Question 1

a. The learning rule adapted for memory is called Hebbian learning. Neurons that fire together wire together. In other words, the rule modifies weights based only on the interaction/correlated activity of two connected neurons. The rule is local and allows for auto-associative memory. To encode a memory, simply input it as a vector containing [-1, 1] and the network will adjust its weights according to ‘remember it’. It will converge to a ‘stable’ point’, a local minimum in the energy landscape.

b. The representation of the sensory input should be a vector with values [-1,1] and the input value should be multiplied by the network's weight matrix until it converges to a stable point. Make sure that the vectors are orthogonal to each other

c. Capacity is determined by network size and how correlated the data input is to each other. If the input data to be remembered, is orthogonal vectors then the learning is maximized. To test the storage capacity in simulations, input ‘dummy vectors’ and check if all previously learned patterns can be retrieved. Continue this until catastrophic forgetting happens.

d. Yes, it is partially dependent on the total amount of nodes the network has such that it has remembered the input. Depends also on the noise, if too noisy then it won’t be able to. Has to do with energy landscape, inputting a partial sensory input will make the network converge to a stable attractor which contains the memory pattern

Question 2

The temperature in a Boltzmann machine is part of the energy function, specifically the calculator of the softmax probabilities during the sampling process. Used to control exploration-exploitation trade-offs during the running of the network. It controls if the output should be 1 or -1. A higher temperature allows for more exploration while a lower one makes the model more deterministic however could make it stuck in local minima.

Question 3

The type of problem is an unsupervised clustering problem where the amount of classes is not known. The network for this type of problem is SOM network where the input is all the features and the output is preferably a 2 or 3D used for visualization and finding groupings. The neighbouring size, neighbouring decay and learning rate are parameters up for model selection. The model learns through competitive leaky learning where the “neighbours” of the winning node get clustered together. The input is all the features, preferably with numerical values, and how “close” each grouping is calculated with cosine distance, taxi-cab distance or Euclidean distance. The output is a topological mapping of the input data. To optimize and estimate generalization, look at the output space and change it up.

Question 4

a. This can be done with Transfer learning. You train the CNN on the available pictures and use data augmentation techniques such that it learns these images efficiently

b. This seems to be both a generative and discrimination task. The first task is to label them, discriminative, however in an unsupervised manner. This can be done with stacked RBMs or DBN that learn the underlying representation without labels, afterwards add labels check reconstruction loss and minimise it as much as possible. Then build and train the CNN on both the labelled data and the data that was labelled by the DBN/Stacked-RBM network

c. This seems to be a generalization task of CNN. This can be done with data augmentation techniques, simply input the images however flipp them, crop them, change lighting etcetera on the input data. Computationally expensive but effective way.

Question 5

ANN could replace the preliminary stage and generate a 0-5 risk score based on the available labelled images.

The network would be a CNN variant model. The input would be all the images concatenated into one and then inputted to the CNN model. The output with a softmax function that maps values 1-5. The architecture, that is amount of convs + pooling layers and if residuals are needed, are up for model selection. The loss function would be a cross-entropy loss. Training is done with back-propagation. To avoid overfitting given the sparse data, data augmentation techniques and L2 regularisation terms are added to the cost function.

The datasplit would be 800-100 where 800 is training and 100 testing. The model selection is done with cross-validation of the dataset, the one with the highest score on the test set while not too different from the training score wins. This is a classification problem task.

Now to answer the question:

a. I would show the results with the x-graph showing patient id and Y with the predictions. Then show my prediction as some kind of function and compare it with the medical staffs prediction, afterwards show the correlations. The higher correlation, the better.

b. Postprocess the first model in (a) and change the output to cross sigmoid and cross-entropy loss. Either that or preprocess the the input data, convert them into the new data then train the model on it. Afterwards evaluate with cross-validation but also Area under curve, F1, precision and recall scores.

c. to build this model, use RNN or LSTM with CNN representations. The input are the images and timestamps while the outputs are the labels 0,1 if it is low or high risk. Trained with back-prop through time-steps. Compare the model from b and a and see if it made a difference.

Question 6 - redo

a. This seems to be a time series forecasting. Assuming that the data is multivariate then a RNN or LSTM network is suggested. How many nodes or how deep the LSTM goes is up for model selection, model selection is done with cross-validation. Loss function would be a MSE, testing would be training on 2-15 years then prediction 2 months ahead then checking