Exploring Capsules

Binghui Peng Runzhou Tao Shunyu Yao
IIIS, Tsinghua University
{pbh15, trz15, yao-sy15}@mails.tsinghua.edu.cn

1 Introduction

Nowadays, convolutional neural networks (CNNs) have received massive success in image processing[1]. From layer to layer, local features of an image are combined to obtain global ones. However, due to the use of methods like pooling, some local information cannot be passed to upper layers, which causes problems.

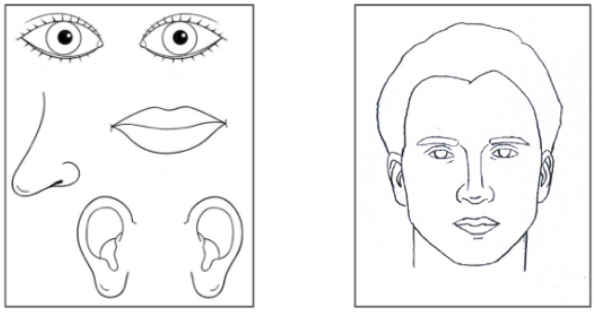

Figure 1: A CNN might mistake the left figure for a face as it identifies two eyes, two ears, a nose and a mouse but ignores their poses.

To mend the flaws of the CNN, Hinton et al. brought forward the concept of capsules [2]. In a CNN, a neuron is a scalar that represents the activation of a specific feature. By contrast, a capsule is a group of neurons, i.e. a vector, whose length represents the probability that a specific type of entity exists and orientation represents the instantiation parameter of the entity. This allows capsules at one level to make predictions, via transformation matrices, for the instantiation parameters of higher-level capsules.

To us, the capsule model is more interesting than the routing algorithm, in that a capsule provides a *equivariant* representation of an entity. Some naturally raised questions are:

1. What is the equivariance of capsule representations? How do we verify it and visualize it?
2. How stable is a network of capsules performing against noise?
3. What is the expressivity of capsules?

Inspired by [8], we answer these questions by conducting several carefully designed experiments. The general principles for the experiments are as follows.

- Fix the learning algorithm (dynamic routing) and the model architecture (CapsNet).
- Modify the dataset by applying transformations, adding noises, relabelling and so on, and see the results.

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

- Compare the results with a CNN when necessary.

The following part of the report is organized as follows. §2 provides a brief introduction to capsules, the CapsNet architecture and the routing algorithm used in the experiments. §3, §4 and §5 study the equivariance, stability and expressivity of the CapsNet via several experiments, and provide insights into the results. §6 discusses the capsule model and our work.

2 Backgrounds

Capsule networks represent a recent breakthrough in neural network architectures. They achieve state of the art accuracy on the MNIST dataset, a feat achieved traditionally by deep convolutional neural network architectures. Capsule networks introduce an alternative to translational invariance other than pooling through the use of modules, or capsules. Two key features distinguish them from CNNs': layer-based squashing and dynamic routing. Whereas CNN's have their individual neuron's squashed through nonlinearities, capsule networks have their output squashed as an entire vector. Capsules replace the scalar-output feature detectors of CNN's with vector-output capsules and max-pooling with routing-by-agreement.

2.1 Architecture

In the paper, we use the same architecture as [5], which is a shallow neuron network with only two convolutional layers and one fully connected layer. Conv1 has $256, 9 \times 9$ convolution kernels with a stride of 1 and ReLU activation. This layer converts pixel intensities to the activities of local feature detectors that are then used as inputs to the primary capsules. The second layer (PrimaryCapsules) is a convolutional capsule layer with 32 channels of convolutional 8D capsules (i.e. each primary capsule contains 8 convolutional units with a 9×9 kernel and a stride of 2). The final Layer (DigitCaps) has one 16D capsule per digit class and each of these capsules receives input from all the capsules in the layer below 2.

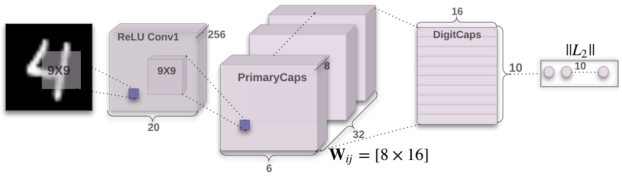

Figure 2: Architecture of CapsNet.

2.2 Algorithm

Capsules output a vector, which means that it is possible to selectively choose which parent in the layer above the capsule is sent to. For each potential parent, the capsule network can increase or decrease the connection strength. This routing by agreement is much more effective at adding invariance than the primitive routing introduced by max-pooling.

2.3 Regularization

Whereas traditional CNN's prevent overfitting by using dropout, Capsule networks are regularized with a reconstruction autoencoder. During training, all activity vectors are masked except for the activity vector corresponding to the correct digit. This activity vector is then used to reconstruct the input image. The output of the digit is then used to compute the loss. This encourages the network to learn a more general representation of images.

2.4 Dataset

All experiment in this paper is performed on 28×28 MNist [3] images that have been shifted by up to 2 pixels in each direction with zero padding. The dataset has 60K and 10K images for training and testing respectively.

3 Equivariance

To put it simply, a function f is *invariant* with respect to a transformation T if $f(T(x)) = f(x)$, and f is *equivariant* with respect to T if $f(T(x)) = T(f(x))$. Capsules are inherently a equivariant representation, and in [5], different dimensions of a capsule are observed to represent different properties of an entity, such as scale, thickness, skew and width. To illustrate the equivariance to the fullest, our key idea is:

> Force the capsules to represent rotation more than other instantiation parameters, by rotating the training images randomly.

Arguably, rotation is the most visible property of a entity and one of the easiest operations to an image. The capsule representation of an image in [5] does not take rotation very importantly, because in the MNIST dataset, digits are almost posed straight up. We rotate the images with random degrees, thus force the capsule representation to take rotation as the most significant instantiation parameter.

3.1 Inner product

Since the capsule representations are nonlinear (after squashing), we cannot expect simple linear relations of these representations, like

$$\text{capsule}(rot(B, d)) \approx \text{capsule}(rot(A, d)) - \text{capsule}(A) + \text{capsule}(B).$$

Here A, B are two images,

Instead, we can turn to cosine similarity, or equivalently, inner product of vectors after normalization. An important observation is that,

$$(\text{capsule}(rot(B, d)), \text{capsule}(A)) \approx (\text{capsule}(rot(A, d)), \text{capsule}(B)).$$

This suggests that rotation indeed dominates the embedding space. Another attempt is to compare

$$(\text{capsule}(rot(A, d)), \text{capsule}(A))$$

when degree d varies. We find that when $d < 90$, the inner product decrease as d increases.

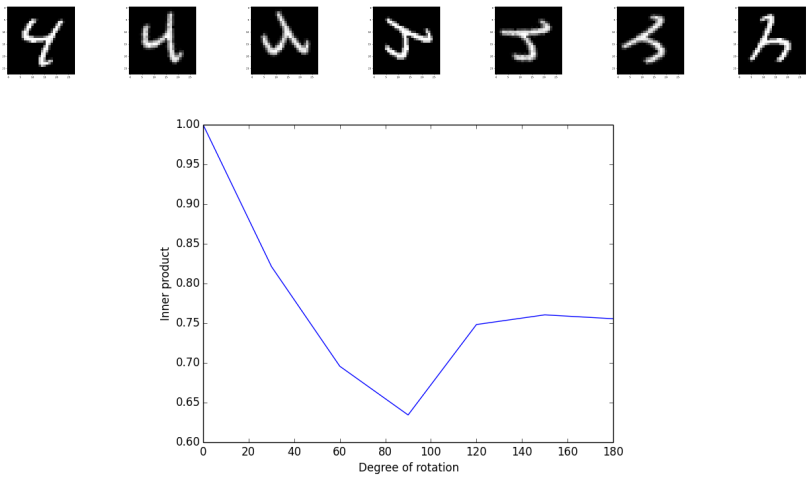

Figure 3: Image A with rotation 30 degrees iteratively. The inner product decreases as d increases when $d < 90$.

3.2 PCA

To push the analysis further, we utilize the principle component analysis (PCA) technique to figure out the exact "rotation" subspace.

After PCA, two dimensions are found far more significant than other dimensions. We extract these two dimensions and plot them in the 2D plane. The plot shows clearly that rotating input digits from 0° to 360° corresponding a path of circle in corresponding $2d$ subspace of capsule representation. For a digit like 7, the path will be a single circle. For a digit like 0, the path will be two circles, since 0 rotated by 180° is already the same. The result is illustrated in Figure 4, where for the left one we rotate a single digit 0 from 0° to 360°,3° degree each time. The orbit is two consecutive oval. For the right one, we do PCA analysis for two different digit 7 together, the orbit of them is two similarly oval.

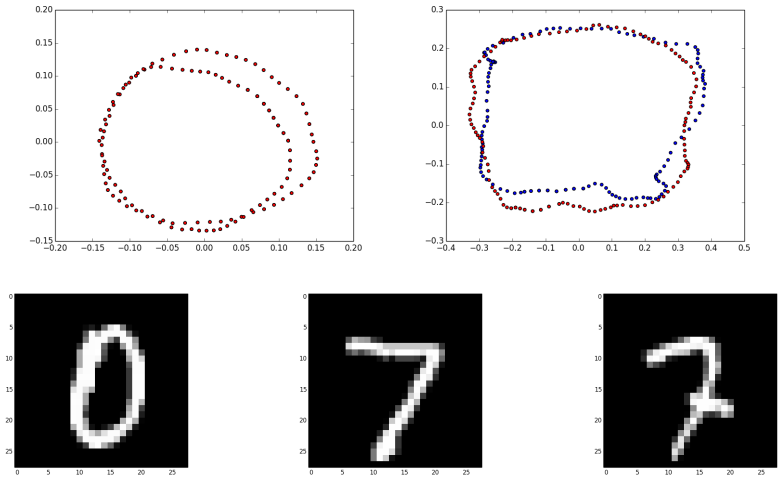

Figure 4: Left figure is the PCA results for 0. Right figure is PCA results for 2 different digit 7.

4 Stability Against Noise

Although deep neural networks (DNNs) achieve state-of-the-art performance in a wide range of tasks, they are vulnerable to small perturbations that cause unexpected classification errors [4].The generalization properties of these learning models were questioned by Szegedy et al[6], when the existence of adversarial examples was revealed. In this section, we test the robustness of capsNet. Instead of designing adversarial example, we explore the stability of capsules under certain kind of noise, i.e. we add noise to testing data of Mnists and evaluate the performance of Capsule. The noise we consider here including the following ones:

- Random Gaussian Perturbation: We add independent random gaussian noise $N(0, \sigma)$ to each pixels.
- Rotation: We randomly choose θ from $[0, 2\pi]$ and rotate θ to the overall picture.

In the following subsection, we evaluate the performance of capsules under various noises of training data, notice that in this part, training sample is still noiseless.

4.1 Accuracy

We first measure the accuracy of Capsules with random Gaussian perturbation and rotation, the results are showed in picture 5. We observe that Capsule is robust against certain amount of noise. For rotation, capsule still achieves formidable accuracy with small angle rotation. When the angle increases to 45 degree, the capsule somehow fails.

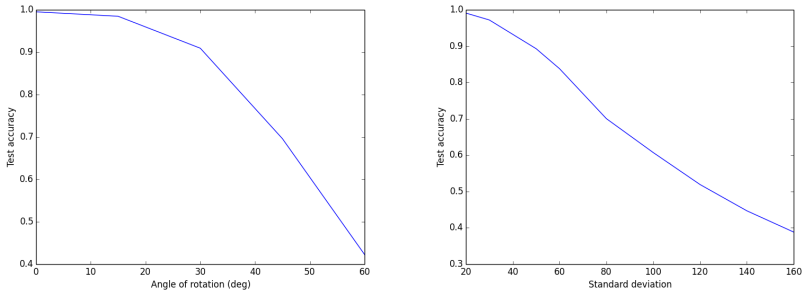

Figure 5: The left figure plots accuracy vs standard deviation of rotation, and the right one plots accuracy vs standard deviation of noise.

4.2 Reconstruction Against Noise

With the help of reconstruction module, we can visualize the effect of noise to capsule, sample pictures are shown in 6, we observe that capsules can even recover the original data, given noise data. First line of the figures show the input with rotation of degree $0°, 15°, 45°$, the reconstruction of input figure follows immediately. The second line of figures show the input with pixel-wise gaussian noise with standard deviation $20, 50, 100$. Notice when deviation is greater than 50 the reconstruction images becomes impossible to recognize.

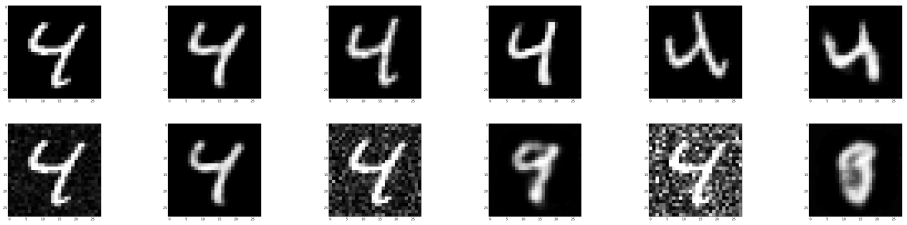

Figure 6: The upper figures are inputs and reconstructions with rotation $(0, 15°, 45°)$, the lower figures are with gaussian noise of $\sigma = (20, 50, 100)$

5 Expressivity

The number of network parameters of the CapsNet is way more than that of a CNN of the same depth. Intuitively, the expressivity of the CapsNet should be way better than the CNN, but it turns out not totally true. We employ the randomization technique in [8] to investigate how our CapsNet learns "random" data. More specifically, we conduct the following two experiments.

1. Replace the training images by completely random pixels.
2. Replace the labels by random labels but leave the images unchanged.

The first experiment is essentially a training-with-noise experiment pushed to a limit, where all information is lost. The second experiment leaves all the structure in images unchanged. It is observed in [8] that a CNN can easily fit random pixels or random labels, and our finding is:

> A capsule network can fit random labels easily, but not random pixels.

For the two experiments, we use a CNN of the same depth for comparison.

5.1 Random Label Experiment

We feed the CNN and the CapsNet to 1000 training images with random labels and train for 50 epoches. Both are able to reach an high accuracy easily, but the CapsNet has an accuracy 96% while

the CNN has an accuracy 86%. Our interpretation is that, the CapsNet can learn the features more accurately, regardless of labels. The expressivity

Maybe more interesting is the result of reconstruction. Here we use the true label of the image as the mask, and reconstruct images using the model trained with reconstruction loss. The main finding is that

- The CapsNet tends to reconstruct digits as a "8", which is a rough superposition of all digits. This has a nice explanation, since randomly assigning labels is equivalent to "mixing" all digits together.
- The training images (with wrong labels) are reconstructed more faithfully than testing images, as can be seen in Figure 7 and 8. This is of course due to the overfitting.

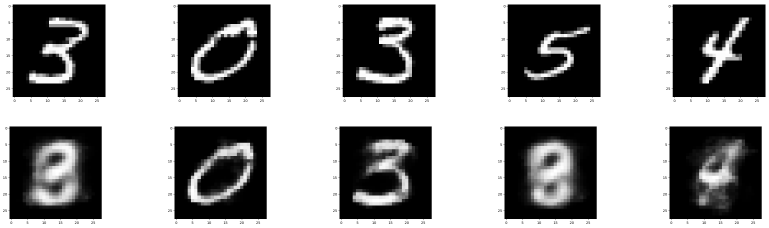

Figure 7: The reconstruction of training images. The first row is the original images and the second row is the corresponding reconstructions.

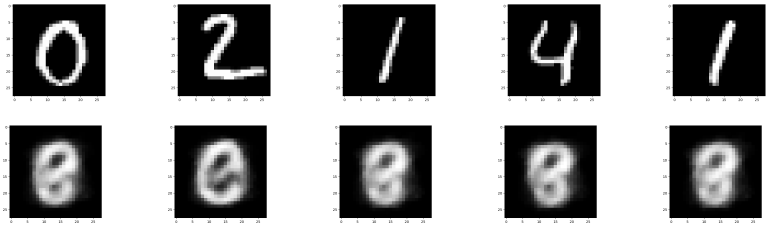

Figure 8: The reconstruction of testing images. The first row is the original images and the second row is the corresponding reconstructions.

5.2 Random Pixel Experiment

On the other hand, learning random pixels for a CapsNet seems harder. Within 50 epochs of learning 1000 images of random pixels, the CNN can reach an accuracy of more than 90% while the CapsNet can only reach an accuracy of about 30%. Does it suggest that the expressivity of CNN outdoes CapsNet, despite having far fewer parameters? Not really, since in 100 epoches the CapsNet can easily overfit, with accuracy more than 98%. So the correct comment is that both CNN and CapsNet have expressivity far beyond 1000 random samples, but CNN overfit faster.

From another perspective, this can be seen a merit of capsules. Capsules preserve local information of an entity precisely, and in order to activate higher-level capsules, certain part-whole pose relationship must be respected. This makes CapsNet hard to learn features out of total randomness, and explains the slower speed to overfit. CapsNet is already reported to be more robust than CNN against adversarial attacks, so the structure of capsules may serve for security purposes.

6 Discussion

In this report, we design various experiment on capsules. Firstly, we verify that capsules really capture important features of input and are indeed equivariant. With this in mind, we move further to test robustness and expressivity of capsule networks. Capsules are robust against certain kinds

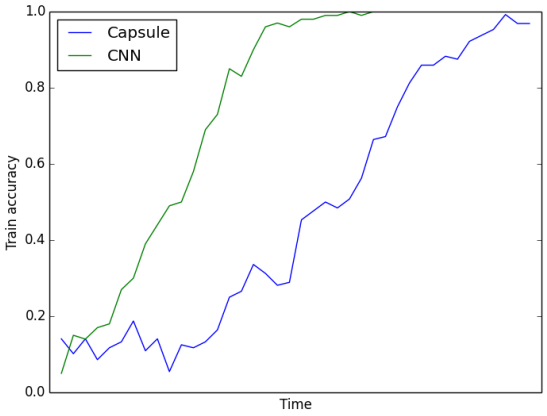

Figure 9: Training accuracy of CapsNet and CNN over time, with 1000 images of random pixels. The end time of the figure is 50 epohes.

of noise (like small pixel-wise noise and permutation) even if it is trained on noiseless data. Last but not the least, we follow ideas in [8] to train CapsNet on completely random data samples. The implication is that, CapsNet has great expressivity, but it is slower and harder to overfit. This has something to do with the nature of capsules, as discuessed in §5.

We focus on the model of capsules rather than the routing algorithm for a reason. The algorithm is simply not satisfying enough, as the learning requires iterations of routing and still back propagation.

Finally, we also propose our idea to other direction other than this report, i.e. design better algorithm or ingenius architecture to achieve better performance on more real life data. Here we also propose our understanding why training capsules on much more complicated dataset is difficult. One obvious reason is that dynamic routing or EM routing is time consuming, which makes training process much longer. The other possible reason is that, reveled by this report, capsules are forced to learn input image with strict part-whole structural properties, while most real life pictures contains non-trivial background, which makes capsule hard to distinguish what it really needs to capture. In order to achieve better performance, getting rid of background 'noise' perhaps are the main difficulty to overcome. One of the experiment results not present in main section is that if we convert pixel value of test data to its opposite(i.e black to white, white to black), the accuracy decreases dramatically. We do not think making neuron network deeper or other tricky techniques in computer vision area could overcome without resolving background problem.

References

[1] HE, K., ZHANG, X., REN, S., AND SUN, J. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (2016), pp. 770–778.

[2] HINTON, G. E., KRIZHEVSKY, A., AND WANG, S. D. Transforming auto-encoders. In Artificial Neural Networks and Machine Learning – ICANN 2011 (Berlin, Heidelberg, 2011), T. Honkela, W. Duch, M. Girolami, and S. Kaski, Eds., Springer Berlin Heidelberg, pp. 44–51.

[3] LECUN, Y., AND CORTES, C. The mnist database of handwritten digits.

[4] ROZSA, A., GUNTHER, M., AND BOULT, T. E. Towards robust deep neural networks with bang. arXiv preprint arXiv:1612.00138 (2016).

[5] SABOUR, S., FROSST, N., AND HINTON, G. E. Dynamic routing between capsules. In Advances in Neural Information Processing Systems 30, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 3859–3869.

[6] SZEGEDY, C., ZAREMBA, W., SUTSKEVER, I., BRUNA, J., ERHAN, D., GOODFELLOW, I., AND FERGUS, R. Intriguing properties of neural networks. Computer Science (2013).

[7] XI, E., BING, S., AND JIN, Y. Capsule network performance on complex data, 2017.

[8] ZHANG, C., BENGIO, S., HARDT, M., RECHT, B., AND VINYALS, O. Understanding deep learning requires rethinking generalization. CoRR abs/1611.03530 (2016).