Computer Science, Machine Learning Department, Carnegie Mellon University
yaohungt@cs.cmu.edu liangkah@andrew.cmu.edu rsalakhu@cs.cmu.edu

## I. Network Design

Fig. 1 provides an easy-to-understand design of ReViSE. In all of our experiments, GoogLeNet is pre-trained on ImageNet [2] images. Without fine-tuning, we directly extract the top layer activations (1024-dim) as our input image features followed by a common $log(1+v)$ pre-processing step. For the textual attributes, we pre-process them through a standard $l_2$ normalization.

In ReViSE, we set $\alpha = 1.0$ in eq. (11), so that we place equal importance on supervised and unsupervised objectives. For the visual auto-encoder, we fix the parameter of the contraction strength $\gamma = 0.1$ in eq. (2). In the following, we omit the bias term in each layer for simplicity. The encoding of visual features is parameterized by a two-hidden layer fully-connected neural network with architecture $d_{v1}-d_{v2}-d_c$, where $d_{v1} = 1024$ is the input dimension of the visual features, $d_{v2} = 500$ is the intermediate layer, and $d_c$ denotes the dimension of the visual codes $\tilde{v}_h$. To encode textual attributes, we consider a single-hidden layer neural network $d_{t1}-d_c$, where $d_{t1}$ is the input dimension of the textual attributes. We choose $d_c = 100$ when $d_{t1} > 100$ and $d_c = 75$ when $d_{t1} < 100$. Furthermore, we do not tie the weights to be learned between the decoding and encoding parts. Parameters for associating distributions of visual and textual codes (MMD Loss) in eqs. (5) (12), and (6) are set as $\beta = \{0.1, 1.0\}$ (chosen by cross-validation) and $\kappa = 32.0$. For the remaining part of our model, we set the architecture of visual and textual code mapping as a single-hidden layer fully-connected neural network with dimension $d_c - 50$. We also adopt a dropout of 0.7.

During the first 100 iterations of training, we set $\lambda = 0$ so that no unsupervised-data adaptation is used while still updating $\hat{I}^{(ut)}_{i,c}$. Note that $\hat{I}^{(ut)}_{i,c}$ are the *inferred labels* for unsupervised data, and *not random* at each iteration. Beginning with the 101th iteration, we set $\lambda = \{0.1, 1.0\}$ (chosen by cross-validation), and the model typically converges within 2000 to 5000 iterations.

We implement ReViSE in TensorFlow [1]. We use Adam [3] for optimization with minibatches of size 1024. We choose $tanh$ for all of our activation functions.

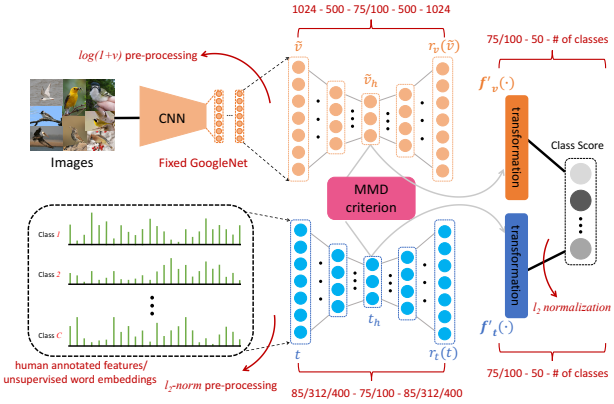


Figure 1. Our designed architecture.

Table I. Value of $\beta$ and $\lambda$.

| Dataset | AwA | | | CUB | | |
|---|---|---|---|---|---|---|
| *attributes* | *att* | *w2v* | *glo* | *att* | *w2v* | *glo* |
| $\beta$ | 0.1 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| $\lambda$ | 1.0 | 1.0 | 1.0 | 1.0 | 0.1 | 0.1 |

## II. Parameters Choice

We have four parameters in our architecture: $\alpha, \beta, \gamma$, and $\kappa$. We fix $\alpha = 1.0$, $\gamma = 0.1$, $\kappa = 32.0$ for all the experiments. Then we set $\lambda = 0.0$ (no unsupervised-data adaptation inference), and perform cross-validation on the splitting set as suggested by [3,46] to determine $\beta$ from $\{0.1, 1.0\}$. Next, with chosen $\beta$, we perform cross-validation to choose $\lambda$ from $\{0.1, 1.0\}$. Table I lists the statistics of $\beta$ and $\lambda$.

Next, we study the power of unsupervised information. We now take CUB dataset with *att* attributes to test the advantage of using unsupervised information, which can be viewed as tuning the parameter $\alpha$ for the unsupervised objective in eq. (11). Originally, $\alpha$ was set to 1.0, which equally weights the contribution of supervised and unsupervised loss. We now alter $\alpha$ as follows: 0.1 to 1.0 by step size of 0.1 and 0.5 to 5.0 by step size of 0.5. The results

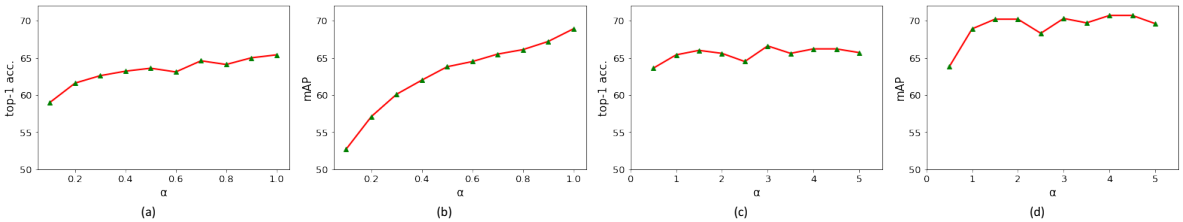


Figure 2. Varying $\alpha$ in two scales: 0.1 to 1.0 and 0.5 to 5.0. (a),(c) display plots for transductive zero-shot recognition and (b),(d) display plots for transductive zero-shot retrieval. CUB dataset with *att* attributes are used in the experiments.

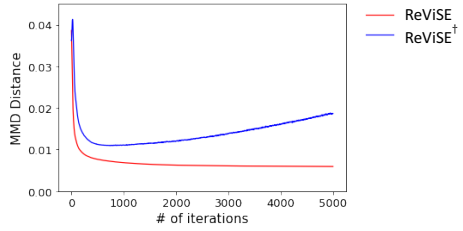


Figure 3. MMD distance w.r.t. # of iterations for our method with and without $\mathcal{L}_{MMD}$. The experiment is conducted on CUB dataset with *att* attributes under transdutive zero-shot setting.

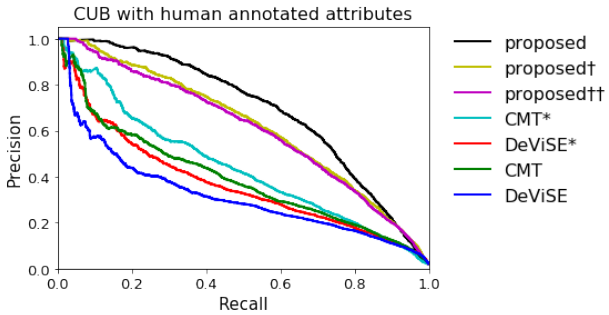


Figure 4. Precision-recall curve comparison for zero-shot retrieval on CUB with human annotated attributes as textual attributes for classes. Best viewed in color.

are shown in Fig. 2. We observe that when $\alpha$ increases from 0.1 to 1.0, the performance increases; however, when $\alpha$ increase from 1.0 to 5.0, the performance stays relatively unchanged. Empirically, we find that ReViSE does not perform better when $\alpha > 1.0$, which is expected, since we should not view unsupervised information more important than supervised information.

## III. Precision-Recall Curve

Fig. 4 is the precision-recall curve for zero-shot retrieval results on CUB dataset with *att* attributes.

## IV. MMD Distance

MMD distance in eq. (5) can be viewed as the *distribution measurement* [13] between visual and textual code. For CUB dataset with *att* attributes under transductive zero-shot experiment, we calculate the MMD distance (on the test codes) in our method with (ReViSE) and without (ReViSE$^\dagger$) $\mathcal{L}_{MMD}$. The results of MMD distance w.r.t. the number of iterations are shown in Fig. 3. We clearly observe that the red curve (ReViSE) has consistently lower value than the blue curve (ReViSE$^\dagger$). Moreover, based on the previous results, ReViSE always performs better than ReViSE$^\dagger$. Hence aligning the distributions across visual and textual codes can better associate cross-modal information and thus lead to more robust visual-semantic embeddings.

Table II. Inductive and transductive zero-shot recognition using top-1 classification accuracy (%).

| Dataset | AwA | | | CUB | | | average |
|---|---|---|---|---|---|---|---|
| *attributes* | *att* | *w2v* | *glo* | *att* | *w2v* | *glo* | top-1 acc. |
| test data not available during training | | | | | | | |
| SOC [30] | 58.6 | 50.8 | 68.0 | 34.7 | 30.9 | 30.6 | 45.6 |
| ConSE [29] | 59.0 | 53.2 | 49.8 | 33.6 | 28.8 | 30.8 | 42.5 |
| SSE [49] | 63.8 | 58.6 | 65.8 | 31.8 | 27.9 | 25.4 | 45.6 |
| SJE [2] | 66.7 | 52.1 | 58.8 | 50.1 | 28.4 | 24.2 | 46.7 |
| ESZSL [37] | 76.8 | 62.2 | 67.7 | 50.3 | 33.4 | 34.1 | 54.1 |
| JLSE [50] | 71.8 | 64.0 | 68.0 | 33.7 | 28.0 | 27.1 | 48.8 |
| LatEm [48] | 71.9 | 61.1 | 62.9 | 45.5 | 31.8 | 32.5 | 51.0 |
| Sync [7] | 72.9 | 62.0 | 67.0 | 48.7 | 31.2 | 32.8 | 52.4 |
| MTE [6] | 77.3 | - | - | 43.3 | - | - | - |
| DeViSE [9] | 67.4 | 67.0 | 66.7 | 40.8 | 28.8 | 25.6 | 49.3 |
| CMT [41] | 67.6 | 69.5 | 68.0 | 42.4 | 29.6 | 25.7 | 50.5 |
| test data available during training | | | | | | | |
| TMV [10] | 89.0 | 69.0 | 88.7 | 51.2 | 32.5 | **38.9** | 61.6 |
| SMS$_{ESZSL}$ [12] | 89.6 | 78.0 | 82.9 | 52.3 | **34.7** | 32.3 | 61.6 |
| DeViSE* [9] | 90.7 | 84.8 | 88.0 | 41.4 | 31.6 | 26.9 | 60.6 |
| CMT* [41] | 89.4 | 87.8 | 81.8 | 43.1 | 31.8 | 28.9 | 60.5 |
| ReViSE$^{\dagger\dagger}$ | 92.1 | 92.3 | 90.3 | 62.4 | 30.0 | 27.5 | 65.8 |
| ReViSE$^\dagger$ | 92.8 | 92.6 | 91.7 | 62.7 | 31.8 | 28.9 | 66.8 |
| ReViSE$^c$ | 73.0 | 67.0 | 73.4 | 53.7 | 26.4 | 28.2 | 53.6 |
| ReViSE | **93.4** | **93.5** | **92.2** | **65.4** | 32.4 | 31.5 | **68.1** |

## V. Comparing with recent state-of-the-art methods

In our main paper, we focus on comparing with deep-embeddings methods. In Table II, we compare other methods for inductive and transductive zero-shot learning. Note that SMS$_{ESZSL}$ adopts ESZSL for its initialization.

## VI. Remarks on Contractive Loss

We find that adding contractive loss to textual auto-encoder doesn't provide much benefit. One possible reason may be the limited number of textual features (200 for CUB). On the other hand, the number of visual features is large (11, 786 for CUB).

## References


[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous systems, 2015. *Software available from tensorflow. org*, 1, 2015. 1

[2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009. 1

[3] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *ICLR*, 2015. 1