All Labels Are Not Created Equal : Enhancing Semi-supervision via Label Grouping and Co-training

Supplementary Material

A. Probabilistic Interpretation of SemCo

In this section, we provide a probabilistic interpretation of our method described in Sec. 4.[1]

We start by recalling the general form of recent pseudo-labeling methods captured by the below formalization for the unsupervised loss:

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{-1}{\mu \cdot n} \sum_{j=1}^{\mu n} \eta_j \log p(y = \hat{y}_j \,|\, \boldsymbol{u}_j, \boldsymbol{\theta}), \tag{1}$$

which is typically added to the loss for the labeled data. To reiterate, we use $\boldsymbol{\theta}$ to represent learnable model parameters. For an unlabelled input $\boldsymbol{u}_i$, $\hat{\boldsymbol{y}}$ denotes the pseudo-label and η is an arbitrary function. The choice of $\hat{y}$ and η gives rise to three distinct variations of pseudo-labeling that can be written as,

$$\hat{y}_j = \arg\max_{y'} p(y = y' \,|\, \boldsymbol{u}_j, \boldsymbol{\theta}) \text{ with } \eta_j = 1, \tag{2}$$

$$\hat{y}_j = \frac{\exp(f_\ell(\boldsymbol{u}_j)/T)}{\sum_k \exp(f_k(\boldsymbol{u}_j)/T)} \text{ with } \eta_j = 1, \text{ and} \tag{3}$$

$$\hat{y}_j = \arg\max_{y'} p(y = y' \,|\, \boldsymbol{u}_j, \boldsymbol{\theta}) \tag{4}$$

$$\text{with } \eta_j = \mathbb{1}(p(y = \hat{y}_j \,|\, \boldsymbol{u}_j, \boldsymbol{\theta}) \geq \tau).$$

The first approach (Eqn. 2) corresponds to naive pseudo-labeling where the class with the highest confidence is used as a pseudo-label regardless of its score, while the second (Eqn. 3) improves on that by employing temperature sharpening [3] via T. For brevity, we use f_ℓ is the onehot logit for class ℓ. Sharpening the pseudo-label implicitly encourages entropy minimizaton [11] whereby the classifier is encouraged to produce high confidence predictions on the unlabeled data. The third approach (Eqn. 4) adopted in [31, 19] where the unlabeled sample is only retained for pseudo-labeling if the max confidence score exceeds a predefined threshold, τ. This simultaneously encourages entropy minimization[2] and decrease confirmation bias (see Sec. 1) by only retaining high confidence samples.

As opposed to above methods, we additionally propose to take a multi-view approach in which we have y' as different representation of the label as well as a grouping of the similar labels potentially (but not necessarily) obtained from it. We consider this additional label to be conditionally independent of the one-hot representation of the label. Specifically, instead of Eqn. 1, we propose to minimize the objective,

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{-1}{\mu \cdot n} \sum_j \log \Big(p(y = \hat{y}_j \,|\, \boldsymbol{u}_j, \boldsymbol{\theta}) p(y' = \hat{y'}_j \,|\, \boldsymbol{u}_j, \boldsymbol{\theta})^{\eta_j} \Big),$$
$$p(y' = \hat{y'}_j \,|\, \boldsymbol{u}_j, \boldsymbol{\theta}) = \sum_c \underbrace{p(y' = \hat{y'}_j \,|\, c, \boldsymbol{u}_j, \boldsymbol{\theta})}_{\text{additional classification}} \underbrace{p(c \,|\, \mathcal{Y}', \boldsymbol{\theta})}_{\text{grouped semantics}}, \tag{5}$$

where $\mathcal{Y}'$ denotes the collection of the labels which we consider to be conditionally independent of the conventional one-hot label. We use the density-based clustering to calculate $p(c \,|\, \mathcal{Y}', \boldsymbol{\theta})$. Then by using Jensen's inequality, we have the following as the upper-bound on the loss in Eqn. 5:

$$\mathcal{L}(\boldsymbol{\theta}) \geq \frac{-1}{\mu \cdot n} \sum_j \eta_j \Big[\log \big(p(y = \hat{y}_j \,|\, \boldsymbol{u}_j, \boldsymbol{\theta}) \big) + \sum_c \underbrace{\log \Big(p(y' = \hat{y'}_j \,|\, c, \boldsymbol{u}_j, \boldsymbol{\theta}) \Big)}_{\text{Sec. 4.1}} \underbrace{p(c \,|\, \mathcal{Y}', \boldsymbol{\theta})}_{\text{Sec. 4.5}} \Big]. \tag{6}$$

which indicates the log-likelihood of the additional labels are weighted by the grouping of their semantic relationships. We use the f_{sc} in the paper to denote the classifier head that predicts these additional labels.

B. Obtaining Label Embeddings using ConceptNet Knowledge Graph

In this section, we elaborate on the process described in Sec. 4.5 which aims to obtain class label embeddings which correlate well with visual similarity. We start by describing the procedure and then we present some qualitative examples to demonstrate its effectiveness.

[1] All section, table, and figure references are following the original paper numbering.

[2] Note that it is equivalent to sharpening with $T \to 0$

B.1. Procedure

We follow a similar procedure to that described in [33] with one crucial difference. Instead of using the entire ConceptNet graph, we use the graph after filtering it to retain only the relations which imply visual similarity (*see* Sec. 5 for more details).

We start with the filtered graph, the GloVe word embeddings matrix, and the word2vec word embeddings matrix. The process comprises two main steps: 1) retrofitting each of the GloVe and word2vec embeddings using the ConceptNet filtered graph to obtain two new sets of embeddings, and 2) combining the two retrofitted sets to obtain our final hybrid embeddings set.

Retrofitting Given the filtered graph and a matrix of word embeddings, the aim is to infer for each term/word a new embedding vector $\boldsymbol{v}_i$ which is close to the original vector $\hat{\boldsymbol{v}}_i$ but also close to the term neighbors in the graph with edges E. This can be achieved by minimizing the following objective function.

$$E(v) = \sum_{i=1}^{n} \left[\alpha_i \left\| \boldsymbol{v}_i - \hat{\boldsymbol{v}}_i \right\|^2 + \sum_{(i,j) \in E} \beta_{ij} \left\| \boldsymbol{v}_i - \boldsymbol{v}_j \right\|^2 \right];$$

with $\alpha_i = 1$ if term i is present in the embeddings vocabulary and zero otherwise; and β_{ij} denoting the weight of the edge connecting term i and term j. Note that the use of α allows optimizing the above objective for terms which appear in the knowledge graph even if it is not present in the vocabulary of the word embeddings [32]. To minimize the above function, we follow the iterative algorithm originally suggested by Faruqui et al. [8] and later extended by Speer et al. [32]. We perform such optimization twice: once for the GloVe embeddings and another for the word2vec.

Combining the Two Sets After applying retrofitting to both matrices, we combine them by finding a globally linear projection that aligns the results based on their common vocabulary. As inspired by [45] and [33], to find such projection, we concatenate the columns of the two matrices and use SVD to reduce their dimensionality to 128. Such alignment allows us to deduce compatible embeddings for terms which appear in one of the vocabularies but not the other. This alignment and merging give rise to a hybrid set of embeddings which combines all three sources: GloVe, word2vec, and ConceptNet filtered graph. We use this set as the basis for establishing the prior on visual similarity among a given set of class labels (*see* Sec. 4).

Handling Out-of-Vocabulary Labels Our obtained embeddings vocabulary consists of approximately 500k different terms and hence provides a reasonable coverage for most of the class labels. However, it might sometimes be the case that one or more of the class labels are missing from the vocabulary. In such event, we employ a fall-out strategy to find the most reasonable alternative. We present the flowchart for our fall-out strategy in Fig. 5.

B.2. Label Grouping Examples

As mentioned in Sec. 4, we apply density-based clustering on the class labels embeddings to group the labels into visually similar concepts. To demonstrate the effectivness of the retrofitted embeddings in capturing said similarity, we compare the clustering output if we use the retrofitted embeddings as opposed to if we use the GloVe distributional embeddings without retrofitting. We perform this comparison for Mini-ImageNet (*see* Table 6,7), CIFAR-100 (*see* Table 8,9), and DomainNet classes (*see* Table 10,11). Note that we only report the groups having more than one member and we omit single-member groups. We observe that in all three cases, clustering the retrofitted embeddings produces groups which largely match our intuition about visual similarity. On the other hand, we notice that clustering the non-retrofitted GloVe embeddings results in grouping labels which usually appear in similar context, even if they are not visually similar. For example, in Table 9, we observe that "sea" was grouped with other classes which are contextually related to "sea", yet bear no visual similarity to it. This is due to the fact that GloVe embeddings are learnt in a way that captures distributional semantics rather than visual semantics. However, when the GloVe embeddings are retrofitted with the ConceptNet filtered graph, we witness an improved grouping which aligns better with visual semantic similarity.

C. Implementation Details

Hyperparameters In our preliminary experiments, we mostly found that our method is not sensitive to the hyperparameters, so we tuned their values via a validation set on a single experiment (CIFAR100 - 2500 labels) then fixed them across all other experiments to the values shown in Table 5. The only exception is the density-based clustering parameter ϵ. The number of clusters (i.e. label groups) is automatically decided based on ϵ, which denotes the maximum cosine distance between two embedding vectors for one to be considered in the same neighbourhood as the other. The larger the value of ϵ, the more aggressive the grouping is (i.e. the more members in each group). Accordingly, we tune ϵ individually for each dataset. We find that $\epsilon = 0.2$ works well for all datasets except Mini-ImageNet where we use $\epsilon = 0.3$ instead. In Fig. 8, we demonstrate the effect of varying ϵ on the error rate using a single split of CIFAR-100 (2500 labeled instances) when training for 100 epochs.

Semantic Classifier Loss We use two different loss functions for our two classifiers, i.e. *cosine loss* for the *Semantic Classifier*, and *cross-entropy* for the *One-Hot Classifier* (*see* Sec. 4). It is, hence, important to consider the scale of

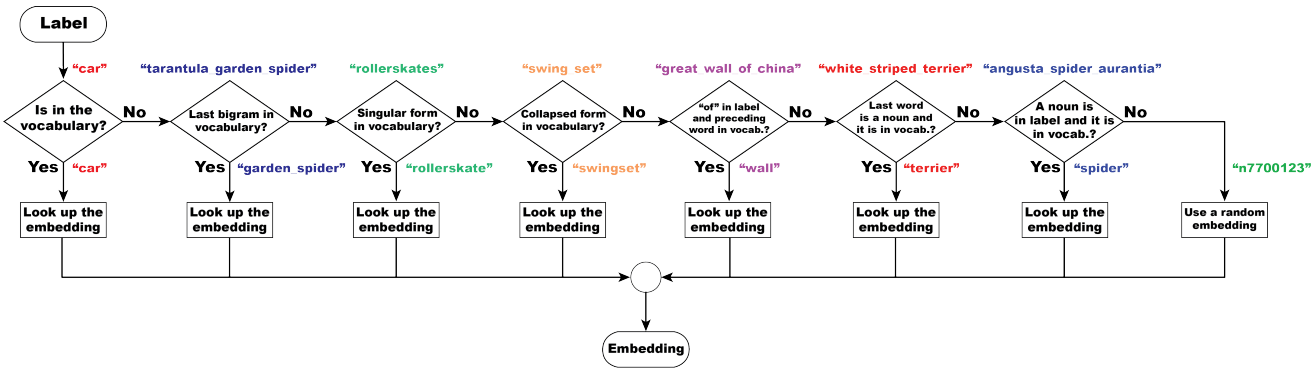


Figure 5: A flowchart describing our label embedding lookup strategy aiming to find the most reasonable embedding for a given class label. We include demonstrative examples for each of the fall-out cases.

Table 5: Hyper-parameters values across all our experiments

Hyper-parameter	Description	Value
λ_u	Unlabeled loss coefficient	1.00
λ_{co}	Co-training loss coefficient	1.00
τ_e	Pseudo-labeling confidence threshold for the *Semantic Classifier*	0.70
τ_o	Pseudo-labeling confidence threshold for the *One-Hot Classifier*	0.95
$batch\_size$	Number of labeled images per batch	64
μ	Ratio between number of unlabeled and labeled images in each batch	3
$images\_per\_epoch$	Number of labeled images per epoch	64×1024
num_epochs	Number of epochs of training	300
lr	learning rate max value (10 epochs warmup then cosine decay)	0.03
$weight\_decay$	Weight decay regualrization coefficient	5.00×10^{-4}
$momentum$	Nesterov momentum for SGD optimizer	0.90
emb_dim	Dimensionality of the label embeddings	128
ϵ	DBSCAN clustering coefficient denoting the maximum distance between two samples for one to be considered as in the neighborhood of the other	0.20

both losses so that one doesn't dominate over the other. *Cosine loss* values are bounded between 0 and 2 while *cross-entropy* values are not. Accordingly, we multiply the *Semantic Classifier* loss by a factor of 3 before applying the back propagation step. We obtained such value by using a held-out validation set on CIFAR-100 (1000 labeled examples) and we fixed it across all other experiments and datasets.

Augmentations As described in Sec. 2, we make use of two types of augmentations, i.e. weak and strong. For weak augmentations, we use random cropping and padding, and random horizontal flips. As for the strong augmentations, we use the RandAugment [6] list of transformations for both our system and the FixMatch baseline.

Inference Since we train two classifiers in our method, during inference time, we can choose one of three options for inference: 1) use the *One-Hot Classifier* prediction, 2) use the *Semantic Classifier* prediction, 3) Average the softmax scores of the two classifiers and use the combined score for prediction. During our validations, we found that the former approach always yields marginally better results, so we use it as our basis for inference. Finally, We also use an exponential moving average of model weights with a decay rate of 0.999.

D. Further Pseudo-labeling Analysis

In Fig. 2 in the main text, we present a comparison between pseudo-labeling statistics (on Mini-ImageNet dataset) of our method versus the baseline. In this section, we elaborate about the experimental setup for obtaining these statistics. Additionally, we provide similar analysis on CIFAR-100 dataset.

For a given dataset, we run our algorithm for 10 epochs of unlabeled data and we capture a highly granular view of the pseudo-labeling statistics for each mini-batch. Consequently, we calculate two metrics: 1) we use the true labels of the unlabeled data samples (which we originally ignore to emulate an SSL setting) to measure the true pseudo-labeling accuracy for each of the classes in the dataset, and 2) we use the classifier confidence scores to calculate the pseudo-

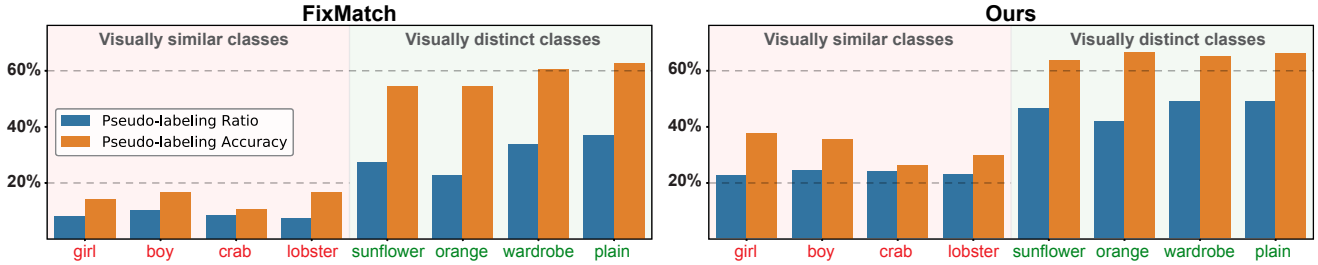
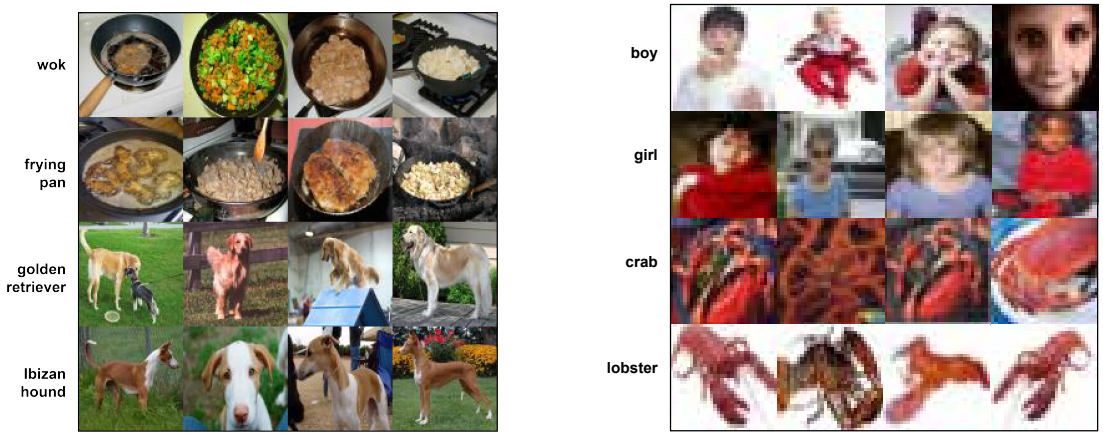


Figure 6: CIFAR-100 confidence-based pseudo-labeling comparison between the baseline (left) and our method (right). *Accuracy* values show how much, on average, pseudo-labels for a given class match the true label, while *Ratio* values show the percentage of samples of a given class which are retained for pseudo-labeling (i.e. with confidence score above the threshold). The two metrics are calculated for the 4 most (red) and least (green) visually similar classes over the first 10 epochs of training.



(a) Mini-ImageNet (b) CIFAR-100

Figure 7: The most confused images for the 4 most visually similar classes of Mini-ImageNet (left) and CIFAR-100 (right). The caption next to each image group denotes the true class to which the image group belongs.

labeling ratio for each class, which represents the amount of unlabeled samples exceeding the confidence threshold and thereby are retained for pseudo-labeling. We repeat the same procedure and measure the same metrics for our baseline [31]. We, then, display those metrics for the 4 classes which were deemed by our clustering method as the most visually similar concepts. Conversely, we also display them for the 4 classes which are deemed most visually distinct. In Fig. 6, we report these metrics for CIFAR-100 dataset (see Fig. 2 for Mini-ImageNet). Additionally, through the same experimental setup described above, we keep track of pseudo-labeling statistics for each individual unlabeled image. We report in Fig. 7 the most confused images among the 4 most visually similar classes for both datasets. We define confusion as the average number of times a given image is incorrectly pseudo-labeled as another class within the 4 classes (*e.g.* *"boy"* pseudo-labeled as *"girl"*).

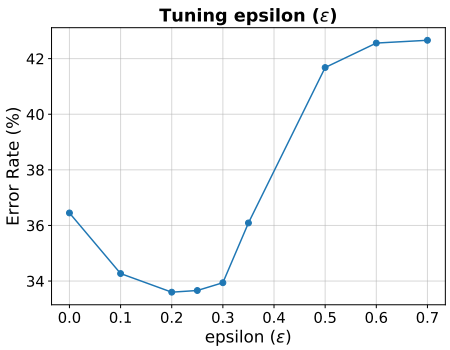


Figure 8: Error rates for different values of ϵ. $\epsilon = 0$ corresponds to no label grouping, while $\epsilon > 0.7$ corresponds to grouping all labels into a single cluster.

Table 6: Mini-ImageNet class groups obtained by clustering the retrofitted embeddings.

Group	Members
Group 1	'French_bulldog', 'Ibizan_hound', 'Saluki', 'Walker_hound', 'golden_retriever', 'malamute', 'miniature_poodle'
Group 2	'catamaran', 'yawl'
Group 3	'frying_pan', 'wok'
Group 4	'horizontal_bar', 'parallel_bars'

Table 7: Mini-ImageNet class groups obtained by clustering the GloVe embeddings.

Group	Members
Group 1	'African_hunting_dog', 'French_bulldog', 'Ibizan_hound', 'Walker_hound', 'golden_retriever', 'miniature_poodle'
Group 2	'combination_lock', 'garbage_truck', 'horizontal_bar', 'parallel_bars', 'pencil_box', 'street_sign'

Table 8: CIFAR-100 class groups obtained by clustering the retrofitted embeddings.

Group	Members
Group 1	'aquarium_fish', 'flatfish', 'trout'
Group 2	'bicycle', 'motorcycle'
Group 3	'boy', 'girl'
Group 4	'crab', 'lobster'
Group 5	'dolphin', 'whale'
Group 6	'man', 'woman'
Group 7	'oak_tree', 'pine_tree'

Table 9: CIFAR-100 class groups obtained by clustering the GloVe embeddings.

Group	Members
Group 1	'aquarium_fish', 'crab', 'dolphin', 'lobster', 'sea', 'shark', 'trout', 'turtle', 'whale'
Group 2	'elephant', 'fox', 'house', 'leopard', 'lion', 'man', 'pickup_truck', 'road', 'table', 'tiger', 'tractor', 'wolf', 'woman'
Group 3	'bicycle', 'motorcycle'
Group 4	'bus', 'train'
Group 5	'crocodile', 'lizard', 'snake'
Group 6	'raccoon', 'squirrel'
Group 7	'oak_tree', 'pine_tree'

Table 10: DomainNet class groups obtained by clustering the retrofitted embeddings.

Group	Members
Group 1	'basketball', 'soccer_ball'
Group 2	'beard', 'goatee', 'moustache'
Group 3	'bicycle', 'motorbike'
Group 4	'birthday_cake', 'cake'
Group 5	'bracelet', 'necklace'
Group 6	'cello', 'clarinet', 'guitar', 'piano', 'saxophone', 'trombone', 'trumpet', 'violin'
Group 7	'crab', 'lobster'
Group 8	'oven', 'stove'
Group 9	'pants', 'shorts', 'underwear'
Group 10	'pickup_truck', 'truck'
Group 11	'wine_bottle', 'wine_glass'

Table 11: DomainNet class groups obtained by clustering the GloVe embeddings.

Group	Members
Group 1	'airplane', 'helicopter'
Group 2	'ambulance', 'hospital'
Group 3	'apple', 'blackberry'
Group 4	'asparagus', 'broccoli', 'peas'
Group 5	'axe', 'knife', 'sword'
Group 6	'backpack', 'suitcase'
Group 7	'banana', 'blueberry', 'pineapple', 'strawberry'
Group 8	'baseball', 'basketball'
Group 9	'baseball_bat', 'bat'
Group 10	'bathtub', 'sink', 'toilet'
Group 11	'beard', 'goatee', 'moustache'
Group 12	'bracelet', 'necklace'
Group 13	'bread', 'cake', 'cookie', 'peanut', 'pizza', 'sandwich'
Group 14	'bus', 'train'
Group 15	'carrot', 'onion', 'potato'
Group 16	'crab', 'dolphin', 'fish', 'lobster', 'octopus', 'shark', 'whale'
Group 17	'crayon', 'pencil'
Group 18	'fireplace', 'microwave', 'oven', 'stove'
Group 19	'jacket', 'pants', 'shorts', 'sweater', 'underwear'
Group 20	'raccoon', 'squirrel'
Group 21	'radio', 'television'
Group 22	'snowflake', 'snowman'
Group 23	'toothbrush', 'toothpaste'