11.1 Nuclear plant

MATEJ SESTAK
950814-6454

One possible approach is look at the problem as a classification task, trying to select one of 4 classes (3 types of failure + Normal state) For this approach, we could use MLP and look at data at each timestep and evaluate. The problem with this approach is low number of training data for failures, as 99.9% off the time the state is normal.

Second approach could be a ~~[illegible]~~ anomaly detection. This approch would work well with training data without failures, as they are the anomaly to be detected. I would use LSTM network, as it keeps some memory and can detect anomalities. Also it can ~~[illegible]~~ evaluate data continuously. Problem ~~[illegible]~~ could be detecting specific failure, as the network would only produce that anomaly is present. ~~[illegible]~~

The general problem with this task is ~~[illegible]~~ unbalance ~~[illegible]~~ betwen normal state and failure, as failures occurs minimaly. Also, the risk of ~~[illegible]~~ not detecting a failure is ~~[illegible]~~ great, as we are talking about nuclear plant.

11.2 ZOOLOGISTS

MATEJ SESTAK
950814-6454

a) This problem is a clustering problem, as we want to group similar images together. I propose to use the self organizing maps (SOM) architecture, as it is design for clustering / visualisation. The input of the network would be the image pixels (200 × 300 = 60 000 values). and output would be a mapping into a 2D plane. The most important hyperparameter is number of nodes in the map, which should rughly correspond to the number of clusters we want - this is also the way how to control the level of grouping (low # units ⇒ few large groups, a lot of units ⇒ many smole groups).

Becouse the images are big (we have 60 000 input features), ~~fo~~ for better results and faster computational time, I propose to center the images and subsample them, to get images with size ~ 40 × 40 pixels with the insect being in the center.

b) To check if the new images with the same labels ~~land~~ end up in the same group, I would use the trained SOM and inputed the new images and check if they would be mapped into the same units. Again, the images needs to be the same size as the trained ones.

C) is on next page.

11.2 Zoologist

C) Now we have labeled data and [illegible] want to get the correct class. This is a classification problem and I would use a multi-layer perceptron (MLP).

The input would be the qualitative subset of the features (# of features = # inputs). The output ~~would be~~ size would be the number of taxonomy cathegories the ~~training~~ data has. The output would represent distribution over the classes. Key hyperparameters are: depth (# of hidden layers), width of hidden layers, learning rate, batch size, # epoch. I would use backpropagation as learning algorithm.

I would divide the data into 3 group (80% training, 10% validation, 10% test), used the training data for training the models, validation for model selection and test for evaluating the performance of the selected model. The performance on test set would be then reported.

11.3 TRUCKING

MATEJ SESTAK
950814-6454

a) The ~~s~~simplest approach is to look at this problem as classification, where input is the DRP image and output is binary (0 - no service, 1 - service needed). MLP can be used as the network. The training data would be mix of the breakdown DRP images and DRP images from normal use. The ratio should correspond to ratio of breakdowns / ~~non-br~~ normal.

Important is that data in each DRP image is from 5 following days, but the ~~days~~ DRPs do not have to be use as a time series during training (but it would make it easier to implement).
I would use backpropagation to train the network. Split the data into training (80%), validation (10%) and test (10%) sets, to use for training, model selection and evaluation, correspondingly.
This approach could also be used for estimating probability of needing service (just use sigmoid on output)

b) This is a ~~regress~~ regression problem, ~~apar~~ I would use ~~Using~~ similar network as in part ~~a, the~~ a. As the output activation, I would use ReLU to get a non-negative output (number of days to next service should not be negative).
Before training, we need to compute the days to nearest ~~the~~ service for all DRPs, so we can train the network. The input is the same as in ⓐ (5×234). I use backpropagation ~~again~~ to train it. Otherwise training is similar to part a.

MATEJ SESTAK
950814-6454

11.4 RNN

In TLFNN, we explicitly specify what previous data are still used in current step (what the network remembers). On the other hand, the ~~ara~~ RNN learns what ~~is~~ ~~[illegible]~~ ~~last outcome~~ ~~was~~ should keep in~~x~~ memory.

The weigh matrix for TLFNN computes the outcome given all inputs in the memory. ~~So all~~ ~~is is~~ ~~the connecting the~~ The layers are then connected by the sum of inputs.

RNNs weigh matrix computes outcome from claped previous outcome and current input. The layer is ~~the~~ only connected to the input layer and itself at previous timestep.

11.5 competition

MATEJ SESTAK
950814-6454

In this competition, your goal is to develop a model that can correctly label unseen data. Use the provided labeled datases for training and the provide us labels for the unlabel test data. The uderlying distribution of the test data could differ from the traing data, so your model needs to be robust to this noise and needs to generalise well.

Your model will be evaluated by classification accuracy on test data, e.g. $\left(\frac{\#\text{correct}}{\#\text{all}}\right)$. As this is a classification task, we are interested only in performance of the model.

To better understand the task, here are 3 possible application in the real world:

1, predicting weater from measurements; initial model is trained with data from sensors X, but it will be deploy with sensors X and Z, wich provide same features, but the values could be a bit different

2, image classification for images taken with different cameras (different resolutions, brightness, ...)

3, generly classification with noisy data.

Q11. 6 VAE 1

MATEJ SESTAK
950814-6454

The concept is sampling from latent distribution and use it as decoder input.

The key idea behind VAE is learning latent state distribution of the samples. The process of generating samples from trained VAE is as follows:

1) Sample from unit normal distribution : $\varepsilon = \mathcal{N}(0,1)$

2) Computed the latent variable from trained mean and std:

$$z = \mu + \sigma \odot \varepsilon$$

3) use the latent variable as an input to the decoder network

4) propagate through decoder, the output is the generated sample

Classical AE learns represantation in a different feature space (higher or lower dimensional than input) but does not learn the underlying distribution of the data. On the other hand, VAE learns the latent distribution of the data, which allows it to generate from it. It also prevents from just rememboring the input samples, as decoder uses sampling from the latent distribution