CapsComplete: Completion of Occluded Images Using Capsule Networks

Vinay Ramesh
Columbia Univeristy
New York, NY
vrr2112@columbia.edu

Connor Goggins
Columbia University
New York, NY
cgg2131@columbia.edu

Abstract

Capsules are clusters of neurons in a neural network. Each capsule exhibits an activity vector that quantifies both the probability that a given entity exists and the specific orientiation of that entity. The nature of the architecture of capsules leaves them well-suited to modeling objects in space in a similar manner to the way the human mind understands the physical world. With these characteristics in mind, we seek to demonstrate the capacity of capsules to effectively model occluded entities by testing and building on their capabilities for classification and reconstruction.

1. Introduction

In 2017, Geoffrey Hinton (Google Brain, University of Toronto) published a paper that has the potential to change the way computers perceive and understand images at a fundamental level. This paper (*Dynamic Routing Between Capsules*, [8]) proposed an entirely new network architecture that leveraged structures that Hinton et al. dubbed "capsules" - groups of neurons whose activity vector represents the instantiation parameters of a specific type of entity within an image. These entities naturally manifest themselves as the component parts of objects identified in the images presented to the network. In this way, Hinton's "CapsNet" more closely mimics the way humans perceive and understand objects in space than standard convolutional neural networks. By effectively modeling entity transformations like skew, yaw, and translation, capsules have an advantage in understanding viewpoint variance and identifying partially occluded objects.

With that said, the development of capsules is so recent that many of their potential applications have yet to be investigated. Consequently, we hope to test the performance of capsule networks on classifying and reconstructing occluded images. In doing so, we build on Hinton's work by proposing and testing two novel techniques for improving the quality of the reconstructed images generated by capsules.

2. Related Work

Hinton first introduced the idea of capsules and why max pooling was ill-posed in his paper *Transforming Autoencoders* [4]. The idea is that when using max pooling, the routing that takes place creates local translational invariance whereas what should be strived for is equivariance - the idea that small perturbations result in corresponding perturbations in activations. Instead of translational invariance, neural networks in computer vision should approach viewpoint invariance through a part-whole relationship.

Sabour and Hinton's landmark 2017 paper provides an implementation of CapsNet, the first neural network using the methods outlined above [8]. Their paper provides the foundation of our work in this paper using coincidence filtering to route features from lower-level layers to higher-level layers. Routing is not necessarily novel. In fact, pooling is essentially the selective routing of only certain neurons, but is often 1 or 0 (max pooling) or evenly distributed (average pooling).

In addition, we also draw inspiration from several improvements on the original CapsNet that were published in 2018: one [3] by Hinton himself, and the other [7] by Phaye et al. from the Indian Institute of Technology. We draw from the original CapsNet paper in order to implement our model, and discuss the implementation of these capsules at length in the next section.

The improvement published by Hinton in 2018 describes a new routing procedure between capsules: EM (expectation maximization) routing. In this procedure, Hinton reworks the architecture of each capsule to include a 4x4 trainable pose matrix that represents the spatial relationship between the viewer and the observed entity. Hinton then outlines a process by which the pose matricies are multiplied by trainable viewpoint-invariant matricies such that their multiplicative product is intended to approximate a part-to-whole entity relationship. The aforementioned product for each capsule is weighted by an assignment coefficient, which is iteratively updated via an expectation maximization algorithm to the point that the products are routed to a higher-order capsule that receives similar products from

other lower-level capsules. This is in contrast to the squashing method proposed in [8] to determine routing agreement.

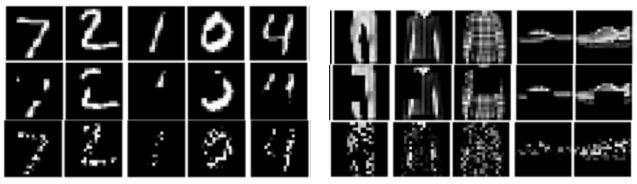
The improvement detailed by Phaye et al. modifies CapsNet in two key ways: first, by adding multi-level capsules as opposed to the single-level capsule layer outlined in [8]; second, by replacing CapsNet's convolutional layers with densely connected convolutional layers. Ultimately, this leads to the network learning to represent entity features in a hierarchical manner.

Other methods studied in class have sought to model part-whole relationships and viewpoint invariance using traditional convolutional networks. Spatial Transformer Networks seek to learn affine transformations in order to orient features or data correctly [6]. Deformable Convolution Networks learn offsets in kernel and routing protocols in order to attain better feature representations and can intuitively be understood to try and 'connect' related but spatially separated features [1].

3. Approach

3.1. Data

We test our proposed strategies for gradient step reconstruction and latent reconstruction on two datasets: MNIST and Fashion-MNIST. We subject each dataset to various degrees of two different types of occlusion: *deterministic occlusion*, in which a box of size axb pixels is placed randomly on the source image and entirely occluded, and *probabilistic occlusion*, in which each pixel has $p\%$ chance of being entirely occluded (where occlusion indicates that the pixels in question are colored entirely black). For examples of the original images on which we train and test our capsule network, refer to the first rows of Figures 1a and 1b; for examples of the original images subjected to *deterministic occlusion*, refer to the second rows of Figures 1a and 1b; for examples of the original images subjected to *probibalistic occlusion*, refer to the third rows of Figures 1a and 1b.



(a) MNIST Data (b) Fashion-MNIST Data

Figure 1:
The First Row is the original images.
The Second Row results from occluding 12x12 blocks
The 3rd row results from occluding with $p = 0.5$

3.2. Capsules

This section focuses on our implementation of capsules based on the work done in CapsLayer [5]. Due to time complexity and functionality considerations we used a CapsNet implementation that used squashing as the nonlinearity instead of expectation maximization.

3.3. Capsule Structure and CapsNet

Capsule Structure: A capsule is composed of a d-dimensional vector denoting the 'pose' of the capsule. The length of this vector is the activation (or probability of presence) of the entity modeled by this capsule. Figure 2 shows a simplified explanation of how poses and activations work [5].

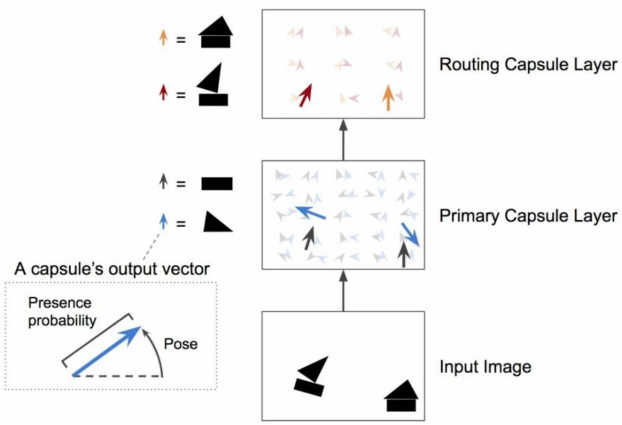


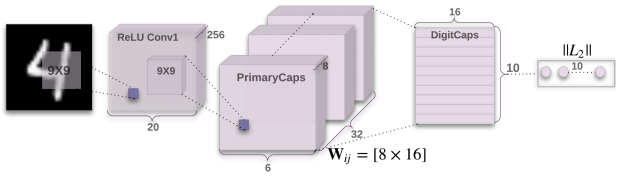
Figure 2: Demonstration of Pose Vectors and routing

Routing Algorithm and Learning: The only learning that occurs in capsule networks (like in neural networks) are the weights of the viewpoint invariant matrices W. Poses, activations, and coefficients from lower- to higher-order capsules are unique to each sample. The nonlinearity introduced is through the squashing function, which places vectors of large length towards 1 and vectors of small length towards 0. In the steps below (outlined in Figure 3), j represents the higher-level capusle and i is the lower-level capsule. u_i is the pose vector of the lower-order capsule. c_{ij} is the routing coefficient from Capsule i to Capsule j. v_j represents the projected higher-order capsule pose. The steps in Figure 3 are iterated three times over for each capsule pair to attain the necessary routing. The b_{ij} (essentially logits for i to be routed to j) are initialized to 0. We did not experiment with different priors for b_{ij}.

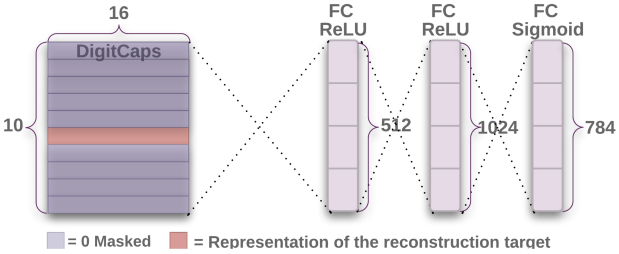
CapsNet. Figures 4a and 4b outline the architecture for CapsNet. Of particular note is that only the routing between the Primary Capsules and the Class Capsules (DigitCaps) uses the Dynamic Routing algorithm. The CapsNet decoder masks the input of all other digits for regeneration.

1. Affine Transformation: $\hat{u}_{j|i} = W_{ij} u_i$
2. Dynamic Routing Calc: $c_{ij} = \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})}$
3. Routing: $s_j = \sum_i c_{ij} \hat{u}_{j|i}$
4. Squashing: $v_j = \frac{\|s_j\|^2}{1+\|s_j\|^2} \frac{s_j}{\|s_j\|}$
5. Dynamic Routing Update: $b_{ij} = b_{ij} + \hat{u}_{j|i} \cdot v_j$

Figure 3: Dynamic Routing Algorithm with Squashing



(a) CapsNet Encoder

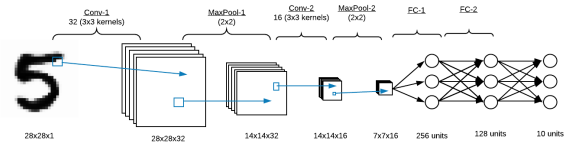


(b) CapsNet Decoder

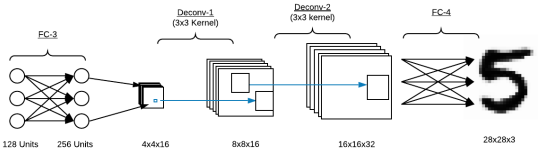
Figure 4: CapsNet Architecture

3.4. Baseline Autoencoder

We constructed a convolutional autoencoder to attain baseline results. The architecture used is shown in Figures 5a and 5b.



(a) Autoencoder Encoder



(b) Autoencoder Decoder

Figure 5: AutoEncoder Architecture

3.5. Loss Function

For both the autoencoder and CapsNet, we use the following Loss Function where $\mathcal{L}_c$ is classification loss. We use cross entropy loss for the autoencoder and separate hinge loss for CapsNet (as Hinton et al. use in the paper [8]). $\mathcal{L}_r$ is the ℓ_2 reconstruction loss.

$$\mathcal{L} = \mathcal{L}_c + \mathcal{L}_r$$

3.6. Gradient-Based Reconstruction Techniques

Our improvement on the reconstructions generated by Hinton's original CapsNet architecture can be decomposed into two primary components: a Gradient Step Mechanism to improve occluded image-based reconstructions, and a method for Latent Class-Based Reconstruction.

First, we leverage a Gradient Step Mechanism to reconstruct the original unaltered image when our network is provided with a specific occluded image. The gradient can be described in the following manner, where we substitute the hinge loss used in the training step by cross-entropy in order to maximize the length of the activation vector:

$$\nabla_x \mathcal{L}(x) = \nabla_x - \log(\Pr(\hat{y}))$$

$$x_{\text{new}} = x - \lambda \nabla_x \mathcal{L}(x), \mathcal{F}_{\text{occ}}(x) = \mathcal{F}(x_{\text{new}})$$

Second, we propose an entirely original system for Latent Class-Based Reconstruction on a trained capsule network. To achieve this, we generate an optimal input image x and reconstruction image $\tilde{x} = \mathcal{F}(x)$ for a given class c (with respect to MNIST, digits 0-9). Input x is optimized such that the probability of the given class c is maximized:

$$\max_{x, \mathcal{F}(x)} \Pr_c(x)$$

Then, the reconstruction of that optimal input image x ($\tilde{x} = \mathcal{F}(x)$) is generated in turn as the latent reconstruction of class c.

4. Experiments

With the approach described above in mind, we test the following metrics:

First, we evaluate the classification accuracy of CapsNet benchmarked against the classification accuracy of the convolutional autoencoder with the training and testing datasets. The datasets in question were subjected to various levels and types of occlusion during both training and testing. Furthermore, the quantity of training data was varied to test the classification accuracy with a more restricted data sample size.

Second, we evaluate the quality of the reconstructions (generated using our Gradient Step Mechanism) associated with individual images from the datasets. Once again, the

datasets in question were subjected to various levels and types of occlusion during both training and testing, and the quantity of training data was varied to test the reconstruction quality with a more restricted data sample size.

Third, we evaluate the quality of the Latent Class-Based Reconstructions after training of both CapsNet and the convolutional autoencoder has completed. Once again, the datasets in question were subjected to various levels and types of occlusion during both training and testing, and the quantity of training data was varied to test the reconstruction quality with a more restricted data sample size. This was computed by optimizing the input image over class probability for each class through the gradient method until convergence.

All tests used our CapsComplete Library (linked here). Training runs on a sample of 10,000 images using a single Nvidia Tesla K80 GPU took approximately 15 minutes to complete.

5. Results

5.1. Classification Accuracy with Various Levels of Occlusion and Quantities of Training Data

We test classification accuracy for CapsNet and the Convolutional Autoencoder with various numbers of training samples. Regarding the results logged in the tables below, in every case the models were trained for 50 epochs with a batch size of 32. For Table 1, the results were obtained using the MNIST dataset subjected to 70% probabilistic occlusion during testing. For Table 2, the results were obtained using the Fashion MNIST dataset subjected to 50% probabilistic occlusion during testing. Bold entries indicate superior performance of CapsNet relative to the baseline. We subjected Fashion MNIST to a slightly reduced degree of occlusion (50% vs 70% for MNIST) due to the increased complexity and variability of each class in the dataset.

# Samples	CapsNet	CapsNet Occ	Autoencoder	Autoencoder Occ
100	**71.01%**	**59.39%**	10.15%	17.61%
500	**87.54%**	**71.33%**	83.04%	71.09%
1000	**93.51%**	**77.65%**	90.64%	74.65%
5000	96.62%	**79.10%**	96.94%	75.65%
10000	97.22%	**85.01%**	97.56%	77.66%

Table 1: Classification test accuracies on MNIST using 70% probabilistic occlusion.

# Samples	CapsNet	CapsNet Occ	Autoencoder	Autoencoder Occ
100	**66.59%**	**52.81%**	20.02%	10.25%
500	**77.71%**	**60.09%**	64.55%	33.02%
1000	**79.74%**	**54.31%**	72.64%	43.58%
5000	**83.78%**	**59.62%**	79.49%	52.87%
10000	83.95%	**63.94%**	86.72%	54.95%

Table 2: Classification test accuracies on Fashion MNIST using 50% probabilistic occlusion.

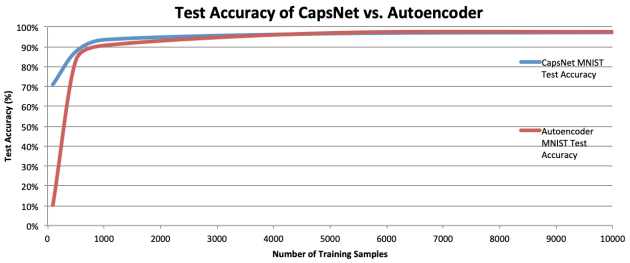
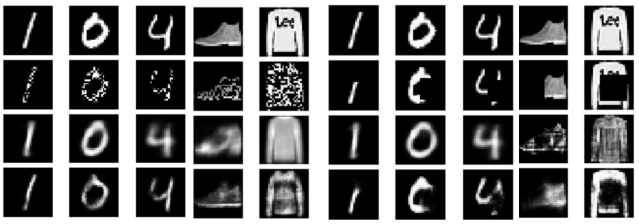


Figure 6: CapsNet performs 7X better than the autoencoder on MNIST with a training sample of 100 images.

5.2. Reconstruction with Both Types of Occlusion

We showcase the results of using our Gradient Step Mechanism for reconstruction in Figures 7a and 7b. The first row in both figures contains sample images from the original dataset. The second row contains the same images subjected to occlusion. The third row contains the CapsNet reconstructions given the occluded image, and the fourth row contains the AutoEncoder reconstructions given the occluded image. These reconstructions were all generated after training both models for 50 epochs on a sample of 10,000 non-occluded images with a batch size of 32.



(a) Probabalistic Occlusion Reconstructions ($p = 0.5$)

(b) Deterministic Occlusion Reconstructions (axb = 14x14)

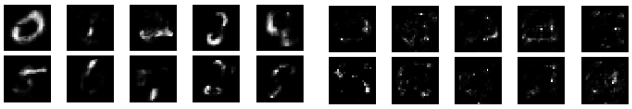
Figure 7: Occluded Images and Corresponding Reconstructions. In row order: input, occluded, CapsNet Reconstruction, Autoencoder Reconstruction

5.3. Latent Reconstruction and Classification Accuracy with Occluded Training Dataset

We also evaluated the Latent Class-Based Reconstruction quality and classification accuracy of CapsNet and the Convolutional Autoencoder after training both models on an occluded dataset (in this case, MNIST subjected to 14x14 deterministic occlusion). The classification accuracy results are logged in Table 3. Once again, bold entries indicate superior performance of CapsNet relative to the baseline. The latent reconstructions for CapsNet are outlined in Figure 8a, and the latent reconstructions for the autoencoder are outlined in Figure 8b.

# Samples	CapsNet	CapsNet Occ	Autoencoder	Autoencoder Occ
10000	**91.52%**	**84.53%**	85.93%	79.01%

Table 3: Classification Test Accuracies after training on deterministically occluded MNIST.



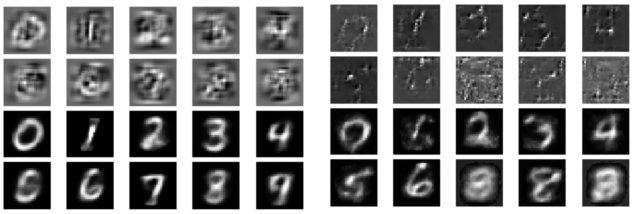
(a) CapsNet Latent Reconstructions (Occluded MNIST)

(b) Autoencoder Latent Reconstructions (Occluded MNIST)

Figure 8: Latent Reconstructions after training on occluded data for 50 epochs with 10K samples and batch size = 32.

5.4. Latent Class-Based Reconstruction

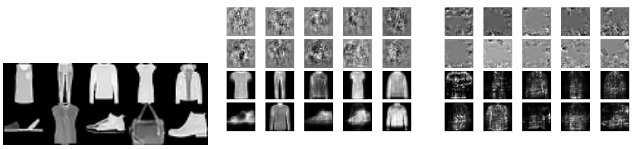
The first two rows in Figures 9a and 9b show the inputs that maximize each class followed by the corresponding reconstructions. These reconstructions were generated after both models were trained for 50 epochs with 10K non-occluded samples from the relevant dataset and batch size = 32.



(a) CapsNet Latent Reconstructions (MNIST)

(b) Autoencoder Latent Reconstructions (MNIST)

Figure 9: MNIST Latent Reconstructions



(a) Original Fashion MNIST Classes

(b) CapsNet Latent Reconstructions (Fashion MNIST)

(c) Autoencoder Latent Reconstructions (Fashion MNIST)

Figure 10: Fashion MNIST Latent Reconstructions

6. Discussion

6.1. Conclusions

Based on our approach, we have shown that capsules maintain a higher degree of class fidelity than generic convolutional networks. In doing so, we used our gradient-based reconstruction techniques to demonstrate the viability of capsule-based neural networks in completing images to better fit with the notion of a class. We have further justified that the gradient based approach creates noisy but empirically better approximations of image completion than the autoencoder, which is unable to complete images to a sufficent degree of accuracy.

Furthermore, the classification accuracy of CapsNet was evidently superior to that of the autoencoder - especially in conditions where the number of samples was significantly limited or when the testing data was occluded. Classification accuracies at higher sample numbers without occlusion in the testing data were comparable between CapsNet and the autoencoder, but in all other cases CapsNet clearly served as a more effective classifier.

6.2. Error Analysis

We believe that some of the minor distortion in the reconstructions generated by capsules could be in part a result of down-weighting reconstructions in favor of classification, which is a natural tradeoff. Further, we believe the squashing activation loses information of the pose vector, which prevents inputs that highly resemble the 'generic' class and inputs that have subtle differences from being accentuated since their probabilities are squashed to the same value. This idea is in line with the concerns raised prior to the creation of the alternative EM Routing for Capsules [3].

Saturation for capsule networks with respect to classification accuracy under larger sample quantities is also a consideration to keep in mind. While we identified that in low-sample ranges capsules are extremely strong in extracting maximal information from minimal datapoints, we believe that larger pose vectors may be able to account for more information as more datapoints are given.

6.3. Challenges

We faced a number of challenges implementing and experimenting with capsule networks in order to accomplish our particular task. Since CapsNet is so new and the authors have not published their source code, there are few implementations to pull from that have the flexibility required to support modification and adaptation of capsule network architectures. Nonetheless, we were able to find a repository called CapsLayer which implemented both Dynamic Routing and EM Routing [5]. Unfortunately, this library was not bug free and we spent significant time trying to fix it. Furthermore, in experimenting with networks that used EM

routing, we were only able to attain 70% classification accuracy on MNIST which is very sub-par. The time complexity of EM-Routing also proved extremely prohibitive during training, since the routing cannot be optimized via GPU.

This computational optimization problem of capsule networks on GPUs is not unique to EM Routing, as Dynamic Routing through squashing faces the same problem with fewer operations per routing iteration. While we achieved state-of-the-art results in the range of training sample sizes required for high performance, in each case CapsNet took approximately 5X as long as the autoencoder to train for the same number of epochs.

Additionally, we faced issues with our gradient method in finding convergence. It was difficult to find a balance between doing a latent reconstruction and doing a more input-based reconstruction. If we performed too many gradient steps, the occluded reconstruction resembled the latent reconstruction more closely than it resembled the original image.

Lastly, we were not able to see any performance enhancements by 'stacking' Primary Capsules and adding additional convolution (without pooling) prior to the PrimaryCaps layer (as Phaye et al. did in *Multi-Level Dense Capsule Networks* [7]). The number of parameters also increased tremendously in an already high parameter space, making the training much slower than desired.

6.4. Future Work

Initially when we set out on this project, we intended to perform occluded image completion on CIFAR-10 data. Unfortunately, our tests at the time for the milestone showed that performance baselines were extremely low. Some work has been done to improve this, including Phaye et al.'s Multi-Level Dense Capsule Networks [7] and Ensemble CapsNets [8]. The logical next step would be to apply our gradient step method to those models and to create a deeper set of layers with the purpose of extracting more complex features for multiple object detection, which is not well-supported by CapsNet at the moment.

From a performance standpoint, something that would make capsules much more practical would be hardware innovations that allow for computational optimization of routing algorithms. Currently, the latency of routing means that capsule networks are not viable for production environments because in spite of the lack of depth, the number of parameters is large and the routing algorithm is slow.

We also want to investigate if there is a capsule equivalent of deconvolution which uses upsampling or strided transposed convolution in order to attain reconstructed images (as we used in our autoencoder). Lastly, as these models become more complex our gradient method would be justified in understanding the entity attributes that lower-level capsules recognize.

7. Conclusion

The initial results provided by Geoffrey Hinton in *Dynamic Routing Between Capsules* [8] showed that capsules are a better approximation for the way humans view and interpret the placement of objects in the physical world than standard convolutional neural networks. However, until now, the classification and reconstruction capabilities of capsules on images subjected to various degrees and types of occlusion have not been fully assessed or improved to their full potential. Our project serves this purpose, and - given our favorable results - we hope that our work will serve as the foundation of future research into the performance of capsule networks on occluded entities.

Our primary contribution to the state-of-the-art is twofold: first, we introduce a Gradient Model which we have shown to outperform the baseline from a reconstruction fidelity standpoint. In every case tested (on non-occluded images and probabilistically/deteministically occluded images from both Fashion MNIST and MNIST with various training sample sizes), the reconstructions generated by CapsNet were more accurate approximations of the non-occluded source images than those generated by the autoencoder (see Figures 7a and 7b). Second, we propose an entirely original method for generating latent reconstructions for each class in the model. Once again, the latent class-based reconstructions generated by CapsNet are consistently more representative of the original class than those generated by the baseline convolutional autoencoder. This is immediately evident upon examination of Figures 8a, 8b, 9a, 9b, 10b, and 10c. In developing an effective and completely new framework for generating reconstructions of classes (as opposed to individual images), we have provided a comprehensive means of evaluating the ability of any trained model to approximate each of its classes. We hope that our progress in this domain will be used by computer vision researchers in the future to test the fidelity of their models to the categories of images that they seek to classify.

In addition to these definitive improvements on the current state-of-the-art, we also prove that capsules are more capable of accurately classifying all types of images than conventional convolutional neural networks. When both networks (CapsNet and the convolutional autoencoder) were trained for the same number of epochs on the same set of image samples, CapsNet achieved comparable (in three cases) or better (in seven cases) performance for every sample size on non-occluded images from both datasets. This alone demonstrates that capsules are better suited to modeling physical entities than standard CNNs. But more importantly, CapsNet outperformed the baseline in every case and on every sample size for occluded images from both datasets, in certain cases achieving classification accuracies that were 5X better than the baseline. This proves that

the ability of capsules to model entities in a more human-intelligible manner always translates to more accurate identification of entities, especially when those entities are partially occluded. It is also important to note that the baseline used for our tests (the convolutional autoencoder) was actually quite sophisticated, making the far-superior performance of CapsNet even more remarkable.

We offer these results of our work in the nascent domain of capsules as proof that capsule networks are worthy of future research, as they could potentially serve as a catalyst for improving accuracy in real-world computer vision tasks in the same way that convolutional neural networks did in the early 2000s with their performance on GPUs. We provide our unique methodologies for generating reconstructions with the hope that they will be applied not just to capsules, but to all subdivisions of computer vision.

Acknowledgements

Thanks to Professor Carl Vondrick and TA Dave Epstein for their support and guidance through this project.

References

[1] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017.

[2] A. Géron. Introduction to capsule networks. `https://www.slideshare.net/aureliengeron/introduction-to-capsule-networks-capsnets`, 2017.

[3] G. Hinton, S. Sabour, and N. Frosst. Matrix capsules with EM routing. 2018.

[4] G. E. Hinton, A. Krizhevsky, and S. D. Wang. Transforming auto-encoders. In *International Conference on Artificial Neural Networks*, pages 44–51. Springer, 2011.

[5] J. H. Huadong Liao. Capslayer: An advanced library for capsule theory. `http://naturomics.com/CapsLayer`, 2017.

[6] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015.

[7] S. S. R. Phaye, A. Sikka, A. Dhall, and D. Bathula. Multi-level dense capsule networks. 2018.

[8] S. Sabour, N. Frosst, and G. Hinton. Dynamic routing between capsules. 2017.