DS@GT AnimalCLEF: Triplet Learning over ViT Manifolds with Nearest Neighbor Classification for Animal Re-identification

Notebook for the LifeCLEF Lab at CLEF 2025

Anthony Miyaguchi[1,*], Chandrasekaran Maruthaiyannan[1] and Charles R. Clark[1,2,*]

[1]Georgia Institute of Technology, North Ave NW, Atlanta, GA 30332
[2]University of Florida, Gainesville, FL 32610

Abstract

This paper details the DS@GT team's entry for the AnimalCLEF 2025 re-identification challenge. Our key finding is that the effectiveness of post-hoc metric learning is highly contingent on the initial quality and domain-specificity of the backbone embeddings. We compare a general-purpose model (DINOv2) with a domain-specific model (MegaDescriptor) as a backbone. A K-Nearest Neighbor classifier with robust thresholding then identifies known individuals or flags new ones. While a triplet-learning projection head improved the performance of the specialized MegaDescriptor model by 0.13 points, it yielded minimal gains (0.03) for the general-purpose DINOv2 on averaged BAKS and BAUS. We demonstrate that the general-purpose manifold is more difficult to reshape for fine-grained tasks, as evidenced by stagnant validation loss and qualitative visualizations. This work highlights the critical limitations of refining general-purpose features for specialized, limited-data re-ID tasks and underscores the importance of domain-specific pre-training. The implementation for this work is publicly available at github.com/dsgt-arc/animalclef-2025.

Keywords

Animal Re-identification, Open-Set Re-identification, Triplet Learning, Metric Learning, Vision Transformer (ViT), DINOv2, MegaDescriptor, Nearest Neighbor Classification, Kaggle, LifeCLEF, DS@GT

1. Introduction

Individual animal identification is helpful for biologists studying animal populations in the wild. The ability to track an animal over time in its natural environment gives insights into behaviors and ecological interactions that are not possible with general census statistics.

In this paper, we describe the solution developed by the DS@GT team for the AnimalCLEF 2025 challenge hosted on Kaggle. We utilize pre-trained, self-supervised vision transformers to embed animal images into embedding space and run a K-NN classifier with statistical thresholding to determine a label for each image. We further refine the manifold learned by the vision transformer using a triplet learning procedure that learns to map individuals in space more effectively, achieved by projecting triplets of images from the ViT embedding space to a new projection for the metric. We hypothesize that a domain-specific backbone (MegaDescriptor) will provide a more suitable initial embedding manifold for triplet-based refinement than a general-purpose backbone (DINOv2), leading to greater performance gains on this specialized re-ID task. Our method can overcome a simple baseline provided by the competition organizers, but further work is necessary.

CLEF 2025 Working Notes, 9 – 12 September 2025, Madrid, Spain
*Corresponding author.
acmiyaguchi@gatech.edu (A. Miyaguchi); chand2020@gatech.edu (C. Maruthaiyannan); cclark339@gatech.edu (C. R. Clark)
0000-0002-9165-8718 (A. Miyaguchi); 0009-0005-5757-4480 (C. Maruthaiyannan); 0009-0008-0415-111X (C. R. Clark)

2. Related Work

2.1. Animal Re-identification

Animal re-identification (re-ID) refers to a system's ability to predict the identity of an individual animal based on its unique physical traits [1]. It is critical for biologists and ecologists to monitor populations, track movement, and study social behavior [2]. Approaches differ in their descriptor strategies. Early methods rely on local descriptors, such as SIFT, SURF, or contour features extracted from key points, to identify individuals via match counts [2]. More recent approaches use deep neural networks with metric learning to generate feature embeddings for identity matching [2]. Hybrid pipelines combine object detection and feature extraction to localize animals (or faces) before identifying them [3, 4]. Numerous datasets support benchmarking: ATRW includes 3,649 images of 92 Amur Tigers in zoos [5]; zebrafishRe-ID offers 2,224 images of 6 zebrafish in lab settings [6]; and Cows2021 provides 13,784 images of 182 cows on a farm [7]. The WildlifeReID-10K dataset aggregates 36 wildlife re-ID datasets, comprising approximately 140,000 images from over 10,000 individuals across multiple species [8]. Although many studies frame re-ID as a closed-set task, this assumption often breaks down in ecological settings where new individuals may appear.

2.2. Vision Transformers for Computer Vision

Dosovitskiy *et al.* introduced the Vision Transformer (ViT), adapting the Transformer architecture to image patches, which achieved competitive performance with less computing when pre-trained on large datasets [9, 10]. However, quadratic scaling in image size led Liu *et al.* to propose the Swin Transformer, which utilizes shifted-window attention for hierarchical, linear-complexity feature extraction and achieves strong performance across dense vision tasks [11]. In re-ID, vision transformers have been adopted to capture both short- and long-range features. TransReID was the first ViT-based method for person re-ID [12], while GorillaVision applied a pre-trained ViT backbone to gorilla face recognition [3].

More recently, self-supervised and multi-modal transformer models have become powerful tools for vision tasks. DINOv2, a self-supervised ViT trained via self-distillation, performs well on fine-grained tasks such as species recognition [13, 14]. CLIP, trained on image-text pairs, learns image embeddings that generalize well across domains, including re-ID [15, 16]. MegaDescriptor, a Swin-based model trained on a large multi-species re-ID dataset, achieves state-of-the-art performance across animal re-ID benchmarks, outperforming models like DINOv2 and CLIP [2].

2.3. Metric Learning

Metric learning underpins many re-ID approaches. Triplet loss trains models on anchor-positive-negative triplets to ensure embeddings of same-identity pairs are closer than those of different identities [17]. Popularized initially in face recognition, triplet learning is widely used in re-ID. To enhance separability, angular and margin-based losses such as ArcFace introduce additive angular margins on the hypersphere [18], with adaptations for re-ID. Recent methods, such as Matryoshka Representation Learning (MRL), produce hierarchical embeddings that encode coarse-to-fine details, enabling the model to dynamically select embedding subspaces for efficient nearest-neighbor search, depending on the retrieval task.

3. Methodology

Our experimentation is structured into two major phases, each serving a distinct purpose in our research. Our first experiment validates the relative performance between DINOv2 [13] and MegaDescriptor [2] using a K-NN classifier. We use the pre-trained models in a zero-shot fashion and tune the threshold distance for new identities in the model using our dataset split. In our second set of experiments, we undertake the task of reshaping the manifold. This involves a deliberate effort to bring images of

the same animal closer together and push images of different animals further apart. The goal is to disambiguate individuals and to make thresholds more robust.

3.1. Competition Evaluation Metric

The competition we participate in employs two specialized evaluation metrics, each of which plays a crucial role in assessing the performance of our models. The first is Balanced Accuracy on Known Samples (BAKS) which measures the ability to identify individuals present in the training set. The second is the Balanced Accuracy on Unknown Samples (BAUS) which is the ability to classify new individuals not in the training set as unknown.

More formally, we define the set of individuals I into known subset I_K and unknown subset I_U. N_c is the the number of images for an individual c in set I.

$$\text{BAKS}(y, \hat{y}) = \frac{1}{|I_K|} \sum_{c \in I_K} \left(\frac{1}{n_c} \sum_{i=1}^{N} \mathbf{1}(y_i = c) \cdot \mathbf{1}(\hat{y}_i = c) \right) \tag{1}$$

$$\text{BAUS}(y, \hat{y}) = \frac{1}{|I_U|} \sum_{c \in I_U} \left(\frac{1}{n_c} \sum_{i=1}^{N} \mathbf{1}(y_i = c) \cdot \mathbf{1}(\hat{y}_i = y_{\text{unknown}}) \right) \tag{2}$$

$$\text{Score}(y, \hat{y}) = \sqrt{\text{BAKS}(y, \hat{y}) \cdot \text{BAUS}(y, \hat{y})} \tag{3}$$

Our final score is then the geometric mean of the two measures. Like the F1-score encourages models to balance precision and recall, the AnimalCLEF metric encourages models to be able to identify known and unknown individuals.

3.2. Dataset Split for Open Set Classification

The training images are stratified by individual into train, validation, and test sets, ensuring a robust and comprehensive dataset split. The training set is used to fit a classification model, the validation set to observe progress across hyperparameter searches, and a test set for objective evaluation. We organize the split in such a way that we can optimize a machine-learning algorithm to fit both the BAKS and BAUS objectives e.g. being able to be accurate about identifications of existing individuals and new individuals.

Table 1
Dataset Split Summary. The train split is used to train the re-identification model to distinguish between known individuals. The validation split is for hyperparameter tuning and model selection. The test split is for final, unbiased performance evaluation. In both validation and test splits, BAKS is calculated on the known individuals and BAUS on the unknown individuals.

Split	Num Individuals	Num Images	Known Individuals	Unknown Individuals
Train	404	3392	404	0
Validation	458	2575	404	54
Test	620	6568	404	216

To predict the labels of existing individuals, we must ensure that there are known individuals shared between the training, validation, and test sets. To predict unknown individuals, we select a set of individuals that are excluded from the training set but are known in the validation and test sets. We describe the statistics of the train-validation-test split in table 1. We use 60% of the training individuals for training, 20% for validation, and the remaining 20% for testing. If an individual has only a single image, it belongs to the training dataset by default.

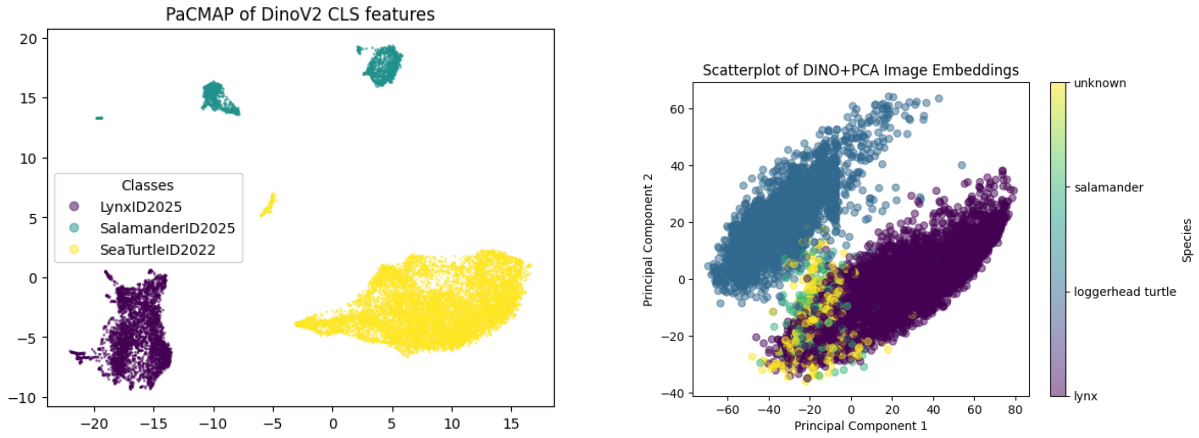
3.3. Transfer Learning via ViT Embedding Extraction

We hypothesize that pre-trained self-supervised vision transformer models learn an adequate feature space for distinguishing individuals. The images are projected onto a lower-dimensional manifold that roughly maps semantic distances found in the original space. The new points are called embeddings and are vectors of numbers that capture the lower dimensional latent space. Embeddings capture semantic similarities between images through inductive biases of the model and the distribution of the training dataset. Vision transformers learn to represent an image through a sequence of tokens derived from patches of the original image in addition to a special token called the classification (CLS) token. We capture and transfer the underlying knowledge by extracting embeddings from the model by extracting the CLS token.

Table 2
Comparison of DINOv2 and MegaDescriptor Foundation Models

Feature	DINOv2 (Base)	MegaDescriptor (Large)
Underlying Architecture	Vision Transformer (ViT-B)	Swin Transformer (Swin-L)
Parameter Size	∼87 Million	∼229 Million
Output Dimension	768	1024
Training Dataset	LVD-142M: A large-scale, general-purpose dataset of 142 million images.	A collection of 29+ public animal re-identification datasets (e.g., WildlifeReID-10k).
Specialization	General-purpose vision foundation model.	Foundation model trained for wildlife re-identification.

We can demonstrate a degree of visual separation by projecting the embeddings onto a 2D manifold, which can be visualized as a scatter plot. In figure 1, we embed the entire dataset using a pre-trained DINOv2 model from HuggingFace. We then use principle component analysis (PCA) and pairwise controlled manifold approximation (PaCMAP [19]) to visualize how the points cluster in space. PCA works by normalizing the vectors into zero-mean and unit-variance matrix and then finding the rotation of the matrix that minimizes the projection in a lower-rank space. The first two dimensions correspond to the principal axes of rotation, as determined through an eigen-decomposition. We find that this projection can clearly separate lynxes from sea turtles, with some overlap between lynxes and salamanders. We compare this approach with PaCMAP, which takes into account both local and global geometry by constructing a graph sampled from the original data. This embedding better captures nuances of the original space and separates images in the training dataset into lynxes, sea turtles, and salamanders.



(a) DINO CLS Embeddings with PaCMAP

(b) DINO CLS Embeddings with PCA

Figure 1: Comparison of dimensionality reduction techniques (PaCMAP vs. PCA) on the DINO CLS token embeddings from the same dataset.

3.4. Nearest Neighbor Classification

The nearest neighbor classification takes the embeddings and determines which individual is closest to the point. Images in the training dataset are used as prototypes for making the classification. We use Faiss [20] to index all of the training image embeddings for queries. We use the L2 distance between points to rank a query vector to all of the training vectors and return the top K points. We look at the nearest point and determine whether this is a new individual by applying a global threshold to the points. If the nearest point is too far, then this is a new point. Otherwise, we return the mode of the top K identities.

We choose the threshold through a procedure that optimizes the competition metric. The threshold is selected by searching 100 linearly spaced points within a range of 3 Median Absolute Deviations (MAD) from the median distance between each image and its nearest neighbor of a different species. The threshold that maximizes the competition score (geometric mean of BAKS and BAUS) on the validation set is chosen. These statistics are robust indicators of the dataset and are less influenced by outliers. The MAD is also a value that is independent of the domain of the thresholds, such that we can describe distances in their modified z-score in the distribution of distances.

3.5. Triplet Learning

We also applied the triplet learning paradigm in order to learn a better representation of the data. The objective of the triplet loss is to ensure that same-identity pairs are closer than those of different identities using an anchor-positive-negative triplet.

$$L = \max(d(x_a, x_p) - d(x_a, x_n) + \alpha, 0)$$

In this formulation, x_a represents the embedding of an anchor image, x_p is the embedding of a positive image from the same individual, and x_n is the embedding of a negative image from a different individual. The function d calculates the L2 distance between two embeddings, and the hyperparameter α represents the margin that enforces separation between the pairs. The loss is minimized only when the distance between the anchor and positive pair is smaller than the distance between the anchor and negative pair by at least the margin α. For our experiments, we followed a standard approach and utilized a unit margin where $\alpha = 1$.

We pre-compute embeddings with DINOv2 and MegaDescriptor-L-384 derived from the CLS token in the ViT. These CLS embeddings were then downsampled by a projection head consisting of two linear layers with GELU activation and dropout sandwiched between them, followed by L2 normalization after the second linear layer. The first linear layer is equal to the size of the original embedding space, and the second layer is set to a value of 256. The model must be parameterized in such a way that it can capture the relationship between triplets, given their locations in the original manifold, with the ability to generalize to new examples. This pipeline is depicted in Figure 2.

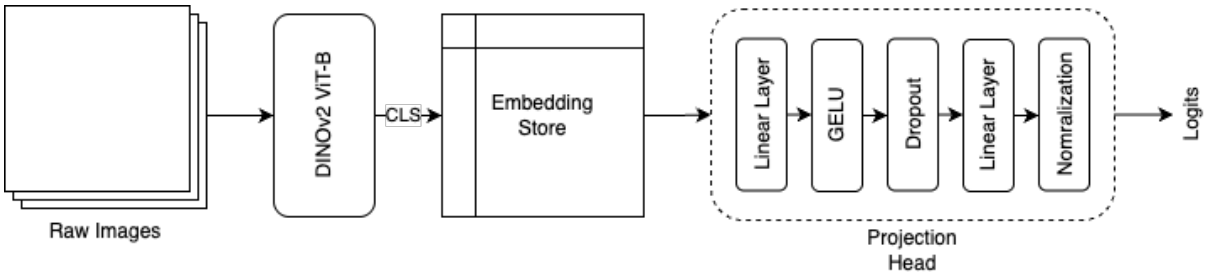


Figure 2: Our triplet learning pipeline. Raw images are pre-processed by a frozen DINOv2 ViT-B model, from which the 768-dimensional CLS embeddings are extracted and stored. From this store, the CLS embeddings are passed to the projection head, which is made up of a linear layer, GELU activation, dropout, a second linear layer, and L2 normalization.

The projection head was trained with a batch size of 200 on the CLS embeddings for the images in the database, split over 100 epochs in total. We experimented with using standard triplet loss [17] as well as a modified triplet loss using Matryoshka Representation Learning [21], both with unit margins. We also experimented with different online triplet mining techniques, specifically random selection and semi-hard negative selection [17]. The Adam optimizer was used with a learning rate of 5×10^{-4}; a linear scheduler was employed for warmup, followed by cosine annealing after the 10th epoch.

4. Results

We report our final private leaderboard score on Kaggle. Our best model, which utilized MegaDescriptor with triplet loss, achieved a score of 0.39 and ranked 103 out of 174. This is 0.09 points above the MegaDescriptor baseline provided by the competition organizers and 0.05 points below the WildFusion baseline.

Table 3
Leaderboard rankings and scores. The rank is given by the private/public rankings, with the public rankings being out of 174 teams. The score is given by the competition metric.

Rank	Name	Public	Private
1/1	DataBoom	0.72781	0.71388
2/2	webmaking	0.65832	0.67420
3/3	hzhzh	0.63448	0.65753
98/88	WildFusion - MegaD. + ALIKED	0.36555	0.44362
100/103	DS@GT LifeCLEF	0.35583	0.39082
126/128	MegaDescriptor-L-384	0.30002	0.30898

Table 4
Model performance of models submitted to the competition.

Name	Public Score	Private Score	Submission Name
dino baseline	0.24371	0.18856	20250313-baseline.csv
megadescriptor baseline	0.28528	0.25967	20250422-baseline.csv
dino linear	0.27116	0.27990	dino-semihard-epoch80-prediction.csv
dino nonlinear	0.18515	0.21752	nonlinear-v3-epoch20.csv
megadescriptor nonlinear	0.35583	0.39082	megadescriptor-nonlinear-128-epoch100.csv

We train several models with our described methodology in table 4. Our baseline models are the result of embedding the data into either DINOv2 base or MegaDescriptor-L-384 and then applying our K-NN classification procedure with thresholding. We train another model with the triplet learning embedding head, with a linear projection from the original embedding space down to a dimension of 256. Finally, we compare a non-linear projection learned by the final methodology.

In addition to the final results, we also report some of the training dynamics of the triplet learning procedure to illustrate the differences between DINOv2 and MegaDescriptor. In figure 3, we see that the triplet loss objective is lower across all epochs in MegaDescriptor over Dino. Both models decrease in loss over the 100 epoch during training, meaning that fewer triplets are violating the margin constraint during training, as observed by the valid triplets found in figure 4. However, we note that while the training loss continues to decrease, the validation loss converges quickly.

Finally, we report the hyperparameter tuning of the k-NN classification threshold in figure 5. For the triplet training epoch with the best validation loss, we run our tuning procedure to find the best threshold.

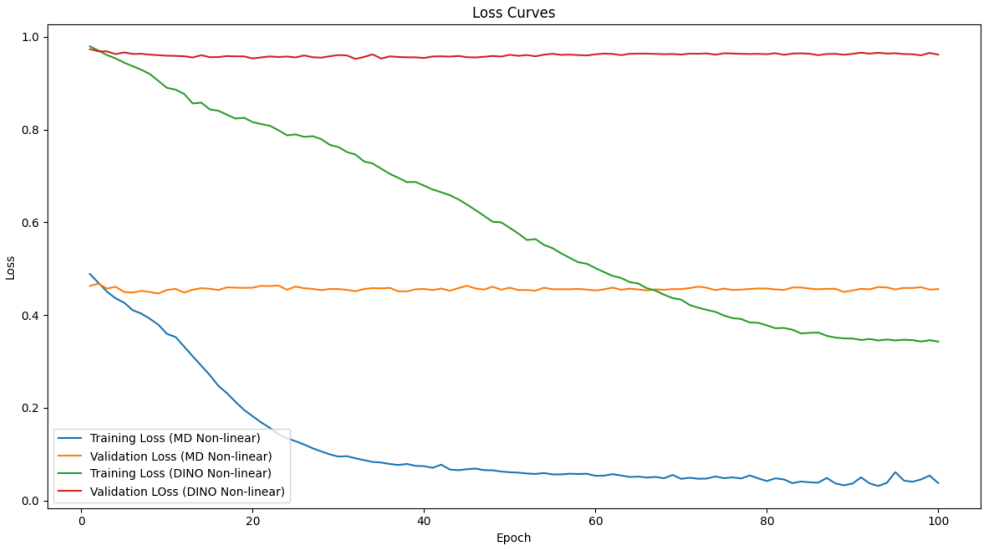


Figure 3: The training losses for the non-linear triplet mined embeddings. We observe that the MegaDescriptor head achieves significantly lower training and validation losses compared to Dino.

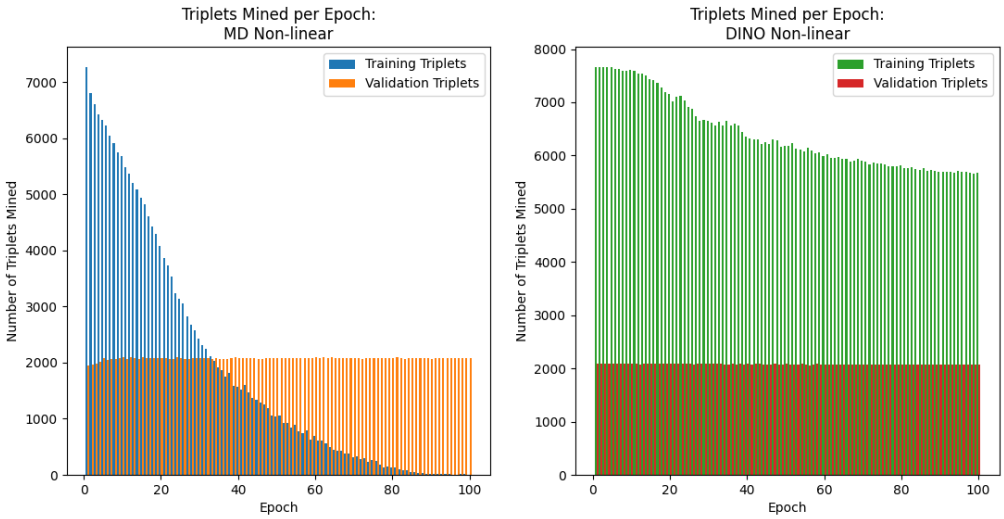
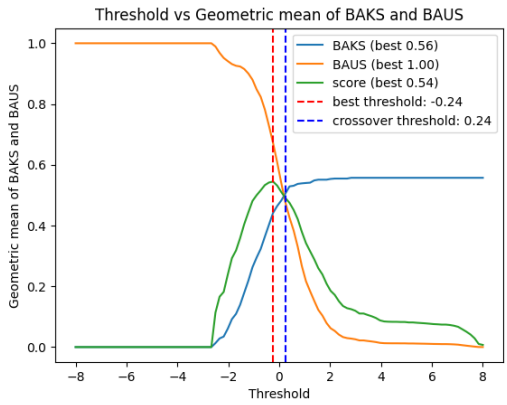


Figure 4: The number of triplets mined at epoch using a semi-hard negative mining routine. We note that MegaDescriptor can reduce the number of triplets that satisfy the semi-hard margin constraints over the 100-epoch training loop to a much larger degree than Dino.

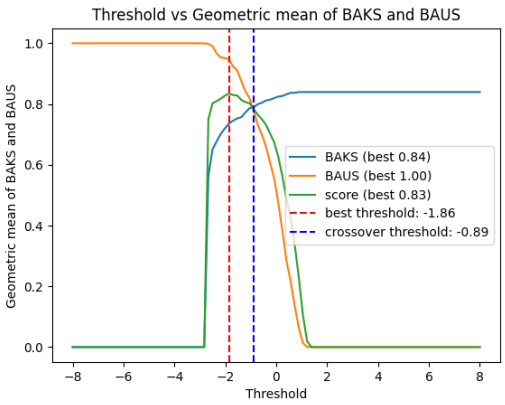
5. Discussion

Through our experiments, we found that domain-specific training is crucial for achieving good performance on re-identification tasks. We see this in our baseline k-NN behavior against the DINOv2 and MegaDescriptor embeddings. We observe a 0.07-point difference on the private leaderboard between these two models, with the MegaDescriptor model performing better.

Our procedure is 0.04 points lower than the competition baseline despite using the same model. This may be caused by improper resizing before calling the model or by selecting an inappropriate metric. The starter notebook uses cosine distance with a fixed threshold of 0.6, while we use Euclidean distance with a threshold chosen by a hyperparameter search influenced by our dataset split. The cosine distance is the more appropriate measure, and requires using an inner-product index with unit-normalized

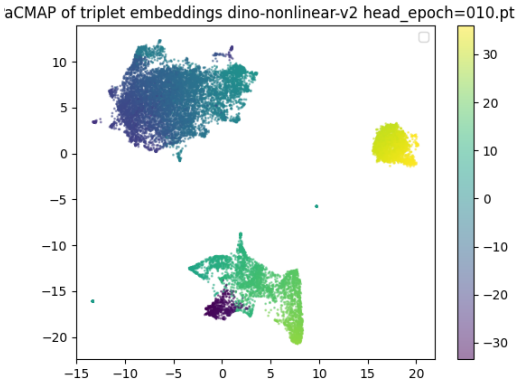


(a) DINO Non-Linear Validation Curve

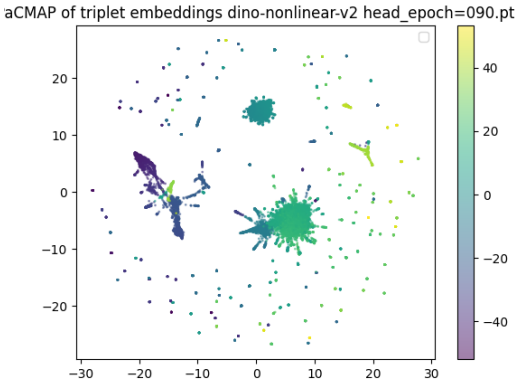


(b) MegaDescriptor Non-Linear Validation Curve

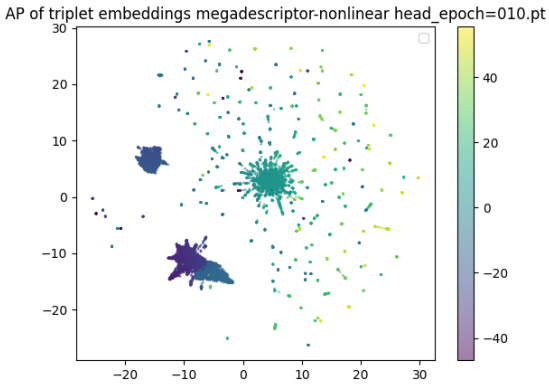
Figure 5: Validation curves used to determine the optimal classification threshold for each model.



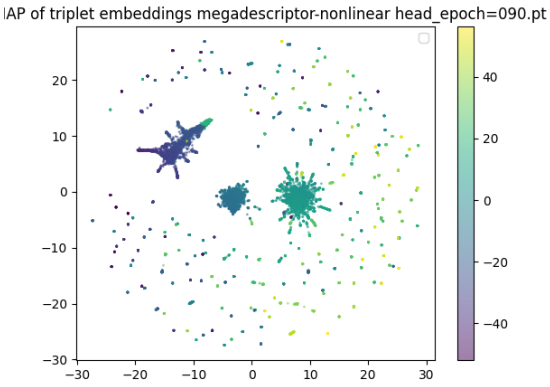
(a) DINO Non-Linear Embedding, Epoch 10



(b) DINO Non-Linear Embedding, Epoch 90



(c) MegaDescriptor N.L. Embedding, Epoch 10



(d) MegaDescriptor N.L. Embedding, Epoch 90

Figure 6: Visualization of triplet embedding spaces for DINO and MegaDescriptor models at early (Epoch 10) and late (Epoch 90) stages of training.

vectors. It is also possible that the implicit assumptions in the dataset split had a substantial impact on the distribution of outputs. We use 60% of the dataset as "known" in our experiments, but it is possible that increasing the set of known images would lead to a different optimal hyperparameter score. While we try to have an offline approximation of the private leaderboard for development, it has proven challenging to find a suitable proxy for local development.

During development, we also found that setting up the triplet loss was particularly tricky. Although we employed a train-validation-test split for our thresholding scheme, we opted for a distinct train-validation split for the triplet learning pipeline. While we could continue to learn and reduce the triplet loss, it also indicated that we were overfitting the geometry of the training dataset. When we accounted for our split, we had a better chance of determining which parameterization of the triplet layer was most effective for us.

After refining our dataset split using our triplet pipeline, we found that reshaping DINOv2 was significantly harder than MegaDescriptor. DINOv2 is a general-purpose feature extractor, and while it performs well on generalized tasks, it is not optimized for fine-grained classification in this context. We did not find a mining strategy or parameterization of the embedding head that would lower the triplet loss on the validation set. Since the training and validation sets are non-overlapping, we can only indirectly influence the triplet scores of the validation set by reshaping the manifold through triplet mining on the training set. Individuals are not clustered as tightly on the DINOv2 manifold, and it becomes difficult to move in a direction that is isomorphic between training and validation sets. The MegaDescriptor triplet learning works comparably well, increasing the model's performance on the task by 0.13, compared to a 0.03 improvement on DINOv2 triplet learning. This could be because the MegaDescriptor model employs a metric objective that combines ArcFace and Triplet Loss, resulting in a minor domain shift overall.

Finally, we observe differences in the triplet learning in figure 6. We use PaCMAP, a graph-theoretic embedding method that takes into account both local and global geometry in shape. The large cluster thus signifies a cloud of points that are challenging to disambiguate. We see that over the process of triplet learning, the DINO model learns to disambiguate clusters of individuals. In the MegaDescriptor model, there is already a large number of clusters. Note that the number of clusters is visually larger than in the DINO model, and this roughly correlates with performance in the final task.

6. Future Work

Reflecting on the poor performance achieved using DINO, it would likely have performed better if we had supplemented training with the WildlifeReID-10K dataset [8]. Previous experience with similarly designed pipelines has made us aware of the data-hungry nature of the triplet learning paradigm. Due to the small size of the provided dataset and the limitations imposed on our triplet mining implementation, the number of unique triplets used during training was likely insufficient. Using the WildlifeReID-10K dataset in conjunction with the provided data would likely alleviate these issues.

Table 5
Comparison of Model Parameters for ViT Backbones

Model Name	Parameters (Millions)
`facebook/dinov2-small`	21
`facebook/dinov2-base`	86
`facebook/dinov2-large`	304
`facebook/dinov2-giant`	1,135
`BVRA/MegaDescriptor-T-224`	28.3
`BVRA/MegaDescriptor-S-224`	49.6
`BVRA/MegaDescriptor-B-224`	109.1
`BVRA/MegaDescriptor-L-224`	228.6
`BVRA/MegaDescriptor-L-384`	228.8
`BVRA/MegaDescriptor-T-CNN-288`	12.2

Additionally, we would like to experiment with a larger number of backbones to ensure that results are comparable. We enumerate a list of models in Table 5 that would provide a concrete starting point for future experiments.

7. Conclusions

We develop a transfer learning solution for the AnimalCLEF 2025 competition, leveraging inherent visual knowledge encoded in vision transformers. We define the nearest neighbor classifier that can tackle the open-set nature of the competition through a rigorously defined thresholding procedure. While our solution ranks higher than the baseline MegaDescriptor solution, there are limitations to our methods that should be addressed by augmenting them with a larger individual dataset and more careful hyperparameter tuning. Code for this paper can be found at https://github.com/dsgt-arc/animalclef-2025.

Acknowledgements

We thank the Data Science at Georgia Tech (DS@GT) CLEF competition group for their support. This research was supported in part through research cyberinfrastructure resources and services provided by the Partnership for an Advanced Computing Environment (PACE) at the Georgia Institute of Technology, Atlanta, Georgia, USA [22].

Declaration on Generative AI

During the preparation of this work, the authors used Gemini Pro and Grammarly in order to: Abstract drafting, formatting assistance, grammar and spelling check. After using these tools/services, the authors reviewed and edited the content as needed and takes full responsibility for the publication's content.

References

[1] L. Adam, L. Picek, V. Čermák, K. Papafitsoros, Animalclef 2025, 2025. URL: https://www.imageclef.org/AnimalCLEF2025.

[2] V. Čermák, L. Picek, L. Adam, K. Papafitsoros, Wildlifedatasets: An open-source toolkit for animal re-identification, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2024, pp. 5953–5963.

[3] L. Laskowski, R. Sawahn, M. Schall, D. Wasmuht, M. Bermejo, G. de Melo, Gorillavision – open-set re-identification of wild gorillas, 2023. URL: https://inf-cv.uni-jena.de/wordpress/wp-content/uploads/2023/09/Talk-12-Maximilian-Schall.pdf.

[4] V. Miele, G. Dussert, B. Spataro, S. Chamaillé-Jammes, D. Allainé, C. Bonenfant, Revisiting animal photo-identification using deep metric learning and network analysis, Methods in Ecology and Evolution 12 (2021) 863–873.

[5] S. Li, J. Li, W. Lin, H. Tang, Atrw: A benchmark for amur tiger re-identification in the wild, 2019. URL: http://arxiv.org/abs/1906.05586.

[6] J. B. Haurum, A. Karpova, M. Pedersen, S. H. Bengtson, T. B. Moeslund, Re-identification of zebrafish using metric learning, in: 2020 IEEE Winter Applications of Computer Vision Workshops (WACVW), IEEE, Snowmass Village, CO, USA, 2020, pp. 1–11. URL: https://ieeexplore.ieee.org/document/9096922/.

[7] J. Gao, T. Burghardt, W. Andrew, A. W. Dowsey, N. W. Campbell, Towards self-supervision for video identification of individual holstein-friesian cattle: The cows2021 dataset, 2021. URL: https://arxiv.org/abs/2105.01938.

[8] L. Adam, V. Čermák, K. Papafitsoros, L. Picek, Wildlifereid-10k: Wildlife re-identification dataset with 10k individual animals, 2025. URL: https://arxiv.org/abs/2406.09211. arXiv:2406.09211.

[9] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby, An image is worth 16x16 words: Trans-

formers for image recognition at scale, in: International Conference on Learning Representations, 2021. URL: https://openreview.net/forum?id=YicbFdNTTy.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, I. Polosukhin, Attention is all you need, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems, volume 30, Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[11] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin transformer: Hierarchical vision transformer using shifted windows, in: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 9992–10002. doi:10.1109/ICCV48922.2021.00986.

[12] S. He, H. Luo, P. Wang, F. Wang, H. Li, W. Jiang, Transreid: Transformer-based object re-identification, in: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 15013–15022.

[13] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al., Dinov2: Learning robust visual features without supervision, arXiv preprint arXiv:2304.07193 (2023).

[14] A. Miyaguchi, M. Gustineli, A. Fischer, R. Lundqvist, Transfer learning with self-supervised vision transformers for snake identification, 2024. URL: https://arxiv.org/abs/2407.06178.

[15] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, I. Sutskever, Learning transferable visual models from natural language supervision, in: International Conference on Machine Learning, 2021. URL: https://api.semanticscholar.org/CorpusID:231591445.

[16] Y. Wu, D. Zhao, J. Zhang, Y. S. Koh, An individual identity-driven framework for animal re-identification, 2024. URL: https://arxiv.org/abs/2410.22927.

[17] F. Schroff, D. Kalenichenko, J. Philbin, Facenet: A unified embedding for face recognition and clustering., in: CVPR, IEEE Computer Society, 2015, pp. 815–823. URL: http://dblp.uni-trier.de/db/conf/cvpr/cvpr2015.html#SchroffKP15.

[18] J. Deng, J. Guo, N. Xue, S. Zafeiriou, Arcface: Additive angular margin loss for deep face recognition, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 4685–4694. doi:10.1109/CVPR.2019.00482.

[19] Y. Wang, H. Huang, C. Rudin, Y. Shaposhnik, Understanding how dimension reduction tools work: An empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization, Journal of Machine Learning Research 22 (2021) 1–73. URL: http://jmlr.org/papers/v22/20-1061.html.

[20] M. Douze, A. Guzhva, C. Deng, J. Johnson, G. Szilvasy, P.-E. Mazaré, M. Lomeli, L. Hosseini, H. Jégou, The faiss library (2024). arXiv:2401.08281.

[21] A. Kusupati, G. Bhatt, A. Rege, M. Wallingford, A. Sinha, V. Ramanujan, W. Howard-Snyder, K. Chen, S. Kakade, P. Jain, A. Farhadi, Matryoshka representation learning, in: Proceedings of the 36th International Conference on Neural Information Processing Systems, NIPS '22, Curran Associates Inc., Red Hook, NY, USA, 2022.

[22] PACE, Partnership for an Advanced Computing Environment (PACE), 2017. URL: http://www.pace.gatech.edu.