Class Anchor Clustering: A Loss for Distance-based Open Set Recognition Supplementary Material

Dimity Miller, Niko Sünderhauf, Michael Milford, Feras Dayoub
Queensland University of Technology, Australian Centre for Robotic Vision
{d24.miller, niko.suenderhauf, michael.milford, feras.dayoub}@qut.edu.au

1. Tuplet loss and Cross-Entropy loss

Given that softmin is

$$\text{softmin}(\mathbf{d})_i = \frac{e^{-\mathbf{d}_i}}{\sum_{k=1}^{N} e^{-\mathbf{d}_k}}, \tag{1}$$

cross-entropy loss applied to the distance vector with softmin can be formulated as

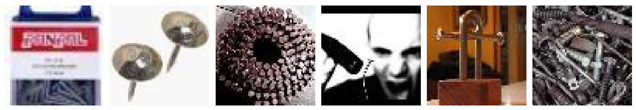
$$\mathcal{L}_{\text{CE}}(\mathbf{x}, y) = -\log(\frac{e^{-\mathbf{d}_y}}{\sum_{k=1}^{N} e^{-\mathbf{d}_k}}). \tag{2}$$

This can be shown as equivalent to our Tuplet loss:

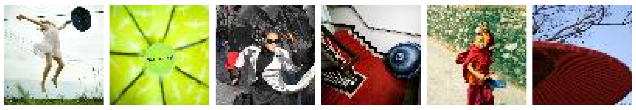
$$\begin{aligned}
\mathcal{L}_{\text{CE}}(\mathbf{x}, y) &= -\log(\frac{e^{-\mathbf{d}_y}}{\sum_{k=1}^{N} e^{-\mathbf{d}_k}}) && (3)\\
&= -\log(\frac{e^{-\mathbf{d}_y}}{e^{-\mathbf{d}_y} + \sum_{k\neq y}^{N} e^{-\mathbf{d}_k}}) && (4)\\
&= \log(\frac{e^{-\mathbf{d}_y} + \sum_{k\neq y}^{N} e^{-\mathbf{d}_k}}{e^{-\mathbf{d}_y}}) && (5)\\
&= \log(1 + \frac{\sum_{k\neq y}^{N} e^{-\mathbf{d}_k}}{e^{-\mathbf{d}_y}}) && (6)\\
&= \log(1 + e^{\mathbf{d}_y}\sum_{k\neq y}^{N} e^{-\mathbf{d}_k}) && (7)\\
&= \log(1 + \sum_{k\neq y}^{N} e^{\mathbf{d}_y} e^{-\mathbf{d}_k}) && (8)\\
&= \log(1 + \sum_{k\neq y}^{N} e^{\mathbf{d}_y - \mathbf{d}_k}) && (9)\\
&= \mathcal{L}_{\text{T}}(\mathbf{x}, y). && (10)
\end{aligned}$$

2. TinyImageNet Examples

Examples of the large visual variations within a single class and presence of features unrelated to the class can be seen in Figure 1.



(a) 'Nail' Class



(b) 'Umbrella' Class

Figure 1: Examples of the large intra-class visual variations present in TinyImageNet. While 'nail' and 'umbrella' might typically be considered unrelated classes, they can both contain humans in the background.

Examples of visually and semantically related classes in TinyImagenet, such as six different breeds of dogs, are shown in Figure 2.

Figure 2: Images from six different, but visually and semantically related, classes in Tiny ImageNet: chihuahua, german shepherd, golden retriever, labrador retriever, standard poodle and yorkshire terrier.

3. Implementation Details

3.1. Network Architecture

The base network architecture f was consistent with the architecture established by [13]. It consists of 9 convolutional layers with batch norm after every layer and dropout after every 3 layers, followed by a fully-connected layer. It is included in the code submitted with this supplementary material, which will also be made public.

3.2. Training Details

The learning rate (lr), number of training epochs and network dropout rate for each dataset is shown in Table 1.

Dataset	LR	Epochs		Dropout
		Anchored	Learnt	
MNIST	0.01	35	70	0.2
	0.001	35	70	0.2
SVHN	0.01	50	75	0.2
	0.001	35	75	0.2
CIFAR*	0.01	150	250	0.2
	0.001	25	150	0.2
TinyImNet	0.01	500	700	0.3
	0.001	300	500	0.3

Table 1: Training details for each dataset for anchored class centres and learnt class centres.

4. Experimental Details for Performance at FP Operating Points

We follow the experimental protocol established by [7]. We train a ResNet-18 on all classes in CIFAR10. Unknown datasets include SVHN and a subset of CIFAR100. The unknown subset of CIFAR100 includes all data from the following superclasses of CIFAR100: large man-made outdoor things, large natural outdoor scenes, large omnivores and herbivores, medium-size mammals, non-insect invertebrates, people, reptiles, small mammals and trees. Note that [7] uses another subset of CIFAR100 as 'known unknowns' during training, with data from the following superclasses: aquatic mammals, fish, flowers, food containers, fruit and vegetables, household electrical devices, household furniture, insects and large carnivores. We do not use any 'known unknown' data during our training.

5. Learning Class Centres with CAC

When *learning* class centres [26] (rather than using our proposed anchored class centres), CAC can become unstable and unable to converge. When the network is first initialised and created, inputs do not exhibit any class-specific clustering behaviour, and thus meaningful class centres cannot be learnt from the input positions. This is not a problem when using our anchored class centres, as we have already dictated the fixed position the inputs should cluster. Center loss [26] similarly faces this issue, and additionally uses cross-entropy loss to aid learning the class centres.

Cross-entropy loss forces inputs to fall into specified regions of the logit space (high activations in the ground truth dimension and low activations in all others), thus starting the class clustering behaviour and making input positions less noisy for learning class centres. Therefore, to provide comparison results with learnt class centres [24], we were required to add cross-entropy loss $\mathcal{L}_{\text{CE}}$ to CAC loss. The final loss used was then:

$$\mathcal{L}_{\text{Final}}(\mathbf{x}, y) = \mathcal{L}_{\text{CAC}}(\mathbf{x}, y) + \mathcal{L}_{\text{CE}}(\mathbf{x}, y) \tag{11}$$

where

$$\mathcal{L}_{\text{CE}}(\mathbf{x}, y) = -\log\left(\frac{e^{\mathbf{z}_y}}{\sum_{k=1}^{N} e^{\mathbf{z}_k}}\right). \tag{12}$$