MS-CapsNet: A Novel Multi-Scale Capsule Network

Canqun Xiang, Lu Zhang, Yi Tang, Wenbin Zou, Chen Xu

MS-CapsNet: A Novel Multi-Scale Capsule Network

Canqun Xiang, Lu Zhang, Yi Tang, Wenbin Zou, Chen Xu

C.Xiang, Y.Tang, W.Zou and C.Xu are with the Shenzhen Key Lab of Advanced Telecommunication and Information Processing, College of Information Engineering, Shenzhen University, Shenzhen, Guangdong, China (e-mail:wzouszu@sina.com).

L.Zhang is with the Univ Rennes, INSA Rennes, CNRS, IETR (Institut d'Electronique et de Tlcommunication de Rennes) - UMR 6164, F-35000 Rennes, France.

***Abstract*—Capsule network is a novel architecture to encode the properties and spatial relationships of the feature in the images, which shows encouraging results on image classification. However, the original capsule network is not suitable for some classification tasks that the detected object has complex internal representations. Hence, we propose Multi-Scale Capsule Network, a novel variation of capsule network to enhance the computational efficiency and representation capacity of capsule network. The proposed Multi-Scale Capsule Network consists of two stages. In the first stage the structural and semantic information are obtained by the multi-scale feature extraction. The second stage, we encode the hierarchy of features to multi-dimensional primary capsule. Moreover, we propose an improved dropout to enhance the robustness of capsule network. Experimental results show that our method has competitive performance on FashionMNIST and CIFAR10 datasets.**

***Index Terms*—Capsule networks, multi-scale, CNNs, deep learning.**

I. Introduction

CONVOLUTIONAL neural networks (CNNs) [1] [2] are the state-of-the-art methods in image classification. However, CNNs have a lot of issues due to their mechanism of routing data. Routing is the process of relaying the information from one layer to another layer. Pooling operations are applied in the CNNs as routing process. The pooling procedures increase the transition invariance and discard lots of important information such as the location and the pose of the objects which are valuable for classification purpose.

Recently, Sabour et al. [3] explored a novel architecture, called Capsule Network (CapsNet), to overcome CNN's shortcomings. The basic idea is encoding the part-whole relationships (e.g., locations, scales, orientations, brightnesses) between various entities which are objects or object parts, and achieving translation equivariance. For example, for an image without face but containing eyes, nose, mouth, etc. The CNN is likely to wrongly assume that this is a face image because it learns all the facial features. By contrast, the CapsNet learns the relationship between these features (e.g., the eyes should be above the nose.), and can successfully recognize that it is not a face. In the CNN, the low-level structure features and high-level semantic features are extracted at bottom layers and top layers respectively. However, in order to preserve the spatial information, the original CapsNet only uses shallow CNN. Due to the absence of the deep semantic information, the CapsNet performs poorly on the classification task of complex datasets. In order to obtain large receptive field in the shallow convolutional structure, a number of large convolutional kernels are used in the CapsNet. It increases the number of trainable parameters, and makes the model easily overfitting.

To obtain robust features and spatial relationships from the raw images, we propose a new architecture, called Multi-Scale Capsule Network (MS-CapsNet). In this framework, we propose the multi-scale convolution [4] [5] and the multi-dimensional capsule. We introduce multi-scale capsule encoding unit at the bottom layer of the original CapsNet [3]. Firstly, the deep convolutional structure is applied for learning robust information. Besides, we use multilevel small convolutional kernel to decrease the number of trainable parameters. Then the semantic information of entity is encoded by high-dimensional capsule, and the shallow feature of entity is encoded by low-dimensional capsule. Secondly, we propose an improved dropout algorithm on the encoded capsules to enhance robustness of the model. Finally, we employ dynamic routing mechanism [3] to fuse information of multi-dimensional capsules.

In summary, this paper has the following contributions: i) We propose a multi-scale capsule network to fully encode hierarchical features of raw images. ii) We propose an improved dropout algorithm for the capsule layer. iii) We investigate the performance of MS-CapsNet on FashionMNIST and CIFAR10 classification tasks. The results show that MS-CapsNets significantly outperform CapsNets.

II. Related work

The traditional deep neural networks might not be efficient in capturing the hierarchical structure of the entities in the images [6] [7] [8]. In order to preserve the spatial information, Hinton et al. [9] proposed the concept of "capsules" in machine learning terminology. The capsule is a vector to represent internal properties that can be used to learn part-whole relationships. CapsNet, as a more effective image recognition algorithm, was first implemented by Sabour et al. [3] in 2017, and has been received a lot of attention from researchers. Since then, some innovative works have promoted the development of CapsNet.

This work is supported by the NSFC Project under Grant 61771321, in part by the Natural Science Foundation of Shenzhen under Grant KQJSCX20170327151357330, Grant JCYJ20170818091621856, and Grant JSGG20170822153717702, and in part by the Interdisciplinary Innovation Team of Shenzhen University.*(Corresponding author: Wenbin Zou.)*

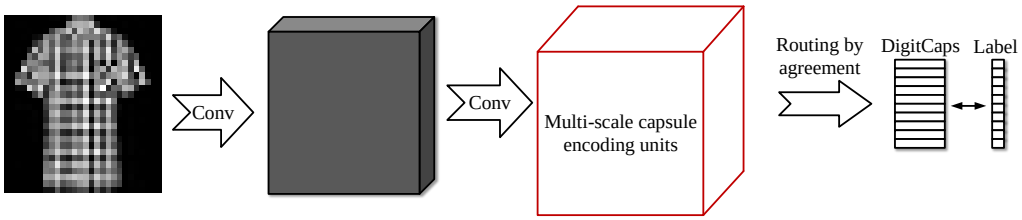


Fig. 1. MS-CapsNet Architecture: A simple MS-CapsNet with 3 layers. The first layer has 256, 9×9 convolution kernels with a stride of 1 and ReLU activation. The second one has 10 multi-scale capsule encoding units. The final layer contains 10 16D digit capsules. The length of the activity vector of each capsule in digit capsule layer indicates presence of an instance of each class and is used to calculate the classification loss.

Hinton describes a version of capsules in [10], where matrix capsule is proposed to learn the relationship between the entity and the observer (the pose). This architecture better represents different properties of the same entity. Chen, et al. [11] embed the routing procedure into the optimization procedure with all other parameters in neural networks, which overcomes the disadvantages that the optimal number of routing procedure has to be found manually. Mohammad. [12] proposes a spectral capsule network, which measures the coincidence as the degree of alignment of the votes from capsules in lower layers in a one-dimensional linear subspace. The proposed method improves stability and convergence speed of capsule network. However, all these methods only consider the spatial structure information, which limits the performance of capsule network.

Recently, the CapsNet has also been introduced into many fields. Jaiswal A, et al. propose the CapsuleGAN [13], a framework that uses capsule networks to replace the standard convolutional neural networks as discriminators. Parnian A, et al. [14] adopt CapsNets for brain tumor classification. James O. [15] introduces siamese capsule networks, a new variant that can be used for pairwise learning tasks. Turab I, et al. [16] successfully utilize capsule routing mechanism for sound event detection. Rodney L, et al. [17] propose convolutional-deconvolutional capsule network, which expands capsule networks to object segmentation and propose the concept of deconvolutional capsules. Aryan M, et al. [18] propose a fast CapsNet for lung cancer screening by applying a consistent dynamic routing mechanism to speed up CapsNet.

III. Multi-scale Capsule Network

In this paper, we take into consideration the hierarchical features, and exploit the multi-dimensional capsules to encode the hierarchical features. As shown in Fig.1, the MS-CapsNet is shallow with two convolutional layers and one fully connected layer. The first layer is a standard convolution layer. The second one is multi-scale capsule encoding units. The final layer is a digit capsule layer. There is a routing between the multi-scale capsule encoding unit and the digit capsule layer. The objective function for the multi-category capsule network can be shown in Eq.(1).

$$L_M = \sum_{j=1}^{J} T_j max(0, m^+ - \|V_j\|)^2 + \lambda(1 - T_j)max(0, \|V_j\| - m^-)^2 \quad (1)$$

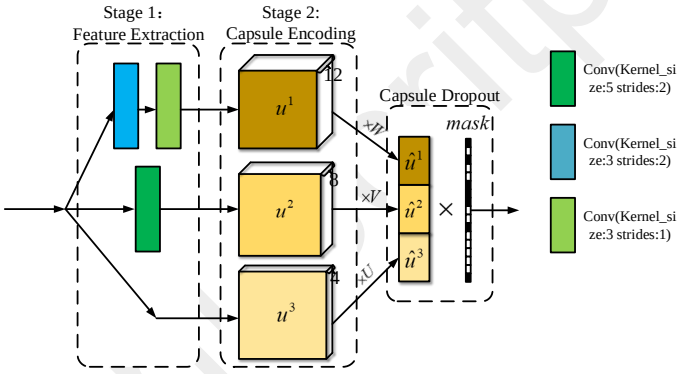


Fig. 2. Multi-Scale Capsule Encoding Architecture: there are three branches in each channel. Each branch has a different level of feature extraction, which is coded into the primary capsule of different dimensions, and then converted to the same dimension by the weight matrix.

where T_j and $\|V_j\|$ represents j-th target labels and the length of j-th digit capsule, respectively. m^+ and m^- denote maximum margin and minimum margin respectively. The λ is down-weighting factor for preventing initial learning from shrinking the lengths of the capsule outputs in the final layer. The total loss is simply the sum of the losses of all digit capsules.

A. Multi-scale Capsule Encoding Unit

A capsule is defined as a group of neurons in the CapsNet. It is a vector that has both direction and length. The direction of capsule captures the entity's attributes, such as orientation and location. The length of capsule represents the probability of entity existence.

$$v_j = \frac{\|s_j\|^2}{1 + \|s_j\|^2} \frac{s_j}{\|s_j\|} \quad (2)$$

The length of capsule is compressed to [0,1] without changing its direction by Eq.(2), so that its length can be interpreted as the probability that a given feature being detected by the capsule. Here v_j is the output of j-th capsule and s_j is its total input.

In the CNN, the hierarchy of features are drawn from different convolution layers. The bottom layers can extract rich structure information, and top layers can extract semantic information. Both of them can support to fully represent the input data. We design a multi-scale structure to extract hierarchy information, and then encode the information into

primary capsule layer. After that the predictive capsules are obtained by the transformation matrix.

As shown in Fig.2, the unit consists of two stages. In the first stage, the structural and semantic information are obtained by the multi-scale feature extraction. The first 2 layers of the top branch are the high-level feature extraction process, which extracts semantic information. The first layer of the middle branch is used for extracting medium-level features. The bottom branch directly employs the original features without trainable parameter layer.

In the second stage, we encode the hierarchy of features into multi-dimensional primary capsule. We exploit the last layer of the three branches to encode high-level, medium-level and low-level features, which obtains 12D, 8D and 4D capsules respectively. Through the employment of three branches, we obtain the multi-dimensional primary capsule. Then, the predicted vectors are computed through different weight matrixes as follows:

$$\widehat{u}^1_{j|i} = W_{ij} u^1_i \tag{3}$$

$$\widehat{u}^2_{j|i} = V_{ij} u^2_i \tag{4}$$

$$\widehat{u}^3_{j|i} = U_{ij} u^3_i \tag{5}$$

$$\widehat{u} = concat(\widehat{u}^1, \widehat{u}^2, \widehat{u}^3) \tag{6}$$

where W, V and U are three weight matrixes between u^1, u^2, u^3 and $\widehat{u}^1, \widehat{u}^2, \widehat{u}^3$ respectively. u^k_i represents i-th primary capsule from k-th branch. $\widehat{u}^k_{j|i}$ represents predict vector between j-th parent capsule and i-th child capsule of k-th branch. The $\widehat{u}$ is the output of this multi-scale capsule encoding structure, which concatenates the results of three branches by function $concat()$. The information is encoded by using a weight matrix between i-th child capsule and j-th parent capsule. During the training, the part-whole relationship for each capsule pair is learned by adjusting the transformation matrix W, V and U.

B. Capsule Dropout

Dropout prevents common overfitting by making other hidden units unreliable. In CapsNet, each of capsule is a vector, the dropout has to discard a vector rather than some elements in the vector. As shown in Fig.3, for a capsule, a standard dropout algorithm [19] can only throw away some of its elements. That changes the direction of the capsule, which results in changing the properties of the entity that the capsule represents, and leads to a false recognition. For example, there are two capsules represent nose (1,1,1) and eye (1,1,0) respectively. The standard dropout algorithm can discard any elements in the nose and eye via the Bernoulli distribution. If the third element of nose is dropped, the nose and eye have the same direction (1,1,0). The phenomenon leads to the difficulty in learning. Therefore, we improve the dropout algorithm for capsule by changing the encoding method of mask. We regard each capsule as a whole, which ensures the direction of capsule has not changing. Then, some capsules are randomly dropped by Bernoulli distribution. Due to the invariance of direction, the improved dropout algorithm is more suitable for the neurons of vector.

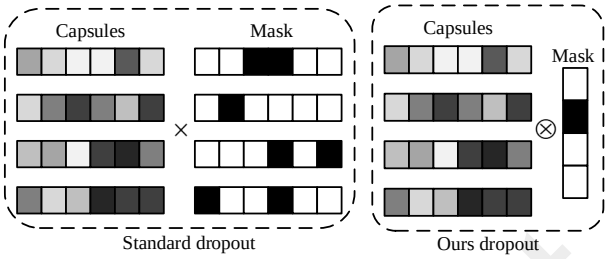


Fig. 3. Capsule Dropout: Our dropout has different encoding of the mask. The gray values represent the true values, the black is 0 and the white is 1. '×' means element-wise multiplication,'⊗' means broadcast multiplication.

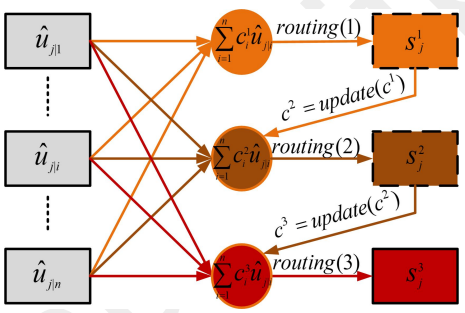


Fig. 4. Dynamic Routing: the routing process of the 3 iterations is shown, and different colors represent a complete iteration.

C. Dynamic Routing

Dynamic routing is a kind of information selection mechanism, which ensures that the outputs of child capsules are sent to the proper parent capsules. In the previous section, the prediction vectors $\widehat{u}$ are computed through weight matrix. The relationship is determined between each parent capsule s_j and the prediction vector $\widehat{u}$ by dynamic routing. As shown in Fig.4, this is the three iteration routing process between a parent capsule s_j and all the prediction vectors $\widehat{u}_{j|i}(i = 1, ..., n)$. In the first iteration, $c^1_i = \frac{1}{n}$ and $s^1_j = \sum_{i=1}^{n} c^1_i \widehat{u}_{j|i}$, where $\sum_j c_j = 1$ and $c_j \geq 0$. It means that each prediction vector contributes the same to the parent capsule, which is an initial state. Then we adjust the routing coefficients c^1 to c^2 by the function $update()$, it is shown as follows:

$$b^{i+1} = b^i + \widehat{u}_j \cdot v_j \tag{7}$$

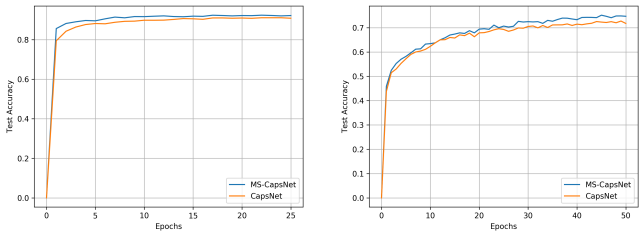
$$c^{i+1} = softmax(b^{i+1}) \tag{8}$$

where b is the coupling coefficient before normalization and $b^1 = 0$, v_j is calculated by Eq.(2). The dynamic routing mechanism will increase the routing coefficient to j-parent capsule by a factor of $\widehat{u}_j \cdot v_j$. Similarly, we can get the parent capsule u^2_j by the coupling coefficient c^2, and then update the coupling coefficient by the parent capsule u^2_j and prediction capsule $\widehat{u}_j$. After three iterations, we obtain the final output of the parent capsule.

IV. Experiments

A. Datasets

To evaluate the performance of the proposed method, we conduct the experiments on the FashionMNIST [20] and CIFAR10 [21] datasets. The FashionMNIST consists of a training



(a) FashionMNIST (b) CIFAR10

Fig. 5. The test prediction accuracy of two models on FashionMNIST and CIFAR10 datasets with iteration increasing. The MS-CapsNet has better performance than the CapsNet.

set of 60,000 examples and a test set of 10,000 examples. Each example is a 28×28 grayscale image, associated with a label from 10 classes. The CIFAR10 consists of 32×32 colored and labeled images coming from 10 different classes, in which each class contains 6,000 images. In our experiments, 50,000 images are used as training data, and 10,000 images are used as testing data.

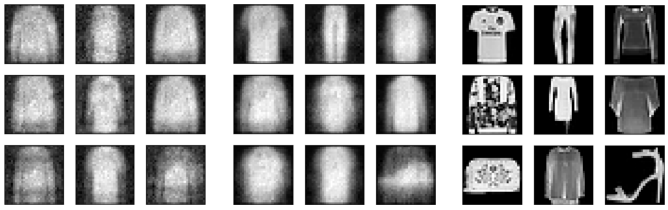
B. System Setup

We implement the MS-CapsNet using the MxNet [22] and employs the Adam optimizing method [23] as the gradient descent algorithm to perform the training. The mini-batch size is set to 128, and the weight decay factor is set to 0.00001.We set different hyper parameters for training FashionMNIST and CIFAR10: the initial learning rate is 0.001 and 0.0001 and the number of iteration is 25 and 50 for converging to optimal solution quickly. The dropout rate is set to 0.4 and 0.1 respectively. The convolution kernel of the first layer of the model is set to 13×13 for the CIFAR10 dataset.

The baseline contains three layers: two convolution and one fully-connected. Conv1 has 256, 9×9 convolutional kernels with stride of 1 and ReLU activation. The second layer is a convolutional capsule layer with 30 channels of convolutional capsules. Each primary capsule is a 8-dimensional vector which is obtained by a convolution with 8 9×9 kernels and a stride of 2 on the output of the previous layer. The final layer is the digit capsule layer which has one 16D capsule per class.

C. Results

Fig.5 shows the test predict accuracy curves of the CapsNet and the MS-CapsNet on FashionMNIST and CIFAR10 datasets. The FashionMNIST is a relatively simple dataset, as it has been size-normalized and centered in a fixed-size image, and each sample is a grayscale image. This regularization alleviates the complexity of datasets, making it easy to represent by neural networks. In contrast to FashionMNIST, CIFAR10 is a more complex dataset, and there are a lot of features and noises. The experiment results show that the MS-CapsNet performs better than the CapsNet on two datasets and has a greater improvement on CIFAR10 dataset. Meanwhile, the MS-CapsNet achieves a faster convergence rate than the CapsNet. The results reveal that the MS-CapsNet is more expressive than the CapsNet because of its multi-scale structure, which has rich feature extraction and coding methods.



(a) Without dropout (b) With dropout (c) Original images

Fig. 6. The results of reconstruction on FashionMNIST.

TABLE I
COMPARISON OF THE BEST TEST ACCURACY, NUMBER OF TRAINABLE PARAMETERS (M IS FOR MILLIONS).

	FashionMNIST		CIFAR10	
	accuracy	$\#params$	accuracy	$\#params$
CapsNet	0.911	25.5M	0.727	26.0M
MS-CapsNet	0.922	**10.8M**	0.751	**11.2M**
MS-CapsNet+Drop	**0.927**	**10.8M**	**0.757**	**11.2M**

The MS-CapsNet has a better robustness on complex datasets, which the detected object has complex internal representations.

As shown in Fig.6, the original images are reconstructed by a training multi-layer perceptron following [3]. The result shows that the proposed dropout method can extract robust features, which reconstruct better performance than without dropout fashion. Furthermore, we find that the standard dropout [19] is non-convergence in our proposed MS-CapsNet, which shows that our proposed capsule dropout is more robust than the standard dropout.

Table 1 shows the comparison of the best test accuracy, and the number of trainable parameters. The performance of the MS-CapsNet is better than that of the CapsNet on two datasets. The best accuracy of MS-CapsNet is higher than that of CapsNet by 1.1% and 2.4% on FashionMNIST and CIFAR10 datasets respectively. The difference between training accuracy and test accuracy is decreased by using improved dropout algorithm, and the test accuracy of the model is further improved on two datasets. In the MS-CapsNet, lots of small convolution kernels are utilized by means of series instead of large convolution kernel. This way can depress the number of parameters and promote the capacity of the model to extract the depth features. In this paper, the number of parameters of CapsNet is two times more than MS-CapsNet, and the test performance of CapsNet is inferior to MS-CapsNet. Objectively, the MS-CapsNet has a better overall improvement.

V. CONCLUSION

In this work, we introduced the MS-CapsNet to enhance the expression of the capsule network. The multi-scale convolution feature extraction and multi-dimensional capsule coding is employed to learn rich represents. Meanwhile, we improve the dropout algorithm to enhance the robustness of the CapsNet. The results indicate that MS-CapsNets perform better on the tested complex dataset and fewer number of trainable parameters are used when better test accuracy is achieved. As our future work, we plan to explore the performance of MS-CapsNet on more complex datasets.

REFERENCES

[1] A.Krizhevsky, I.Sutskever, and G.E.Hinton. "ImageNet classification with deep convolutional neural networks," in *Proc.IEEE NIPS*, Nov.2012, pp.1097-1105.

[2] K.He, X.Zhang, S.Ren, and J.Sun. "Deep residual learning for image recognition," in *Proc.IEEE CVPR*, Jun.2016, pp.770-778.

[3] S.Sabour, N.Frosst and G.E.Hinton. "Dynamic routing between capsules," in *Proc.IEEE NIPS*, Nov.2017, pp.3856-3866.

[4] L.Jongpil and N.Juhan. "Multi-Level and multi-scale feature aggregation using pretrained convolutional neural networks for music auto-tagging," in *IEEE Signal Processing Letters*, vol.24, no.8, pp.1208-1212, Aug.2017.

[5] P.Y.Wang, H.Zhang and V.M.Patel. "SAR image despeckling using a convolutional neural network," in *IEEE Signal Processing Letters*, vol.24, no.12, pp.1763-1767, Dec.2017.

[6] E.Oyallon and M.Stphane. "Deep roto-translation scattering for object classification,"in *Proc.IEEE CVPR*, Jun.2015, pp.2865-2873.

[7] D.E.Worrall, S.J.Garbin, D.Turmukhambetov, and G.J.Brostow. "Harmonic networks: Deep translation and rotation equivariance," in *Proc.IEEE CVPR*, Jul.2017, pp.5028-5037.

[8] T.Cohen and M.Welling. "Group equivariant convolutional networks," in *Proc.IEEE ICML*, Jun.2016, pp.2990C2999.

[9] G.E.Hinton, A.Krizhevsky, and S.D.Wang. "Transforming auto-encoders," in *Proc.International Conference on Artificial Neural Networks*, Jun.2011, pp.44-51.

[10] G.E.Hinton, N.Frosst, and S.Sabour. "Matrix capsules with EM routing," in *Proc.ICLR*, Feb.2018.

[11] Z.H.Chen and D.Crandall. "Generalized capsule networks with trainable routing procedure," in *arXiv preprint arXiv:1808.08692*, Aug.2018.

[12] M.T.Bahadori. "Spectral capsule networks," in *Proc.ICLR*, Feb.2018.

[13] A.Jaiswal, W.AbdAlmageed, and P.Natarajan. "CapsuleGAN: generative adversarial capsule network," *arXiv preprint arXiv:1802.06167*, 2018.

[14] P.Afshar, A.Mohammadi, and K.N.Plataniotis. "Brain tumor type classification via capsule networks," *arXiv preprint arXiv:1802.10200*, 2018.

[15] J.O.Neill. "Siamese capsule networks," *arXiv preprint arXiv:1805.07242*, 2018.

[16] T.Iqbal, Y.Xu, Q.Kong, and W.Wang. "Capsule routing for sound event detection," *arXiv preprint arXiv:1806.04699*, 2018.

[17] R.LaLonde and U.Bagci. "Capsules for object segmentation," *arXiv preprint arXiv:1804.04241*, 2018.

[18] A.Mobiny and N.H.Van. "Fast capsNet for lung cancer screening," *arXiv preprint arXiv:1806.07416*, 2018.

[19] N.Srivastava, G.E.Hinton, A.Krizhevsky, I.Sutskever, and R.Salakhutdinov. "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol.15, no.1, pp.1929-1958, Jun.2014.

[20] H.Xiao, K.Rasul, and R.Vollgraf. "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[21] A.Krizhevsky. "Learning multiple layers of features from tiny images," Apr.2009.

[22] T.Chen, M.Li, Y.Li, M.Lin, N.Wang, M.Wang, and Z.Zhang. "Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems," in *CoRR*, abs/1512.01274, 2015.

[23] K.Diederik and B.Jimmy. "Adam: A method for stochastic optimization," in *Proc.ICLR*, Dec.2014.