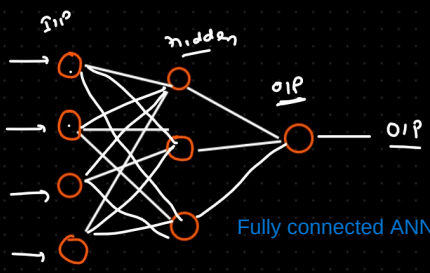
RNN

3 types of Neural Networks

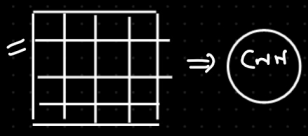
- ANN →
- CNN
- RNN



Fully connected ANN

Using ANN we can solve only classification and regression problem statement where dataset is present in the tabular or structured form

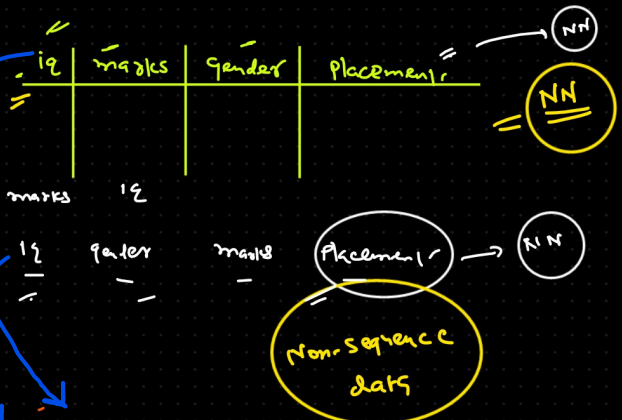
- Regression, classification

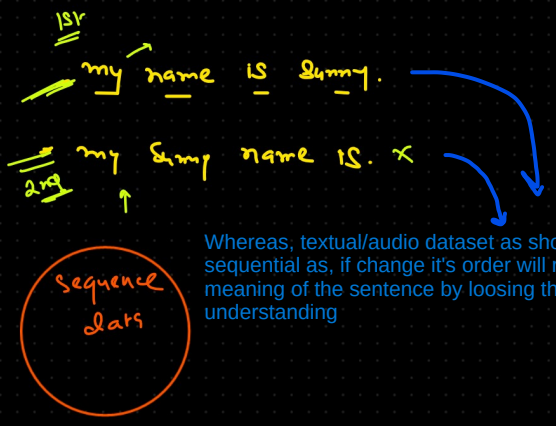


In grid like dataset (images/videos) which is unstructured we can apply CNN

For sequential data we have RNN

RNN ⇒ text, Audio, Speech, time related data (sequence)





Whereas, textual/audio dataset as shown above is sequential as, if change it's order will result in changing the meaning of the sentence by loosing the contextual understanding

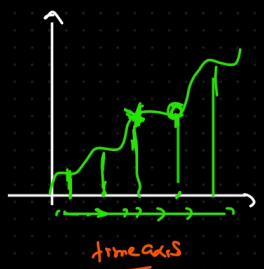
In above tabular dataset Placement is target variable. If we are passing input in {iq,marks,gender} or {iq,gender,marks} order then Placement will remain unaffected. This means tabular dataset are non sequential as order or sequence of input doesn't affect the output

Ex.

Examples of sequence dataset

1. Text
2. Time Series data
3. Speech.
4. DNA.
5. Protine
6. Purchase history
7. Surfing history (Web surfing history)

Google stock price



NLP ⇒

1. Heuristic approach (rule based approach)
2. ML based. ⇒ naive bayes
3. DL based ⇒ RNN

Different NLP model building approaches. Already, discussed in the previous lecture

Sequence data

- In today's match India will score 300

ANN — encoding

Numeric rep of data

Manual approach of representing the text into numbers

1. Document matrix
2. TF-IDF
3. OHE
4. Integer encoding

(Sparse matrix — more zero less no. — info. won't be of semantic)

Disadvantage of using this manual approach

semantic means sequence

Automated approach of representing the text into numbers using neural networks

Word embedding

Word2vec, ELMO, BERT, Roberta, Glove ⇒ NN

1. Dense vector
2. Semantic (more contextual)

Advantages of using the automated approach

Why RNN is useful and why not ANN in case of textual or any other sequence based dataset

1. Hi my name is Sunny.
2. I love cricket matches
3. India will won today's match

⇒ ANN — Vector Encoding

A vector encoding technique called one hot encoding done

- [1,0,0,0,0 ... 0] upto 14 0's → Hi
- [0,1,0,0,0 ... 0] upto 14 0's → my
- [0,0,1,0,0 ... 0] upto 14 0's → name

One hot encoding

14 = Vocabulary or unique word count

[hi, my, name, ... match]

- [1,0,0,0,0,0 ...]
- [0,1,0,0,0,0 ...]

For first sentence total there are 70 vector embedding space required, Hence, 70 neurons will be required in the Input layer

5x14 = 70

IIP

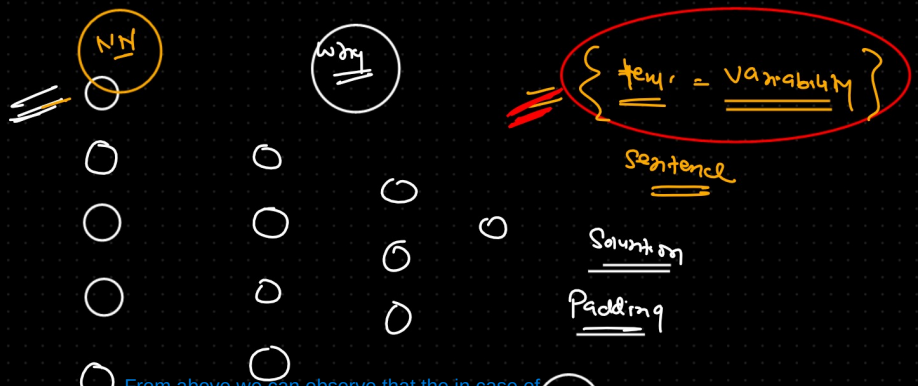
14 14 14 14 14

5x14 = 70

Similarly for 3rd sentence also 70 neurons will be required in the Input layer

For the 2nd sentence 56 neurons will be required at the I/P layer

$4 \times 14 = 56$



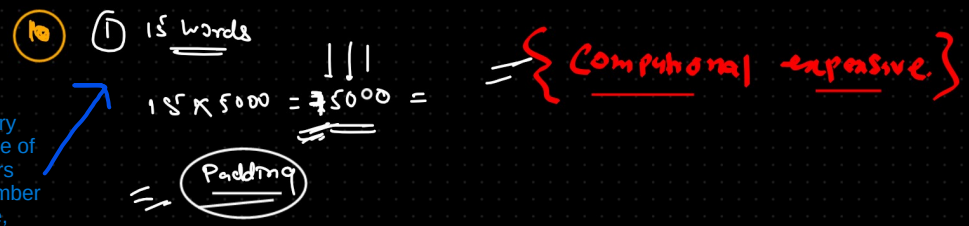
Solution

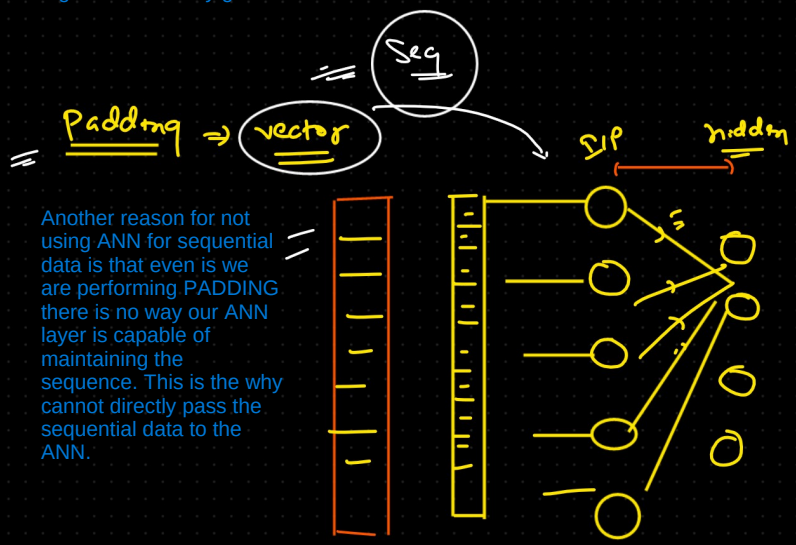
Padding

From above we can observe that the in case of sentences length of each sentence changes which also changes the number of neurons required at the input layer. But Dynamically we cannot change the length of the input layer. This can be first reason that why we cannot pass sequential data like text directly to the ANN and require special kind of NN called RNN

As a solution to variability of text length we can perform PADDING and convert the vectors into same length. This will make dynamic length of I/P layer a static one

In real word scenario their will be 5k vocabulary and at least 15 sentences(most basic use case of NLP). This will increase the numbers of vectors and then applying padding in such a huge number is very much computational expensive. Hence, Padding is also not very good solution



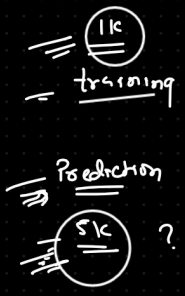


Another reason for not using ANN for sequential data is that even is we are performing PADDING there is no way our ANN layer is capable of maintaining the sequence. This is the why cannot directly pass the sequential data to the ANN.

Reason summarized. Tell if someone asks in the interview

Why not ANN for the text ?

1. text input → varying the size
2. Padding → computational expensive
3. not able to Preserve sequence
4. Prediction Problem



say for example ANN is trained on top of sentences where highest length of any sentence is 1K words. And say during prediction a sentence having 5k words is passed. So in such a case due to huge difference in the length of the words for both training and prediction, a prediction problem arises.

RNN ⇒ ANN

Roadmap RNN ⇒

1. Simple RNN (FP, BP)
2. Problems (exploding, vanishing)
3. Lstm, GRU
4. Deep RNN / Stacked RNN
5. Bidirectional RNN

{ one to many, many to one, many to many }

Web application which are NLP powered:
1. text2data.com == This can be used to perform the sentimental analysis
2. Next word prediction when we are drafting the email
3. English to Hindi translator by google
4. https://milhidaka.github.io/chainer-image-caption/ == This is generating image caption demo app where we can attach a picture and model running behind it will be able to provide a caption for that particular image

Context or Core Idea behind RNN: (Recurrent means to Repeat. RNN is the Timestamp based Neural Network)

RNN posses the internal memory. Let's understand how?
Actually, in the case of RNN at time stamp t the output that we are going to get through neural network, we are going to provide this output back to the input layer at timestamp t+1 via feedback loop. So in time stamp t+1 input layer will have vector embeddings of next word+ output of hidden layer at t timestamp. This is how we are going to preserve the sequence of any sequential data. This is how RNN retain the memory.

This will be decoded even in more better way in the future classes.