& train

Q3] How would you define a Deep neural n/w?

Ans ⇒ A deep neural network is a neural network with a certain level of complexity. OR A neural n/w with more than 2 layers

⇒ Deep neural n/w uses sophisticated mathematical modeling to process to process data in complex ways.

⇒ A neural n/w, is a technology built to simulate the activity of the human brain.- specifically, pattern recognition & the passage of inputs through various layers of simulated neural connections.

⇒ Each layer in the NN performs specific type of sorting & ordering in a process that is refered as "feature hierarchy".

⇒ They are harder to train because they have hyperparameters i.e learning rate, momentum, different types of regularization and the layer of the n/w can vary in type, number & width.

⇒ Also as the network gets deeper & deeper the ~~gad~~ gradient vanishes. Example:- When n hidden layers uses an activation function like the sigmoid function, n small derivates are multiplied together. Thus the decrease in gradient is exponential as we propagate down the intial layers.

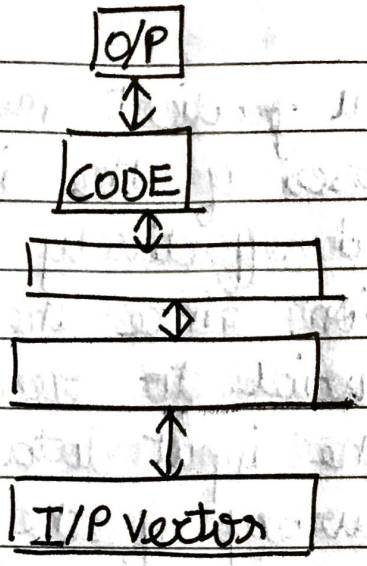
⇒ Since a small gradient means that the weights & biases of the intial layers will not be updated effectively whith each training session. Since these intial layers are often crucial to recognizing the core elements of the input data, it can lead to overall inaccuracy of the whole network.

Q] List and describe some moredern techniques that may be used to train a deep neural network.

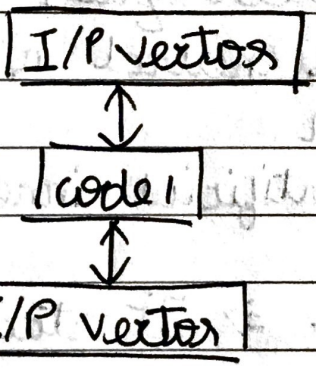
Ans ⇒ The two techniques that may be used to train a deep neural n/w is
* Pre-train
* Use of artificial inner targets

Pre-training :- ⇒ It stacks deep n/w layer by layer.
⇒ Thus making sure that the first hidden layer is capable of representing meaning-full input before adding a second layer to it.
⇒ Then later making sure that the second layer represents the first layer meaningful before adding a third layer. The process continous till n layers
⇒ This can also be achieved via auto-association.

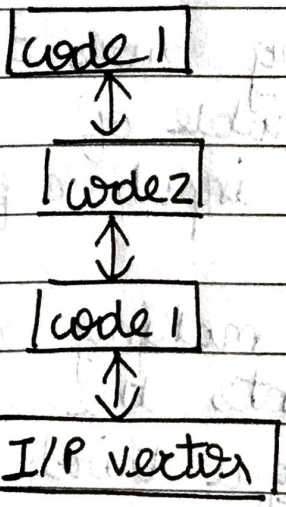
①


o/p
CODE
I/P vectors

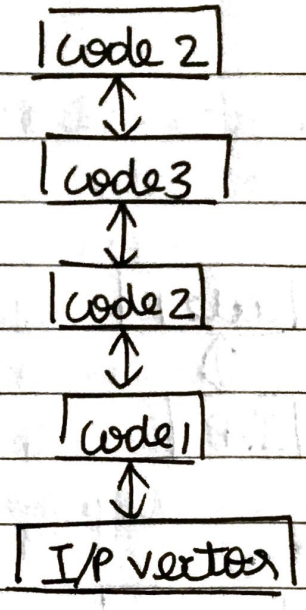
②


I/P vectors
code 1
I/P vectors

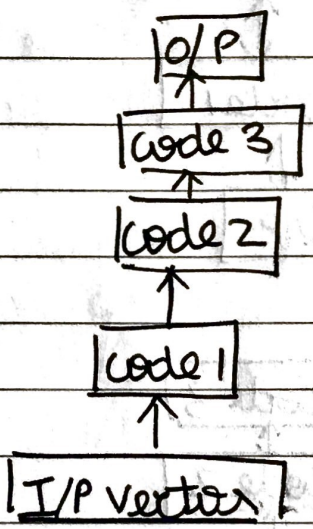
③


code 1
code 2
code 1
I/P vectors

④



⑤



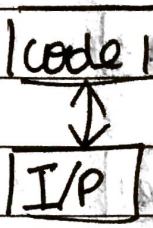
⇒ We can also use stochastic gradient descent, because it models weights are updated after each iteration using the backpropogation of errors. (In boltzman)

⇒ The idea is training the BM for so that we can place local minima where we want them

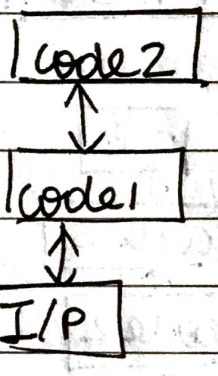
⇒ Alternative pretraining for RBM ⇒

* It follows a deep belief network.

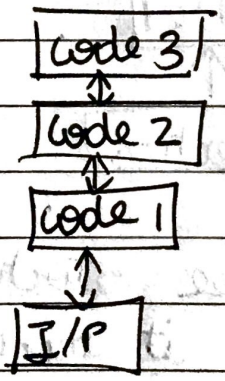
①



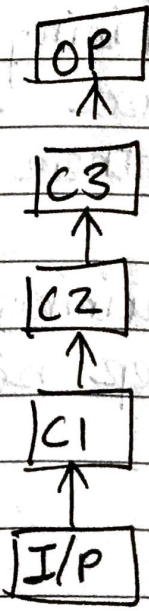
②



③

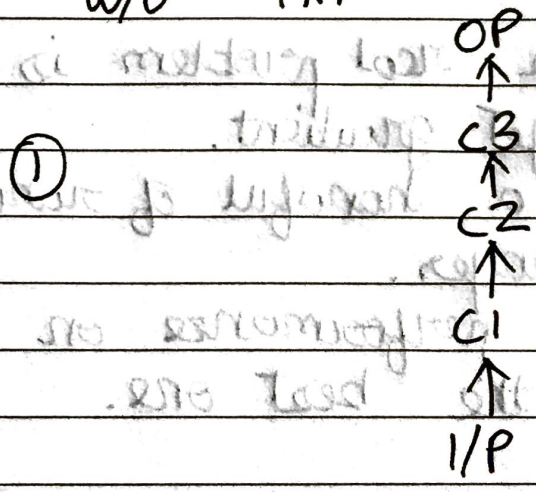


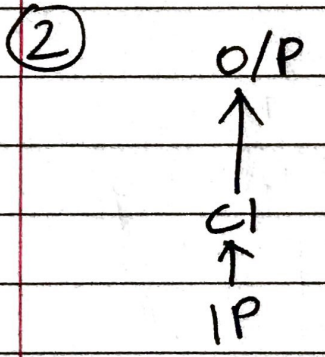
④

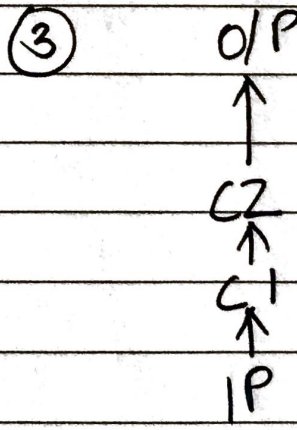


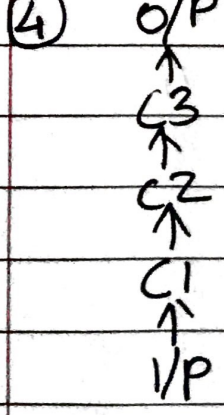
⇒ Retraining by auto-association :- * It can be a advantage as we can use unlabelled data. But go wrong if we aren't considering at all the property we want to predict. We compress regardless of the property. If its lossy, the loss can be in the wrong place.

W/O AA









⇒ About pretraining without auto-association :-

(Big) Disadvantage :- ⇒ inability to use unlabelled data. Hence we need to target at all stages. More training ↑.

(Big) advantage :- ⇒ You compress based on the property you are trying to predict. Hence. If its lossy, the loss is in right place.

⇒ Artificial target :- The real problem is that inner layers don't get gradient.

* So we generate a handful of random hard targets for the layer.
* We observe their performance on example & select the best one.