Machine Learning

SVHN

THINGS I SHOULD DO:

* grayscale the data
* add the LeNet architecture model images from the papers!
* Blog post: SVHN network as a detector? I think not.

1. Abstract
2. Introduction

* Hypothesis – Ensembles should beat the Neural Network
* SVHN data set
* Ensembles

1. Related Work

* SVHN papers (all 3)
* Find one or two ensemble papers
* Find a paper that talks about reusing existing Neural Nets for different applications (same architecture different application).

1. Augmented SVHN dataset

* Train contains 73,257 and Extra contained 531,131, so I augmented Train with 7 random rotations of +/-15 degrees per image.
* Final sizes are TrainAugmented 586056.
* TrainAugmented+Extra = 1117187 examples.
* Split TrainAugmented into 929989 for training and 187,198 for validation.
* Reason for augmentation is that I wanted more hard examples, and I didn’t want the network to get lazy by learning the easy examples. An alternative approach would be to simply duplicate train so that the network sees it as often as extra, but I didn’t want the miss an opportunity to add rotation invariance to the data.

1. Model

* Using the same model used to successfully train MNIST; LeNet
* The reseason for using the MNIST LeNet architecture is the similarity in the kind of data. They are both 32x32, and both about digit recognition.
* 1 normal network
  + Trained on train32x32, augmented train32x32, and extra32x32
* 1 ensemble of 20 networks with 10th the data each with ~200% data coverage
* first 10 ensembles are guaranteed covering 100%, where as the remaining 10 are random.

1. Results

* Single Network results:
  + accuracy = 0.908538
  + comparted to:

|  |  |
| --- | --- |
| Algorithm | SVHN-Test Accuracy |
| Binary Features (WDCH)  HOG  LeNetEnsemble / Augmented  Stacked Sparse Auto-Encoders  K-Means  ConvNet / MS / Average  LeNet / Augmented  ConvNet / MS / L2 / Smaller training  LeNet / Augmented / Background/ Gray  **LeNet / Augmented / Background**  ConvNet / SS / L2  ConvNet / MS / L2  ConvNet / MS / L12  ConvNet / MS / L4  **Deeply Supervised Nets** | 63.3%  85.0%  87.95%  89.7 %  90.6%  90.75%  90.85%  91.55%  91.95%  **92.77%**  94.28%  94.33%  94.76%  94.85% **98.08%** |
| **Human Performance** | **98.0%** |

As Reported by [8] and altered to include DNS results and my own

* Individual ensemble results
  + 85.00%, 83.08%, 84.62%, 86.15%, 86.54%,  
    83.77%, 82.70%, 85.00%, 84.62%, 81.92%  
    83.85%, 84.41%, 85.81%, 82.69%, 86.15%  
    84.23%, 86.54%, 86.15%, 85.69%, 82.31%
* Combined Ensemble result is 87.97%. This is 6.05% better than the lowest and 1.43% better than the highest.
* Agreements
  + Perfect/Near perfect agreement (>18)? -- 18905
  + Mostly agree(>10 and <=18)? -- 5705
  + Split (<11 No 50%+ majority)? -- 1422
* Compare to itself (1 vs 12ensemble)
  + Hypothesis potentially wrong, as ensemble does worse that single network. This might be because the upper limit of this architecture is 90.85% due to irreducible error cause by the model. A future hypothesis would extend the number of experts in the ensemble until this number is converged upon or passed. Only experiments can tell.
* <Compare to other papers results>
* <Compare to kMeans, MoG>

1. Conclusion
2. References

* Make a reference for all of the papers and cite the course for anything you use from the course material
* [8] Y.Netzer,T.Wang,A.Coates,A.Bissacco,B.Wu,and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

1. Supplementary (Solving the detection problem attempt 1)
2. Mini Intro
   * + The network trained on house numbers can also be used to detect house numbers with a slight modification to the algorithm; 32x32 sliding window on scaled versions of an image should produce detection regions of size 32x32, 64x64, and 128x128 on the scaled image when a label with low entropy is generated. If high entropy, then it is likely just background noise.
3. Related Work
   * + Find one paper, or look at how the current papers do it
4. Data Set
   * + SVHN again, but this time sliding window cropped 32x32.
5. Model
   * + Sliding window 32x32 crop 3 scales
     + Success when softmax produces confidence (low entropy)
     + My guess is that when the network sees a number it will predict it better than when it sees a background. Thus in all of the frames where a number is approximately centered, it will predict the number, and when it’s the background we’ll end up with a noisy mess.
     + Algorithm
       1. Crop full image into sliding window images of 32x32
       2. Run network on each crop image and build a matrix of classification for each center pixel
       3. Search through result for some to be determined nxn matrix containing same classification. Choose the center of the matrix as the locations of the 32x32.
6. Results
   * + The assumption about background results were wrong. The network is very consistent about certain kinds of backgrounds. Vertical edges get classified as ones, flat textureless surfaces get classified as zeroes. Comically, the network has created clusters.
     + Here is the result of the slide windows, where each pixel represents the center of a 32x32 image:  
         
       Where the original image is:  
       
     + The 9 classification is not obviously found in this solution
7. Conclusion
   * + The network is really good at classifying similar images the same way, and thus is not a candidate for detection. A follow up hypothesis might be along the lines of adding an 11th class which would be dedicated to background.
8. Supplementary (Solving the detection problem attempt 1)
   1. Mini Intro
      * The network trained for house numbers can be extended to be a network used for detection. To detect the difference between a number and background we can give the network some background samples, and create an 11th class for the network to train upon.
   2. Related Work
   3. Data Set
      * We will be using the Augmented SVHN data set as specified earlier, but this time we will be adding and labeling a set of data for an 11th class – background.
   4. Model
      * Sliding window 32x32 crop 3
      * With the new class, the network should be able to distinguish between background and non-background. This will yield the possibility for detection.
      * Algorithm
        + 1. Crop full image into sliding window images of 32x32
          2. Run network on each crop image and build a matrix of classification for each center pixel
          3. Search through the result in a linear fashion, and every time a point contains a classification, attempt to grow a circle as large as possible capturing neighbouring pixels.
          4. Find the maximum radius, and this is now your model citizen for radius sizes allowing for a 15% deviation in size.
          5. Filter the result to within radius size of 15% of the model radius.
          6. For each class, eliminate centers that have overlapping lapping radius=16 with other centers of the same class, thus finding the maximum sole radius.
          7. 32x32 Bounding box for this digit is top left=(x-16,y-16), dimension=(32,32).
   5. Results
      * Flaws in the algorithm, and remedies:
9. If the result is very flat, the radius will be small, allowing for a lot of bad predictions. Works better on square images.
   1. If the prediction space is less than than 32x32, then perform with ellipse instead of circle.
10. If the digits in the image are very large or very small, they will not be detected.
    1. Correct, and thus a pyramid style scaling technique can be used to extend this algorithm.
       * The assumption about background results were wrong. The network is very consistent about certain kinds of backgrounds. Vertical edges get classified as ones, flat textureless surfaces get classified as zeroes. Comically, the network has created clusters.
       * Here is the result of the slide windows, where each pixel represents the center of a 32x32 image:  
            
         It’s detections:  
         ../../../../Dropbox/cs2151_Project/detectionPredictions_withBackgroundClass.png  
         Over-layed:  
           
         Here is another example:  
           
         And it’s prediction solution:  
           
         Over-layed:  
           
         Applying the algorithm gives us:  
           
         Note that the 3 and 7 are captured, the rest is mostly background and there are some false predictions, but they are less round.  
         In this case the 3 and the 7 are larger than 32x32 bounding boxes and this could be solved via a scaling pyramid, which is beyond the scope of this venture and is open for later work.
       * The above is far superior to that without a background class. The white background is caused by the 11th class, so it appears that most of the background is indeed being classified as background. As you can see, the 9 is captured in the beige blob center-left at the bottom. Sadly, it looks as if our background dataset failed to capture enough of what constitutes a non-number “background,” thus we are getting misclassified backgrounds.
       * There is as much background in the data set as there is for each digit, but this clearly is not sufficient. If time had permitted, there would be a 10 times more background representation than any one digits.
    2. Conclusion