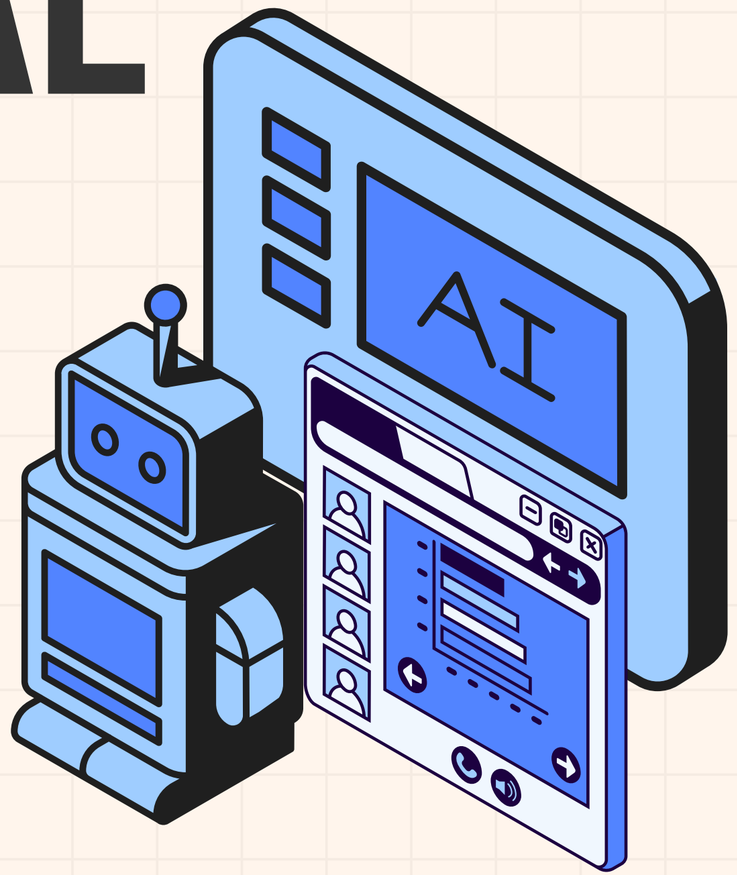
IMAGE RETRIEVAL
FACE ATTRIBUTES

Deep Learning project

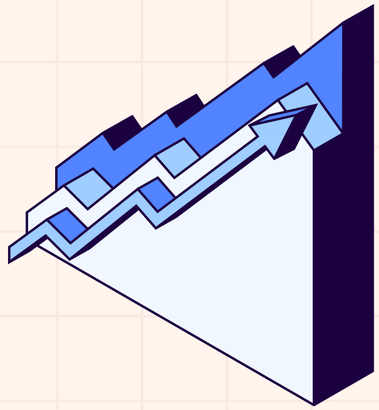


Giuseppe Marchesani - Giovanni Castoldi

PROJECT OBJECTIVE

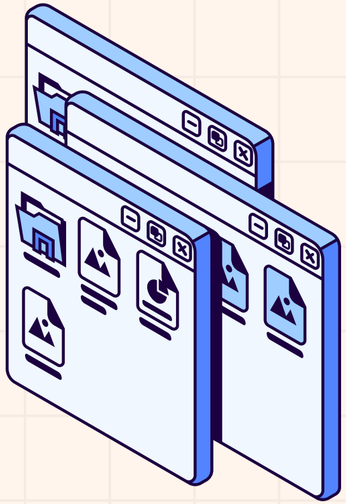
Goal

Build a system that can retrieve face images based on a set of semantic attributes (e.g., “Smiling”, “Male”, “Blond Hair”).



Core idea:

Map images and attribute vectors into a shared embedding space,
then retrieve images by cosine similarity.



BACKGROUND

Why This Problem Is Important

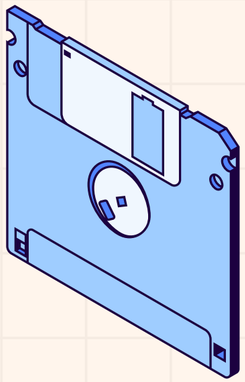
- Attribute-based retrieval allows natural, human-interpretable search.
- It is more flexible than classification: because the system must learn a continuous geometry of similarity, not just assign labels.

Approach

This project applies Deep Metric Learning, where the objective is to:

- pull matching pairs (image + correct attributes) closer in embedding space;
- push non-matching (negative) pairs further apart.

DATA DESCRIPTION AND PREPROCESSING

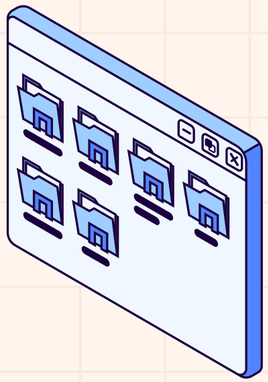


DATASET: CELEBA

- ~200,000 celebrity **face images**
- 40 binary **facial attributes** per image

Examples of attributes:

- Smiling
- Male
- Young
- Blond_Hair
- Wavy_Hair
- Heavy_Makeup
- Mustache
- Eyeglasses



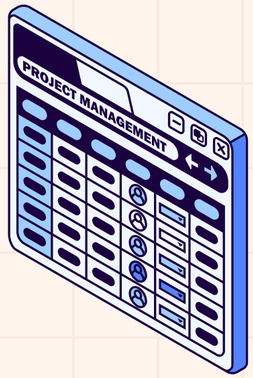


IMAGE PREPROCESSING

Images are **preprocessed using standard ImageNet transformations:** Resize, Crop, Normalize.

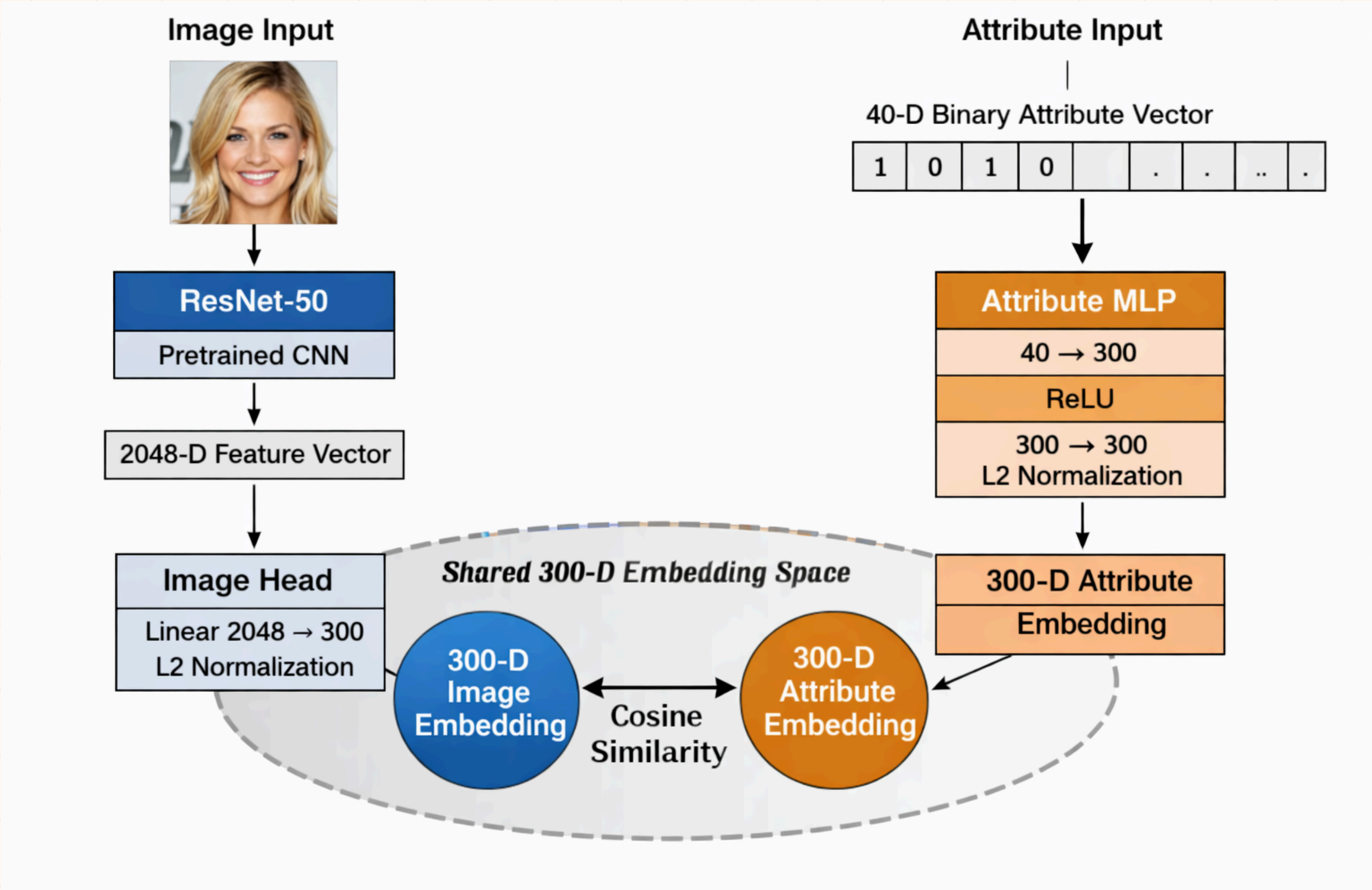
A pretrained **ResNet-50** (ImageNet) is **used as a frozen feature extractor.**
These features remain fixed throughout training (no fine-tuning).

ATTRIBUTE PREPROCESSING

CelebA encodes **binary attributes** as -1 / +1.
We **converted them to 0 / 1** for consistency with neural networks and metric learning.

We also **generated hard negatives** by flipping 1 to 3 bits in the attribute vector.

NEURAL NETWORK ARCHITECTURE



MODEL TRAINING

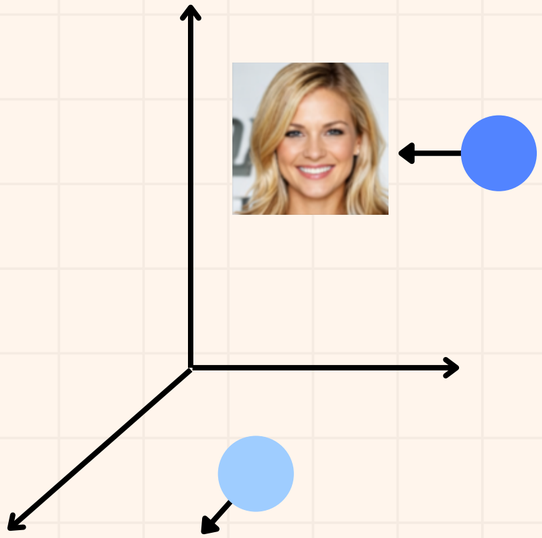
TRAINING SRATEGY

Weakly Supervised Triplet Loss

Image anchor

Correct attribute embedding

All other attribute embedding in the batch

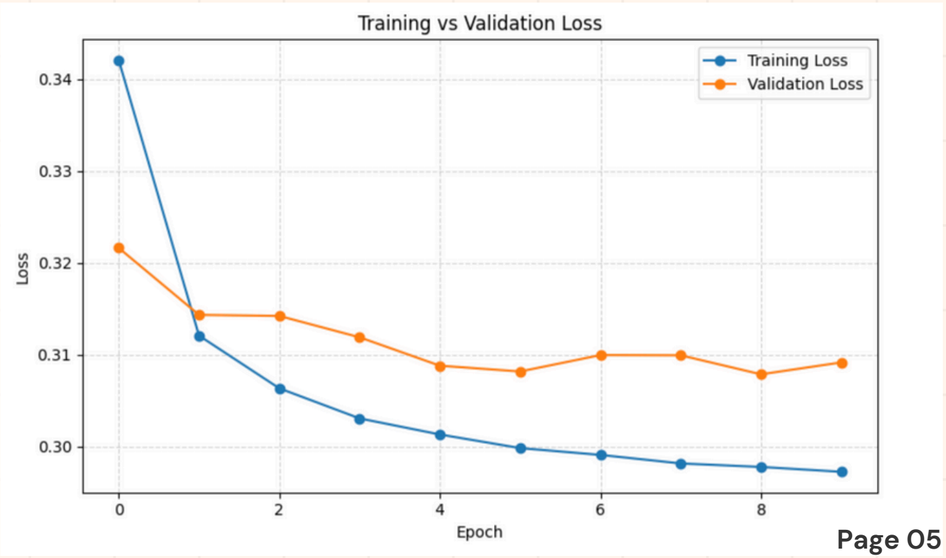


$$L = \max(0, d_{pos} - d_{neg} + \text{margin})$$

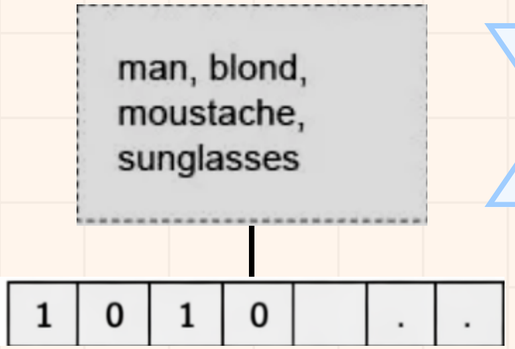
TRAINING SETUP

- **Optimizer**: Adam
- **Margin**: 0.5
- **Batch size**: 128
- **Epochs**: 10
- **Hard negatives per batch**: 4

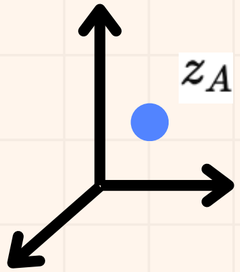
$$L_{\text{total}} = L_{\text{triplet}} + 0.5 \cdot L_{\text{hard\_negatives}}$$



RETRIEVAL



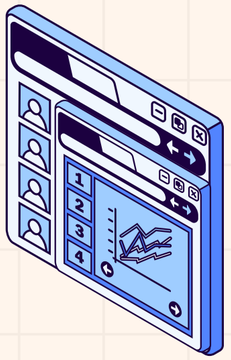
attribute query



encode it into a 300-D embedding

$$\text{scores} = \text{image\_embeddings} \cdot z_A$$

computes cosine similarity with all precomputed image embeddings



ranks all images by similarity and returns the Top-K most similar images

EVALUATION

To evaluate how well the model retrieves correct images, we performed 200 evaluation queries.

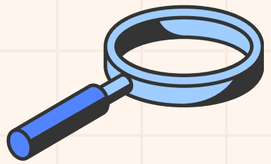
EVALUATION QUERY

- Select a random test image (ground-truth)
- Randomly choose 3 of its 40 attributes
- Encode the query into a 300-D attribute embedding
- Compute similarity with all precomputed image embeddings.
- Rank all images by similarity and retrieve Top-K.

COMPUTING METRICS

For each query, we measure:

- Precision@K
- Recall@K
- mAP



Focus next slide

RESULTS AND CONCLUSIONS

QUANTITATIVE RESULTS

P@10 = 0.636

In the Top-10, more than half of the images are relevant.

R@10 = 0.00127

Recall is naturally near zero because only few images in the dataset match exactly 3 chosen attributes.

mAP = 0.4988

This value measures the quality of the entire ranking.
Relevant images tend to appear early in the ranking,
The ordering among them is meaningful

QUALITATIVE RESULTS