Beam Search

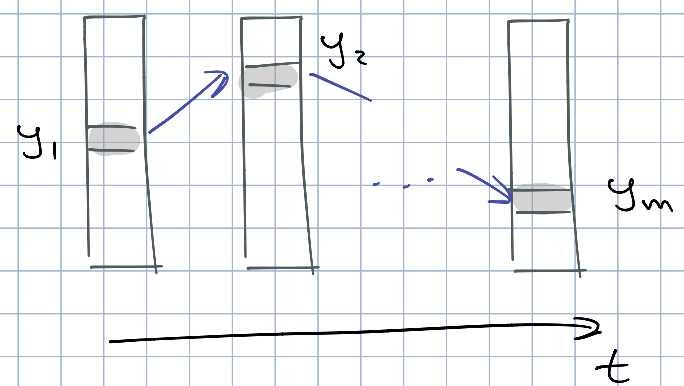
$x_1, x_2, \ldots, x_n \longmapsto h$

Generation process

$$p(y_1 | h)$$
$$p(y_2 | y_1, h)$$
$$\vdots$$

$$p(y_1 y_2 \ldots y_m | h) = p(y_1|h)\, p(y_2|y_1,h) \cdot \ldots \cdot p(y_m | y_1 \ldots y_{m-1} | h)$$

Each step of the way we predict prob dist over words



Therefore $p(y_{1 \ldots m} | h) = \hat{p}_{y_1} \cdot \ldots \cdot \hat{p}_{y_m}$

We want to find $\underset{y_1, \ldots, y_m}{\operatorname{argmax}} P(y_1 \ldots y_m | h)$

At each timestamp t, we can keep fixed size pool of candidates, thy to add new char, and leave only most likely sequences.

Attention

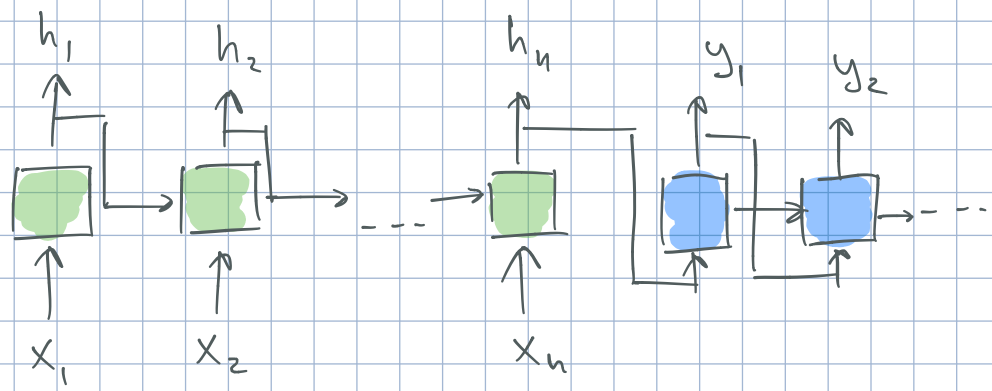


Fig: Encoder - Decoder Architecture

Intuition : Instead of using just last hidden state ($h = h_n$) of encoder as an input to decoder at time t we can use

$$\alpha_1 h_1 + \alpha_2 h_2 + \dots + \alpha_n h_n ,$$

where $0 \le \alpha_i \le 1$ and $\sum \alpha_i = 1$,

For convinience $\alpha_i = \dfrac{\exp(\hat{\alpha}_i)}{\sum_j \exp(\hat{\alpha}_j)}$, where

$\hat{\alpha}_i$ "depends" on position in the output (y_t)

More formaly, $\hat{\alpha}_i = MLP(h_i, h^d_{t-1})$, where h^d_{t-1} – hidden state of decoder at previous timestamp.

Transformer

Self-Attention

$A(q, k, V)$ = attention based repr of a word

Intuition:

Given words seq $x_1, x_2, \dots, x_5$.

Vector representation for x_3 could be just simple emb. OR it could depend on the context $(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n)$

$$A(q, k, V) = \sum_i \frac{\exp(q \cdot k_i)}{\sum_j \exp(q \cdot k_j)} v_i \; ,$$

where

q – query $(W^Q \cdot x_3)$
k_i – key $(W^K \cdot x_3)$
v_i – value $(W^V \cdot x_3)$

Trainable matrices

we do this simulteneously for all words in a sentence and get $A_1, A_2, \dots, A_5$

Remark

q_t – is like a question about x_t, and
k_i – is an answer, therefore
$\langle q_t, k_i \rangle$ – measures how good k_i answers q_t

Since we want weights to add up to 1 we use softmax

$$\alpha_i = \frac{\exp(q_t \cdot k_i)}{\sum_j \exp(q_t \cdot k_j)}$$

And to get final repr of A_t

$$\sum_i \alpha_i v_i$$

Matrix form of Self-Attention

$$A(Q,K,V) = \underbrace{\text{softmax}\left(\frac{QK^T}{\underbrace{\sqrt{d_k}}_{\text{for scaling}}}\right)}_{\text{row-wise softmax}} V$$

$$Q = (q_1, q_2, \dots, q_n)^T \in \mathbb{R}^{n \times d}$$

$$K = (k_1, k_2, \dots, k_n)^T \in \mathbb{R}^{n \times d}$$

$$V = (v_1, v_2, \dots, v_n)^T \in \mathbb{R}^{n \times d}$$

$$A(Q,K,V) = (A_1, A_2, \dots, A_n)^T \in \mathbb{R}^{n \times d}$$

A_i – context dependent vec repr of word x_i

Multi-Head Attention

Recall that previously we were thinking about q's as questions.

If we repeat comp of $A(Q, K, V)$ with different matrices W^Q, W^K, W^V it will give us an opportunity to ask different questions

Let h denote number of heads

$$\text{head}_i = A(W_i^Q Q, W_i^K K, W_i^V \cdot V), \quad i = 1, \dots, h$$

will give us $\mathbb{R}^{n \times d}$ self-attention matrices

Q: How to build final representation for each word?

A:
$$\underbrace{\text{concat}(\text{head}_1, \dots, \text{head}_h)}_{\mathbb{R}^{n \times (d \cdot h)}} \cdot \underbrace{W_{out}}_{\mathbb{R}^{d \cdot h \times l}} \in \mathbb{R}^{n \times l}$$

! This approach allows us to build even reacher context dependent words represent-s.

Transformer Architecture

Remarks

* MLP after M-H Attention is applyed independently for each position (aka row of MHA matrix), hence outputs matrix
* For stacking multiple layers, output matrix from previous layer needs to be transformed into matrices Q, K, V (How?)

(Probably using simple mat mul)