Visual Product Recognition

Professional

Customer

Problem Statement

How can we improve image retrieval by bridging the gap between professional and consumer-captured images across real world domains?

Potential Applications

- *E-commerce & Online Market Places*
  - Users upload a photo, and the system finds visually similar products.
- *Surveillance and Security*
  - Find instances of a person or vehicle across multiple camera feeds.
- *Healthcare & Medical Imaging*
  - Retrieve/Organise similar X-rays, MRIs, or pathology slides for diagnosis aid.

Potential Impact

- Ranked *list of similar images*
- *Class Prediction*
- *Image Clustering*
- *Dataset Label Propagation*

Literature Review

Paper 1

Gordo et al. (2017) present an end-to-end deep learning framework for image retrieval, addressing three key challenges:

Challenges

- Noisy training data in landmark datasets
- Suboptimal architectures for retrieval tasks
- Ineffective training procedures

Proposed Solutions

- Automatic dataset cleaning to refine training labels
- Differentiable R-MAC architecture for region-based feature aggregation
- Triplet-loss Siamese network optimized for retrieval embeddings

Performance

Dataset-mAPScore

Dataset	mAP Score
Oxford 5k	94.7%
Paris 6k	96.6%
Holidays	94.8%
(Best results achieved with large input images)	

Advantage Over Traditional Methods

- Outperforms local descriptor indexing (SIFT/SURF)
- Exceeds spatial verification approaches
- More efficient than binary hashing techniques (LSH)

Source: Gordo, A., Almazán, J., Revaud, J., & Larlus, D. (2017). End-to-End learning of deep visual representations for image retrieval. International Journal of Computer Vision, 124(2), 237–254. https://doi.org/10.1007/s11263-017-1016-8

Paper 2

Chum et al. (2007) propose an object retrieval system focused on maximizing recall through visual query expansion:

Challenges

- Precision-recall tradeoff in large databases
- Visual word instability from feature quantization
- Missed matches due to viewpoint/lighting variations

Innovations

- Spatial verification: Geometric consistency checks suppress false positives
- Generative feature model: Learns latent feature distributions from verified results

Performance

Dataset-Scale-Outcome

Dataset	Scale	Outcome
Oxford Buildings	5,000 images	Achievedtotal recall
Flickr	1M+ images	Significant precision boost

Advantage Over Alternatives

- Outperforms standard text-inspired query expansion
- More robust than pure CNN-based feature matching (2007 context)
- Avoids computational overhead of graph-based retrieval

Source: Chum, O., Philbin, J., Sivic, J., Isard, M., & Zisserman, A. (2007). Total Recall: Automatic Query Expansion with a Generative Feature Model for Object Retrieval. 2007 IEEE 11th International Conference on Computer Vision. doi:10.1109/iccv.2007.4408891

Paper 3

Conde et al. (2022) present a CLIP-based ViT solution for open-world image retrieval in the Google Universal Image Embedding Challenge:

Challenges

- Multi-domain adaptation (landmarks, food, artwork)
- Limited labeled data for specialized domains
- Embedding generalization across object types

Innovations

- CLIP-ViT fine-tuning: Leverage contrastive pre-training for zero-shot transfer
- PCA compression: Reduce embeddings from 1024D → 64D while preserving info
- Strategic sampling: 9,691 classes with ≥4 images, capped at 50/class

Performance

Metric-Score-CompetitionRank

Metric	Score	Competition Rank
Mean Precision@5	0.688	4th/1,022 teams

Advantage Over Alternatives

Method-Strength-Limitation

Method	Strength	Limitation
CNN (ResNet)	Domain-specific tuning	Limited cross-domain transfer
Swin Transformer	Hierarchical features	Computational overhead
SimCLR/BYOL	Self-supervised	Requires custom pretraining

Source: Conde, M. V., Aerlic, I., & Jégou, S. (2022). General image descriptors for open world image retrieval using ViT CLIP. arXiv preprint arXiv:2210.11141.

Paper 4

Bai et al (2020) Products-10K: A Large-scale Product Recognition Dataset

Challenges

- Nearly 10,000 visually similar SKUs, hard to distinguish
- High variation in customer images (background, lighting, angle)
- Imbalanced sample counts across categories

Innovations

- Human-verified, SKU-level labels (noise <0.5%)
- Mix of in-shop and real-world customer photos
- Advanced training tricks: high-res images, balanced finetuning, metric-guided loss

Performance

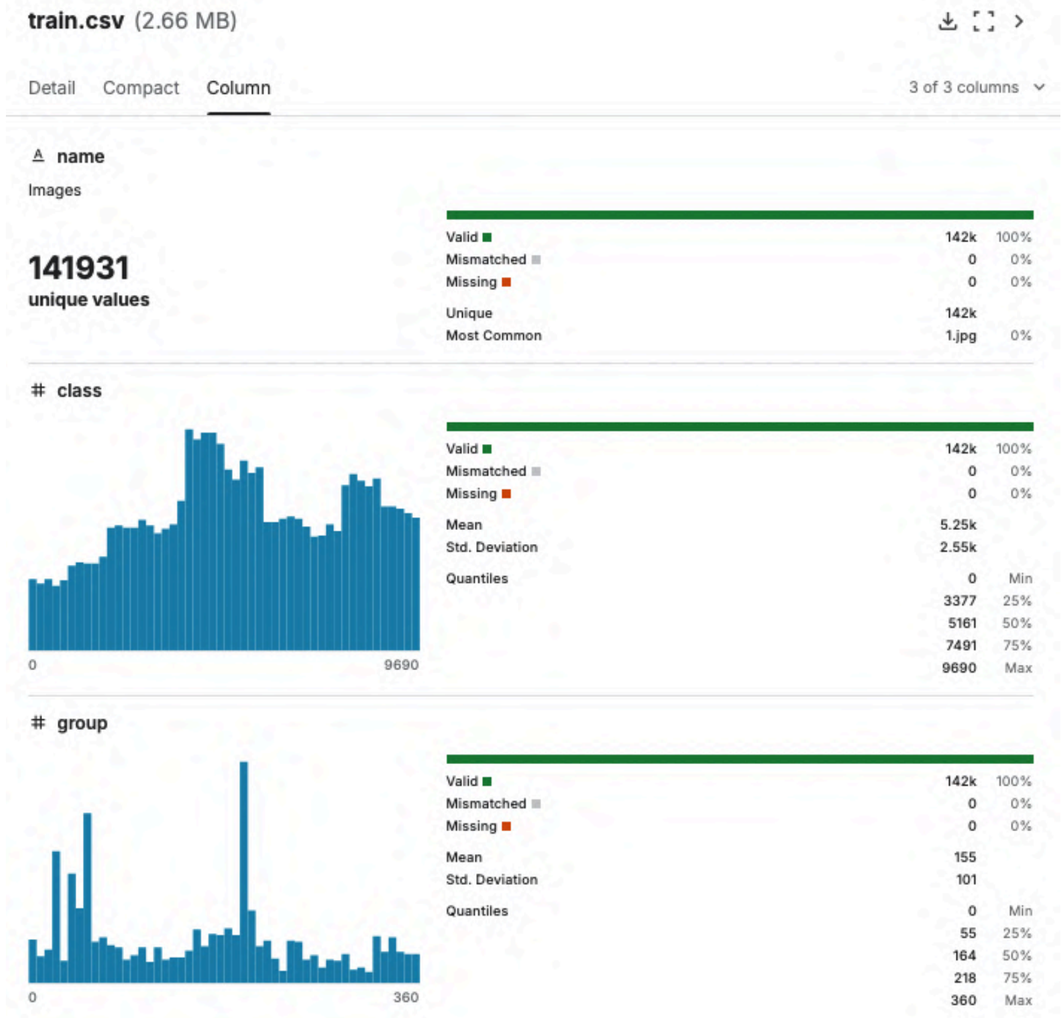
Dataset	Scale	Outcome
Products-10K	150,000 images, 10,000 SKUs	Enables benchmarking of fine-grained recognition; new training tricks yield significant accuracy improvements[1]

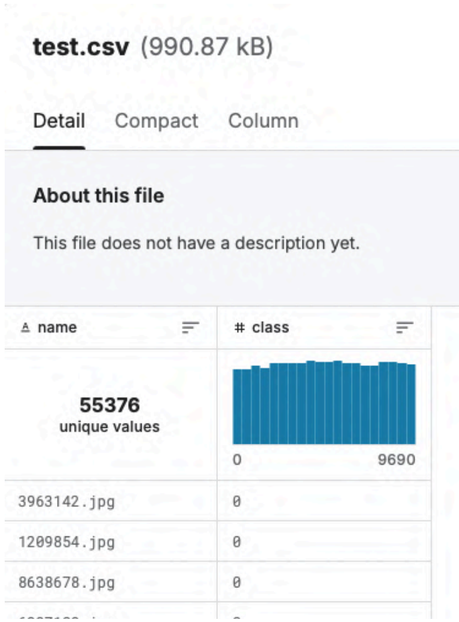
Advantage Over Alternatives

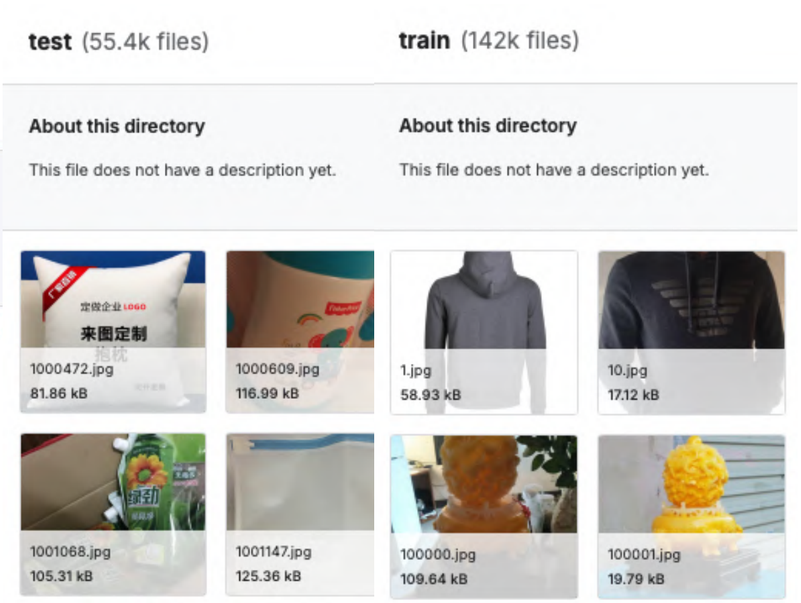
- Largest publicly available fine-grained product dataset at SKU level, surpassing previous datasets in both scale and diversity.
- Realistic evaluation: Includes challenging customer images, unlike many datasets that only provide clean, studio images.
- Supports research on practical, real-world product recognition systems, bridging the gap between academic benchmarks and commercial needs.

Features Preprocessing

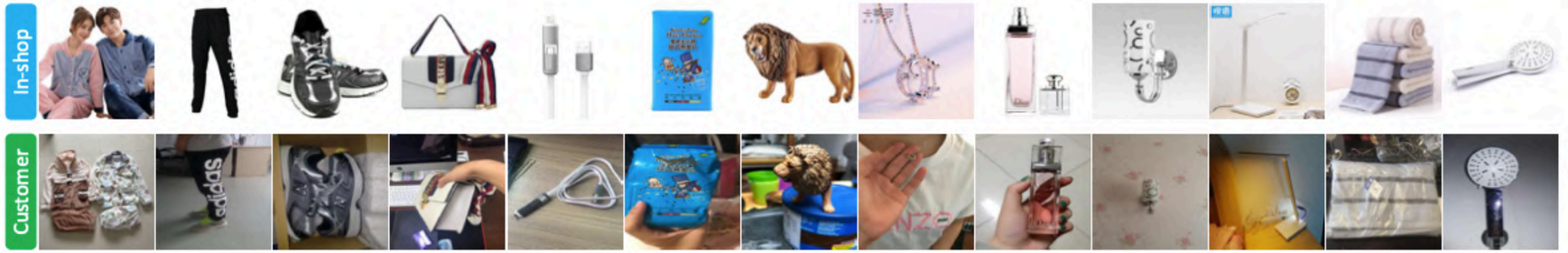
Dataset



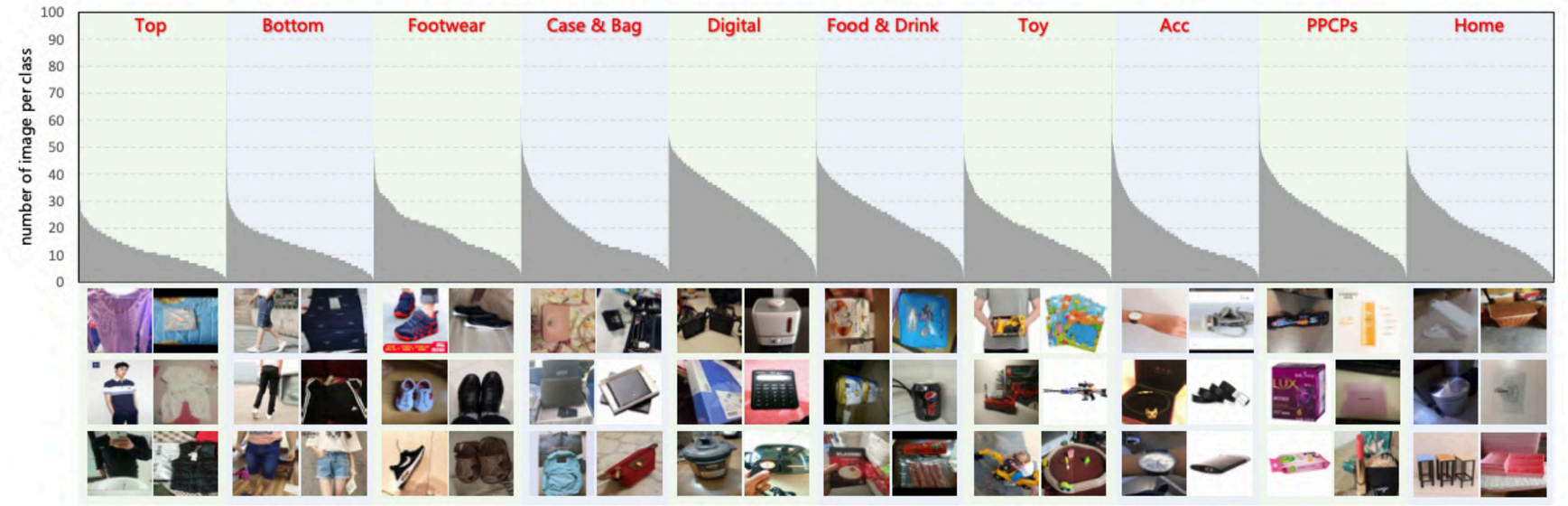




- **Open Source** - Kaggle
- **Large-Scale**: ~150K images from 10K SKUs on JD.com across diverse product categories.
- **Label Graph**: Includes a detailed product label graph with hierarchical relationships.
- **Realistic Data**: Mix of pro and customer images with imbalanced distribution.
- **Clean Labels**: Manually verified.
- **Public Access**: Available for non-commercial research and education use.



Images in Product-10K are collected from in-shop photos (the first row) and customer images (the second row). The SKU label of images in each column is same.

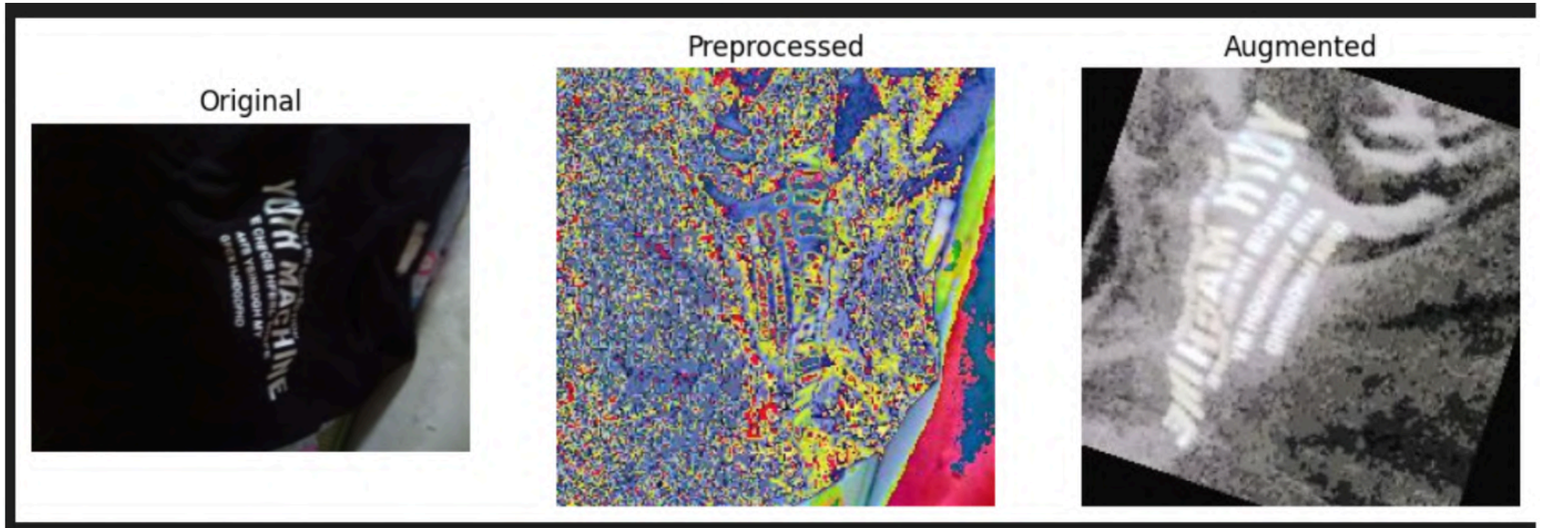


Examples of images in Products-10K dataset. Both of the clean in-shop photos and realistic customer images are collected.

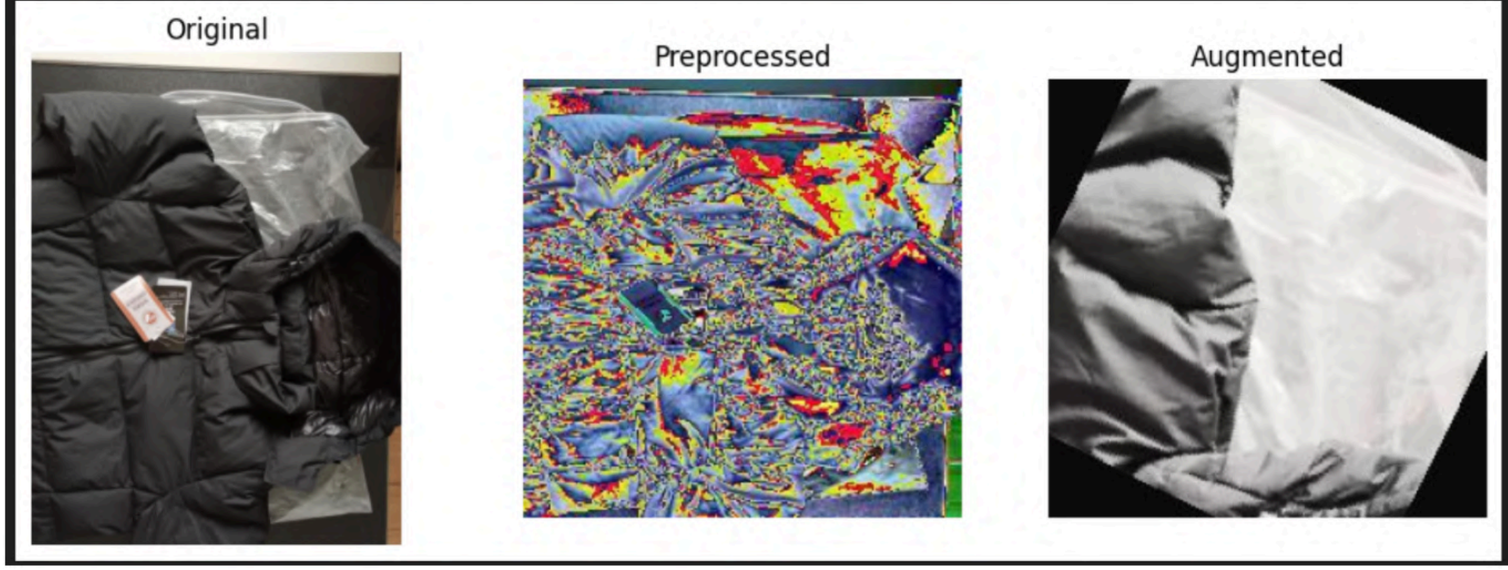
Source: Yalong Bai, Yuxiang Chen, Wei Yu, Linfang Wang, Wei Zhang. "Products-10K: A Large-scale Product Recognition Dataset".

First Steps of Feature Preprocessing

- **transforms.Resize((224, 224))**
  - Resizes all images to 224x224 pixels.
  - This is necessary because most pre-trained models (like ResNet, ViT, and EfficientNet) expect a fixed input size.
  - Ensures uniformity across the dataset, preventing shape mismatches.
- **transforms.ToTensor()**
  - Converts the image from a PIL image (or NumPy array) to a PyTorch tensor.
  - Normalizes pixel values from 0-255 to 0-1 by dividing by 255.
  - Makes the data compatible with PyTorch's tensor-based computations.
- **transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])**
  - Normalizes each RGB channel using the mean and standard deviation values from the ImageNet dataset.
  - Helps the model generalize better by centering pixel values around zero mean and unit variance.
  - Essential when using a pre-trained model like ResNet, as it was trained with this normalization.



🔹 Showing images for: 639.jpg



Augmentation

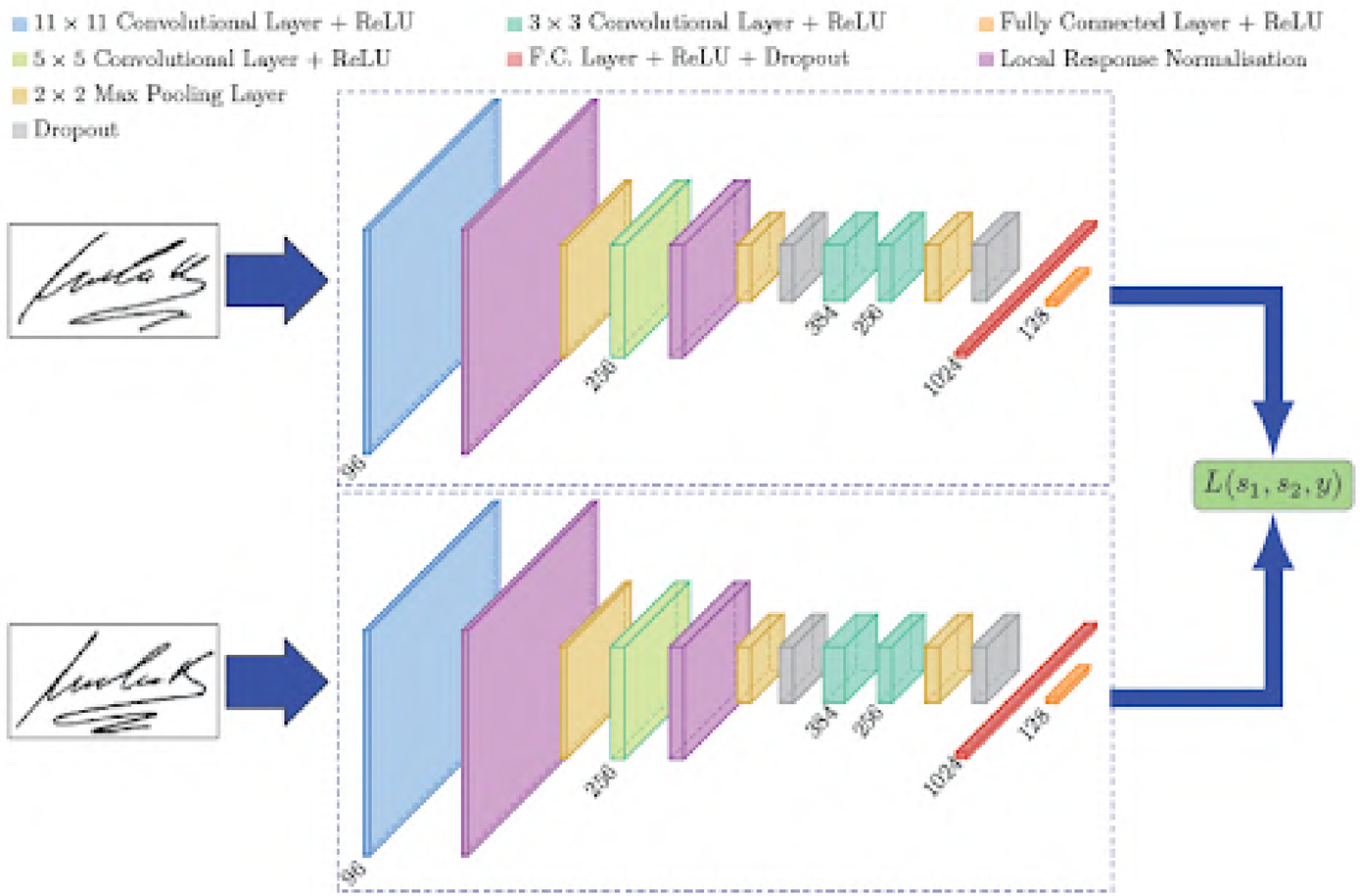
- **1. transforms.Resize((256, 256))**
  - Resizes the image to 256×256 pixels before further transformations.
  - Ensures a consistent input size for cropping operations.
  - Helps handle images of different sizes.
- **2. transforms.RandomResizedCrop(224)**
  - Randomly crops a 224×224 region from the resized image.
  - Varies the scale and aspect ratio to introduce variation.
  - Prevents overfitting by exposing the model to different image regions.
- **3. transforms.RandomRotation(50)**
  - Rotates the image randomly by ±50 degrees.
  - Helps the model become rotation-invariant, useful when objects may appear at different angles.
  - Too large a rotation (>50°) might make some objects unrecognizable.
- **4. transforms.RandomHorizontalFlip()**
  - Flips the image horizontally with a 50% probability.
  - Useful for object categories that don't have a strict left-right orientation (e.g., animals, landscapes).
  - Not recommended for text-based images.
- **5. transforms.ColorJitter(brightness=0.1, contrast=0.1, saturation=0.1, hue=0.1)**
  - Randomly adjusts brightness, contrast, saturation, and hue to mimic lighting variations.
  - Prevents the model from relying too much on fixed color distributions.
  - Small values (0.1) ensure that distortions are not too extreme.
- **6. transforms.ToTensor()**
  - Converts the PIL image into a PyTorch tensor (shape: [C, H, W]).
  - Normalizes pixel values from [0, 255] to [0, 1] for deep learning models.

Why is this augmentation important?

- Improves model generalisation by simulating real-world variations.
- Reduces overfitting by ensuring the model doesn’t memorize specific training images.
- Boosts robustness to changes in viewpoint, illumination, and object transformations.

ML Methodology

Siamese Network



Siamese Network for Fine-Grained Product Recognition

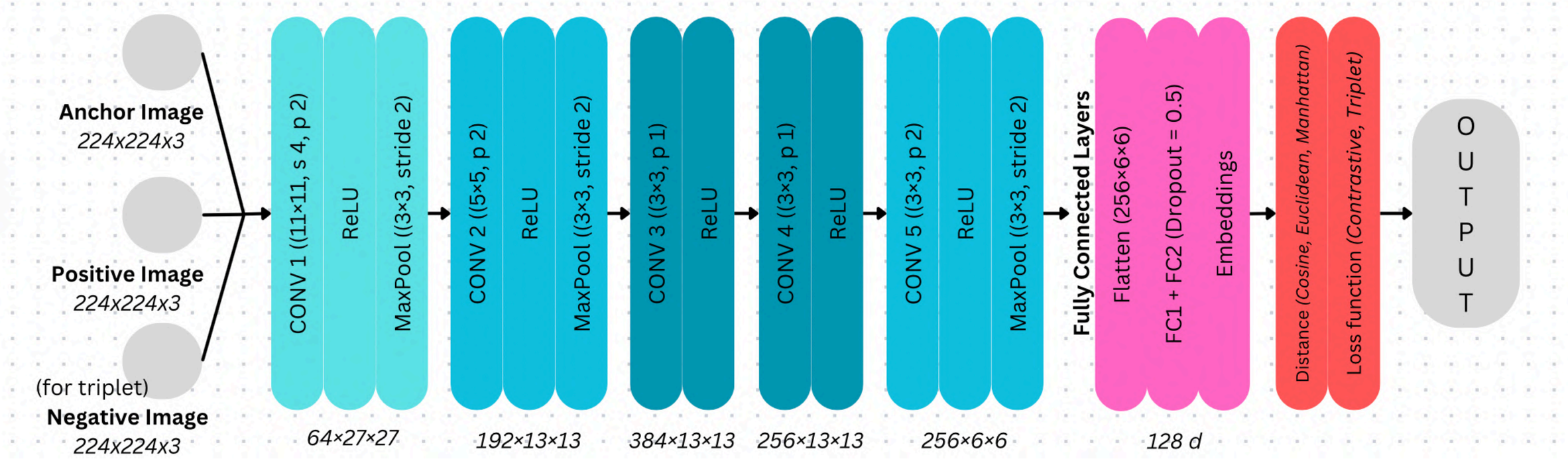
- Learns similarity between image pairs — ideal for class imbalance & unseen products
- Two identical CNNs extract embeddings, compared via Euclidean / Manhattan / Cosine distance
- Trained with contrastive and triplet loss for discriminative embeddings

Boosted with Pretrained Backbones

- Uses ResNet50 / EfficientNet-B0 pretrained on ImageNet
- Faster convergence, better performance on limited data
- Supports catalog updates without full retraining

Source: https://builtin.com/machine-learning/siamese-network

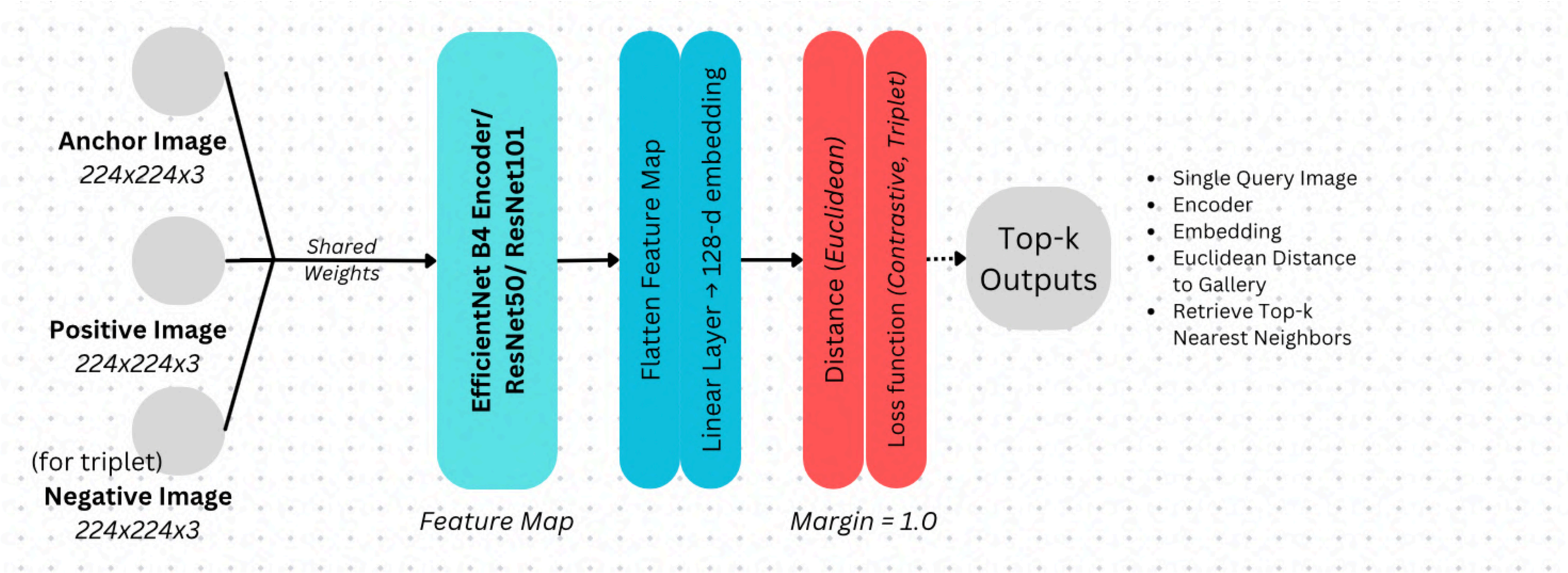
From Scratch



Training Setup:

- Batch size: 32
- Optimizer: Adam
- LR: 0.001
- Loss: Contrastive or Triplet
- Distance: Cosine / Euclidean / Manhattan
- Total Params: ~90M
- Embedding Dim: 128
- Normalization:
  - Mean [0.485, 0.456, 0.406]
  - Std [0.229, 0.224, 0.225]

Pre-trained



Training Setup:

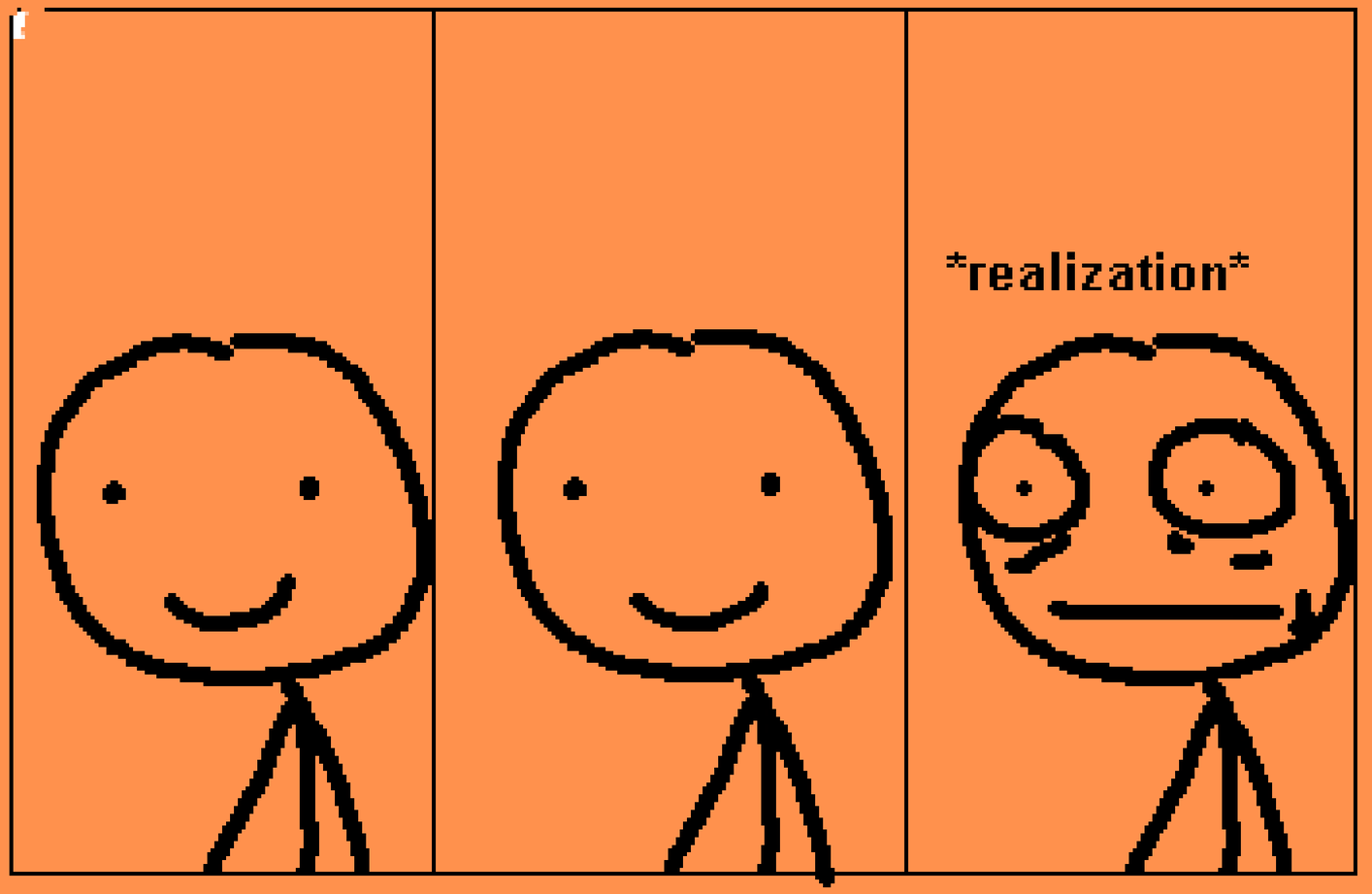
- Batch Size: 32
- LR: 0.001
- Optimizer: Adam
- Epochs: 10
- Augmentations: Horizontal Flip, Rotation, Color Jitter

Results

Distance Metrics	Loss Function	MAP@5 (10 epoch)
Cosine	Contrastive	0.069
Euclidean	Contrastive	0.070
Manhattan	Contrastive	0.068
Manhattan	Triplet	0.033
Euclidean	Triplet	0.027
Cosine	Triplet	0.023

Pre-trained Model (only with Euclidean Contrastive)	Parameters	MAP@5 Score (10 epoch)
ResNet-101 (from torchvision)	44 Million	0.1225
ResNet-50 (from torchvision)	25.6 Million	0.1163
EfficientNet-B4 (from timm)	19 Million	0.1842

Output



SUBSET FINE-TUNING!

What is subset finetuning?
Why is it important?

Fine-tuning on a small but relevant subset of the training data can lead to better generalization, faster training, and superior image retrieval performance, compared to naive fine-tuning on the entire training set.

Why is it Important?

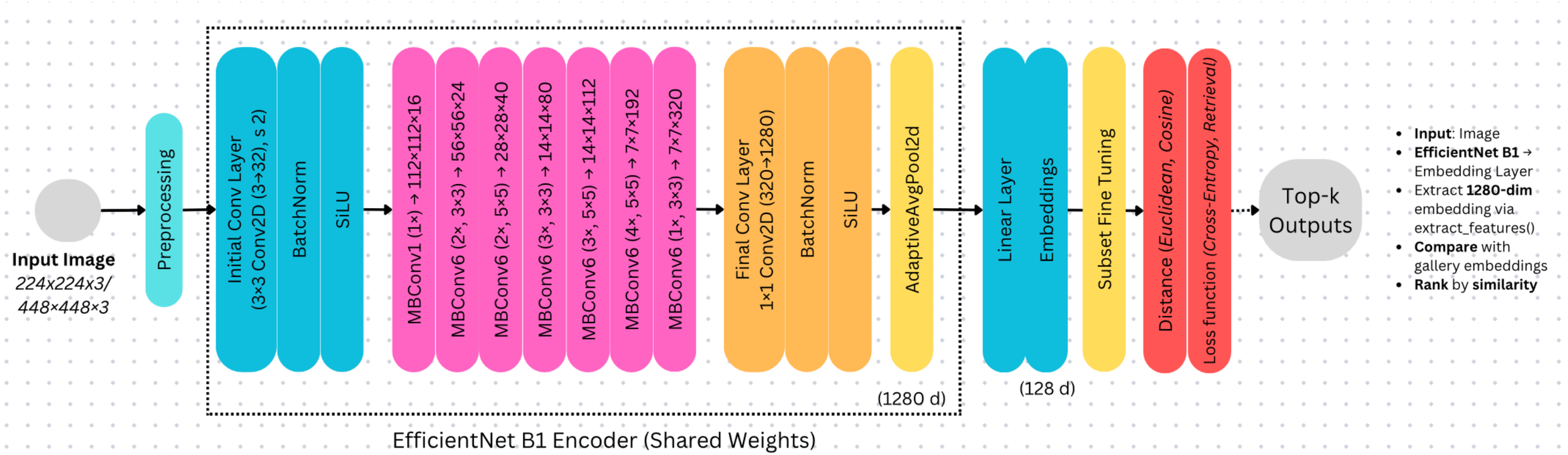
1. **Reduces Overfitting**: Fine-tuning on the entire dataset may lead to overfitting, especially when many classes are not relevant to the test set. Subset fine-tuning focuses only on relevant examples, helping the model generalize better.
2. **Efficiency**: Training on a smaller, carefully chosen subset significantly reduces computational cost and time without hurting performance—in some cases, it even improves it.
3. **Improved Performance**: The paper shows that subset fine-tuning leads to better results on image retrieval benchmarks like Revisited Oxford and Paris datasets, achieving state-of-the-art performance.

How is it Relevant for Image Retrieval?

In image retrieval, the goal is to find images in a gallery that are visually similar to a given query image. Fine-tuning a model on a carefully selected subset of training images that are visually and semantically similar to the expected queries:

- Enhances Discrimination
- Boosts Retrieval Accuracy
- Domain Adaptation

Pre-trained

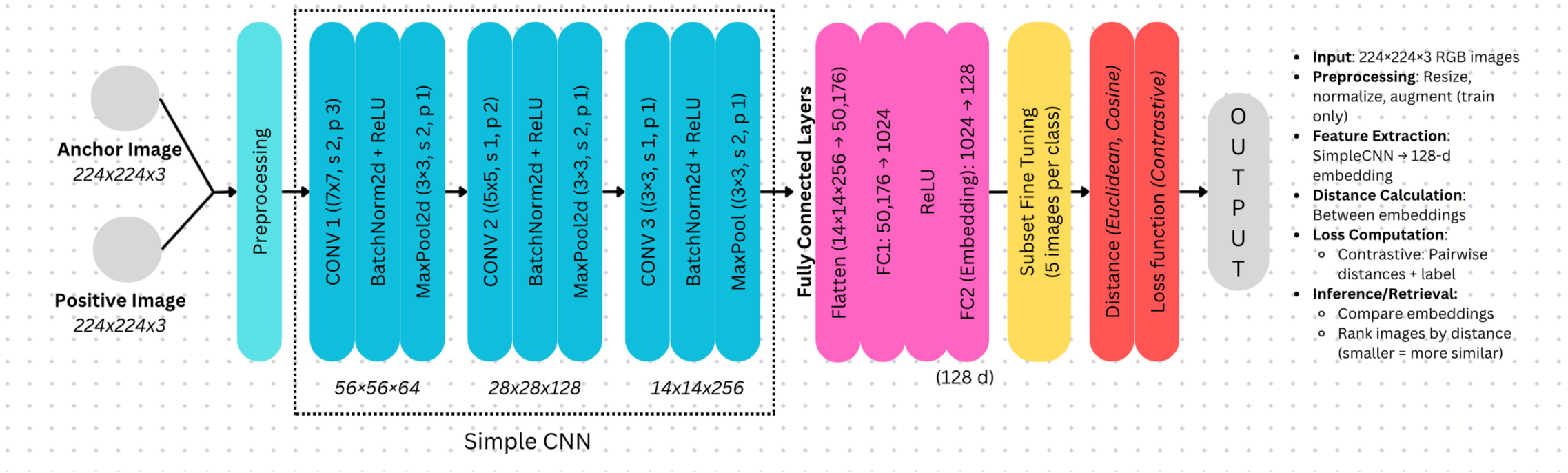


Training Setup:

- Batch Size: 16
- Learning Rate: 3e-4 → 3e-5 for fine-tuning
- Optimizer: Adam (weight decay = 1e-4)
- Epochs: 10
- Scheduler: MultiStepLR (milestones: 2, 4, 6)

Pre-trained with EfficientNet-B1 (cosine)	**mAP@5** 0.6366

From Scratch



Training Setup:

- Batch Size: 16
- Learning Rate: 0.001 (initial), 0.0001 (fine-tuning)
- Optimizer: Adam
- Epochs: 1 (initial), 3 (default fine-tune)
- Loss: Contrastive (default)
- Distance: Euclidean, Cosine

From Scratch (Cosine + Contrastive)	**mAP@5** 0.9704

looking for coffee?

Query Image
Class: 5136, Group: -1

Rank 1: Same Class
Group: 152
Similarity: 0.6524

Rank 2: Same Class
Group: 152
Similarity: 0.6436

Rank 3: Different Class
Group: 277
Similarity: 0.6395

Rank 4: Same Class
Group: 152
Similarity: 0.6275

Rank 5: Same Class
Group: 152
Similarity: 0.6248

and some ramen.

Query Image
Class: 5155, Group: -1

Rank 1: Different Class
Group: 174
Similarity: 0.7817

Rank 2: Same Class
Group: 174
Similarity: 0.7678

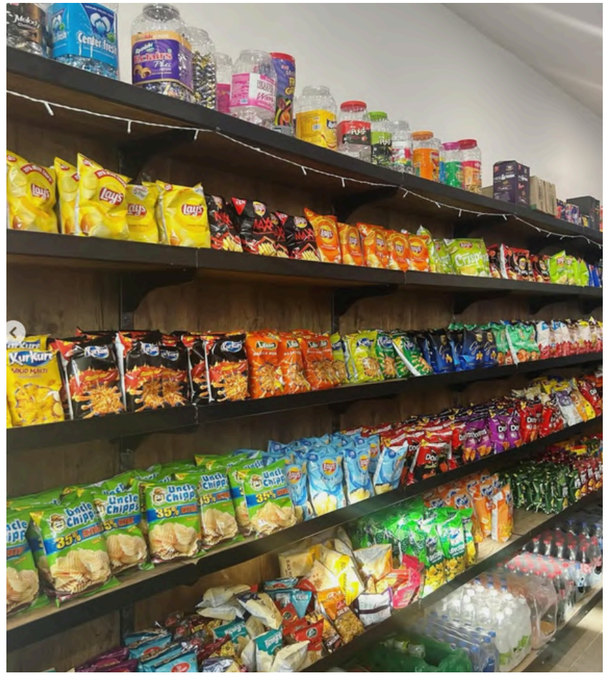
Rank 3: Different Class
Group: 170
Similarity: 0.7558

Rank 4: Different Class
Group: 174
Similarity: 0.7527

Rank 5: Different Class
Group: 170
Similarity: 0.7373

perfect for all nighters!

toh, ab, aage kya?



TUCK
Aadam's
SHOP

Challenges that can be faced in deploying

- Models are extremely computationally expensive to train.
- Requires regular updating of image gallery and retraining fine-tuned models if used.
- Products with minor visual differences (e.g., same brand, different flavors) may confuse the system.
- Manually Updating the dataset is difficult and heavy.

Performance Metrics

Performance is evaluated using **mean Average Precision** at top-5 images **(mAP@5)**.

The equation for the mean average precision (mAP) of a set of queries is:

$$MAP = \frac{\sum_{q=1}^{Q} AveP(q)}{Q}$$

Mean average precision equation. | Image: Ren Jie Tan

Where Q is the number of queries in the set and AveP(q) is the average precision (AP) for a given query, q.

What the formula is essentially telling us is that, for a given query, q, we calculate its corresponding AP, and then the mean of all these AP scores would give us a single number, called the mAP. This quantifies how good our model is at performing the query.

https://builtin.com/articles/mean-average-precision

Recall & Precision

$$\text{Recall (TPR)} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Recall formula of a given class in classification. | Image: Ren Jie Tan

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Precision formula of a given class in classification. | Image: Ren Jie Tan

We could suspect that for a given classification model, there lies a trade-off between its precision and recall performance.

Precision in Information Retrieval

Precision represents the ratio of relevant documents the model retrieves based on a user's query over the total number of retrieved documents.

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

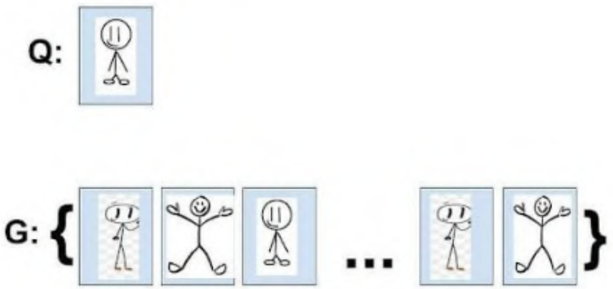
Precision formula for information retrieval. | Image: Ren Jie Tan

We can have a formula where the model is only assessed by considering its top-most queries. The measure is called precision at k or P@K.

How is this relevant to image retrieval?

We shall define the following variables:

- **Q** is the user query.
- **G** is a set of labeled data in the database.
- **d(i,j)** is the score function to show how similar object **i** is to **j**.
- **G'** which an ordered set of **G** according to score function **d(,)**.
- **k** to be the index of **G'**.



User querying G with a document Q. | Image: Ren Jie Tan

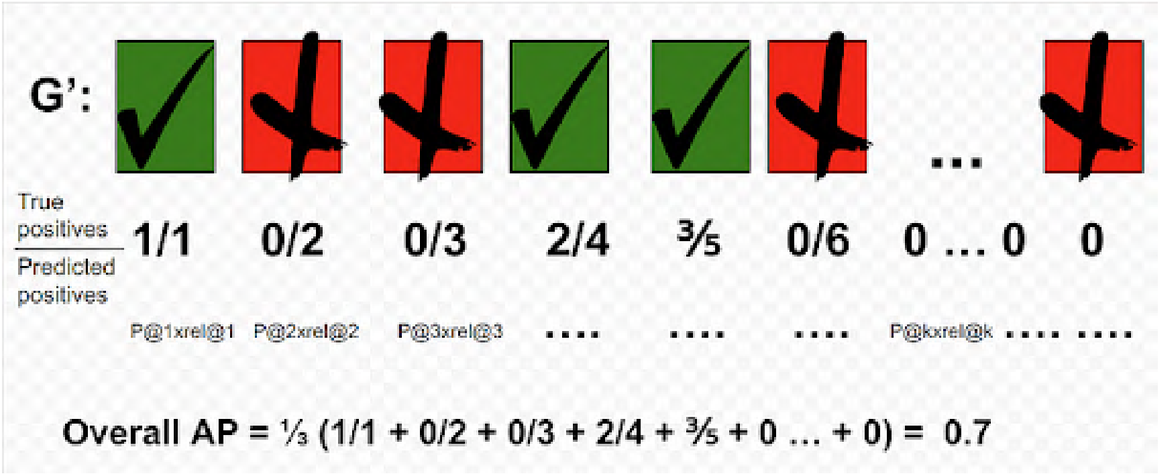
After calculating the d(,) for each of the documents with Q, we can sort G and get G'.

Say the model returns the following G':

Model returned sorted query results G'. | Image: Ren Jie Tan

Using the Precision formula above, we get the following:

- P@1 = 1/1 = 1
- P@2 = 1/2 = 0.5
- P@3 = 1/3 = 0.33
- P@4 = 2/4 = 0.5
- P@5 = 3/5 = 0.6
- P@n = 3/n



Calculating Mean Average Precision

For each query, Q, we can calculate a corresponding AP. A user can have as many queries as they like against this labeled database. The mAP is simply the mean of all the queries that the user made.

$$\text{mAP} = \frac{1}{N}\sum_{i=1}^{N}\text{AP}_i$$

Mean Average Precision formula for information retrieval. | Image: Ren Jie Tan

best mAP@5 achieved

Model Type	mAP@5	mAP@5
From Scratch	0.070	0.9704
Pre-trained with EfficientNet	0.184	0.6366

EfficientB1 with 7.8 million parameters.

aur ab

Bibliography

- Gordo, A., Almazán, J., Revaud, J., & Larlus, D. (2017). End-to-End learning of deep visual representations for image retrieval. International Journal of Computer Vision, 124(2), 237–254. https://doi.org/10.1007/s11263-017-1016-8
- Chum, O., Philbin, J., Sivic, J., Isard, M., & Zisserman, A. (2007). Total Recall: Automatic Query Expansion with a Generative Feature Model for Object Retrieval. 2007 IEEE 11th International Conference on Computer Vision. doi:10.1109/iccv.2007.4408891
- Conde, M. V., Aerlic, I., & Jégou, S. (2022). General image descriptors for open world image retrieval using ViT CLIP. arXiv preprint arXiv:2210.11141.
- Yalong Bai, Yuxiang Chen, Wei Yu, Linfang Wang, Wei Zhang. "Products-10K: A Large-scale Product Recognition Dataset".
- https://builtin.com/articles/mean-average-precision

thank you.