Critique of Dynamic Routing Between Capsules

Srikanth Sarma
ME14B027

January 29, 2018

Abstract

In this critique document, I would be summarizing the contents of the paper [1] and also critique the shortcomings of the paper. Finally, I would be discussing the possible extensions of the paper [1].

1 Summary

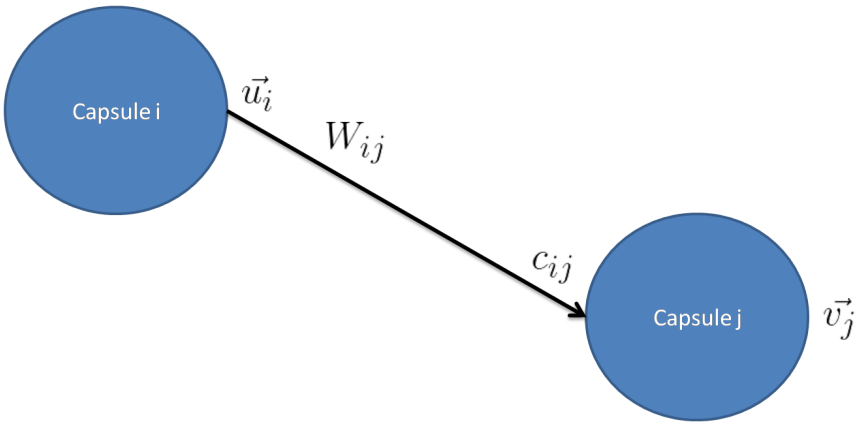

Figure 1: The capsule i and capsule j learn corresponding features feature i and feature j. The characteristic of a typical connection in a capsule network is a weight matrix W_{ij} and a coupling coefficient.

The paper starts with a revolutionary concept of using vector valued activations in networks instead of the standard scalar valued activation used in neural networks. The basic activation elements in these networks are termed as capsules (hence the name capsule networks). At a first look, capsules might just look like a bunch of neurons. But, the way the activations are fed to the further layers and the way the network is trained makes them represent the pose (configuration/ orientation) of the features being detected which differentiates it from a bunch of neurons. The magnitude of this vector is a measure of the probability of having detected the feature (corresponding to the capsule) while the direction is the pose of the feature being detected. These capsules are arranged layer wise with connections from one layer to next just like in a neural network. Unlike in a neural network, the connections convey vector information instead of a scalar. The connections are characterized by a weight matrix W_{ij} (analogous to scalar weights in neural networks) and a scalar coupling coefficient c_{ij}. The schematic shown in figure 1 shows a typical connection in a capsule network. The mathematical equations governing the connection of the capsule i to capsule j are as follows:

$$\vec{u_{j|i}} = W_{ij}\vec{u_i} \tag{1}$$

$$\vec{s_j} = \sum_i c_{ij}\vec{u_{j|i}} \tag{2}$$

$$\vec{v_j} = \frac{||\vec{s_j}||^2}{1+||\vec{s_j}||^2}\frac{\vec{s_j}}{||\vec{s_j}||} \tag{3}$$

Here, $\vec{u_{j|i}}$ is the prediction of pose of the feature j given the pose of the feature i. $\vec{u_i}$ is the pose of the feature i, c_{ij} is the coupling coefficient and W_{ij} is the transformation matrix projecting the pose of feature i on to the space of the pose of feature j. This way of using linear transformation as a basis for prediction of higher level features from lower level features is inspired from inverse graphics.
The later part of the paper focuses on the methodology adopted to train the network. Training has two parts to it. One is to minimize a loss function while the other is to assign significance of the pose of the lower level features in predicting the pose of the higher level feature being detected in the immediate next layer. In other words, estimating W_{ij} to reduce the required cost while assigning values of c_{ij} based on the significance of capsule i on capsule j. The loss function not only contains the cross entropy error for classification but also a reconstruction error defined by the error between the actual image and the image reconstructed from the pose vector of the object of the correct class. This reconstruction is done by a decoder which can typically be a multi layered neural network. This reconstruction error is the reason for the activation vectors to attain their interpretation as the pose of the feature being detected. To estimate the values of the coupling coefficients c_{ij}, a "routing-by-agreement" technique is used. This simply means that low level capsules will be given more importance if their prediction of pose of the high level feature is similar to the actual pose. Cosine similarity is used to quantify the similarity. To update the coefficients c_{ij}, for a given j, the current values of c_{ij} are considered to represent the prior distribution of importance/ significance while the softmax of the cosine similarity represents the likelihood of the importance/ significance. Hence, the product of these two would by Bayes theorem represent the posterior distribution of importance/ significance of the different capsules connect to a particular capsule. The mathematical equations corresponding to this "routing-by-agreement" algorithm are as follows:

$$c_{ij} \propto c_{ij} \exp\left(\vec{u_{j|i}} \cdot \vec{v_j}\right) \tag{4}$$

To normalize c_{ij},

$$c_{ij} = \frac{c_{ij} \exp\left(\vec{u_{j|i}} \cdot \vec{v_j}\right)}{\sum_i c_{ij} \exp\left(\vec{u_{j|i}} \cdot \vec{v_j}\right)} \tag{5}$$

The weight matrices W_{ij} are estimated solely based on minimizing the loss function. Thus, all parameters of a capsule network can be estimated given the data. In the paper, a capsule network is trained and is shown to perform better than the state of art networks on the MNIST data set. After training on MNIST it has been shown how the different components of the activation vector actually represent different components of pose of features like the skewness, thickness, width etc of strokes in written digits.

Because Capsule networks use pose of lower level features to estimate the pose of higher level features, it is claimed that they would perform better in detecting multiple objects (highest level features). To substantiate this claim, a multiMNIST data set was generated by overlapping different images from MNIST. The performance of capsule networks on this dataset proved the earlier claim.

2 Critique

1. In the network trained over MNIST dataset, the dimensions of the activation vectors in primary layer capsules were half of that of the activation vectors in secondary (final) layer capsules. This choice of dimensions was not justified within the paper.
   Possible explanation:
   It is intuitive that lower level features need only lower dimensional pose while higher level features need higher dimensional pose since higher level features are a collection of lower level features. For example, if face is a higher level feature, eyes, nose and ears are lower level features. The pose of a face would need to include the include the pose of all the individual low level features while also capturing the spacial relationships between these features.

2. The need for a coupling coefficient has not been justified. Having $\vec{s_j} = \sum_i W'_{ij} \vec{u_i}$ instead of $\vec{s_j} = \sum_i c_{ij} W_{ij} \vec{u_i}$ would've been convenient for direct back propagation.
   Possible explanation:
   The weight matrix W_{ij} and the coupling coefficient c_{ij} play different roles while training.

The coupling coefficient takes care of the significance of the prediction of the pose of the high level feature compared to the predictions from other capsules. Whereas the weight matrix takes care of producing the best prediction of the pose of the high level feature given the pose of a low level feature. If W_{ij} and c_{ij} were unified, the comparative significance of the predictions will not be considered. Hence, having both W_{ij} and c_{ij} might improve the total prediction of by means of the weighted average.

3. Estimation of the pose of the high level feature needs only the pose of the low level feature. But, in (1), the probability of having detected the low level feature is also taken into account.
Possible explanation:
Inclusion of the probability might have been done for two reasons. One is that this is the only way the information of the probability of having detected the low level feature passes on through layers. Obviously, the probability of having detected a high level feature must depend on the probability of having detected the corresponding low level features. This is exactly similar to what happens in typical CNNs. Other reason might be to weigh the predictions based on the probabilities of the corresponding low level features being detected. We don't want a low level feature which is barely detected to dominate the prediction of the high level feature.

4. An AGI would want to learn all on it's own and hence be able to get rid of the hyper-parameter declaration. But, capsule networks introduce new hyper-parameters in the form of dimension of the pose vector of each capsule.

3 Extensions

1. Dynamic sizing of dimensions of the vector activation of capsules:
Currently, the dimensions of the pose (vector activation) is a hyper-parameter to be decided before hand. In the current paper [1], all features in a single layer were said to have pose of the same dimensions. This need not always be true. A few features might need more pose dimensions than others. So, instead of assigning dimensions before hand, we must find a way to learn the pose vector dimensions based on the generalization ability it offers. One Naive (computational nightmare) way is to perform a grid search in the parameter space (integer) containing the dimensions of the pose vectors.

2. Classify actions performed by objects:
Since capsule networks can learn the pose of the objects being detected, a video, sequence of images can be reduced into a time series of the pose of the object. This when coupled with a LSTM can probably do better at classifying actions performed by an object. For example, dancing can be seen as a rhythmic movement of hands and legs which will be reflected in the time series of the pose vector.

3. Reconstruction of the corresponding 3D object as a regularization instead of just the 2D image:
As mentioned in [2], reconstruction of the entire 3D image instead of just the 2D image might serve as a better regularization technique since reconstruction of the 3D object would need more precise pose vectors to be learned.

References

[1] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *Advances in Neural Information Processing Systems*, pages 3859–3869, 2017.

[2] Edgar Xi, Selina Bing, and Yang Jin. Capsule network performance on complex data. *arXiv preprint arXiv:1712.03480*, 2017.