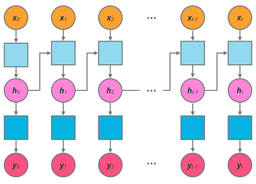
Unrolling an RNN

It's easier to understand how this works over time if we unroll it.



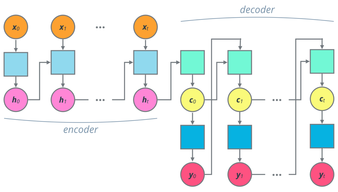
Basic RNN: Unrolled

Each copy of the network you see represents its state at the respective time step. At any time *t*, the recurrent layer receives input x_t as well as the state vector from the previous step, h_{t-1}. This process is continued till the entire input is exhausted.

The main drawback of such a simple model is that we are trying to read the corresponding output for each input word immediately. This would only work in situations where the source and target language have an almost one-to-one mapping between words.

Encoder-Decoder Architecture

What we should ideally do is to let the network learn an internal representation of the entire input sentence, and then start generating the output translation. In fact, you need two different networks in order to achieve this.



Encoder-Decoder: Unrolled

The first is called an Encoder, which accepts the source sentence, one word at a time, and captures its overall meaning in a single vector. This is simply the state vector at the last time step. Note that the encoder network is not used to produce any outputs.

The second network is called a Decoder, which then interprets the final sentence vector and expands it into the corresponding sentence in the target language, again one word at a time.

The first time step for the decoder network is special. It is fed in the final sentence vector from the encoder h_t, and given a sentinel input to kickstart the process. The recurrent portion of the network produces a state vector c_0 and with that the fully-connected portion produces the first output word in the target language, y_0.

At each subsequent time step *t*, the decoder network uses its own previous state c_{t-1} as well as its own previous output y_{t-1} in order to produce the current output, y_t.

This process is typically continued for a fixed number of iterations, with the idea that the network will start producing special padding symbols after all meaningful words have been generated. Alternately, the network could be trained to output a stop symbol, such as a period (.) to indicate that the translation is complete.