Received: July 9, 2022. Revised: August 25, 2022.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

ER-Caps: ELU Residual Capsule Network for Complex Images Classification

Omaima El Alaoui-Elfels[1]* **Taoufiq Gadi[1]**

[1]*Hassan First University of Settat, Faculty of Sciences and Techniques. Mathematics, Computer Science and Engineering Laboratory, 26000 – Settat, Morocco*
*Corresponding author's Email: omaima.ee@gmail.com

Abstract: Capsule Network has proved its performance against image deformation, through its ability to preserve space relations among features. However, the Capsule Network underperforms at classifying complex images datasets. Through an in-depth examination of Residual Neural Network, we present a novel Capsule Network named ER-caps to improve the model performance toward complex images classification and reconstruction. The architecture of ER-caps uses customized residual blocks to enhance feature extraction without using any pooling methods to conserve spatial information among features and preserve the Capsule Network equivariance. Our model adopts the RELU and the ELU activation function to ensure fast convergence and uses an advanced decoder to allow better reconstruction for complex images. The experiments show that the suggested model outperforms the baseline Capsule Network with dynamic routing on a variety of benchmark datasets CIFAR-10, CIFAR-100, and SVHN by approximately: 17 %, 9 %, and 4 % respectively. Moreover, ER-caps reduces the number of training parameters.

Keywords: Residual neural network, Convolutional neural network, Capsule network, Dynamic routing, ELU activation function, Complex images.

1. Introduction

Convolutional Neural Network (CNN) has shown its potential in object recognition [1], and image classification [2]. The deeper structures CNN have been introduced to enhance the feature extraction performance by stacking convolution and pooling layers [3,4] as well as by training as much data as possible. However, these networks have shown issues with gradient exploding, gradient vanishing, and accuracy saturation, which impedes an effective training. To alleviate these issues, Residual Network (ResNet) introduced skip connections between the input and the output of residual block [5], which achieved high precision and stability.

CNN has shown a high performance for image recognition and classification tasks. Nevertheless, the performance of CNN decreases drastically if the image has been rotated, deformed, or tilted. It is due to the inability of CNN to consider the spatial relationship between features. From the use of the max-pooling layer to extract the most significant features and as well as from the image size reduction, spatial information between object entities and other useful features can get lost, resulting in false positive classifications. The use of data augmentation for a multi-angle dataset can alleviate the above issue but cannot eradicate it.

Capsule Network (CapsNet) [6] was introduced in response to the aforementioned shortcomings of CNN; it includes one convolutional layer followed by a primary capsule layer (PrimCaps) and a class capsule layer (ClassCaps) to make classification predictions. Then use a fully connected decoder to reconstruct the input image. CapsNet replaced convolutional scalar values with vectors (capsules). The capsule feature is more discriminative than its scalar values counterpart and determines the translation equivariance characteristic by conserving the spatial relationship among features. However, it did not perform well on complex and colored datasets (such as CIFAR-10, CIFAR-100). This is explained by its tendency to focus too much on unimportant image details. Moreover, the number of convolutional layers used in the shallow CapsNet is not sufficient to extract complex features.

Attempting to address the aforementioned issues of CapsNet, authors in [11] presented a deep CapsNet by adding three convolutional capsule layers (ConvCaps) before the PimaryCaps layer connected by the Dynamic Routing (DR) algorithm. This was the first successful attempt at stacking Capsule layers; however, using DR algorithm several times increases the model complexity [7]. In reference [8], the idea of authors was to stack capsule layers using the concept of dense connections to make the use of hierarchical features more effective. The authors in [12] tended to train deep CapsNet using residual connections with multiple capsule layers. These architectures improved the performance of CapsNet on classifying complex images, but stacking dense capsule blocks increases the number of training parameters; therefore, it requires more training time. Shruthi Bhamidi and El-Sharkawy [14] processed the input by eight basic residual blocks before the PrimCaps layer, while in [13] Res2Net processed the input before the PrimCaps layer to extrapolate multiscale characteristics and use the Squeeze-and Excitation (SE) block to peak useful features and eliminate ineffective ones. The SE block decreases the number of training parameter; however, it uses average pooling which runs counter the CapsNet concept. El Alaoui-Elfels and Gadi [15, 16] used two parallel convolutional layers to process the input image and combine their feature maps by the multiplication gate to select features before the PrimCaps layer. These works motivated us to investigate the effect of the multiplication operator to inject the input to the output of the residual block.

All the works cited previously demonstrated the potent potential of CapsNet, and some architectures upgrade the classification accuracy on complex datasets by using pooling layer, which deviates somewhat from original aim of CapsNet, whereas other works stacked capsules layer which makes the model more complex. Therefore, in order to resolve the previously mentioned challenges of CapsNet [6] and enhance its ability to process complex datasets, we present the ELU Residual Capsule Network (ER-Caps), which is different from the above-mentioned works in two aspects. First, we propose an improved customized residual neural network (ResNet) to enhance features extraction without using any pooling methods. Second, we combine it with CapsNet with an advanced decoder to create a model that benefits from both residual and capsule operations and better reconstruct the input images. The key contributions of our work are the following:

- Originating from the ResNet, we use multiple residual blocks on top of the CapsNet to extract features from complex images. The skip connections used in the ResNet help the information to flow deeper into the network layers. Moreover, they mitigate exploding and vanishing gradient issues.
- We use the Exponential Linear Unit (ELU) and the Rectified Linear Unit (RELU) alternatively in the residual blocks for better performance and faster convergence.
- We perform element-wise multiplication to inject the residual identity to the output of the residual block, this shows higher accuracy than the addition operator does.
- We apply small convolutional kernels on the primary capsule layer to reduce the number of training parameters in the capsule layers.
- We use the ELU activation function in the first two layers of the decoder in order to improve the reconstruction of complex images. This function allows negative input values and therefore passes more information, which results in better reconstructed images.
- We validate the suggested models on several benchmark complex datasets to prove the generalization ability of our model.

We trained our models on the CIFAR-10, CIFAR-100 [9], and SVHN [10] datasets without using any data augmentation method, any ensemble networks or other common training tricks. ER-Caps shows competitive results on the above-mentioned test datasets over most state-of-the-art CapsNet approaches. Furthermore, it allows faster convergence and better reconstruction results.

The rest of this paper is structured as follows: we present the background of CapsNet and ResNet in section 2. In section 3, we detail the proposed ER-Caps model. We demonstrate, compare, and discuss our experimental results in section 4. Finally, we conclude this work in section 5.

2. Background

2.1 CapNet

CapsNet has attracted more and more attention since it has greater picture analysis skills and conforms to the peculiarities of the human perception system. CapsNet presents a new concept that uses capsules instead of scalers, where a capsule is a group of neurons that represents instantiation parameters such as size, hue, position, rotation, texture. The output of a capsule is a vector whose length presents

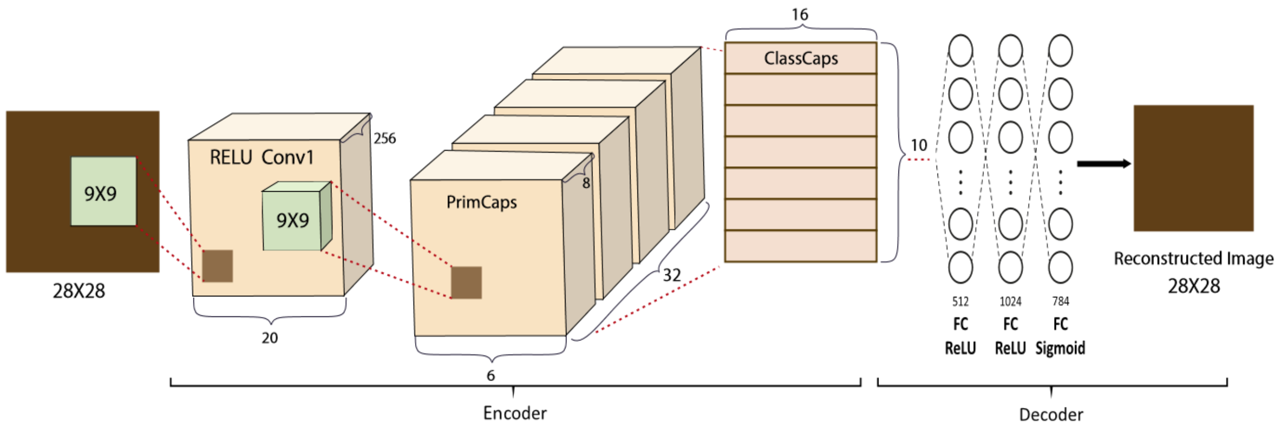

Figure. 1 The baseline CapsNet's structure

the probability of the existence of an entity. If the input image has been rotated, then the output vector keeps the same length, but its orientation changes (equivariance).

The CapsNet [6] architecture was designed for MNIST images. It includes a convolution layer, a PrimCaps layer and a ClassCaps layer for the encoder part as well as three fully connected layers as a decoder (Fig. 1). A convolution layer (256 kernels, 9x9 size, stride of 1, RELU activation function) is applied to the input image to extract features. The resulting feature maps are processed by the PrimCaps layer, using eight convolutional capsules (32 channels, 9x9 kernels, stride of 2). The output feature maps are then encapsulated to vectors of 8 scalars to constitute the primary capsules. These low-level capsules are mapped to class capsules (16D) through the DR algorithm, which replaces the pooling operation to preserve the spatial relationships between object entities. The low-level capsules have a narrow receptive field and can only detect the posture and presence of a specific entity, while the high-level capsules can detect larger and more complicated objects. The fully connected (FC) layers are used to reconstruct the input image via three layers. The first two layers are activated by the RELU function and the third layer by the Sigmoid function.

In the DR algorithm, the output of capsule *i* in PrimCaps is represented by u_i, which is multiplied by W_{ij} to compute the $\hat{u}_{j|i}$, the prediction vector of capsule *i* for the capsule *j* in the ClassCaps Eq. (1). W_{ij} is the transformation weight matrix, which is used to encode the relationship between part and whole object.

$$\hat{u}_{j|i} = W_{ij} u_i \tag{1}$$

Each capsule *j* calculates the weighted sum S_j of the prediction vectors $\hat{u}_{j|i}$ and the coupling coefficient C_{ij} Eq. (2) and then applies the squash function Eq. (3) to ensure that vector length V_j is between 0 and 1, to represent the probability of existence.

$$S_j = \sum_{i=1}^{n} C_{ij}\, \hat{u}_{j|i} \tag{2}$$

$$V_j = \frac{||S_j||^2}{1+||S_j||^2} \frac{S_j}{||S_j||} \tag{3}$$

The coupling coefficient represents the connection strength between parent capsule and child capsule. It is updated iteratively through Eq. (4), where b_{ij} is matching degree between capsule *i* and *j*.

$$C_{ij} = \frac{e^{bij}}{\sum_n e^{bin}} \tag{4}$$

$$b_{ij} = b_{ij} + V_j \hat{u}_{j|i} \tag{5}$$

The loss as described in [6] is the sum of the classification loss and the reconstruction loss.

2.2 ResNet

For complex tasks, deeper networks outperform shallow ones, but they are harder to train [17] and the stacking of layers causes the vanishing gradient issue. Highway networks [18] were the first framework that succeeded to train deep networks of a high number of layers. Highway networks use gating units to optimize and train networks with several layers effectively. ResNets [5] introduced skip connections with the use of identity transformation. Unlike highway networks, they do not provide additional parameters or increase the computing complexity.

The principle of a basic residual block is illustrated in Fig. 2 [5]. It consists of a convolutional

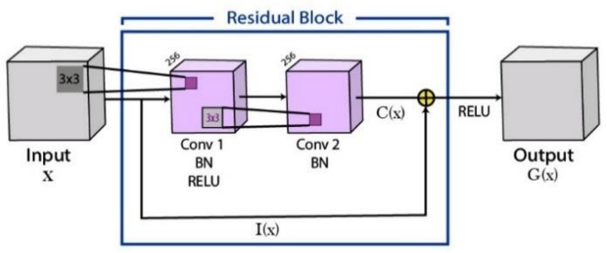

Figure. 2 The basic residual block

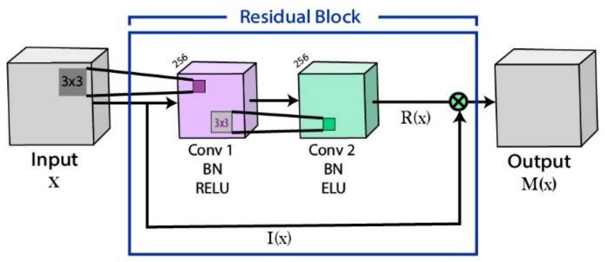

Figure. 3 The proposed residual block structure

layer (Conv 1), batch normalization layer (BN), RELU activation [19] and a second convolutional layer (Conv 2) followed by another BN layer. The residual block adds a direct connection between the input and output of the residual block as follows:

$$G(x) = RELU(C(x) + I(x)) \qquad (7)$$

Where:

- *x* is the input of the residual block.
- *C(x)* represents multiple convolutional transformations used in the residual block.
- *I(x)* is the identity function used to shrink the dimensions of *x* to match those of *C(x)*.
- *G(x)* is the output of the residual block, which is activated by default with the RELU function.

Authors in [20] investigated the performance of ELU function in ResNet by replacing RELU and Batch Normalization with ELU. In [21] the authors changed the single size of convolutional filters into convolution kernels of different sizes to adaptively extract the image features.

3. Proposed framework

The convolutional layer used in the baseline CapsNet [6] merely converts the pixel intensities into feature maps of local feature detectors, which are passed to the PrimCaps layer. For complicated image datasets, this may not be sufficient for further processing in the capsules. To enhance the performance of CapsNet for complex images, we propose to process the input image first by a convolutional layer with small kernels of size 3x3 followed by batch normalization and activated through the RELU function. We then pass the resulting feature maps to an advanced residual network to extract complex features and map the output to the PrimCaps. We maintain the concept of CapsNet that did not use any pooling and focused on equivariance rather than invariance. We modified the baseline ResNet [5] to include three residual blocks based on the skip connections and without using any pooling to keep all spatial information. Each of the residual blocks consists of two convolutional layers (Conv1 and Conv2) followed by a dropout to prevent overfitting and to enhance the generalization ability (Fig. 3).

- Conv1: 256 channel, 3x3 kernel size, a stride of 1, followed by batch normalization and activated through the RELU function.
- Conv2: 256 channel, 3x3 kernel size, a stride of 1, followed by batch normalization and activated through the ELU function.

We choose the ELU function to activate the second convolutional layer in each residual block since it demonstrates its potential in mitigating the vanishing gradient issue, enhancing the network performance and speeding up the training, which allows a fast convergence [20,15]. The ELU is defined as:

$$ELU(x) = \begin{cases} \alpha\,(e^{x} - 1), & x < 0 \\ x, & x \geq 0 \end{cases} \qquad (8)$$

α is a positive hyperparameter that controls where ELU saturates at negative input. The ELU function includes negative values that push the mean near zero, which speeds up the training.

Inspired by [22] that used element-wise multiplication to select features instead of the pooling, we tested a combination of addition gate and element-wise multiplication gate to concatenate the identity to the output of Conv2. The element-wise multiplication gate and the addition gate are defined in Eq. (9) and Eq. (10), respectively:

$$M(x) = R(x) \times I(x) \qquad (9)$$

$$A(x) = R(x) + I(x) \qquad (10)$$

Where R is the output of convolutional layers used in our residual block and I was defined in section 2.2.

Received: July 9, 2022. Revised: August 25, 2022.

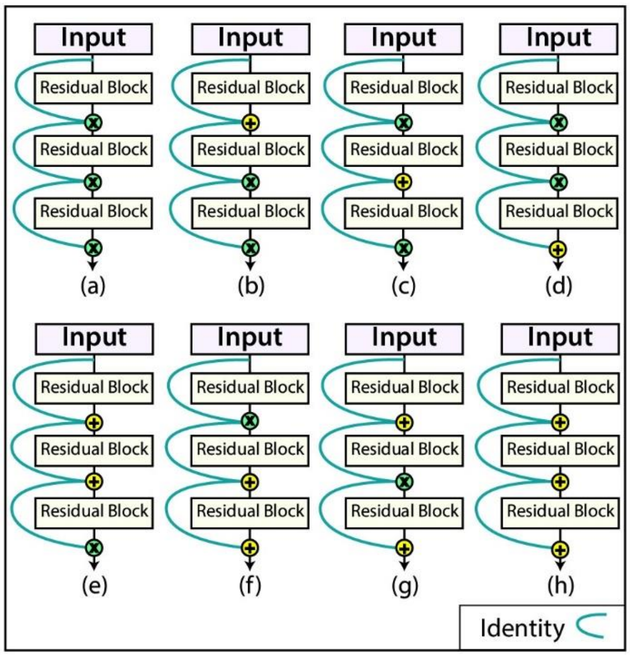

Figure. 4 Residual subnets with different gates to concatenate residual blocks

The proposed residual subnets are presented in Fig. 4.

- (a) Residual subnet with three element-wise multiplication gates (xxx).
- (b) Residual subnet with addition gate and two element-wise multiplication gates respectively (+xx).
- (c) Residual subnet with element-wise multiplication gate, addition gate, and element-wise multiplication gate respectively (x+x).
- (d) Residual subnet with two element-wise multiplication gates and addition gate respectively (xx+).
- (e) Residual subnet with two addition gates and element-wise multiplication gate respectively (++x).
- (f) Residual subnet with element-wise multiplication gate and two addition gates respectively (x++).
- (g) Residual subnet with addition gate, element-wise multiplication gate, and addition gate respectively (+x+).
- (h) Residual subnets with three addition gates (+++).

To argue whether the ELU activation function indeed enhances the network performance, we keep the same architecture of the (d) model, and we replace ELU with RELU in all residual blocks. In order to reduce the number of training parameters, we use small kernels of 4x4 size for the convolution of the PrimCaps layer. The result is encapsulated into 64 primary capsules, mapped to class capsules and connected to the reconstruction network Fig. 5.

To enhance the decoder to reconstruct complex images, we use the ELU activation function for the first two fully connected layers in the reconstruction step. The ELU includes a negative value, which lets more information be passed to the next layer and allows better reconstruction.

4. Experiments and results

4.1 Datasets

We utilize CIFAR-10, CIFAR-100 and SVHN datasets to evaluate the performance of our model. The CIFAR-10 and CIFAR-100 are complex datasets, where each image contains rich features as well as background and noise. CIFAR-10 contains 60000 32x32x3 pictures belonging to 10 classes, with 50000 training images and 10000 test images. CIFAR-100 is similar to CIFAR-10, but it includes 100 classes of 600 pictures, 500 images for the training set and 100 for the test set. The SVHN (Street View House Numbers) dataset contains 10 classes of images, 73257 images for the training set and 26032 images for the test set, with the same image dimensions as CIFAR-10. The SVHN images are not as complex as the CIFAR-10 dataset, but they are also colored images with some the background noise.

4.2 Evaluation metrics

We adopt four classification metrics to evaluate the proposed ER-Caps and the compared models. The accuracy represents the fraction of the correctly predicted classes. The precision is the ratio of the true positive observations to the total predicted positive ones. The recall describes the proportion of the correctly predicted positive samples to the total actual positive samples. Finally, the F1-score measures the misclassified classes. These metrics are defined as:

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \tag{11}$$

$$Precision = \frac{TP}{TP+FP} \tag{12}$$

$$Recall = \frac{TP}{TP+FN} \tag{13}$$

$$F1 - score = 2 \times \frac{Recall \times Precision}{Recall+Precision} \tag{14}$$

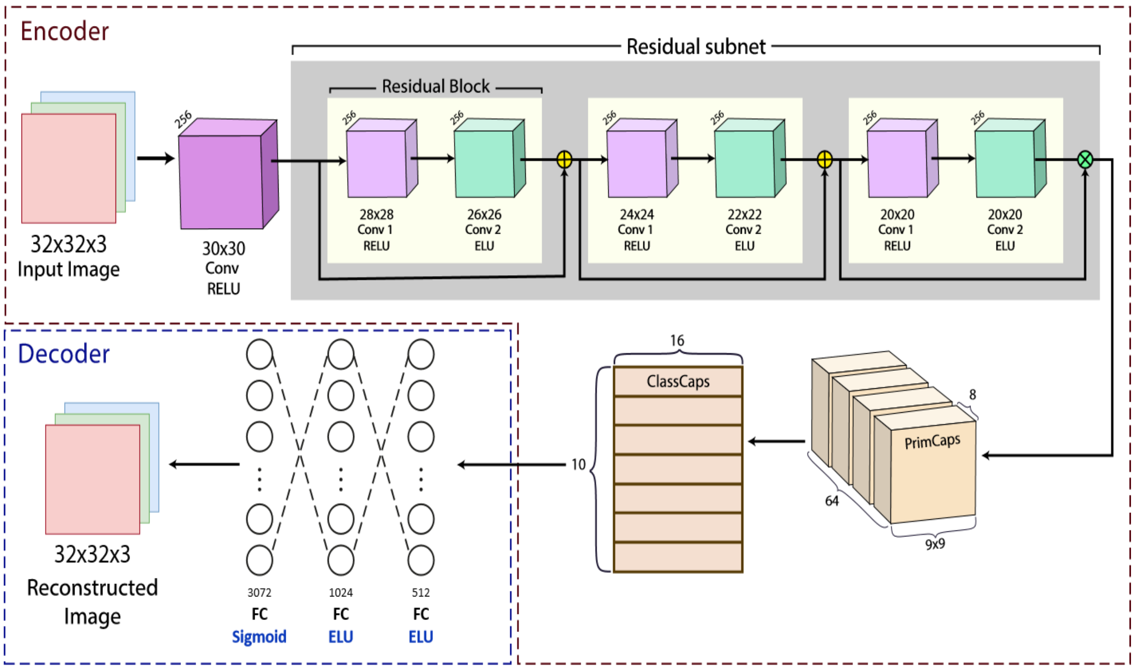

Figure. 5 The architecture of ER-Caps

TP and *TN* represent the number of true positive observations and true negative observations respectively, while *FP* and *FN* mean the number of false positives and false negatives. We use the loss to measure how good the prediction model performs in terms of predicting the intended result. The loss is same as [6].

4.3 System setup

The environment of our experiments is Windows 10, Python, Pytorch, 8-core CPU, NVIDIA GeForce RTX 3070 GPU and 16GB of memory. In our experiments, we use a mini-batch of 40 samples for SVHN and CIFAR-10 datasets and 10 samples for the CIFAR-100 dataset to avoid running out of memory. We use the Adam optimizer [23] with an initial learning rate of 0.001. The number of training epochs is 100 for CIFAR-100 and 70 for CIFAR-10 and SVHN. We set the number of iterations to 3 to fine-tune the routing coefficient. We set α of the ELU function to 1 and the dropout rate to 0.05. We adopt normalization for all datasets before the training and the test. We fixed various parameters based on previous experience. All the aforementioned parameters were set to achieve a good classification accuracy and maintain it across all datasets. We train our models on all the datasets without using any special weight initialization method, data augmentation, ensemble averaging or other training tricks. We repeat each experiment 3 times on each dataset.

The baseline-CapsNet processes the input image by a convolutional layer, primary capsule layer, and class capsule layer, which is similar to the original CapsNet [6]. The first convolutional layer uses 256 convolutional filters with size 9x9, stride of 1 and RELU activation function. The primary capsule layer is a convolutional capsule layer that applies 8 9x9 convolutional kernels and a stride of 2 to build 64 capsules with 8 dimensions at each pixel location.

4.4 Ablation study

In this section, we aim to prove the effectiveness of employing the multiplication gate, using the RELU and ELU activation units alternately, varying the number of residual blocks, and applying the small convolution kernels. All ablation experiments are carried out on the CIFAR-10 dataset.

We first evaluate our framework using different gate combinations for the residual subnet of our ER-Caps network. We define eight models with three residual blocks before the PrimCaps layer. They all respect the same architecture, but use different gates to inject the residual output of the previous block to the output of the next block as described in Fig. 4.

Table 1 shows the comparison of the eight models' performance, where "x" represents the element-wise multiplication gate and "+" represents the addition gate. When the "x" gate is used between all the residual blocks, the model gets a lower accuracy compared to the other models. Consecutive "x" gates could not select better features while mixed gates improve the accuracy. Among the eight combinations,

Table 1. Comparison among different gate combinations for the Residual subnet on CIFAR-10 dataset

Model	Accuracy	Loss
xxx (a)	79.8%	0.316
+xx (b)	84.57%	0.254
x+x (c)	85.03%	0.232
xx+ (d)	81.04%	0.281
++x (e)	**87.83%**	**0.204**
x++ (f)	84.09%	0.233
+x+ (g)	84.02%	0.246
+++ (h)	86.81%	0.226

Table 2. Comparison between ER-Caps and No-res on the CIFAR-10 dataset

	Accuracy	Convergence epoch
No-res	86.25%	46
ER-Caps	**87.83 %**	**36**

Table 3. Comparison of ER-Caps with small convolution kernels for PrimCaps layer on the CIFAR-10 dataset

	kernel size	Parameters (Million)	Accuracy
Our models	9x9	17.92M	87.39%
	6x6	12.02M	87.85%
	4x4	9.4M	**88.54**%
	3x3	8.48M	87.79%

Table 4. Performance comparison of three residual blocks and four residual blocks for ER-Caps on the CIFAR-10 dataset

Model	Accuracy	Parameters
3 residual blocks ++x	88.54%	9.4M
4 residual blocks ++xx	86.9%	10.58M
4 residual blocks ++x+	85.62%	
4 residual blocks ++++	87.19%	
4 residual blocks +++x	88.34%	

the best performing network is the one incorporating two additional gates and one element-wise multiplication gate respectively (++x) to connect the residual block input to its output. Its performance has increased by 8 % compared to model (a). It even has exceeded model (h) that uses "+" gate between all residual blocks, as the regular ResNet does. The multiplication gate in the last residual block can select better features and thereby improves the performance of the residual subnet. We adopt model (e) in the following experiments, and we call it ER-Caps.

Second, to illustrate the power of the residual subnet on the network performance, we create a model (No-res) respecting the same architecture of ER-Caps without using the residual connection. Table 2 demonstrates that the ER-Caps gets the higher accuracy of 87.83 % compared to 86.25 % of the No-res and that it converges faster. This demonstrates that only stacking the convolution layer before the primary capsules is not enough to enhance the model performance.

Third, we use small convolutional kernels for the PrimCaps layer not only to reduce the computational costs by reducing the number of parameters but also to enhance the model performance. Table 3 displays the comparison of the ER-Caps network performance on the CIFAR-10 dataset with different convolution kernel sizes for the PrimCaps layer. The results show that the 4x4 kernel size can improve the network performance while at the same time remarkably reducing the number of training parameters.

Fourth, previous studies in ResNet have shown that multiple residual blocks can improve image classification performance. In order to investigate the effect of adding more residual blocks to our model, we train our model with four residual blocks instead of three. Table 4 displays the accuracy of ER-Caps with three residual blocks (++x) compared to four residual blocks using different gates. Unexpectedly, the accuracy has dropped with four residual blocks. The (+++x) network is the only model with four residual blocks that achieved a very close accuracy of 88.34 % compared to the (++x) model's 88.54 %. We also observe that with four residual blocks, the number of the training parameters becomes higher, it reaches 10.58M. Our intuition is to increase CapsNet classification performance and reduce the number of training parameters to decrease the model complexity. Our proposed model (++x) with only three blocks can learn sufficient feature information while keeping the network simple. From Table 4 and Table 1 we notice that using the multiplication gate in the last residual block leads to greater performance. In the first residual blocks, multiplication gates inhibit some features to pass, which influences the model performance negatively, while in the last residual block, they act as a control tower to select features mapped to the PrimCaps layer.

Last, the ELU activation is known for its capacity to speed up the training, overcome the vanishing grading issue, and improve the performance of the network compared to RELU. We test the performance of our model with the RELU activation (ER-Caps*) for all convolution layers of the network except the convolution of PrimCaps and we do the same with ELU activation (ER-Caps**). These two models are trained with 100 epochs. The results are displayed in Table 5 and Fig. 6. Table 5 presents a comparison of model accuracy and the training time for each epoch on the CIFAR-10 dataset, the results show that ER-Caps gets the higher accuracy and the lower loss over the other models. Fig.6 presents the

Received: July 9, 2022. Revised: August 25, 2022.

Table 5. Comparison of test accuracy for the model with RELU and ELU activation for residual convolution layers on the CIFAR-10 dataset

Model	Accuracy	Training time (minute/epoch)
ER-Caps	**88.54%**	**05:22**
ER-Caps*	88.38%	06:04
ER-Caps**	85.55%	06:15

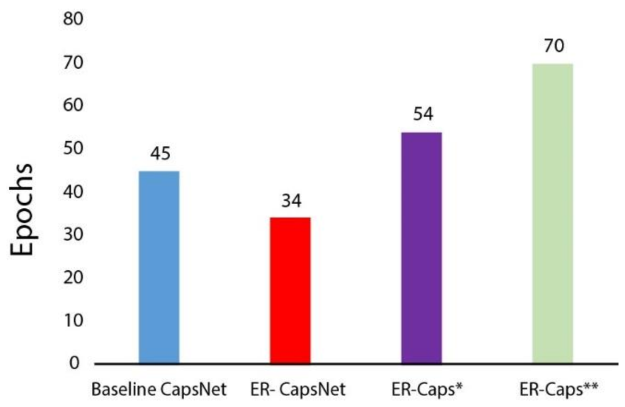

Figure. 6 Models convergence on the CIFAR-10 dataset according to the epochs

number of epochs needed for each model to achieve the best test accuracy on the CIFAR-10 dataset. From Fig. 6, it can be seen that 34 epochs were needed for ER-Caps to reach the best test accuracy, whereas 45 epochs were needed for the baseline CapsNet, 54 for ER-Caps*, and 70 for ER-Caps**. It can be observed that the performance of ER-Caps that uses RELU activation for Conv1 and ELU for Conv2 is better than adopting ELU or RELU for all activation. The ER-Caps framework performs best in terms of accuracy and training time required per epoch.

4.5 Performance evaluation

In this section, we start by comparing the accuracy and the loss of the ER-Caps and baseline CapsNet models during training and test on SVHN, CIFAR-10, and CIFAR-100 datasets (Fig. 7). Then, we compare the test accuracy of our proposed model with the state-of-the-art based CapsNet models (Table 6) on the same datasets, and we analyse the number of training parameters and epochs (Table 7) on the CIFAR-10 dataset.

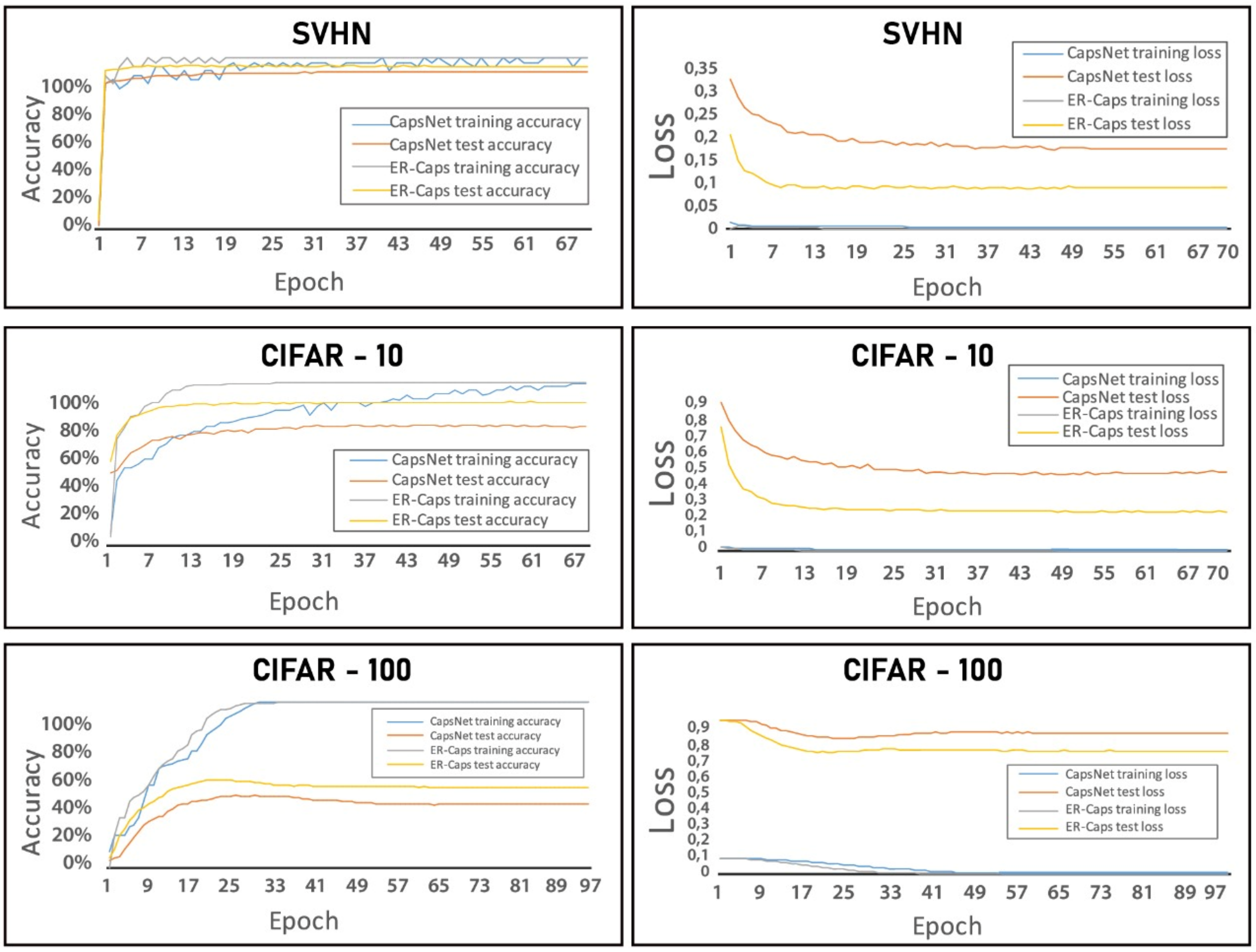

Figure. 7 Learning curves of ER-Caps and CapsNet on the benchmark datasets

Table 6. Classification accuracy of ER-Caps and state-of-the-art CapsNet-based models on CIFAR-10, CIFAR-100, and SVHN datasets

Model	CIFAR 10 (%)	CIFAR 100 (%)	SVHN(%)
Inverted dot product [36]	82.13	–	–
Baseline CapsNet	71.91	43.74	92.33
	89.40 (7-ensemble)	–	95.7
Fast Inference [24]	70.33	–	–
Max–min [25]	75.92	–	–
Fsc-CapsNet [26]	78.73	–	–
Residual CapsNet [27]	75.62	–	–
STAR-Caps [28]	91.23	67.66	–
RS-CapsNet [13]	89.81	64.14	96.50
Residual Capsule Network [14]	84.16	–	–
HitNet[29]	73.30	–	94.50
Quaternion CapsNet [30]	82.21	–	95.37
Multi-lane [31]	76.79	–	–
MS-CapsNet [32]	75.70	–	–
Gabor CapsNet [33]	85.24	68.17	–
DenseCaps[8]	89.41	–	95.99
DA-CapsNet[34]	85.47	–	94.82
ER-Caps	88.54	52.88	95.67

Table 7. Analysis of different Capsule Network models' parameters on the CIFAR-10 dataset

Model	Parameters	Training epochs
Baseline CapsNet	14.43M	70
7-ensemble CapsNet[6]	101.5 M	500
Multi-lane [31]	14.25 M	20
MS-CapsNet [32]	11.2 M	50
DenseCaps [8]	16 M	100
HitNet [29]	8.89M	250
Residual Capsule Network [14]	11.86M	120
ER-Caps	9.4 M	70

Table 8. Precision, recall, and F1-score of the baseline CapsNet and ER-CapsNet on the CIFAR-10 dataset

Models	Precision	Recall	F1-score
Baseline CapsNet	74.1%	74.1%	74.1%
ER-Caps	88.1%	88.3%	88.2%

ER-Caps needs fewer epochs to converge (Fig. 7) than the baseline CapsNet, except on the CIFAR-100 dataset. Moreover, it can be observed that when the two networks converge, our model attains the higher accuracy and the smaller loss. It surpasses the baseline CapsNet accuracy by 17 %, 9 %, and 4 % on CIFAR-10, CIFAR-100, and SVHN datasets, respectively.

In Table 6, '-' indicates that the results for the corresponding dataset are not given in the related experiments of the model. ER-Caps outperforms most of the other CapsNet models on CIFAR-10 and gets competitive results on the SVHN. STAR-Caps [28] and RS-CapsNet [13] have slightly better accuracies on CIFAR10. However, both models use pooling to reduce the number of parameters and filter the background, which is criticized in [35, 6], since pooling causes loss of some useful spatial information and reduction of the network equivariance. The 7-Ensemble of CapsNet and DenseCaps uses a tremendous number of parameters and only reaches a slightly higher accuracy. From the single models trained on CIFAR-10, ER-Caps gets the second fewest parameters (Table 7) beneath the HitNet [29], but it highly outruns the HitNet performance. In addition, our framework reduces the parameter number by 34.85 % on the CIFAR-10 compared to the baseline CapsNet. Despite the Multilane [24] needing the fewest training epochs, its number of parameters is as high as the baseline CapsNet and the classification results are poor.

In order to validate our model performance, we compare the precision, the recall, and the F1-score metrics of ER-Caps to the baseline CapsNet on the CIFAR-10 dataset (Table 8). Our model has obvious ascendancy in all classification metrics, exceeding the baseline CapsNet results by about 14 %. The performance disparity between ER-Caps and demonstrates that there are still a lot of aspects for improvement, regardless of the routing method or CapsNet design.

The confusion matrix is computed to evaluate

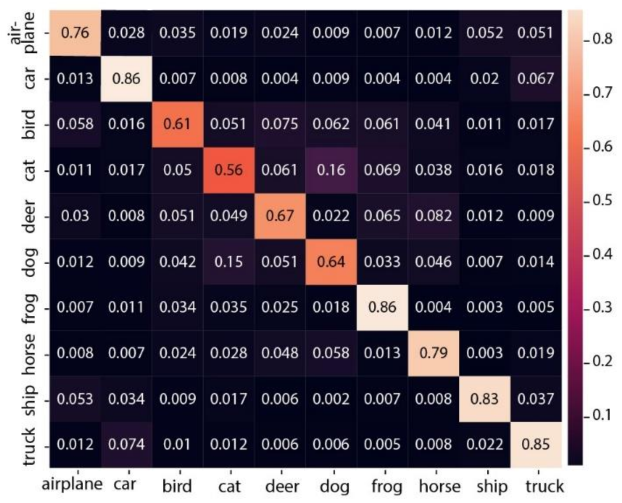

Figure. 8 Confusion matrix of the baseline CapsNet on the CIFAR-10 dataset

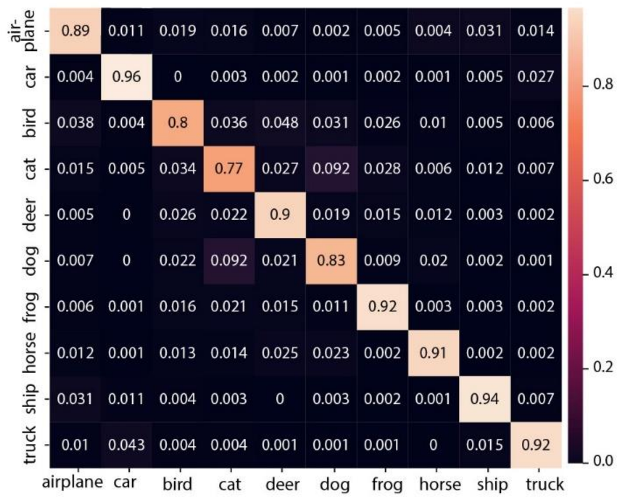

Figure. 9 Confusion matrix of ER-Caps on the CIFAR-10 dataset

ER-Caps results and to compare with the baseline CapsNet. It is a detailed matrix of the model's performance on each class. We use it to examine confusion between multiple classes and analyse classification errors efficiently. The column represents the real labels, while the row represents the predicted labels. Figs. 8 and 9 illustrate the confusion matrix of the baseline CapsNet and the ER-Caps respectively for the CIFAR-10 dataset. The analysis of the confusion matrix for the baseline CapsNet shows that four classes reach an accuracy superior to 80 %. For the other classes, the confusion is large. Four categories achieved an accuracy lower than 60 %. As for the ER-Caps, nine classes achieve an accuracy higher than 80 %, six of which even reach over 90 %. Only the class of 'cat' achieves an

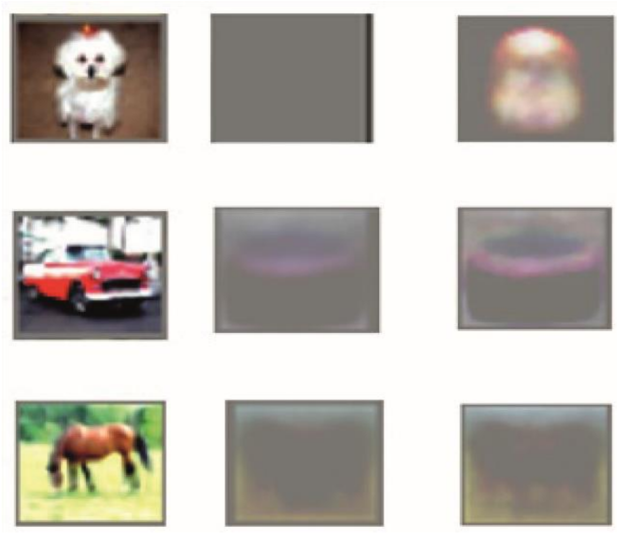

Figure. 10 Comparison among the input images of CIFAR-10 dataset and the reconstructed images of the baseline CapsNet and the ER-Caps

accuracy of 77 % since it can be confused with the 'dog' class. This confusion can be explained by the small image size. It makes the image blurry and therefore, the two classes have similar features, which makes it difficult to distinguish one from the other. Nevertheless, our proposed model, compared to the baseline CapsNet accuracy, has achieved a great improvement of about 20 % higher accuracy.

These numbers show that ER-Caps outperforms the baseline CapsNet and achieves competitive results compared to the other CapsNet models. Our framework based on residual blocks with the ELU activation and the multiplication gate demonstrates that it can enhance classification performance. Furthermore, CapsNet as a new network of deep learning still has many unknowns to work out, which requires patience and time.

4.6 Reconstruction

In order to demonstrate the performance of our proposed decoder, we compare the reconstructed images of the baseline CapsNet with the ones reconstructed by ER-Caps in CIFAR-10 (Fig. 10). The left column (a) of Fig. 10 shows the input images, the middle column (b) is the reconstructed images of baseline CapsNet and the right column (c) is the images reconstructed by our model ER-Caps. The images (c) reconstructed by ER-Caps contain more texture; hence they can more accurately express the posture and texture of a dog, a car, and a horse. It can be seen that the baseline CapsNet cannot reconstruct the dog image, while ER-Caps design has achieved to reconstruct more features of the image. Moreover, ER-Caps is more prominent in focusing more information on the image and processing its characteristics. To the best of our knowledge, this is the first work that applies ELU activation unit on the decoder of CapsNet and it achieves better reconstruction with only three fully connected layers.

5. Conclusion and future work

In this work, we leverage the power of the Residual Network with the best characteristics of the Capsule Network to introduce the ER-Caps network, a framework for complex image classification. Our model improves feature extraction using an improved residual subnet without any pooling to keep the spatial relationships between features and enhance the reconstruction result. ER-Caps performs well compared to the state-of-the-art Capsule Network models on CIFAR-10 datasets. Our model surpasses the baseline CapsNet in terms of accuracy by approximately 17 %, 9 %, and 4 % on CIFAR-10, CIFAR-100, and SVHN respectively, and it reduces the number of parameters remarkably by 34.26 %. The concept of the Capsule Network and dynamic routing used to connect capsules requires a higher computational cost. Moreover, it is challenging to apply the CapsNet to high-resolution images and datasets with a high number of classes. Therefore, we intend to improve the DR algorithm of ER-Caps and enhance its computational efficiency, and we plan to boost its performance on more complex large datasets such as ImageNet in the future.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

The contribution of the authors is elaborated as follows:

- Omaima El Alaoui-Elfels: Conceptualization, implementation, visualization, and writing the paper.
- Taoufiq Gadi: supervision, results' validation, paper reviewing, and editing.

All authors read and approved the final manuscript.

References

[1] J. C. Rangel, J. M. Gomez, C. R. Gonzalez, I. G. Varea, and M. Cazorla, "Semi-supervised 3D object recognition through CNN labeling", *Applied Soft Computing*, Vol. 65, pp. 603-613, 2018.

[2] H. Han, Y. Li, and X. Zhu, "Convolutional neural network learning for generic data classification", *Information Sciences*, Vol. 477, pp 448-465, 2019.

[3] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review", *Neural Computation*, Vol. 29, No. 9, pp. 2352-2449, 2017.

[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for largescale image recognition", In: *Proc. of Third International Conference on Learning Representations*, 2015.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition", In: *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770-778, 2016, doi: 10.1109/CVPR.2016.90.

[6] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules", *Advances in Neural Information Processing Systems*, Vol. 30, pp. 3856-3866, 2017.

[7] M. K. Patrick, A. F. Adekoya, A. A. Mighty, and B. Y. Edward, "Capsule networks–a survey", *Journal of King Saud University-Computer and Information Sciences*, Vol. 34, No. 1, pp. 1295-1310, 2022.

[8] G. Sun, S. Ding, T. Sun, C. Zhang, and W. Du, "A novel dense capsule network based on dense capsule layers", *Applied Intelligence*, Vol. 52, No. 3 pp. 3066-3076, 2022.

[9] A. Krizhevsky and G. E. Hinton, "Learning multiple layers of features from tiny images", *Master's Thesis*, University of Tront, 2009.

[10] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning", *Advances in Neural Information Processing Systems Workshop Deep Learning and Unsupervised Feature Learning*, 2011.

[11] Y. Xiong, G. Su, S. Ye, Y. Sun, and Y. Sun, "Deeper capsule network for complex data", In: *Proc. of International Joint Conference on Neural Networks*, pp. 1-8, 2019.

[12] J. Gugglberger, D. Peer, and A. R. S´anchez, "Training deep capsule networks with residual connections", In: *Proc. of International Conference on Artificial Neural Networks*, pp. 541-552, 2021.

[13] S. Yang, F. Lee, R. Miao, J. Cai, L. Chen, W. Yao, K. Kotani, and Q. Chen, "RS-CapsNet: An advanced capsule network", *IEEE Access*, Vol. 8, pp. 85007-85018, 2020, doi: 10.1109/ACCESS.2020.2992655.

[14] S. B. S. Bhamidi and M. E. Sharkawy, "Residual capsule network", In: *Proc. of 10th Annual*

Ubiquitous Computing, Electronics & Mobile Communication Conference, pp. 0557-0560, 2019.

[15] O. E. A. Elfels and T. Gadi, “EMG-CapsNet: Elu Multiplication Gate Capsule Network for Complex Images Classification”, In: *Proc. of International Conference on Soft Computing and Pattern Recognition*, Vol. 417, pp. 97-108, 2021.

[16] O. E. A. Elfels and T. Gadi, “TG-CapsNet: Two Gates Capsule Network for Complex Features Extraction”, *Journal of Information Assurance and Security*, Vol. 17, pp. 156-164, 2022.

[17] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks”, In: *Proc. of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249-256, 2010.

[18] R. K. Srivastava, K. Greff, and J. Schmidhuber, “Highway networks”, In: *Proc. of the 32nd International Conference on Machine Learning Deep Learning workshop*, 2015.

[19] V. Nair and G. E. Hinton, “Rectified linear units improve restricted Boltzmann machines”, In: *Proc. of the Twenty-seventh International Conference on International Conference on Machine Learning*, pp. 807-814, 2010.

[20] A. Shah, E. Kadam, H. Shah, S. Shinde, and S. Shingade, “Deep residual networks with exponential linear unit”, In: *Proc. of the Third International Symposium on Computer Vision and the Internet*, pp. 59-65, 2016.

[21] J. Li, F. Fang, K. Mei, and G. Zhang, “Multi-scale residual network for image super-resolution”, In: *Proc. of the European Conference on Computer Vision*, pp. 517-532, 2018.

[22] K. Jaeyoung, J. Sion, P. Eunjeong, and C. Sungchul, “Text classification using capsules Neurocomputing”, Vol. 376, pp. 214-221, 2020.

[23] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization”, In: *Proc. of the International Conference on Learning Representations (Poster)*, 2015.

[24] Z. Zhao, A. Kleinhans, G. Sandhu, I. Patel, and K. P. Unnikrishnan, “Fast Inference in Capsule Networks Using Accumulated Routing Coefficients”, *The Computing Research Repository*, 2019.

[25] Z. Zhao, A. Kleinhans, G. Sandhu, I. Patel, and K. Unnikrishnan, “Capsule Networks with Max-Min Normalization”, *The Computing Research Repository*, 2019.

[26] T. Han, R. Sun, F. Shao, and Y. Sui, “Feature and spatial relationship coding capsule network”, *Journal of Electronic Imaging*, Vol. 29, No. 2, p. 023004, 2020.

[27] X. Ding, N. Wang, X. Gao, J. Li, and X. Wang, “Group reconstruction and max-pooling residual capsule network”, In: *Proc. of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, Macao, China, pp. 2237-2243, 2019.

[28] K. Ahmed and L. Torresani, “STAR-Caps: Capsule Networks with Straight Through Attentive Routing”, In: *Proc. of Advances in Neural Information Processing Systems*, Vancouver, BC, Canada, pp. 9098-9107, 2019.

[29] A. Deliege, A. Cioppa, and M. V. Droogenbroeck, “Hitnet: a neural network with capsules embedded in a hit-or-miss layer, extended with hybrid data augmentation and ghost capsules”, *The Computing Research Repository*, 2018.

[30] Y. Zhao, T. Birdal, J. E. Lenssen, E. Menegatti, L. Guibas, and F. Tombari, “Quaternion equivariant capsule networks for 3D point clouds”, In: *Proc. of European Conference on Computer Vision*, Glasgow, UK, pp. 1-19, 2020.

[31] S. Chang and J. Liu, “Multi-lane capsule network for classifying images with complex background”, *IEEE Access*, Vol. 8, pp. 79876-79886, 2020 , doi: 10.1109/ACCESS.2020.2990700.

[32] C. Xiang, L. Zhang, Y. Tang, W. Zou, and C. Xu, “MS-CapsNet: A novel multi-scale capsule network”, *IEEE Signal Processing Letters*, Vol. 25, No. 12, pp. 1850-1854, 2018, doi: 10.1109/LSP.2018.2873892.

[33] M. A. Ayidzoe, Y. Yu, P. K. Mensah, J. Cai, K. Adu, and Y. Tang, “Gabor capsule network with preprocessing blocks for the recognition of complex images”, *Machine Vision and Applications*, Vol. 32, No. 4, pp. 1-16, 2021.

[34] W. Huang and F. Zhou, “DA-CapsNet: dual attention mechanism capsule network”, *Scientific Reports*, Vol. 10, No. 1, pp. 1-13, 2020.

[35] G. E. Hinton, A. Krizhevsky, and S. D. Wang, “Transforming Auto-Encoders”, In: *Proc. of International Conference on Artificial Neural Networks*, Espoo, Finland, pp. 44-51, 2011.

[36] Y. H. H. Tsai, N. Srivastava, H. Goh, and R. Salakhutdinov, “Capsules with Inverted Dot-Product Attention Routing”, In: *Proc. of the International Conference on Learning Representations*, Addis Ababa, Ethiopia, 2020.