A. Loss calculation algorithm

B. The Analysis of Error Rates

First define the misclassification of unseen as seen classes for the classifier $\widehat{y}_\alpha$, based on $d_\alpha(\mathbf{z}_v, \mathbf{p}(y))$:

$$\mathbb{P}\{\widehat{y}_\alpha \in \mathcal{Y}^{tr} | y_v \in \mathcal{Y}^{ts}\} = \mathbb{P}\left\{(1+\alpha) \min_{y \in \mathcal{Y}^{tr}} d(\mathbf{z}_v, \mathbf{p}(y)) < \min_{y \in \mathcal{Y}^{ts}} d(\mathbf{z}_v, \mathbf{p}(y)) | y_v \in \mathcal{Y}^{ts}\right\} , \tag{3}$$

We show that $\mathbb{P}\{\widehat{y} \in \mathcal{Y}^{tr} | y_v \in \mathcal{Y}^{ts}\} \geq \mathbb{P}\{\widehat{y}_\alpha \in \mathcal{Y}^{tr} | y_v \in \mathcal{Y}^{ts}\}$. Let us define $\delta_{tr} \equiv \min_{y \in \mathcal{Y}^{tr}} d(\mathbf{z}_v, \mathbf{p}(y))$ and $\delta_{ts} \equiv \min_{y \in \mathcal{Y}^{ts}} d(\mathbf{z}_v, \mathbf{p}(y))$, then Equation (3) can be rewritten as:

$$\mathbb{P}\{\widehat{y}_\alpha \in \mathcal{Y}^{tr} | y_v \in \mathcal{Y}^{ts}\} = \mathbb{P}\left\{(1+\alpha)\delta_{tr} < \delta_{ts} | y_v \in \mathcal{Y}^{ts}\right\} . \tag{4}$$

Let us consider the probability of event $\delta_{tr} < \delta_{ts}$ and decompose it as follows:

$$\begin{aligned}
&\mathbb{P}\left\{\delta_{tr} < \delta_{ts} | y_v \in \mathcal{Y}^{ts}\right\} = \mathbb{P}\left\{(1+\alpha)\delta_{tr} < (1+\alpha)\delta_{ts} | y_v \in \mathcal{Y}^{ts}\right\} \\
&= \mathbb{P}\left\{(1+\alpha)\delta_{tr} < \delta_{ts} \cup \delta_{ts} \leq (1+\alpha)\delta_{tr} < (1+\alpha)\delta_{ts} | y_v \in \mathcal{Y}^{ts}\right\} \\
&= \mathbb{P}\left\{(1+\alpha)\delta_{tr} < \delta_{ts} | y_v \in \mathcal{Y}^{ts}\right\} + \mathbb{P}\left\{\delta_{ts} \leq (1+\alpha)\delta_{tr} < (1+\alpha)\delta_{ts} | y_v \in \mathcal{Y}^{ts}\right\} \\
&- \mathbb{P}\left\{(1+\alpha)\delta_{tr} < \delta_{ts} \cap \delta_{ts} \leq (1+\alpha)\delta_{tr} < (1+\alpha)\delta_{ts} | y_v \in \mathcal{Y}^{ts}\right\} \\
&= \mathbb{P}\left\{(1+\alpha)\delta_{tr} < \delta_{ts} | y_v \in \mathcal{Y}^{ts}\right\} + \mathbb{P}\left\{\delta_{ts} \leq (1+\alpha)\delta_{tr} < (1+\alpha)\delta_{ts} | y_v \in \mathcal{Y}^{ts}\right\} .
\end{aligned}$$

The transitions are based on the relationship between probabilities of arbitrary events A and B, $\mathbb{P}\{A \cup B\} = \mathbb{P}\{A\} + \mathbb{P}\{B\} - \mathbb{P}\{A \cap B\}$, and in our case $\mathbb{P}\{A \cap B\} = 0$. This implies that:

$$\begin{aligned}
\mathbb{P}\{\widehat{y}_\alpha \in \mathcal{Y}^{tr} | y_v \in \mathcal{Y}^{ts}\} &= \mathbb{P}\{\widehat{y} \in \mathcal{Y}^{tr} | y_v \in \mathcal{Y}^{ts}\} - \mathbb{P}\left\{\frac{\delta_{ts}}{(1+\alpha)} \leq \delta_{tr} < \delta_{ts} | y_v \in \mathcal{Y}^{ts}\right\} \\
&\leq \mathbb{P}\{\widehat{y} \in \mathcal{Y}^{tr} | y_v \in \mathcal{Y}^{ts}\}.
\end{aligned} \tag{5}$$

We have just shown that for a non-negative α the probability of misclassifying an image from an unseen class as one of the seen classes is smaller for the decision rule $\widehat{y}_\alpha$ than for the original decision rule $\widehat{y}$. In fact, we can make a stronger claim. Since δ_{ts} and δ_{tr} are non-negative, it is clear that the length of interval $[\delta_{ts}/(1+\alpha), \delta_{ts})$ increases as α increases, and hence probability that δ_{tr} falls in this interval is non-decreasing with increasing α. Thus we have for any $0 \leq \alpha_1 \leq \alpha_2$, $\mathbb{P}\{\widehat{y}_{\alpha_1} \in \mathcal{Y}^{tr} | y_v \in \mathcal{Y}^{ts}\} \geq \mathbb{P}\{\widehat{y}_{\alpha_2} \in \mathcal{Y}^{tr} | y_v \in \mathcal{Y}^{ts}\}$, *i.e.* $\mathbb{P}\{\widehat{y}_\alpha \in \mathcal{Y}^{tr} | y_v \in \mathcal{Y}^{ts}\}$ is a monotone non-increasing function of α and we can reduce it by increasing α.

Algorithm 1 Loss calculation for a single optimization iteration of the proposed method. N is the number of instances in the training set $\mathcal{S}^{tr}$, B is the number of instances per batch, C is the number of classes in the train set. RANDOMSAMPLE$(\mathcal{S}, B)$ denotes a set of B elements chosen uniformly at random from a set $\mathcal{S}$, without replacement.

Input: Training set $\mathcal{S}^{tr} = \{(v_1, t_1, y_1), \ldots, (v_N, t_N, y_N)\}$, $\lambda \in [0, 1]$, $\kappa \in [0, 1]$.
Output: The loss $J(\phi, \theta)$ for a randomly sampled training batch.

$\mathcal{I} \leftarrow$ RANDOMSAMPLE$(\{1, \ldots, N\}, B)$ ▷ Select B instance indices for batch
$J_{TC}(\theta), J_{IC}(\phi) \leftarrow 0, 0$ ▷ Initialize classification losses
for i in $\mathcal{I}$ **do**
  $\mathbf{z}_{v_i}, \mathbf{z}_{t_i} \leftarrow f_\phi(v_i), f_\theta(t_i)$ ▷ Embed images and texts
  $p_I \leftarrow \mathrm{softmax}(\mathbf{W}_I \mathbf{z}_{v_i} + \mathbf{b}_I)$ ▷ Image classifier probabilities
  $p_T \leftarrow \mathrm{softmax}(\mathbf{W}_T \mathbf{z}_{t_i} + \mathbf{b}_T)$ ▷ Text classifier probabilities
  $J_{TC}(\theta) \leftarrow J_{TC}(\theta) + \frac{1}{B} \mathrm{crossentropy}(p_T, y_i)$ ▷ Text classification loss
  $J_{IC}(\phi) \leftarrow J_{IC}(\phi) + \frac{1}{B} \mathrm{crossentropy}(p_I, y_i)$ ▷ Image classification loss
end for
$J_{TR}(\phi, \theta), J_{IR}(\phi, \theta) \leftarrow 0, 0$ ▷ Initialize retrieval losses
for i in $\mathcal{I}$ **do**

$$J_{TR}(\phi, \theta) \leftarrow J_{TR}(\phi, \theta) + \frac{1}{B} \left[d(\mathbf{z}_{v_i}, \mathbf{z}_{t_i}) + \log \sum_{j \in \mathcal{I}} \exp(-d(\mathbf{z}_{v_i}, \mathbf{z}_{t_j})) \right]$$ ▷ Text retrieval loss

$$J_{IR}(\phi, \theta) \leftarrow J_{IR}(\phi, \theta) + \frac{1}{B} \left[d(\mathbf{z}_{v_i}, \mathbf{z}_{t_i}) + \log \sum_{j \in \mathcal{I}} \exp(-d(\mathbf{z}_{t_i}, \mathbf{z}_{v_j})) \right]$$ ▷ Image retrieval loss

end for
$J(\phi, \theta) \leftarrow \lambda J_{TR}(\phi, \theta) + (1 - \lambda) J_{IR}(\phi, \theta)$ ▷ Add retrieval loss to the total loss
$J(\phi, \theta) \leftarrow (1 - \kappa) J(\phi, \theta) + \frac{\kappa}{2} (J_{TC}(\theta) + J_{IC}(\phi))$ ▷ Add classification loss to the total loss

C. Constructing validation sets

The validation set is constructed by further splitting the train set on CUB and FLOWERS. For example, CUB has a train set of 5875 images from 100 seen classes and a validation set of 2946 images from 50 unseen classes. We further divide the train set into 4700 train images from 100 seen classes, 1175 seen validation images (4700 + 1175 = 5875) and we use all the 2946 images from 50 classes as the unseen validation set.

D. Architecture and Training Details

The text feature extractor is built by cascading two residual CNN blocks, followed by a BiLSTM. Each block has 3 convolutional/batch norm layers. The number of filters in the blocks is 128 and 256, BiLSTM has 512 filters for forward and backward branches (1024 total). All variables in the convolutional stack (including the batch normalization parameters γ and β) are L2-penalized with weight 0.001. The image feature extractor is a ResNet-101 with fixed weights pretrained on the split of ImageNet proposed by Xian et al. [30]. In this work we use precomputed image features, available in [28] for CUB and in [29] for FLOWERS. Image and text features are projected in the common embedding space of size 1024 with FC layers and no non-linearity. They are preceded with a dropout of 0.25. The trainable components of the model are trained for 150k batches of size 32 using SGD with initial learning rate of 0.1 that is annealed by a factor of 10 every 50k batches. For each batch, we sample 32 instances, each instance includes a vector of precomputed ResNet-101 features and 10 text descriptions corresponding to it, according to the original dataset definition [20]. All 10 text descriptions are processed via the CNN/LSTM stack and the resulting embeddings are average pooled to create a vector representation of length 1024.