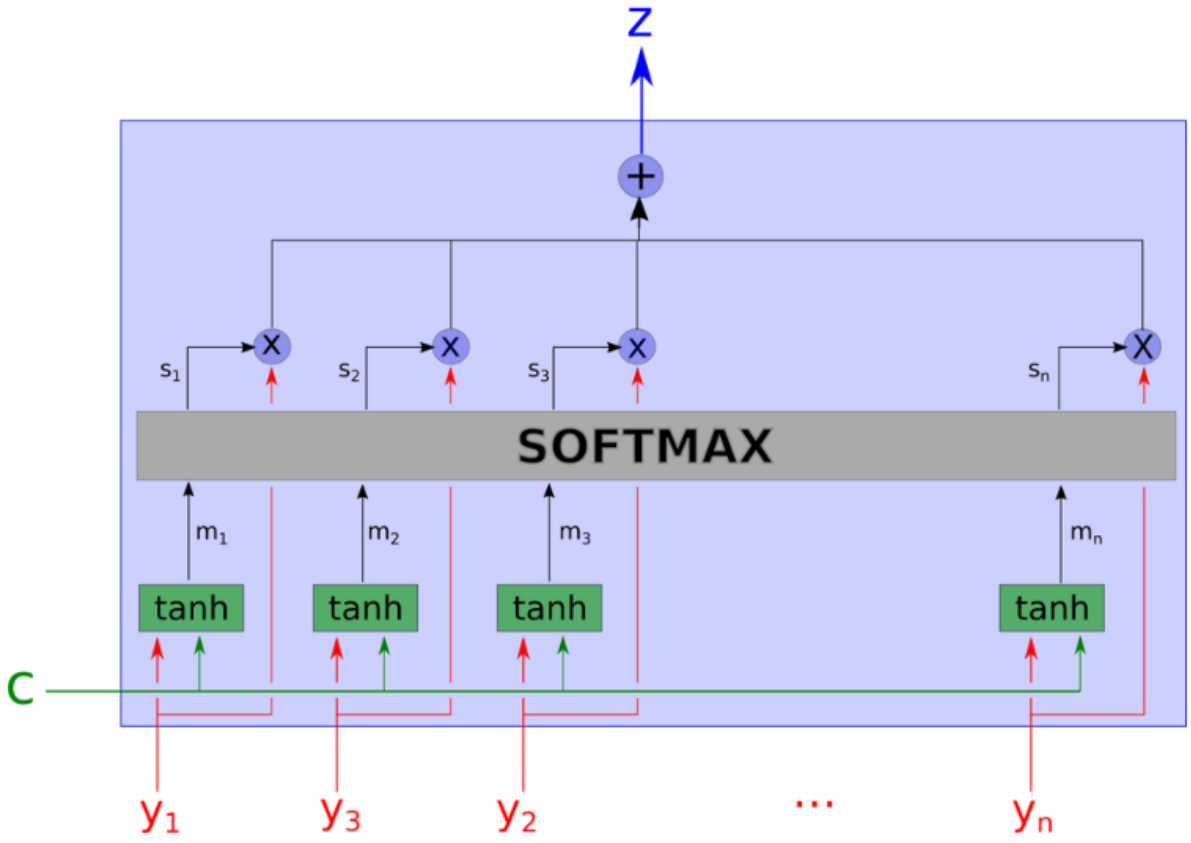
Amir Masoud Shaker

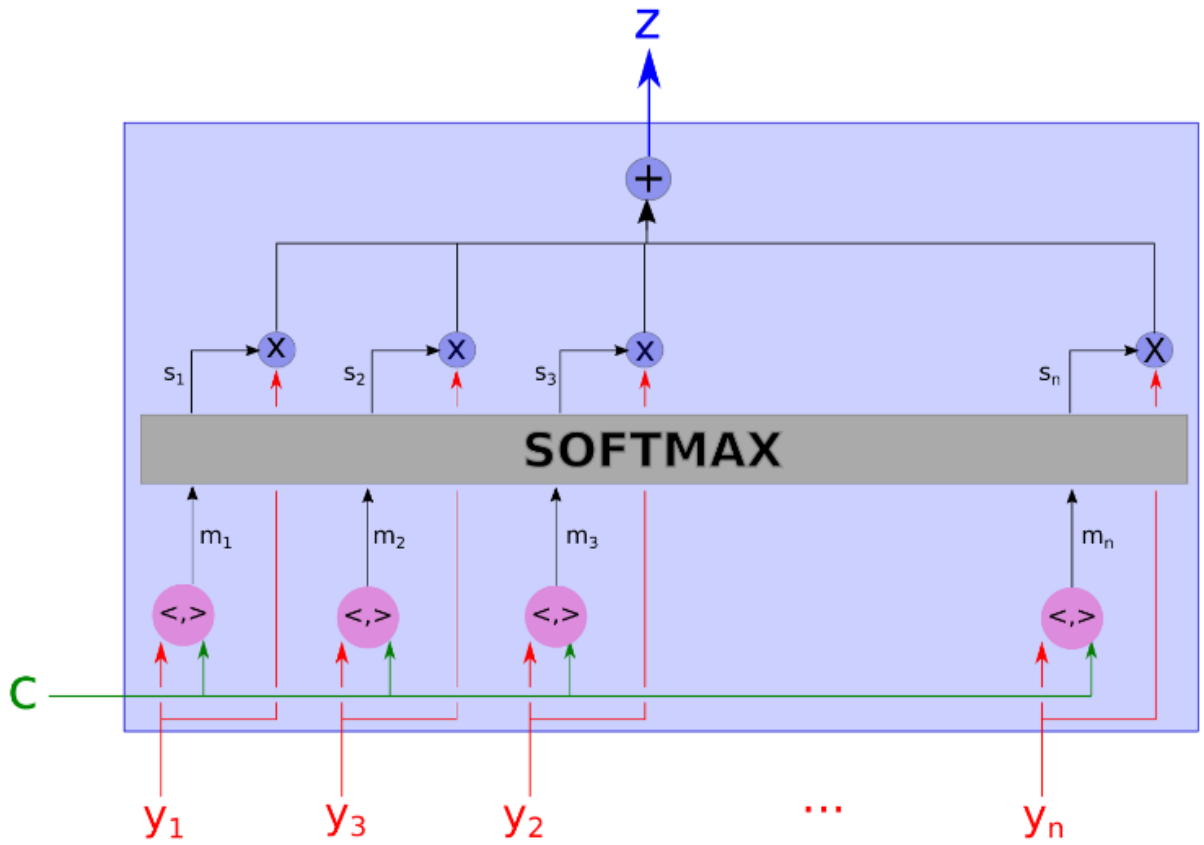
97243081

Part 1: Questions

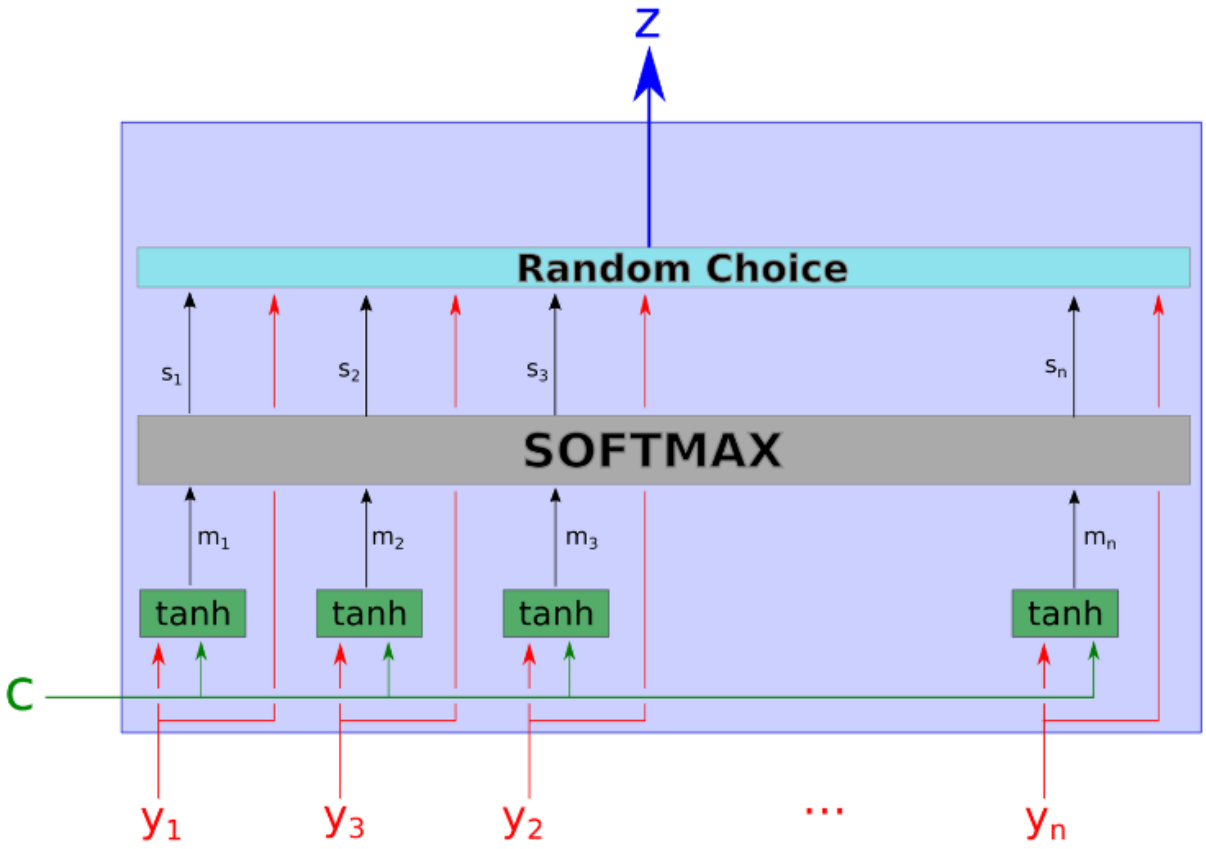
1. Explain the difference between soft attention and hard attention.

Soft attention is a fully differentiable deterministic mechanism that can be plugged into an existing system, and the gradients are propagated through the attention mechanism at the same time they are propagated through the rest of the network. Figures below show two kinds of Attention model that use soft attention mechanism:





Hard attention is a stochastic process: instead of using all the hidden states as an input for the decoding, the system samples a hidden state yi with the probabilities si. In order to propagate a gradient through this process, we estimate the gradient by Monte Carlo sampling. Figure below shows a Hard Attention model:



Here are the formulas:

Soft Attention	*Hard Attention*
Attention score is used as weights in the weighted average context vector calculation. This is a differentiable function.	*Attention score is used as the probability of the i-th location getting selected. We could use a simple argmax to make the selection, but it is not differentiable and so complex techniques are employed.*
$\mathbb{E}_{p(s_t \mid a)}[\hat{\mathbf{z}}_t] = \sum_{i=1}^{L} \alpha_{t,i}\mathbf{a}_i$	$\hat{\mathbf{z}}_t = \sum_i s_{t,i}\mathbf{a}_i$ $p(s_{t,i} = 1 \mid s_{j<t}, \mathbf{a}) = \alpha_{t,i}$ $\tilde{s_t^n} \sim \text{Multinoulli}_L(\{\alpha_i^n\})$

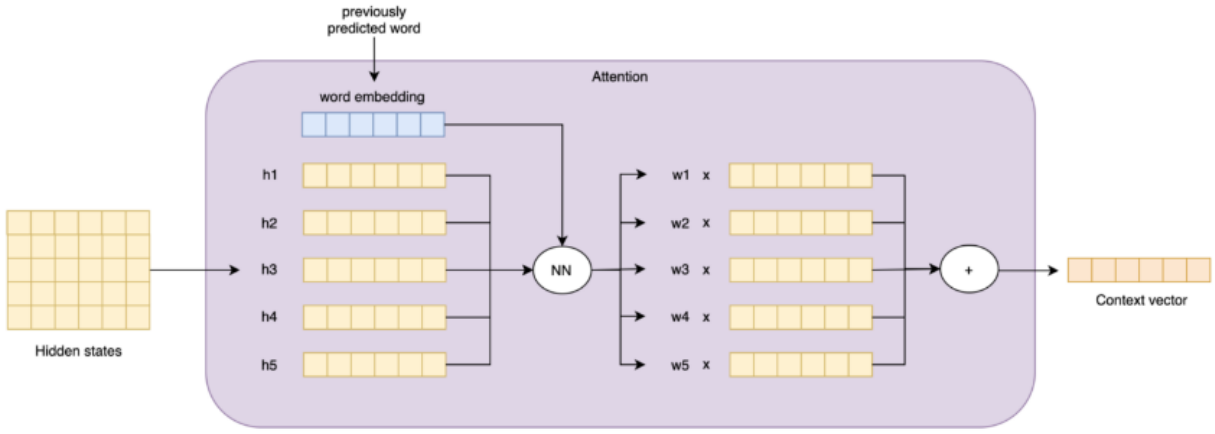
"a" represents encoder/input hidden states, "α" represents the attention scores, "$s_{t,i}$" is a one-hot variable with "1" if "i-th" location is to be selected.

2. Explain the difference between global attention and local attention completely and by drawing a figure.

Global Attention is an Attention mechanism that considers all the hidden states in creating the context vector.

It does so by performing a weighted sum, where each specific weight is computed by a feedforward NN taking into account its specific hidden state and what word is the model translating at the moment.

Figure below shows how Global Attention works:

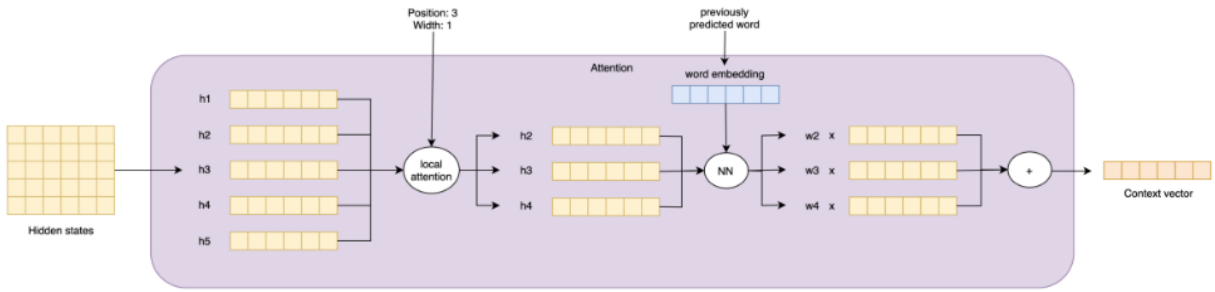


When Global Attention is applied, a lot of computation occurs. This is because all the hidden states must be taken into consideration, concatenated into a matrix, and processed by a NN to compute their weights

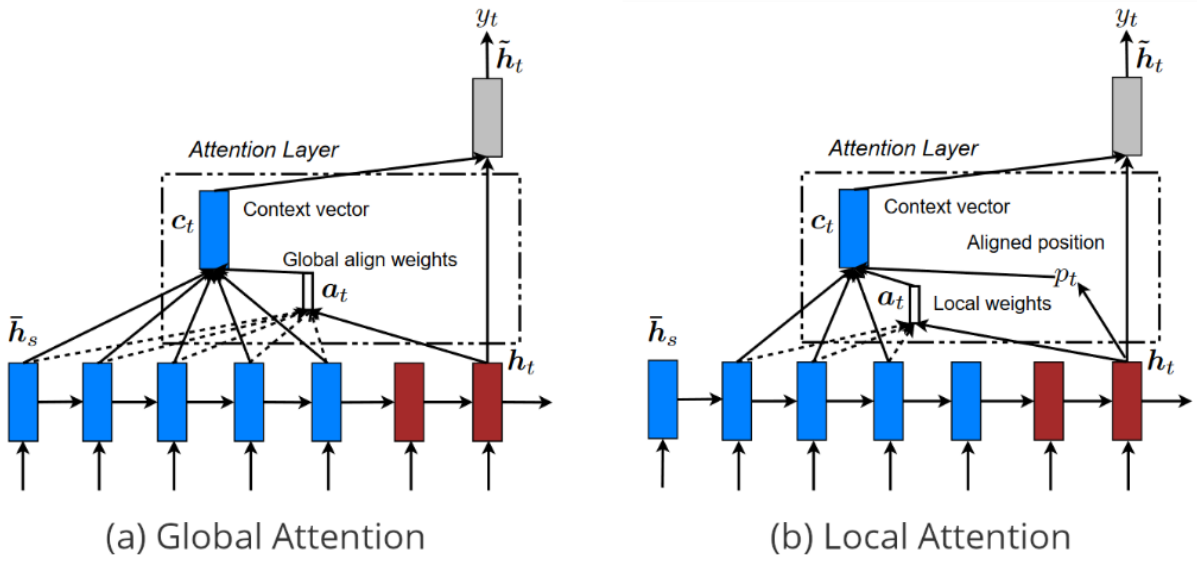
We can reduce the number of computations without sacrificing quality by Local Attention.

Local Attention is an Attention mechanism that considers only a subset of all the hidden states in creating the context vector. The subset can be obtained in many different ways, such as with Monotonic Alignment and Predictive Alignment.

Figure below shows how Local Attention with Monotonic Alignment works:



Lastly, we can see the figures of the two mechanisms together:

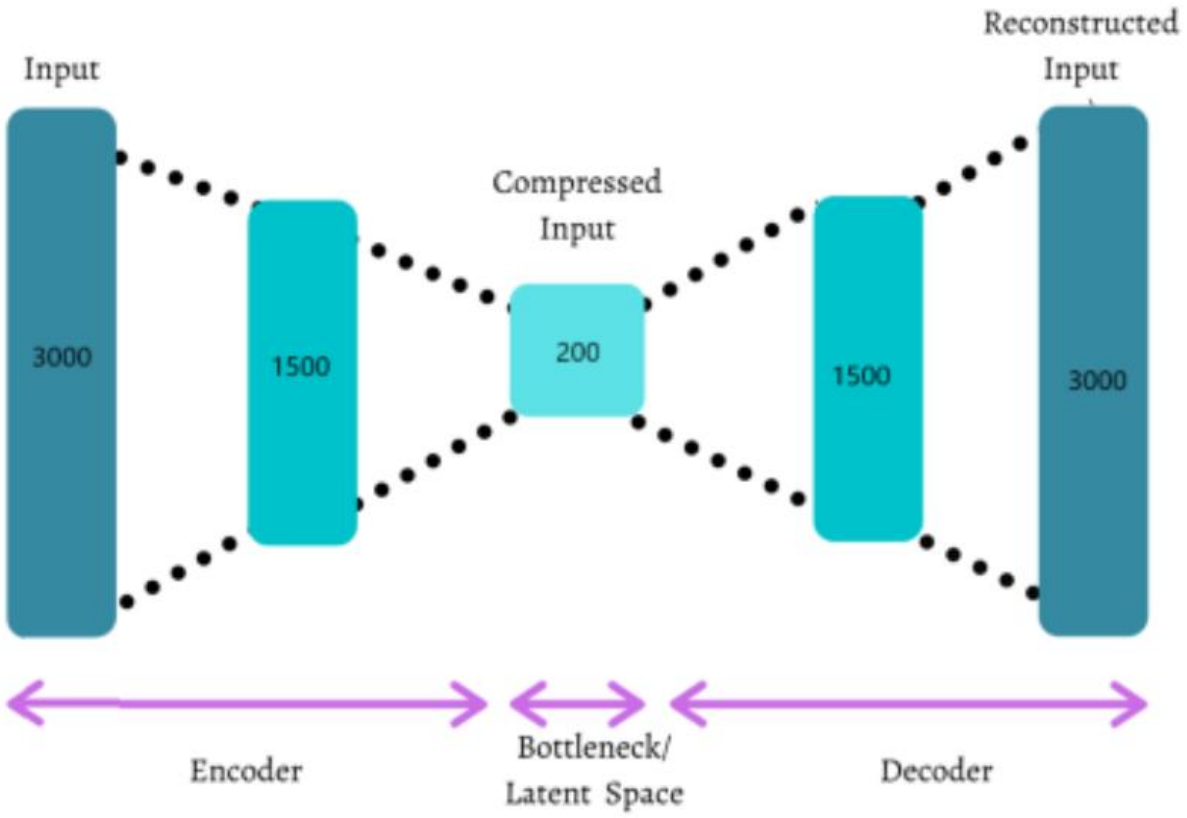


3. Explain why autoencoder is a self-supervised method?

Because it generates its own labels from the training data and learns a representation of the input data.

4. Is autoencoder a dimensionality reduction method? Why?

Yes. Autoencoders train the network to explain the natural structure in the data into efficient lower-dimensional representation. It does this by using decoding and encoding strategy to minimize the reconstruction error.



5. In what ways are the architecture of encoder and decoder in an autoencoder similar? What indicators or features are required to be the same in these two parts?

Both the encoder and decoder are fully-connected feedforward neural networks.

The decoder's output network is a mirror image of the input encoder in a more detailed manner.

The encoder and decoder have the same dimensional values.

The Number of nodes in the autoencoder should be the same in both encoder and decoder. The layer of decoder and encoder must be symmetric.

6. What tasks are autoencoder models used for? Explain briefly (at least two cases).

Dimensionality Reduction, Outlier Detection, Denoising, Feature Extraction, Image Coloring, Watermark removal from Images.

Explanation:

- **Denoising** autoencoder represents a modification of the basic autoencoder, where the input is partially corrupted by adding noise to it. To repair the partially destroyed input, the denoising autoencoder has to discover and capture relationships between dimensions of input in order to infer missing pieces. The noise is controlled by a stochastic mapping.

- Autoencoders can be used as a **feature extractor** for classification or regression tasks. Autoencoders take un-labeled data and learn efficient codings about the structure of the data that can be used for supervised learning tasks.
  After training an autoencoder network using a sample of training data, we can ignore the decoder part of the autoencoder, and only use the encoder to convert raw input data of higher dimension to a lower dimension encoded space. This lower dimension of data can be used as a feature for supervised tasks.

Part 2: Implementation

Colab link:
https://colab.research.google.com/drive/1xT0pTdEDIoBs1dmsbfX8hMlHFNRQxtXX?usp=sharing

Explanation: I've provided comments in my code.

Also, all the figures are provided in the notebook.

References:

Ref1, Ref2, Ref3, Ref4, Ref5, Ref6, Ref7, Ref8, Ref9, Ref10