Natural Language Processing with RNNs and Attention

Oreily Publications

13 May 2024

1 Introduction

When Alan Turing imagined his famous Turing test1 in 1950, his objective was to evaluate a machine's ability to match human intelligence. He could have tested for many things, such as the ability to recognize cats in pictures, play chess, compose music, or escape a maze, but, interestingly, he chose a linguistic task. More specifically, he devised a chatbot capable of fooling its interlocutor into thinking it was human.2 This test does have its weaknesses: a set of hardcoded rules can fool unsuspecting or naive humans (e.g., the machine could give vague predefined answers in response to some keywords; it could pretend that it is joking or drunk, to get a pass on its weirdest answers; or it could escape difficult questions by answering them with its own questions), and many aspects of human intelligence are utterly ignored (e.g., the ability to interpret nonverbal communication such as facial expressions, or to learn a manual task). But the test does highlight the fact that mastering language is arguably Homo sapiens's greatest cognitive ability. Can we build a machine that can read and write natural language?

2 Approach

A common approach for natural language tasks is to use recurrent neural networks. We will therefore continue to explore RNNs (introduced in Chapter 15), starting with a character RNN, trained to predict the next character in a sentence. This will allow us to generate some original text, and in the process we will see how to build a Tensor- Flow Dataset on a very long sequence. We will first use a stateless RNN (which learns on random portions of text at each iteration, without any information on the rest of the text), then we will build a stateful RNN (which preserves the hidden state between training iterations and continues reading where it left off, allowing it to learn longer patterns). Next, we will build an RNN to perform sentiment analysis (e.g., reading movie reviews and extracting the rater's feeling about the movie), this time treating sentences as sequences of words, rather than characters. Then we will show how RNNs can be used to build an Encoder–Decoder architecture capable of performing neural machine translation (NMT). For this, we will use the seq2seq API provided by the TensorFlow Addons project.