Self-Checkout Product Class Verification using Center Loss approach

Bernardas Ciapas
Institute of Data Science and Digital Technologies
Vilnius University
Akademijos str. 4
Vilnius, LT-08663,
Lithuania
bernardas.ciapas@mif.vu.lt

Povilas Treigys, Ph.D.
Institute of Data Science and Digital Technologies
Vilnius University
Akademijos str. 4
Vilnius, LT-08663,
Lithuania
povilas.treigys@mif.vu.lt

ABSTRACT

The traditional image classifiers are not capable to verify if samples belong to specified classes due to several reasons: classifiers do not provide boundaries between in-class and out-of-class samples; although classifiers provide separation boundaries between known classes, classifiers' latent features tend to have high intra-class variance; classifiers often predict high probabilities for out-of-distribution samples; training classifiers on unbalanced data results in bias towards over-represented classes. The nature of the class verification problem requires a different loss function than the ubiquitous cross entropy loss in traditional classifiers: input to a class verification function includes a suggested class in addition to an image. As opposed to outlier detection, space is transformed to be not only separable, but discriminative between in-class and out-of-class inputs. In this paper, class verification based on a euclidean distance from the class centre is proposed and implemented. Class centres are learnt by training on a centre loss function. The method's effectiveness is shown on a self-checkout image dataset of 194 food retail products. The results show that a two-fold loss function is not only useful to verify class, but does not degrade classification performance - thus, the same neural network is usable both for classification and verification.

Keywords

Self-checkout images, class verification, centre loss, outlier detection.

1 INTRODUCTION

Real-world computer vision tasks include a need to verify claims that an image contains a claimed type of object. A popular research area of class verification is face verification ([1], [15], [18], [14], [21], [5], [13]), where a class represents a person. In face verification, the computer vision task is to verify if a person in an image is the same person he/she claims to be. The negative samples are usually ID of another person than in the image. Another popular research area in class verification is predicting image authenticity, given an image and a class in that image ([8], [11]). The negative examples are usually images generated by conditional generative adversarial networks (GAN). In food retail self-checkouts, a computer vision task attempts to check if a product in an image is the same product as a customer's chosen one. Negative samples are images of products other than the customer's choice.

The class verification task is a binary classification task that takes two inputs: an image and a label ([21], [2]). A label is one of the trained classes. An image contains one of the following: a) an object of the same class as the label b) an object of a different trained class than the label or c) out-of-distribution input (OOD - any object of the unknown class or no object at all). The goal of the class verification task is to separate a) ("Correct") from the rest ("Incorrect") - whether an object in an image belongs to the claimed class or not. Class verification does not need to distinguish between b) and c) - whether an object in an image is of any other known class, or an unknown class, or does not contain an object at all.

The class verification task differs from other classic computer vision tasks - classification and detection. Multi-class classification algorithms predict the dis-

tribution of probabilities only among a list of known classes. An image containing an object of an unknown class passed to a classification neural network leads to unpredictable results, whereas the class verification task requires it to be rejected as "Incorrect" no matter what label is passed as input. Passing an image containing a known class' object to a classifier is likely to yield a higher probability for the correct class. Still, the probability boundary that separates correct class from incorrect is unknown. Thus, classification networks cannot be used for verifying class identities directly. Detection networks usually consist of two steps: 1) predicting object patches with the highest objectness probabilities (whether an object of a known class exists) and 2) predicting probability distributions of the patches between the known classes (classifying). Thus, detection algorithm errors in predicting objectness are penalized differently from errors classifying known objects. Nevertheless, class verification tasks are indifferent to the existence of objects other than the claimed class. Training detection networks require labels with bounding boxes around the objects, but class verification tasks are indifferent to object location within images - both during training and inference. Detection networks usually classify image crops having the highest objectness scores in a similar way as classification networks: they distribute class probabilities among the list of known classes. Therefore, a similar lack of boundary between correct and incorrect classes in detection networks makes them directly unusable to verify classes.

Despite the lack of proper loss function for verifying classes in image classification and detection neural networks, their ability to extract image features has been widely demonstrated [25], [19], [10]. Knowledge transfer is widely used between different tasks. Therefore, the backbones of neural classification or detection neural networks are likely useful in class verification if the loss function is changed.

Class verification task relates to conditional outlier detection task ([16], [17]). Outlier detection algorithms draw a boundary between in-distribution and out-of-distribution samples, and judge new samples based their relation to that boundary. However, outlier detection tasks do not make an explicit attempt to transform space in such a way that all samples (of a single class) are placed nearby.

Class verification task has a wide variety of applications. In the context of face verification, the suggested class comes from a presented ID, which must be confirmed by class verification. In the context of self-checkouts, the suggested class is a customer's chosen product from a picklist menu, which must be confirmed by class verification.

2 RELATED WORKS

Images are multi-dimensional data points that must be reduced in dimensionality prior to applying machine learning techniques. The most common by far are convolutional neural networks. [24], [4] use a fully connected layer of a classifier.

Outlier detection estimators judge samples by learnt boundary between in-distribution and out-of-distribution samples. Boundary shapes differ by method: robust covariance [17] learns ellipsoid-, one-class SVM [7] learns hyperplane-, isolation forest [6] learns any-shaped boundaries. This research uses a simple hypershere-shaped boundary. However, our loss function pushes latent space variables of any class to the same point ("class centre"), thus a centre-enclosing hypersphere-shaped boundary suffices.

The class prototype is a generalization of multiple data samples of a single class. Multiple research attempts have been made to derive a class prototype given a set of data samples. [9] uses the term "class prototype in a semantic space", which is category vectors (one per category). They construct the category vectors by using auxiliary textual information about the classes of interest. We do not use any textual or other information about the classes to train category vectors - mostly because discriminative textual information is not easily obtainable for the country-, chain-, or store-specific classes of self-checkout products.

A typical task of class verification is a well-researched face verification. Siamese network in [18] effectively learns a distinction function - whether two images belong to the same class (person) or not. It consists of two identical networks with shared weights and a distinction layer that measures the euclidean distance between embeddings of a fully connected layer. A similar concept is employed in Triplet Loss [14], except it uses three images to calculate a loss function: an anchor, a positive (same class/persons') and a negative (another person's). The Anchor+Positive pair is trained to output an opposite value than the Anchor+Negative pair. Both distinction function-based methods - Siamese and Triplet - require reference images (or their embeddings) during inference. Although that's usually satisfiable when a number of images per class is small, using big training datasets faces several challenges: first, different reference images lead to different verification results; second, inferring against multitude of reference images is rarely feasible due to performance and storage reasons.

Research in artificial intelligence safety attempts to verify if the input is consistent with known (in-distribution) samples. Deep Verifier Networks (DVN) [2] use an autoencoder's latent layer's activations to estimate the density of known samples. Samples with latent activations inconsistent with the density model are rejected as adversarial. DVN does not attempt to model latent space where intra-class samples are clustered together. Thus DVN does not derive class prototypes. Since our "adversarial" samples are images of other than the declared class, we suggest that deriving a class prototype is meaningful.

Another way to verify class is to derive a class prototype during training and then compare an input sample against the prototype during inference. The Discriminative Feature Learning [21] derives a class centre using a neural net's latent layer's activations. They use a two-fold loss function: one member is a standard classifier's cross-entropy loss; another is the euclidean distance from a class prototype's centre. Class centres are updated in every iteration, thus "learned". Training such a two-fold loss function pushes the latent layer activations closer together for samples of the same classes. The first member of such a loss function - cross-entropy - ensures that different class centres are separable, i.e. do not regress to the same point. They perform extensive experiments to pick the best weight between the summands of the loss function. In this article, authors use the same two-fold loss function (Eq. 1) in the experiments. In addition, authors perform experiments on the best size of the latent layer. Discriminative Feature Learning focuses on verification and only uses cross-entropy loss to separate class centres but does not provide classification results. We recognize that classification results are as important as verification results, thus provide both and compare classification-only versus classifier-plus-verifier results.

The loss function of class-prototype-based methods measure the distance between class prototype's and a sample's embeddings. SphereFace [13], ArcFace [5] measure the angular distance between class prototype's and sample's embeddings, then modify cross-entropy loss function to use angular distances. They show better discrimination of inter-class features than regular cross-entropy. Their unifold loss function does not allow to adjust classification vs. verification relative importance, but this research' loss function (Eq. 1) allows it using λ hyperparameter. Since this research' primary focus is verification, and authors only include cross-entropy loss in order to preserve class separability, i.e. not to regress all class centres to the same point, it is important to adjust this relative importance.

3 METHODS

3.1 Dataset

Authors used the same dataset of retail products in the self-checkout environment as in their other research articles [3] and [4]. The dataset contains 194 different food retail products that do not usually carry barcodes. Thus, they need to be identified using different methods at the time of checkout. The training dataset was balanced by data augmentation and contained roughly 10K different images per class, most being augmented variations of original images. Neither the testing nor training set included out-of-distribution samples - samples of unknown classes. All the negative samples (i.e. "Incorrect" selections) were generated by labelling an image with one of the available classes other than the correct class. Out-of-distribution samples were not included due to difficulties in collecting them. Authors recognize that including out-of-distribution samples might be helpful in further research.

3.2 Architecture Details

Authors started architecture experiments with their own individual class classifier's backbone that is explained in detail in [3]. The classifier's backbone contains 7 convolutional and 2 dense blocks. Each block contains a Convolutional or a Dense layer, followed by a BatchNorm layer, followed by a ReLU activation. Each convolutional layer is followed by a MaxPool layer. It was shown to perform in other research papers [3] and [4] on the same self-checkout dataset. Presumably, this implies that the architecture is fit to carry enough information through the network layers about the classness of sample images. In addition, the architecture contains little parameters (3.2mln) compared to leading architectures on big sets like ImageNet - CoCa [22] (2100mln), ViT-G/14 [23] (1843mln), EfficientNet [20] (11mln and up).
In addition to the original classifier, the authors added a Center Loss layer (explained below) and an Extra Dense layer. The final model architecture is shown in Fig.1. Experiments were performed, and results were reported using different sizes of the Extra Dense layers. Training without the Extra Dense layer did not saturate the loss function.

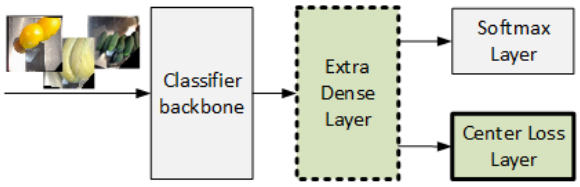

Figure 1: Model architecture

3.3 Center Loss layer

The Center Loss (CL) layer takes two inputs: activations $\in \mathbb{R}^{m*cnt}$ (m - number of minibatch samples; cnt - count of neurons in the extra dense layer) and image labels (m one-hot vectors). It has an internal parameter of class centres $\in \mathbb{R}^{n*cnt}$ (n - number of classes). The output of the CL layer is the difference vector $\in \mathbb{R}^{m*cnt}$ between samples' corresponding class centres and samples' activations of the extra dense layer.

3.4 Loss function

The class verification task's loss function should return a high value when the image does not contain an object of the claimed class and a low value when it does. Authors suggest a concept of a class centre (or a class prototype) - virtual data points in latent space, one per class. The design of the loss function was twofold: first, the intra-class proximity had to be developed; second, the inter-class difference had to be preserved. Such a loss function allows verifying if the sample belongs to the class based on its distance from other samples of that class.
The entire loss function (Eq. 1) is a weighted sum of Cross Entropy loss and Center loss, having a relative weight hyperparameter λ.

$$L = L_S + \lambda * L_C \tag{1}$$

where:

L – Total Loss
L_S – Cross Entropy Loss of Softmax
λ – Center Loss weight (hyperparameter)
L_C – Center Loss

Cross Entropy loss (Eq. 2) preserves differences between classes. Without Cross Entropy loss, the centres of all classes are likely to regress to one point. Cross Entropy loss does not minimize differences between various samples of the same class.

$$L_S = -\sum_{i=1}^{m} log \frac{e^{W_{y_i}^T * x_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T * x_i + b_j}} \tag{2}$$

where:

L_S – Cross Entropy Loss of Softmax
m – Number of samples
n – Number of classes
x_i – i-th sample's activations extra dense layer
y_i – i-th sample's label
$W \in \mathbb{R}^{cnt*n}$ – Weights, last dense layer
$b \in \mathbb{R}^n$ – Biases, last dense layer
cnt – Count neurons, extra dense layer (hyper-p)

Center Loss (Eq. 3) aims to minimize the distance between various intra-class samples. A concept of the class centre is introduced: it is an average vector of all samples in that class of the extra dense layer's activations. The sample's distance from the class centre is calculated as L_2 norm of the difference between the sample's and the class centre's vector. Although other distance types than Euclidean are available, authors limited this research to L_2 only.

$$L_C = \frac{1}{2} \sum_{i=1}^{m} \|x_i - c_{y_i}\|_2^2 \tag{3}$$

where:

L_C – Center Loss
m – Number of samples
x_i – i-th sample's activations of the extra dense layer
y_i – i-th sample's label
c_{y_i} – Center of the y_i-th class

Whereas centre loss attempts to minimize the distance between a "class centre" and samples of that class, it does not attempt to maximize inter-class distances. Although it is possible extending the loss function to punish low inter-class distances would result in an even better separation of classes, authors left it out of this research.

3.5 Training Details

The Center Loss layer did not have any trainable parameters updateable by gradient descent. Yet, the internal parameter of class centres was updated in each iteration: only centres of the classes represented by the samples in a minibatch were updated, whereas unrepresented class centres were left untouched. Class centres were updated as shown in Eq. 4.

$$Center = Center + \alpha * (Activations - Center) \tag{4}$$

where:

$Center$ – center of a a sample's class $\in \mathbb{R}^{cnt}$
$Activations$ – extra dense layer's activations $\in \mathbb{R}^{cnt}$
α – center's learning rate (hyperparameter)
cnt – count of neurons extra dense layer (hyper-p)

Authors trained for up to ten epochs with a patience criteria of five epochs (i.e. training was stopped and best weights restored if the five last epochs of training did not improve the validation loss function value). Training on a relatively big training set of two million images (about 10 thousand per class) usually saturated in the

first 1-2 epochs. Therefore a maximum of 10 epochs was never reached. The criteria for the best weights selection and early stopping was total validation loss, which is the sum of Softmax layer loss and Centre loss. Adam [12] optimizer with a default learning rate of 0.001 was used.
Authors had to choose a proper λ value (relative weight of Centre loss vs Softmax' loss). The value was chosen 0.1 from previous experiments [21]. Finding the best value of λ was left out of scope of this paper and needs to be investigated in further research.
The training was executed on a GPU Nvidia Tesla V100-SXM2-32GB. The training duration of 1 epoch on about two million images varied between 45 and 55 minutes.

3.6 Testing Details

For every test image, authors measured the distance from every class centre. That gave $m*n$ measurements (m - dataset size; n - number of classes), of which m were positive ("Correct" selections) and $m*(n-1)$ were negative ("Incorrect" selections). The results were calculated by giving weight $(n-1)$ to the positive measurements so that the "Correct" and the "Incorrect" classes were balanced. The test set did not contain any out-of-distribution (OOD) samples, i.e. samples outside the known list of classes.

4 RESULTS

The main result in Table 1 shows class verification Equal Error Rate (EER, or Error Rate@False Positive Rate=False Negative Rate) and Receiver Operating Characteristic's Area Under Curve (ROC AUC). Authors exclude neuron counts before near-saturation was reached.

Neuron Count	EER	ROC AUC
2048	**0.073**	0.978
1536	**0.073**	0.978
1024	0.076	0.976
768	**0.073**	**0.979**
512	0.076	0.974
256	0.110	0.956

Table 1: Equal Error Rate (EER) and ROC Area Under Curve (AUC) for various neuron counts in Extra Dense layer

Figure 2 displays ROC curves for various distances from centre thresholds and for various number of neurons in the extra dense layer. ROC Area Under Curve saturates when the Center Loss layer reaches approximately 512-768 neurons.

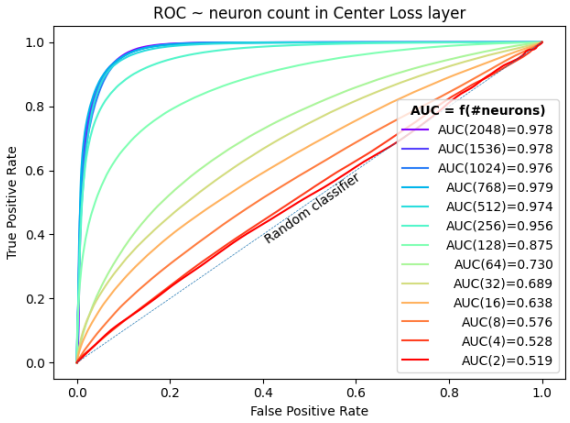

Figure 2: Receiver Operating Characteristic (ROC) curves for various numbers of neurons in the extra dense layer

Figure 3 shows ROC AUC's experimentally found dependency on the number of neurons in Extra Dense layer. AUC climbs steeply until it saturates at about 512 neurons. An increase in neuron count above 512 does not improve AUC.

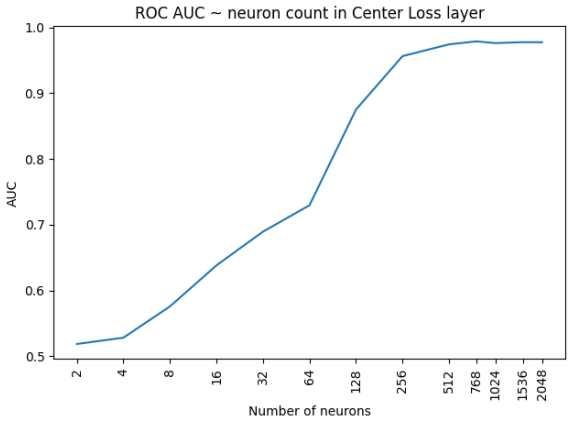

Figure 3: ROC Area Under Curve (AUC) dependency on the number of neurons in the extra dense layer

Figure 4 depicts the accuracy of the individual class classification. The original classifier performed at 73.2% accuracy on the validation set. Authors expected that an additional component of Center Loss in the Loss function would decrease the accuracy of individual class classification. However, the graph shows approximately the same individual classification accuracy when the Extra Dense layer contains at least 8 neurons: 73.2%±0.8%.

Figure 5 shows the relative importance of the Loss function summands: L_S (Softmax' cross-entropy loss)

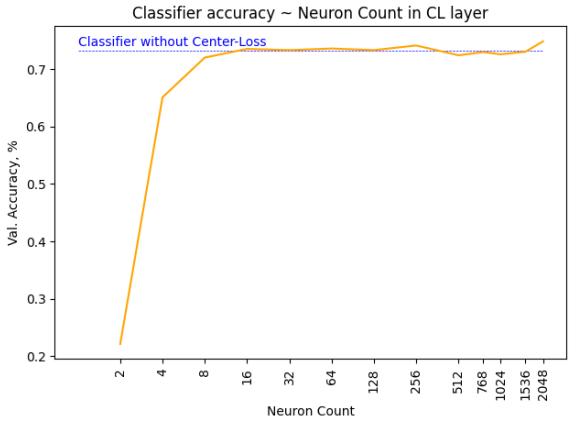

Figure 4: Classification accuracy dependency on the number of neurons in the extra dense layer

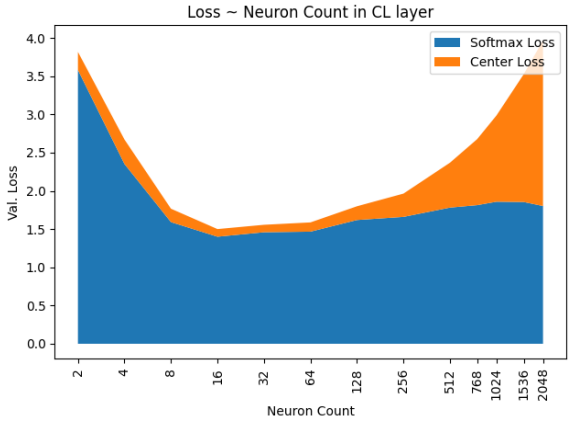

Figure 5: Loss (Softmax and Center) relative values and dependency on number of neurons in the extra dense layer

and L_C (CL layer loss). The Softmax's cross-entropy loss gains minimum value at approximately 8-16 neurons in the Extra Dense layer, then stays at about the same rate upon adding more neurons. This relates to the fact that classification accuracy also saturates at about the same 8 to 16 neurons. Center Loss obtains its minimum values between 8 and 128 neurons, then rises steeply outside this range, mainly due to rising dimensionality of space where distances are calculated.

Figure 6 depicts sample distance distributions for the selected number of neurons in the Extra Dense layer. The separation between distances from samples' own class centres ("Correct" classes) versus from other class centres ("Incorrect" classes) increases with the increase of neurons in the Extra Dense Layer until saturation is reached. The mean distance of both - Correct and Incorrect - increases with the number of neurons.

Figure 7 shows sample images and their distances from selected class centers. Threshold at Equal Error Rate (Thr.@EER) mark separating line between positive (below the line) and negative (above the line) verification result, when False Positive Rate equals False Negative Rate.

4.1 Architecture experiments

Authors performed experiments without the Extra Dense layer shown in Figure 1. With Extra Dense excluded, training did not saturate the loss function and did not achieve satisfiable separation in distances between "Correct" and "Incorrect" classes.

5 ACKNOWLEDGMENTS

Authors express gratitude to Vilnius University Mathematics and Informatics department Information Technologies Open Access Center for providing high-performance computing (HPC) resources used in this research.

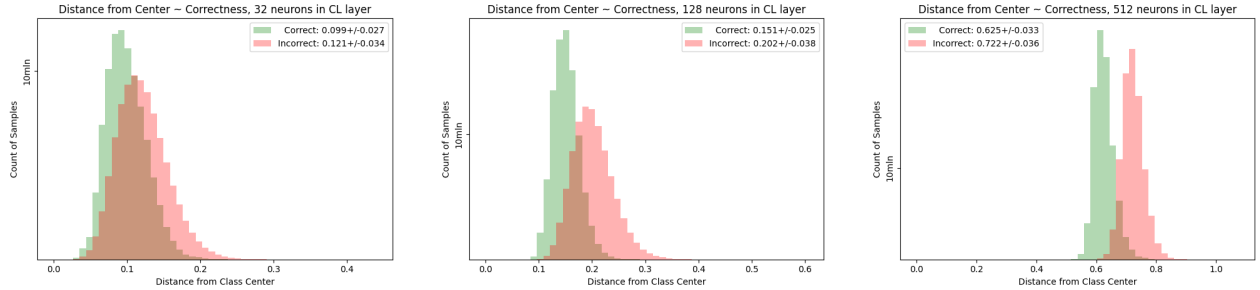

Figure 6: Distance distribution of CL layer activations from the same (Correct) vs other (Incorrect) class centres

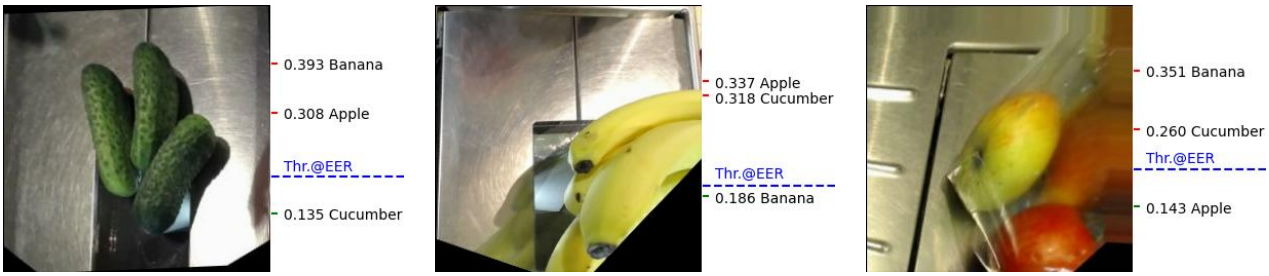

Figure 7: Sample images and their distances from selected class centers

6 CONCLUSIONS

Our work reinforces the value of using the Center Loss function to verify sample's belonging to a class. In a self-checkout products dataset we received 92.7% verification accuracy (at Equal Error Rate). Authors discovered that adding a centre loss function to discriminate class features did not negatively affect classification accuracy: 73.2%±0.8% (vs 73.2% without Center Loss). Thus the same neural network can be used both for classification and verification without sacrificing accuracy. We showed a minimum number of neurons is necessary for both classification accuracy and class verification accuracy to saturate. Once saturated, adding more neurons does not improve classification or verification accuracy.

7 REFERENCES

[1] Ghaliya Alfarsi, Jasiya Jabbar, Ragad M Tawafak, Abir Alsidiri, and Maryam Alsinani. Techniques for face verification: Literature review. In *2019 International Arab Conference on Information Technology (ACIT)*, pages 107–112. IEEE, 2019.

[2] Tong Che, Xiaofeng Liu, Site Li, Yubin Ge, Ruixiang Zhang, Caiming Xiong, and Yoshua Bengio. Deep verifier networks: Verification of deep discriminative models with deep generative models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7002–7010, 2021.

[3] Bernardas Čiapas and Povilas Treigys. High F-score model for recognizing object visibility in images with occluded objects of interest. *Baltic journal of modern computing*, 9(1):35–48, 2021.

[4] Bernardas Ciapas and Povilas Treigys. Retail Self-checkout Image Classification Performance: Similar Class Grouping or Individual Class Classification Approach. In *International Baltic Conference on Digital Business and Intelligent Systems*, pages 167–182. Springer, 2022.

[5] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4690–4699, 2019.

[6] Zhiguo Ding and Minrui Fei. An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window. *IFAC Proceedings Volumes*, 46(20):12–17, 2013.

[7] Sarah M Erfani, Sutharshan Rajasegarar, Shanika Karunasekera, and Christopher Leckie. High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning. *Pattern Recognition*, 58:121–134, 2016.

[8] Xiaowei Huang, Marta Kwiatkowska, Sen Wang, and Min Wu. Safety verification of deep neural networks. In *International conference on computer aided verification*, pages 3–29. Springer, 2017.

[9] Huajie Jiang, Ruiping Wang, Shiguang Shan, and Xilin Chen. Learning Class Prototypes via Structure Alignment for Zero-Shot Recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)*, sep 2018.

[10] G Jignesh Chowdary, Narinder Singh Punn, Sanjay Kumar Sonbhadra, and Sonali Agarwal. Face mask detection using transfer learning of inceptionv3. In *International Conference on Big Data Analytics*, pages 81–90. Springer, 2020.

[11] Sydney M Katz, Anthony L Corso, Christopher A Strong, and Mykel J Kochenderfer. Verification of image-based neural network controllers using generative models. *Journal of Aerospace Information Systems*, 19(9):574–584, 2022.

[12] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[13] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 212–220, 2017.

[14] 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*, pages 815–823, 2015.

[15] Azzam Sleit, R Abu-Hurra, and Wesam Almobaideen. Lower-quarter-based face verification using correlation filter. *The Imaging Science Journal*, 59(1):41–48, 2011.

[16] Xiuyao Song, Mingxi Wu, Christopher Jermaine, and Sanjay Ranka. Conditional anomaly detection. *IEEE Transactions on knowledge and Data Engineering*, 19(5):631–645, 2007.

[17] Xin Sun, Zhenning Yang, Chi Zhang, Keck-Voon Ling, and Guohao Peng. Conditional Gaussian Distribution Learning for Open Set Recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, jun 2020.

[18] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1701–1708, jun 2014.

[19] Srikanth Tammina. Transfer learning using vgg-16 with deep convolutional neural network for classifying images. *International Journal of Scientific and Research Publications (IJSRP)*, 9(10):143–150, 2019.

[20] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[21] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *European conference on computer vision*, pages 499–515. Springer, 2016.

[22] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models, 2022. *URL https://arxiv.org/abs/2205.01917*, 2022.

[23] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12104–12113, 2022.

[24] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12203–12213, 2020.

[25] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020.