CapsNet vs CNN : Analysis Of The Effects Of Varying Feature Spatial Arrangement

Ugenteraan Manogaran*, Ya Ping Wong**, and Boon Yian Ng

Faculty of Computing and Informatics, Multimedia University, Persiaran Multimedia, 63100 Cyberjaya, Selangor, Malaysia

Abstract. Despite the success over the recent years, convolutional neural network (CNN) has a major limitation of the inability to retain spatial relationship between learned features in deeper layers. Capsule network with dynamic routing (CapsNet) was introduced in 2017 with a speculation that CapsNet can overcome this limitation. In our research, we created a suitable collection of datasets and implemented a simple CNN model and a CapsNet model with similar complexity to test this speculation. Experimental results show that both the implemented CNN and CapsNet models have the ability to capture the spatial relationship between learned features. Counterintuitively, our experiments show that our CNN model outperforms our CapsNet model using our datasets. This implies that the speculation does not seem to be entirely correct. This might be due to the fact that our datasets are too simple, hence requiring a simple CNN model. We further recommend future research to be conducted using deeper models and more complex datasets to test the speculation.

Keywords: CapsNet, Convolutional Neural Network , Spatial Relationship

1 Introduction

Ever since Krizhevsky et al. [1] demonstrated the outstanding performance of a convolutional neural network (CNN) model on ImageNet, CNN has become the center of attraction for computer vision researchers to solve problems such as image segmentation, object detection, object localization, image classification, and image retrieval. Some of the well-known CNN models are AlexNet [1], GoogleNet [2], VGG-16 [3], YOLO [4], and RCNN [5].

A significant advantage of CNN is its ability to maintain translation invariance for feature detection [6]. This means that the position of an object that is known by CNN in the input image does not affect the performance of CNN. CNN is able to recognize the object regardless of its position in the image. This is achieved by the use of pooling layers in CNN which are also responsible for reducing the size of feature maps.

* E-mail : m.ugenteraan15@gmail.com
** E-mail : ypwong@mmu.edu.my

However, this advantage of CNN inevitably leads to its drawbacks. Two major drawbacks are the lack of rotational invariance [8] and the failure to retain spatial relationships between features [7]. The failure to be invariant to rotations would cause CNN to produce false negatives when an object that is known by the network is rotated to a certain extent. On the other hand, the failure of CNN to retain spatial relationships between features would cause the network to produce false positives [9]. Even though CNNs can achieve state-of-the-art results on many challenges despite these drawbacks, the drawbacks can become serious concerns in applications such as in security systems.

To overcome the lack of rotational invariance, augmenting the training data by rotations became a standard practice in the training of CNN models [10]. However, the training time would also increase tremendously. Therefore, in order to solve the drawbacks of CNN, Sabour, S. et al. [11] proposed a novel neural network architecture known as capsule network with dynamic routing (CapsNet). Unlike CNN, CapsNet produces outputs in the form of vectors instead of scalars. This allows CapsNet to retain the properties of an object such as rotation, skewness, and thickness.

It has also been explicitly speculated in a number of research papers [11],[9], [10], that CapsNet would be able to retain the spatial relationships between features in contrast to CNN. In other words, features of an object in the wrong places such as a face with eyes below the nose instead of above the nose would still be a face to CNN. However, CapsNet is speculated to be able to identify it as a non-face.

To the best of our knowledge, there has been no research literature that has produced comprehensive experiments and analysis on this speculation. In this paper we present our experiments and analysis to test both CapsNet and CNN on this speculation. We generated our own dataset for the purpose of our experiment and implemented a CapsNet and a CNN model. Evaluation of these models were done on the generated dataset in such a way that the speculation is tested.

2 Related Works

The concept of CNN was first proposed by LeCun et al. [12] in 1989. However, due to the lack of computational power and availability of dataset, it was not until recent years that researchers are able to develop feasible models utilizing modern high-performance computers. One notable breakthrough was the work of Krizhevsky et al. [1] which achieved state-of-the-art performance in the ImageNet challenge [13] in 2012. Since then, countless researches have been conducted to develop more advanced CNN models to be used successfully in real-world applications such as speech recognition [14], gait recognition [15], steering controls in self-driving cars [16], human crowd detection [17], and medical image segmentation [18].

Despite successful demonstrations of CNN models, one of the pioneers, Geoffrey Hinton argued that the current CNNs “are misguided in what they are trying

to achieve" [19], due to the use of pooling layers for subsampling in CNN models. The models lose the ability to compute precise spatial relationships between learned features in the deeper layers. When a pooling layer is used in between convolutional layers, only the most active neuron in a local region of a feature map would be retained while the rest of the neurons are disregarded. Such disregard of neurons causes the loss of spatial information of the features.

Furthermore, due to the use of scalars instead of vectors, properties of features such as orientation, thickness, and skewness are lost. Therefore, Hinton, G. E. et al. [19] proposed to group neurons together as vectors and use them to represent the features of an object. These vectors are called capsules.

In 2017, Hinton and his team [11] proposed an architecture called capsule networks with dynamic routing (CapsNet) that performed better than CNN on the MNIST dataset. It achieved state-of-the-art results in MNIST with only 0.25% of test error. The CapsNet model which achieved 99.23% on the expanded MNIST set were able to reach an accuracy of 79% on affNIST test set while a CNN that achieved 99.22% accuracy on the expanded MNIST test set only achieved 66% accuracy on affNIST test set. This proves that CapsNet is more robust to affine transformations.

We have implemented CapsNet based on the original research paper [11] and through the reconstruction network as mentioned in the paper, it can be seen that CapsNet preserves the properties of the features well as shown in Figure 1.

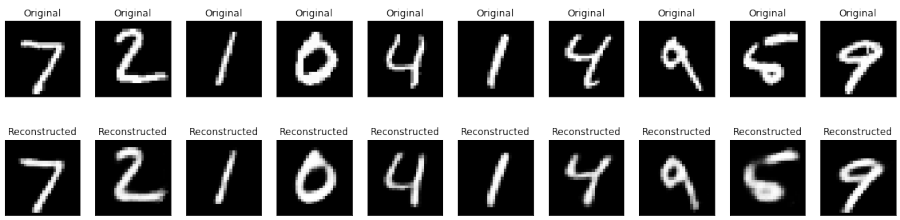

Fig. 1: Original (Top Row) vs Reconstructed (Bottom Row) images.

Following the success of CapsNet on MNIST, studies have been conducted to push the capability of CapsNet. LaLonde, R. et al. [20] proposed a novel architecture called SegCaps which is based on CapsNet to perform image segmentation. SegCaps outperformed previous state-of-the-art networks even though SegCaps had lower number of parameters on the LUNA16 subset of the LIDC-IDRI database. In a different research [7], a CapsNet model outperformed a CNN model with similar complexity in Human Action Recognition task on KTH and UFC-sports dataset.

One of the speculated properties of a CapsNet model is its ability to retain spatial relationships between learned feature unlike a CNN model [11],[9], [10]. In other words, the relative positions of features are insignificant to a CNN model. This causes a CNN model to produce false positives such as labelling an image

of a face with eyes below the nose as a face. In contrast to CNN, CapsNet is speculated to be able to avoid such false positives. However, to date, no studies have been conducted to test this speculation. In this paper, we seek to test this speculation to gain deeper insights into CapsNet.

3 Methodology

We implemented a CNN model and a CapsNet model for this study. In general, a CNN model consists of several convolutional layers with pooling layers between them followed by fully-connected layers as explained in section 3.2. A CapsNet model consists of several convolutional layers without any pooling layers followed by a primary capsule layer and a digit capsule layer as explained in section 3.3. Both of the models were designed to have the same number of layers in order for them to be comparable.

In order to test the speculation, we need to design a dataset in such a way that there are two classes of images containing the same features but the features from different classes have different spatial arrangements. Training our models directly on such a dataset may not yield any insight into the models as the models will learn to identify the shape of the entire objects successfully instead of the distinct features as intended.

Therefore, we prepared two groups of datasets whereby the first group contains images of only distinct features while the second group contains objects formed by the composition of the features. Our models are first trained on the dataset from Group 1. Once the training is completed, the weights of the convolutional layers in both models will be frozen while the weights of the rest of the layers will be re-trained on the dataset from Group 2. This will ensure that our models learn the distinct features first before learning to identify the objects using the learned features. This strategy is known as transfer learning.

Below we describe in detail regarding the dataset generation, testing of convolutional neural network model, and testing of capsule network with dynamic routing model. Since our datasets only consist of objects with simple features, relatively simple models should be sufficient to achieve good accuracy on the evaluations.

3.1 Dataset Generation

Our dataset consists of two groups. Figure 2 shows samples from Group 1, which contains images of arrows and non-arrows. Figure 3 shows samples from Group 2, which contains images of equilateral triangles and rectangles. Each image is of 64 x 64 pixels.

We chose to use generated images in our dataset because there is too much ambiguity in real-life images. Furthermore, simple polygon objects were chosen as they are well-defined mathematically. This would enable us to test out different ideas on how to design our experiments. Table 1 shows the organizations of our datasets.

(a) Fig. 2 (a) Arrows

(b) Fig. 2 (b) Non-Arrows

Fig. 1: Samples from Group 1

(a) Fig. 3 (a) Triangles

(b) Fig. 3 (b) Rectangles

Fig. 2: Samples from Group 2

Dataset	**Description**	**Subsets**	**Number of Images**
Group 1	Contain images of arrows and non-arrows	Training Set 1	500
		Training Set 2	1000
		Training Set 3	2000
		Testing Set	2000
Group 2	Contain images of equilateral triangles and rectangles.	Training Set 1	500
		Training Set 2	1000
		Training Set 3	2000
		Testing Set	2000

Table 1: Organizations of our datasets

3.2 Convolutional Neural Network (CNN)

We implemented a CNN model using Tensorflow that has 3 convolutional layers and 2 fully-connected layers. Max pooling layer was implemented after each convolutional layer. Rectified Linear Unit (ReLU) was used as the activation function on every layer except for the output layer. Dropouts were also applied to prevent the model from overfitting.

As mentioned in the methodology section above, we carried out our experiment by first training our CNN model on the dataset from Group 1 and once the model was trained, we re-trained the weights of the fully-connected layers of the model on the dataset from Group 2 while freezing the weights of the convolutional layers. After each training, the trained model was evaluated on the testing sets from their respective groups.

3.3 Capsule Network with Dynamic Routing (CapsNet)

Our CapsNet model was also implemented using Tensorflow. We implemented 3 convolutional layers, 1 primary capsule layer and 1 digit capsule layer. The architecture of CapsNet is similar to the original paper [11] except that we added an extra convolutional layer and we used 16-D capsules on primary capsule layer and 32-D capsules in digit capsule layer. We used the activation function as proposed in the paper. To prevent overfitting, a reconstruction network [11] was used. There were no pooling layers used.

Similar to CNN, our experiment was carried out by first training our CapsNet model on the dataset from Group 1 and once the model was trained, we re-trained the weights of the primary capsule layer and digit capsule layer of the model on the dataset from Group 2 while freezing the weights of the convolutional layers. The trained model was evaluated on the testing sets from their respective groups after each training.

4 Experimental Results and Discussion

The trainings and the evaluations of the models were performed on a workstation running on Ubuntu 16.04 equipped with 16 GB RAM, Core i7-6700K processor, and two NVIDIA GTX1080Ti GPUs. The models were trained using the training subsets and were evaluated on their respective testing sets. The evaluation results in terms of accuracy (acc), precision (prec), recall (rec) and F1-score (F1) for both models are shown in Table 2 below.

	Subset 1				**Subset 2**				**Subset 3**			
(%)	Acc	Prec	Rec	F1	Acc	Prec	Rec	F1	Acc	Prec	Rec	F1
Group 1 Triangles vs Rectangles	88.4	88.7	87.2	87.9	91	92.5	89	90.7	90.4	92.8	87.6	90.1
Group 2 Arrows vs Non-Arrows	67.6	70.6	59.7	64.7	77.8	86.6	62.1	72.3	80.4	83.9	75.1	79.3

Table 2 (a): Evaluation Results for CapsNet

	Subset 1				Subset 2				Subset 3			
(%)	Acc	Prec	Rec	F1	Acc	Prec	Rec	F1	Acc	Prec	Rec	F1
Group 1 Triangles vs Rectangles	98.5	99.4	97.1	98.2	99.3	99.5	98.9	99.2	99.6	99.8	99.2	99.6
Group 2 Arrows vs Non-Arrows	92.6	95	87.2	90.9	95.8	95.2	95.8	95.5	96.6	96.8	92.5	94.6

Table 2 (b): Evaluation Results for CNN

All the images were shuffled in their respective sets and normalized before they were used for training and evaluation purposes. From Table 2 (a), it is evident that CapsNet is able to determine whether a given image contains an arrow or non-arrow by computing the spatial relationship between the learned features. It can also be seen in Table 2 (b) that CNN has achieved near-perfect accuracies. This is due to the fact that the generated datasets do not contain any real-world noise.

We expected the CNN model to perform worse than CapsNet based on the speculation stated earlier but it can be seen from the results that CNN actually performed better than CapsNet. This might be due to the dataset being too simple hence not requiring a deeper CNN model.

The use of pooling layers in between the convolutional layers should cause the loss of spatial information of the features in a CNN. Hence, it might be the case where our model is not deep enough. We expected our CNN model to perform poorly to at least some degree due to the use of 3 pooling layers but based on the results this is not the case. We chose a CNN model with only 3 pooling layers due to the simplicity of the datasets. From the results, it is evident that the problem of retaining the spatial relationship between features is not a serious issue for a relatively shallow model such as a model with only 3 pooling layers. However, it is questionable whether a deeper CNN model would perform well on a more complex dataset or not.

In our experiment, the objects in the images are formed by composing simple features. There is only one equilateral triangle and one rectangle in every image. Given the success of CNN, identifying such generated simple objects without real-world noise is rather a trivial task for CNN. This could be another reason for the high accuracy that CNN models have achieved in this experiment despite the use of pooling layers. Our implementations are publicly available in this github link. [1]

[1] https://github.com/MMU-VisionLab/CapsNet-vs-CNN

5 Conclusions and Future Work

In this work, we have designed an experiment to test the speculation that CapsNet is able to retain the spatial relationship between features better than CNN. In order to carry out the experiment, we have generated our own datasets.

From our results, both the shallow CNN and CapsNet models have shown the capability to retain the spatial relationship between features. However, the speculation that CapsNet is able to retain spatial relationship between features better than CNN does not seem to be true for shallow models on simple datasets. It is still uncertain whether this speculation is true for deeper models on more complex datasets and on noisy datasets.

Considering the fact that CNN has been developed extensively since its invention in 1989 [12], it is possible that our experiment was too simple for CNN. CapsNet on the other hand, is still at a rudimentary stage and the fact that its performance level is close to CNN in this experiment means that CapsNet has great potential.

Future research in this area should consider the usage of more complex features to represent the objects in the datasets and deeper models in order to further understand the capabilities and limitations of these models. Gaining deeper insights on these models in the retention of spatial relationship between features will guide future developments in a better way.

6 Acknowledgement

The authors are grateful to the Ministry of Higher Education, Malaysia and Multimedia University for the financial support provided by the Fundamental Research Grant Scheme (MMUE/150030) and MMU Internal Grant Scheme (MMUI/170110).

References

1. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097-1105).
2. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1-9).
3. Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
4. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 779-788).
5. Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 580-587).

6. Nair, P., Doshi, R., & Keselj, S. (2018). Pushing the limits of capsule networks. Technical note.
7. Algamdi, A. M., Sanchez, V., & Li, C. T. (2019). Learning temporal information from spatial information using CapsNets for human action recognition. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 3867-3871).
8. Xi, E., Bing, S., & Jin, Y. (2017). Capsule network performance on complex data. arXiv preprint arXiv:1712.03480.
9. Xiang, C., Zhang, L., Tang, Y., Zou, W., & Xu, C. (2018). MS-CapsNet: A novel multi-scale capsule network. *IEEE signal processing letters*, 25(12), 1850-1854.
10. Chidester, B., Do, M. N., & Ma, J. (2018). Rotation equivariance and invariance in convolutional neural networks. arXiv preprint arXiv:1805.12301.
11. Sabour, S., Frosst, N., & Hinton, G. E. (2017). Dynamic routing between capsules. In *Advances in neural information processing systems* (pp. 3856-3866).
12. LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4), 541-551.
13. Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition.*
14. Palaz, D., Magimai-Doss, M., & Collobert, R. (2015). Analysis of CNN-based speech recognition system using raw speech as input. In *Sixteenth Annual Conference of the International Speech Communication Association.*
15. Zhang, C., Liu, W., Ma, H., & Fu, H. (2016). Siamese neural network based gait recognition for human identification. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 2832-2836).
16. Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., & Zhang, X. (2016). End to end learning for self-driving cars. arXiv preprint arXiv:1604.07316.
17. Tzelepi, M., & Tefas, A. (2017). Human crowd detection for drone flight safety using convolutional neural networks. In *25th European Signal Processing Conference (EUSIPCO)* (pp. 743-747). IEEE.
18. Milletari, F., Navab, N., & Ahmadi, S. A. (2016). V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *IEEE Fourth International Conference on 3D Vision (3DV)* (pp. 565-571).
19. Hinton, G. E., Krizhevsky, A., & Wang, S. D. (2011). Transforming auto-encoders. In *Proceedings of the 21th International Conference on Artificial Neural Networks-Volume Part I* (pp. 44-51).
20. LaLonde, R., & Bagci, U. (2018). Capsules for object segmentation. arXiv preprint arXiv:1804.04241.