DD2437 Online Exam Answers, 23VT (2023/03/14)

Question 1 (11p)

A) The diagnostic system for distinguishing good from flawed dampers can be built as a neural network classifier with a single (sigmoidal) or two outputs (recommended also for later extensions). The key challenge is to combine image data with spectral vectors. Although image sizes are relatively small it is recommended to use convolutional neural networks (CNNs) to learn representations if sufficient amount of data is available. After training CNNs (using dichotomous labels: good vs flawed) the top representational layer, i.e. the last convolutional+pooling layer, can be concatenated directly with a corresponding 1024-long spectral vector, and then on top of that two or three fully connected layers can be added and trained back-to-back with backprop for joint binary classification. Alternatively, there could be a separate training process for 1024 spectral vectors to learn representations that are concatenated with the CNN representations of image data, as mentioned above. The representations of 1024-sized spectral data could be learnt using an autoencoder (AE) or deep belief nets (DBN, RBM) or other architectures suitable for learning representations (either in a supervised way as CNNs above or without labels – autoencoders).
For any network development, validation and evaluation, there should be a clear split of data into training/validation (over which cross-validation scheme could be used for model selection) and unseen test (for the final evaluation, conclusive comparisons but not for any evaluation aimed at model selection) data sets.
To test whether the classification based on spectral representations would match that of the combined spectral and image representations, one should compare the output of the CNN network for images with the output of a classification network developed for spectral data, which could be a multi-layer perceptron (MLP) built either on the learnt representations (e.g. DBN, AE) or taking raw 1024-sized vectors directly as input. Computational complexity of the two networks should be taken into account in the comparison but ultimately the size of fully connected layers on top of different representations should be subject to separate model selection processes, independent for each input source configuration. To examine the suitability of the diagnostic classifier to generalise across different car types, an analogous comparative study should be performed – average of car type specific classifiers vs a single classifier trained on data originating from all car types. All these comparative analyses should be performed independently at each site and then the results should be aggregated across sites. The two questions can also be studied independently though it would be more attractive to study interaction effects to see if the difference between car-type-specific and car-type-generic classifiers depends on the site.

B) Here the only comparison that should be made is between site specific classifiers developed in A and an analogous classifier trained on data from all sites. It is recommended that cross-validation across sites (with a test fold corresponding to a specific site) is performed in the latter case and the obtained results are directly compared with corresponding site-specific classifiers, e.g. if site X constitutes a test fold for one out of n fold configurations in n-fold cross-validation, the X-site performance of that site-generic classifier developed/cross-validated on the remaining n-1 fold data should be compared to the performance of X-site-specific classifier on the matching portion of samples from that site. This way a set of site-dependent comparisons can be made, i.e. one per site, which could be statistically aggregated in a suitable null hypothesis testing procedure. The null hypothesis would be that average site-specific and site-generic results are the same.

C) If only flawed dampers are concerned, no other labels exist and the problem boils down to unsupervised clustering. This can be performed with the use of self-organising maps (SOMs), which also allow for 3D visualisation when the 3D output grid is used. The size of the input layer should correspond to the dimensionality of the learnt data representations. The Euclidean distance metric or cosine distance metric could be used to measure the similarity and decide the winning (best matching) node in the grid. Following the process of SOM learning with successively shrinking neighbourhood size in the output grid, it could be observed how topographically are distributed grid nodes in the 3D space. If they form distinct groups than the clustering can be easily performed by matching those distinct grid groupings with clusters. Then the input samples that activate the best matching unit in one of the grid unit clusters would be assigned the corresponding cluster membership.

To assess the similarity and difference between clusters one can just measure the distance between cluster units in the grid space (average pair-wise distance between grid units in two clusters).

Similarly, to test how the samples corresponding to good dampers relate to the clusters of flawed damper samples, one can just study the activation pattern of the grid units in response to those "good samples". It is likely that units in one of the "flawed" clusters will activate more often (as best matching units) than those in other clusters. *Alternatively*, one could re-train SOM on all samples - flawed and good dampers, and expect that the good dampers sample will belong to one distinct cluster in the SOM grid space. Measuring pair-wise distances (in the SOM grid space) between that "good damper" cluster and the other "flawed damper" clusters would provide an answer as to which group of flawed dampers is most similar to good dampers.

Testing the compatibility of groupings between different sites and car types could be performed analogously: i) either by developing site/car type – specific SOMs and studying the activation patterns for samples belonging to a different site/car type or ii) by co-developing a single SOM for all the data and studying whether samples corresponding to the same site/car type tend to activate units in the same grid unit cluster (whether the winners/best matching unit are in the same cluster).

D) The supervised feed-forward neural network developed, optimised and selected in p. A could be extended by adding extra output nodes with sigmoidal activation functions to account for extra classes describing different types of technical flaws (classes derived in p. C from the analysis of distinct sample groups corresponding to flawed dampers). With re-labelled data (or with enhanced labelling) the training, validation and evaluation process should be repeated. One of many challenges is whether there is a sufficient amount of data per new classes and whether these new classes are relatively equally represented.

Question 2 (2p)

A) First iteration

first element: -1*0 + 1*1 + 1*1 + -1*0 + -1*-1 + 1*0= 3, thresholded at 0 gives 1, so first element, out = 1
second element, out = 1
third element, out = 1
fourth element, out = -1
fifth element, out = -1
sixth element, out = 1
thus p = 1 1 1 -1 -1 1

Second iteration
first element, out = 1
second element, out = 1
third element, out = 1
fourth element, out = -1
fifth element, out = -1
sixth element, out = 1
thus p = 1 1 1 -1 -1 1 which is the same as in iteration 1, so we have reached stability.

Answer: p=[1 1 1 - 1 -1 1]

The correct pattern is [1 1 1 -1 -1 1] (the first number is flipped) and we get this already after one round of updates. However, we need to do one more update to show we get the same output one more time so as to show it is a stable pattern. Also, since we ask for an iterative update, we do need to see you are doing iterative recall, so a batch solution does not give points.

B) The output of a node is dictated by a sigmoidal transfer function of the weighted sum of the input, however this defines the probability of output being 1 and with (1-that probability) of output being -1. The flipping of nodes makes the nodes produce a Boltzmann distribution. It enables the network states to cover the probability distribution of the data.

Question 3 (10p)

A) The presented problem can be categorised as a system identification.

As there is no underlying physical model of the phenomena or the model due to its complexity requires excessive compute power, a neural network could be used as "black box" to translate descriptors into the state of the system. This mapping can be learnt from training data. If the detailed model is available it could be run in representative scenarios to provide training data (concern about the computational cost is still valid but if the model is run only for generating training data the cost is bearable).

For system identification, a simple feedforward network such as a multi-layer perceptron is sufficient. The number of inputs should correspond to the number of descriptors and the number of outputs – to the number of system state variables depending on the state encoding scheme. The number of hidden layers and their dimensionality depends on the problem complexity and should be found based on a systematic search for these metaparameters with the validation performance used as a key criterion.

The network can be trained using backprop.

B) The problem of failure detection (not prediction) can be conceptualised as a classification problem with the classes corresponding to normal and abnormal states (various failure states). If the number of failure occurrences is very low, the problem can also be seen as anomaly detection. The bottom line is that a set of specific input (sensor readouts) patterns associated with the resulting failure has to be detected.

Assuming that we have training data available (sensor readouts) with the corresponding truck diagnoses (failure or normal condition), and that we first investigate static multivariate patterns (not their sequences), a supervised learning approach (e.g. gradient descent/backprop) can be used to train a multi-layer perceptron to recognise input patterns correlated with the system failure.

The number of network inputs would correspond to the number of sensors and the output layer configuration would depend on the system state encoding (a classical configuration for classification would imply the number of outputs with sigmoidal activation function equal to the number of classes but there are also other encoding strategies). The hidden layers should be configured as a result of metaparameter tuning, proposed above in a).

The first hypothesis about the relevance of multivariate sensor readout patterns to the performance of failure identification could be tested by comparing the quality of detection obtained with the proposed nonlinear multivariate regression approach relying on the multi-layer perceptron network with that of simple nonlinear but univariate regression models, as originally tested in the task. In general, the relevance of specific input variable (sensor) combinations can be examined by comparing networks (in terms of validation performance in failure detection) with different input configurations or by means of sensitivity analysis of the network error wrt. specific inputs.

The second hypothesis as to whether sequences of input patterns are more meaningful rather than static patterns can be tested similarly as the first hypothesis. In particular, a comparative analysis of different networks (to be precise, networks configured to account for various input configurations) with the (validation/test) detection performance as a comparative criterion would cast light on the hypothesis. In the given example, multi-layer perceptrons with inputs encoding time-delayed representations of sensor readouts (sequences) could be studied comparatively with networks that make predictions based only on current values of these readouts (static). One could employ here a recurrent neural network for sequences, though a comparison with a multi-layer perceptron receiving different inputs (i.e. only current input states), as proposed in the beginning of this solution, would not necessarily be fair. It is important that all the networks, i.e. with different input configurations, are optimised with the use of training and validation data (cross-validation for model selection is one of available options), and that for the conclusive outcome the comparison is made based on the testing performance (networks evaluated on unseen data). Depending on the amount of available data and given the potential high dimensionality of the network input to exploit the time-delayed representations, there is a risk of overfitting that may render the comparison inconclusive.

Question 4 (3p)

When we back-propagate the error over multiple time points using BPTT, every time step presents a similar problem as we see in a standard multi-layer perceptron with multiple hidden layers. For each time-step, gradients often get smaller.

The consequences when we fix this by imposing a restricted learning-window (restricted number of time steps over which errors are propagated) is that we also restrict the temporal span over which the network can make associations between parts of the input.

Question 5 (10p)

Alternative using a Hopfield network.

This is an optimization problem. (This is a traveling salesperson problem.) I will use a Hopfield network motivated by the need to do optimization. Input to the network is a binary vector of length 134 where +1 codes for a hotel with a guest and -1 codes for a hotel with no guest. Output is a binary vector of length 134 with the coding given above. I will use a single Hopfield layer where nodes have output {-1, 1} using the sign function as transfer function. The network has N^2 nodes, where N is the number of hotels to visit. Training is not done. Instead the rules of the game are implemented as large weights and distances as small weights, all made so that the energy function to be minimized leads to the solution of the problem. Distances here refer to the travel time distance

between two hotels. With regard to generalization, it does not really apply to this case, I will need to make a new instantiation of the network for every travel given the specific hotels to visit (but the connectivity rules are the same and no training is needed so this is not a computational problem). When it comes to hyperparameters, one hyperparameter would be the relative magnitude of weight penalty for violating the rules relative to the magnitude of distance. Challenges includes constructing the rules dictating the energy penalty for the hard rules of the TSP, the smaller penalty for distance is probably not tricky to set.

Alternative using a SOM network.

This is an optimization problem. (This is a traveling salesperson problem.) I will use a SOM-network given that this is a TSP-like problem (see lab 2). Input to the network is the position of each of the hotels, output is the activities of the RBFs and we plot the result in the input space showing lines between neighbouring cities. Regarding architecture details, for the number of RBFs, I will use the number of hotels of the present tour. Furthermore, the neighbourhood connectivity is 1-dimensional and has wrap-around. Training is done using the winner-takes-all rule to first select the winner and then use the Delta rule to modify RBF parameters of the winner and all of its neighbours. I will need to experiment with initial size of the neighbourhood and learning rate and also decrement of these per iteration. Challenges includes how to map the travel time difference between two cities to the input. It is also challenging to quantify the robustness and generalization capabilities of SOM networks.

Question 6 (8p)

It is a classification problem that can be addressed by a shallow multi-layer perceptron (MLP) or radial-basis function (RBF) network to lower the risk of overfitting.

The risk is high since the number of samples is low in comparison to the number of attributes. Since the top triage priority is of key importance, one can pay less attention to the precision for low priority triage cases. Consequently, in the attempt to lower the number of attributes (as mentioned, it is high relative the number of samples) one could either skip 7 questions when representing patients' questionnaire responses or just represent them as a binary attribute (the question has been given or not) as it is hard to assume that they contain more insightful discriminative information. A similar encoding approach one can take to the other 9 questions, answered only by some of the high-priority patients. In that case, because of that high-priority status removing these questions is not recommended. Overall, the encoding scheme could be based either on attributing integer values to each categorical variable (multiple-choice question) or on holistic binary encoding altogether, i.e. for multiple-choice questions one could place 1 for the chosen alternative in the string of bits, e.g. 00010 to encode that the fourth out five answer options was chosen. For yes/no questions binary encoding is suited in either case.

The key parameter of MLP or RBF is the size of the hidden layer. For RBFs it is also the width defining response properties of RBF units. The output activation function could be sigmoid and the number of outputs would correspond to the number of triage classes. The hyperparameters should be chosen based on the estimate of the generalisation error using cross-validation. So, 300 out of 400 data points could be chosen for training /validation with 5-folf cross-validation and the remaining 100 could be used as unseen test for the final testing. It is important to stratify the folds and training/validation and test subsets. As mentioned, the risk of overfitting is high so the learning (backprop for MLP and least squares for the last RBF layer) should include regularisation and the aforementioned reduction of data dimensionality is also acceptable especially if it affects mostly low-priority triage patients. One could also think of reducing the number of classes by grouping low-priority groups – the resolution of classification output should be optimised with high-priority triage category in mind. To further support good performance for high-priority triage categories one could modify loss function to better reflect this. Also, the network's output should be ultimately quantified with specificity/sensitivity or ROC measures with focus on high-priority triage samples. Depending on the doctor's preferences the operating point (between highly sensitive and highly specific classification outcome) for the network should be chosen. The prediction themselves could be communicated to the doctors as normalised values of the output sigmoidal layer, i.e. out_i = exp sigmoutput_i / sum(exp sigmoutput_j) (j=1...number of outputs).

Overall, the challenges are concerned with limited amount of data (especially for high-priority triage category), skewed class distribution, rather high dimensionality of samples, missing values, which all pose a risk for the network to overfit, provide a skewed response and potentially underperform for the samples we should care most about – the top-priority triage category. Hence, the recommended steps mentioned above should be taken and evaluated in extensive testing that includes doctors' feedback.

Finally, if there were more data to be obtained the most important would be to collect it from high-priority triage patients. As for the new questions one would need to balance the advantage of more information with the increased data dimensionality, which without being compensated for by the higher number of samples (patients) could further undermine the generalisation power of the network. Obviously, more questions imply a larger input layer to begin with.