ML/AI Lab 5

1.3> XOR Dataset

Architecture: 1st layer is a Fully Connected Layer with 2 in-nodes and 4 out-nodes and Relu activation.

with random seed as 42, 4 was the minimum number of hidden nodes for >90% accuracy

With higher number of hidden modes (eg: 6 or 7) it is more than 90% for most random seeds.

Second layer is also a Fully Connected Layer with 4 in-nodes 2 out-nodes and softmax octivation.

The 2 out-nodes are the predictions.

Test accuracy with seed = 42: 98.8%. 99.2% for 7 hidden, 98.8% for 4 hidden rodes

Lr = 0.1

batch -size = 20

With 7 hidden nodes:

seed $421 \rightarrow 99.6\%$, seed $32 \rightarrow 98.4\%$, seed $3 \rightarrow 97.4\%$, seed $4 \rightarrow 98.8\%$, seed $5 \rightarrow 88.8\%$.

With 6 hidden nodes also most seeds were giving > 90%.

accuracy but 7 had a better success rate.

(6 can also be an ans-similar results to 7)

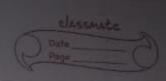
Note: In my wode I have used 7 hidden nodes.

So first layer has 7 out nodes,

so wond layer has 7 in nodes.

With 5 hidden nodes, seed 3 - 68.5%, 7-81%, 914 -> 72-71, 15-1858/.

So 6/7 is monimal.



V live dataset

Anhitecture: 1st layer is a Fully Connected Layer with 2 in nodes and 2 out-nodes and Relu activation. With random seed as 42, 2 was the minimum number of nodes for 40% accuracy.

With higher number of hidden nodes (eg: 3) or 6, it is >90%. for most random seeds.

Second layer is also a Fully Connected Layer with 2 in-rodes 2 out-nodes and softmax activation.

The 2 out-nodes are the predictions.

#epochs: 15

lr: 0.1

batch_size: 20

for 5 hidden nodes. Test acaracy with seed: 42: 99.4 %. for 2 hidden rodes

With 5 hidden nodes:

seed: 4 - 98-67. seed 3- 997. seed 7-, 97.97. seed: 71 - 997. seed 421 - 98.67.

Note: In my code 9 have used 5 hidden rodes.

So first layer has 5 out_rodes,

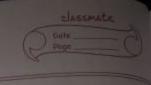
second layer has 5 in_nodes.

is also a valid and which gives seed 7-> 98.5%, 6-> 99.5%, 5-> 88.8%.

With 3 hidden nodes, seed 12 -> 76.3%, 17-> 80.8%, 34-> 78.5%, seed 5-> 80%, seed 12-> 76.3%, 17-> 80.8%,

59-> 78.2%

So 4/5 is minimal



MNIST

Seed

42

(e) Architecture: First layer is a Fully Connected Layer with 784 in-nodes and 15 out-nodes and ReLU activation.

Second layer is a Felly Connected Layer with 15 in nodes
15 out-nodes and Relu activation.

(Recall: It is well known convention to keep equal
number of hidden nodes across hidden layers)

Final layer is a Fully Connected Layer with 15 in-nodes, and 10 out-nodes representing 10 classes with softmax activation.

#epochs: 5

Lr: 0.1

batch_size: 50

Test Accuracy 93.49%. 92.51%.

94.147. 93.61%. 71 92.58%. 69

Average Test Acuracy: 93.27%.



() CIFARIO

Architecture: Input to first layer is the 3×32×32 image for each training example in the batch.

First layer is a ConvalutionLayer that takes $3\times32\times32$ input image and atputs a deep representation of Size $32\times10\times10$ by using 96 (32×3 -out-depth \times indepth) convolution filters of size 5×5 with stride as 3. Activation is ReLU. No padding-

2nd layer is Ang Pooling Layer which gives output of size 32 X4 X4 by using an average fitter of size 4 X4 with stride = 2.

3rd Layer is Flatten Layer with output of 512 nodes.

Final layer is Fully Connected Layer with 512 input nodes
and 10 cleares as the output with softmax activation

epoch : 50

(10 epoch infficient to cores 35%)

(10 epoch infficient to cores 35%)

batch_size: 50

Seed Test Acuracy 49.5%.

42 50.2% 6 49.1% 5 46.8%

5 40.01

Arg Test Acuracy: 49.647.

I have saved model p with seed = 4.
With seed = 42 is also given.

tab 5

Asignment 5

1. Task 1: CMVs take into account the spatial structure and classify using smaller number of parameters than MLP.

The intermediate layer of the CNN will be responsible for different features of the image which will help the lest softmax layer to activate a reticular category.

We will be referring to 3 pictures in Figure 2 as (1)(2) and (3).

In 1) for example, some activations will be responsible for identifying parts of the door, some will take care of the bornets. There will be layer and activations corresponding to headlight, front, windows, numberplate, logo, top, wheels, windshield etc.

In (2), the features taken to care by intermediate activations will be: handle, tyres, mimors, seate, engine-like shorter visible on the side:

Note that they there maybe subdivisions within them also.

(Learning different parts of a feature: spokes of wheels, gothe of heading ht, edges of mindscreen, boundaries of seat etc.

(3) is almost same as (1) modulo the shift so the identifiable features will remain same

Now, let us discuss helpful properties of CAN for this task

Note: we could have many feature maps in our network
leading to multiple parallel pipelines merging at the
fully connected layer. In this case, the individual pipelines
maybe responsible for separate features. (door/bonnet).

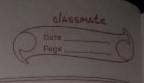
glot a features help in distinguishing size and location of image. Door one 3 man features of CRNS:
local connections through filters: neighboring signals more important than ones for away 94 helps to represent local structure of a feature.

(Eq: connections b/w doors and wheels in OSB or the connections b/w seat and engine of the bike in (2) weight/Parameter shoring: It holps in identifying recurring components (Eg: wheels, headlights, doors, mirrors esc.) It can be insualized as morning patches.

The intuition here is that neighbouring signals / zixels affect in a similar way of location. This in particular helps to identify (1) and (2) as

the same class since our CNN is robust to shifts Some other properties like max pooling provide a sharpening effect and will help to avoid the noise / background as in 2) It will help ignore the road from the bike image
It helps us to pick plat of many signals and is
locally translation invariant

Strides help in downsampling and moving pateres at
reader beginning. regular frequencies.



2. Task 2: A naire approach would be to classify parts of the image , independently.

Since we have annotated data, one thing we can do is - if we are expecting more than one object in the same image, we can have the model output the probabilities of each class rather than just outputting the predicted class. Then, the idea is that we can filter out very low probabilities and keep those scores above a certain threshold.

So, noted of using softmax, we should use sigmoid for our final layer so that each label is measured on its own menits (and not compared against its neighbours)

Other technique me can have labels with more than 1 1 in a now.

in a xow.

cg: [cor_label, bike-label] \in \left[\left[0, 0 \right] \left[0, 1 \right], \left[1, 1 \right] \right]

instead of only [0, 1] or \in \in \in 0 \right].

Some other complicated techniques exist like image segmentations

Some other complicated techniques exist like image segmentation and bounding boxes. (Algorithms like YOLO [You only Look Once] and SSD [Single Shot Multibox Detector])

They look at images and predict bounding boxes for the claves.

Since the classes are separated me can use this.

"mage segmentation" which classifies on a per-pixel basis can also be used. We may we pretrained models to first segment objects in the scene & then fifter out object regions using those segmented lakels.

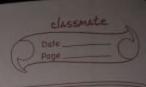
(This has limitation of "per-pixel".)

Interview important limitation is that we need with the objects its contained. The network be labelled be trained with some multi-vehicle images.

Setting the threshold in my first method is another concern. (Hyperparameter optimization may have to be used). This network will perform goodly on overlapped images. Class Imbalance is also a senous issue since it will be better if images with nell separated vehicles can be fed to the network during training else it might always predict one We may also use R-CNN model (Reg. on 8 with CNN features). which takes different regions and predicts class for the regions. dimitation is that since it is 2-stage (detect region then learn) it cannot be implemented real time. The search region time could lead to generation of bad candidate proposals since to learning happening at that stage.

Cropping may lead to loss of information.

It is complex (R-CNN) since we extract many regions for the image and the no. of CNN features also increases each image and the no. of CNN features also increases drastically.



3. Task 3: The condition mentioned is called occlusion.

We could do 2 things: I is train the classifier using gorbially

Research has been done in this area and it stars that such training reduces spatial support of filters Also, we can have a training process that uses regularization to shrink spatial support for filters. By smaller spatial support, we meen small support relation meen small support relative to fitters in networks trained

as usual.

An occlusion robust classifier should use features that do not vely on spatial support of entire object but only a part of it. If it does so it can be robust without being trained on secun occluded objects.

Interior May be it will not handle all aspects of occlusion.

We also need some localization to handle occlusion.

Returning to the first method, we could use dataset augmentation. But if me just change the detaset the accuracy will degrade with testing on high acclusions so we can also do something else

Integrating novel preprocessing stages to segment the input and inpaint occlusions can help. A CNN so modified will be effective even with high occlusion. Such a net north will also be more accurate on unoccluded images. These results depend on success ful segmentation. A good dataset would be where occlusions are easy to segment from figure and background.

Achieving similar results on more datasets world require finding a method (dimitation) challeng mg to Splito figure, backgrand, and orcluding pixels in the input