Summary of key topics from homework 6

What is a pre-trained NLP model?

A pre-trained model is a model that has already been trained on a huge amount of data. Usually, private people do not have the capabilities to train a model themselves to such an extent. Later, fine-tuning can make the model more specific for a certain task. These models can be used for various NLP tasks, including sentiment analysis, named entity recognition, and language translation.

How do I load them?

You can use libraries and frameworks such as Hugging Face’s Transformers, TensorFlow, or PyTorch to load a pre-trained NLP model. The specific steps for loading the model will depend on the library you choose. What is tokenization? What does fine-tuning mean? What types of NLP Models are there? What possibilities do I have with the Transformers package?

What is tokenization?

Tokenization refers to the process of breaking down a text into smaller units, known as tokens. These tokens can be words, sub-words, or even characters. This step is crucial in NLP tasks as it allows machine learning models to understand and process the text.

What is fine-tuning?

Fine-tuning involves taking a pre-trained NLP model and training it further on a smaller, specific dataset. This allows the model to adapt to a particular task and can improve its performance, as it has already gained a general understanding of natural language from its pre-training.

What types of NLP models are there?

There are several types of NLP models, including Bag of Words models, Neural Networks (such as Recurrent Neural Networks and Convolutional Neural Networks), Sequence-to-Sequence models, Attention models, and Transformers (such as BERT, GPT-2, and T5).

How is a transformer architecture built?

A transformer is a neural network architecture introduced in 2017, in a paper titled “Attention is all you need”, with a key feature being the self-attention mechanism. The architecture consists of an encoder and a decoder, each with multiple layers containing a multi-head self-attention mechanism and a feed-forward network. The self-attention mechanism allows the model to weigh the importance of different parts of the input sequence. The final layer produces a probability distribution over the output vocabulary. The transformer architecture effectively captures dependencies between input elements, making it well-suited for sequence-to-sequence tasks.