Sequnce to Sequence modelling

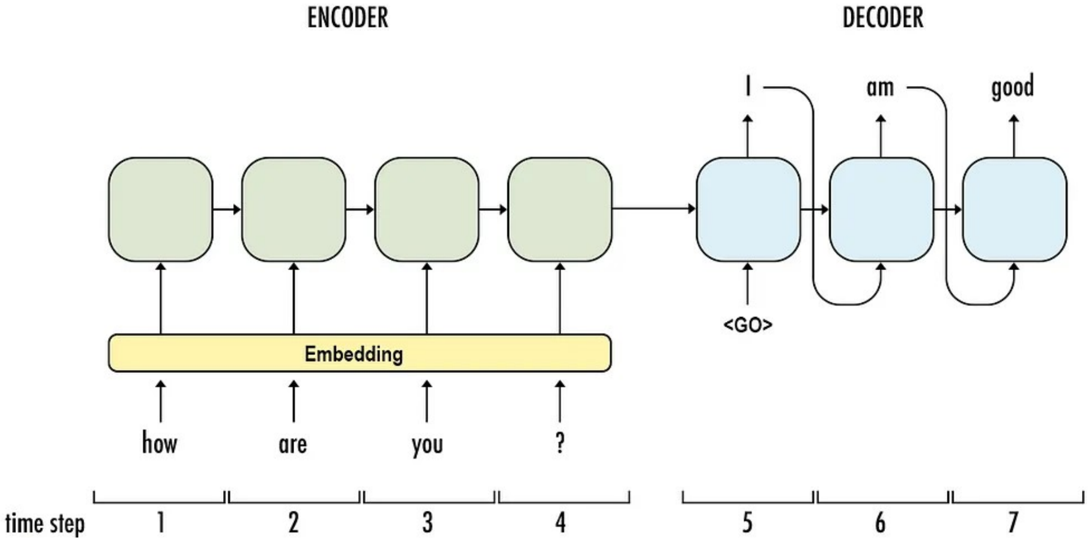
With Teacher Forcing and Attention Mechanism

This notebook explores the implementation of a Sequence-to-Sequence (seq2seq) model with attention and teacher forcing for the task of text summarization.

- **Text Summarization**: The process of condensing a longer piece of text (e.g., an article, document) into a shorter version while preserving the most important information.
- **Seq2seq Models**: A class of neural networks designed to handle sequence-to-sequence tasks, such as machine translation, text summarization, and question answering. They consist of an encoder that processes the input sequence and a decoder that generates the output sequence.
- **Attention Mechanism**: A key component in modern seq2seq models that allows the decoder to focus on different parts of the input sequence when generating each output token. This improves the model's ability to capture long-range dependencies and produce more accurate translations or summaries. (My notebook : https://www.kaggle.com/code/divyanshvishwkarma/seq2seq-with-attention-mechanism)
- **Teacher Forcing**: A training technique where the ground truth output tokens are fed as input to the decoder during training. This helps stabilize training and improve the quality of the generated output, especially in the early stages of training. (My notebook : https://www.kaggle.com/code/divyanshvishwkarma/teacher-forcing-in-seq2seq-tensorflow-and-keras)

Seq2Seq models

Seq2Seq models are a type of neural network architecture designed to handle tasks involving sequential data, such as machine translation and text summarization. They consist of two main components: an encoder, which processes the input sequence and creates a context vector, and a decoder, which generates the output sequence based on the context vector. Seq2Seq models have revolutionized many NLP tasks by effectively transforming one sequence into another.



About the data

Link : https://www.kaggle.com/datasets/gowrishankarp/newspaper-text-summarization-cnn-dailymail

The CNN / DailyMail Dataset is an English-language dataset containing just over 300k unique news articles as written by journalists at CNN and the Daily Mail. The current version supports both extractive and abstractive summarization, though the original version was created for machine reading and comprehension and abstractive question answering.

In [6]:

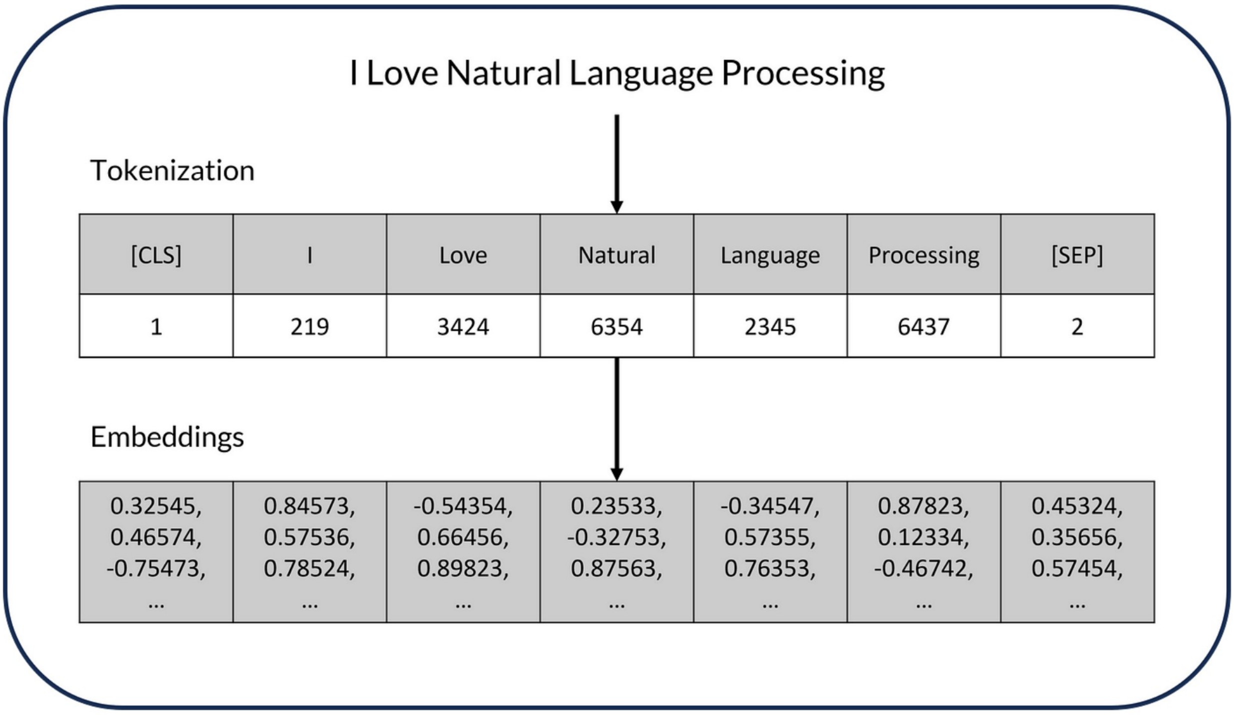
```
train.head(10)
```

Out[6]:

	article	highlights
0	By . Associated Press . PUBLISHED: . 14:11 EST...	Bishop John Folda, of North Dakota, is taking ...
1	Kabul, Afghanistan (CNN) -- China's top securi...	China's top security official visited Afghanis...
2	(CNN) -- Virgin, a leading branded venture cap...	The Virgin Group was founded by Richard Branso...
3	By . Chris Pleasance . Police are hunting for ...	Two men filmed taking iPad from canoe rental o...
4	Baghdad (CNN) -- Radical Iraqi cleric Muqtada ...	Muqtada al-Sadr has been in Iran since 2007 .\...
5	PUBLISHED: . 07:04 EST, 9 January 2014 . | . U...	Zhu Sanni, 23, had been left alone at home for...
6	Kabul, Afghanistan (CNN) -- Thousands of bottl...	Official: Bottles are almost exclusively from ...
7	(CNN) -- Tour de France race director Christia...	The 2013 Tour de France will start from the Fr...
8	(CNN) -- Hundreds filed by a casket on Sunday ...	Wes Leonard collapsed after scoring a winning ...
9	Earlier this season I picked Thierry Henry as ...	Sportsmail columnist Martin Keown was honoured...

Preprocessing data

The initial step involves converting the input and target text into sequences of tokens, which can be individual words or sub-word units. This is typically achieved through tokenization techniques. To ensure uniform input shapes for the model, the sequences are then padded with special tokens (e.g., <PAD>) to achieve equal lengths. Finally, to provide clear boundaries for the model, special start (<START>) and end (<END>) tokens are added to the beginning and end of the target sequences, respectively. This preprocessed data is then ready to be fed into the seq2seq model for training and inference.



[CLS]	I	Love	Natural	Language	Processing	[SEP]
1	219	3424	6354	2345	6437	2

0.32545, 0.46574, -0.75473, …	0.84573, 0.57536, 0.78524, …	-0.54354, 0.66456, 0.89823, …	0.23533, -0.32753, 0.87563, …	-0.34547, 0.57355, 0.76353, …	0.87823, 0.12334, -0.46742, …	0.45324, 0.35656, 0.57454, …

In [9]:

```
X, y = np.array(train.iloc[:, 0:1]), np.array(train.iloc[:,1:2])
```

Adding "start" and "end" token to the label datapoints

In [11]:

```
START = '<start>'
END = '<end>'
PAD = '<PAD>'
```

In [12]:

```
y = [f"{START} {text} {END}" for text in y]
```

In [13]:

```
size = -10
X_test, y_test = X[size:], y[size:]
X, y = X[:size], y[:size]
```

Preparing Tokenizer and finding vocabulary size

In [14]:

```
e_tk, d_tk = Tokenizer(), Tokenizer()
e_tk.fit_on_texts(X)
d_tk.fit_on_texts(y)
```

Converting text to sequences, padding them and finalizing the three series (enc_inputs, dec_inputs, targets)
analogous to (X, dec_target_input, y)