ID2223 – Scalable Machine Learning and Deep Learning

Final Project - Capsule Networks

Course Coordinators:
Amir H. Payberah
Kamal Hakimzadeh
Jim Dowling

Group *ScalableGroup*

Mohamed Gabr
Óttar Guðmundsson

2019-01-27

Introduction

In November 2017, Hinton et al. [1] proposed a new type of network that could potentially solve the problems of standard convolutions. Introducing a new concept to networks called Capsules, they output a vector instead of a single value or matrix, allowing the possibility to choose a parent in the layer above the capsule that it is sent to. For each potential parent, the capsule network can increase or decrease the connection strength for the identified object called routing by agreement as seen in Figure 1. This method is supposedly much more effective at adding invariance than the primitive routing introduced by max-pooling.

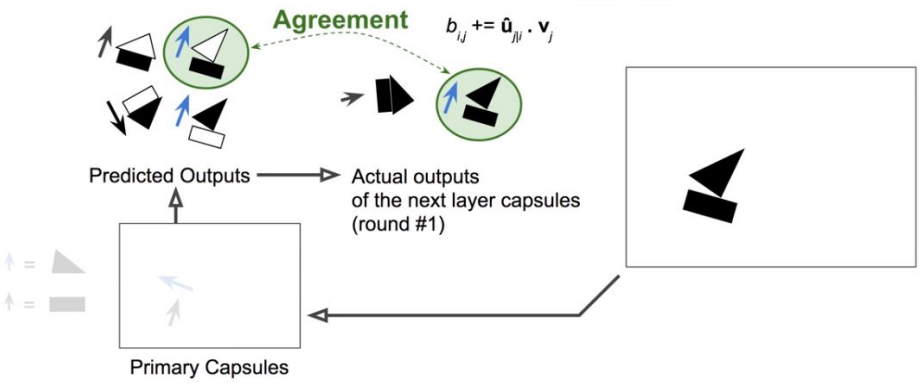

Figure 1: A demonstration of the vector outputs predicting a boat
as demonstrated by Aurélien Géron [2].

By outputting vectors of the identified features in Figure 1, the capsules pick the parent with the strongest connection for the first iteration. In further iterations, it manages to combine higher level routed features to form an object that will finally be classified by the network.

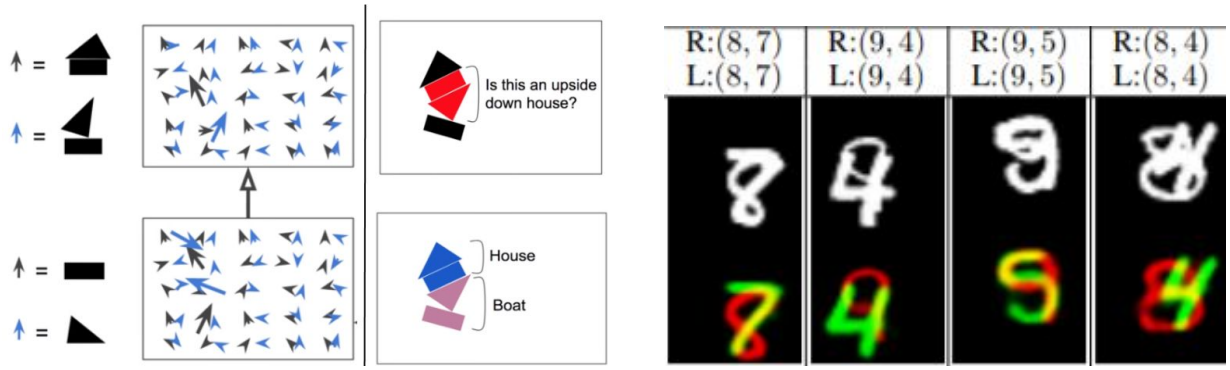

Figure 2: To the left, one of the problems is presented in a simplified case. If two features overlap that might confuse traditional networks, the routing algorithm still manages to capture both objects as the agreement factor for a house and a boat is stronger than an upside down house. On the right, two digits have been stacked on top of each other presenting a cluttered problem. The capsule network successfully captures the features of both objects and colors the different relative features to their object.

Pros and cons mentioned in external research

The new concept that capsule networks present have numerous benefits, making it unique compared to an MLP and CNN. In the original paper, an accuracy of 99.8% is achieved on the MNIST dataset [3] which is the state of the art and got close to 90% on CIFAR10 [4] by increasing the number of primary capsules and routing rounds. Fewer data points are needed to train the network, as data augmentation methods like image rotation and flipping are not needed since the network captures rotation, thickness, and skewing. As the routing by agreement technique considers the preserved position and the pose of features, it can account for overlapping objects as demonstrated in Figure 2. The downsides are that this structure does not work well on complex data as seen on CIFAR10. One of the big issues is that capsules try to account for everything in the image [5], so when the backgrounds are too varied, the network does not manage to capture a good model to account for all the backgrounds. The network has not been tested on ImageNet and bigger images, which might result in disappointing results just as it did with CIFAR100 where it reached only 18% accuracy [6]. Furthermore, the training is considerably slower due to the inner loop of the algorithm when applying the routing by agreement.

Functions

In the paper, he mentions that there are few different ways to represent the capsules. In this example, we followed the original paper as close as possible. We present explanations of the most important functions in the implementation.

def squash(s, axis=-1, epsilon=1e-7, name=None):

The length of the output vector of each capsule represents the probability of an entity, that is represented by the input to the capsule. A non-linear squashing function is used to shrink short vectors to nearly zero while the larger ones are closer to 1. This makes the capsules only capture the most relevant features.

def primarycapsule(output_vec_no, output_vect_length, cap_dims, X):

The primary capsule is composed of two convolutional layers. As an input, the data is given, as well as the dimensions of the output vector for each capsule and the number of maps. For each capsule, there are 32 vectors of 8 dimensions each. In our case, we had 36 capsules, which means that we had 1152 vectors of 8 dimensions each for each input.

def digitalcapsule(primary_out, digital_cap_no, digital_cap_dim):

The idea here is to compute one prediction per digital capsule for each of the outputs of the primary capsule. The number of digital capsules is determined by the number of classes, in our cases it is digits and they are 10. Thus, we have 10 digital capsules each outputting a prediction of dimension 16. This is obtained by having weights that are trained to pass from primary capsule i to digital capsule j. Each weight is actually a matrix since we are passing from a vector of size 8 to a vector of size 16. Since we have 1152 primary capsules and 10 digital capsules, we have 1152*10 weight matrices of size 8*16 to be trained. This is the role of the digital capsule. Take 1152 inputs of size 8 and output 10 vectors of size 16 for each of the 1152.

def routing_1(batch_size, digitCapsules):

In the round 1 of the routing by agreement, we pass from having 1152 vectors of size 16 per digital capsule, to one vector of size 16 per digital capsule by doing a weighted sum between of the 1152 vectors. We end up having one vector of size 16 per digital

capsule. Since we have 10 digital capsules, we have 10 vectors of size 16. It is important to note that weights here are scalars.

def routing_2(initialWeights, digitCapsulesOutput, digitCapsulesPrediction):

In round 2 of the routing agreement, we are also doing a weighted sum the same way like in round 1. The only difference is that the weights represent the similarity between each of the 1152 vectors and the real output. We obtain this similarity by doing the scalar product (which gives out a scalar in this case).

def margin_loss(routingInput, mPlus = 0.9, mMinus = 0.1, lambdaVal = 0.5):
For capsules, the length of the initial vector represents that probability that entity for capsule exists. However, the top capsule should be the same length as the classes for k if it is present in the image. Thus the network needs its own special loss function that allows for multiple digits.

$$L_k = T_k \ \max(0, m^+ - ||\mathbf{v}_k||)^2 + \lambda \ (1 - T_k) \ \max(0, ||\mathbf{v}_k|| - m^-)^2$$

Note that the λ is the down-weighting for loss of absent digits. The total loss is then the sum of the losses of all digit capsules in an image.

Results

Learning Rate	Accuracy Without Regularization	Accuracy With Regularization
0.001	10%	12%
0.0001	99.08%	99.06%

Comments: We can see that the learning rate 0.001 was too high, which may have lead to an overshooting of the local minima. By simply dividing the initial learning rate by a factor of 10, we were able to obtain very high accuracy. The choice of this performance indicator is motivated in the discussion.

Discussion

The capsule network only demonstrated its effectiveness for the MNIST dataset and mentioned its problems regarding ImageNet, suggesting that capsules should mainly consider images as an input. It did not mention using the architecture for any other type

of data such as audio or text, meaning that there is a whole lot of work to be done. In the paper, the features of an image captured by a CNN are mapped together. These features are used as an input to the primary capsule, which manages the vector directions for classifications. In modern text classification such as Word2Vec [7], the relations of words are also captured using vectors. Maybe capsules can serve as a new technique for other data classification, such as NLP. If not, data points can still be plotted into images, like audio files as spectrograms, and trained using capsule networks.

Regarding the obtained results, we can clearly infer that the reconstruction step is not needed to obtain good accuracy results. Reconstruction in this network plays the role of regularization, to prevent overfitting. It is also important to reflect that we chose accuracy as a performance indicator since our dataset is not biased to one class more than another, which means that if the classifier is not working correctly accuracy will still point out this anomaly.

When we tested out RELU, we found out that there is no need to pass to the leaky version, so we stayed with the normal one. ReLU is a good choice since it is a non-saturating activation function.

Finally, to accelerate the training speed, we added momentum, the Adam optimizer to be more specific. We manually changed the learning rate until we obtain a good descent. It was not hard to find one and a grid search was not needed.

Further Improvements

Different activation functions can be tested, and see which one performs the best. Additionally, using a cluster service like the Google cloud platform will accelerate the training process, and hence enable possibilities of performing coarse-fine searches on the hyperparameters and test different functions.

References

[1] Sabour, Sara, Nicholas Frosst, and Geoffrey E. Hinton. „Dynamic Routing Between Capsules“. arXiv:1710.09829 [cs], 26. október 2017. [Online]. Available: http://arxiv.org/abs/1710.09829. [Accessed: 04-Dec-2018]

[2] Aurélien Géron, Capsule Networks (CapsNets) – Tutorial. [Online]. Available: https://www.youtube.com/watch?v=pPN8d0E3900&list=WL&index=81&t=0s. [Accessed: 17-Dec-2018].

[3] “MNIST handwritten digit database, Yann LeCun, Corinna Cortes and Chris Burges.” [Online]. Available: http://yann.lecun.com/exdb/mnist/. [Accessed: 17-Dec-2018].

[4] “CIFAR-10 and CIFAR-100 datasets.” [Online]. Available: https://www.cs.toronto.edu/~kriz/cifar.html. [Accessed: 17-Dec-2018].

[5] E. Xi, S. Bing, and Y. Jin, “Capsule Network Performance on Complex Data,” *arXiv:1712.03480 [cs, stat]*, Dec. 2017. [Online]. Available: https://arxiv.org/pdf/1712.03480.pdf. [Accessed: 17-Dec-2018].

[6] R. Mukhometzianov and J. Carrillo, “CapsNet comparative performance evaluation for image classification,” *arXiv:1805.11195 [cs, stat]*, May 2018. [Online]. Available: https://arxiv.org/ftp/arxiv/papers/1805/1805.11195.pdf. [Accessed: 17-Dec-2018].

[7] S. Banerjee, “Word2Vec — a baby step in Deep Learning but a giant leap towards Natural Language Processing,” *Medium*, 12-May-2018. [Online]. Available: https://medium.com/explore-artificial-intelligence/word2vec-a-baby-step-in-deep-learning-but-a-giant-leap-towards-natural-language-processing-40fe4e8602ba. [Accessed: 17-Dec-2018].