Where's Waldo? Instance Level Image Retrieval

By: Mikayla Pugel, David Knapp, Ferdous Alam

Abstract:

Instance level retrieval is the process of using a query image to search a database of images that matches the object. This process is often accomplished by extracting features from the images and then using these features in a retrieval process to retrieve images with similar features to the query image (4). Reranking then improves the retrieved images by reranking the images using techniques such as Geometric verification and Reranking Transformers. In this work we improve upon the current best practice of the joint optimization of feature extraction and reranking that uses Reranking Transformers as well as Geometric Verification to rerank the images to improve the accuracy of the images retrieved. We also work to create a prototype process which allows for a user to test their own image on the image retrieval model and to see the top images retrieved for their query image. Lastly, we introduced a new dataset that reapply's the work we completed to see how adaptable this process is to other datasets. (Research in this paper is based off of the work done in this paper https://arxiv.org/pdf/2103.12236.pdf)

Introduction:

Instance level retrieval (ILR) is the process of retrieving a specific instance of an object, not just the object's class. In simpler terms it requires retrieving images often for large databases representing the same object or scene as the one depicted in a query image. Since the relevancy is defined at the instance level, it is more challenging compared to traditional image retrieval methods.

The two primary components of image representations required to solve the problem involve extracting global and local image features. "A global feature summarizes the entire contents of an image, leading to a compact representation but discarding information about spatial arrangement of visual elements that may be characteristic of unique examples. Local features, on the other hand, comprise descriptors and geometry information about specific image regions; they are especially useful to match images depicting the same objects." (11)

Instance retrieval is a fundamental problem in the multimedia field for its various applications. These use cases apply to many markets such as tourism for landmark detection, e-commerce for brand identification, and law enforcement to recreate and/or identify objects/scenes from partial images. Despite the many real-world applications and challenging aspects of the task, ILR has attracted less attention than category-level recognition (CLR) tasks, due to the lack of large-scale datasets with accurate ground truth which is a tedious process.

Technical Approach:

The process is a joint optimized process which means that the extracted features and training of the model are a more merged process, instead of two separate stages. The overall process flow is found in the appendix of this paper in figures 1 and 2. The training process begins with

transformations to the training images. The features are then extracted from the images and the global retrieval model is trained. In order to train the Reranking transformer model, the top 100 neighbors for each query image are needed, therefore the global retrieval model is run to obtain these results so then the RRT model can be trained. For testing the precision of the models, the data is again transformed and features are extracted in the same fashion as before. The top 100-neighbors of each query image are then found using the global retrieval model and these 100-neighbors are reranked by the RRT model and the accuracy results are calculated.

The model specifications and architecture are described in detail in the appendix under the titles model specifications and model architecture, only important features we worked with will be explained here. First, our main focus was on image processing. When training, the images are transformed by randomly cropping them to 224 by 224 and then each image is randomly flipped. For testing, the images are transformed by being resized to 256 by 256 and then cropped at the center. It is also interesting to note that the global model and the RRT model are trained with different convoluted neural networks, one uses stochastic gradient descent and the other uses AdamW.

The model was tested in three different ways which include only reranking with the contrastive loss global model, reranking with the global model and then reranking with an RRT model with a frozen backbone, and lastly reranking with global model and then reranking with an RRT model with a fine tuned backbone. The authors of the paper show that the combination of the contrastive loss global model then reranked with the RRT which is finetuned obtains the best results, which we will be using as our baseline for our experiments.

Model Evaluation:

Our evaluation metric that we were primarily concerned with is recall@k. Since our focus is on improving the reranking, we have 2 options for improving the model. We could either make tweaks to the architecture, or we could make transformations on the images. Since making changes to the architecture would require retraining the entire model, we focused to this point on making transformations to the images.

Image transformation also serves as a measure of feature importance. By observing the change in recall from adjusting certain features, we can gauge how important they are to the model. We experimented by applying a random perspective shift to each image, converting each image to grayscale, and increasing the contrast on each image. We see that transforming the image with the random perspective, grayscale, and adjusting contrast all lower our recall scores.

As we continued experimenting we wanted to see how the cropping and resizing of the images affected the model, the original paper applied a center crop transformation for each test image and a resizing to ensure the images were the same size, as well as applying a horizontal flip to 50% of the images. We removed the crop but kept the resizing which was necessary as the model requires all images to be the same size. From this transformation, we saw that removing the cropping improves the recall. Our next experiment we wanted to try removing the random horizontal flip, however this seemed to have no effect on the results. Next we applied a Gaussian

blur to the images since this transformation smoothes uneven pixel values in an image by cutting out the extreme outliers, but again we saw no real change in the results (1). And lastly we tried normalizing the pixels in the image, which only greatly reduced the recall accuracy. The reason we decided to try and normalize the pixels in the image is that normalizing allows for less non-zero gradients during training, and therefore the neurons in the network will learn faster and that the channel information can be mixed (9). All the results from these experiments are shown in the appendix under Experiment Recall Results.

Testing Other Datasets:

For testing another dataset we have chosen the kaggle products 10k dataset. This dataset has 150,000 photos and 10,000 different products. This dataset contains product categories from categories including Fashion, food, healthcare, and household commodities. And these images have a class id which is the id of the product category, and a group id which is the top group id of a given category (10).

Once the data was formatted in the correct way, using the model to rerank the nearest 100 neighest neighbors worked very well. The recall accuracy values did decently well considering how many more photos and categories there are. If we had more time, we were hoping to be able to examine which categories did better than others, however this will have to wait for future work. The results for this dataset are shown in the appendix under Experiment Recall Results.

Demo:

We were able to operationalize the reranking transformers process. The original paper processed images in batches, but we wanted to show how we could potentially turn this methodology into a product. In order to do this, we saved the features of our test image dataset, as well as two models into pickle files: the original CO model, and the Reranking Transformer. Then we set up a Jupyter Notebook to be able to process any user uploaded image. That image is then put through the CO model to extract its features and start the nearest neighbors process to compare against all images in the test set. Once the 100 nearest neighbors are found, the test image and the neighbors are put through the reranking transformer to rerank the images. A video demo can be seen in the github for this project.

Summary:

Overall, our team learned a lot about computer vision, specifically with image transformation and extracting descriptors. We tested new image transformations to improve the current image recognition process. These transformations would impact the descriptors that were extracted and we wanted to see how modifying the images, and therefore modifying the descriptors, would impact the image retrieval process. What we found was that most transformations did not impact the model's accuracy significantly, which we expected as local descriptors are robust to change in how the image is translated (2). However, we did see an increase in the model's accuracy when we removed the image crop that the original process used. This is because global descriptors are not robust, and since we change the overall image, we had significant changes to

the global descriptor, which did marginally improve the model's accuracy. Lastly, we created the prototype process explained in the demo section as well as testing a new dataset to show the applications of this research and how it can be reapplied to different datasets.

Appendix:

Technical Process:

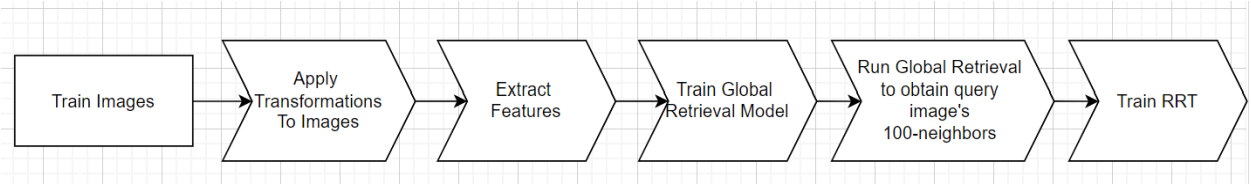

Fig. 1

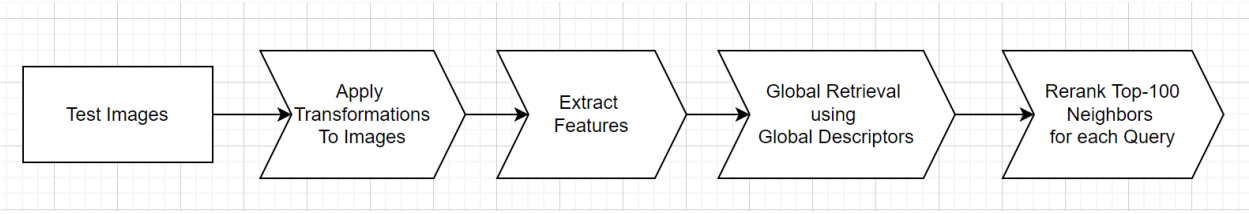

Fig. 2

Model Specification:

Image Processing During Training:
- Each image randomly cropped to 224 by 224
- Each image is randomly flipped

Image Processing During Testing:
- Each image is resized to 256 by 256 then cropped at the center to 224 by 224

For Feature Extraction:
- Using ResNet50 features are extracted and which creates 49 (7 by 7) local descriptors for each image
- The global descriptors are obtained by spatially averaging the local responses
- Both the global and local descriptors are linearly projected to a dimension of 128

Model Architecture:

Global Model:
- trained with contrastive loss
- Batch size = 800
- Test batch size = 800
- Only one image scale is used so no scale embedding is used
- trained using SGD with Nesterov momentum for 100 epochs
- learning rate of 0.001
- weight decay of 0.0004

- momentum of 0.9

RRT Model:

- trained on top of the global model
- using both freezing and finetuning CNN backbone
- trained with AdamW for 100 epochs
- learning rate of 0.0001

Image Query and Retrieval Examples:

Supplementary Information:

Terms Defined: (8)

Finetuning - Implements transfer learning in that the model will not only train its current stage, but finetune all the previous stages
Frozen - Freezes all previous stages during the training of the current stage
Weight Decay - regularization technique by adding a small penalty to the loss function, this is normally used to prevent overfitting and avoids exploding gradient
Learning Rate - controls how much we are adjusting the weights of our network with respect to the loss gradient. The lower the value, the slower we travel along the downward slope
Momentum - accelerates gradient descent in the relevant direction and dampens oscillations
Nesterov momentum - An extension of momentum that involves calculating the the decaying moving average of the gradients of projects positions in the search space rather than the actual positions themselves. This allows the search to slow while approaching the minima.
Contrastive Loss - takes the output of the network for a positive example and calculates its distance to an example of the same class and contrasts that with the distance to negative example.

CNN Explained: (5)

- The first layer in a CNN is always a convolutional layer
- Next CNN contains pooling layers which simplify the data by reducing its dimensionality (shortening training time and helps curb the problem of overfitting)
- The last layers are the fully connected layers, in which every neuron in the first is connected to every neuron in the next. This process looks at which features most accurately describe the specific classes which results in a simple vector of probabilities

Adam W Explained: (6)

- AdamW is a stochastic optimization method that modifies the typical implementation of weight decay in Adam by decoupling weight decay from the gradient update.

Stochastic Gradient Descent Explained: (7)

- Samples are selected randomly instead of the whole data set for each iteration
- Batch denotes the total number of samples from a dataset that is used for calculating the gradient for each iteration

- Since only one sample from the dataset is chosen at random for each iteration, the path taken by the algorithm to reach the minima is usually noisier than normal GD
- It may also require more iterations than normal GD, but computationally it is much less expensive

Experiment Recall Results:

Method	R@1	R@10	R@100
Baseline (CO)	*80.7*	*91.9*	*96.9*
Baseline with Reranking (frozen)	*81.8*	*92.4*	*96.9*
Baseline with Reranking (finetuned)	*84.5*	*93.2*	*96.6*
Reranking with RandomPerspective()	69.3	87.8	96.6
Reranking with grayscale transform	67.2	87.7	96.6
Reranking with contrast adjustment	71.0	88.2	96.6
Reranking removing cropping and resizing	86.3	94.2	97.1
Reranking removing random horizontal flip	84.44	93.25	96.7
Reranking with Gaussian Blur	84.4	93.21	96.7
Reranking with Normalization	12.83	32.33	65.97
10K Products Dataset	34.73	54.07	73.42

References:

1. https://www.adobe.com/creativecloud/photography/discover/gaussian-blur.html
2. https://cvexplained.wordpress.com/2020/07/20/10-1-what-are-image-descriptors-feature-descriptors-and-feature-vectors/
3. https://towardsdatascience.com/instance-level-recognition-6afa229e2151

4. https://filebox.ece.vt.edu/~jbhuang/teaching/ece6554/sp17/lectures/Lecture_02_Instance_Recognition.pdf
5. https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939
6. https://paperswithcode.com/method/adamw
7. https://towardsdatascience.com/stochastic-gradient-descent-clearly-explained-53d239905d31
8. https://towardsdatascience.com/the-beginners-glossary-of-neural-network-terms-a9617354078
9. https://inside-machinelearning.com/en/why-and-how-to-normalize-data-object-detection-on-image-in-pytorch-part-1/
10. https://www.kaggle.com/c/products-10k
11. https://ai.googleblog.com/2020/09/advancing-instance-level-recognition.html