Deploying AI

Understanding Foundation Models

```
$ echo "Data Science Institute"
```

Introduction

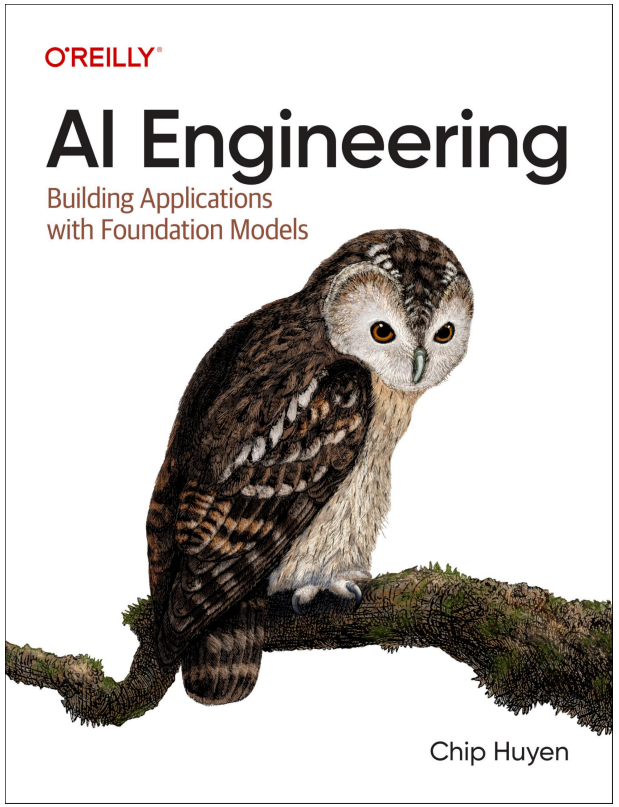
Agenda

Agenda

- From machine learning to foundation models via deep learning
- Training, pre-training, post-training models
- Sampling, hallucinations, and the probabilistic nature of AI

AI Engineering

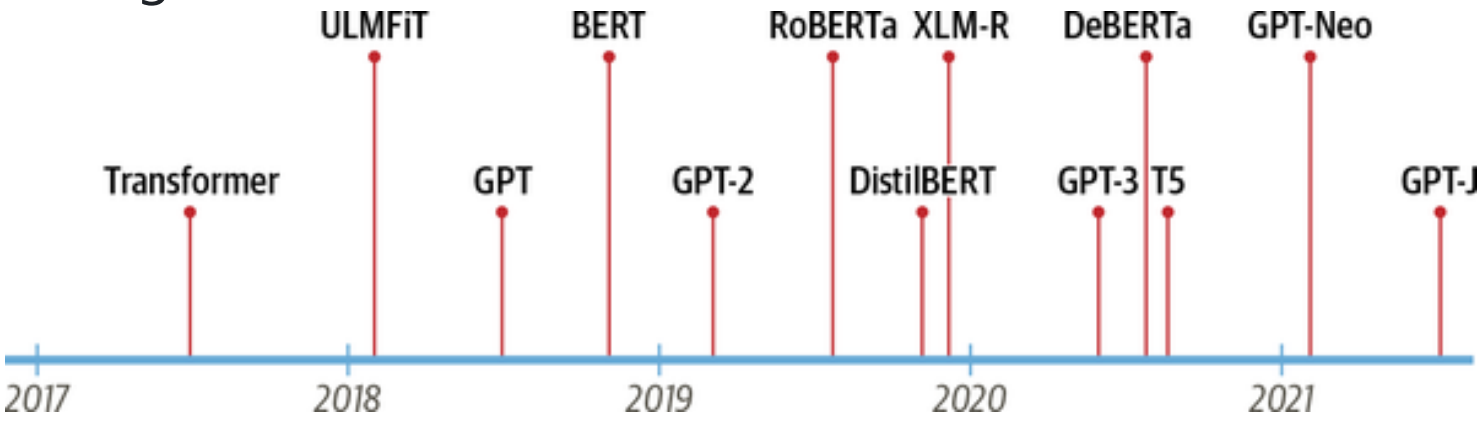
We will be covering Chapter 2 of AI Engineering, by Chip Huyen.



Understanding Foundation Models

Two Key Innovations

- Two key innovations have led to the current state of generative models:
  - Attention Mechanism
  - Transfer Learning



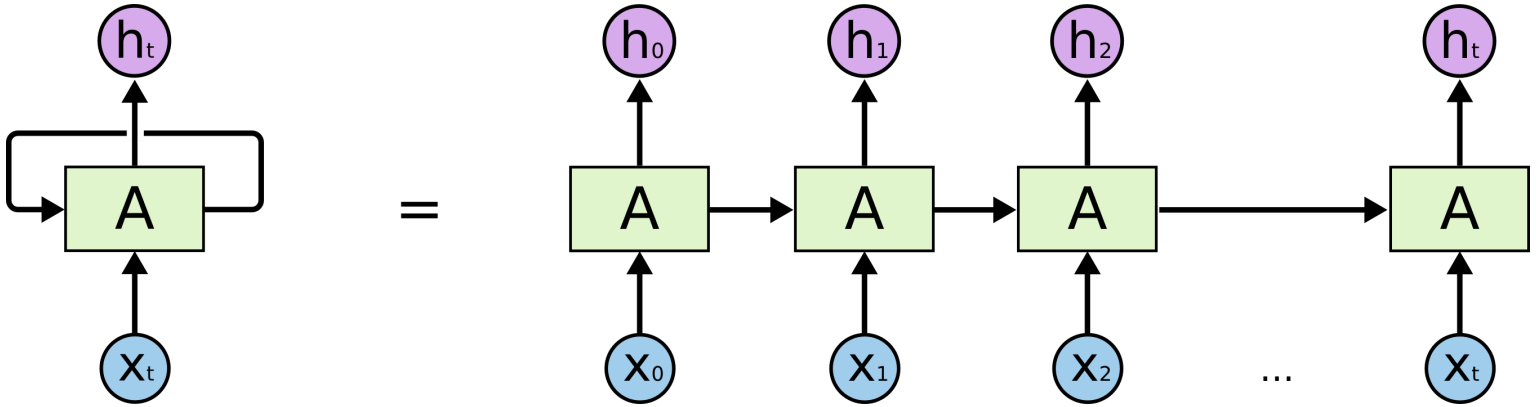
(Tunstall et al, 2022)

Key Models

> "In 2017, researchers at Google published a paper that proposed a novel neural network architecture for sequence modelling. Dubbed the Transformer, this architecture outperformed recurrent neural networks (RNNs) on machine translation tasks, both in terms of translation quality and training cost.
>
> In parallel, an effective transfer learning method called ULMFiT showed that training long short-term memory (LSTM) networks on a very large and diverse corpus could produce state-of-the-art text classifiers with little labeled data." (Tunstall et al, 2022)

Recurrent Neural Nets

- Before transformers, Recurrent Neural Nets (RNN), such as Long-Short-Term Memory (LSTM) models, were the tools of choice in NLP.
- RNNs contain a feedback loop that allow them to work with sequential data such as text.
- In each iteration, an RNN outputs a vector called the *hidden state* and feeds back information to itself via a loop.



An unrolled RNN (Olah, 2015)

Sequence-to-Sequence Models

- Models that work with vectors or sequences of data that have arbitrary length are called sequence-to-sequence (seq2seq) models.
- seq2seq models generally implement an encoder-decoder approach:
    - The encoder maps the input sequence into the *last hidden state*.
    - The decoder maps the *last hidden state* into an output sequence.
- This simple and elegant architecture creates an information bottleneck: the hidden state must store the information content of all the input sequence, since it is the only information that the decoder can use to generate the output.

Encoder-Decoder Framework

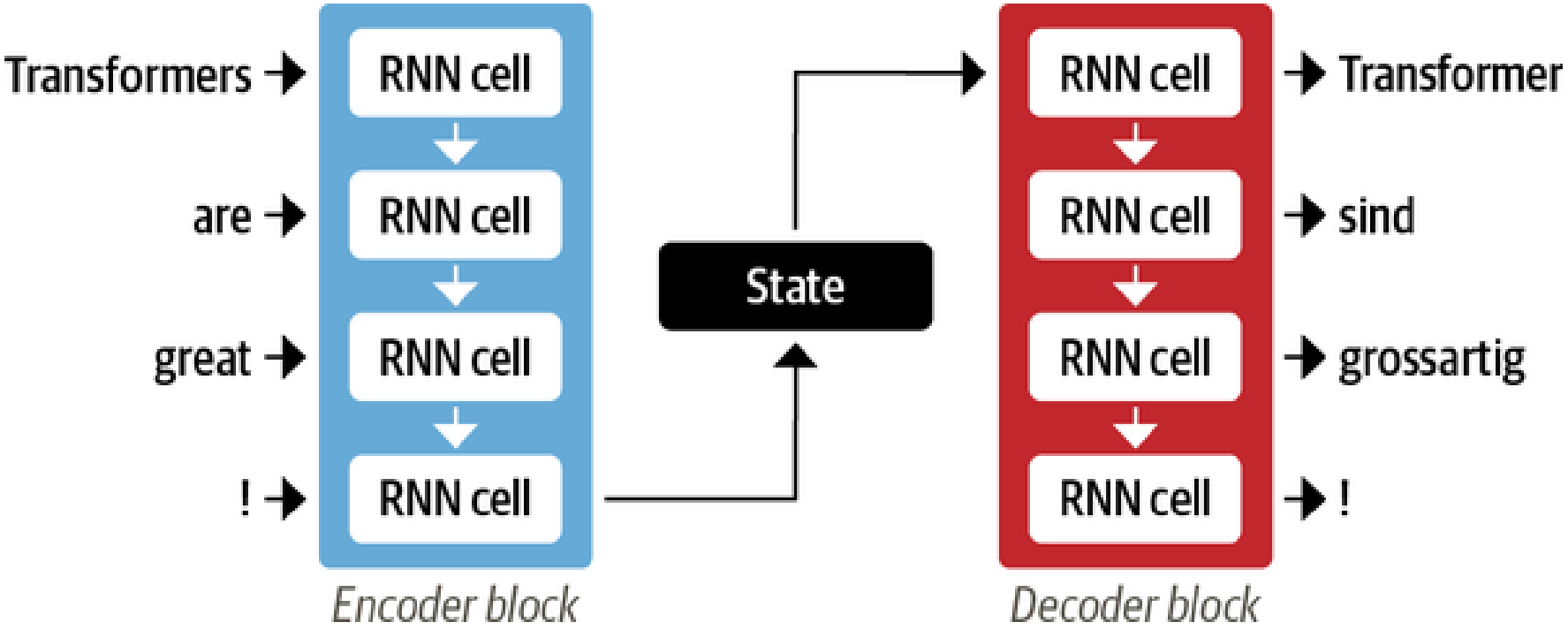


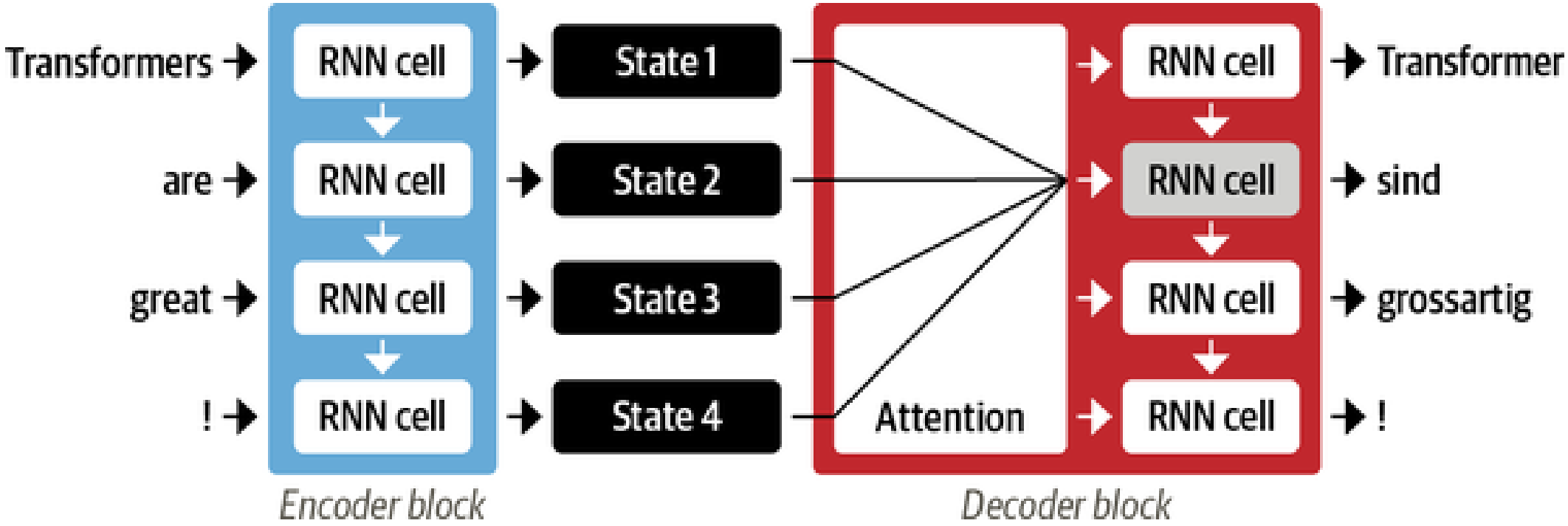
Illustration of a translation task using RNN (Tunstall et al, 2022)

Issues with seq2seq

There are two issues with seq2seq:

- Vanilla seq2seq decoder generates output tokens using only the *final* hidden state of the input.
- Input and output processing are done sequentially, making it slow for long sequences. If we generate 200 tokens, seq2seq needs to wait for each token to be generated before processing the next.

The Attention Mechanism



Translation task and attention mechanism (Tunstall et al, 2022)

Attention Enhances Performance

- Instead of producing a single hidden state for the input sequence, the encoder produces a hidden state at each step that the encoder can access.
- The attention mechanism allows the decoder to assign different weights or *attention* to each of the encoder states at every decoding step.
- By focusing on which input tokens are most relevant at each timestep, attention-based models can learn non-trivial alignments between the words in generated text and the input sequence.
- Attention enhances performance.

Seq2seq (RNN-based) vs Transformer

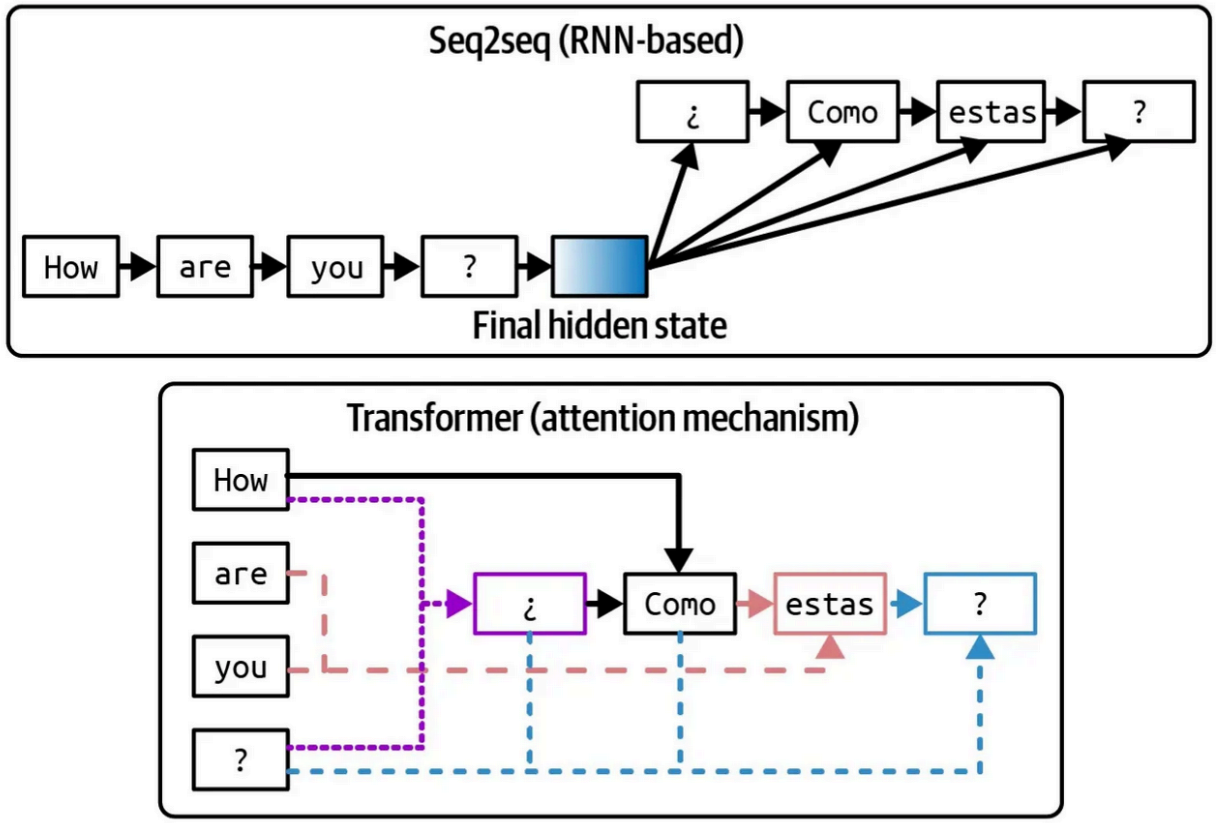


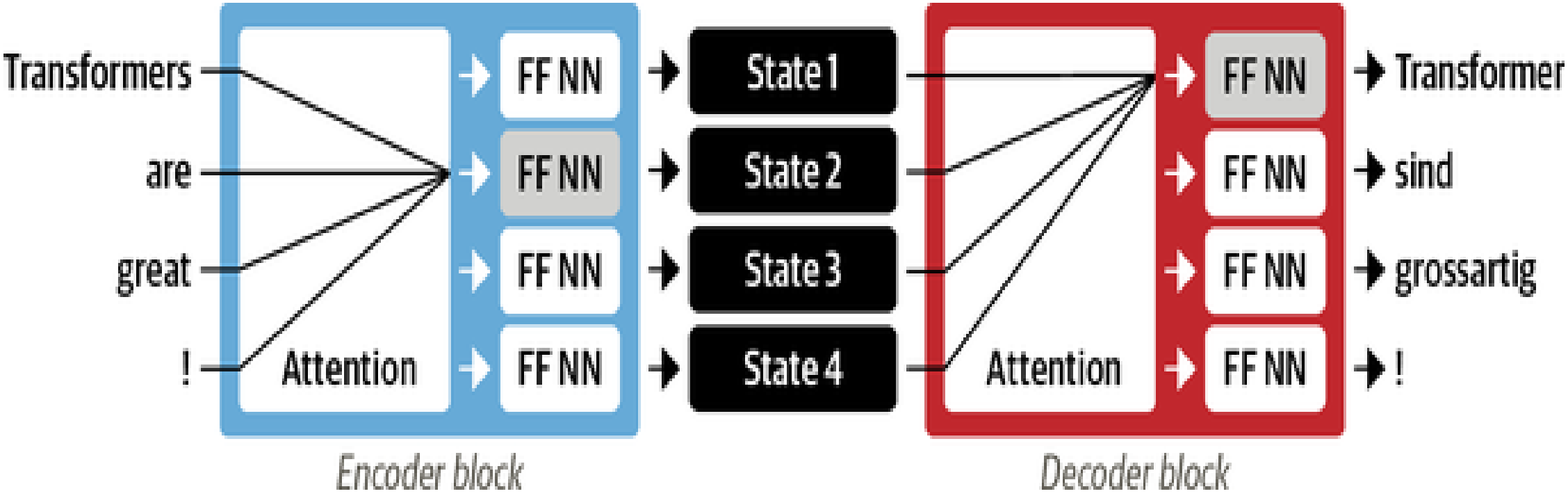
Figure 2-4. Seq2seq architecture versus transformer architecture. For the transformer architecture, the arrows show the tokens that the decoder attends to when generating each output token.

(Huyen, 2025)

Self-Attention (1/2)

- In the RNN architecture, computations are sequential and cannot be parallelized across the input sequence.
- Transformer paradigm: remove recurrence and rely entirely on a special form of attention called *self-attention*.
- Self-attention allows the attention mechanism to operate on all the states in the same layer of the neural network.
- A distinguishing characteristic of this type of models is self-supervision. Self-supervised learning takes advantage of natural labels: any text sequence can be used as labelled data.

Self-Attention (2/2)



Self-Attention (Tunstall et al, 2022)

Inference for Transformer-Based Language Models

Inference for transformer-based language models requires two steps:

- Prefill
  - Process input tokens in parallel.
  - Create the intermediate state necessary to generate the first output token.
- Decode
  - The model generates one output token at a time.

Attention

- The attention mechanism uses key, values, and query vectors.
- **Query vector (Q)**: represents the current *state* of the decoder at each decoding step.
- Each **key vector (K)** represents a previous token. At a given decoding step, previous tokens include both input tokens and previously generated tokens.

Attention

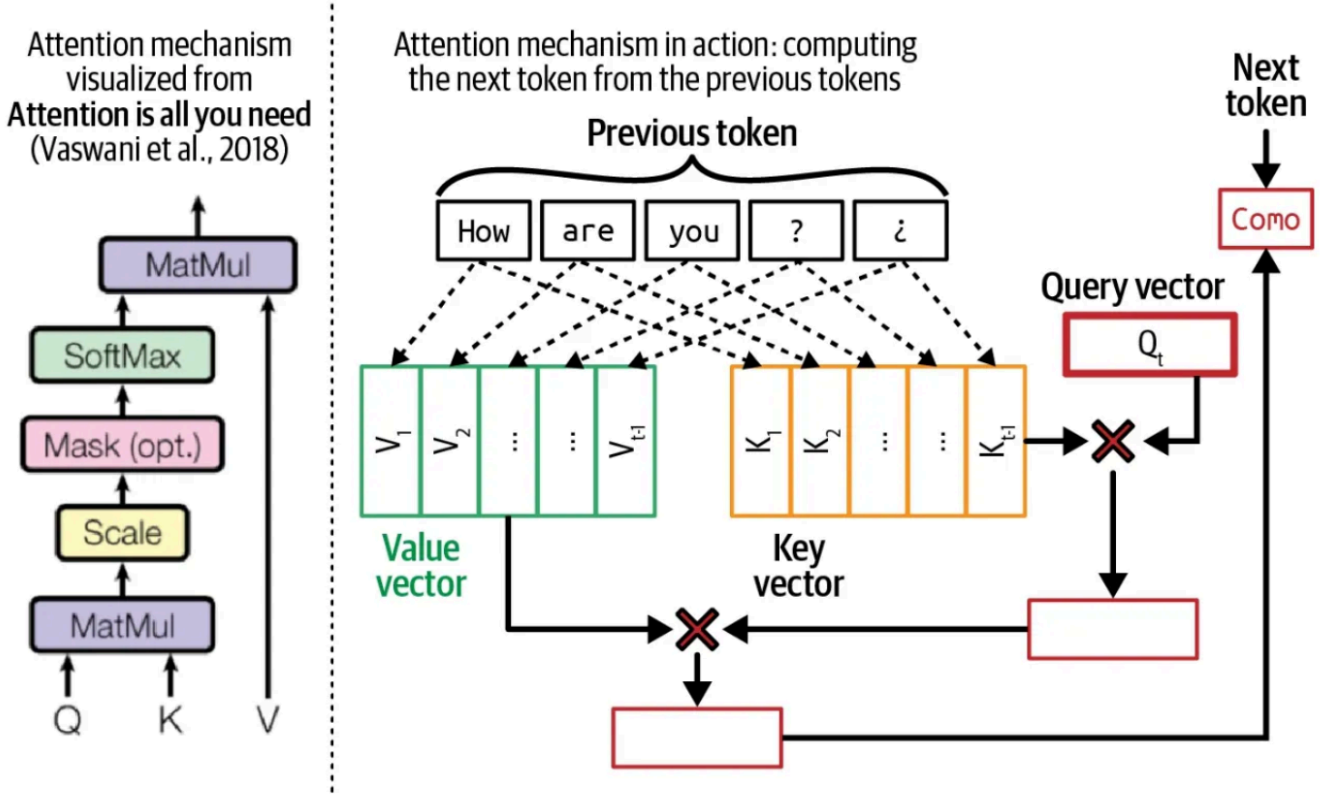


Figure 2-5. An example of the attention mechanism in action next to its high-level visualization from the famous transformer paper, "Attention Is All You Need" (Vaswani et al., 2017).

(Huyen, 2025)

Training, pre-training, post-training models

- Training data
  - Multilingual Models
  - Domain-Specific Models

Training Data

- An AI model is only as good as the data it was trained on. If there is no Spanish in the training data, the model cannot perform an English-Spanish translation.
- Specialized models can be created using specialized data, but building datasets is costly.

Standard Datasets

- Standard datasets are many times used to train LLMs.
    - CommonCrawl: non-profit sporadically crawls the internet and in 2022-2023 crawled 2-3 billion pages per month.
    - Colossal Clean Crawled Corpus (C4): Google provides a subset of Common Crawl.
- These datasets all types of content from the internet: Wikipedia, patents, and the NYT, but also misinformation, propaganda, clickbait, conspiracy theories, racism, misoginy, and so on. (WP, 2023)

Multilingual Models

- Modeling
  - Transformer Architecture
  - Attention Mechanism
  - Model Size
- Post-Training
  - Supervisde Finetuning
  - Preference Finetuning

Sampling, hallucinations, and the probabilistic nature of AI

References

References

- Dodge, Jesse et al. “Documenting the English Colossal Clean Crawled Corpus.” ArXiv abs/2104.08758 (2021).
- Huyen, Chip. Designing machine learning systems. O'Reilly Media, Inc., 2022
- Schaul, Kevin, et al. Inside the secret list of websites that make AI like ChatGPT sound smart. Washington Post: April 19, 2023 (URL).
- Olah, Chris. Understanding LSTM Networks. (colah.github.io, 2015)
- Tunstall, Lewis, Leandro Von Werra, and Thomas Wolf. Natural language processing with transformers. "O'Reilly Media, Inc.", 2022.