UE21CS343BB2

Topics in Deep Learning

Dr. Shylaja S S
Director of Cloud Computing & Big Data (CCBD), Centre for Data Sciences & Applied Machine Learning (CDSAML)
Department of Computer Science and Engineering
shylaja.sharath@pes.edu

Ack: Anashua Dattidar, Teaching Assistant

What are Seq2Seq tasks?

Tasks in which the input and output are both sequences such are known as Seq2Seq tasks. The most intuitive of them all is machine translation as it is widely used and also will be used to explain all concepts related to the encoder decoder architecture in this lecture.

The main challenge while dealing with Seq2Seq data is :

1. The input and output sequences are of variable length
2. The length of the input sequence maynot be the same as that of the output sequence.

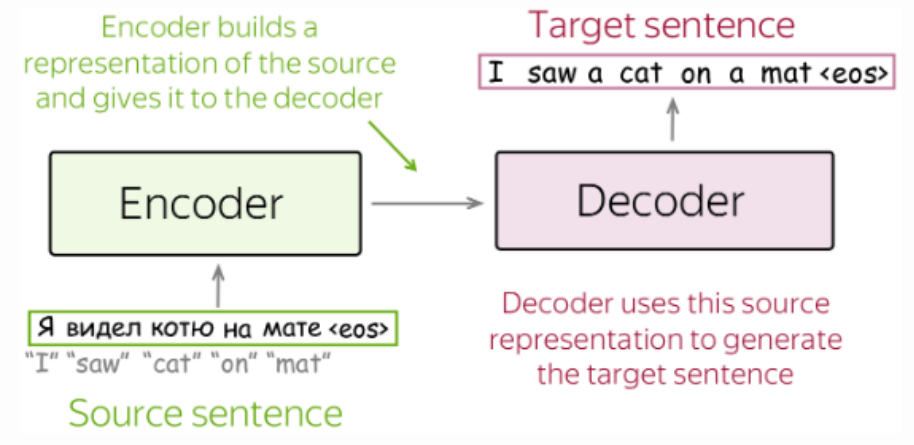
One of the preliminary approaches to dealing with such data was introduced by Ilya Sutskever and the Google research team in their 2014 paper “Sequence to Sequence Learning with Neural Networks”[1]. This came to be known as the Encoder Decoder model.

Encoder Decoder Model

Encoder-decoder is the standard modeling paradigm for sequence-to-sequence tasks. This framework consists of two components:

- encoder - reads source sequence and produces its representation(also known as the context vector);
- decoder - uses source representation from the encoder to generate the target sequence.

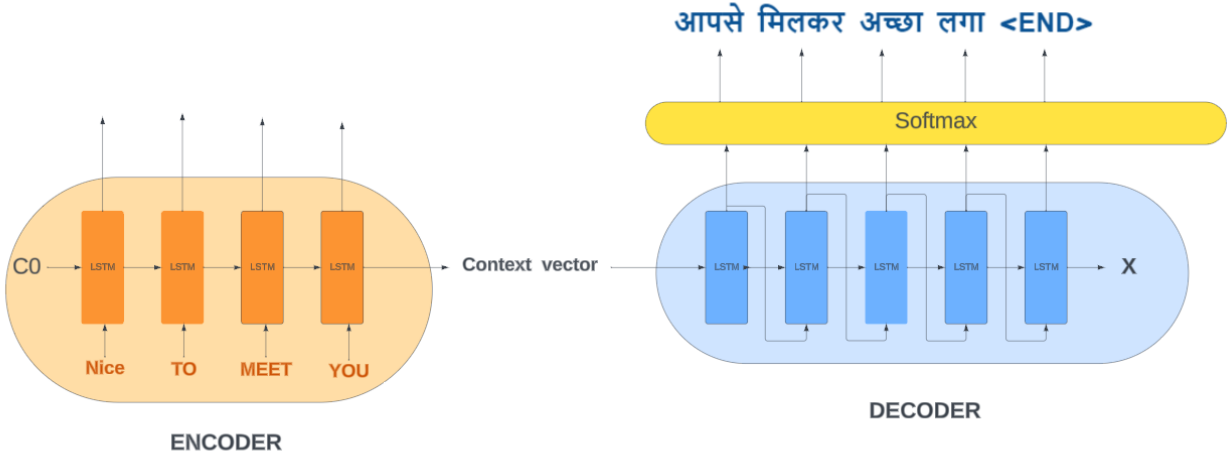
In this and the upcoming , we'll see different models, but they all have this encoder-decoder structure.



Encoder Decoder Model

The simplest encoder-decoder model consists of two RNNs (LSTMs are preferred): one for the encoder and another for the decoder. Encoder RNN reads the source sentence, and the final state is used as the initial state of the decoder RNN. The hope is that the final encoder state "encodes" all information about the source, and the decoder can generate the target sentence based on this vector.

This model can have different modifications: for example, the encoder and decoder can have several layers. Such a model with several layers was used, for example, in the paper Sequence to Sequence Learning with Neural Networks - one of the first attempts to solve sequence-to-sequence tasks using neural networks.



How is this Encoder Decoder Architecture Trained ?

Assume that we have the following dataset and we are about to train an encoder decoder architecture on it. So our english vocabulary contains 5 words and the hindi vocabulary contains 6 words + 2 special tokens which indicate when to start and end a sentence.

English Data	Hindi Data
Think about it	इसके बारे में सोचो
It is okay	ठीक है

English vocabulary	Hindi vocabulary
Think , about, it, is, okay	इसके, बारे , में , सोचो , ठीक, है, <start> , <end>

How is this Encoder Decoder Architecture Trained ?

Now to train our model we first tokenize our sentences and then we encode our words. For ease of understanding we will use one hot encoding but in a realistic scenario one will use something like word2vec for encoding.

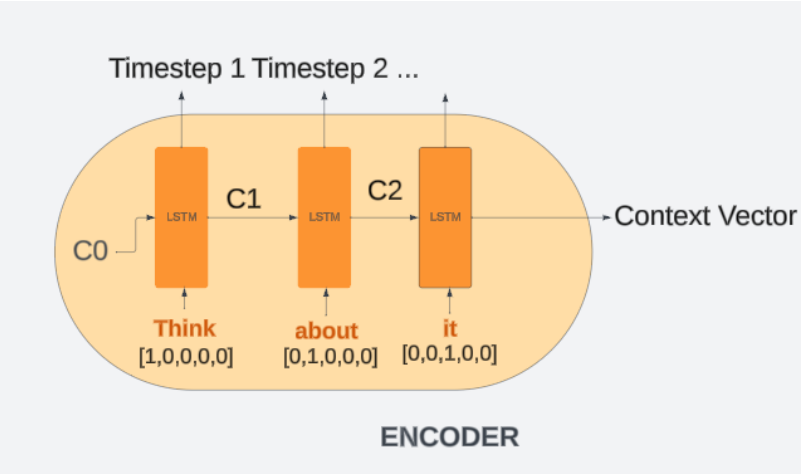
Word	Encoding
Think	[1,0,0,0,0]
about	[0,1,0,0,0]
it	[0,0,1,0,0]
is	[0,0,0,1,0]
okay	[0,0,0,0,1]

Word	Encoding
<start>	[1,0,0,0,0,0,0,0]
इसके	[0,1,0,0,0,0,0,0]
बारे	[0,0,1,0,0,0,0,0]
में	[0,0,0,1,0,0,0,0]
सोचो	[0,0,0,0,1,0,0,0]
ठीक	[0,0,0,0,0,1,0,0]
है	[0,0,0,0,0,0,1,0]
<end>	[0,0,0,0,0,0,0,1]

How is this Encoder Decoder Architecture Trained ?

Let us assume a model consisting of 1 layer of LSTM units for an encoder and 1 layer for the decoder along with a softmax layer. Thus initially all the weights of the network are randomly initialized.

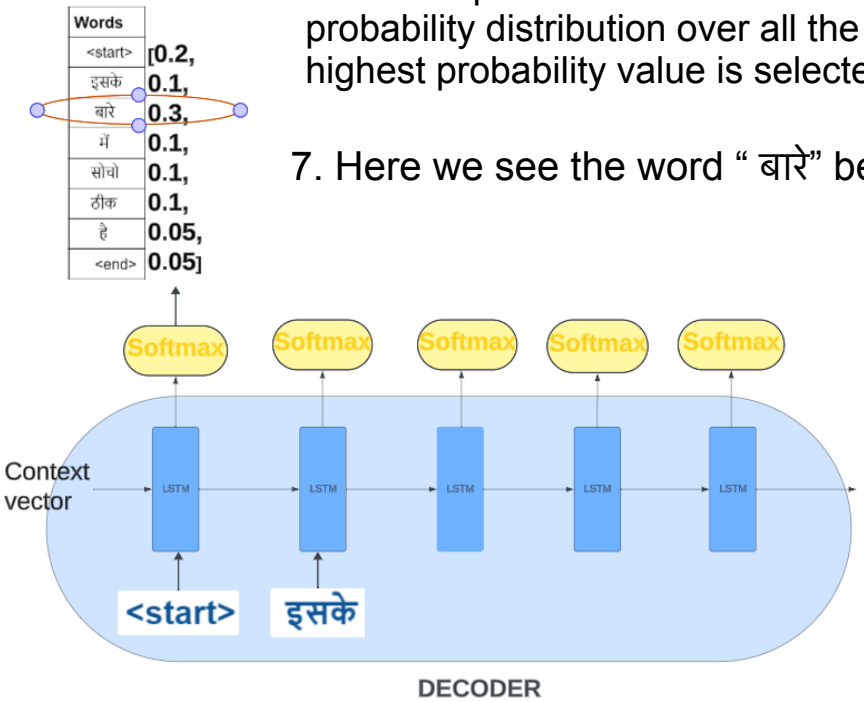
1. In the first timestep the first LSTM unit takes in an arbitrary hidden state value C0 along with the word embedding for the word “Think”.
2. It computers a new hidden state which is sent to the next unit which takes in the next word “about” (it is meant word embedding) as input.



3. This continues for the entire sentence and the output for the last cell is called the context vector this vector is sent over to the decoder.

How is this Encoder Decoder Architecture Trained ?

4. The hidden state input to the first cell of the decoder is the context vector and it also takes in the input a word embedding corresponding the speciat token **<start>** .
5. When the decoder encounters this token it starts generating a new sequence.



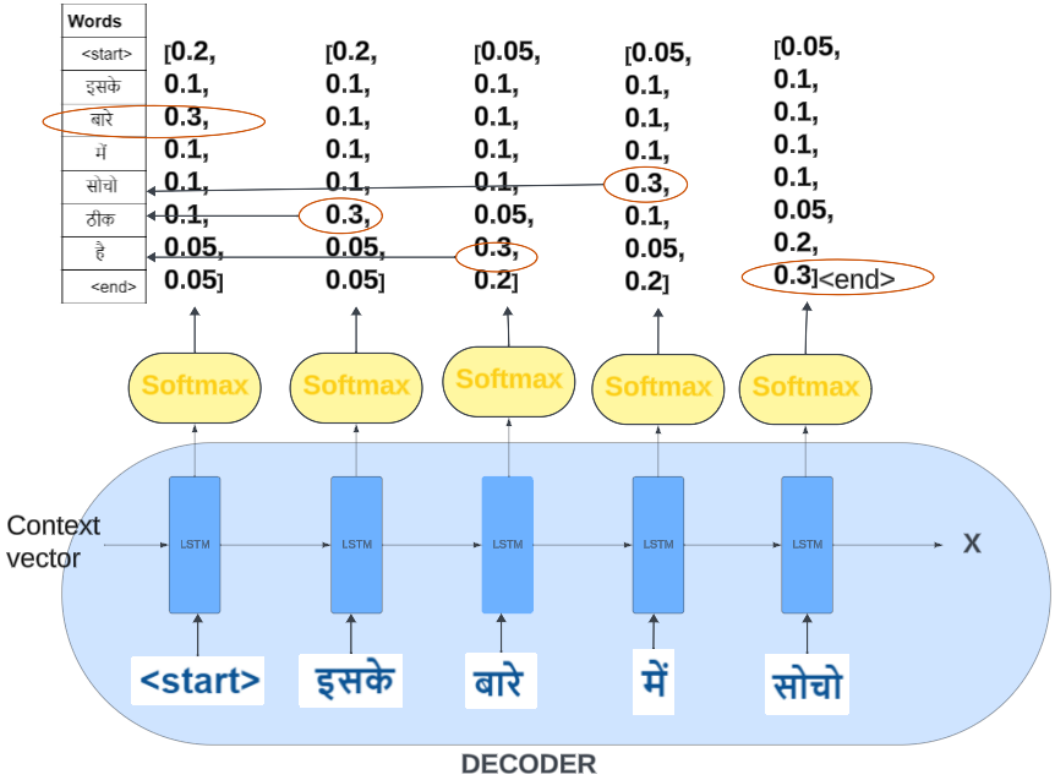
6. The output of the first cell of the decoder is passed to a softmax layer which outputs a probability distribution over all the words in the vocabulary . The word corresponding the highest probability value is selected as the output.

7. Here we see the word " बारे" being predicted which is a wrong prediction .

8. Ideally this predicted word should be forwarded to the next timestep . But the researchers behind this paper noticed that if the wrong prediction is forwarded the training is very slow , thus they decided to input the correct word to each cell while training. This idea is also known as "Teacher Forcing"

How is this Encoder Decoder Architecture Trained ?

9. Thus the word embedding of the next correct word and the hidden state is forwarded to the second cell which again returns a probability distribution from which the next predicted word is chosen.

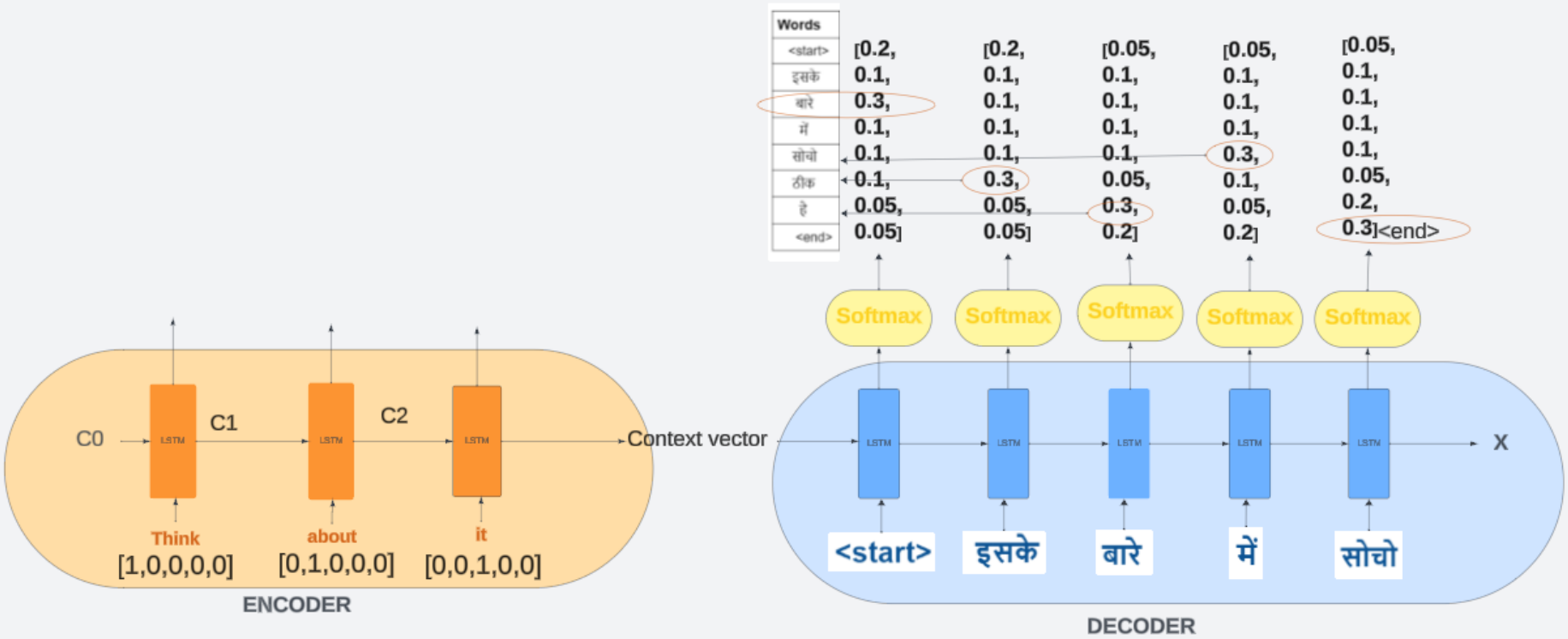


10. This process continues till the <end> token is predicted , on encountering the <end> token the model stops generating more words.

11. After this the loss is calculated and then the weights are adjusted via back propagation similar to that in an LSTM.

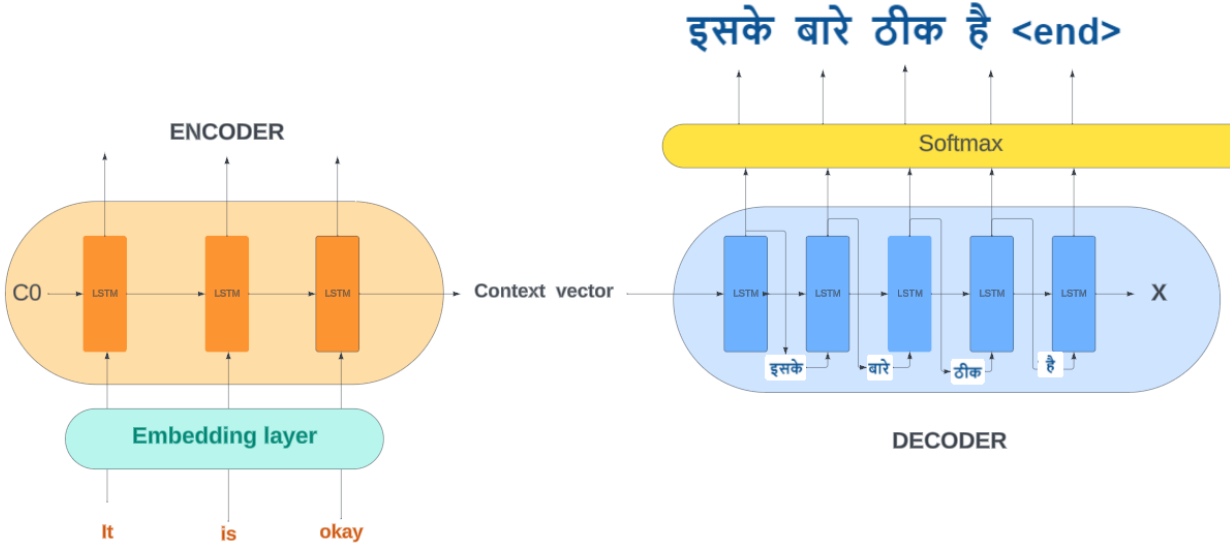
12. This process then continues for all the samples in the training set after which the model is said to be trained.

Full forward pass …



Prediction

Now that we are done training our model we would like to use it to translate the sentence “ It is okay” . As all the words belong to our vocabulary we are good to go.

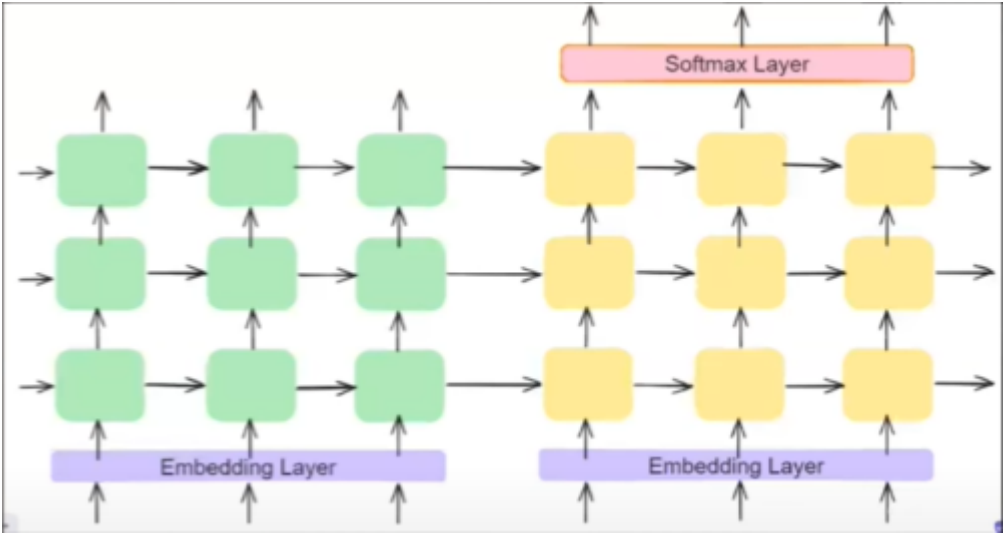


Wong predictions are quite common !!

1. The word embeddings for “It” , “is” and “okay” are fed to the encoder in 3 consecutive timesteps.
2. After the context vector is computed the decoder takes in as input the start token and generates the first prediction . Previously while training we would ignore this prediction if wrong but now we will use this generated word irrespective of the ground truth as the input to the next cell and so on till an <end> token is generated.

Illya Sutskever's Architecture

The model used by Ilya Sutskever and the Google research team in their 2014 paper "Sequence to Sequence Learning with Neural Networks"[1] is a similar network but with the following changes.



1. The model was focused on translating english to french.
2. The model was trained on a subset of 12 million sentences with 348 million french words and 304 million english words.
3. They use a deeper network with both encoder and decoder having 4 layers of LSTM Units with a 1000 units in each layer . This allows them to model more sophisticated data.
4. The model achieved a BLEU score of 34.81 surpassing the previous state of the art model.

Reversing the Source Sentences

While the LSTM is capable of solving problems with long term dependencies they discovered that the LSTM learns much better when the source sentences are reversed (the target sentences are not reversed). By doing so, the LSTM's test perplexity dropped from 5.8 to 4.7, and the test BLEU scores of its decoded translations increased from 25.9 to 30.6.

- The authors believe that it is caused by the introduction of many short term dependencies to the dataset. Normally, when a source sentence is concatenated with a target sentence, each word in the source sentence is far from its corresponding word in the target sentence.
- As a result, the problem has a large “minimal time lag” By reversing the words in the source sentence, the average distance between corresponding words in the source and target language is unchanged. However, the first few words in the source language are now very close to the first few words in the target language, so the problem's minimal time lag is greatly reduced.
- Thus, backpropagation has an easier time “establishing communication” between the source sentence and the target sentence, which in turn results in substantially improved overall performance.
- Initially they believed that reversing the input sentences would only lead to more confident predictions in the early parts of the target sentence and to less confident predictions in the later parts. How ever, LSTMs trained on reversed source sentences did much better on long sentences than LSTMs 4 trained on the raw source sentences which suggests that reversing the input sentences results in LSTMs with better memory utilization.
- However this observation is specific to their case and may not fetch better results every time

A note on Parallelization

- They used a C++ implementation of deep LSTM with the configuration from the previous section on a single GPU processes a speed of approximately 1,700 words per second.
- This was too slow so they parallelized thee model using an 8-GPU machine.
- Each layer of the LSTM was executed on a different GPU and communicated its activations to the next GPU / layer as soon as they were computed.
- The models have 4 layers of LSTMs, each of which resides on a separate GPU.
- The remaining 4 GPUs were used to parallelize the softmax, so each GPU was responsible for multiplying by a 1000× 20000 matrix.
- The resulting implementation achieved a speed of 6,300 (both English and French) words per second with a minibatch size of 128.
- Training took about a ten days with this implementation

We hope that the encoder decoder model was interesting to learn about . However the current state of the art in any NLP task is much better than this model , so we would encourage students to think about the possible pitfalls in the model demonstrated in this lecture.

References

[1]I. Sutskever, O. Vinyals, and Q. V. Le, ‘Sequence to Sequence Learning with Neural Networks’, *arXiv [cs.CL]*. 2014.

[2] https://www.youtube.com/watch?v=KiL74WsgxoA

[3] https://www.youtube.com/watch?v=zbdong_h-x4

[4] https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

UE21CS343BB2

Topics in Deep Learning

Dr. Shylaja S S
Director of Cloud Computing & Big Data (CCBD), Centre for Data Sciences & Applied Machine Learning (CDSAML)
Department of Computer Science and Engineering
shylaja.sharath@pes.edu

Ack: Anashua Dattidar, Teaching Assistant