Towards Learning Image Similarity from General Triplet Labels

Supplementary Material

Dataset	Protocol	Number of Image Triplets		
		train.	val.	test
THINGS	regular	1.20M	299.2K	23.2K
	teacher	598.4K	149.6K	11.6K
	student	640B	23B	23B
IHSJC	regular	2.96M	739.1K	60.0K
	teacher	0.99M	246.4K	20.0K
	student	4.5T	166B	166B
Yummly	regular	77.3K	19.3K	1.4K
	teacher	77.3K	19.3K	1.4K
	student	102.7K	3.4K	3.4K

Table 5. Dataset split statistics by protocol. Through supervision from the teachers, the student models access far more triplets than the directly trained models.

6. Dataset Statistics

Detailed dataset statistics can be found in table 5.

Both the regular and the teacher protocol essentially **partition the existing set of labeled triplets**, discarding some of them. The regular protocol has 2x and 3x more image triplets than the teacher protocol for the THINGS and IHSJC datasets respectively, as it creates 2 and 3 image triplets for each class triplet. The Yummly dataset does not have class information, so the regular protocol simply uses the labeled image triplets, exactly like the teacher protocol.

The student protocol **makes labeled triplets from all the groups of three images** that are fully in the training, validation or test subset. Because teacher models provide an embedding for each image, it is possible to label any image triplet by comparing the distances between the three embeddings. In theory, the student protocol creates many more image triplets than the regular protocol, for example 4.5 trillion vs. 3 million for the training subset of IHSJC. In practice, batching limits the numbers of triplets formed, but the difference in order of magnitude remains the same.

7. Existing Triplet based Metric Learning Methods on the New Datasets

The existing triplet based metric learning methods do batch sampling and batch mining prior to taking a gradient descent step for a batch. Typically, the methods randomly sample $B/2$ different classes and then 2 images per class, where B is the batch size. They generate $O(B^2)$ triplets by adding to each of the $B/2$ same-class pairs one of the $B-2$ images of a different class. Finally, the most informative triplets are chosen according to some criterion and gradient descent is performed only based on the losses for these triplets, see algorithm 1.

Algorithm 1 Triplet based methods on standard datasets

Require: N dataset size, B batch size; cls image classes
  for $b \leftarrow 1$ to $\frac{N}{B}$ **do**
    sample $\frac{B}{2}$ classes and 2 images per class
    form $\frac{B^2}{2}$ image triplets (i, j, k) with $cls(i) = cls(j)$ and $cls(j) \neq cls(k)$
    select informative triplets
    compute losses for triplets
    do gradient descent
  end for

Algorithm 2 Triplet based methods on new datasets

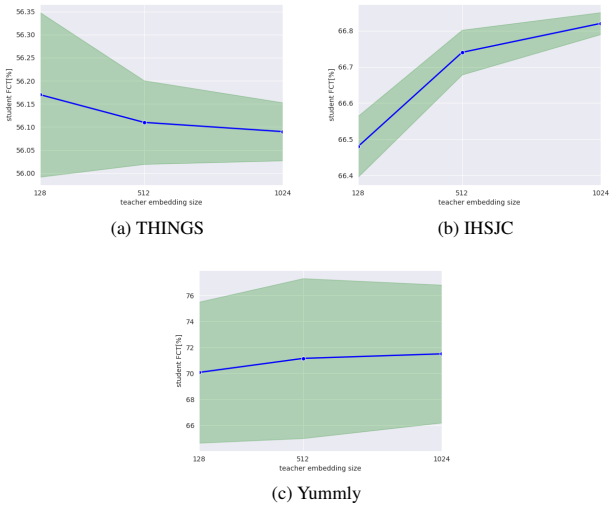
Require: N dataset size, B batch size; $labeled\_triplets$ image triplets (i, j, k) s.t. i and j close and k far
  for $b \leftarrow 1$ to $\frac{N}{B}$ **do**
    sample $\frac{B}{3}$ image triplets from $labeled\_triplets$
    compute losses for triplets
    do gradient descent
  end for

The new datasets in the psychology, neuroscience and human-computer interaction literature are **triplet labeled** as opposed to **class based**, so the scheme described above does not apply. The direct way to form a batch of B images is to randomly sample $B/3$ triplets from the set of labeled triplets, see algorithm 2. Because of the very low density of labeled triplets overall (the 1.46M labeled triplets are only 0.14% of the possible triplets for THINGS), practically no other labeled triplets than the original $B/3$ can be found in a batch. $O(B)$ versus $O(B^2)$ is a dramatic reduction in the number of labeled triplets in a batch (the typical B is over 100), which makes it impossible to mine informative triplets. The existing triplet based methods collapse to doing gradient descent on batches labeled with small and arbitrary sets of triplets.

Another effect of the low density of labeled triplets is that an epoch of direct training takes a large number of batches. For THINGS, for example, it takes roughly 14,400 batches of size 249 to go through the 1.2M labeled triplets in the training set. By contrast, student training takes 82 batches of size 249 to go through the 20K images in the training set. Student training sees each image once but direct training needs to repeat an image a large number of times to cover all the labeled triplets it appears in. Had the density of labeled triplets been close to 100%, $O(B^3)$ labeled triplets would

Approach	THINGS FCT[%]	IHSJC FCT[%]	Yummly FCT[%]
TS+RC [13]	57.87 ± 0.09	68.13 ± 0.12	76.66 ± 3.76
TS+RTM	57.80 ± 0.09	68.03 ± 0.06	76.23 ± 4.96
TS+RF	58.14 ± 0.09	68.00 ± 0.12	75.46 ± 4.42
TS+RI	57.87 ± 0.05	67.57 ± 0.18	76.18 ± 4.53
TS+RMS [38] (as in [13])	57.82 ± 0.05	67.55 ± 0.07	76.13 ± 3.18
TS+MTT [42]	57.99 ± 0.17	67.67 ± 0.06	76.94 ± 3.95
TS+RKD [25]	**58.18** ± 0.12	**68.17** ± 0.09	**77.50** ± 3.37
TS+STMR	58.16 ± 0.05	67.96 ± 0.19	77.11 ± 4.08

Table 6. Results across datasets and losses with a ViTB backbone. The relative order of losses from table 1 is largely preserved.



(a) THINGS (b) IHSJC

(c) Yummly

Figure 7. The quality of the student model (trained with RKD loss) with respect to the teacher embedding size. Larger sizes produce slight improvements for IHSJC and Yummly and slight regression for THINGS. The change is statistically significant only for IHSJC.

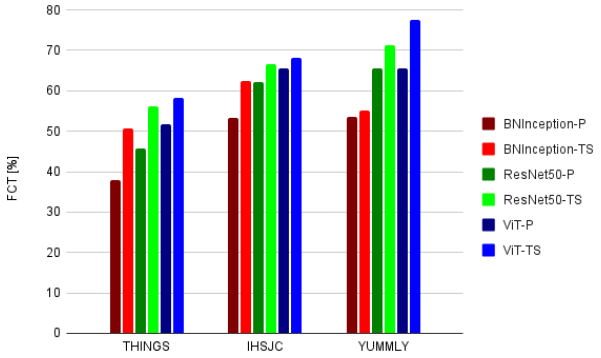


Figure 8. Importance of the backbone. Compared to the pretrained model (-P, dark red, green and blue bars), our approach brings consistent gains. As expected, results with our approach (-TS, light red, green and blue bars) improve for more powerful backbones.

have been formed for any set of B images (many triplets would have had images in common). However, since the density is very low, e.g. 0.1%, batch formation must sample from the set of labeled triplets to produce $O(B)$ triplets in a batch of size B, as few of these triplets have images in common.

8. Dependence on Teacher Embedding Size

We show the dependence of the student model quality on the size of the teacher embedding in figure 7. The loss used to train the student was RKD. Larger embedding sizes are detrimental for THINGS, but beneficial for IHSJC and Yummly. We further observe that for IHJSC the gains are small, ∼0.3%, while for THINGS and Yummly they are not statistically significant (Welch's t-test yields $p > 0.4$ and $p > 0.7$ respectively). The results of our approach are not sensitive to the number of teachers or the size of the teacher embedding.

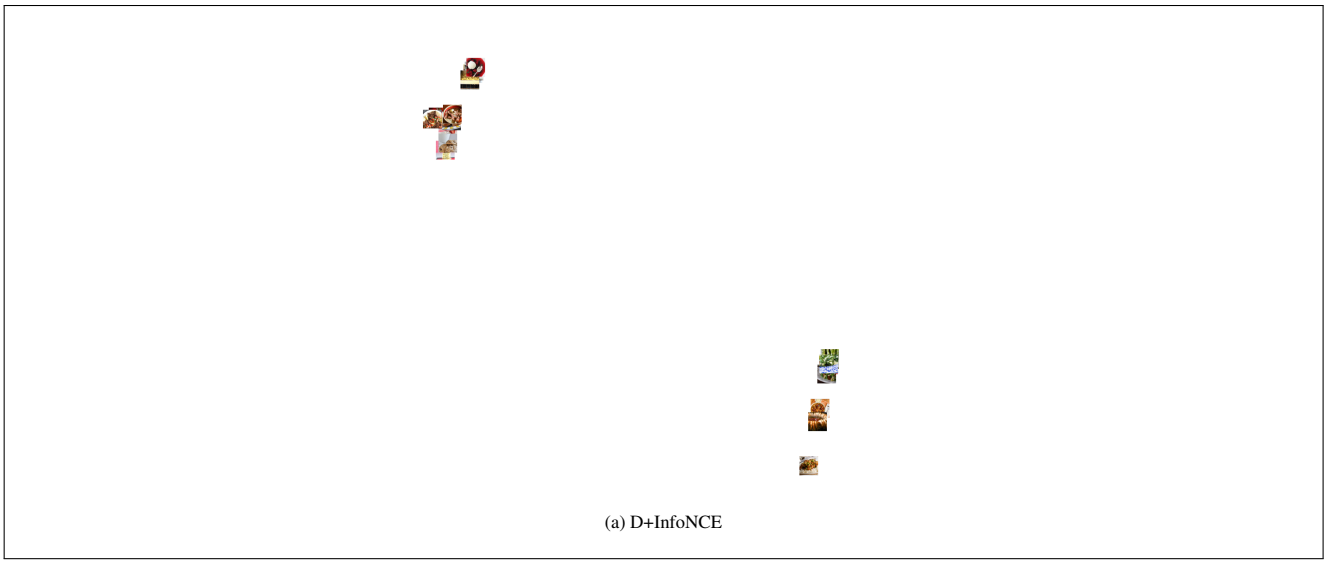
9. Dependence on Backbone

As transformers are becoming the dominant computer vision model architecture, we also experimented with a ViT-base backbone [17]. The results in table 6 show that the relative order of losses from the main results in table 1 stays roughly the same.

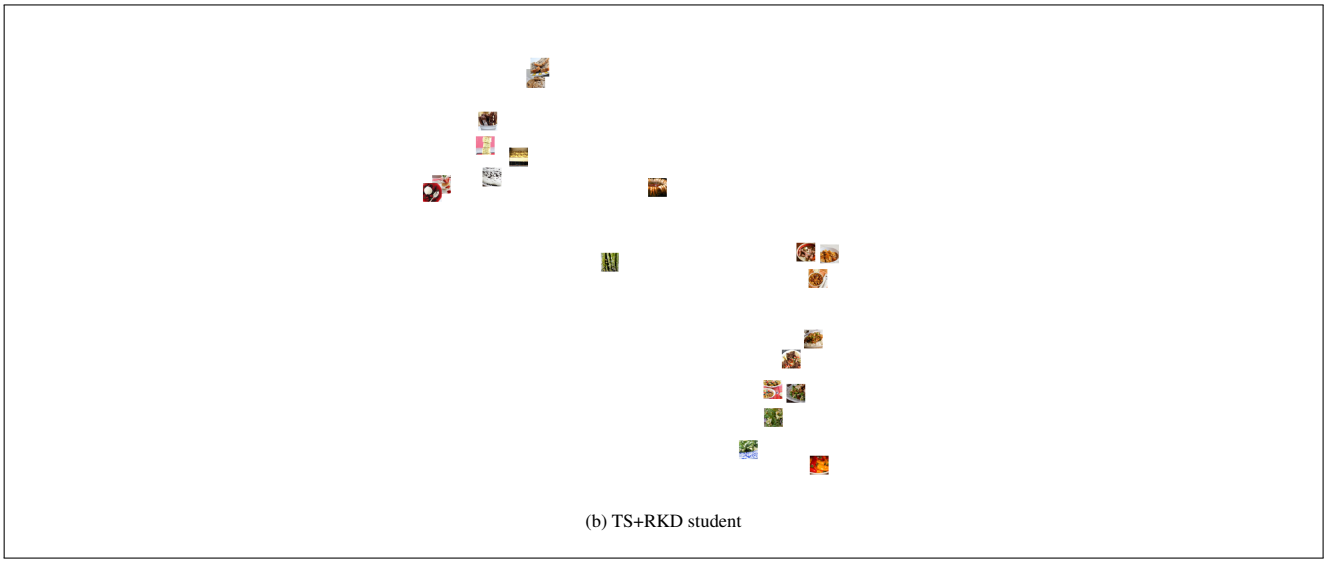
Because the new datasets have not been used for metric learning before, it is important to check that the methods studied are better than just using a pretrained backbone and that better backbones lead to better results. In figure 8, we show the performance of three pretrained models [9] [7] [17] vs. the same models finetuned with our approach with the RKD loss. The gains are consistent across datasets and some are very large, e.g. 10.5% on THINGS with ResNet-50. Also, our approach does better when a stronger backbone is used. There are signs of saturation on IHSJC, but note that for this dataset the student is fairly close to the teachers' metrics, 66% vs 71%.

10. Analysis of Embedding Spaces

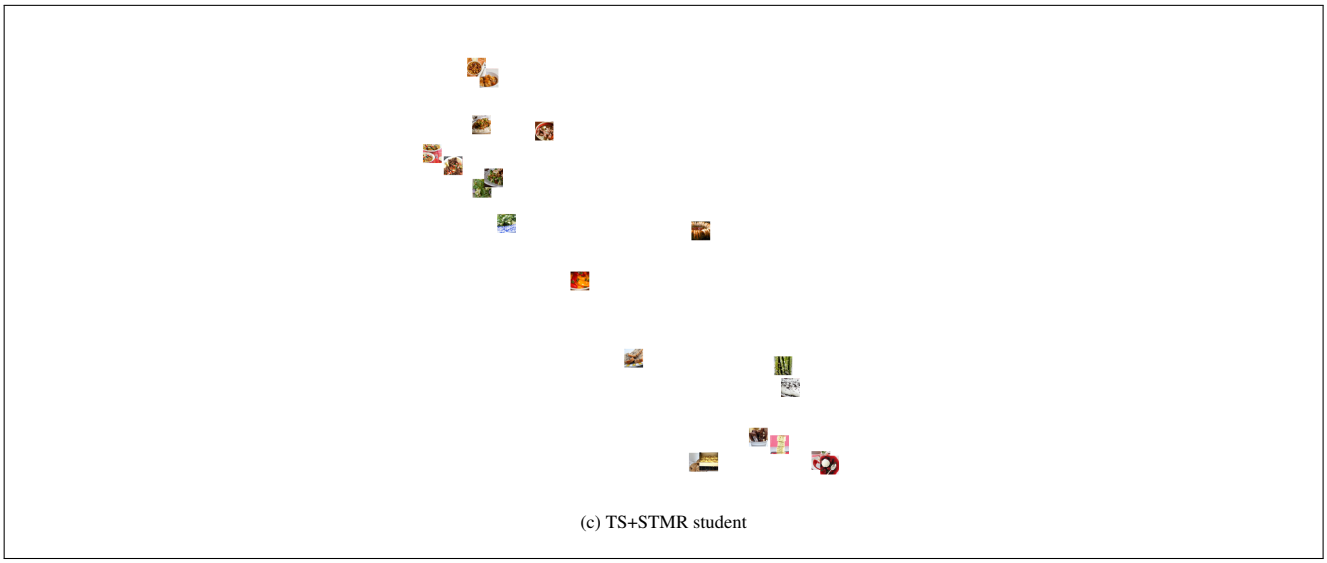
To further understand our approach from a qualitative point of view, we analyzed how it places images in the embedding space. In figures 9, 11 and 13, we show the results of TSNE on embeddings of the test images of the three datasets. The best directly trained loss (D+InfoNCE) and the two best losses in our teacher-student approach (TS+RKD and TS+STMR) are compared. We also compared against teachers in figures 10, 12, and 14, which show the results of TSNE on the validation sets. In each case, the TSNE parameters were set to the same values to ensure meaningful comparison. We observe that the directly trained models tend to produce fewer and denser clusters in the embedding space, and thus to lose detail compared with the teacher and student models in our approach.



(a) D+InfoNCE



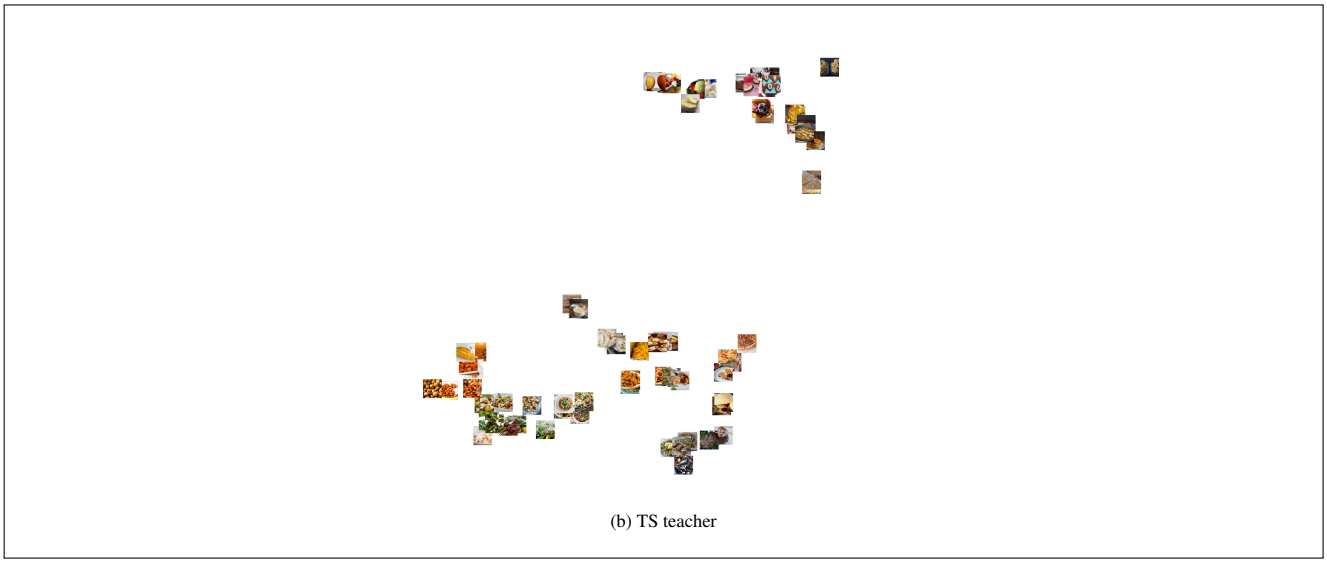
(b) TS+RKD student



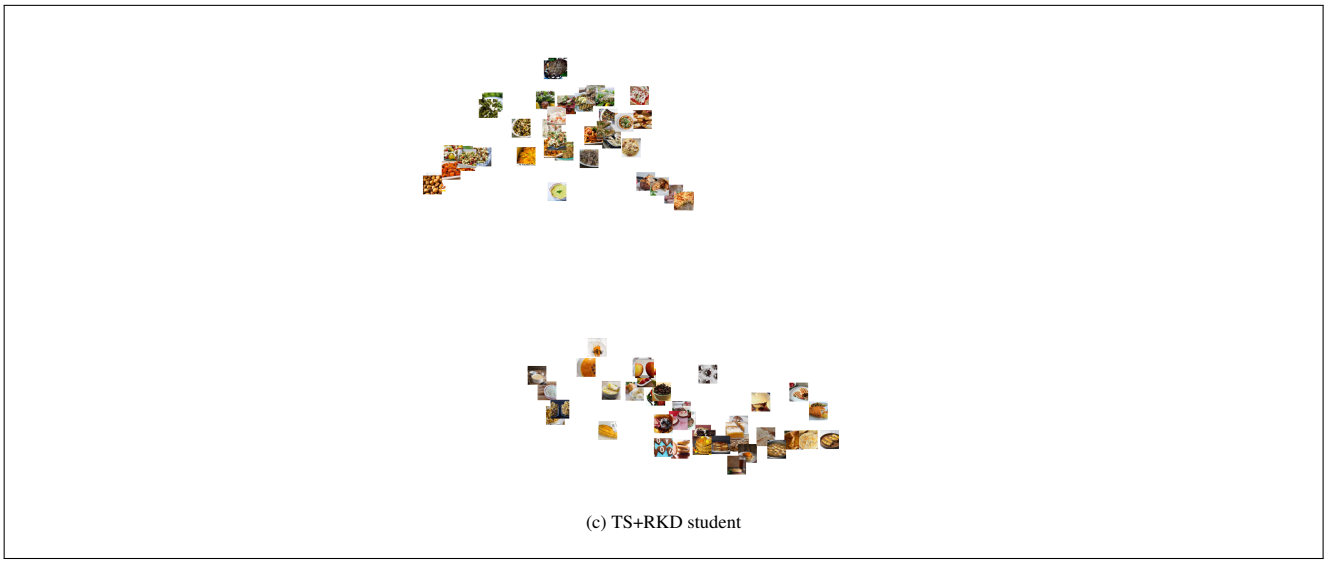
(c) TS+STMR student

Figure 9. Embedding spaces for the Yummly test subset (20 images). The directly trained model creates two clusters while the student models spread images apart more. TSNE parameters: perplexity 5, learning rate 1.

(a) D+InfoNCE



(b) TS teacher



(c) TS+RKD student

Figure 10. Embedding spaces for the Yummly training/validation subset (80 images). The directly trained model creates two clusters, while the teacher and student models spread images apart more. TSNE parameters: perplexity 10, learning rate 1.

(a) D+InfoNCE

(b) TS+RKD student

(c) TS+STMR student

Figure 11. Embedding spaces for the THINGS test subset (5,200 images). The directly trained model creates fewer and denser clusters than the student models. TSNE parameters: perplexity 20, learning rate 1.

(a) D+InfoNCE

(b) TS teacher

(c) TS+RKD student

Figure 12. Embedding spaces for the THINGS training/validation subset (1,483 images, class representatives). The directly trained model creates fewer and denser clusters than the teacher and student models. TSNE parameters: perplexity 20, learning rate 1.

(a) D+InfoNCE

(b) TS+RKD student

(c) TS+STMR student

Figure 13. Embedding spaces for the IHSJC test subset (10,000 images). The directly trained model creates denser clusters than the student models. TSNE parameters: perplexity 25, learning rate 1.

(a) D+InfoNCE

(b) TS teacher

(c) TS+RKD student

Figure 14. Embedding spaces for the IHSJC training/validation subset (800 images, class representatives). The directly trained model creates fewer clusters than the teacher and student models. TSNE parameters: perplexity 20, learning rate 1.