# Causal Attention

① masked attention (AKA) -> special form of self-attention

② Restricts model to only consider previous and current inputs in a sequence when processing any given token

③ contrasts self-attention mechanism, which allows access to the entire input sequence at once

④ When computing attention scores, causal attention mechanism ensures that model only factors in tokens that occur at or before current token in sequence

⑤ To achieve this in GPT like LLMs, for each token processed we mask out future tokens, which come after current token in the input text

"—" represents some input

"X" represents mask

| | Your | Jour, | starts | w/ | one | step |
|---|---|---|---|---|---|---|
| Your | — | — | — | — | — | — |
| Jour. | — | — | — | — | — | — |
| starts | — | — | — | — | — | — |
| w/ | — | — | — | — | — | — |
| one | — | — | — | — | — | — |
| step | — | — | — | — | — | — |

all "—" sum up to 1.0

| | Your | Jour, | starts | w/ | one | step |
|---|---|---|---|---|---|---|
| Your | — | X | X | X | X | X |
| Jour, | — | — | X | X | X | X |
| starts | — | — | — | X | X | X |
| w/ | — | — | — | — | X | X |
| one | — | — | — | — | — | X |
| step | — | — | — | — | — | — |

masked out future tokens for "Your" token

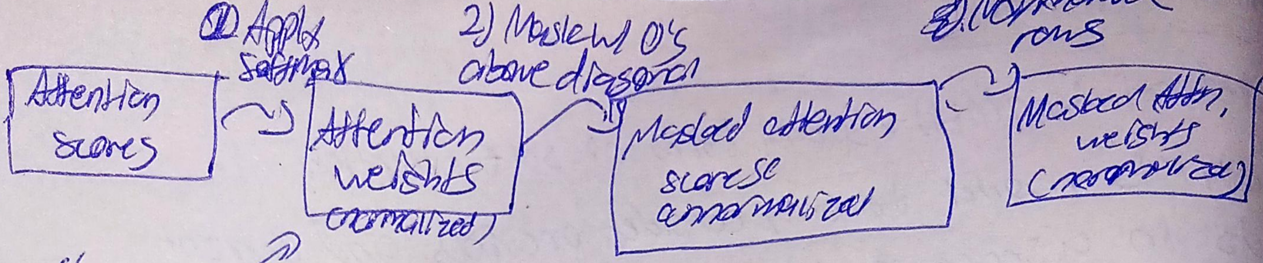
6 (context size assumption)

↳ Mask out the attention weights above the diagonal, and normalize the non masked attention weights, such that attention weights sum up to 1 in each row.

* Applying a causal attention mask *

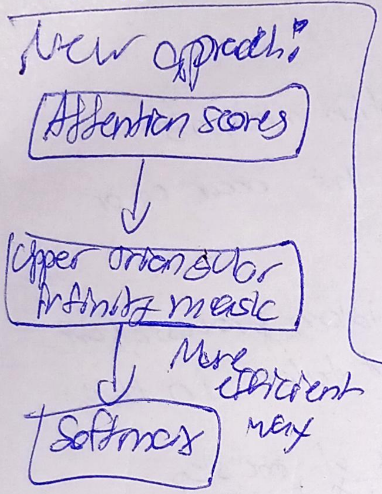
strategy: Get attention weights

↳ zero out elements above the diagonal and normalize the resulting matrix.


① Apply softmax
2) Mask w/ 0's above diagonal
3) Normalize rows
Attention scores
Attention weights (normalized)
Masked attention scores (unnormalized)
Masked Attn. weights (normalized)

"Normalized" means that sum of values in each row is 1

↳ Leads to data leakage as elements have knowledge of the masked bits through the initial softmax normalization (b/c it factors in SUM of all elements in array including the masked bits which haven't been zeroed out yet in next step)



*Masking additional attention weights w/ dropout*

↳ Dropout is a deep learning technique where randomly selected hidden layer units are ignored during training

↳ This prevents overfitting and improves generalization performance.

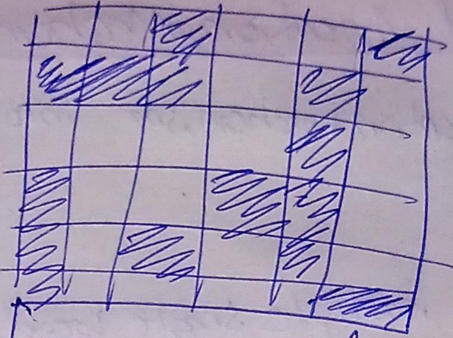
↳ In transformer architecture, including models like GPT, dropout in the attention mechanism is applied in 2 specific areas

↓ after calculating attention scores

↘ After applying attention weights to value vectors

↳ Applying dropout after calculating attention weights is more common, and we consider that.

|  | Yor | Jun. | Starts | w/ | one | step |
|---|---|---|---|---|---|---|
| Yor | – | X | X | X | X | X |
| Jor. | – | – | X | X | X | X |
| Start | – | – | – | X | X | X |
| w/ | – | – | – | – | X | X |
| one | – | – | – | – | – | X |
| step | – | – | – | – | – | – |

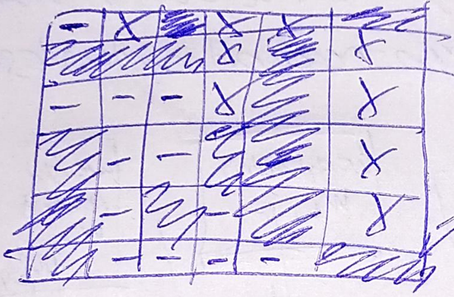


Dropout mask w/ random positions to be dropped

"–" means some value that is useful other "–" in the row to sum to 1

"X" means masked

"▨" means dropped neuron



Dropout mask applied to the attention scores will zero out certain attention scores

* Now in code, implement a Causal Attention class, which incorporates Causal Attention and Dropout modifications into the Self Attention class we implemented earlier