Cam-softmax for discriminative deep feature learning

Suveges, Tamas; McKenna, Stephen

Published in:
Proceedings of the 25th International Conference on Pattern Recognition

DOI:
10.1109/ICPR48806.2021.9412895

Publication date:
2021

Document Version
Peer reviewed version

Cam-softmax for discriminative deep feature learning

Tamas Suveges
School of Science and Engineering
University of Dundee
Dundee, Scotland, U.K.
email: t.suveges@dundee.ac.uk

Stephen McKenna
School of Science and Engineering
University of Dundee
Dundee, Scotland, U.K.
ORCID: 0000-0003-0530-2035

***Abstract*—Deep convolutional neural networks are widely used to learn feature spaces for image classification tasks. We propose *cam-softmax*, a generalisation of the final layer activations and softmax function, that encourages deep feature spaces to exhibit high intra-class compactness and high inter-class separability. We provide an algorithm to automatically adapt the method's main hyperparameter so that it gradually diverges from the standard activations and softmax method during training. We report experiments using CASIA-Webface, LFW, and YTF face datasets demonstrating that cam-softmax leads to representations well suited to open-set face recognition and face pair matching. Furthermore, we provide empirical evidence that cam-softmax provides some robustness to class labelling errors in training data, making it of potential use for deep learning from large datasets with poorly verified labels.**

I. Introduction

Deep convolutional neural networks (DCNNs), that learn to map images to abstract representations using local compositional structure in their many layers of neurons, have dominated advances in image classification tasks in recent years. Deep representation learning can yield feature spaces useful for discriminating between classes not directly represented in the training data. Ideally, a deep feature space should exhibit high intra-class compactness and high inter-class separability. We propose a modification to the final layer activations and softmax function that encourages such feature spaces to be learned.

There are many applications for which acquiring reliably labelled data sets for deep learning is challenging or prohibitively expensive; the scale of the datasets used for deep learning makes quality control challenging, resulting in datasets with large numbers of incorrectly labelled examples [1]. The method we propose exhibits robustness to such labelling errors, facilitating learning from large unreliably labelled datasets. It does so by automatically adapting a parameter that controls attenuation of learning gradients for examples likely to be mislabelled, and amplification of gradients for examples likely to be correctly labelled.

After describing and discussing the proposed method, which we call *cam-softmax*, we report experiments on several tasks and datasets. We compare to Sphereface [2] and AM-softmax[3] because of their competitive performance and similarities in approach. Firstly, we evaluate performance on two pairwise face matching benchmarks, LFW and YTF, and obtain a substantial performance boost over the softmax activation, reaching performance in line with the state-of-the-art in direct comparisons. Secondly, given that performance on the LFW pairwise matching benchmark is becoming saturated, we follow the more challenging open-set protocol described in [4] and obtain performance gains compared to standard softmax, am-softmax [3], and sphereface [2]. Thirdly, we demonstrate cam-softmax's robustness to labelling errors using the face datasets and Fashion-MNIST [5] dataset with deliberately corrupted class labels. We investigate its ability to preferentially attenuate learning gradients for mislabelled examples.

In summary, the main contributions of this paper are as follows.

1) A heuristic modification to the final layer activations and softmax function that (i) leads to learned deep feature spaces exhibiting high intra-class compactness and inter-class separability, and (ii) facilitates learning from noisily labelled training data.
2) A method to automatically tune the main hyperparameter of the method, avoiding the need for time-consuming parameter search.
3) Face recognition experiments on the LFW and YTF pairwise benchmarks as well as on the LFW open-set benchmark, with direct comparison of cam-softmax, Sphereface [2], AMSoftmax [3], and the baseline standard activations with softmax.
4) Experiments, using face matching, CIFAR-10 and Fashion-MNIST datasets, investigating the effect of training set label noise on cam-softmax and those other methods.

II. Related Work

DCNNs are now widely used for learning discriminative image representations, achieving state-of-the-art results on many tasks such as face recognition [2, 6] and object recognition [7]. Deep representation learning has been used for scenarios in

This research received funding from the UK Engineering and Physical Sciences Research Council (EPSRC grant EP/N014278/1).

We would like to thank members of the CVIP cluster and the ACE-LP project for helpful discussions and feedback.

which classes not present in training data need to be discriminated, important examples being verification on previously unseen classes, and open-set recognition. Methods designed to learn generalisable representations that encourage intra-class compactness and inter-class separability, even for previously unseen classes, include those that introduce an extra loss term directly on the deep features [6, 8, 9, 10], and those that modify the softmax activation function [2, 3, 11, 12, 13, 14]. We note in passing that stochastic modifications to activations have been proposed such as the noisy softmax used to avoid early saturation [15]. Here we focus on deterministic modifications.

Methods that introduce an extra loss term can create difficulties during training. Multiple loss terms need careful weighting to balance their contributions correctly. Facenet [6] is trained on triplets of examples, however the selection of these is not straightforward. The exponential relationship between dataset size and the number of possible triplets necessitates a carefully defined policy for triplet selection. Centre loss [8] and contrastive loss [10] augment cross-entropy loss with terms computed directly on feature space. Centre loss requires a mechanism to track every class centre in feature space as the model is trained. Contrastive loss considers pairs of training examples, encouraging their feature represenations to be similar if they are in the same class and dissimilar otherwise.

Large margin softmax loss modifies softmax by introducing an angular margin between classes and uses cosine similarity [11]. SphereFace [2] takes this idea further by incorporating weight vector normalization. However, both use an activation function that has multiple discontinuities in its first derivative and multiple zero-crossing in its second derivative which leads to convergence problems. This is eased somewhat by using a linear combination of the standard softmax with their modified activation; a free parameter λ controls the relative weights of these two terms and is kept high enough that the standard softmax dominates. However, even after the introduction of this weighting parameter, the authors' published code normalizes the gradient to further tackle the issue. We would like to highlight that our method is considerably less involved and easier to implement while not suffering from convergence problems. Furthermore, whereas the margin parameter in [2] is unbounded and constrained to be integer, the c parameter in our method is bounded and continuous, enabling more precise control.

Other studies introduce additive margins to the softmax activation function [3, 12]. This is equivalent to adding a constant bias to the output of the last fully connected layer that represents the expected label. Additive margin shifts the range of the cosine function by a constant. Unlike cam-softmax this lacks the ability to control the magnitude of the gradient based on the appropriateness of the extracted deep features.

Methods modifying the softmax activation function [2, 3, 11, 12, 13, 14] commonly introduce a margin parameter. Some studies [2, 11] introduce an extra parameter to control the rate with which the margin parameter is changed towards its final value during training. Performing grid-search on the final value and the rate of change is computationally expensive. In III-D we suggest a dynamic approach to reduce the number of free parameters. In our method, the main parameter is bounded and automatically adjusted during training without requiring its final value to be determined beforehand.

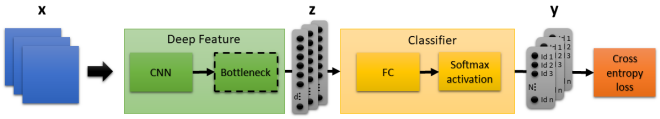


Fig. 1. A deep neural network trained to map input images to class labels via a hidden representation, $\boldsymbol{z}$. The bottleneck and FC components are fully connected layers

III. CAM-SOFTMAX

A. Softmax and cross-entropy

Fig. 1 shows a typical deep image classifier set-up; a network maps input images $\boldsymbol{x}$ to deep feature vectors $\boldsymbol{z} \in \mathcal{R}^d$, and a final layer performs classification in this feature space. Given a training set of N examples, where $t^{(n)}$ is the target class for the n^{th} example, the cross-entropy loss is often used:

$$L = -\frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K}\mathbf{1}(t^{(n)}=k)\log y_k \tag{1}$$

Here y_k is the probability of class k, computed using a softmax activation function:

$$y_k = \frac{e^{\boldsymbol{w}_k^T\boldsymbol{z}}}{\sum_j e^{\boldsymbol{w}_j^T\boldsymbol{z}}} \tag{2}$$

If both the deep feature vector and the weight vectors are normalised, a neuron's activation is the cosine of the angle, θ, that its weight vector makes with the feature vector, i.e., $\boldsymbol{w}_j^T\boldsymbol{z} = \cos\theta_j$. In what follows we will assume these vectors have been normalised.

B. Attenuating the target neuron's activation

We aim to modify computation of the final layer softmax and activations so that the learned deep feature representation, $\boldsymbol{z}$, will exhibit better intra-class compactness and inter-class separability. This is achieved by encouraging reduced angles between $\boldsymbol{z}$ and target neuron weight vectors. The modification we propose has the target class neuron compute its activation differently from other output neurons during training. A softmax activation function (3) is applied to activations computed according to (4) where the neuron for the target class computes its activation using (5). Here t denotes the target for the current example (and superscript n has been omitted).

$$y_k = \frac{e^{a_k}}{\sum_j e^{a_j}} \tag{3}$$

$$a_k = \mathbf{1}(k\neq t)(\boldsymbol{w}_k^T\boldsymbol{z}) + \mathbf{1}(k=t)(f(\boldsymbol{z},\boldsymbol{w}_k;c)) \tag{4}$$

$$f(\boldsymbol{z},\boldsymbol{w}_k;c) = \frac{(\cos\theta_k+1)^{g(c)}}{2^{(g(c)-1)}} - 1 \tag{5}$$

The parameter $c \in [0, \pi]$ can be considered to be a free parameter. For convenience we parameterise c such that $f(\boldsymbol{z}, \boldsymbol{w}_k; c) = 0$ when $\theta_k = c$,

$$g(c) \;=\; -\frac{1}{\log_2(\cos c \;+\; 1) - 1} \tag{6}$$

In particular, setting $c = \frac{\pi}{2}$ recovers the standard activations normally used with softmax (Equation (2)). Setting $c < \frac{\pi}{2}$ reduces the target neuron's activation and therefore encourages the network to reduce the angle between the feature vector and the neuron's weight vector. Fig. 2(a) plots the activation as a function of θ_k for three different values of c. Note that these activations are always bounded by 1 and -1.

Equations (7) and (8) give partial derivatives required for backprop for the case $k = t$.

$$\frac{\partial a_k}{\partial \boldsymbol{z}} = g(c)\boldsymbol{w}_k^T(1 - \boldsymbol{z}^2)(\cos\theta_k + 1)^{g(c)-1}\frac{1}{2^{g(c)-1}} + \frac{\boldsymbol{z}(\cos\theta_k + 1)^{g(c)-1} - 2^{g(c)-1}\boldsymbol{z}}{2^{g(c)-1}} \tag{7}$$

$$\frac{\partial a_k}{\partial \boldsymbol{w}} = g(c)\boldsymbol{z}_k^T(1 - \boldsymbol{w}^2)(\cos\theta_k + 1)^{g(c)-1}\frac{1}{2^{g(c)-1}} + \frac{\boldsymbol{w}(\cos\theta_k + 1)^{g(c)-1} - 2^{g(c)-1}\boldsymbol{w}}{2^{g(c)-1}} \tag{8}$$

Fig. 2(b) plots the relationship between the partial derivative $\frac{\partial a_t}{\partial \boldsymbol{w}_t}$ and the angle θ_t for three different values of the parameter c. The derivative has a simple form compared to Sphereface [2] where the second derivative has multiple zero crossings.

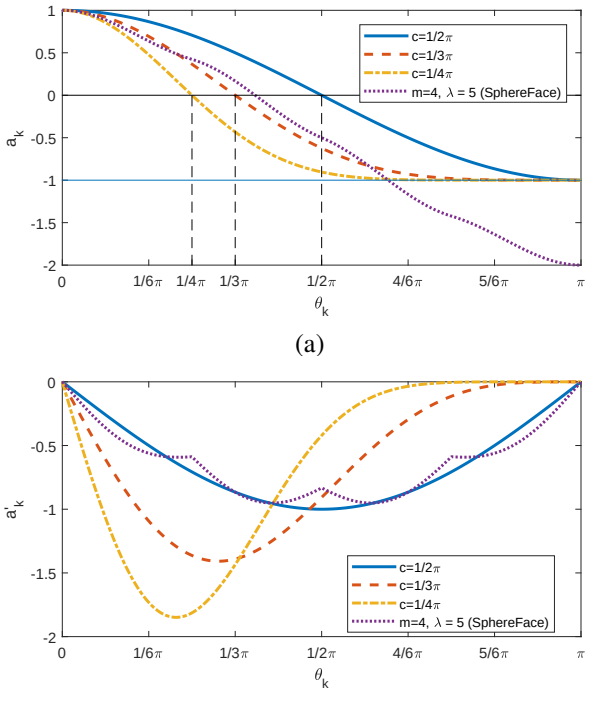
Finally, a scale parameter, s, and an offset parameter, m, can be added in a similar manner to the angular margin proposed in [3]. Equation (9) shows the resulting activation for $k = t$. We refer to the use of softmax with this activation as *cam-softmax*. The use of scale and offset is discussed elsewhere [3, 16, 17, 18]. Accordingly we fix their values to $s = 30$ and $m = 0.25$ and do not study their effect further here.

$$f_{cam}(\boldsymbol{z}, \boldsymbol{w}_k; c) = \frac{s((\cos\theta_k + 1)^{g(c)} - m)}{2^{(g(c)-1)}} - 1 \tag{9}$$

C. Discussion of cam-softmax

Cam-softmax can be used to obtain deep feature space representations with improved intra-class compactness and inter-class separation (see Section IV). When trained on a sufficiently large dataset, the feature space then provides a good representation for discriminating between previously unseen classes from the same domain. As can be seen in Fig. 2(a), reducing c below $\pi/2$ reduces the activation of the target neuron, decreasing the probability computed by softmax for the target class. Achieving a low loss then requires the use a feature space in which feature vectors for the target class are concentrated closer to the target neuron's weight vector.

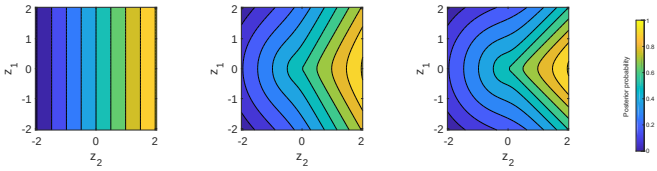
Consider a toy setting in which $\boldsymbol{z}$ is two-dimensional ($d = 2$) and there are just two output neurons corresponding to two classes. Fig. 3(a) shows a function that could be computed by



(a)

(b)

Fig. 2. (a) Target activation a_k versus θ_k and (b) derivative of a_k with respect to θ_k, for three different values of c. (The activation and its derivative from [2, 11] are also shown for $m = 4$, $\lambda = 5$).



(a) (b) (c)

Fig. 3. The effect of c in the case of two classes and two-dimensional feature space. A function computed by a neuron with (a) a standard softmax activation ($c = \frac{\pi}{2}$), (b) and (c) a modified softmax ($c = \frac{\pi}{3}$ and $c = \frac{\pi}{4}$).

one of these output neurons with standard softmax activations. Notice that the decision boundary is $\boldsymbol{w}_1^T\boldsymbol{z} - \boldsymbol{w}_2^T\boldsymbol{z} = 0$. The loss function encourages feature vectors for class k to have larger signed distances from their corresponding discriminant (orthogonal to $\boldsymbol{w}_k$) than from other discriminants, since larger distance results in larger posterior probability. Spreading out the features computed for a class in directions orthogonal to $\boldsymbol{w}_k$ will not alter the neuron's output and thus will not affect this neuron's contribution to the loss. However, the contribution of other output neurons to the loss is likely to be affected by such a change. Rather than relying on other class distributions to prevent a class's feature vectors from being unnecessarily spread out, our modification encourages this behaviour more directly. Returning to our two-dimensional example, Figs. 3(b) and 3(c) show functions that could be computed by an output neuron by setting f as suggested in (5) or (9).

Furthermore, cam-softmax exhibits robustness to training set label noise. It can be used to a learn classifier from training data in which some of the training examples have incorrect class labels. Assuming that the model learns increasingly class-compact features during training, training examples with high angular distance between their feature vector $\boldsymbol{z}$ and target weight vector $\boldsymbol{w}$ are likely to be incorrectly labelled (or very hard) examples, especially towards the end of training. These are the examples with large loss. By decreasing c, back-propagation gradients close to correct predictions will tend to be amplified whereas for badly incorrect predictions they will tend to be attenuated. This can be seen in Fig. 2(b) and by examining the derivatives in (8) and (7). Attenuating gradients belonging to hard or mislabelled examples enables the model to converge further on the other examples. Therefore, in order to encourage robustness to mislabelled data, c can by decreased during training.

D. Reducing c during training

We propose a method to automatically reduce the value of parameter c during training that avoids the need to specify its final value. Decreasing c encourages intra-class compactness and inter-class separation in the learned deep feature space. We use the ratio of within-class to between-class distance to decide when to decrease c. We use temporal averages to estimate the within-class and between-class distances during training. Their ratio, R_n, is calculated at time-step n for a window of size N using (10). The parameter c is decreased by λ at time-step n if $R_n = \min\{R_i\}_{i=0}^{i=n}$. (In our experiments, $\lambda = 0.0002$ and $N = 100$).

$$R_n = \frac{\sum_{i=0}^{N} \theta_{t^{n-i},n-i}}{\sum_{i=0}^{N} \frac{1}{K} \sum_{k \neq t}^{K} \theta_{k^{n-i},n-1}} \tag{10}$$

IV. Experiments

A. Training on CASIA-WebFace

The CASIA-WebFace [19] dataset is a widely used publicly available dataset of images for face recognition research. It has almost half a million images of more than ten thousand faces downloaded from the IMDb website. It is commonly used as a training set for the Labeled Faces in the Wild (LFW) benchmark due to its availability and size. We used the CASIA-Webface dataset to train our networks so that our experiments can be repeated by other researchers. We used the network architecture and training protocol presented in [2], implemented in the Torch framework and running with an NVIDIA GeForce GTX 1080 Graphics Card. We reduced the learning rate from 0.1 to 0.0001 over 18 epochs on a logarithmic schedule. We trained for 20 epochs. When we run comparisons with AM-softmax and Sphereface methods, we trained them using our implementation set-up ("our settings") to enable a fair and direct comparison.

Fig. 4 (a) shows the value of parameter c changing automatically during training. Fig. 5 uses the t-SNE visualisation algorithm to illustrate the ability of cam-softmax to increase intra-class compactness. Plotted are eight sampled classes from the training set, and eight face classes not present at all in the training set. In both cases, examples of the same class formed tighter clusters when using cam-softmax.

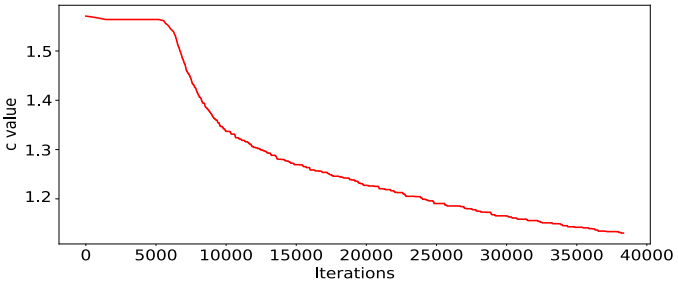


Fig. 4. Change in parameter c during training on CASIA-Webface

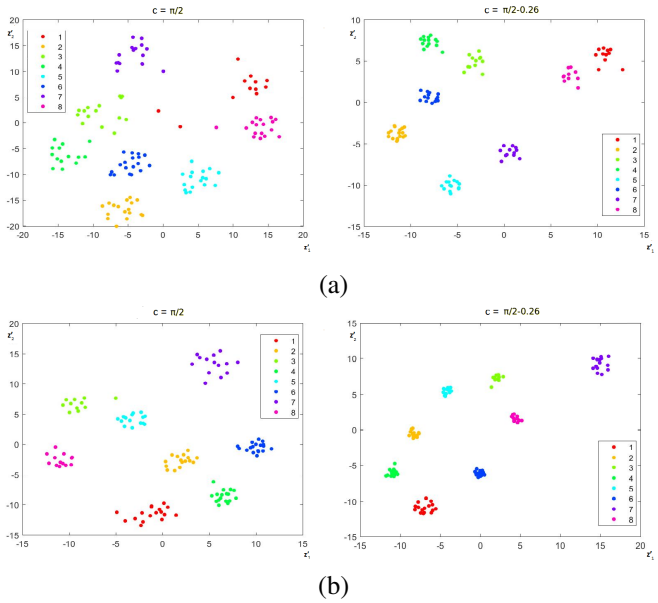


Fig. 5. t-SNE feature space visualisations using standard activations and softmax ($c = \pi/2$) and cam-softmax ($c = \pi/2 - 0.26$). Visualisations in feature space of (a) images of 8 faces sampled from the training set, and (b) images of 8 faces not present in the training set (from the LFW dataset).

B. Face pair matching

We followed the *unrestricted with labelled outside data* protocol suggested by [20] to evaluate face pair matching performance. This protocol randomly selects pairs and defines a 10-fold cross validation, equally balancing the same and different samples in each fold. The aim is to maximise pair matching accuracy on the 10 folds. The pair matching task is to decide whether a pair of previously unseen face images are of the same person or not. Furthermore, we evaluated pair matching on the YTF face video dataset [21]. To match videos in YTF we calculate distances between all face images pairs in two videos and take an average score. There are 6,000 LFW pairs and 10,000 YTF test pairs; in each set half are of the same face and half are of different faces.

Table I gives the average accuracies achieved over 10 folds. We include LFW results reported in the literature but note that

these might not be directly comparable due to various different factors at training and test time. Cam-softmax was more accurate than using standard activations and softmax (equivalent to cam-softmax with c fixed at $\pi/2$). Cam-softmax and AM-softmax had similar accuracies on LFW. Cam-softmax was more accurate than AM-softmax and sphereface on YTF.

C. Open-set face recognition

Real life applications often require discrimination between known and unknown people as well as identification of known identities. For open-set recognition, we follow the protocol proposed by [4]. We summarize the protocol here for completeness. First, we divide the LFW dataset into knowns and unknowns. The set of known identities, K, consists of those with three or more images. The set of unknowns, U, consists of those with only a single image. Identities with two or three images are not considered. Let I be a set of identities in K. We use superscript to indicate identity, i.e., K^i contains images of the ith person, $i \in I$. We further divide the known population into a gallery set $G \in K$ and a probe set $P \in K$ such that the first three images of an identity belong to the gallery $G^i = K^i_{[1,3]}$ and the rest belong to the probe $P^i = K^i_{(3,|K^i|]}$. To calculate similarity $s(\cdot)$ between an identity and a query we take the maximum of the cosine similarities between the query and each image of that identity. For the jth probe of identity i we calculate a ranking measure according to (11). To calculate detection rate for a given rank r we use (12) .

$$rank(p^i_j) = |\{k | s(G^i, p^i_j) <= s(G^k, p^i_j); k \in I\}| \quad (11)$$

$$DR(r) = \frac{|\{p^i_j | rank(p^i_j) <= r; p^i_j \in P\}|}{|P|} \quad (12)$$

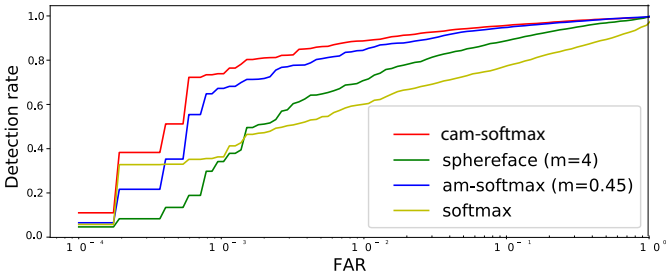
We evaluate identification performance at different False Alarm Rates (FAR) where FAR is a proxy parameter for a similarity threshold τ. First, we calculate similarity scores between known gallery identities I and unknown examples U and sort those in descending order according to (**??**). The threshold τ for a specific FAR is computed as in [4]. Identification rate for a similarity threshold τ is calculated as;

$$\frac{|\{p^i_j | s(G^i, p^i_j) \geq \tau; p^i_j \in P\}|}{|P|} \quad (13)$$

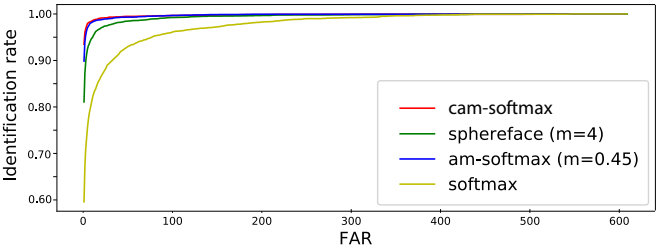
Figs. 6 (a) and (b) show LFW detection and identification rates at different FARs for cam-softmax, am-softmax, sphereface, and softmax. Cam-softmax performed best.

D. Robustness to labelling errors

We built a toy example to confirm that gradients are indeed preferentially reduced for mislabelled training examples. First, we trained a network with standard final layer activations to convergence on the (correctly labelled) CIFAR10 dataset. We then mislabelled half of the dataset uniformly at random and ran one last epoch on the trained model without weight update. We ran this last epoch independently using the standard activations and softmax, and then using cam-softmax instead. During



(a)



(b)

Fig. 6. (a) Detection rates and (b) identification rates on LFW following the open set recognition protocol.

this epoch we recorded layer-wise gradients for the correctly labelled examples and the incorrectly labelled examples. Fig. 7 shows, as expected, that lowering the parameter c decreased the gradients for mislabelled examples and amplified those for correct examples.

We deliberately mislabelled examples in the Fashion-MNIST training set [5]. This dataset contains 10 classes of clothing. In real world scenarios classes with similar visual properties are confused more often, e.g., it is more likely that a pullover gets confused with a coat rather then with a bag. To mimic this behaviour we first trained a linear kernel SVM to obtain a confusion matrix M. For each class c_i we calculated the misclassification probability by summing relevant entries: $\sum_{j \neq i} M_{i,j}$. Classes were then randomly selected proportional to their misclassification probability. Once a class was selected, an example of that class was selected for mislabelling uniformly at random without replacement. To generate a new incorrect label for an example in c_i we use the relevant row of the confusion matrix, M_i as a probability distribution to sample the new label while guaranteeing that the correct label is not assigned. The mislabelling rate is denoted r. For example, exactly 50% of examples are mislabelled when $r = 0.5$ (assuming the data are perfectly labelled to start with). Note that when $r = 1.0$ (i.e., every label is incorrect) the model might learn not to assign the right label, and perform worse than chance at test time. For Fashion-MNIST we used a network with two convolutional layers with kernel size $(2*2)$, and 16 and 32 filters, respectively, trained for 40 epochs. Fig 8 reports accuracies obtained for different mislabelling rates, r. Cam-softmax was the most robust to mislabelling, especially for high r.

Methods	reported (LFW)	our settings (LFW)	our settings (YTF)	$r = 0.5$ (LFW)	$r = 0.5$ (YTF)
Softmax	97.08%	96.50%	86.84%	95.35%	85.66%
AM-Softmax	99.17%	**98.90%**	90.46%	98.53%	**90.12%**
Sphereface	**99.42%**	98.30%	89.80%	97.93%	88.70%
cam-softmax	98.89%	**98.89%**	**90.57%**	**98.71%**	**90.12%**

TABLE I

ACCURACY ON LFW AND YTF PAIR MATCHING. COLUMN 2: ACCURACIES REPORTED IN LITERATURE. COLUMNS 3 AND 4: ACCURACIES UNDER OUR SETTINGS (FOR DIRECT COMPARISONS). COLUMNS 5 AND 6: ACCURACIES WHEN HALF THE TRAINING DATA ARE INCORRECTLY LABELLED.

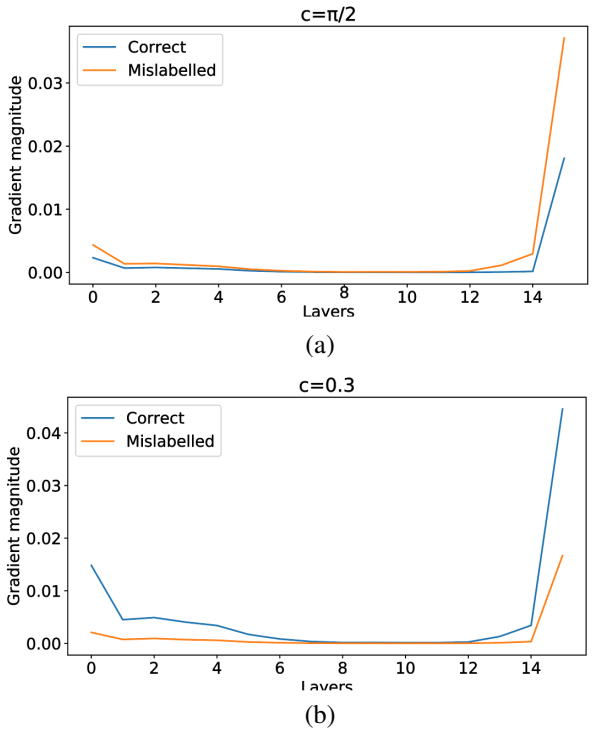


Fig. 7. Average gradient magnitudes of a trained model on correct and mislabelled examples using (a) standard activations and softmax, and (b) *cam*-softmax with $c = 0.3$

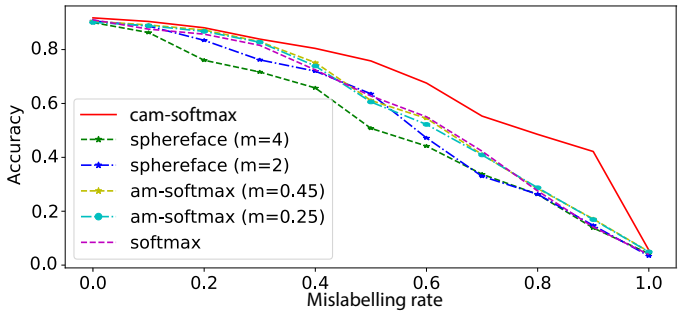


Fig. 8. Accuracy versus mislabelling rate on the Fashion-MNIST dataset

Finally, we randomly mislabelled half of the examples in the CASIA face dataset by selecting new labels uniformly at random from the incorrect labels. We trained the networks on it and tested on the face benchmarks. The last two columns of Table I report the accuracies obtained. Cam-softmax achieved

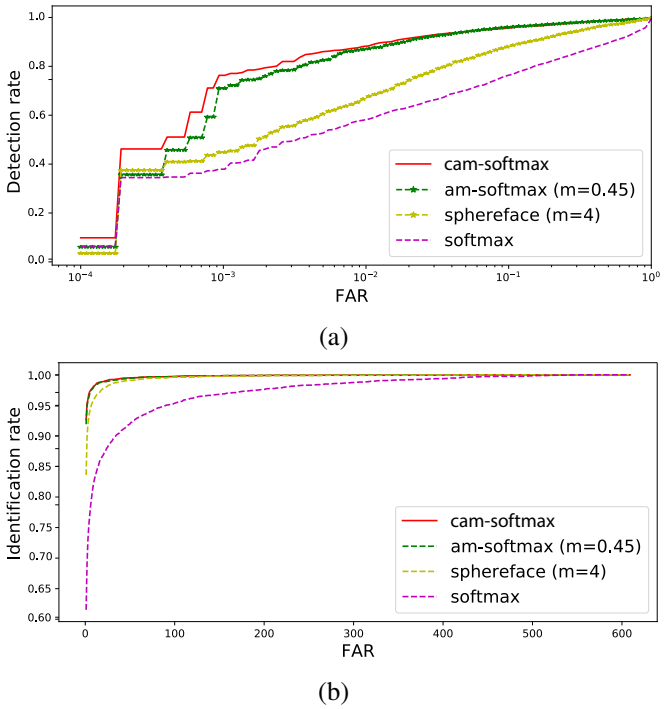


Fig. 9. (a) Detection rates and (b) identification rates for LFW open set recognition when models were trained on CASIA-Webface with 50% mislabelled examples.

the best pair matching performance on LFW, and equal best on YTF. Figs. 9 (a) and (b) show that cam-softmax obtained better detection rates and identification rates on the open-set recognition task.

V. DISCUSSION AND CONCLUSION

We presented a readily implemented modification to the final softmax layer activations and demonstrated its potential benefits. It induces deep feature spaces in which classes are more concentrated. It automatically attenuates gradients of badly incorrect, or hard, examples and tends to amplify those belonging to clean examples, allowing convergence to good solutions even when many training examples are mislabelled.

We trained models on CASIA-Webface [19] and tested on LFW [20] and YTF [21]. Cam-softmax performed well in comparison experiments on open-set recognition and pair matching tasks. We corrupted Fashion-MNIST and Casia-WebFace training datasets by mislabelling examples and

showed that cam-softmax exhibits robustness to labelling errors.

Unsurprisingly, increases in face recognition accuracy are possible by using even larger (often private) training sets [6, 15, 22]. We would similarly expect cam-softmax to benefit from larger training sets. Finally, we would emphasise that face recognition performance also depends to a large extent on the preceding alignment of face images. We used the *cp2tform* and *imtransform* functions built into Matlab to align faces according to facial landmarks determined by MTCNN [23]. Switching alignment algorithm to the one available in OpenCV resulted in loss of test accuracy, e.g., the best pair matching accuracy dropped to 97.6%.

The experiments in this paper mislabelled training data in an artificial way. Future work could usefully explore cam-softmax performance on datasets mislabelled unintentionally.

REFERENCES

[1] Fei Wang, Liren Chen, Cheng Li, Shiyao Huang, Yanjie Chen, Chen Qian, and Chen Change Loy, "The devil of face recognition is in the noise," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 765–780.

[2] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song, "Sphereface: Deep hypersphere embedding for face recognition," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6738–6746, 2017.

[3] F. Wang, J. Cheng, W. Liu, and H. Liu, "Additive margin softmax for face verification," *IEEE Signal Processing Letters*, vol. 25, no. 7, pp. 926–930, July 2018.

[4] Walter J Scheirer, Anderson de Rezende Rocha, Archana Sapkota, and Terrance E Boult, "Toward open set recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 7, pp. 1757–1772, 2013.

[5] Han Xiao, Kashif Rasul, and Roland Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," 2017.

[6] Florian Schroff, Dmitry Kalenichenko, and James Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 815–823.

[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[8] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao, "A discriminative feature learning approach for deep face recognition," in *European Conference on Computer Vision*. Springer, 2016, pp. 499–515.

[9] Sumit Chopra, Raia Hadsell, and Yann LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. IEEE, 2005, vol. 1, pp. 539–546.

[10] Raia Hadsell, Sumit Chopra, and Yann LeCun, "Dimensionality reduction by learning an invariant mapping," in *Computer vision and pattern recognition, 2006 IEEE Computer Society Conference on*. IEEE, 2006, vol. 2, pp. 1735–1742.

[11] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang, "Large-margin softmax loss for convolutional neural networks.," in *ICML*, 2016, pp. 507–516.

[12] Xuezhi Liang, Xiaobo Wang, Zhen Lei, Shengcai Liao, and Stan Z Li, "Soft-margin softmax for deep classification," in *International Conference on Neural Information Processing*. Springer, 2017, pp. 413–421.

[13] Jiankang Deng, Jia Guo, and Stefanos Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," *CoRR*, vol. abs/1801.07698, 2018.

[14] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Zhifeng Li, Dihong Gong, Jingchao Zhou, and Wei Liu, "Cosface: Large margin cosine loss for deep face recognition," *CoRR*, vol. abs/1801.09414, 2018.

[15] B. Chen, W. Deng, and J. Du, "Noisy softmax: Improving the generalization ability of DCNN via postponing the early softmax saturation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 4021–4030.

[16] Feng Wang, Xiang Xiang, Jian Cheng, and Alan L. Yuille, "Normface: L2 hypersphere embedding for face verification," in *ACM Multimedia*, 2017.

[17] Yu Liu, Hongyang Li, and Xiaogang Wang, "Rethinking feature discrimination and polymerization for large-scale recognition," *arXiv preprint arXiv:1710.00870*, 2017.

[18] Rajeev Ranjan, Carlos D Castillo, and Rama Chellappa, "L2-constrained softmax loss for discriminative face verification," *arXiv preprint arXiv:1703.09507*, 2017.

[19] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li, "Learning face representation from scratch," *arXiv preprint arXiv:1411.7923*, 2014.

[20] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," Tech. Rep. 07-49, University of Massachusetts, Amherst, October 2007.

[21] Lior Wolf, Tal Hassner, and Itay Maoz, "Face recognition in unconstrained videos with matched background similarity," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 529–534.

[22] Yi Sun, Xiaogang Wang, and Xiaoou Tang, "Deeply learned face representations are sparse, selective, and robust," in *CVPR*, 2015, pp. 2892–2900.

[23] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, Oct 2016.