Coat Based Identification Of Holstein-Friesian Cattle Using Deep Metric Learning

Asheesh Sharma

24 December 2021

https://github.com/Asheeshkrsharma/Identification-OpenCows

Introduction

Identifying individuals in a group of animals is an active research topic as it paves the way to behavioural studies for lameness and illness assessment. In this project, we independently replicated the deep metric learning technique proposed by Andrew et al. (2021) to train a Residual Convolutional Neural Network (Resnet) on a dataset of Holstein-Friesians. The technique relies on learning a 128-dimensional latent feature space that, when used to fit simple algorithms such as K-Nearest Neighbours, can perform open-set identification. We also implemented the proposed novel hybrid loss function and compared its performance with traditional losses used in identification tasks.

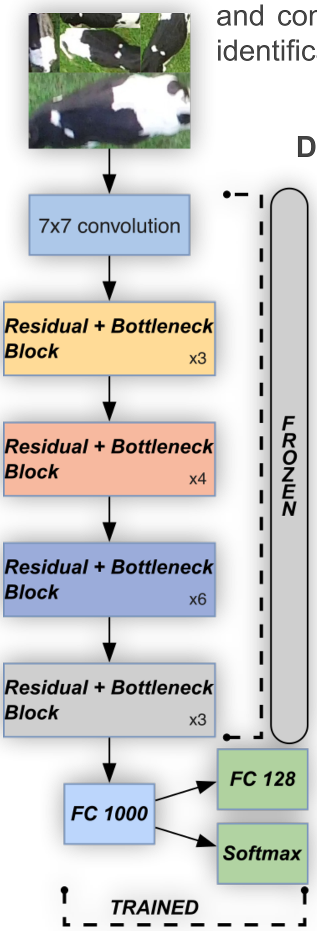

Fig 2: The block diagram of a Resnet-50.

Methodology

Dataset: To identify individual cows in a herd, we used the OpenCows2020 dataset collected at the Wyndhurst farm, University of Bristol, (U.K.), released by the authors. The dataset consists of 4,736 regions of interest, extracted from RGB images of 46 individuals, as shown in **Fig. 2**.

Metric learning and Open-set identification: Many real-world problems require an identification algorithm to work with previously unseen examples. In their study, Andrew et al. (2021) trained the Resnet-50 model (**Fig. 2**) to learn a latent space that clusters images of identical individuals with good separation from others. The fully connected layer "FC 128" outputs a 128-dimensional embedding, which can be used as a metric of input similarity in clustering and classification algorithms such as kNN. In this study, we assessed the generalisation capability of the model by varying the percentage of unseen examples and monitoring the convergence. For instance, in **Fig. 1**, we withhold 25% examples; therefore, the distribution is 20% open.

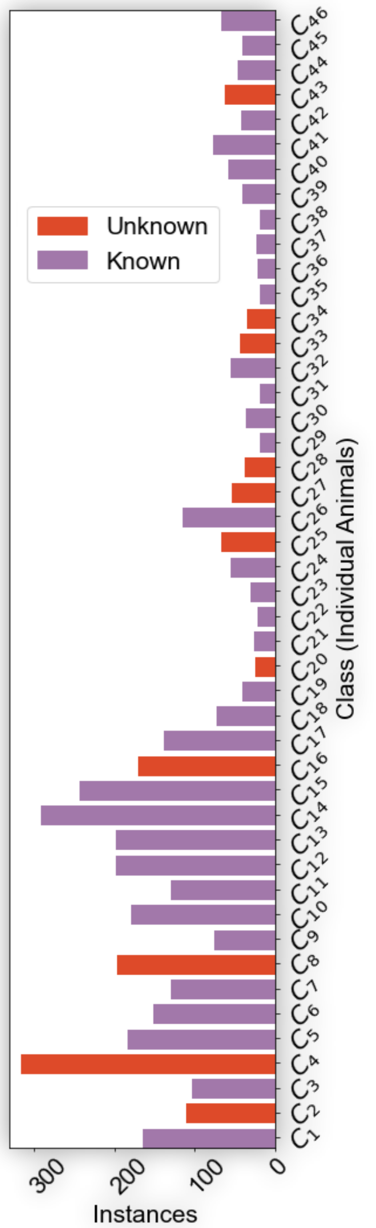

Fig 1: Images per cow used for training on open-set task (in Purple, 75%). The unknowns (in Red, 25%) were withheld.

Training setup: The Resnet-50 variant has four stages of residual blocks (see **Fig. 2**). Fine-tuning such networks is computationally expensive and can lead to sub-optimal convergence without prior hyper-parameterisation. Contrary to the author's approach of fine-tuning the entire model at a slower learning rate, we chose *not* to update the residual layers (see "Frozen", **Fig. 2)** and trained just the last three fully-connected layers (**Fig. 2**, "Trained"). This allowed us to train the network faster. Apart from the "Frozen" strategy, we chose a higher learning rate of $1e^{-1}$, and a smaller weight decay of $1e^{-7}$. Other parameters were kept same, including 100 training iterations and a momentum of 0. An exhaustive list of all hyper-parameters has been included in the accompanying Jupyter notebook.

Loss functions: The triplet loss incentivises the model for decreasing the distance $d(x_a, x_p)$ between a positive image and an anchor, while penalising for reducing the distance $d(x_a, x_n)$ to a negative X_n. A training iteration computes the loss on the 128-dimensional Resnet embeddings x_p, x_a, and x_n using **Eq. 1.** The margin parameter α prevents the loss function from reaching negative values, when $d(x_a, x_p) < d(x_a, x_n)$. In practice, the margin value needs to be tuned by hyper-parameterisation. The authors proposed a hybrid loss which combines Reciprocal Triplet Loss $\mathbb{L}_{RTL}$ (**Eq. 2**), that remediates the negative loss issue by replacing $d(x_a, x_n)$ by its reciprocal, and adding a softmax term (see **Eq. 3**). In this study, we investigated the quantitative and qualitative advantage of the proposed loss function $\mathbb{L}_{SoftMax+RTL}$ (**Eq. 3**) over $\mathbb{L}_{SoftMax+TL}$ loss function which is commonly used in deep metric learning tasks (**Eq. 4**).

$$\mathbb{L}_{TL} = max(0,\ d(x_a, x_p) - d(x_a, x_n) + \alpha) \quad \cdots (1)$$

$$\mathbb{L}_{RTL} = d(x_a, x_p) + \frac{1}{d(x_a, x_n)} \quad \cdots (2)$$

$$\mathbb{L}_{SoftMax+RTL} = \mathbb{L}_{SoftMax} + \lambda \cdot \mathbb{L}_{RTL}$$

$$\text{where } \mathbb{L}_{SoftMax} = -log\left(\frac{e^{x_{class}}}{\sum_i e^{xi}}\right) \quad \cdots (3)$$

$$\mathbb{L}_{SoftMax+TL} = \mathbb{L}_{SoftMax} + \lambda \cdot \mathbb{L}_{TL} \quad \cdots (4)$$

Results and Conclusion

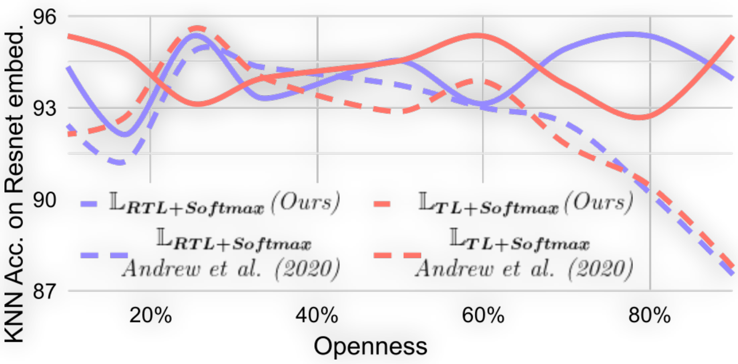

***Fig 3**: Convergence of our models compared to Andrew et al. (2021), with varying percentage of images withheld (or openness).*

Quantitative assessment: Fig. 3 shows the accuracy of the K-nearest neighbour algorithm fitted on the Resnet-50 model embeddings for a varying percentage of withheld image data (or openness) for $\mathbb{L}_{SoftMax+TL}$ and $\mathbb{L}_{SoftMax+RTL}$ loss functions. It can be observed that our models converged better than the authors' on larger openness (from 40 to 100%). However, our results closely agree with the authors for an openness below 60%. This result can be attributed to our strategy of freezing the residual blocks (**Fig. 2**). Since the blocks are not updated, the knowledge learnt by the image-net weights is better retained, while the actual learning happens only in the final fully connected layers. Since Andrew et al. (2021) fine-tuned the entire network, the sharp decline can be attributed to "early stopping", i.e. the model is trained for a smaller number of iterations than ideal. Finally, examining the repository by the authors revealed a flaw in the OpenCows2020.py Dataloader class (see lines 183-194). The Imagnet weights used by the authors (and us) for fine-tuning the model requires mean normalisation, as discussed in the PyTorch documentation.

Qualitative assessment: To visualise the kNN boundaries, the 128-dimensional embeddings from the test-train split were jointly reduced to two dimensions with t-SNE for a consistent latent space. A kNN was fitted on the reduced train embeddings. **Fig 4.** shows the kNN decision boundaries along with t-SNE reduced embedding. It can be observed that embeddings obtained from $\mathbb{L}_{SoftMax+TL}$ are tightly clustered, which can potentially crowd the latent space as the number of individuals increases (**Fig. 4a**). $\mathbb{L}_{SoftMax+RTL}$ on the other hand has relaxed boundaries and evenly spread samples within the clusters (**Fig. 4b**). Therefore, **Fig 4.** demonstrates the advantage of the proposed loss function $\mathbb{L}_{SoftMax+RTL}$ over $\mathbb{L}_{SoftMax+TL}$.

Conclusion: We obtained better model convergence by modifying the training method and fixing the mean-normalisation. We achieved a higher accuracy of 98.1% ($\mathbb{L}_{SoftMax+RTL}$) and 97.7% ($\mathbb{L}_{SoftMax+TL}$) for the 67-33% train-test split by training the model for 200 iterations, showing the signs of early stopping (authors 94.36% and 94.06% correspondingly). However, we found that including the softmax loss term with the reciprocal triplet loss improves the latent space.

Reference: Andrew, W. *et al.* Visual identification of individual Holstein-Friesian cattle via deep metric learning. *Comput Electron Agr* **185**, 106133 (2021). DOI:https://doi.org/10.1016/j.compag.2021.106133

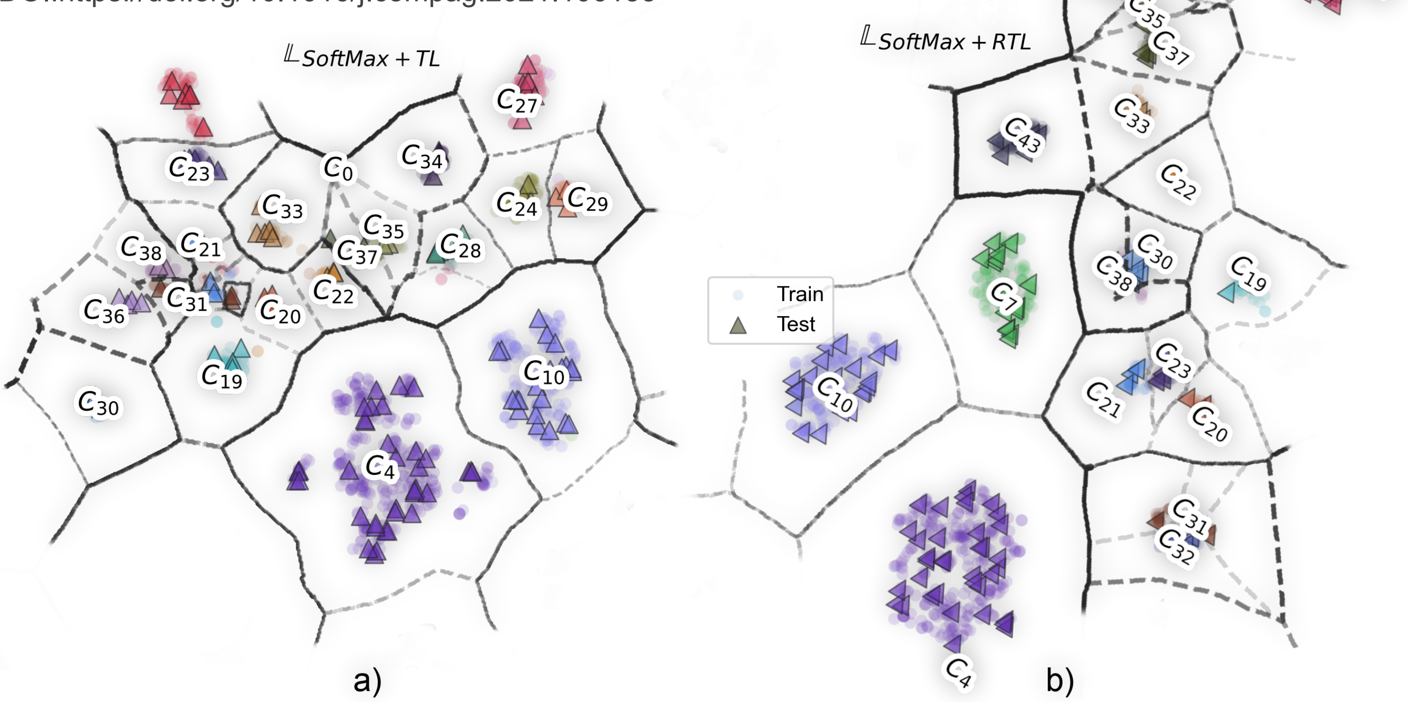

***Fig 4**: Shows t-SNE reduced embeddings for individual cows ($C_1 \cdots C_{46}$). a) $\mathbb{L}_{SoftMax+TL}$ embeddings show tighter clusters and overlapping decision boundaries (in dashed black), b) $\mathbb{L}_{SoftMax+RTL}$ embeddings are evenly spread with clearer decision boundaries.*