Experimentation on Outlier Detection and Zero Shot Learning using Siamese and C2C-Siamese Network

Mahesh Latnekar mrlatnek@iu.edu

Yash Kumar yashkuma@iu.edu

Sumanyu Garg sumgarg@iu.edu

Hrishikesh Paul hrpaul@iu.edu

Abstract

We consider two related problems of Outlier Detection and Zero-Shot Learning (ZSL) using Siamese Network (SN) and Case-to-Case Siamese Networks (C2C-SN). The present methods of ZSL and outlier detection use semi-supervised techniques like transfer learning, data augmentation by adding synthetic or out-of-distribution images, vector embeddings for training. These methods will fail for a novice observation. We propose a method based on Case-Based Reasoning (CBR) using simple Siamese Networks and Case-to-Case Siamese Networks (C2C-SN) for the task of Fully Unsupervised Outlier Detection and Fully Unsupervised Zero-Shot Learning.

Note: C2C-Sn architecture was developed by Ye et al. [1]

1. Introduction

There has been a significant improvement in the task of visual object classification due to the availability of large-scale datasets like ImageNet and improved computational resources like GPUs and TPUs. Deep Convolutional Neural Networks (DCNNs) have been trained on large datasets for multiclass classification on over 100,000 categories of images. Creating such datasets requires extensive human labor in terms of image collection and annotation. Consequently, there exist many categories of objects for which large datasets might not exist. Additionally, we might encounter a novel observation, such as a novel disease, in which case we do not have much information about it.

As a result, DCNNs are not trained on such categories before being deployed for commercial purposes. Many DCNNs deployed in applications like self-driving cars and medical image analysis are prone to throw erroneous predictions when tested on such images, which might result in harmful, even fatal outcomes. In our work, we test the capabilities of Siamese Networks and Case-to-Case Siamese Networks in Outlier Detection. We find that classifying objects as outliers is not sufficient for certain tasks, and many applications demand differentiation between unseen classes. Furthermore, we provide a framework to differentiate between different unseen objects and classes, which we describe as Zero-Shot Learning.

For this task, we use three important methods – 1)Case to Case Approach- Pattern of similarities and differences between the two classes is unique. 2) Vector of Votes: Results from multiple votes of agreement and disagreement between different Networks, 3) Case-Based Reasoning: Explaining something a network has not seen by linking it to something it has seen.

We further explain Siamese Networks and Case-to-Case Siamese Networks[1].

1.1. Siamese Networks

Siamese Networks are a special type of neural network architecture that learns to differentiate between two given inputs. It consists of two identical neural networks that take one input image each. The output layer of the two networks is then passed into a contrastive loss function, which calculates the similarity score between the two input images. The architecture is shown below:

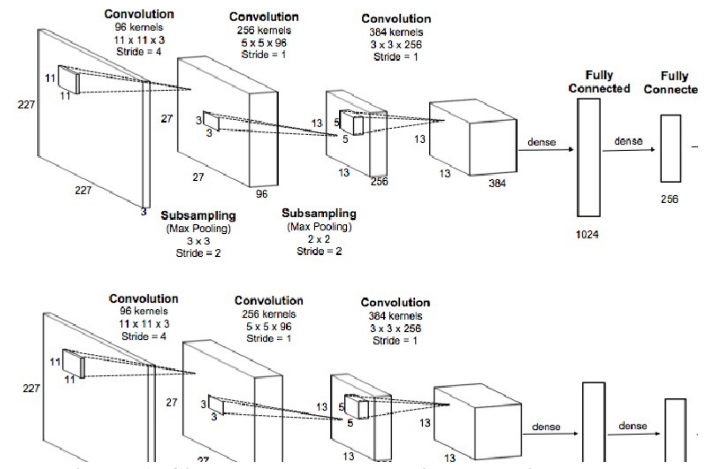

Figure 1: Siamese network architecture [3]

Traditionally, a neural network learns to predict a class if it has seen the class examples during training. This poses a problem when we add/remove new classes to the data. In this case, we have to update the neural network and retrain it on the whole dataset. The siamese neural network capitalizes on powerful discrimination features to generalize the predictive power not just to new data but to entirely new classes from unknown distributions. The main idea behind siamese networks is that they can learn useful data descriptors that are passed to subsequent networks for comparison. For example, considering MNIST, there will be (0-9) classes that give a total of 100 different combinations that can be passed into the Siamese network. The output layer is a linear classifier that classifies whether they belong to the same class or not. In the subsequent section, we aim to differentiate two classes using One-shot learning and Zero-shot learning.

1.2. Case-to-Case Siamese Networks

C2C-SN employs the advantages of feature extraction from a simple Siamese network to provide Case-Based Reasoning (CBR) using the Class to Class (C2C) approach. CBR methods provide an explanation of new problems by linking them to old experiences, i.e., they use patterns learned from seen classes to interpret new classes. CBR systems do this by matching patterns of similarity and differences of unseen classes with those of seen classes. Recently, there has been a growing interest in using CBR systems to provide explanations of black-box systems like neural networks[2]. To facilitate CBR, the difference of the high-quality output feature vectors from the twin networks of the siamese networks are then processed by another neural network which learns between class patterns. On the other hand, the class to class approach is based on the assumption that there exists a unique pattern of similarity and difference between two specific classes. As an intuition, consider the similarity and differences between soccer and football and compare them to the same between soccer and baseball. Similarities between soccer and football would be- multiple athletes running simultaneously and rectangular fields, while the difference will be helmets or lack thereof. These similarities and differences will be different for soccer and baseball.

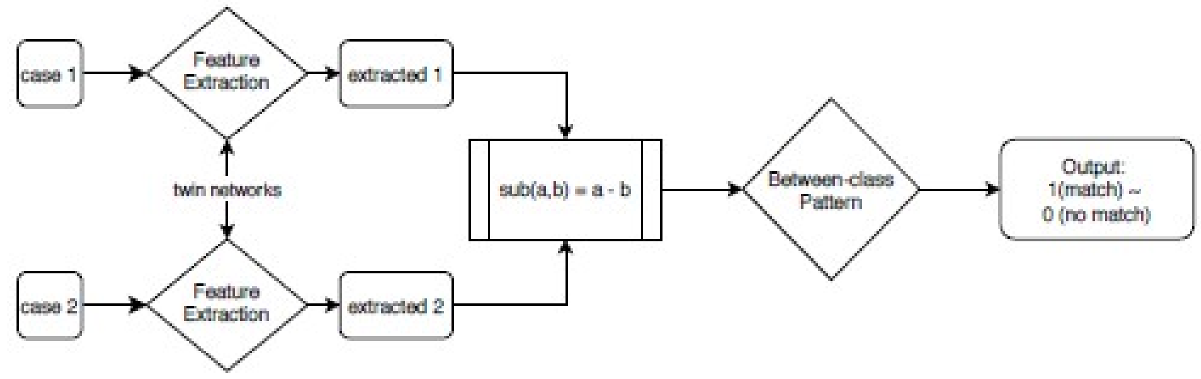

Figure 2: Case to Case Siamese Network Architecture [1]

C2C-SNs are twin networks used to extract features from pairs of classes. A single C2C-SN is reserved for a single unique pair of classes which gives us NxN networks for N classes. For example, for MNIST, there will be 100 unique networks for the 10 digits (0-9). Some of these networks will be 0-0 SN, 0-1SN, 1-0 SN, 5-6 SN, and so on. The general structure of a single C2C-SN is shown in Figure 2, the upper body is the same as the classic Siamese Network, but the lower distance function is replaced by a difference operator, which takes the difference between the extracted features. This difference is then passed to a neural network which learns the interclass pattern and outputs values between 0 and 1 to indicate the extent the extracted features match with the target pattern. For example, a well-trained 5-6 SN will output 1 if the case pair provided to it is 5-6, but if the case pair is cx-cy where cx!=5 and cy!= 6, then the output will be 0. C2C-SN is built on the premise that the pattern of similarity and difference between every pair of classes is unique, so to extract the features relevant to two specific classes, we require a network unique to those classes. Ye et al. [1] provide experimental validation that C2C-SN performs better at a few shot segmentations than normal SNs and also demonstrates the ability of C2C-SN to provide an explanation by case-based reasoning (CBR). In the next section, we discuss our approach to ZSL using C2C-SN.

2. Background and Related Work

2.1. Outlier Detection

In computer vision, we assume that image annotations are generated from a particular distribution of images. By using DCNNs, we try to model or replicate the distribution, which generates the annotations of a particular image. An outlier is an observation that diverges from the overall pattern of the data, and so, when an outlier image is processed by the network, the resultant annotations are no longer reliable. Additionally, modern neural networks, due to their complexity and the complexity of the data, find it difficult to distinguish between an anomaly and in-distribution examples. Moreover, DCNNs are incapable of recognizing when their prediction is wrong. Deep Neural Networks have been proven to predict a high probability for anomalies. This can create major problems as such architectures are being increasingly deployed in commercial applications like self-driving cars and medical image analysis. For example, if a self-driving car using visual object detection, captures an unseen object, the network might still recognize the object as one of the known cases. As a result, the network might pass harmful instructions to the car. Similarly, in medical image analysis, an anomaly can be misdiagnosed as a known medical phenomenon that might even prove fatal.

2.2. Zero-Shot Learning

For approaches like few-shot learning and one-shot learning, we require very few samples (~15) or just 1 example, respectively, in order to detect seen/labeled objects. Zero-shot learning methods go even further, in the sense that they require very few labeled/seen data points in the training phase. Zero-Shot Learning can be achieved by a method called transfer learning. In this, we take the knowledge already known from classes (in our case the seen examples) and apply it to those images of the unknown classes. Therefore, we can make use of some of the knowledge that a network has learned during some tasks and apply that knowledge to a completely separate task. For example, if a network has been trained to recognize objects like dogs and cats, some of the knowledge from that network can be used to detect x-ray scans. This paradigm is particularly important in the case of Zero-Shot Learning as the labels of the target classes are not seen. Having a network trained on certain examples, and using that knowledge, combined with some external knowledge (like changing the weights in the output layer), can be used to classify images that have not been previously seen by the classifier.
In traditional ZSL, features extracted from Convolutional Neural Networks are then mapped to semantic embeddings. These semantic embeddings include word vectors [8], text description[9], human gaze [10], etc. See Figure 1 for the structure of Traditional ZSL. If a ZSL is trained on 100 categories of animals and has a semantic embedding space describing 10000 animal categories, then if a NN is provided previously unseen animal which has a word vector embedding, traditional ZSL can still predict that animal. But the good performance displayed by traditional ZSL comes at the cost of expensive human labor. Embedding spaces like Word2Vec suffer from visual semantic discrepancy. For example, there are no relevant word vector encodings for classifying MNIST digits. In MNIST, a word vector of digit 5 may contain vector embeddings of its descriptions like 'numeric,' 'digit' etc. but these don't help in differentiating 5 from 6. Word vectors of 'sharp left corner,' 'curve at the bottom,' will be useful to differentiate 5 from 6, but these might not be necessarily present in the corpus which Word2Vec is trained on. To create such a corpus, someone would have to write a book describing the shape of numbers- a task that costs human labor and is not useful for any other application.
Additionally, traditional approaches show bias towards seen source classes, as shown in Figure 1.[2]

Zero-Shot Learning methods are divided into two groups based on the availability of labels of the target classes. They are inductive and transductive. In Transductive Zero Shot[6], [7] Learning, the training phase consists of the labeled source data and unlabeled target data and uses both the data to conduct zero-shot learning. Therefore, it is some-what a semi-supervised learning technique, where the train and the test data are available. On the other hand, Inductive Zero-Shot Learning [4],[5] consists only of the labeled source data during the training phase. In conventional settings, we use the unlabeled target data for testing. Whereas in the generalized setting, we make use of the labeled source data, as well as the unlabeled target data for testing. [2]

3. Methodology and Results

For all our experiments, the neural networks will be trained on the digits 0-5 (seen digits), and 6-9 will be out of a distribution or unseen digits.

3.1. Outlier Detection

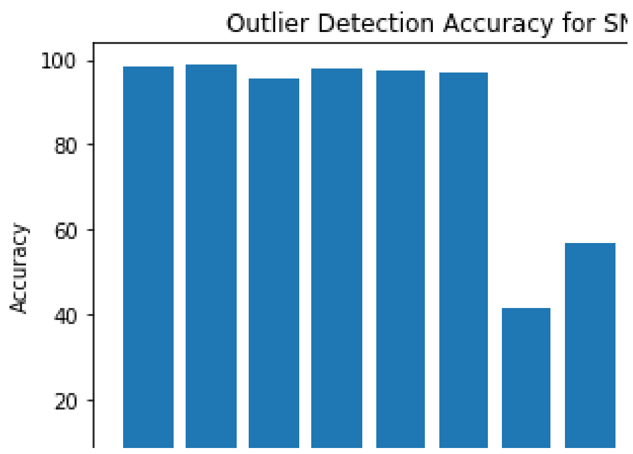

Figure 3: Outlier Detection Accuracy (SN)

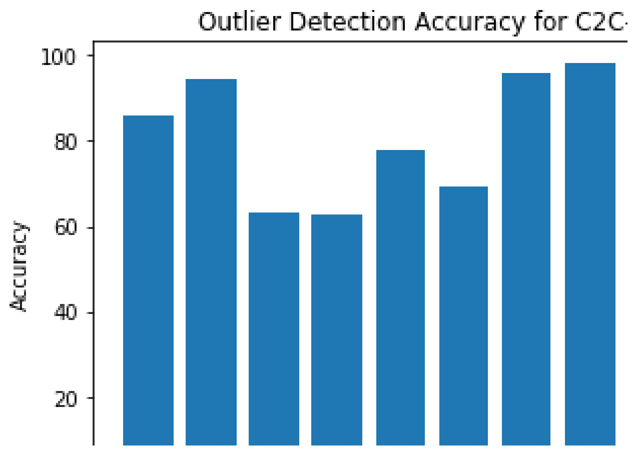

Figure 4: Outlier Detection Accuracy (C2C - SN)

3.1.a) With Siamese Network: We trained a simple 3 layer deep fully connected siamese network with 128 Relu activations in each layer. We use two dropout layers with a rate of 0.1 to avoid overfitting. Contrastive loss is used for backpropagation. The network is trained on an equal number of positive and negative pairs of each seen digit. During testing, we provide one of the twin networks with 10 anchor images of each seen digit. For a single test example, we compute the average of Euclidean distance between the outputs of the twin network overall anchor images of a seen digit. The test example matches the seen digit if this average is less than 0.1. This matching is carried out for all the seen digits, which results in a vector of length 6 for every test image. We tested 500 images each of the seen and unseen digits to obtain the results in Fig3.

Figures 5a and 6a show us the counts of positive matches for a seen and unseen digit, respectively. As can be seen, almost all positive matches of the test 0's are with anchor 0's. On the other hand, test 6's have multiple matches. The test image is therefore considered an outlier if its vector is a zero vector or has more matches than 1 or else it is considered a seen digit. The accuracies were calculated as follows-

For seen digits

$$\frac{\#\ of\ images\ of\ classified\ as\ inliers}{Total\ number\ of\ images}$$

For unseen digits

$$\frac{\#\ of\ images\ of\ classified\ as\ outlierss}{Total\ number\ of\ images}$$

3.1.b) With C2C-SN: The twin networks were similar to the simple Siamese Network with the upper Euclidean distance layer replaced by subtraction followed by a fully connected network to learn patterns between the extracted features. This output is then passed to a sigmoid function, which outputs values between 0 and 1. Here, every pair of seen digits is assigned a C2C-SN in the sequence 0-0SN, 0-1SN,....., 3-3SN,....5-5SN. Hence, we have NxN C2C-SNs for N seen digits, which give an NxN (6x6 in our case) length vector, with 1's for a match and 0's for a mismatch. Every such Ci-Cj network is trained on 5000 positive and negative pairs. Of the negative pairs, 35% are Cx-Cj pairs where x!=i, 35% are Ci-Cy pairs where y!=j and 30% are Cx-Cy pairs where x!=i and y!=j. During testing, we again had 10 anchor images of the digit Ci (in Ci-Cj) as an input image on the left network. A test image was passed through the right network of all C2C-SNs and the average of sigmoid value over the 10 anchor images was computed for the purpose of classification. Here, for a single test image, we'll get a 36 length vector, and the position assigned to individual C2C-SNs in this vector is important, as we classify an object as an outlier based on the activations at these positions. Refer to Figure 7 and 8a. The positions are arranged as follows- 0-0, 0-1, 0-2,......,5-4, 5-5, 5-6. There are 6 Ci's arranged at an interval of 6. We classify an image as an inlier if there are 6 positive activations at a distance of 6 from each other. Every other case is an outlier. We conduct this experiment with 500 images of each signed and unsigned digit. The result of the experiment is shown in fig.

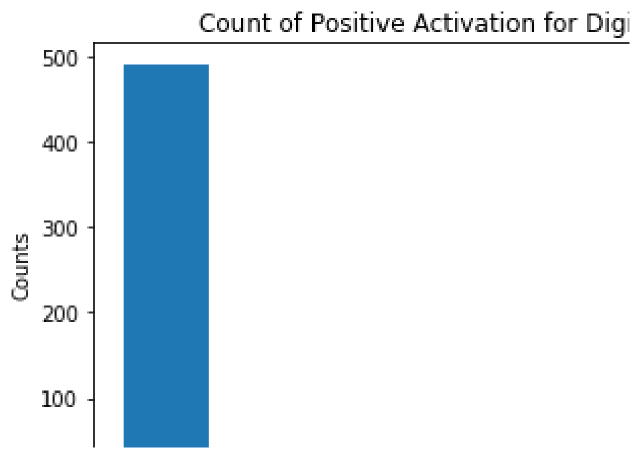

Figure 5a: Count of similarity for seen (SN)

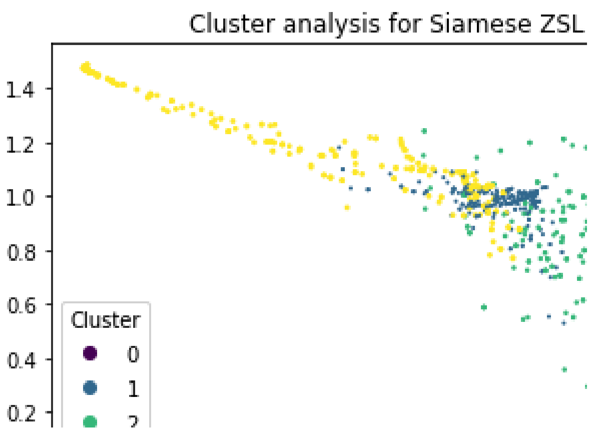

Figure 5b: K-means clustering for unseen digits

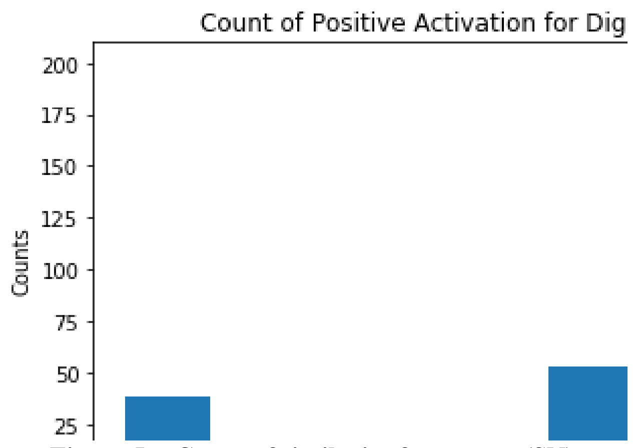

Figure 5a: Count of similarity for unseen (SN)

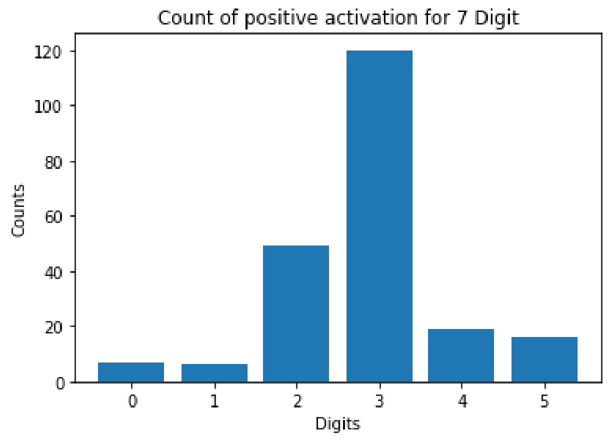

Figure 6b: Count of similarity for unseen (SN)

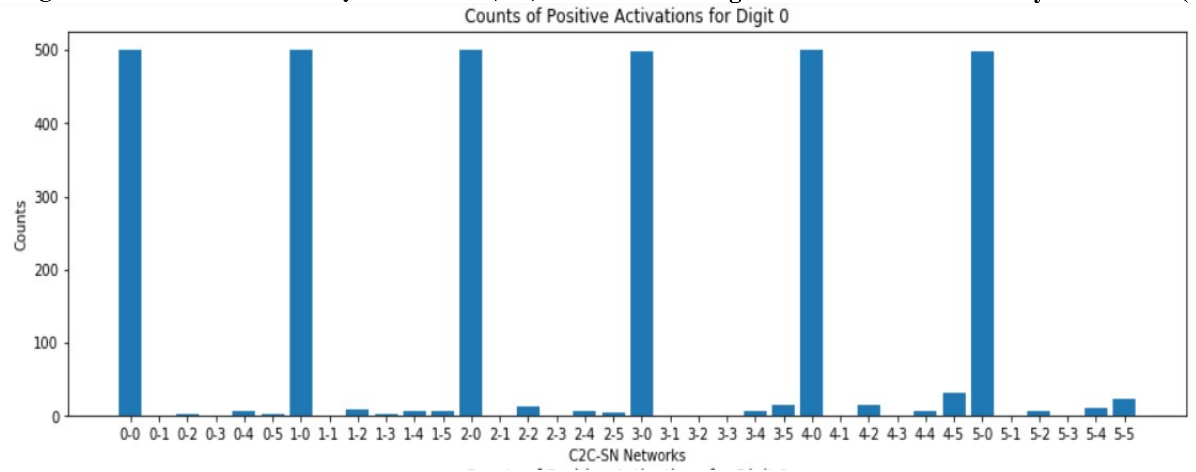

Figure 7: Count of similarity for seen (C2C-SN)

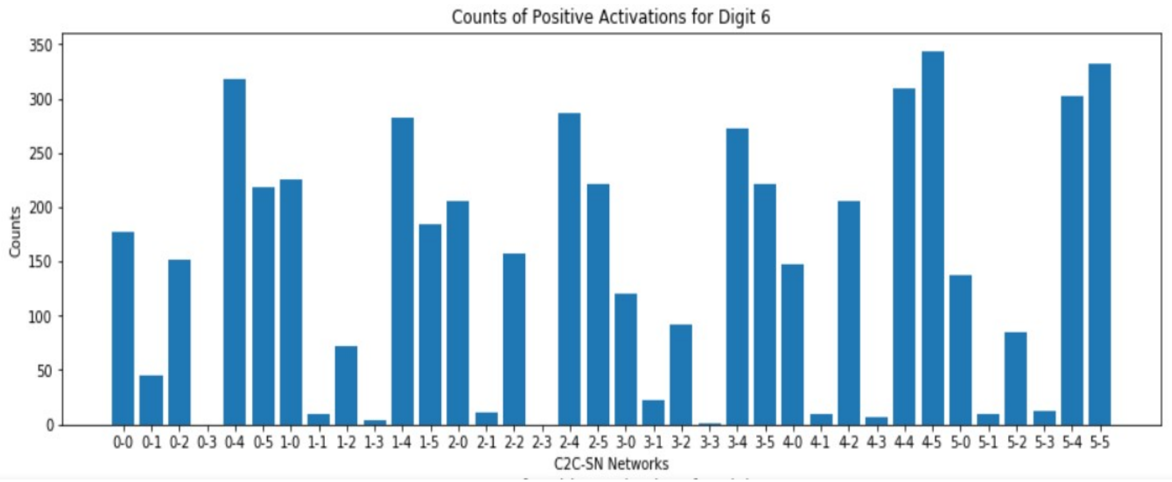

Figure 8a: Count of similarity for unseen (C2C-SN)

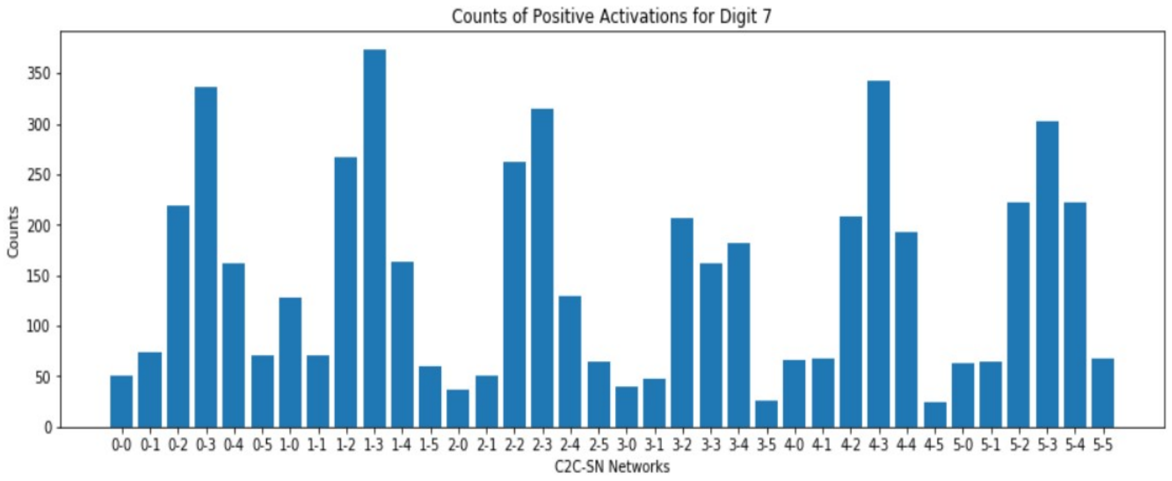

Figure 8b: Count of similarity for unseen (C2C-SN)

Unseen digit	Cluster mode (SN)	Cluster mode (C2C-SN)
6	3	2
7	2	1
8	0	3
9	1	2

Table 1

Cluster	Unseen digit mode (SN)	Unseen digit mode (C2C-SN)
0	8	6
1	9	7
2	7	9
3	6	8

Table 2

Put images of 2 different unseen count patterns.
In section 3.1.a and 3.1.b, we got vectors of size 6 and 36 respectfully, as outputs, which we then used to detect outliers. As we can see from Figure 6(a,b) and 8(a,b), the pattern of vectors for 6 & 7 are quite different. For siamese networks, 6's have most of their positive activations for 5,

4, and 0, while 7's have positive activations at 3. Similarly, for C2C-SNs, 6's have most of their positive activations for j = 0,4,5 in Ci-Cj. We thus take advantage of different patterns of these distance and sigmoid vectors, respectively, to cluster them as separate digits. This is a step forward from outlier detection to zero-shot classification. However, it is not a traditional zero-shot classification using semantic embeddings wherein we already have semantic vectors for out-of-distribution classes. We contend that our approach is more suited for the zero-shot label as it is completely unsupervised, whereas other traditional approaches are at best semi-supervised.

3.2.a) We performed K-means clustering on the output vector from experiments 3.1.a and 3.1.b. The results are shown in Table 1 and Table 2. In Table 1, we found out the mode of clusters for each unseen digit for both simple SN and C2C-SN. Similarly, in Table 2, we calculated the mode of unseen digits for each cluster. In Figure 5b, we plotted the results of Table 1 for SN in a 2d space to differentiate between each cluster. We have discussed the results and their importance in the discussion section.

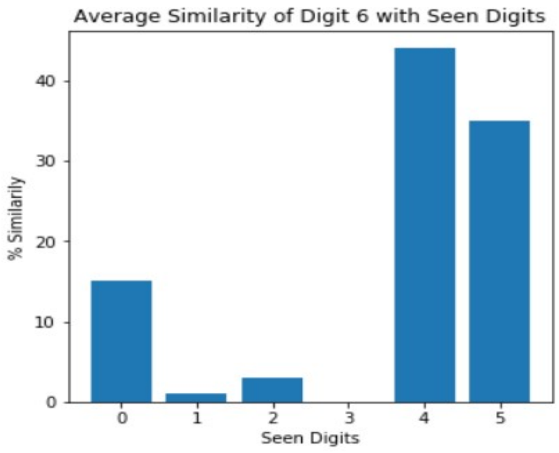

Figure 9: Mean % similarity for unseen digit

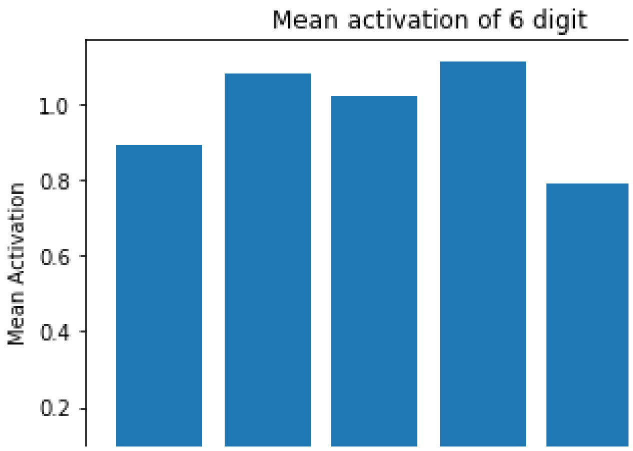

Figure 10: Mean difference similarity for unseen digit

3.2.b) Further, we used K- Nearest Neighbors classifier to fit the output vectors of the seen digits and predicted the output labels of unseen digits for the vectors corresponding to C2C-SN. In Figure 9, we can see that for 20 nearest neighbors, and most 6's got labeled 0, 4, and 5. This is a case-based reasoning approach to explaining unseen digits through seen digits. Similarly, for output vectors corresponding to the simple siamese network, we found the mean distance vectors for all unseen digits w.r.t the seen digits (See Figure 10).

4. Discussion

In outlier detection, for seen digits, we can see that the simple SN performs better than C2C-SN. This is because our conditions for detecting outliers are stringent, as discussed in experiment section 3.1. For the simple SN, only 6 networks have to agree with each other, whereas in C2C-SN, 36 networks have to agree with each other to detect an outlier, and there is a big room for error when it comes to seen digits. However, for unseen digits, C2C-SN performs better due to the same stringent conditions- it takes into account more votes than simple SN (36 vs. 6). Thus the probability of an unseen example having more than 6 positive activations spread periodically is low. As our idea of ZSL depends on outlier detection to work well, C2C-SNs are more suited for the ZSL task than simple SN. In the Zero-Shot Learning experiments (see Table 1), for simple SN, we get different cluster modes for each digit, whereas, in C2C-SN, there are only two digits with the same cluster mode. On the other hand (see Table 2, each cluster has a different digit mode for both SN and C2C-SN. We can concur that simple siamese networks and C2C-SN can produce different vector patterns for different unseen classes. This can be clustered to differentiate them from each other to achieve a fully unsupervised zero-shot learning. From experiment 3.2.b), it is clear that unseen or out-of-distribution patterns can be related to seen patterns. We can thus develop a Case-Based Reasoning method to identify unseen and novel observations. For example, given that 6 has been clustered with 0,4 and 5, we can intuitively imagine that 6 might have a curve at the bottom, like 0 and Similar experiments can be conducted on rare animal species and novel diseases by linking them to seen animals and diseases without using word embeddings which in many cases might not exist.

5. Further Improvements

We have conducted experiments on a relatively simple architecture of the Siamese Network and non-complex data. Additionally, our criteria for outlier detection was very stringent. Our clustering algorithms too are very basic. With more complex neural networks that are deep and specialized (example: DCNNs), with more complex data, light criteria, and specialized clustering algorithms, our method of fully-unsupervised outlier detection and zero-shot learning can lead to better results and more robust models. Case-Based Reasoning combined with black-box neural networks, can help explain properties of unseen and novel data.

6. Conclusion

We present a novel method of fully unsupervised outlier detection and zero-shot learning and provide a Case-Based Reasoning method to categorize and analyze out-of-distribution and novel observation. Our approach can be extremely useful in safety-critical applications like self-driving cars and many automated medical procedures. We show that the C2C-SN architecture can provide very good performance on outlier detection, and the Case-to-Case approach can be an effective way to develop Case-Based Reasoning systems. We provide experimental validation to our approach.

7. References

[1] Xiaomeng Ye, David Leake, William Huibregtse, Mehmet Dalkilic. Applying Class-to-class Siamese Networks to Explain Classifications with Supportive and Contrastive Cases

[2] Keane, M., Kenny, E.: How case-based reasoning explained neural networks: An XAI survey of post-hoc explanation-by-example in ANN-CBR twins. In: Case-Based Reasoning Research and Development, ICCBR 2019. Springer, Berlin (2019), in press

[3] Sanjeev J. Rao, Yufei Wang, Garrison W Cottrell. A Deep Siamese Neural Network Learns the Human-Perceived Similarity Structure of Facial ExpressionsWithout Explicit Categories

[4] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, T. Mikolov et al. Devise: A deep visual-semantic embedding model. In Advances in neural information processing systems, pages 2121–2129, 2013

[5] M. Norouzi, T. Mikolov, S. Bengio, Y. Singer, J. Shlens,A. Frome, G. S. Corrado, and J. Dean. Zeroshot learning by convex combination of semantic embeddings. In In Proceedings of ICLR. Citeseer, 2014.

[6] E. Kodirov, T. Xiang, Z. Fu, and S. Gong. Unsupervised domain adaptation for zero-shot learning. In Proceedings of the IEEE International Conference on Computer Vision, pages 2452–2460, 2015.

[7] Y. Fu, T. M. Hospedales, T. Xiang, and S. Gong. Transductive multi-view zero-shot learning. IEEE transactions on pattern analysis and machine intelligence, 37(11):2332–2345, 2015.

[8] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, T. Mikolov, et al. Devise: A deep visual-semantic embedding model. In Advances in neural information processing systems, pages 2121–2129, 2013.

[9] S. Reed, Z. Akata, H. Lee, and B. Schiele. Learning deep representations of fine-grained visual descriptions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 49–58, 2016.

[10] N. Karessli, Z. Akata, B. Schiele, and A. Bulling. Gaze embeddings for zero-shot image classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2017.

[11] Wei Wang, Vincent W. Zheng, Han Yu, Chunyan Miao. A Survey of Zero-Shot Learning: Settings, Methods, and Applications

[12] Jie Song , Chengchao Shen , Yezhou Yang , Yang Liu , and Mingli Song. Transductive Unbiased Embedding for Zero-Shot Learning