Discovering Image Fragment Relationships

1 Task Description

The goal is to design a model that can group unordered image fragments back to their original source images. Each source image has a square format with 64 x 64 pixels and 3 color channels. For data generation, a single sample is created from 10 images randomly selected from the provided ImageNet64 dataset. Each image is fragmented into 4x4 non-overlapping fragments and the resulting collection of image fragments are shuffled. This results in an unordered collection of 160 image fragments per sample.

2 Approach

The approach utilizes a simple neural network (linear or convolutional model) that is applied on each image fragment (16x16x3 tensor). The linear version of the model flattens the image fragment into a single input that feeds into a linear layer which produces an output feature vector with dimension d. The convolutional version of the model acts upon the image fragment as if it were a full image and consists of two stacked convolutional layers with a ReLU activation in between. A global mean pooling layer acts on the output of the 2nd convolutional layer and produces an output feature vector with dimension d.

The model is trained so that the image fragment features generated by the model can be used for mapping back to the original source image. Specifically, the model is trained so that image fragments from the same source image produce "similar" features, while image fragments from different source images produce "distinct" features. To facilitate this type of training, we employ a contrastive learning approach. The contrastive learning loss is defined as follows.

For image fragments f_i and f_j, a similarity label, $Y_{i,j}$, is defined such that $Y_{i,j} = 1$ if f_i and f_j come from the same source image and $Y_{i,j} = 0$ if f_i and f_j come from different images. For each i, j pair, the contrastive loss is defined as:

$$CL_{i,j} = 0.5 \left[Y_{i,j} D_{i,j}^2 + (1 - Y_{i,j}) max(0, M - D_{i,j})^2 \right] \tag{1}$$

Here, $D_{i,j}$ is the Euclidean distance between the model-generated features for image fragments f_i and f_j. Since each sample has 160 image fragments, the contrastive loss is computed for all image fragment pairs as above and averaged over all pairs to get the aggregate contrastive loss per sample. The training is performed over the provided ImageNet64 dataset using an SGD optimizer, with batches containing multiple samples (which in turn contain image fragments from 10 images).

During inference for each sample, we first generate the features for each image fragment and then apply the K-means clustering algorithm on the model-generated features to predict image fragment groupings. To evaluate the quality of the grouping, we calculate both recall and precision for fragment pairwise relationships. **Recall** measures the fraction of fragment pairs that truly belong to the same source image and were successfully grouped together by the algorithm. **Precision** measures the fraction of predicted fragment pairs that actually belong to the same source image. The recall and precision measures are calculated for each sample and then averaged over all samples in the test set for evaluation.

3 Results

Figure 1 illustrates the training dynamics by showing the decay of mean contrastive loss across training steps for both linear and convolutional models under different output feature dimensions. Figure 2 shows the change in recall and precision metrics, averaged over the test set, after each epoch of training. For equivalent output feature dimensions, the linear models appear to achieve higher recall (but lower precision) compared to their convolutional counterparts, suggesting a trade-off in the models' feature generation characteristics.

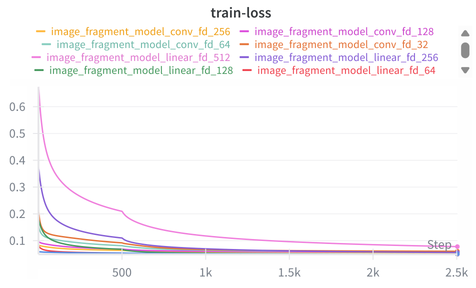

Figure 1: Training loss decay for the different models (linear and convolutional with different output feature dimensions)

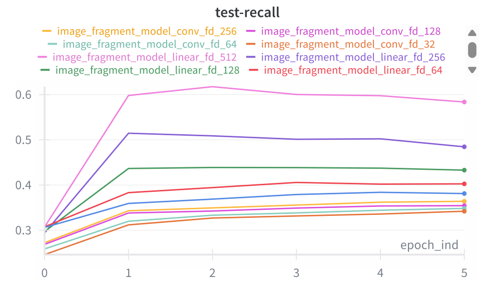

Figure 2: Mean recall for the different models on test set after each training epoch

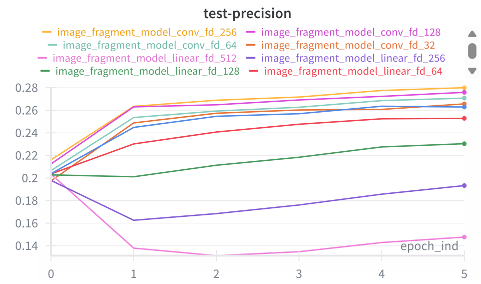

Figure 3: Mean precision for the different models on test set after each training epoch

Beyond quantiative test metrics, we also do a visual assessment of the algorithm's grouping results. Figure 4 shows a visualization of a random sample with 160 unordered image fragments. Figure 5 displays these same fragments, but reorganized based on the cluster assignments generated by the algorithm. As one can see, the neighboring fragments in Figure 5 exhibit strong visual similarity confirming the efficacy of the algorithm in grouping related image fragments.

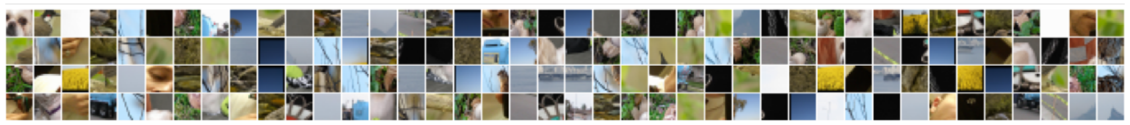

Figure 4: Display of unordered images fragments for a random sample

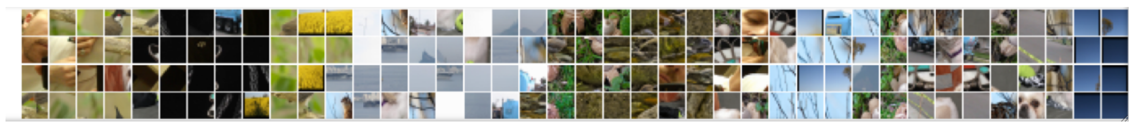

Figure 5: Display of same image fragments reorganized according to cluster assignments

4 Discussion

For a fragment size of 16, the recall and precision metrics for discovering pairwise relationships by the model may appear modest (for e.g. a recall of 43% and precision of 23% for the linear model with output feature dimension of 128). But this is to be expected in the context of the small number of true relationships amongst the universe of all possible relationships. Depending on the application context (for e.g. gene-gene interaction discovery), discovering 43% of previously unknown true relationships could be of real practical value. Moreover, pairwise relationships can be ranked based on similarity metrics or distance metrics thus enabling prioritization of relationships for further discovery or analysis.

Effects of fragment-size: The developed code allows experimentation across different fragment sizes. For brevity, we focus on the linear and convolutional models with 128-dimensional output features, comparing the recall and precision metrics after 5 epochs of training. Table 1 below shows the changes in recall and precision wrt fragment-size for both models. Broadly speaking, recall increases consistently with the fragment size with a relatively minor change in the precision. This is to be expected for two reasons: larger fragments contain more features enabling easier association with the source image, and as the fragment-size increases, the total number of fragments decreases, thereby reducing the total number of possible pairwise relationships.

Fragment-size:	8	16	32
Recall:	37%	43%	50%
Precision:	25%	23%	26%

Fragment-size:	8	16	32
Recall:	NA	35%	41%
Precision:	NA	28%	29%

Table 1: Effects of fragment-size for linear (left) and convolutional (right) models. The results for fragment-size 8 are not available for the convolutional model because the fragment-size is too small to accommodate the filter size.

Effects of random noise and rotations to fragments: We also analyzed how introducing random noise and rotations to image fragments effect model performance. The same augmentor provided with the assignment code was used, but applied to the individual fragments and with varying augmentation levels. Figure 6 shows the change in recall and precisions relative to augmentation level for the 128-dimensional feature models. We observe a significant decrease in performance even at small augmentation levels. One way to mitigate this could be to introduce fragment augmentation during the training process as well to improve model robustness.

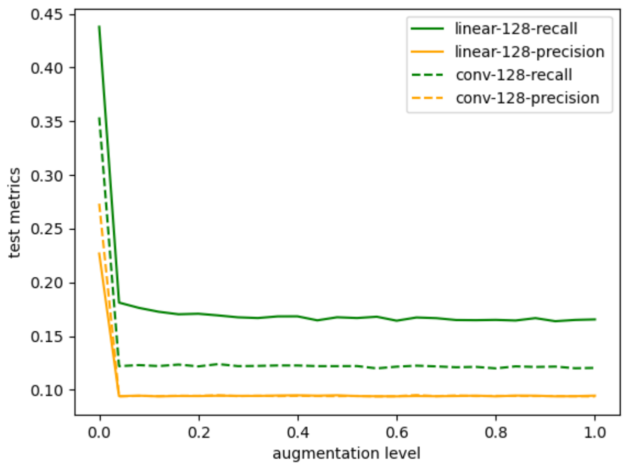

Figure 6: Change in model performance relative to fragment augmentation level

Other possible explorations: One approach worth exploring would be to use an autoencoder model trained on individual fragments. Once such an autoencoder is trained, the features coming out from the encoder component can be used for clustering the image fragments. Another approach would be to use the same training framework described here, but the distance-based contrastive loss could be replaced with a cosine similarity-based loss function (for e.g., see SimCLR). With the current framework, the effects of batch-size, choice of optimizer etc. may also be thoroughly explored.