Identifying Individual People

Presenters:
Abdelrahman Zahran
Fatma Melisa Küçük
Hasan Tarık Yumbul

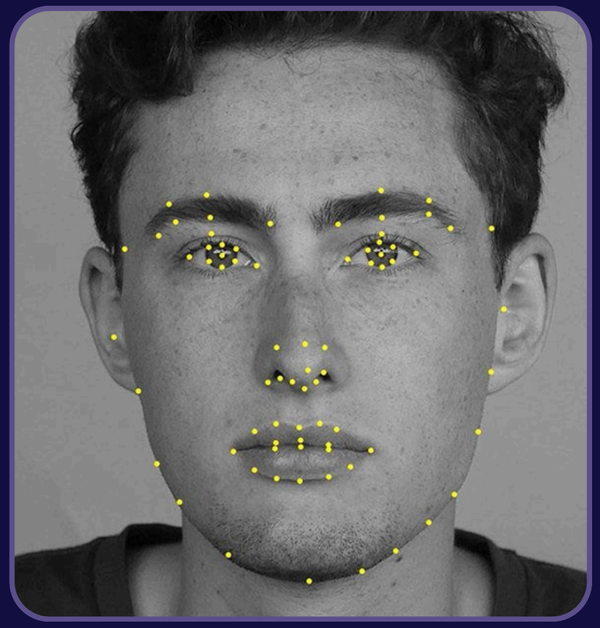
Problem and Motivation

We need to identify people in visual data for several reasons:

- Security and surveillance
- Smart retail and behavior tracking
- Video analytics

Solution

A combination of;



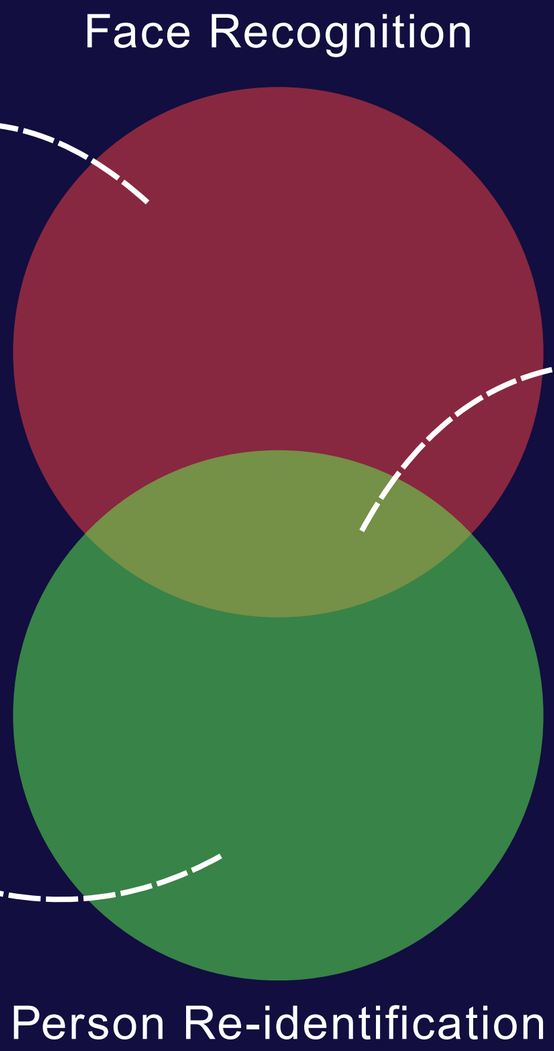
Face Recognition

Training deep learning models to recognize individual people by detecting their facial properties

Person Re-identificaion

Creating neural networks to detect certain shapes and colors of individual people’s bodies and outfits

What are the hardships?



- Expressions and Emotion
- Aging

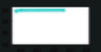
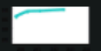
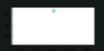
- Similar Clothing or Appearance
- Crowded Scenes
- Spatial Constraints
- Background Clutter

- Variations in Illumination
- Pose and Viewpoint Variation
- Occlusion
- Low-Resolution and Quality Issues
- Data Scarcity and Imbalance
- Computational Complexity and Scalability
- Temporal Changes and Identity Ambiguity
- Bias and Ethical Concerns
- Adversarial Attacks
- Limited Labeled Data

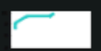
What is the current situation?

Most Famous Models & Datasets:

Face Recognition

Trend	Dataset	Best Model
	LFW	GhostFaceNetV2-1 (MS1MV3)
	CFP-FP	GhostFaceNetV2-1
	MLFW	MS1MV2, R100, SFace
	CelebA+masks	Fine-tuned ArcFace
	CASIA-WebFace+masks	Fine-tuned ArcFace
	IJB-B	ArcFace+CSFM
	AgeDB-30	Prodpoly
	Color FERET	PIC - QMagFace
	CALFW	Prodpoly
	mebeblurf	PIC - MagFace

Person Re-identification

Trend	Dataset	Best Model
	Market-1501	st-ReID(RE, RK)
	DukeMTMC-reID	DenseIL
	MSMT17	CLIP-ReID (with re-ranking)
	Occluded-DukeMTMC	KPR + SOLIDER
	Market-1501-C	TransReID
	MARS	B-BOT + OSM + CL Centers* (Re-rank)
	CUHK03 labeled	Weakly Pre-training (ResNet101+RK)
	CUHK03	Proposed SGGNN
	CUHK03 detected	Top-DB-Net + RK
	PRID2011	B-BOT + Attention and CL loss*

We will be explaining and competing this model

Spatial-Temporal Person Re-identification (st-ReID)

Spatial-Temporal Person Re-identification

8 Dec 2018 · Guangcong Wang, Jian-Huang Lai, Peigen Huang, Xiaohua Xie · Edit social preview

Most of current person re-identification (ReID) methods neglect a spatial-temporal constraint. Given a query image, conventional methods compute the feature distances between the query image and all the gallery images and return a similarity ranked table. When the gallery database is very large in practice, these approaches fail to obtain a good performance due to appearance ambiguity across different camera views. In this paper, we propose a novel two-stream spatial-temporal person ReID (st-ReID) framework that mines both visual semantic information and spatial-temporal information. To this end, a joint similarity metric with Logistic Smoothing (LS) is introduced to integrate two kinds of heterogeneous information into a unified framework. To approximate a complex spatial-temporal probability distribution, we develop a fast Histogram-Parzen (HP) method. With the help of the spatial-temporal constraint, the st-ReID model eliminates lots of irrelevant images and thus narrows the gallery database. Without bells and whistles, our st-ReID method achieves rank-1 accuracy of 98.1\% on Market-1501 and 94.4\% on DukeMTMC-reID, improving from the baselines 91.2\% and 83.8\%, respectively, outperforming all previous state-of-the-art methods by a large margin.

PDF | Abstract

Code

Edit

Wanggcong/Spatial-Temporal-Re-ident... official	★ 396	PyTorch
SurajDonthi/Multi-Camera-Person-Re-... Quickstart in Colab	★ 228	PyTorch
BonaventureR/person-reid	★ 3	PyTorch

Tasks

Edit

Person Re-Identification

Datasets

Edit

Market-1501 DukeMTMC-reID

Task	Dataset	Model	Metric Name	Metric Value	Global Rank	Uses Extra Training Data
Person Re-Identification	DukeMTMC-reID	st-ReID(RE, RK,Cam)	Rank-1	94.5	# 2	×
			mAP	92.7	# 5	×
Person Re-Identification	Market-1501	st-ReID(RE, RK)	Rank-1	98.0	# 1	✓
			Rank-5	98.9	# 1	✓
			mAP	95.5	# 4	✓

https://paperswithcode.com/paper/spatial-temporal-person-re-identification

Key Novelties of st-ReID

Two-Stream Architecture

- **Visual Stream:** Extracts semantic appearance features (color, texture, etc.).
- **Spatial-Temporal Stream:** Incorporates where and when a person appears, leveraging camera topology and timestamps

Logistic Smoothing Joint Similarity Metric

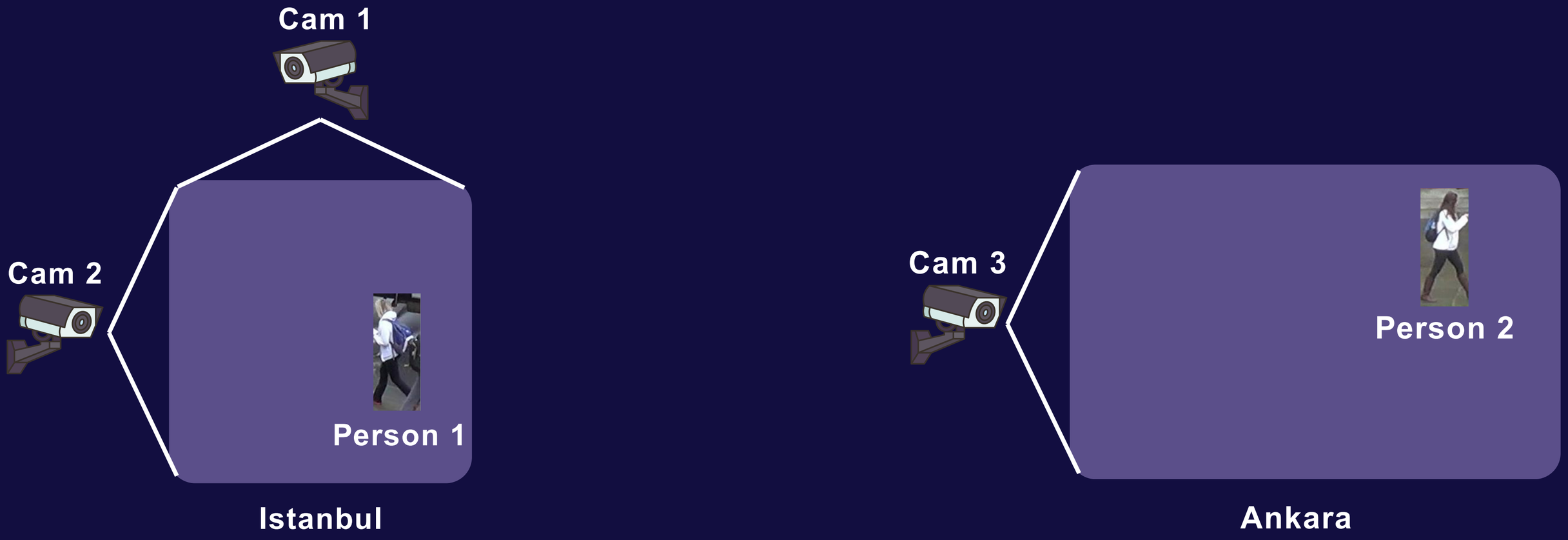
- Seamlessly blends appearance similarity with spatial-temporal likelihood, discounting unlikely matches (e.g., distant camera within too short time)

Histogram-Parzen Density Estimation

- Learns a flexible, data-driven spatial-temporal probability distribution between camera pairs, enabling more accurate transition modeling

Basically, a person re-identification algorithm that not only **detects people**, but also considers the **date and the location a person is or can be present**. And it **learns from these patterns** to decide better next time.

Visualization



~1.1 hr on flight (445.4 km)

Person 1 is detected by cams 1 and 2 at 14:23 on 01/07/2025 in Istanbul

CORRECT

Person 2 was a false positive for Person 1 at 14:32 on 01/07/2025 in Ankara

DISCARDED

Behind the Screen

1. Input

- Query image I_i and gallery image I_j
- Camera IDs: c_i, c_j, Timestamps: t_i, t_j

2. Visual Feature Extraction

- Extract features x_i, x_j using Part-based Convolutional Baseline (PCB) + ResNet50
- Compute cosine similarity:

$$s(\mathbf{x_i}, \mathbf{x_j}) = \frac{\mathbf{x_i} \cdot \mathbf{x_j}}{||\mathbf{x_i}||||\mathbf{x_j}||}$$

3. Spatial-Temporal Probability Estimation

- Build histogram of time gaps k between cameras $c_i \rightarrow c_j$:

$$\hat{p}(y=1|k, c_i, c_j) = \frac{n^k_{c_i c_j}}{\sum_l n^l_{c_i c_j}}$$

- Smooth with Parzen kernel:

$$p(y=1|k, c_i, c_j) = \frac{1}{Z} \sum_l \hat{p}(y=1|l, c_i, c_j) K(l-k)$$

- Gaussian kernel:

$$K(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-x^2}{2\sigma^2}}$$

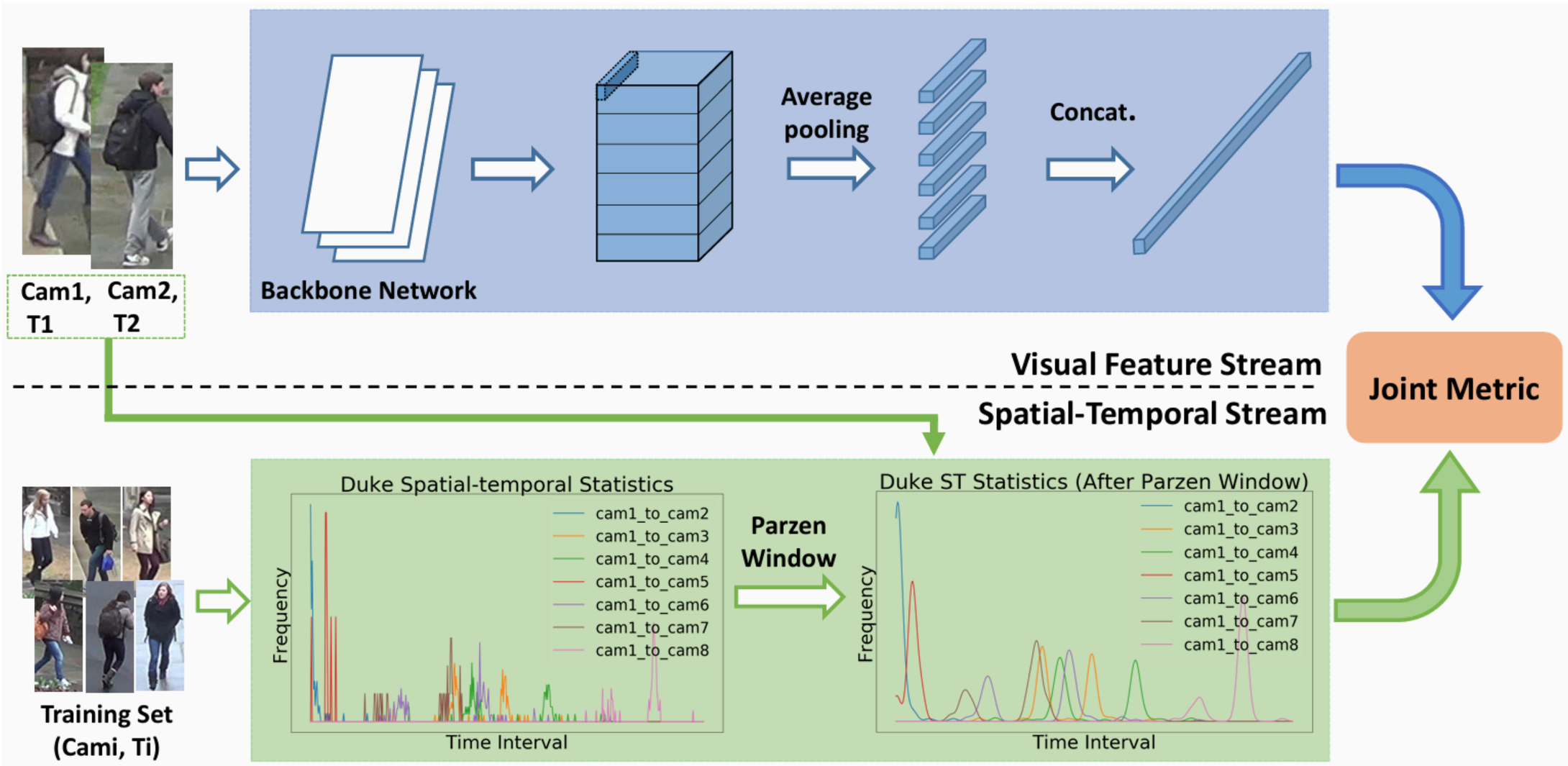
4. Final Joint Similarity Score

- Apply logistic smoothing:

$$f(x; \lambda, \gamma) = \frac{1}{1 + \lambda e^{-\gamma x}}$$

- Compute joint probability:

$$p_{joint} = f(s; \lambda_0, \gamma_0) f(p_{st}; \lambda_1, \gamma_1)$$


Cam1, T1
Cam2, T2
Backbone Network
Average pooling
Concat.
Visual Feature Stream
Spatial-Temporal Stream
Joint Metric
Training Set (Cami, Ti)
Duke Spatial-temporal Statistics
Frequency
Time Interval
cam1_to_cam2
cam1_to_cam3
cam1_to_cam4
cam1_to_cam5
cam1_to_cam6
cam1_to_cam7
cam1_to_cam8
Parzen Window
Duke ST Statistics (After Parzen Window)

Our Model: Identrix

A dual-backbone model that fuses two CNNs to extract richer, complementary features.

1 Data Preprocessing

Labels

Anchor: an image of a person
Positive: a different image of the same person
Negative: an image of a different person

TripletDataset:
Loads images from Market-1501 gallery split.

---

2 Dual-backbone Architecture

Encoders

ResNet50: Extracts local fine-grained features
ViT-B16 (Vision Transformer): Captures global semantic context

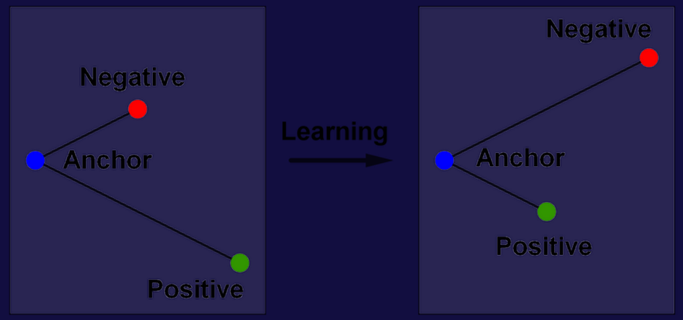
Leverage two complementary CNN backbones to extract richer person features for Re-ID embedding learning.

---

3 Loss Function

Triplet Loss, compares a baseline input to positive input and a negative input.

$$\mathcal{L} = \frac{1}{B}\sum_{i=1}^{B} -\log \frac{\exp\left(\alpha \cdot \mathrm{sim}(a_i, p_i)\right)}{\sum_{j=1}^{B} \exp\left(\alpha \cdot \mathrm{sim}(a_i, c_j)\right)}$$



A Detailed Look at the Embedding Creation and Data Handlers

DualBackboneNet Architecture

- Combines ResNet50 (CNN) for local features and ViT (Transformer) for global context.
- Projected to shared space (embedding_dim=128), fused via attention mechanism.
- Final fused vector passed through output head → learned embedding.

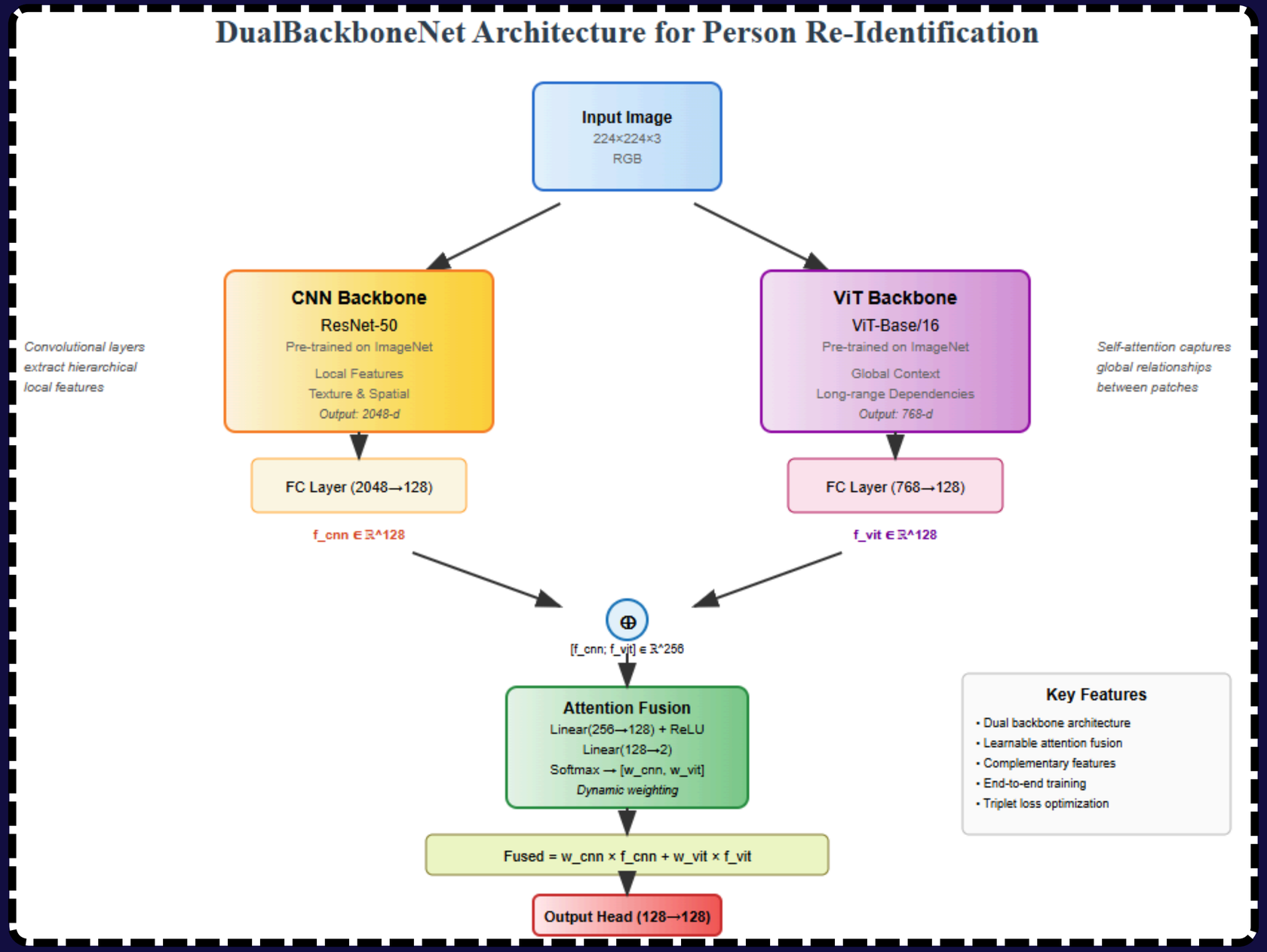
TripletDataset for Metric Learning

- Loads Market-1501 gallery set.
- Dynamically forms (anchor, positive, negative) triplets:
  - Anchor/Positive: 2 images of the same person.
  - Negative: image of a different person.
- Ensures ≥2 images per ID; applies random rotation, flip, color jitter.

Embedding & Matching Logic

- get_embeddings(): Preprocess images → forward pass through model → NumPy array.
- match_topk(): Computes cosine similarity between query & gallery embeddings → Top-K ranking.
- Used in both evaluation and Streamlit demo.

Deep into Dual Backbone Architecture



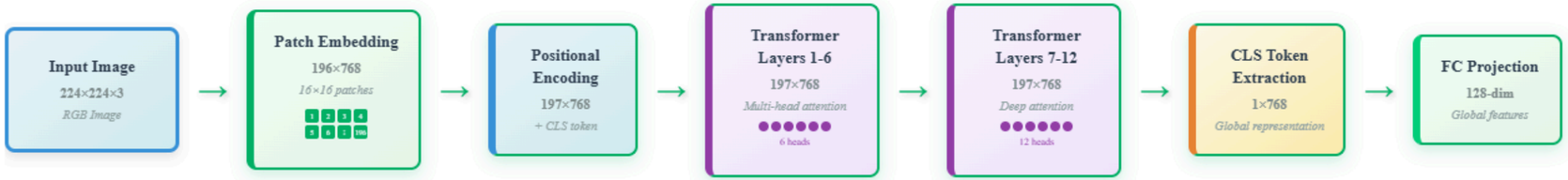
CNN Backbone: ResNet-50

Convolutional Neural Network for Local Feature Extraction

25.6M	2048	5	128
Total Parameters	Feature Dimension	Residual Stages	Output Embedding

Vision Transformer (ViT-Base)

Self-Attention for Global Context Understanding



86.4M	768	12	12	196	128
Total Parameters	Hidden Dimension	Transformer Layers	Attention Heads	Image Patches	Output Embedding

Attention-based Feature Fusion

Adaptive Weighting of CNN and Transformer Features

CNN Features
128-dimensional
Local spatial features

ViT Features
128-dimensional
Global contextual features

↓

Concatenation
256-dim vector

→

MLP Attention
256→128→2

→

Softmax
w_1, w_2

→

Weighted Sum
128-dim output

Fusion Formula

$$f_{concat} = [f_{CNN}; f_{ViT}] \in \mathbb{R}^{256}$$

$$\alpha = MLP(f_{concat}) \in \mathbb{R}^{2}$$

$$[w_1, w_2] = Softmax(\alpha)$$

$$f_{fused} = w_1 \cdot f_{CNN} + w_2 \cdot f_{ViT}$$

Constraint: $w_1 + w_2 = 1$, $w_1, w_2 \geq 0$

MLP Attention Network	**256→128→2** Network Architecture	**ReLU** Activation Function
Softmax Weight Normalization	**Adaptive** Feature Weighting	**128-dim** Final Embedding

Output Layer: Person Embedding

Final Feature Vector for Person Re-Identification

Person Embedding

128-dimensional

Optimized for metric learning with triplet loss

128	Triplet Loss
Embedding Dimension	Optimization

Training Pipeline

Training Overview:

Model: DualBackboneNet (ResNet50 + ViT)

Loss Function: Triplet Loss (contrastive embedding learning)

Optimizer: Adam (LR = 1e-4)

Frameworks: PyTorch, WandB for logging

Custom Triplet Sampling

TripletDataset dynamically generates anchor–positive–negative triplets.

Anchor/Positive: Two different images of the same ID

Negative: Image from a different ID

Requires ≥2 images per ID (checked on init)

Data Augmentation

```
transform_train = T.Compose([
    T.Resize((224, 224)),
    T.RandomHorizontalFlip(p=0.5),
    T.RandomRotation(10),
    T.ColorJitter(brightness=0.2, contrast=0.2, saturation=0.2, hue=0.1),
    T.ToTensor(),
    T.Normalize([0.5]*3, [0.5]*3) # Normalize to [-1, 1]
])
```

Helps model generalize across lighting, pose, and orientation.

Training Loop per Batch

```
anchor_emb = model(anchor)
positive_emb = model(pos)
negative_emb = model(neg)

loss = triplet_loss(anchor_emb, positive_emb, negative_emb)

optimizer.zero_grad()
loss.backward()
optimizer.step()
```

Embeddings are computed via the full dual-backbone model. Triplet loss ensures that Anchor–positive closer than anchor–negative by a margin

Logging / Monitoring

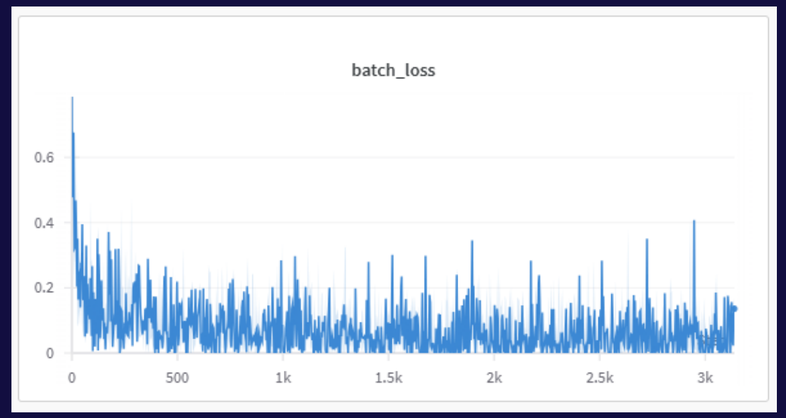
Tracked via Weights & Biases

- Batch loss
- Epoch loss
- Learning rate
- Model checkpoint artifacts

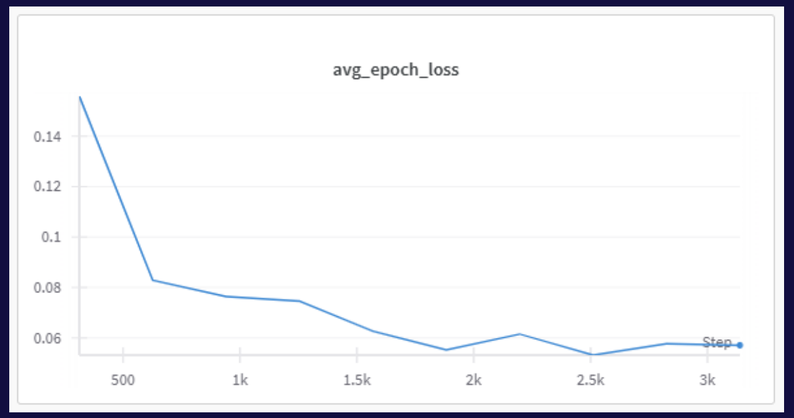
Training supports multi-GPU (DataParallel) if available

Model is saved at every epoch and also uploaded as W&B artifact with metadata

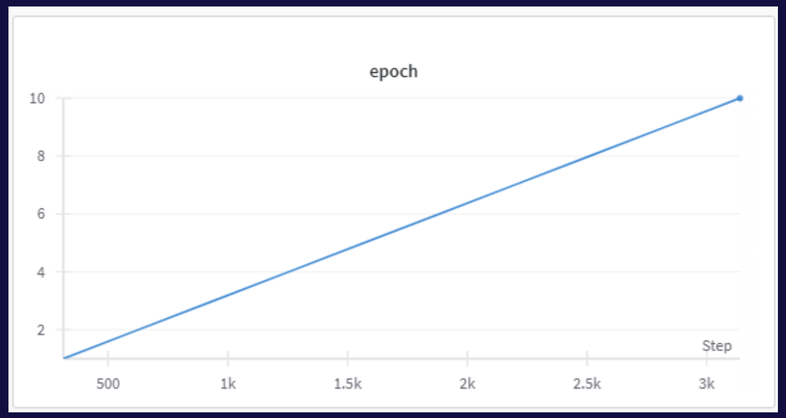
Results & Evaluation



Stable Learning
The batch loss graph shows a sharp drop early in training and stabilizes with low variance, indicating effective convergence and no sign of instability.



Progressive Improvement
The average epoch loss steadily decreases, demonstrating consistent model improvement over time with diminishing returns, a classic sign of healthy training.



No Overfitting Detected
The epoch vs step trend aligns with the loss metrics, and there's no sudden spike or plateau, suggesting the model generalizes well without signs of overfitting.

Results & Evaluation

Thanks for Listening!