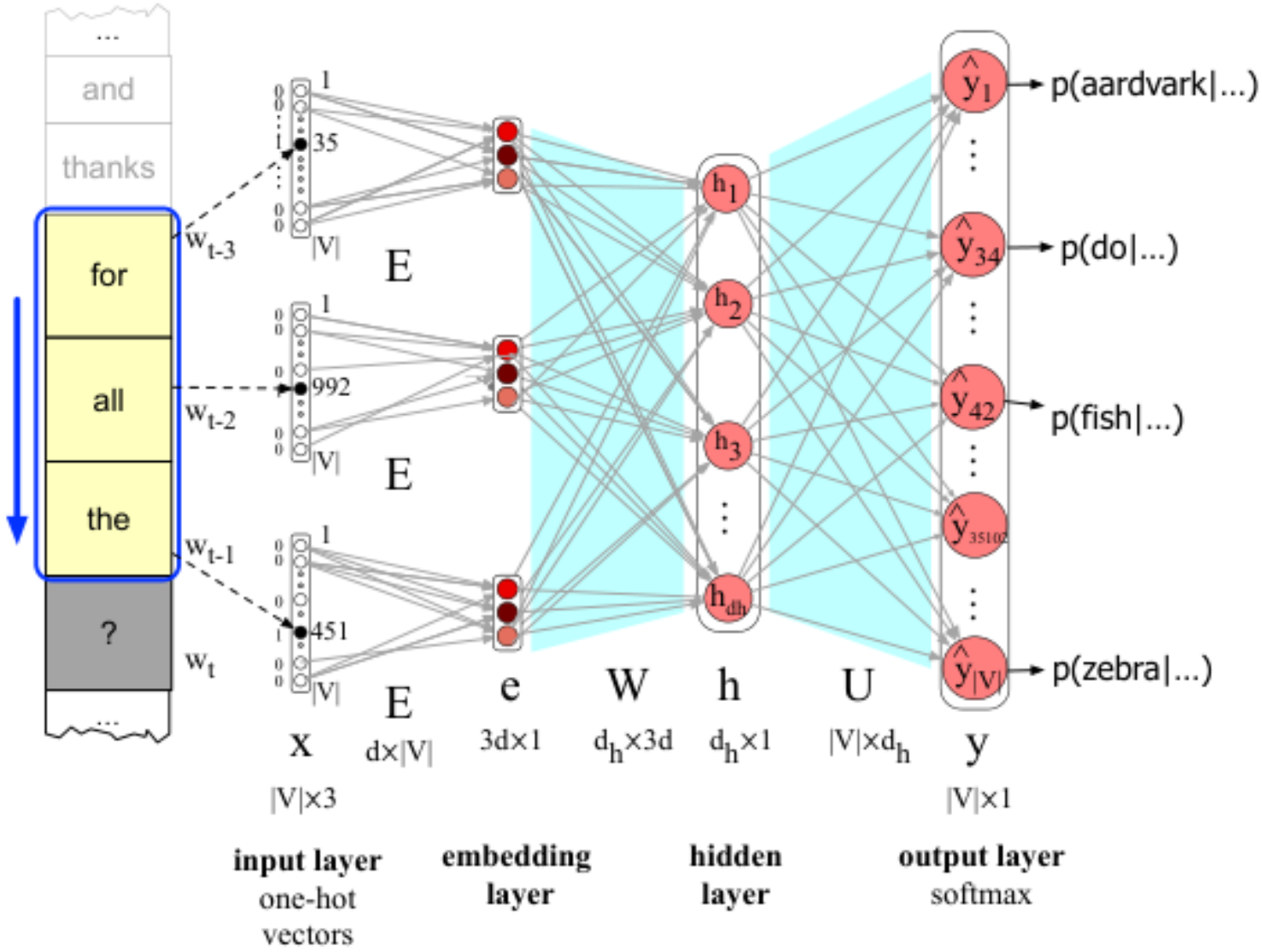
Neural Language Models

From Jurafsky and Martin

- Long been discussed; began to take off ∼2013
- Recurrent neural nets
- Transformers: the major idea
  - Many improvements in computing and architecture
  - Separate pretraining (large, expensive) from fine-tuning (targeted, efficient)

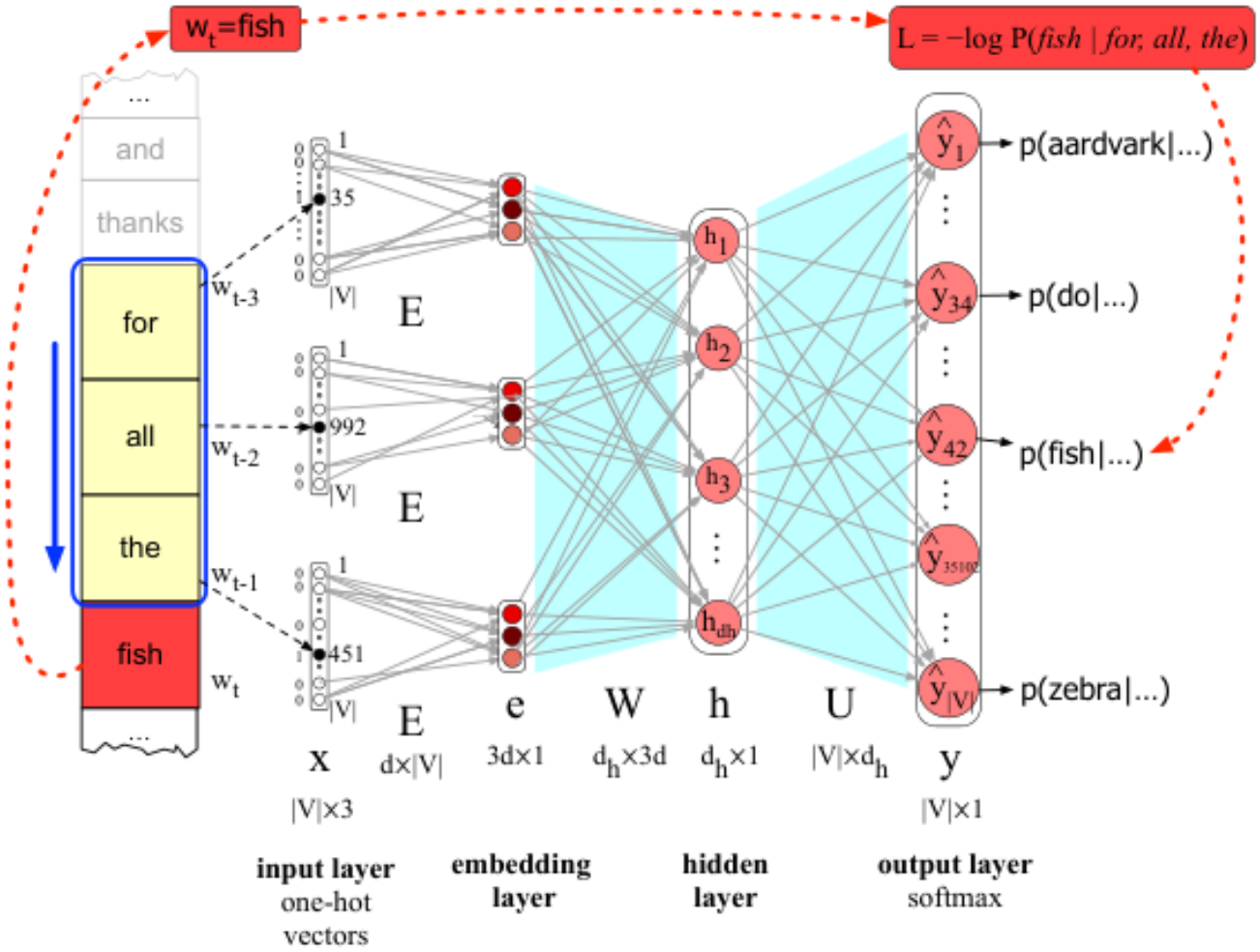
Forward Inference in a Feedforward Neural Language Model

Figure 7.17. Shows a context of three preceding tokens



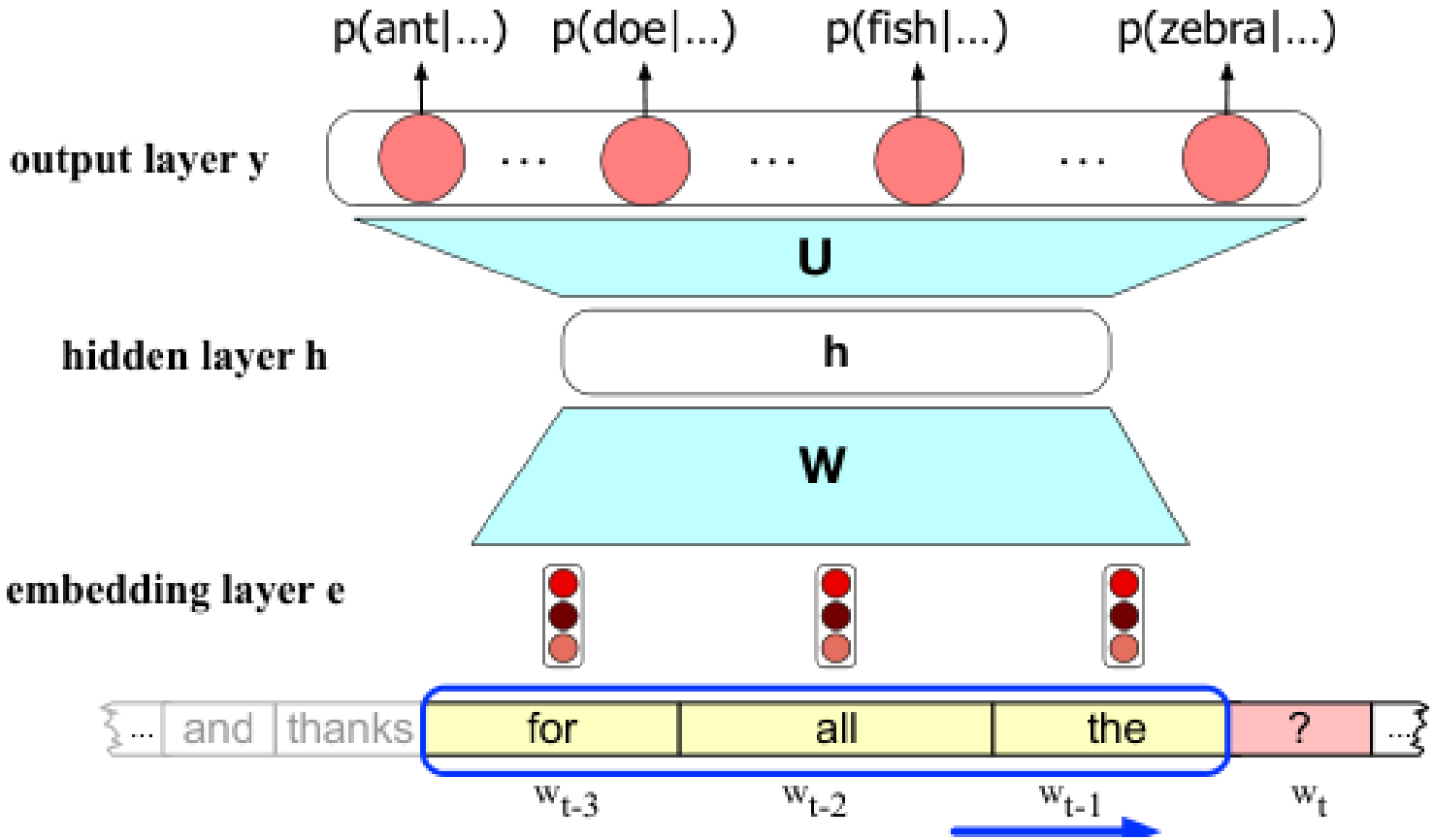
Learning Embeddings

Figure 7.18. Learn embeddings based on loss with respect to the actual word



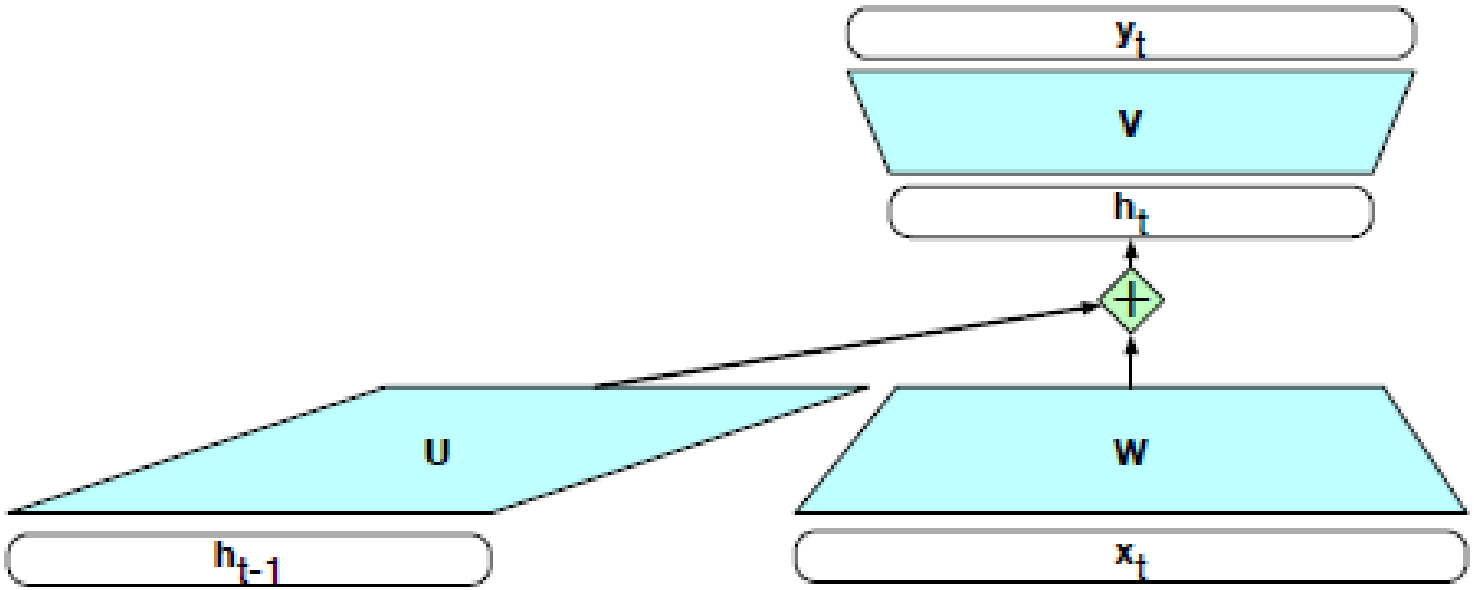
Forward Inference: Sliding Window

Figure 9.1 (from a previous edition)



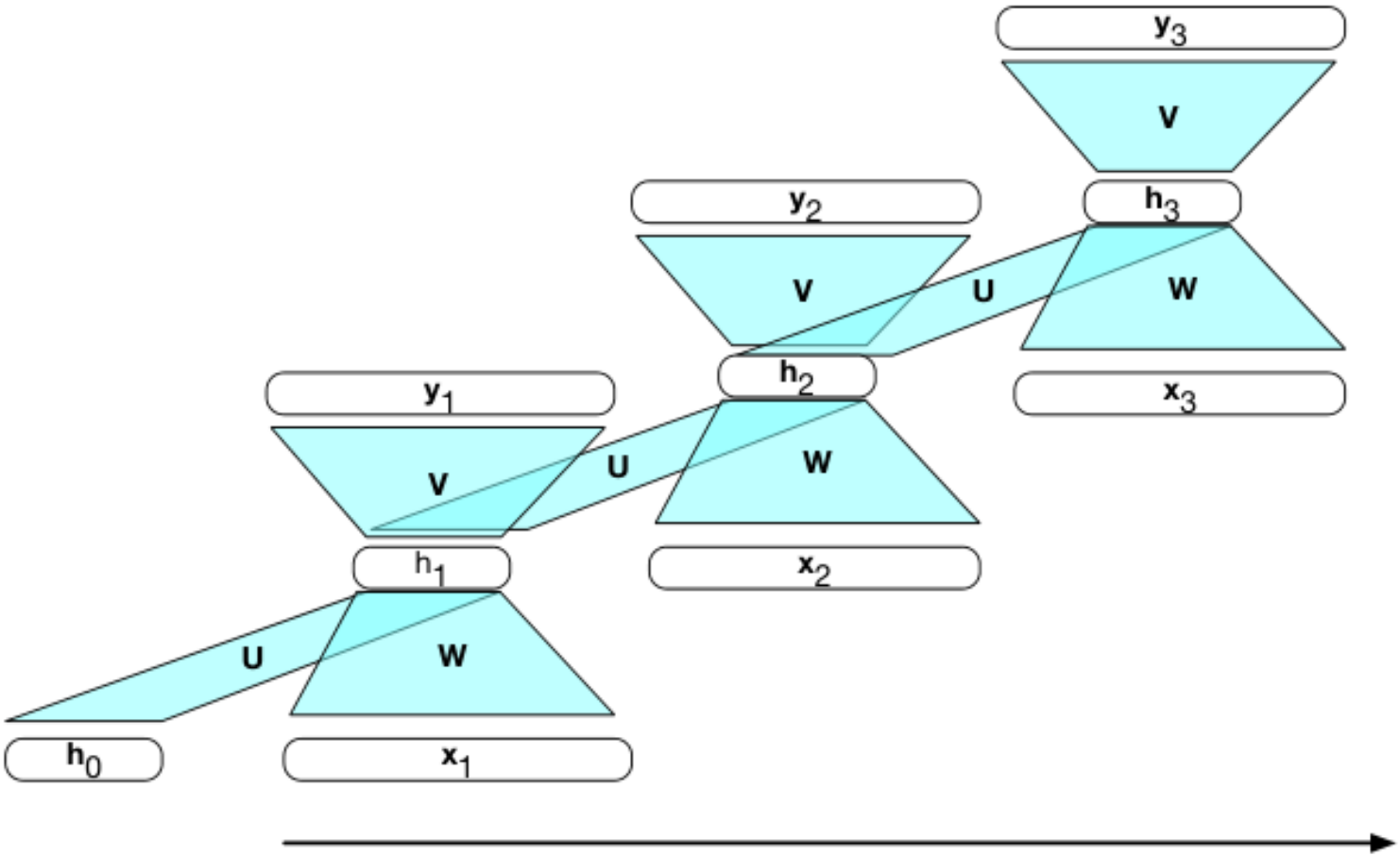
Recurrent Neural Network (RNN)

Figure 8.2. The hidden state is incrementally built up



RNN Unrolled Over Time

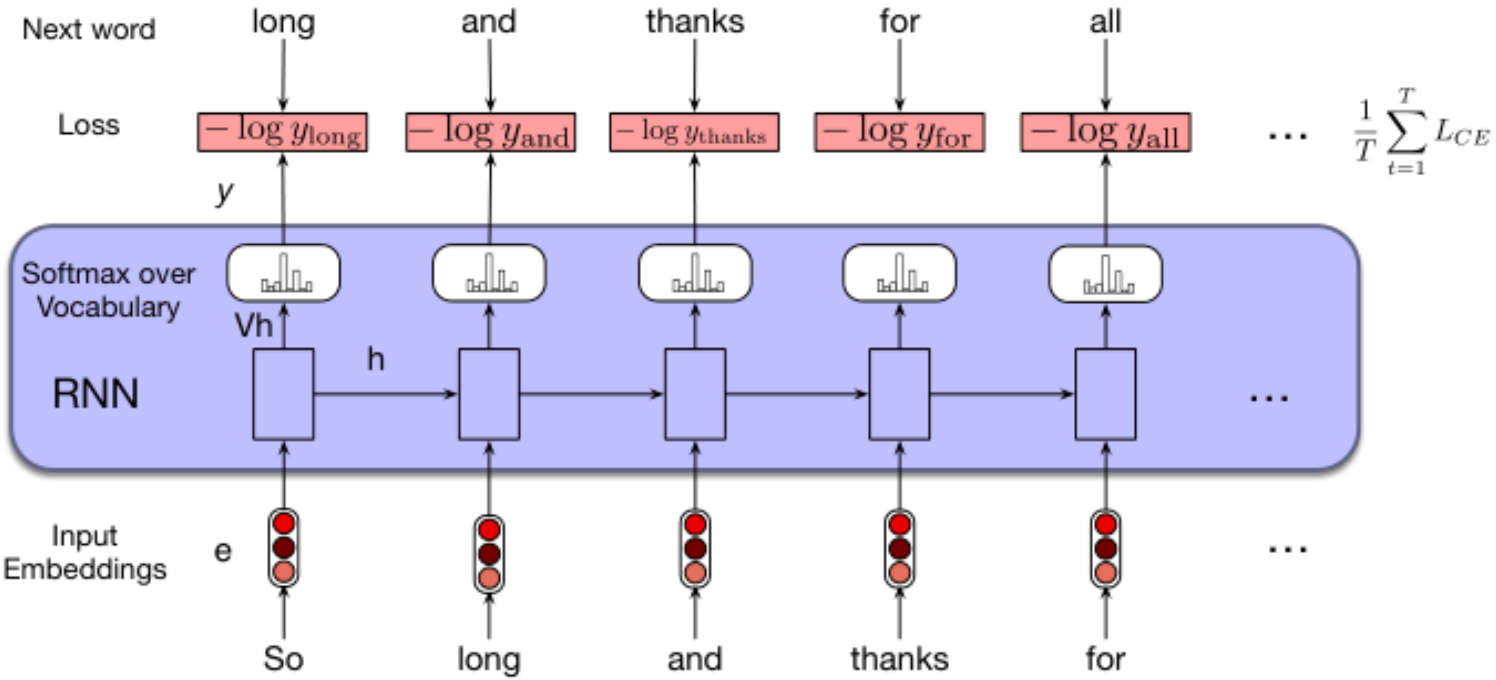
Figure 8.4. Notice the long chain



Training an RNN as a Language Model

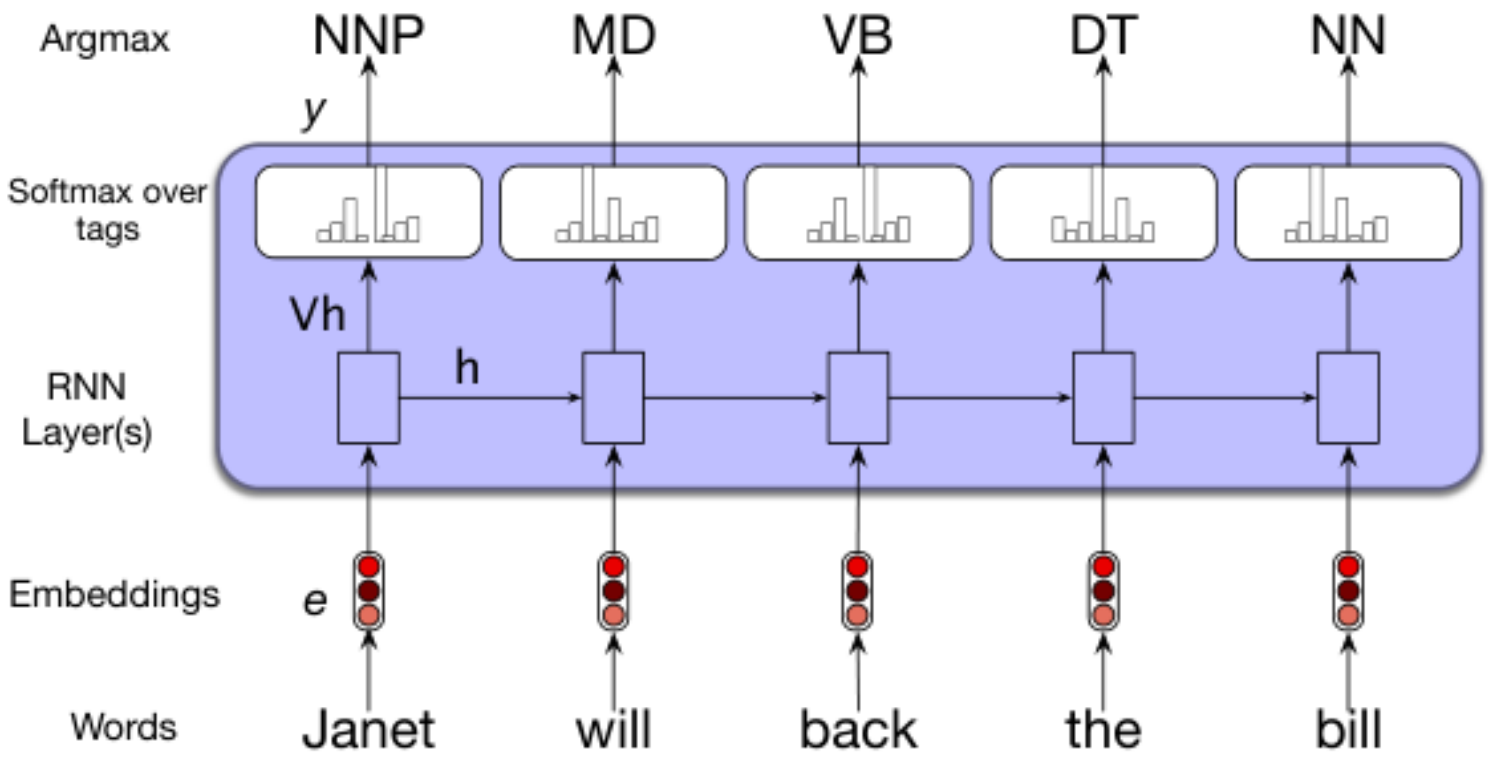
Figure 8.6. Trains iteratively

Uses correct token for subsequent steps so the errors don't accumulate



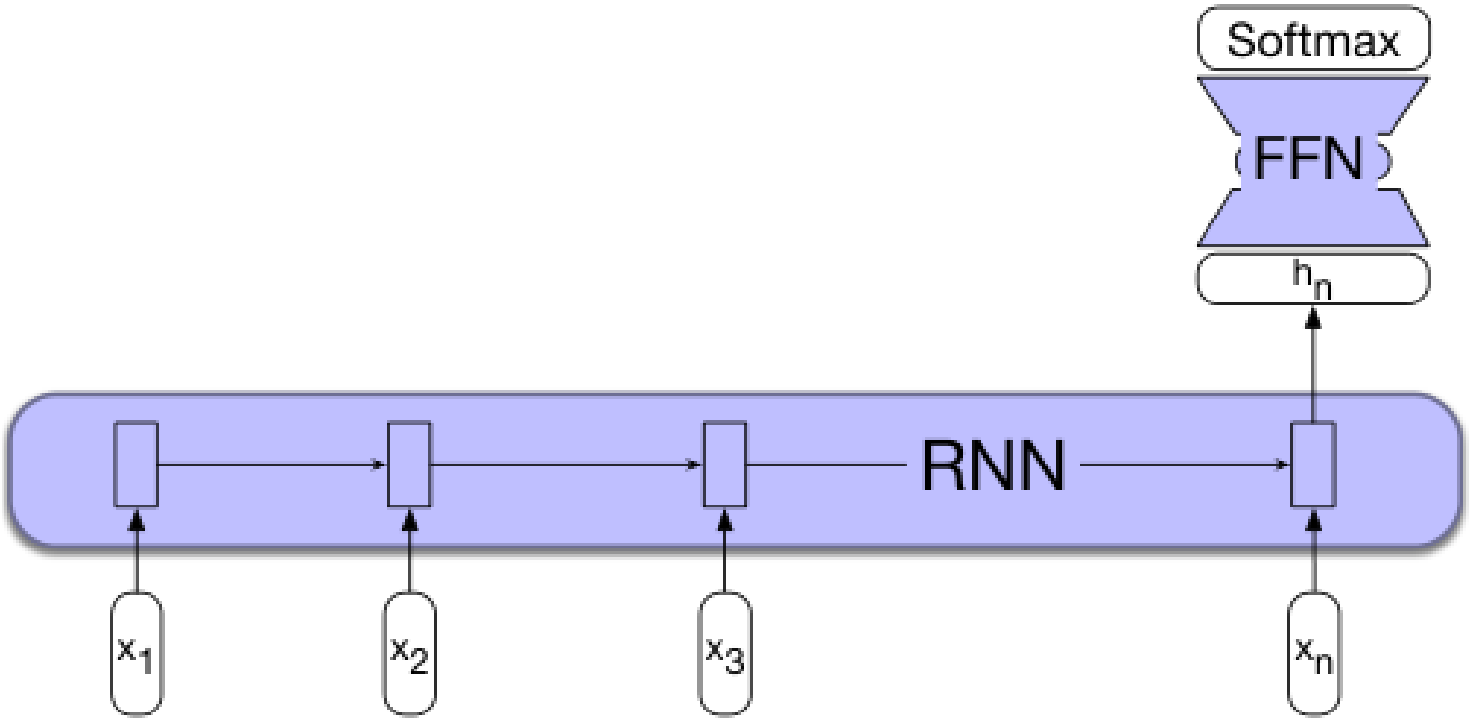
POS Tagging via an RNN

Figure 8.7. Example of sequence labeling



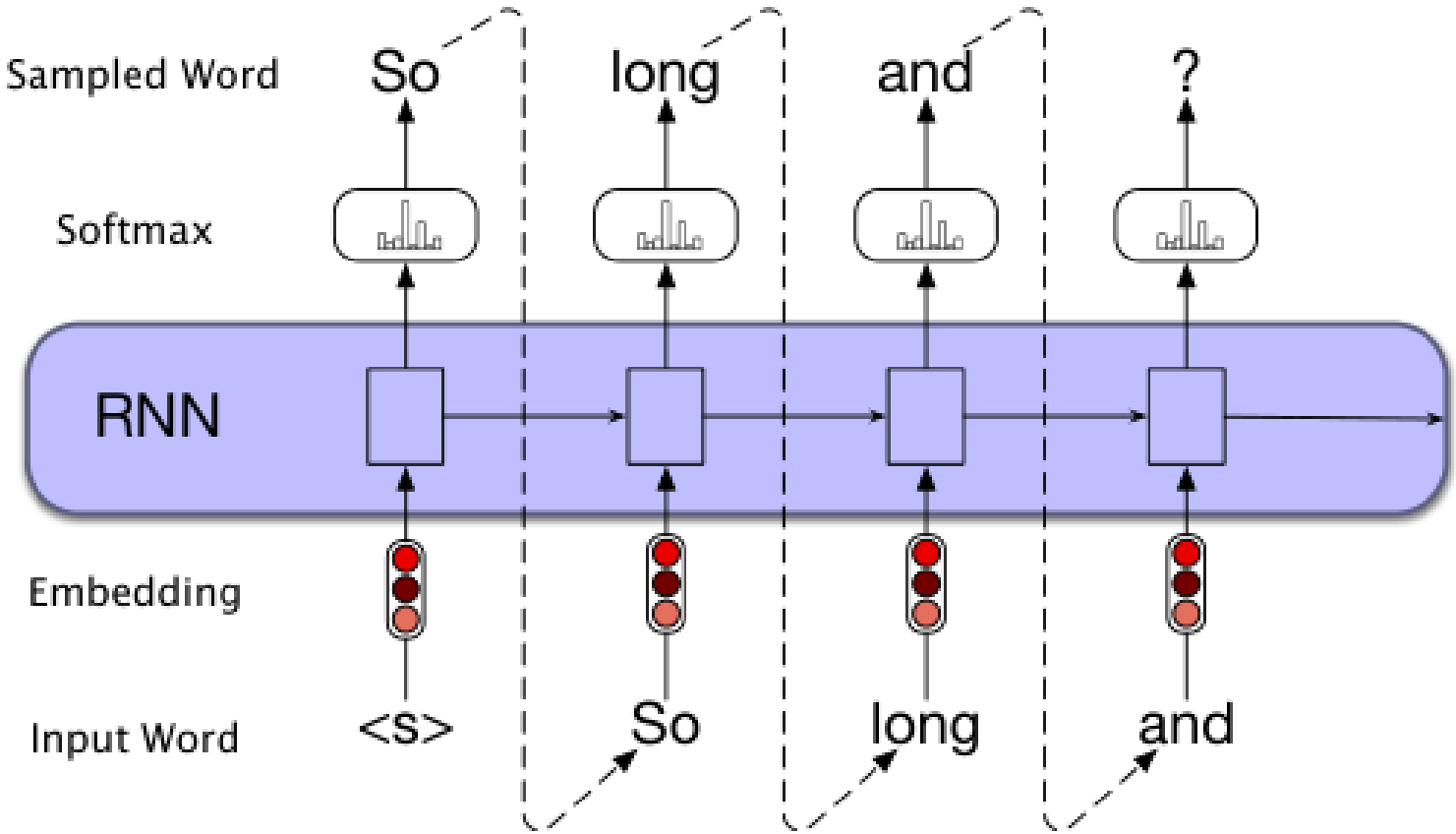
Sequence Classification

Figure 8.8. Uses the last hidden state to classify



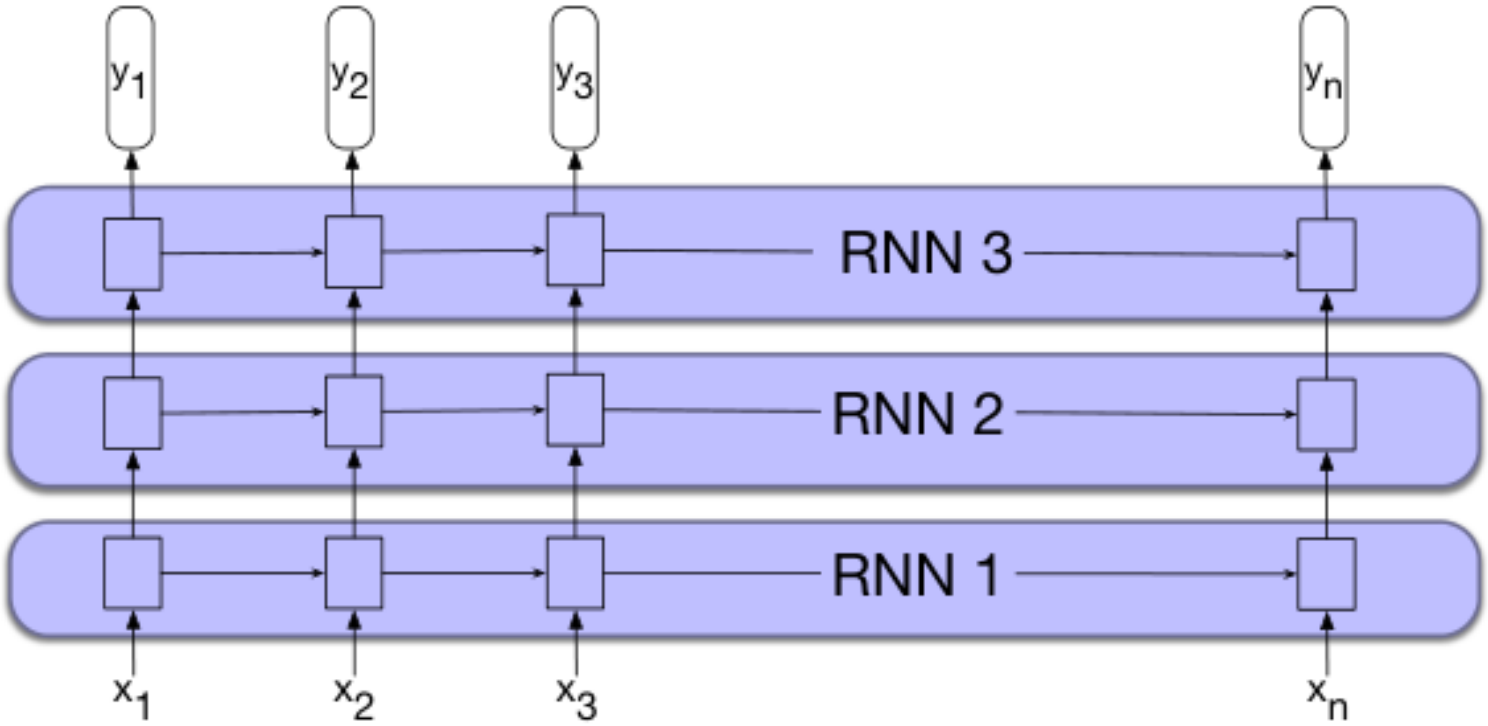
Autoregressive Generation with an RNN Language Model

Figure 8.9



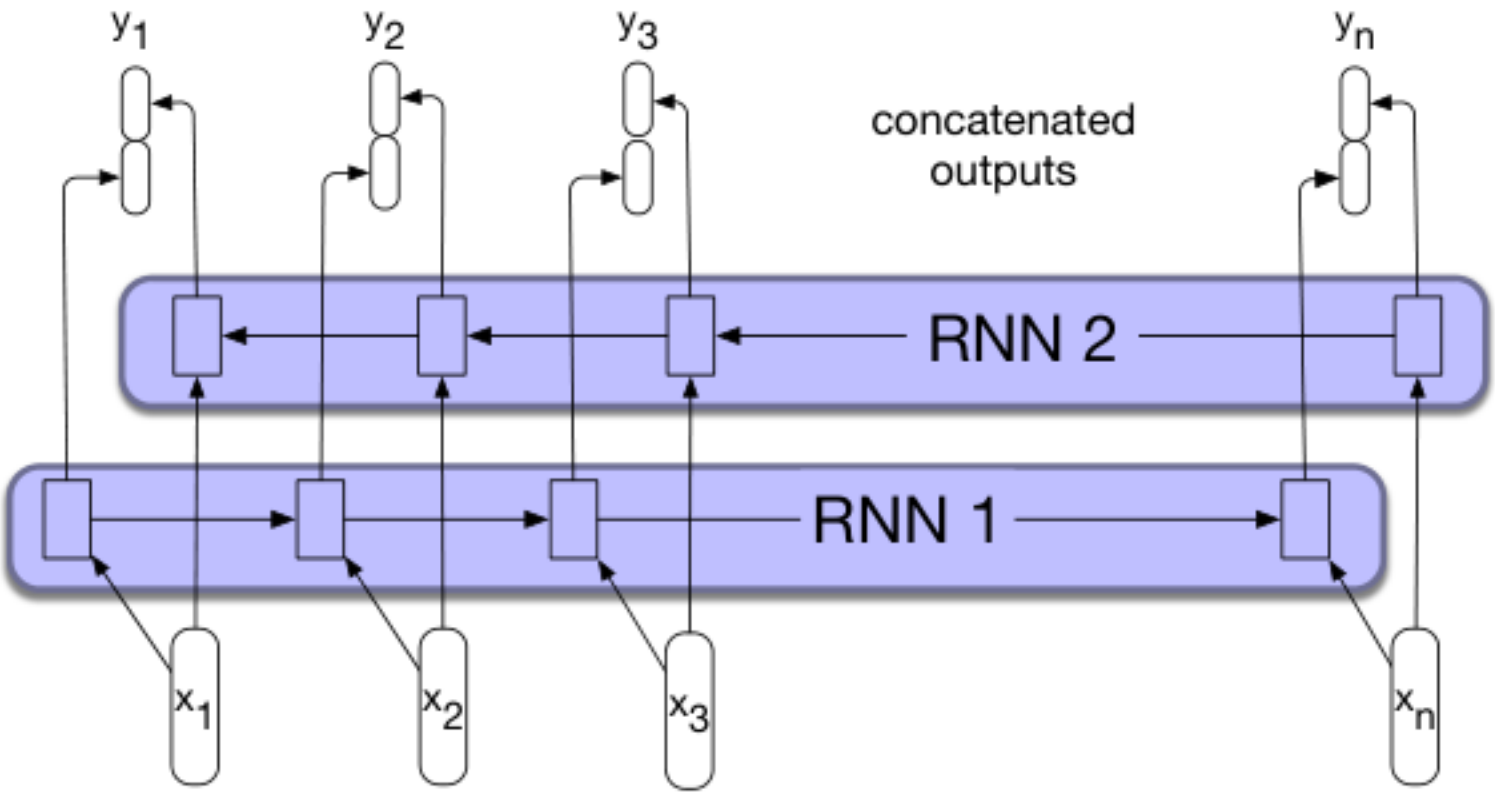
Stacked RNNs

Figure 8.10. Each layer captures a distinct level of abstraction



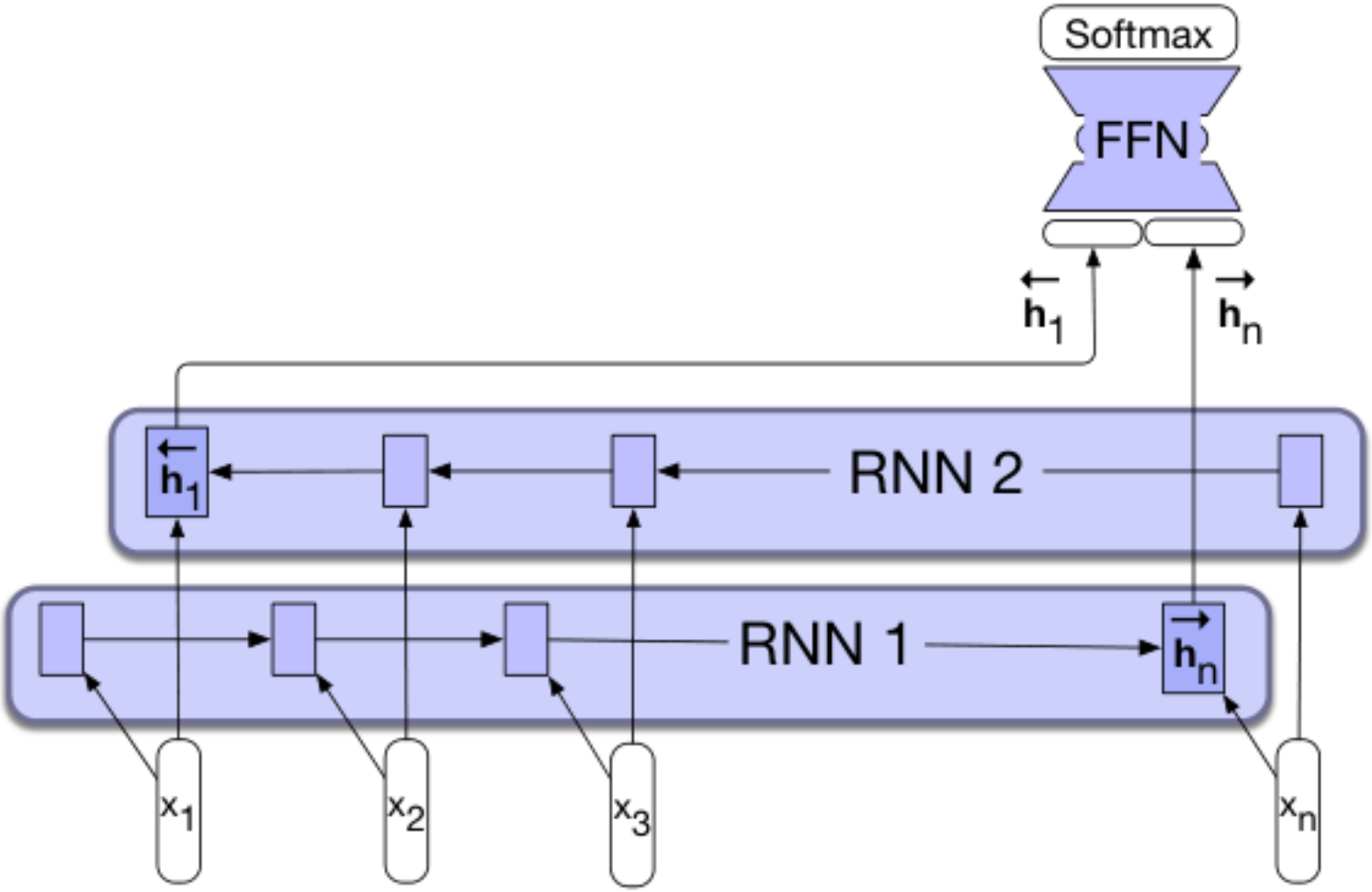
Bidirectional RNN

Figure 8.11. Each output is a concatenation of the forward and backward outputs



Bidirectional RNN for Sequence Classification

Figure 8.12. Combines the last hidden states of forward and backward components

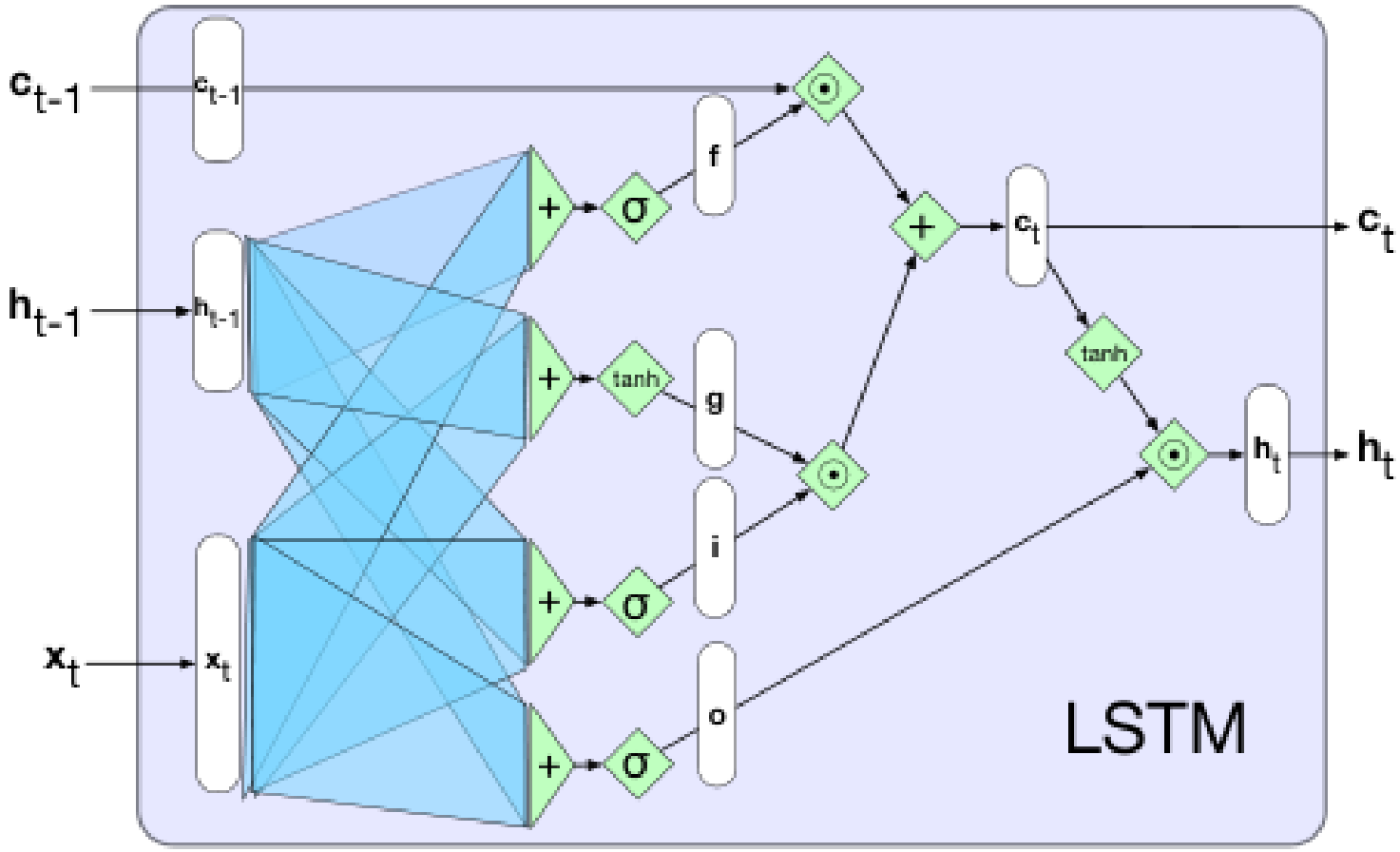


Long Short-Term Memory (LSTM) Unit, Computationally

Figure 8.13.

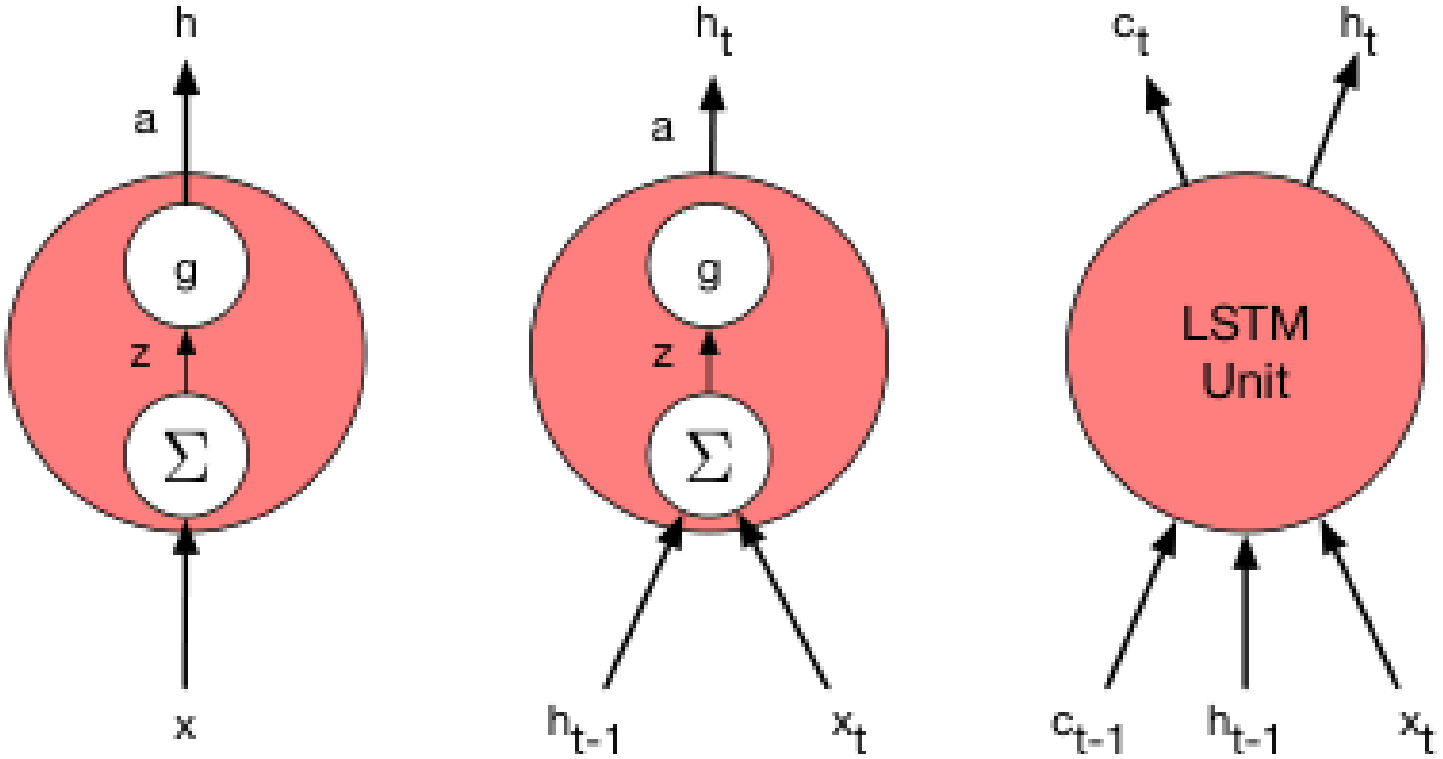
Inputs: current token, previous hidden state, previous context

Outputs: new hidden state, new context



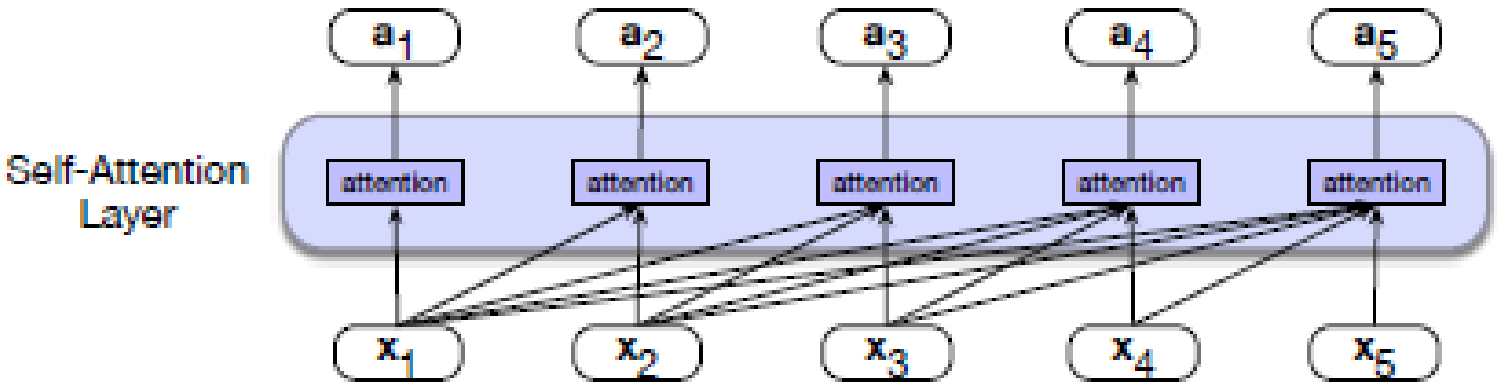
Comparing Neural Units

Figure 8.14. Feedforward neuron; RNN unit; LSTM unit



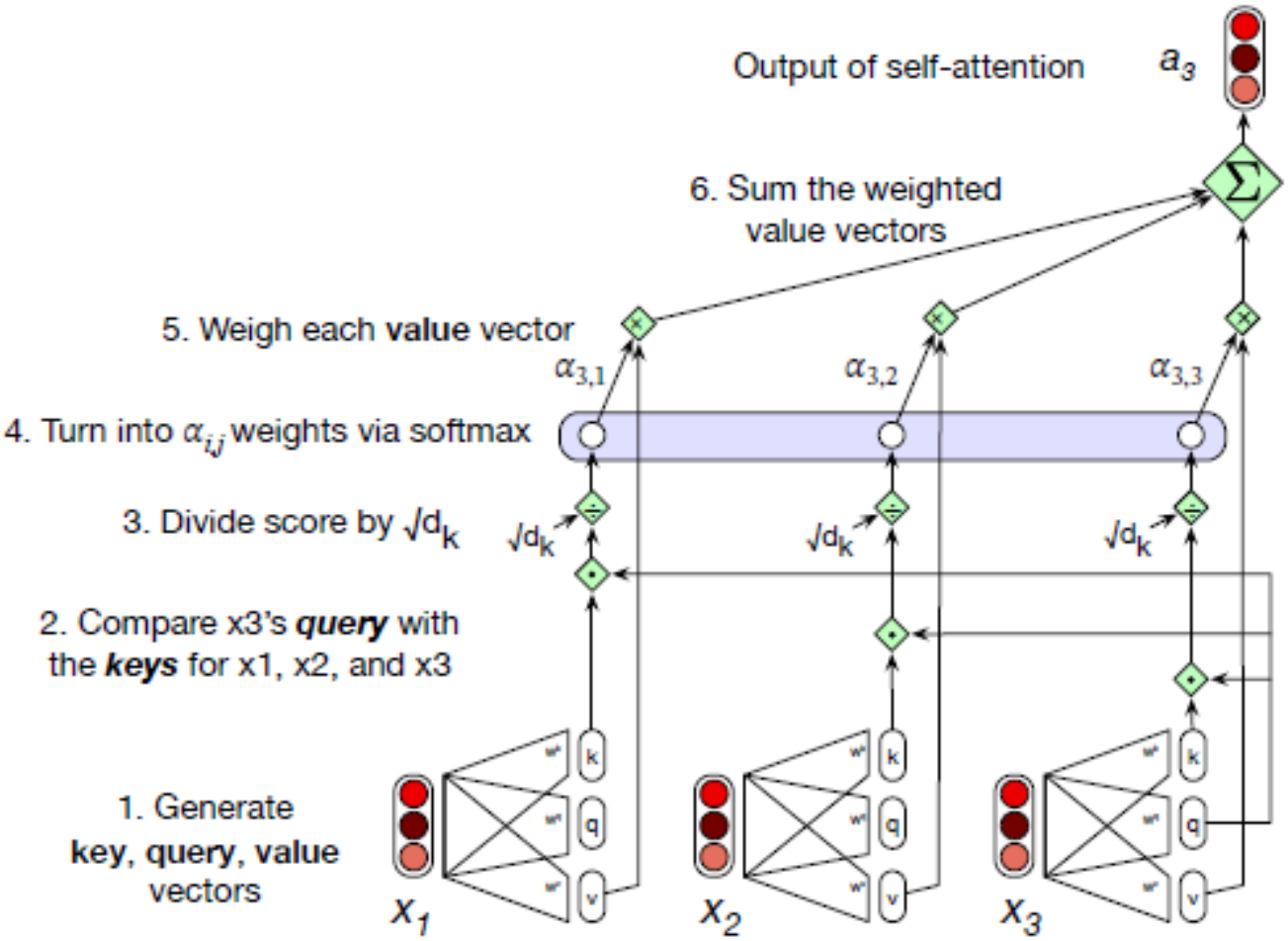
Self-Attention: Information Flow

Figure 9.3. Each unit attends to all previous tokens
Unlike in RNNs, there is no information flow between the units



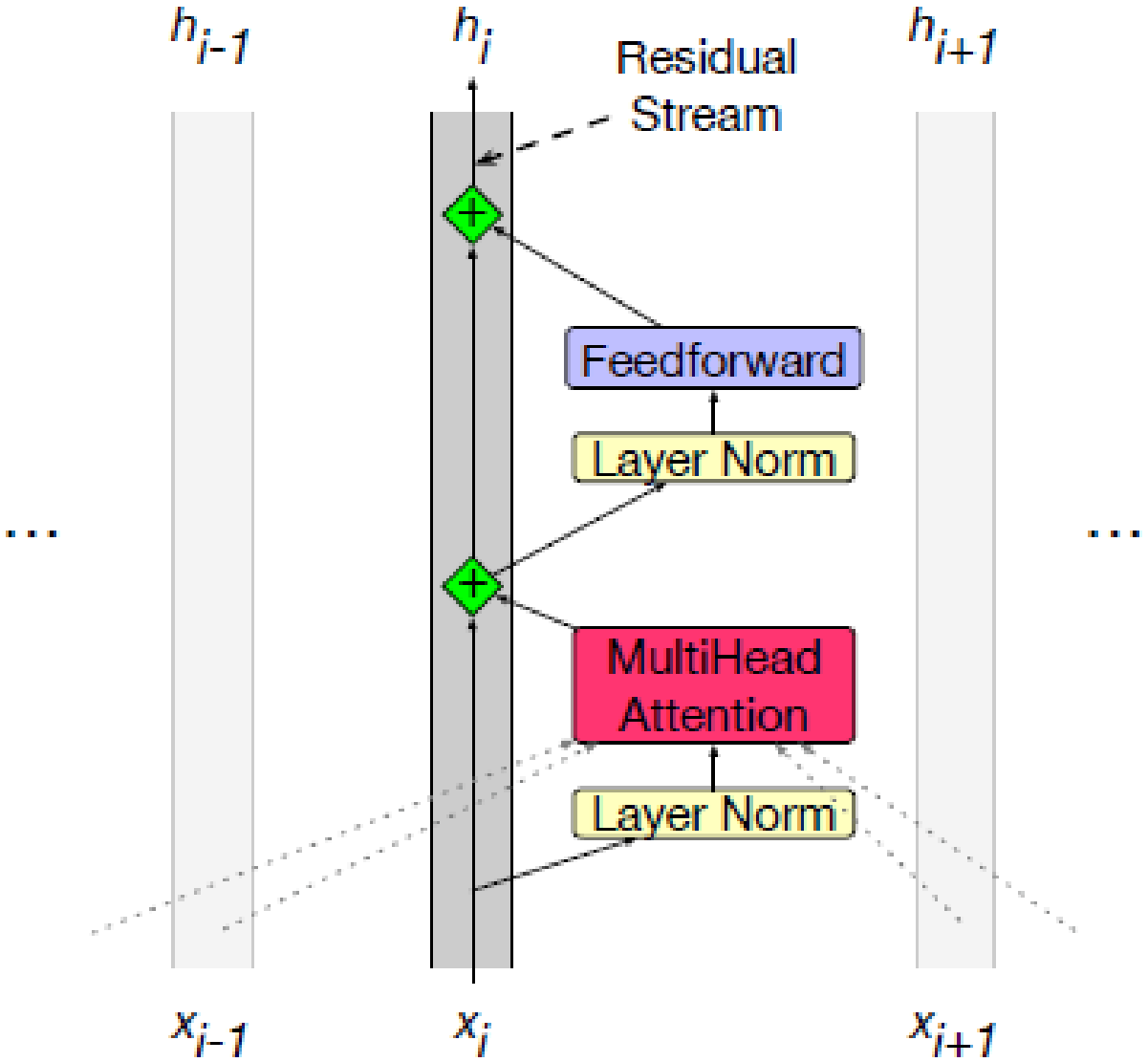
Query-Key-Value Paradigm for Self-Attention

Figure 9.4. Causal (left-to-right) self-attention to calculate the third element



Transformer Block

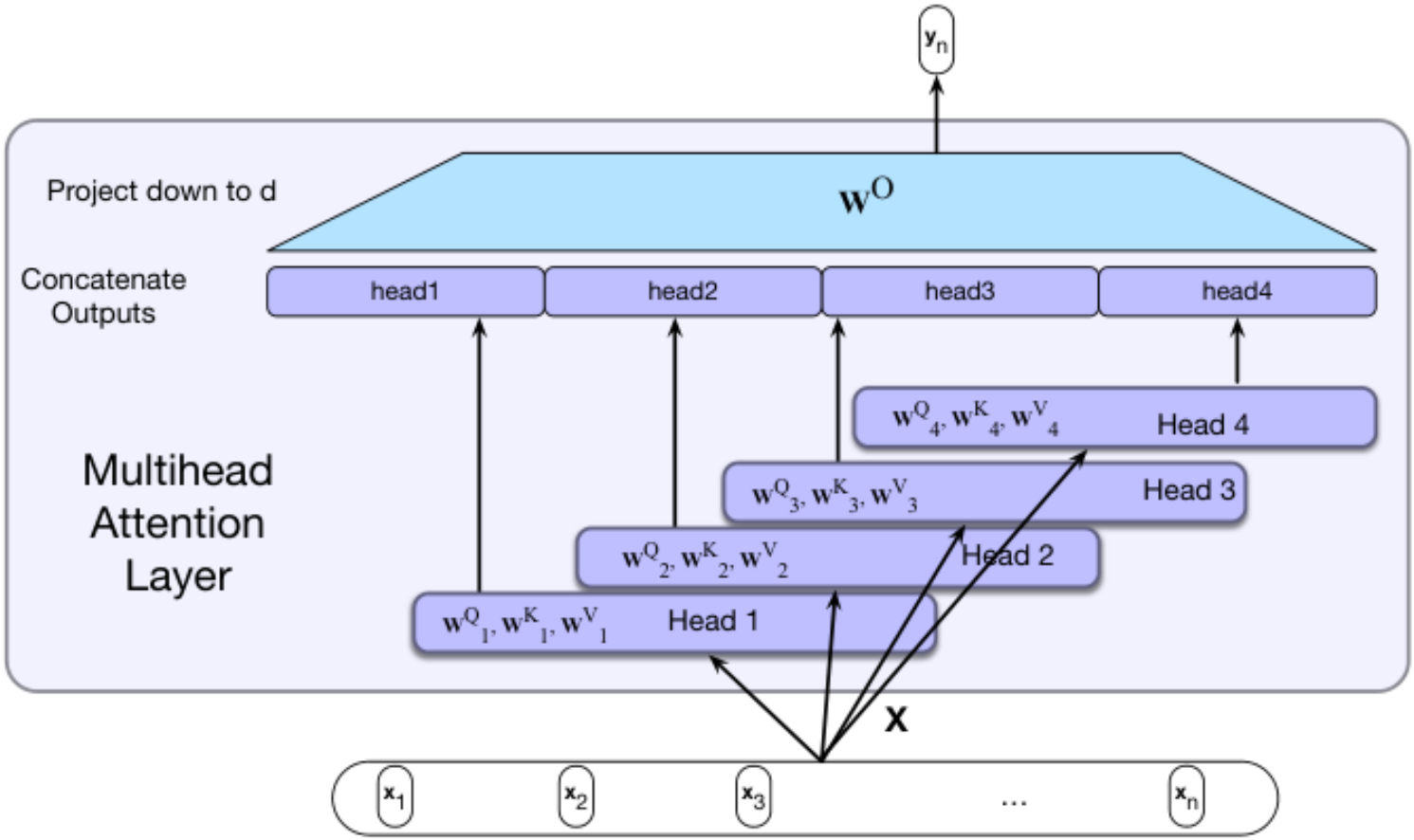
Figure 9.6. Residual connections bypass complex layers to improve learning



Multihead Self-Attention: Capturing Distinct Concerns

Figure 9.19 from a previous edition

Separate heads (separate query-key-value matrices) for syntax, semantics, discourse, . . .



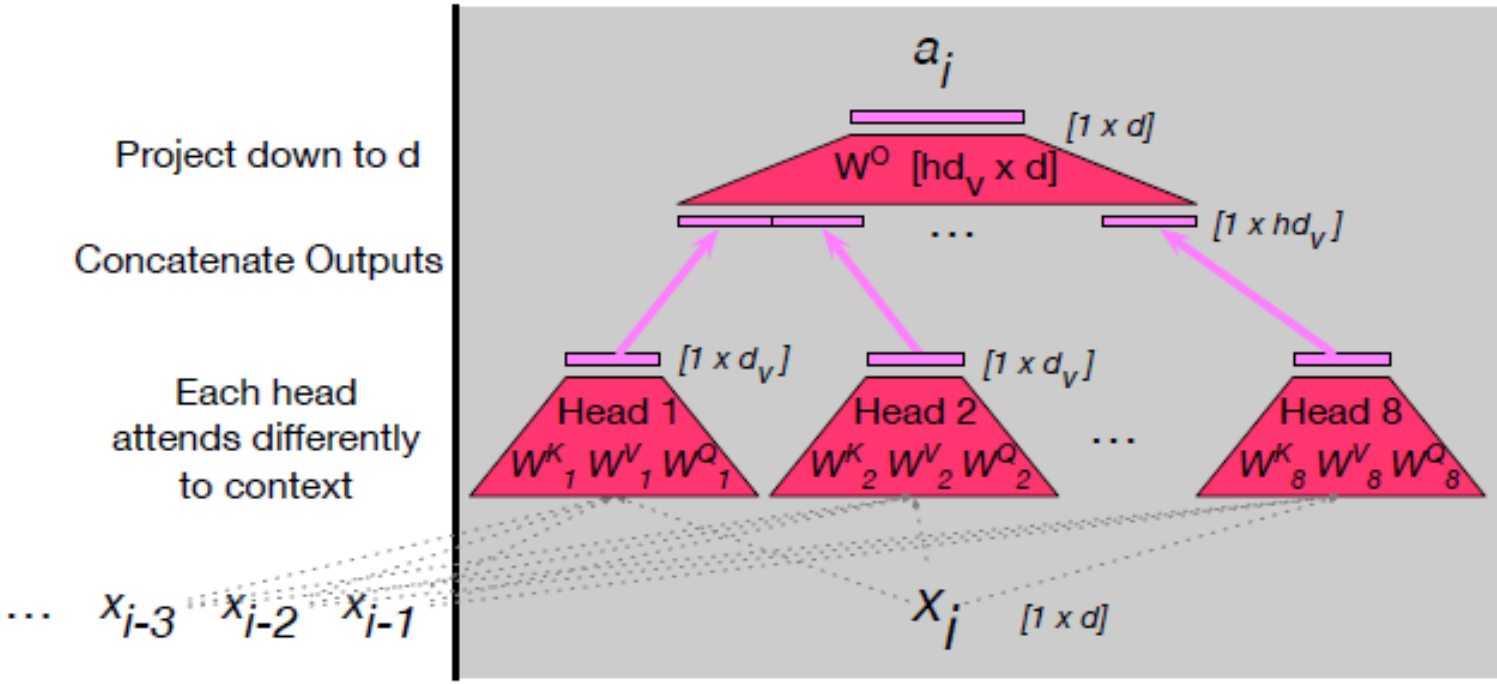
Multihead Self-Attention: Capturing Distinct Concerns

Figure 9.5.

h heads, each with its key, query, value matrices

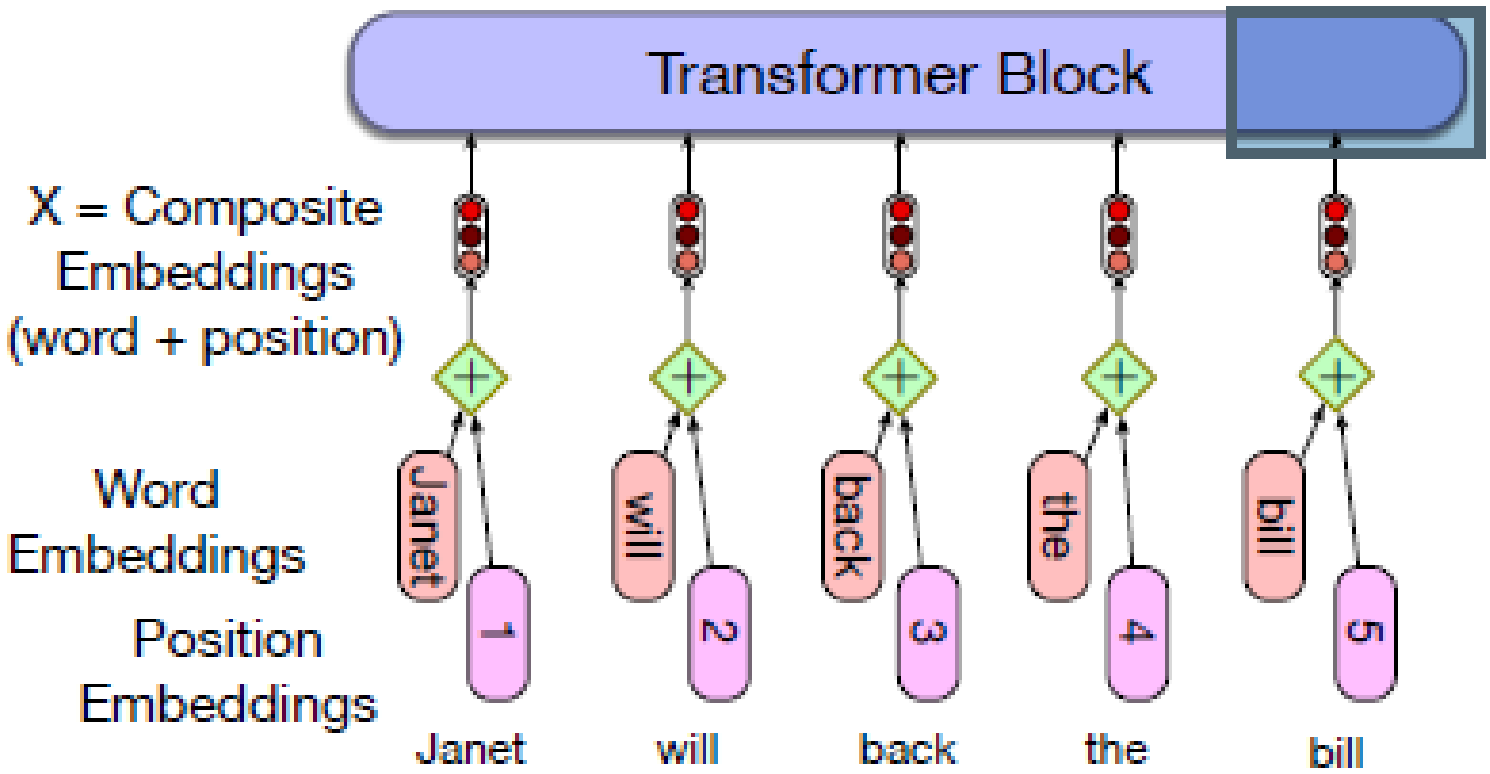
Concatenate value vectors produced by the heads

Project down to the same size as the input



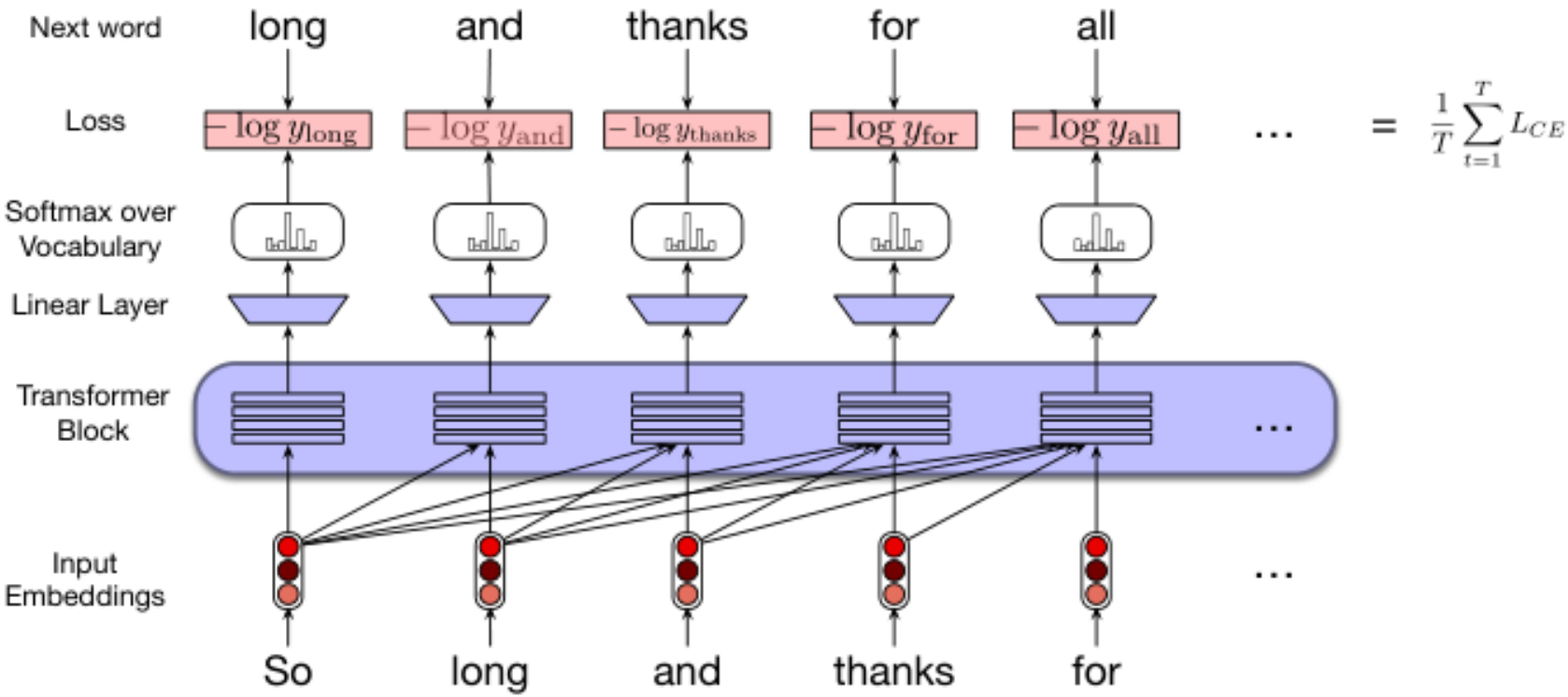
Positional Embeddings to Model Word Order

Figure 9.13. Learn embeddings for each position similarly to token embeddings add position embeddings to embeddings of the respective tokens



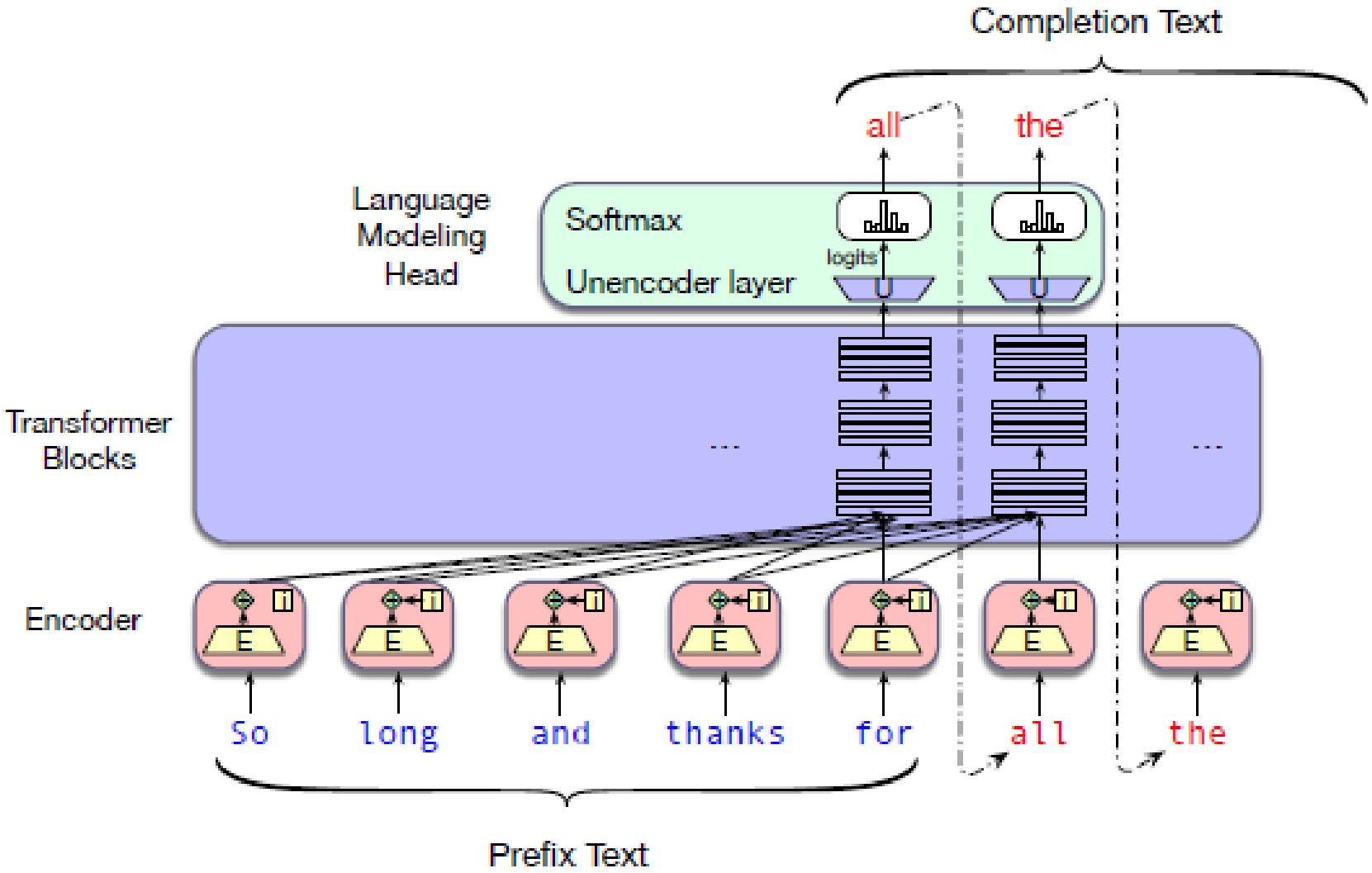
Training a Transformer as a Language Model

Figure 10.4



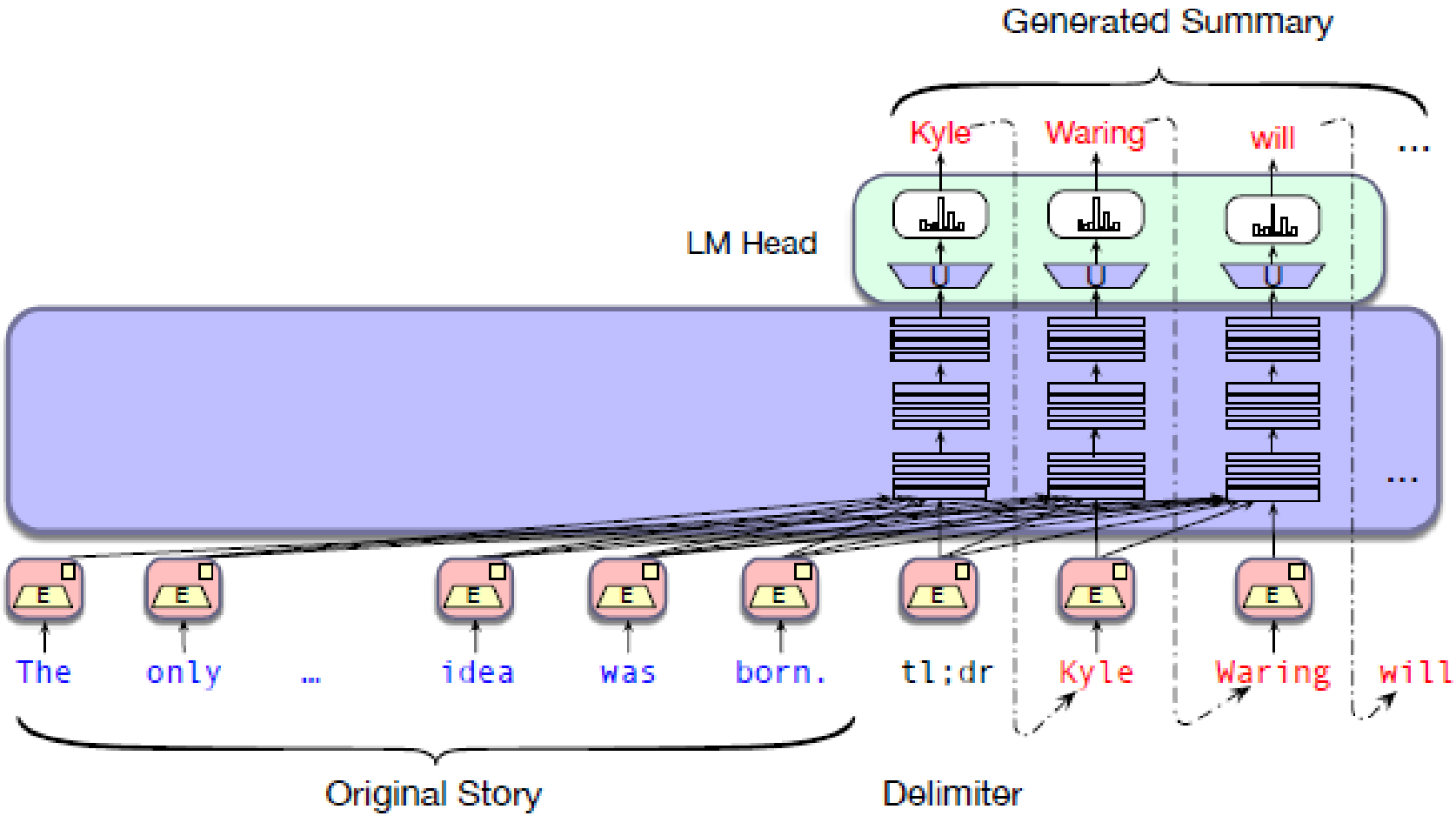
Autoregressive Text Completion with Transformers

Figure 10.1. Similar to what we saw with RNNs



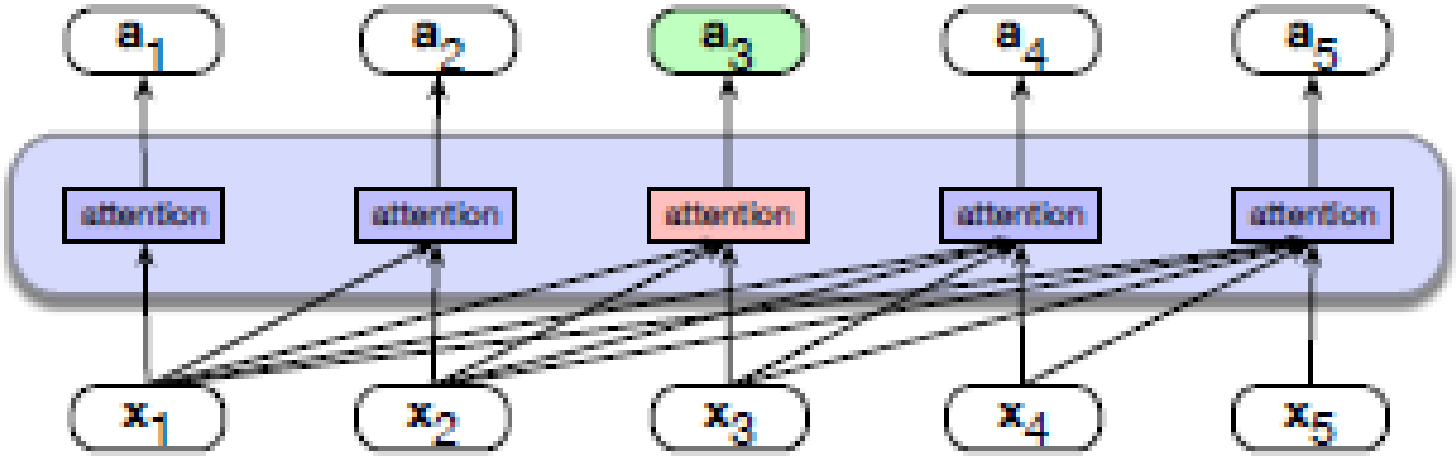
Summarization with Transformers

Figure 10.3. Train with actual story-summary pairs



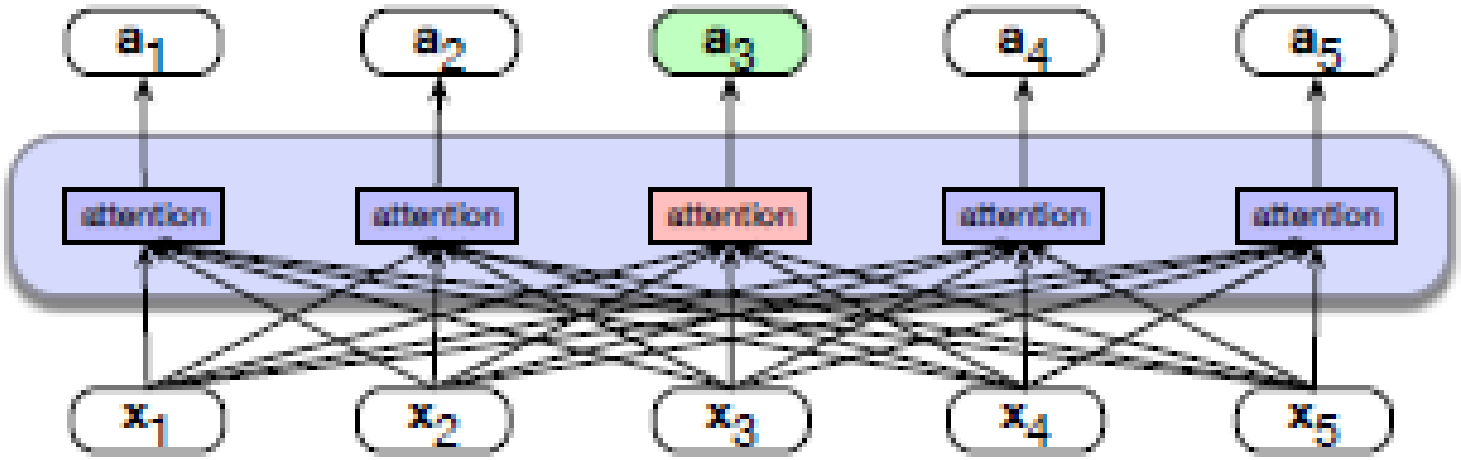
Causal, Backward Looking Transformer

Figure 11.1a. (Same as Figure 9.3)
Causal because it doesn't look at "future" tokens



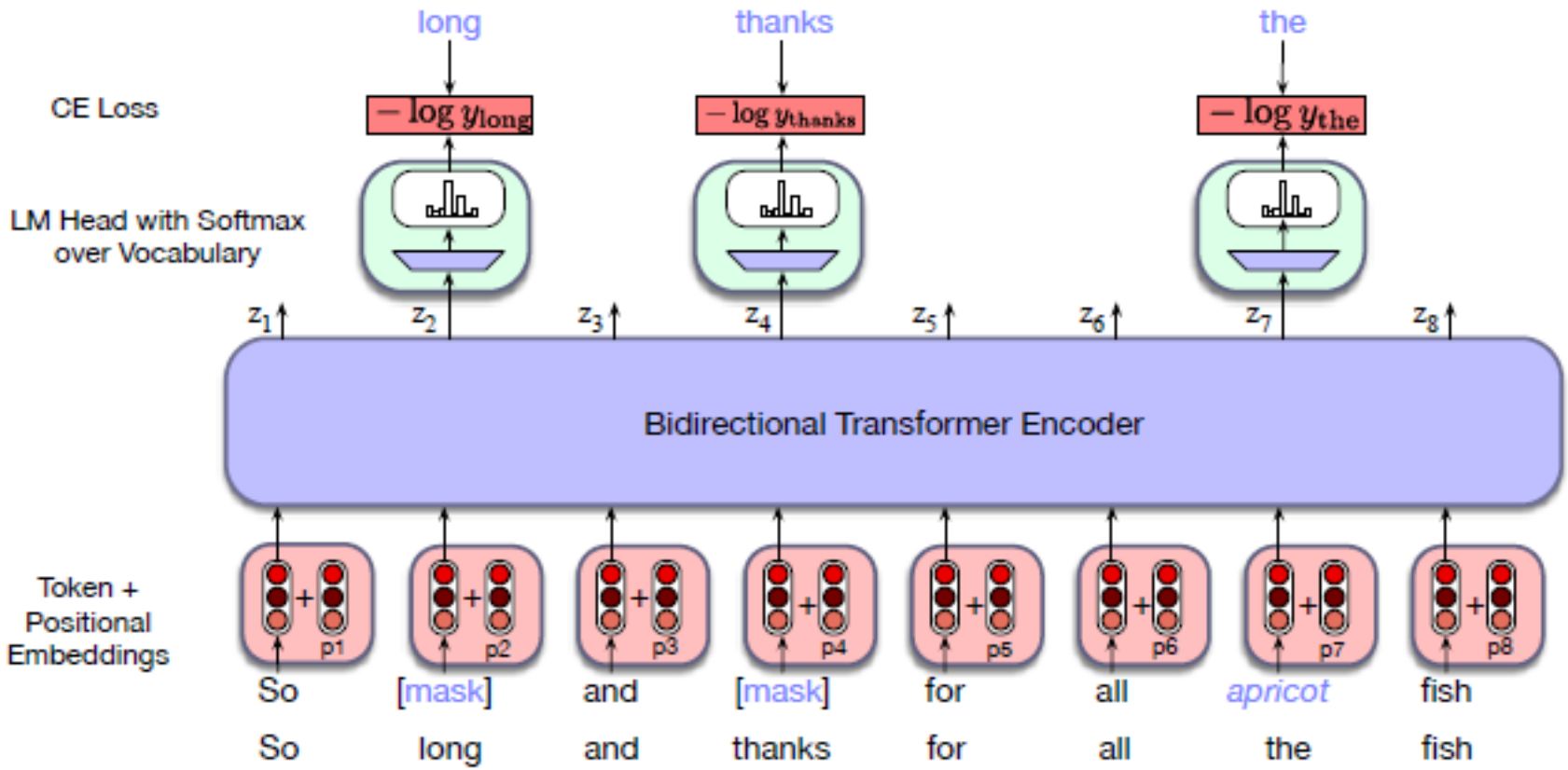
Bidirectional Self-Attention Model

Figure 11.1b. Looks at future (subsequent) tokens



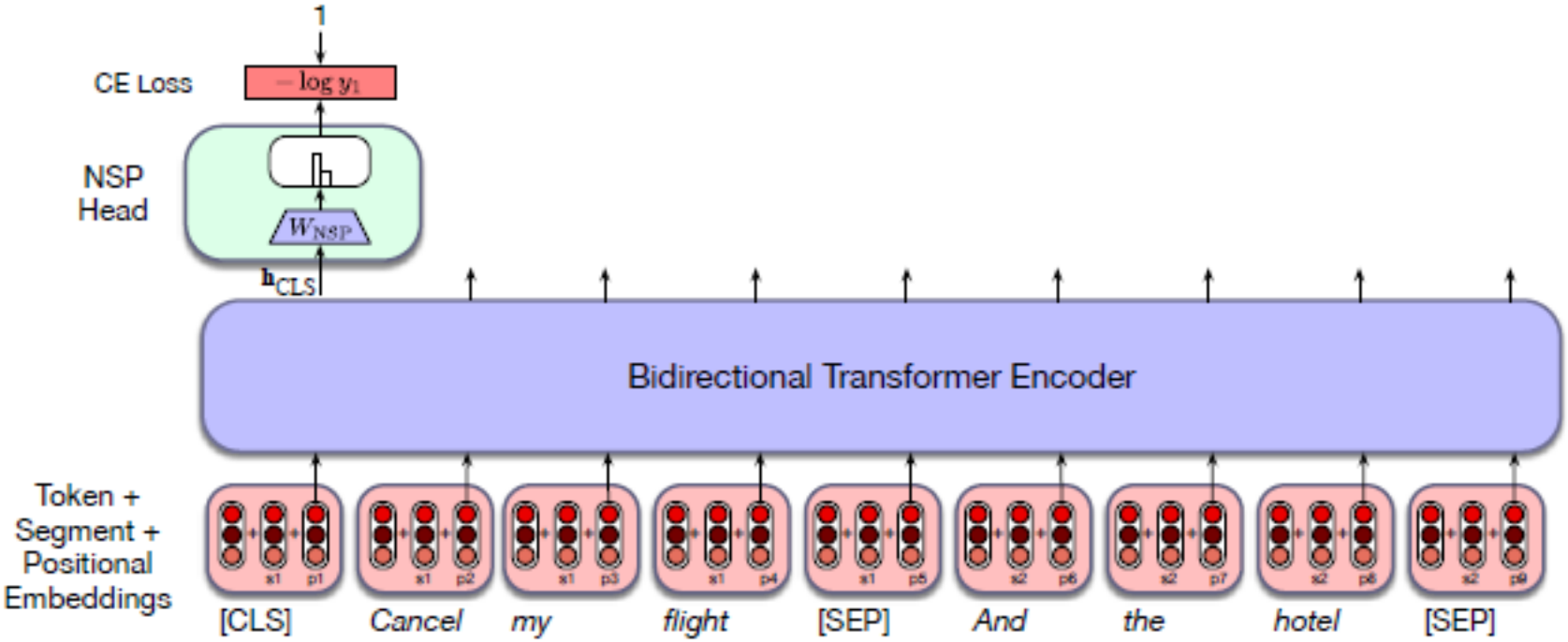
Masked Language Model Training

Figure 11.3. In BERT, 15% tokens are sample, of which 80% become [MASK], 10% become another random token, 10% remain unchanged



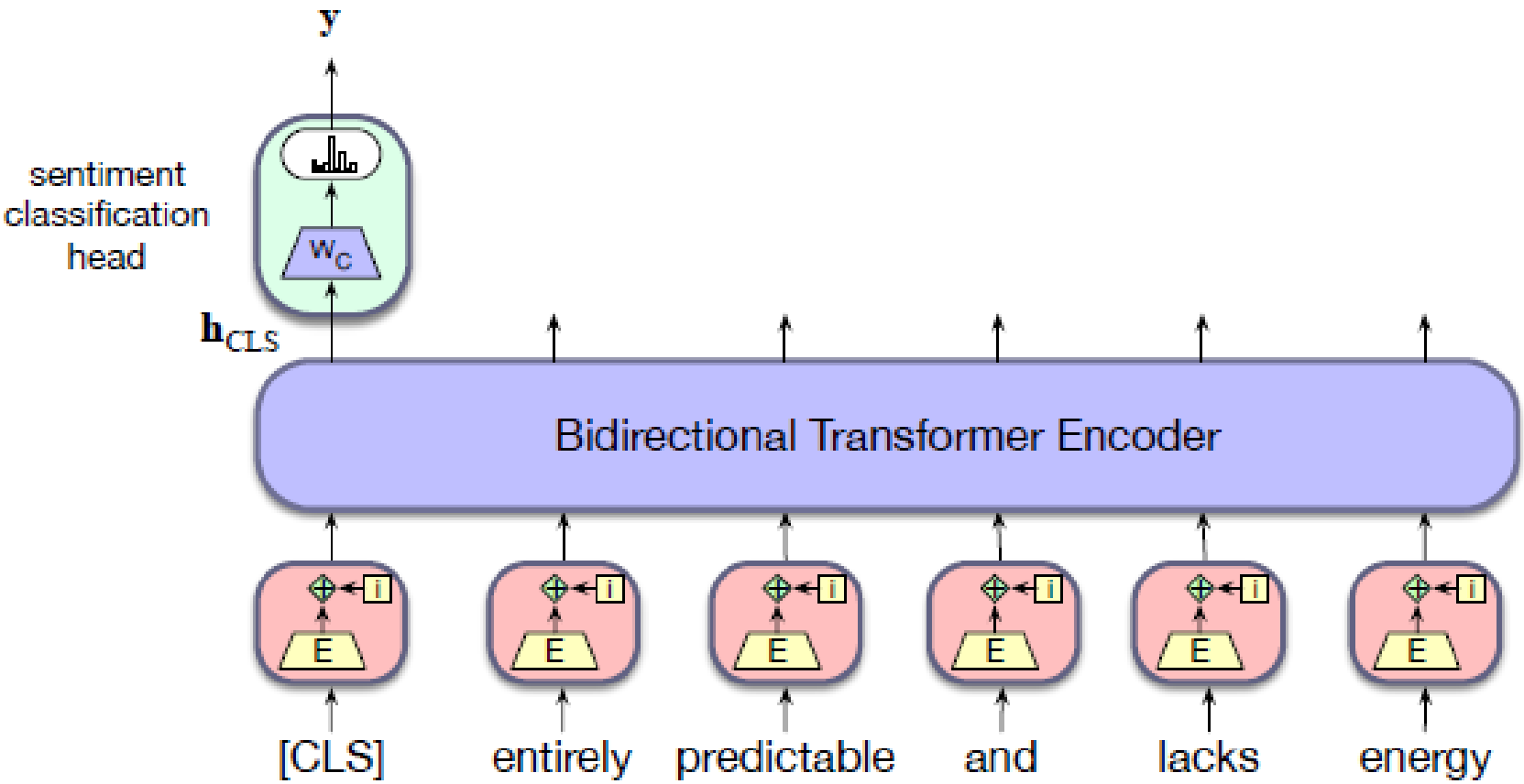
Next Sentence Prediction

Figure 11.4



Sentiment Classification

Figure 11.9



Contextual Embeddings

Figure 11.5. The outputs encode each token's meaning in context
Customary to use the mean of the last four layers