Sequences: seq2seq

Jill-Jênn Vie

Oct 24, 2025

seq2seq (Cho et al. 2014; Sutskever et al. 2014)

Input $x_{1:n}$
Output $y_{1:m}$

$$\begin{cases} h_{1:n} = encoder(x_{1:n}) \in \mathbb{R}^{n \times d} \\ y_{1:m} = decoder(h_{1:n}) \end{cases}$$

	encoder	decoder		citations
2014	lstm	lstm	hochreiter 1997	135143 +491
Still 2014	gru	attention & gru	bahdanau 2014	40780 +63
2017	attention	attention	vaswani 2017	209004 -8902
2018 BERT	attention		devlin 2018	146686 +604
2018 GPT		attention	radford 2018	16133 +111

k nearest neighbors

Who saw this in high school?

Algorithm 1 k nearest neighbors among n in d dimensions

for all i from 1 to n **do**

  Compute distance from $\boldsymbol{x}$ to $\boldsymbol{x}_i$, i.e. $||\boldsymbol{x} - \boldsymbol{x}_i||^2$

Find k nearest neighbors of $\boldsymbol{x}$, argmin distance

Compute average of their y_i, or the majority class

- Complexity to find neighbors of $\boldsymbol{x}$ in X ? $O(nd + kn)$
- For every token in the sequence: $O(mnd + mkn)$

k nearest neighbors

Who saw this in high school?

Algorithm 2 k nearest neighbors among n in d dimensions

---

for all i from 1 to n **do**

  Compute similarity between $\boldsymbol{x}$ and $\boldsymbol{x}_i$, i.e. $\boldsymbol{x}^T\boldsymbol{x}_i$

Find k nearest neighbors of $\boldsymbol{x}$, argmax similarity

Compute average of their y_i, or the majority class

- Complexity to find neighbors of $\boldsymbol{x}$ in X ? $O(nd + kn)$
- For every token in the sequence: $O(mnd + mkn)$

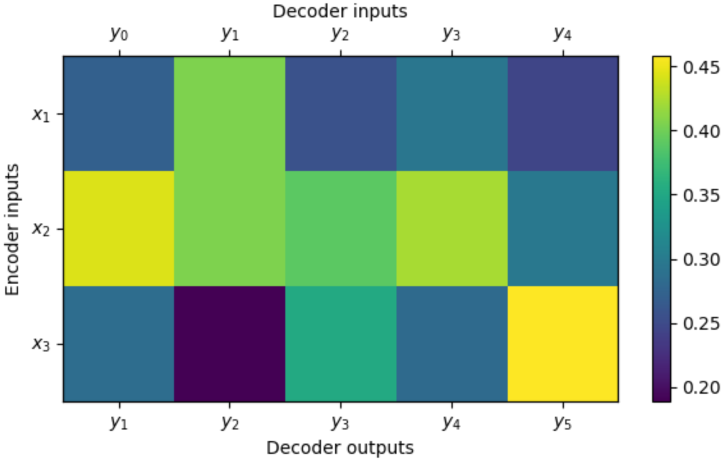
Variant: weighted k nearest neighbors

Similarity $w_i = \boldsymbol{x}_{test}^T \boldsymbol{x}_{train_i} / N_z$ (renormalized to sum to 1)

I predict for $\boldsymbol{x}_{test}$: $\hat{y} = \sum_{i=1}^{n} w_i y_{train_i}$

Attention is $softmax(QK^T)V$
A linear combination of values V with weights corresponding to similarity (e.g. dot product) between queries Q and keys K
Understanding KNN is a first step to understand attention mechanism.

Attention (Bahdanau, Cho, and Bengio 2015)



$$h_{1:n} = encoder(x_{1:n})$$

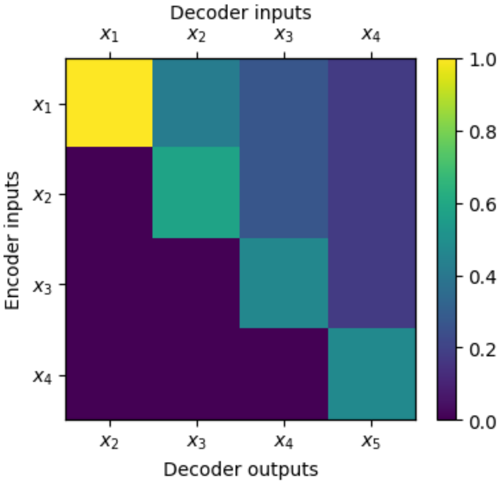
$$\alpha_{ij} = softmax_i(s_j^T h_i)$$

Output y_{j+1} depends on

$$y_j \text{ and } \sum_{i=1}^{n} \alpha_{ij} h_i .$$

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio (2015). "Neural Machine Translation by Jointly Learning to Align and Translate". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. URL: http://arxiv.org/abs/1409.0473

Masked self-attention (autoregressive)



$$q_{1:n}, k_{1:n}, v_{1:n} = Linear(x_{1:n})$$

$$\alpha_{ij} = softmax_i(q_j^T k_i)$$

Output x_{j+1} depends on

$$\sum_{i=1}^{j} \alpha_{ij} v_i.$$

Ashish Vaswani et al. (2017). “Attention is all you need”. In: *Advances in neural information processing systems* 30

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2015). "Neural Machine Translation by Jointly Learning to Align and Translate". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. URL: http://arxiv.org/abs/1409.0473.

Cho, Kyunghyun et al. (2014). "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734.

Devlin, Jacob et al. (2019). "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long short-term memory". In: *Neural computation* 9.8, pp. 1735–1780.

Radford, Alec et al. (2018). "Improving language understanding by generative pre-training". In.

Vaswani, Ashish et al. (2017). "Attention is all you need". In: *Advances in neural information processing systems* 30.