From Boltzmann-Gibbs distribution to numerically stable online softmax on GPU with proofs

Jedrzej Maczan

May 29, 2026

1 The Boltzmann-Gibbs distribution

The precursor of softmax [1] is defined as:

$$\mathrm{p}(i) = \frac{1}{Q} e^{(-\frac{\epsilon_i}{k_B T})} = \frac{e^{(-\frac{\epsilon_i}{k_B T})}}{\sum_{j=1}^{M} e^{(-\frac{\epsilon_j}{k_B T})}} \tag{1}$$

2 Softmax

Softmax [2] is defined as:

$$\begin{aligned} \text{softmax} &: \mathbb{R}^K \to (0,1)^K \\ \text{softmax}(x_i) &= \frac{e^{x_i}}{\sum_{j=1}^{V} e^{x_j}} \end{aligned} \tag{2}$$

3 Safe softmax

Safe softmax [3] [4] is defined as:

$$\text{safe-softmax}(x_i) = \frac{e^{x_i - \max_j x_j}}{\sum_{j=1}^{V} e^{x_j - \max_j x_j}} \tag{3}$$

Theorem 3.1. *Safe softmax is equivalent function to softmax.*

Proof.

$$\begin{aligned}
\text{safe-softmax}(x_i) &= \frac{e^{x_i - \max_j x_j}}{\sum_{j=1}^{V} e^{x_j - \max_j x_j}} \\
&= \frac{e^{x_i} e^{-\max_j x_j}}{\sum_{j=1}^{V} e^{x_j} e^{-\max_j x_j}} \\
&= \frac{e^{x_i} \cancel{e^{-\max_j x_j}}}{\cancel{e^{-\max_j x_j}} \sum_{j=1}^{V} e^{x_j}} \\
&= \frac{e^{x_i}}{\sum_{j=1}^{V} e^{x_j}} \\
&= \text{softmax}(x_i)
\end{aligned} \tag{4}$$

■

4 Online safe softmax

Online safe softmax was introduced and proved by Milakov and Gimelshein [3].
m is a running max, s is a running sum

$$\text{Let } m_i \leftarrow \max(m_{i-1}, x_i), s_i \leftarrow s_{i-1} e^{m_{i-1} - m_i} + e^{x_i - m_i}$$

Theorem 4.1. *Online safe softmax is equivalent to safe softmax*

Proof. Base case ($V = 1$):

$$m_1 \leftarrow max_{k=1}^{1} x_k = x_1$$

$$s_1 = \sum_{j=1}^{1} e^{x_j - m_1} = e^{x_1 - m_1}$$

Inductive step:

$$\begin{aligned}
m_n &\leftarrow max(m_{n-1}, x_n) \\
&= max(max_{k=1}^{n-1} x_k, x_n) \\
&= max_{k=1}^{n} x_k
\end{aligned} \tag{5}$$

$$
\begin{aligned}
s_n &\leftarrow s_{n-1}e^{m_{n-1}-m_n} + e^{x_n-m_n} \\
&= (\sum_{j=1}^{n-1} e^{x_j-m_{n-1}})e^{m_{n-1}-m_n} + e^{x_n-m_n} \\
&= (\sum_{j=1}^{n-1} e^{x_j-m_n}) + e^{x_n-m_n} \\
&= \sum_{j=1}^{n} e^{x_j-m_n}
\end{aligned} \tag{6}
$$

■

5 Generalized online softmax version

Generalized version has been defined by Milakov and Gimelshein as:

$$\begin{bmatrix} m_V \\ s_V \end{bmatrix} = \begin{bmatrix} x_1 \\ 1 \end{bmatrix} \oplus \begin{bmatrix} x_2 \\ 1 \end{bmatrix} \oplus ... \oplus \begin{bmatrix} x_V \\ 1 \end{bmatrix}$$

$$m_V, s_V, x_i \in \mathbb{R}$$

$$\oplus : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}^2$$

$$\begin{bmatrix} m_i \\ s_i \end{bmatrix} \oplus \begin{bmatrix} m_j \\ s_j \end{bmatrix} = \begin{bmatrix} \max(m_i, m_j) \\ s_i e^{m_i - \max(m_i,m_j)} + s_j e^{m_j - \max(m_i,m_j)} \end{bmatrix}$$

Theorem 5.1. *Generalized online safe softmax is commutative.*

Proof.

$$A \oplus B = B \oplus A$$

$$\max(m_i, m_j) = \max(m_j, m_i)$$

$$A \oplus B = s_i e^{m_i - \max(m_i,m_j)} + s_j e^{m_j - \max(m_i,m_j)}$$
$$B \oplus A = s_j e^{m_j - \max(m_i,m_j)} + s_i e^{m_i - \max(m_i,m_j)}$$

Equal by commutativity of addition.

■

Theorem 5.2. *Generalized online safe softmax is associative.*

Proof.

$$(A \oplus B) \oplus C = A \oplus (B \oplus C)$$

A $\oplus$ B:

$$m_{AB} = \max(m_A, m_B), s_{AB} = s_A e^{m_A - m_{AB}} + s_B e^{m_B - m_{AB}}$$

(A $\oplus$ B) $\oplus$ C:

$$m_{(AB)C} = \max(m_{AB}, m_C)$$

$$\begin{aligned} s_{(AB)C} &= s_{AB} e^{m_{AB} - m_{(AB)C}} + s_C e^{m_C - m_{(AB)C}} \\ &= (s_A e^{m_A - m_{AB}} + s_B e^{m_B - m_{AB}}) e^{m_{AB} - m_{(AB)C}} + s_C e^{m_C - m_{(AB)C}} \\ &= s_A e^{m_A - m_{(AB)C}} + s_B e^{m_B - m_{(AB)C}} + s_C e^{m_C - m_{(AB)C}} \end{aligned} \tag{7}$$

A $\oplus$ (B $\oplus$ C):

$$m_{A(BC)} = \max(m_A, m_{BC})$$

$$\begin{aligned} s_{A(BC)} &= s_A e^{m_A - m_{A(BC)}} + s_{BC} e^{m_{BC} - m_{A(BC)}} \\ &= s_A e^{m_A - m_{A(BC)}} + (s_B e^{m_B - m_{BC}} + s_C e^{m_C - m_{A(BC)}}) e^{m_{BC} - m_{A(BC)}} \\ &= s_A e^{m_A - m_{A(BC)}} + s_B e^{m_B - m_{A(BC)}} + s_C e^{m_C - m_{A(BC)}} \end{aligned} \tag{8}$$

Now let's see if $m_{A(BC)} = m_{(AB)C}$. Remember that max is associative. So:

$$m_{(AB)C} = \max(m_{AB}, m_C) = \max(\max(m_A, m_B), m_C) = \max(m_A, m_B, m_C)$$

$$m_{A(BC)} = \max(m_A, m_{BC}) = \max(m_A, \max(m_B, m_C)) = \max(m_A, m_B, m_C)$$

$$m_{A(BC)} = m_{(AB)C}$$

Since above is true, let's substitute the former in the equation for $s_{A(BC)}$:

$$s_{A(BC)} = s_A e^{m_A - m_{(AB)C}} + s_B e^{m_B - m_{(AB)C}} + s_C e^{m_C - m_{(AB)C}} = s_{AB(C)} \tag{9}$$

■

This version of softmax can be written down as CUDA kernel and run in SIMT on GPU, like here.

References

[1] Ludwig Boltzmann. *Studien über das Gleichgewicht der lebendigen Kraft zwischen bewegten materiellen Punkten*. Kk Hof-und Staatsdruckerei, 1868.

[2] John S Bridle. "Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition". In: *Neurocomputing: Algorithms, architectures and applications*. Springer, 1990, pp. 227–236.

[3] Maxim Milakov and Natalia Gimelshein. *Online normalizer calculation for softmax*. 2018. arXiv: 1805.02867 [cs.LG]. URL: https://arxiv.org/pdf/1805.02867.

[4] Zihao Ye. *From Online Softmax to FlashAttention*. 2023. URL: https://courses.cs.washington.edu/courses/cse599m/23sp/notes/flashattn.pdf.