TensorFlow in Action (Part 1)

Chapter 5 : State-of-the-art in deep learning: Transformers

This chapter focuses on **Transformers**, the architecture behind recent breakthroughs in NLP and many other domains. It begins with the problem of representing text as numbers: raw strings must be tokenized into pieces (characters, words, or subwords), mapped to integer IDs, and then embedded into continuous vectors so that similar tokens have nearby representations. The author discusses common schemes like word-level and subword-level tokenization, and shows how an embedding layer in TensorFlow/Keras turns token IDs into dense vectors that serve as the model's input.

The core of the chapter is an intuitive and technical walkthrough of the **Transformer model**. Starting from the encoder–decoder view, the encoder converts an input sequence into a sequence of contextual representations, while the decoder consumes these representations and previously generated tokens to produce outputs (for example, in translation). The key innovation is the **self-attention** mechanism: rather than processing tokens strictly left-to-right like RNNs or focusing on local neighborhoods like CNNs, each token can directly "attend" to all other tokens in the sequence by computing attention weights based on learned queries, keys, and values. The chapter explains this first with simple scalar examples and analogies (such as a cooking competition), then generalizes to vector and matrix form, highlighting how self-attention captures long-range dependencies efficiently.

Building on single-head self-attention, the chapter introduces **multi-head attention**, where several attention "heads" run in parallel with different learned projections, allowing the model to focus on different types of relationships at once. Each Transformer layer combines multi-head attention with position-wise fully connected (feed-forward) layers, residual connections, and normalization to stabilize training and help gradient flow. The author also covers masked self-attention in the decoder, which prevents positions from attending to future tokens so that generation remains autoregressive. By the end, you have a conceptual blueprint for how a full Transformer encoder–decoder stack is built in TensorFlow: embedding and positional encoding layers, multiple blocks of attention plus feed-forward sublayers, and an output layer that produces token probabilities.