What is 'attention' in the context of LSTM? Why is it needed or what benefit does it provide over a regular seq2seq architecture with LSTMs? [2 points]

The problem with traditional word embeddings (like Word2Vec, GLoVe) is that they capture semantic meaning in a general context. However, a word could have different interpretations when used in a sentence with other words (River Bank and Money Bank). To address this, we need a mechanism that can pay attention to neiboring words in a sentence and build contextual embeddings for the word. This process is called attention mechanism.

In simple terms, attention is the weighted sum: **Bank**$_{\text{Contextual Embedding}}$ = **0.3 River**$_{\text{Embedding}}$ + **0.7 Bank**$_{\text{Embedding}}$

*Attention Weights

These are especially useful in Seq2Seq architectures (Machine Translation) where we need semantic meaning. If we put multiple LSTM blocks in parallel (Multi heads) and layers in parallel (Encoder layers) instead of a simple LSTM architecture, we can encapsulate different perspectives of an **ambiguous sentence**. A sentence like "A man saw an astronaut with a telescope" could have two different interpretations and this multi head attention via LSTMs helps capture both perspectives.

Also, there are longer sequences with a single negation word like **'don't', 'never' etc** and that changes the entire meaning of the sentence. Such situations require an attention mechanism that can capture the meaning of the all the words in relation to the single negation/double negatives etc.

Practical Exercise [8 points]

For this task, I picked French as the target language. I used English as the source language. The main work I did was collecting and preparing data.

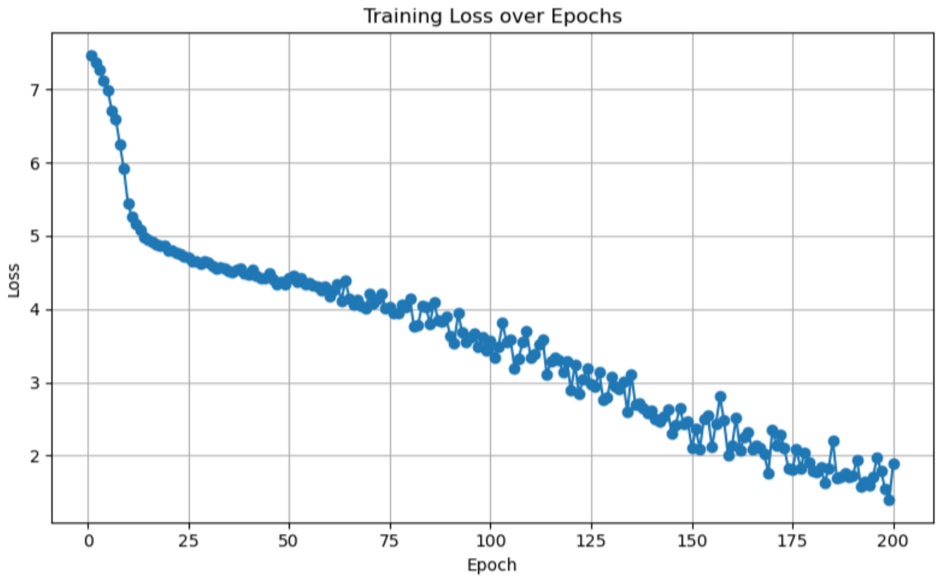
I used an English–French sentence dataset. I cleaned the text by:

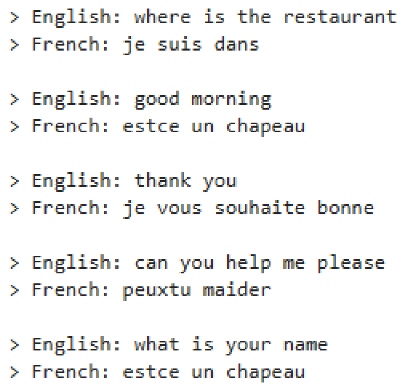
- Removing punctuation
- Lowercasing all the words
- Stripping extra spaces

Hyperparameters:

- Embedding Size: 128
- Hidden Size: 256
- Epochs: 200

Below is the graph showing training loss over 200 epochs. It shows that the model learned gradually, with the loss decreasing smoothly.





```
> English: where is the restaurant
> French: je suis dans

> English: good morning
> French: estce un chapeau

> English: thank you
> French: je vous souhaite bonne

> English: can you help me please
> French: peuxtu maider

> English: what is your name
> French: estce un chapeau
```