RNN networks

This notebook is for RNN related reflection questions in KAMK Syväoppiminen 1 -course.

Print your name

In [1]:

```python
## Your code here
print("Exercise by: Janne Bragge")
```

```
Exercise by: Janne Bragge
```

Reflection

1. What is RNN? What about LSTM or GRU?
2. When you should initialize RNN hidden layer? And how you should initialize layer?
3. What is attention method?
4. What is generative network?
5. What are Start-of-Sentence and End-of-Sentence markers?
6. What is Bleu score?
7. What are transformer networks?

Your answers here...

1.

Recurrent Neural Networks (RNNs) are a type of neural network designed to handle sequential data, such as text, speech, or time series data. Unlike traditional feedforward neural networks, RNNs have a feedback loop that allows them to maintain a memory of past inputs. This memory enables them to process sequences of data, where the order of the data points is important.

One of the key challenges in training RNNs is the vanishing gradient problem. This occurs when the gradients used to update the network's weights become very small as they are propagated back through time. This can make it difficult for the network to learn long-range dependencies in the data.

Long Short-Term Memory (LSTM) networks are a type of RNN that addresses the vanishing gradient problem. LSTMs use a special type of memory cell that can store information for long periods. This allows them to learn long-range dependencies in the data.

Gated Recurrent Units (GRUs) are another type of RNN that addresses the vanishing gradient problem. GRUs are similar to LSTMs, but they have a simpler architecture. This makes them faster to train than LSTMs.

Overall, RNNs, LSTMs, and GRUs are powerful tools for processing sequential data. They have been used to achieve state-of-the-art results in a variety of tasks, such as natural language processing, speech recognition, and machine translation. (Pointer, 2019)

2.

The RNN hidden state must be initialized before processing a sequence. The most common method is to initialize the hidden state with a zero vector. This is a simple and effective method, but in some cases, it may be useful to use other initialization methods, such as random values or pre-trained values.

Hidden state initialization is important because it can affect network performance. If the hidden state is initialized incorrectly, the network may learn incorrect dependencies or fail to learn long dependencies. (Pointer, 2019)

3.

The attention method is a technique that allows a neural network to focus on specific parts of the input. This is especially useful for tasks where the input has long dependencies, such as machine translation.

The attention mechanism works by allowing the network to calculate weights for each part of the input. These weights indicate how important each part is for a given output. The network then uses these weights to calculate a weighted average of the input, which serves as the network's output. (Pointer, 2019)

4.

Generative networks are neural networks designed to generate new data that resembles training data. These networks are widely used in tasks such as image generation, text generation, and music generation.

One of the most well-known types of generative networks is Generative Adversarial Networks (GANs). GANs consist of two networks: a generator and a discriminator. The generator produces new data, while the discriminator tries to distinguish the generated data from the real data. These two networks are trained together until the generator is able to produce data that the discriminator cannot distinguish from the real data. (Foster, 2023)

5.

Start-of-Sentence and End-of-Sentence markers are special characters used to indicate the beginning and end of a sequence. These markers are commonly used in natural language processing, especially in tasks such as machine translation and text generation.

The Start-of-Sentence marker (often) is added to the beginning of the sequence, while the End-of-Sentence marker (often) is added to the end of the sequence. These markers help the network learn where the sequence starts and ends. (Foster, 2023)

6.

The BLEU score is a metric used to evaluate the quality of machine translations. It compares the machine translation to one or more human-made reference translations and calculates a score based on how many n-grams (groups of consecutive words) the machine translation shares with the reference translations.

The BLEU score is a common metric in evaluating machine translations because it is automatic and easy to calculate. However, BLEU scores also have limitations because they do not take into account semantic or syntactic structure. (Geekforgeeks, 20.3.2025)

7.

Transformer networks are a neural network architecture that has revolutionized natural language processing. Unlike RNNs, which process sequences step by step, transformer networks process the entire sequence simultaneously. This allows them to learn long dependencies more efficiently and quickly.

Transformer networks use an attention mechanism that allows the network to focus on specific parts of the input. They also use layered interconnected networks that allow the network to learn complex dependencies.

Transformer networks have achieved significant results in many natural language processing tasks, such as machine translation, text generation, and question-answering systems. (Builtin, 20.3.2025)

References

Foster, D. (2023). Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play (2. painos). O'Reilly Media, Inc.

Pointer, I. (2019). Programming PyTorch for Deep Learning: Creating and Deploying Deep Learning Applications. O'Reilly Media, Inc.

NLP – BLEU Score for Evaluating Neural Machine Translation – Python, Geekforgeeks (https://www.geeksforgeeks.org/nlp-bleu-score-for-evaluating-neural-machine-translation-python/)

Transformer neural networks: a step-by-step breakdown, Builtin (https://builtin.com/artificial-intelligence/transformer-neural-network)