Exploring Dynamic Routing As A Pooling Layer*

Lei Zhao
TakingData
Beijing, China
bhneo@126.com

Lei Huang
Inception Institute of Artificial Intelligence
Abu Dhabi, UAE
huanglei36060520@gmail.com

Abstract

Dynamic routing is a routing-by-agreement mechanism which is important for achieving the equivariance and invariance properties for capsule network (CapsNet). It is valuable to explore the nature of dynamic routing for better understanding of the capsule idea and further improving the performance of neural networks. This paper explores the dynamic routing from the pooling perspective. We modify the original dynamic routing algorithm for better applying it in traditional Convolutional Neural Networks (CNNs) as a pooling layer. We also use a parameter λ in softmax to smoothly adjust the sparsity in the routing, which leads to lower cost compared to the original dynamic routing. We experimentally show that the dynamic routing can be applied to beyond the capsule network to improve the performance of CNNs, and the coupling coefficients generated by the routing can be used to generate heatmaps which provide visual explanations to some extent. Further, the proposed dynamic routing method, combining a CNN backbone, achieves better results with much fewer parameters than the baselines on aff-NIST and multi-MNIST tasks.

1. Introduction

The pooling layers are extensively used in current CNN architectures to provide translational invariance and reduce parameters, which however leads to information loss such as position, size, rotation, scale [5]. Hinton *et al*. [5] address this by amplifying the neuron representation with vector-output capsules that are collections of neurons. A capsule represents an object, and the activity-vector of a capsule encodes the instantiation parameters of this object. When the viewing condition changes, the instantiation parameters change, but the capsule representing still stay active. Such a property is called equivariance and invariance [5, 17], and can be used to build visual relationships between capsules of different layers with a characteristic of assigning parts to wholes, while pooling layers model visual relationship by selecting the response regions of interest in the previous layer, with reduced representation.

Dynamic routing is introduced by Sabour *et al*. [17] to achieve the parts-to-wholes relationship for capsule network. It works by routing output-vectors from lower-level capsules to upper-level ones iteratively. This iterative algorithm selects the most appropriate parent capsule so that the active capsules in the network represent nodes in a parse tree, and connections between adjacent layers get sparser when using more routing iterations. The main problem of this dynamic routing algorithm is its expensive costs, both in terms of memory and computation [8]. The costs are further amplified with growing routing iterations for achieving sparser connections in a parse tree. These problems of dynamic routing limit the practice of CapsNet.

This paper explores the dynamic routing as a pooling layer, and expects to exploits the advantages of both the dynamic routing and pooling layer, based on our observation that the weighted sum process in dynamic routing can be viewed as a pooling operation. We modify the original routing algorithm to make it be flexibly inserted in common CNN architectures. We also use a parameter λ to adjust the sparse extent of connection smoothly, which turned out to have a similar effect as multiple iterations of routing with reduced computation costs. We further show how to apply our dynamic routing in traditional CNNs. The experimental results on several CNN models show that the proposed dynamic routing pooling achieves better performance than max/average pooling. And the coupling coefficients generated by the routing show the different importances of the input capsules in spatial, from which we can use these coefficients to generate heatmaps to provide visual explanations. We also show that the proposed dynamic routing, combined with the 20-layer residual network that acts as a primary capsule extractor, achieve better results with much fewer parameters than the baselines on aff-NIST and multi-MNIST of the paper [17].

*The first workshop on Statistical Deep Learning for Computer Vision, in *Seoul, Korea*, 2019.

2. Related Work

The concept of capsules was firstly introduced by Hinton *et al*. [5] to address the representational limitations. Sabour *et al*. [17] proposed CapsNet with dynamic routing algorithm which achieved a state-of-the-art result on MNIST dataset. Hinton *et al*. [16] proposed a new iterative routing procedure based on the EM algorithm which achieved an impressive result on the small-NORB dataset. After that, the idea of capsule was applied to many specific tasks to improve the traditional models [2] [8] [23]. And a lot of efforts have been made to seek better capsule architectures [1] [10] [14] [19], while some works focused on investigating the inherent properties of the routing-by-agreement mechanism [9] [18]. Our work aims at exploring applying the dynamic routing to improve the traditional networks by using it as a pooling operation.

3. Methods

According to the original CapsNet in the paper [17], prediction vectors $\hat{u}_{j|i}$ from child capsules are weighted summed to a parent capsule s_j as:

$$\boldsymbol{s}_j \leftarrow \sum_i c_{ij} \hat{\boldsymbol{u}}_{j|i}. \tag{1}$$

With larger c_{ij}, capsule i in layer l gets a stronger connection with capsule j in layer $l+1$. The coupling coefficients between capsule i and all its parent capsules are summed to 1, since the $softmax$ operation is along the dimension j:

$$c_{ij} = \frac{\exp b_{ij}}{\sum_k \exp b_{ik}}, \tag{2}$$

where b_{ij} are the log prior probabilities that capsule i should be coupled to capsule j [17].

We observe that Formula 1 can be viewed as a generalized pooling operation, where the pooling operation is performed over the capsules, rather than the spatial location. We thus explore the dynamic routing from the pooling perspective, and try to apply it as an alternative for max/average pooling.

3.1. Modifications on Softmax Operation

One essential step of pooling is how to calculate the coefficients c_{ij} in Formula 1. We first observe that when dynamic routing is embedded in several kinds of networks as a pooling layer, performing $softmax$ along dimension j as described in Formula 2 can cause the instability in training if no normalization operation such as **squash** performed after the pooling. We conjecture the potential reason is the large values in output tensor, which caused by large coefficients c_{ij}. We further find that Formula 2 is not appropriate for the situation where the multiple child capsules in layer l route to only one parent capsule in layer $l+1$. We thus perform the $softmax$ along dimension i instead of j, and the coupling coefficients c_{ij} are calculated as:

$$c_{ij} = \frac{\exp b_{ij}}{\sum_k \exp b_{kj}} \quad \sum_i c_{ij} = 1. \tag{3}$$

To simplify denotation, we omit the parent capsule index j (the j-th pooled output) in the subsequent sections.

3.2. Controlling the Extent of Sparsity

A sparser connection between capsule layers can ensure the emergence of a parse tree in the network, this parse tree represents the hierarchical composition of objects out of smaller and smaller components [13], which lead to better generalization and interpretability.

The original dynamic routing [17] updates coupling coefficients for certain parent capsule by accumulating the distance b_i for child capsule i in each iteration:

$$b_i \leftarrow b_i + \sigma \tag{4}$$

The sparsity of the coupling coefficient is usually controlled by the iteration number. More iterations of dynamic routing make b_i larger, and with the amplification of $softmax$ operation, lower values are suppressed almost to 0, while the larger one gets close to 1. One can achieve sparser-tree with larger iterations. However, larger iterations mean more computational costs. Inspired by the works[4] [11] [15] [21] [20], we introduce a parameter λ in this $softmax$ to adjust "soft" extent:

$$c_i = \frac{\exp \lambda b_i}{\sum_k \exp \lambda b_k}. \tag{5}$$

The $softmax$ is an operation that normalizes the input vector and suppresses the smaller elements in the vector exponentially, and this λ can be used to adjust the extent of suppressing. With a large enough λ, the differences between b_i [17] will be amplified, and "winners" in the competition will get almost 100% proportion, thus making the routing between capsules sparser. It would act like a max pooling operation but determined by vectors instead of scalars. When setting λ to 0, the routing would be equivalent to the average pooling. One advantage of using λ to control the extent of sparsity is that the good sparsity will be achieved with fewer iterations, thus with reduced computation costs. Besides, our method is the generalization of max/average pooling, and we can achieve better performance with the appropriate λ according to the experiments.

3.3. Vector-Normed Dynamic Routing

Our proposed dynamic routing is described in Algorithm 1. We first initialize the b_{ij} by zeros and get the initial c_{ij} by

Algorithm 1 Vector-Normed Dynamic Routing

Input: $\hat{\boldsymbol{u}}_i, \lambda, r$
Output: $\boldsymbol{v}$

$\hat{\boldsymbol{u}}_i^{'} \leftarrow norm(\hat{\boldsymbol{u}}_i)$
$\forall b_i, b_i \leftarrow 0$
$c_i \leftarrow softmax_j(b_i)$
for r iterations **do**
  $\boldsymbol{s} \leftarrow \sum_i c_i \hat{\boldsymbol{u}}_i$
  $\boldsymbol{v} \leftarrow norm(\boldsymbol{s})$
  $b_i \leftarrow b_i + \hat{\boldsymbol{u}}_i^{'} \cdot \boldsymbol{v}$
  $c_i \leftarrow softmax_i(\lambda b_i)$
end for
$\boldsymbol{s} \leftarrow \sum_i c_i \hat{\boldsymbol{u}}_i$
$\boldsymbol{v} \leftarrow activation(\boldsymbol{s})$ {activation can be any other function according to the network}
return $\boldsymbol{v}$

Formula 3. We then start the iterations for calculating the coupling coefficients c_i, but we use **vector normalization** instead of **squash** to compute the proposal capsule $\boldsymbol{v}$ whose length is 1. The vector normalization is computed as:

$$\hat{\boldsymbol{u}}_i^{'} = \frac{\boldsymbol{u}_i}{\|\boldsymbol{u}_i\|}, \quad \boldsymbol{v} = \frac{\boldsymbol{s}}{\|\boldsymbol{s}\|}. \tag{6}$$

The agreements between $\hat{\boldsymbol{u}}_i^{'}$ and $\boldsymbol{v}$ are computed by dot product: $\hat{\boldsymbol{u}}_i^{'} \cdot \boldsymbol{v}$, and we use λ in the $softmax$, as described in Formula 5. Another difference from [17] is that we separate the computation of c_i from the routing procedure, *i.e.*, we compute the c_i from routing iterations ahead, and then use it to compute the final capsule. This process makes the attempt of other activation operations possible other than only $squash$.

3.4. Dynamic Routing as a Pooling Layer

In traditional CNN architecture, there is usually a pooling layer following by the end of convolutional layers, which can be used to reduce parameters and provide invariance. We can replace this pooling layer with dynamic routing to provide better representation.

Here we propose a fairly straightforward implementation, which showed in Figure 1, this form can be applied to replace the layer such as the average pooling in resnet [3]. By choosing a good strategy of grouping neurons into capsules, we can achieve a better result than max and average pooling.

4. Experiments

4.1. Comparison with Max and Average Pooling

We build a simple network which has 4 convolutional layers followed by a pooling layer which produce output tensor with 64 neurons, then a fully-connected layer makes the class predictions through softmax outputs. Each convolutional layers include 64 filters with 5x5 kernel size and activated by rectified linear units. We conduct experiments on 3 popular datasets: Fashion-MNIST [22], cropped SVHN [12], CIFAR10 [7].

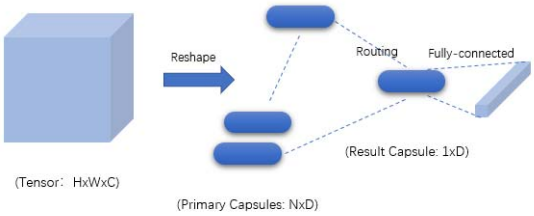

Figure 1. Applying dynamic routing as a pooling layer between Convolutional and Dense layer, the input tensor with shape $W \times H \times C$ can be grouped into N capsules by reshaping or any other ways. These capsules then routed into 1 or a few capsules according to the situation, before mapping into the result tensor by a fully-connect layer.

The model is trained with a batch size of 128 and optimized by Adam [6] with an initial learning rate of 0.001. We set the pooling layer as max/average pooling, and dynamic routing respectively to do the experiment. Data augmentation (random-crop) are performed during this experiment. Result in Table 1 shows that the test error get reduced when using dynamic routing layer, here we set λ as 0.5 and each capsule contain 16 neurons, which is searched on our constructed validation set (5000 examples from the training set).

# dataset	error(%)		
	max	average	routing
Fashion-MNIST	7.86	7.61	**7.52**
cropped SVHN	6.51	6.47	**6.02**
CIFAR10	21.39	19.35	**18.39**

Table 1. Comparison of average pooling, max pooling and dynamic routing.

4.2. Applying to Residual Networks

We apply our method on residual networks [3] by replacing the average pooling layer with dynamic routing, and train residual networks with depth 20, 32, 44, 56 on CIFAR10 by following the same configuration as in the paper [3]. Here we run all the experiment for 3 times and calculate the average error to make sure a reliable comparison. We set λ to 5, which is searched on our constructed validation set (5000 examples from the training set). From the results in Table 2, we can see our proposed dynamic routing pooling achieves better performance over the 4 models, compared to the baselines.

# layers	# params	error(%)	
		baseline	routing
20	0.27M	8.73	**8.49**
32	0.46M	7.89	**7.65**
44	0.66M	7.22	**7.10**
56	0.85M	6.97	**6.75**

Table 2. Results of our proposed dynamic routing on residual networks. All the models achieve improvement with a dynamic routing layer.

Table 3 shows the heatmaps generated from the coupling coefficients c, which indicates that the model learned to focus on specific areas and ignore backgrounds by coupling coefficients c.

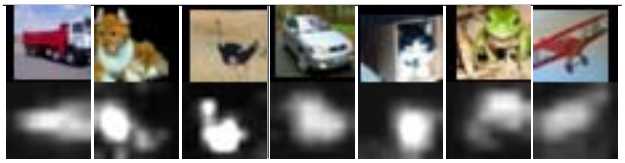

Table 3. The upper images are from CIFAR10 dataset (randomly cropped), and the lower images are their corresponding heatmaps generated by coupling coefficients c.

4.3. Aff-NIST and Multi-MNIST Dataset

We use resnet-20 as a backbone which acts as a primary capsule extractor, but we replace the last 2 layers(the average pooling and fully-connected layer) with dynamic routing, which route the tensor into 10 capsules, and **squash** is used as activation. These 10 capsules are followed by a reconstruction module same as the original capsule network architecture in [17], as described in Figure 2.

We replicate the experiment of Sabour *et al*. [17] on the aff-NIST[1] dataset. In which each example in the training set is an MNIST digit placed randomly on a black background of 40x40 pixels. The model never trained with affine transformations other than translation and any natural transformation seen in the standard MNIST, we compare the results on both test sets of expanded MNIST and aff-NIST to evaluate the model. The results on aff-NIST showed in Table 4 indicate that the model is robust on aff-NIST with the backbone of resnet, and model size is reduced largely (about 2.5M) compare to the original capsule networks (about 13.4M).

We also replicate the experiment of multi-MNIST, the dataset is created by merging each sample with the digit of other class and both the samples are shifted up to four pixels randomly in each direction, resulting in a 36x36 image. We achieve a good result when set λ as 5, with an error rate of

[1] http://www.cs.toronto.edu/~tijmen/affNIST/

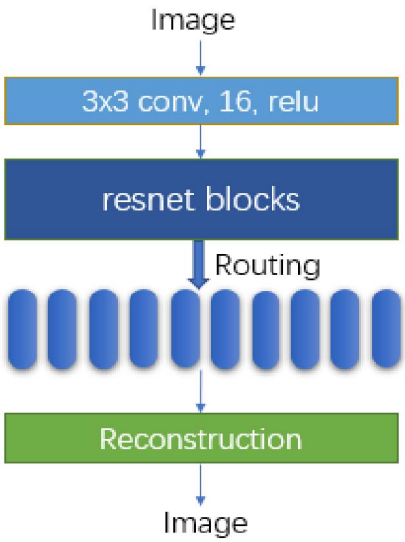

Figure 2. A CapsNet uses resnet as the backbone, which acts as a primary capsule extractor.

# method	error(%)	
	expanded MNIST	aff-NIST
baseline	0.77	21
R-1-32	1.70	**5.12**
R-5-32	1.76	**5.37**
R-10-32	2.29	**6.59**
R-15-32	3.81	**9.59**

Table 4. Results on aff-NIST dataset. R-x-y:routing with the λ as constant value x, and the number of neurons in each capsules is y.

4.6% on the training set and 5.2% on the test set, which is similar to the result in the paper [17], but the model is much smaller with a size of only 2.2M compared to the original capsule network.

5. Conclusion

In this paper, we propose an implementation of dynamic routing algorithm which can be flexibly applied to traditional CNN architecture as a pooling layer. We do experiment on different kinds of models by applying our proposed dynamic routing as a pooling layer, the results indicate that our dynamic routing can achieve better performance compared to the max/average pooling layer. And with the coupling coefficients c generated by the routing, our method can provide visual explanations to some extent. The replicated experiment on aff-NIST and multi-MNIST datasets indicate that the model does pooling with dynamic routing is also robust to affine transformations, with less model size than the original capsule network. Our future works include better understand the nature of routing-as-pooling, and this might lead to new ideas of building models with better interpretability.

References

[1] M. T. Bahadori. Spectral capsule networks. 2018.

[2] K. Duarte, Y. Rawat, and M. Shah. Videocapsulenet: A simplified network for action detection. In *Advances in Neural Information Processing Systems*, pages 7610–7619, 2018.

[3] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[4] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

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

[6] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[7] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

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

[9] J. E. Lenssen, M. Fey, and P. Libuschewski. Group equivariant capsule networks. In *Advances in Neural Information Processing Systems*, pages 8844–8853, 2018.

[10] H. Li, X. Guo, B. DaiWanli Ouyang, and X. Wang. Neural network encapsulation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 252–267, 2018.

[11] Y. Liu, H. Li, and X. Wang. Rethinking feature discrimination and polymerization for large-scale recognition. *arXiv preprint arXiv:1710.00870*, 2017.

[12] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[13] D. Peer, S. Stabinger, and A. Rodriguez-Sanchez. Training deep capsule networks. *arXiv preprint arXiv:1812.09707*, 2018.

[14] S. S. R. Phaye, A. Sikka, A. Dhall, and D. R. Bathula. Multi-level dense capsule networks. In *Asian Conference on Computer Vision*, pages 577–592. Springer, 2018.

[15] R. Ranjan, C. D. Castillo, and R. Chellappa. L2-constrained softmax loss for discriminative face verification. *arXiv preprint arXiv:1703.09507*, 2017.

[16] S. Sabour, N. Frosst, and G. Hinton. Matrix capsules with em routing. In *6th International Conference on Learning Representations, ICLR*, 2018.

[17] S. Sabour, N. Frosst, and G. E. Hinton. Dynamic routing between capsules. In *Advances in neural information processing systems*, pages 3856–3866, 2017.

[18] A. Shahroudnejad, P. Afshar, K. N. Plataniotis, and A. Mohammadi. Improved explainability of capsule networks: Relevance path by agreement. In *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 549–553. IEEE, 2018.

[19] D. Wang and Q. Liu. An optimization view on dynamic routing between capsules. 2018.

[20] F. Wang, J. Cheng, W. Liu, and H. Liu. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 25(7):926–930, 2018.

[21] F. Wang, X. Xiang, J. Cheng, and A. L. Yuille. Normface: l 2 hypersphere embedding for face verification. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1041–1049. ACM, 2017.

[22] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[23] W. Zhao, J. Ye, M. Yang, Z. Lei, S. Zhang, and Z. Zhao. Investigating capsule networks with dynamic routing for text classification. *arXiv preprint arXiv:1804.00538*, 2018.