Log-Sum-Exp and Gumbel Intersection

Motivation

The "smooth maximum" puzzle: What if you want to take the maximum of two numbers, but you need the result to be **smooth** (differentiable) for optimization? The hard maximum $\max(x, y)$ has a sharp corner—it's not differentiable at $x = y$.

Enter log-sum-exp: $\beta \log\left(e^{\frac{x}{\beta}} + e^{\frac{y}{\beta}}\right)$. This is a smooth approximation to the maximum that becomes exact as $\beta \to 0$. It's like replacing a sharp corner with a smooth curve that gets sharper as you zoom in.

When computing box intersections, we need to find the maximum of two Gumbel-distributed coordinates. The remarkable fact—and here's the beautiful part—is that the location parameter of this maximum is given by the log-sum-exp function. It's not just a convenient approximation; it's the **exact** answer when working with Gumbel distributions.

The "magic trick": The log-sum-exp function appears naturally in the Gumbel-Max trick: if we add Gumbel noise to deterministic values and take the maximum, the resulting distribution's location parameter is the log-sum-exp of the original values. This provides both a theoretical foundation and a practical computational tool for box intersection operations. It's as if Gumbel distributions were designed to produce this elegant result.

The temperature parameter β controls the "softness" of the maximum: as $\beta \to 0$, we recover the hard maximum (sharp corner); as $\beta \to \infty$, we approach the arithmetic mean (completely smooth). This flexibility allows us to interpolate between deterministic and probabilistic behavior—like a dial that controls how "hard" or "soft" our maximum operation is.

The Gumbel-Max trick: This result is the foundation of the Gumbel-Max trick, a technique widely used in machine learning for sampling from categorical distributions. If we have k categories with log-probabilities $\log p_1, \ldots, \log p_k$, we can sample by adding independent Gumbel noise to each and taking the argmax. The resulting distribution is exactly the categorical distribution with probabilities $p_1, \ldots, p_k$. This trick connects Gumbel distributions to discrete sampling and provides a differentiable relaxation (Gumbel-Softmax) for gradient-based optimization. In our context, we're using the same mathematical structure: adding Gumbel noise and taking maxima naturally produces log-sum-exp in the location parameter.

Connection to max-stability: The Gumbel-Max trick relies on the max-stability property (see the Gumbel Max-Stability document): if we add Gumbel noise to log-probabilities and take the maximum, the result remains Gumbel-distributed. This is why the trick works—the maximum of Gumbel-distributed values (log-probabilities plus Gumbel noise) is itself Gumbel-distributed, with location parameter given by log-sum-exp. The Gumbel-Softmax distribution (Jang et al., 2016; Maddison et al., 2016) replaces the non-differentiable argmax with a differentiable softmax, controlled by temperature β. As $\beta \to 0$, Gumbel-Softmax approaches the hard categorical distribution; as $\beta \to \infty$, it approaches uniform. This temperature-controlled interpolation is exactly what we use in box embeddings: β controls the "softness" of box boundaries, interpolating between hard boxes ($\beta \to 0$) and very soft boxes ($\beta \to \infty$).

Historical context: The Gumbel-Max trick has deep roots in statistics and social sciences. The connection between Gumbel distributions and log-sum-exp was known in social sciences before its application to machine learning. The trick is also related to the reparameterization trick in variational inference: by separating deterministic parameters from stochastic noise, we can compute

gradients through the sampling process. The Gumbel-Softmax distribution (Jang et al., 2016; Maddison et al., 2016) extends this by replacing the non-differentiable argmax with a differentiable softmax, controlled by a temperature parameter. This temperature parameter allows interpolation between discrete (low temperature) and continuous (high temperature) behavior, enabling effective gradient-based optimization of discrete latent variables.

Definition

The **log-sum-exp function** with temperature $\beta > 0$ is:

$$\text{lse}_{\beta(x,y)} = \beta \log\left(e^{\frac{x}{\beta}} + e^{\frac{y}{\beta}}\right)$$

We denote this as $\text{lse}_{\beta(x,y)}$ to emphasize the temperature parameter β, which controls the "softness" of the operation. For Gumbel boxes, intersection coordinates are computed using log-sum-exp to determine the location parameters of the resulting Gumbel distributions.

Statement

Theorem (Gumbel-Max Property). If $G_1, G_2 \sim \text{Gumbel}(0, \beta)$ are independent **centered** Gumbel random variables (location parameter 0), then:

$$\max(x + G_1, y + G_2) \sim \text{Gumbel}\left(\text{lse}_{\beta(x,y)}, \beta\right)$$

The location parameter of the maximum is the log-sum-exp of the input locations x and y, while the scale parameter β remains unchanged. This is a manifestation of max-stability (see the Gumbel Max-Stability document): the maximum of Gumbel-distributed variables remains Gumbel-distributed, with the location parameter determined by log-sum-exp.

Proof

The CDF of $\max(x + G_1, y + G_2)$ is:

$$P(\max(x + G_1, y + G_2) \leq z) = P(x + G_1 \leq z \wedge y + G_2 \leq z)$$

Since G_1 and G_2 are independent:

$$= P(G_1 \leq z - x) * P(G_2 \leq z - y)$$

For $\text{Gumbel}(0, \beta)$, the CDF is $F(z) = e^{-e^{-\frac{z}{\beta}}}$, so:

$$= e^{-e^{-\frac{z-x}{\beta}}} * e^{-e^{-\frac{z-y}{\beta}}} = e^{-\left(e^{-\frac{z-x}{\beta}} + e^{-\frac{z-y}{\beta}}\right)}$$

Factoring out $e^{-\frac{z}{\beta}}$ from the sum:

$$= e^{-e^{-\frac{z}{\beta}}\left(e^{\frac{x}{\beta}} + e^{\frac{y}{\beta}}\right)}$$

Now we manipulate the exponent to match the Gumbel CDF form. We want to write this as $e^{-e^{-\frac{z-\mu}{\beta}}}$ for some location parameter μ.

To do this, we factor the exponent:

$$e^{-\frac{z}{\beta}}\left(e^{\frac{x}{\beta}} + e^{\frac{y}{\beta}}\right) = e^{-\frac{z}{\beta} + \ln\left(e^{\frac{x}{\beta}} + e^{\frac{y}{\beta}}\right)} = e^{-\frac{z - \beta \ln\left(e^{\frac{x}{\beta}} + e^{\frac{y}{\beta}}\right)}{\beta}}$$

This follows from the identity: $e^a * b = e^{a + \ln b}$ when $b > 0$.

Therefore:

$$= e^{-e^{-\frac{z-\beta \ln\left(e^{\frac{x}{\beta}}+e^{\frac{y}{\beta}}\right)}{\beta}}}$$

This is the CDF of $\text{Gumbel}\left(\beta \ln\left(e^{\frac{x}{\beta}} + e^{\frac{y}{\beta}}\right), \beta\right)$. Since $\beta \ln\left(e^{\frac{x}{\beta}} + e^{\frac{y}{\beta}}\right) = \text{lse}_{\beta(x,y)}$, we have:

$$\max(x + G_1, y + G_2) \sim \text{Gumbel}\left(\text{lse}_{\beta(x,y)}, \beta\right)$$

Connection to max-stability: This result is a manifestation of max-stability (see the Gumbel Max-Stability document). The maximum of Gumbel-distributed variables remains Gumbel-distributed, with the location parameter determined by log-sum-exp. This is why log-sum-exp appears naturally in Gumbel box intersection operations: it's the exact location parameter of the maximum, not just an approximation.

Why log-sum-exp appears: This result emerges naturally from the structure of Gumbel distributions. The Gumbel CDF has the form $e^{-e^{-\frac{x-\mu}{\beta}}}$, and when we multiply two such CDFs (for the maximum), the exponential structure leads to a sum of exponentials in the exponent. Taking the logarithm of this sum (after appropriate scaling) gives the log-sum-exp function. This is why log-sum-exp is the "natural" operation for Gumbel distributions, just as addition is natural for normal distributions.

The appearance of log-sum-exp is not arbitrary—it's a consequence of the exponential family structure of Gumbel distributions. When we compute the CDF of the maximum $P(\max(x + G_1, y + G_2) \leq z)$, we get $P(x + G_1 \leq z) * P(y + G_2 \leq z) = e^{-e^{-\frac{z-x}{\beta}}} * e^{-e^{-\frac{z-y}{\beta}}}$. The product of exponentials becomes a sum in the exponent: $e^{-\left(e^{-\frac{z-x}{\beta}}+e^{-\frac{z-y}{\beta}}\right)}$. To match the Gumbel CDF form $e^{-e^{-\frac{z-\mu}{\beta}}}$, we need $e^{-\frac{z-\mu}{\beta}} = e^{-\frac{z-x}{\beta}} + e^{-\frac{z-y}{\beta}}$. Factoring out $e^{-\frac{z}{\beta}}$ and solving for μ gives $\mu = \beta \ln\left(e^{\frac{x}{\beta}} + e^{\frac{y}{\beta}}\right)$, which is exactly the log-sum-exp function. This algebraic manipulation reveals why log-sum-exp is the "natural" operation for Gumbel distributions.

Numerical Stability

Direct computation of $e^{\frac{x}{\beta}} + e^{\frac{y}{\beta}}$ can overflow when $\frac{x}{\beta}$ or $\frac{y}{\beta}$ are large. Similarly, it can underflow when both are very negative, leading to loss of precision. The stable form is:

$$\text{lse}_{\beta(x,y)} = \max(x, y) + \beta \log\left(1 + e^{-\left|x-y\right|_{\beta}}\right)$$

This is the log-sum-exp trick, a fundamental technique in numerical computing. By shifting the computation by the maximum value, we ensure that the largest exponentiated term is $e^0 = 1$, preventing overflow. The correction term is bounded: $0 \leq \beta \log\left(1 + e^{-\left|x-y\right|_{\beta}}\right) \leq \beta \log 2$, ensuring numerical stability.

Why this works: The key insight is that we can shift all values by an arbitrary constant c without changing the result: $\log\left(\sum_i e^{x_i}\right) = c + \log\left(\sum_i e^{x_i - c}\right)$. By choosing $c = \max_i x_i$, we ensure that at least one term in the sum is $e^0 = 1$, and all other terms are ≤ 1, preventing overflow. This trick is essential when working with log-probabilities in statistical modeling, where values can have arbitrary scale depending on the likelihood function and number of data points. The same principle applies to our temperature-scaled version: by factoring out $e^{\frac{\max(x,y)}{\beta}}$, we work with bounded terms that cannot overflow.

Application to Box Intersection

For Gumbel boxes, intersection coordinates are computed using log-sum-exp to determine the location parameters:

- **Intersection minimum**: $z_{\{\text{cap},i\}} \sim \text{MaxGumbel}\left(\text{lse}_{\beta\left(\mu^A_{z,i},\mu^B_{z,i}\right)}, \beta\right)$ This is the maximum of the two boxes' minimum coordinates. The log-sum-exp gives the location parameter of the resulting MaxGumbel distribution. The maximum operation appears because we need the "outermost" minimum coordinate—the intersection's minimum is the maximum of the two boxes' minima.
- **Intersection maximum**: $Z_{\{\text{cap},i\}} \sim \text{MinGumbel}\left(\text{lse}_{\beta\left(\mu^A_{Z,i},\mu^B_{Z,i}\right)}, \beta\right)$ This is the minimum of the two boxes' maximum coordinates. Again, log-sum-exp provides the location parameter. The minimum operation appears because we need the "innermost" maximum coordinate—the intersection's maximum is the minimum of the two boxes' maxima.

This construction preserves the Gumbel distribution family through intersection operations, maintaining algebraic closure (see the Gumbel Max-Stability document). The scale parameter β remains unchanged, ensuring consistency with the original box definitions.

Computational complexity: Computing the intersection of two d-dimensional Gumbel boxes requires $2d$ log-sum-exp operations (one for each min and max coordinate in each dimension), giving time complexity $O(d)$. Each log-sum-exp operation is $O(1)$ using the stable form, making box intersection highly efficient. For N boxes, computing all pairwise intersections costs $O(N^2 d)$, which is the same complexity as computing all pairwise distances for point embeddings. This efficiency is crucial for practical applications where many intersection queries are needed.

Limits

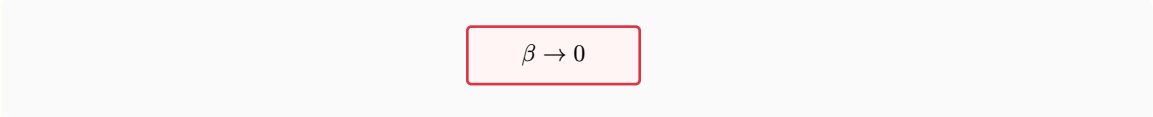
- As $\beta \to 0$: $\text{lse}_{\beta(x,y)} \to \max(x, y)$ (hard maximum). The correction term $\beta \log\left(1 + e^{-|x-y\frac{1}{\beta}}\right) \to 0$, recovering the deterministic maximum. This limit is crucial: it shows that log-sum-exp is a smooth approximation to the maximum function, making it useful for optimization problems where we need differentiability but want behavior close to the hard maximum.
- As $\beta \to \infty$: $\text{lse}_{\beta(x,y)} \to \frac{x+y}{2}$ (arithmetic mean). For large β, the exponential terms become similar, and the log-sum-exp approaches the average. This limit shows that log-sum-exp interpolates between the maximum (most selective) and the mean (most inclusive) operations.

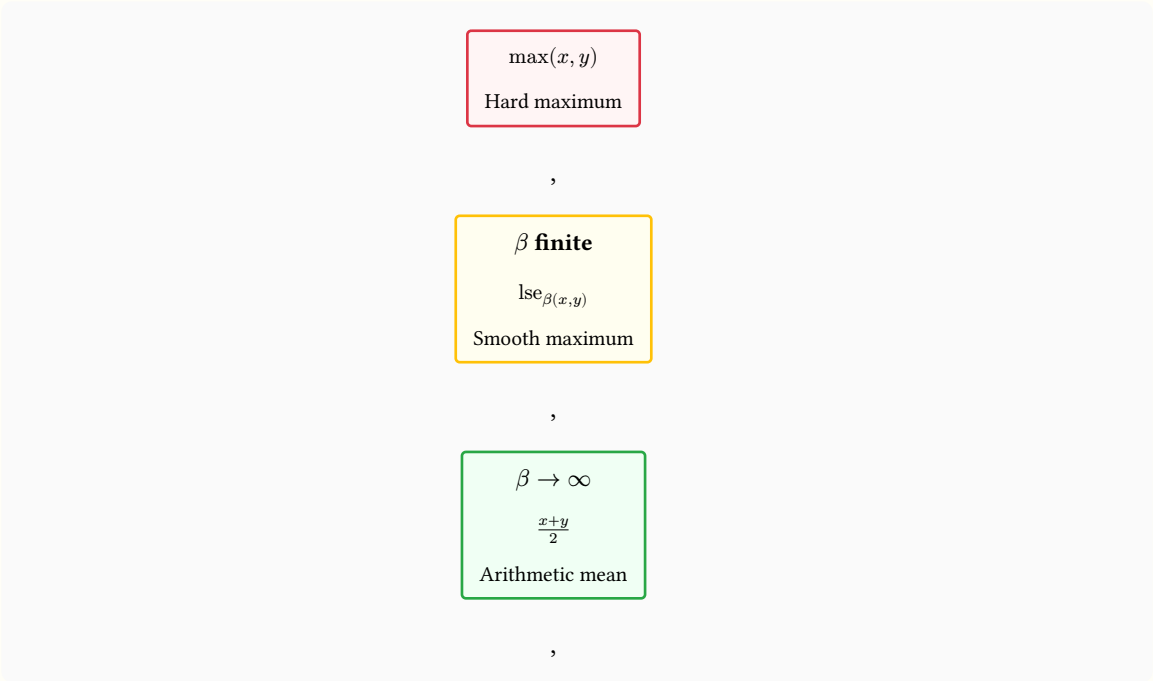
The temperature parameter β controls the "softness" of the maximum operation, interpolating between deterministic (small β) and probabilistic (large β) behavior. This interpolation property is why log-sum-exp is sometimes called a "smooth maximum" or "soft maximum" in the machine learning literature. The function satisfies the bounds: $\max(x, y) < \text{lse}_{\beta(x,y)} \leq \max(x, y) + \beta \log 2$, showing that it's always slightly larger than the hard maximum but never more than $\beta \log 2$ away from it.

Example

For $x = 100.0$, $y = 100.5$, $\beta = 0.1$:

Visual representation: The diagram below shows how log-sum-exp interpolates between the maximum and mean:





Unstable computation:

$$\text{lse}_{\beta(x,y)} = 0.1 * \log(e^{1000} + e^{1005})$$

(overflows!)

Stable computation:

$$\text{lse}_{\beta(x,y)} = 100.5 + 0.1 * \log(1 + e^{-5}) \approx 100.5 + 0.0067 = 100.5067$$

The stable form avoids overflow by working with bounded correction terms. Notice that the result (100.5067) is very close to the maximum (100.5), with a small correction term that ensures smoothness.

Notes

Connection to softmax: The log-sum-exp function is the logarithm of the normalization constant in the softmax function. If we have log-probabilities $\log p_1, \ldots, \log p_k$, then $\text{softmax}_i = \exp(\log p_i - \text{lse}(\log p_1, \ldots, \log p_k))$. This connection explains why log-sum-exp appears so frequently in machine learning: it's the natural operation for normalizing log-probabilities.

Numerical stability in practice: The log-sum-exp trick is one of the most important numerical stability techniques in machine learning. It's used in virtually every implementation of softmax, attention mechanisms, and probabilistic models. The key insight—shifting by the maximum before exponentiating—applies to any sum of exponentials, making it a fundamental tool for working with log-probabilities.

Beyond box embeddings: Log-sum-exp appears in many contexts beyond Gumbel distributions: in attention mechanisms (as the normalization in softmax attention), in probabilistic graphical models (as the log-partition function), and in optimization (as a smooth approximation to the maximum). The ubiquity of log-sum-exp suggests it's a fundamental operation in probabilistic computation, just as addition is fundamental in arithmetic.

Numerical stability edge cases:

1. **Extreme temperature values**: When $\beta \to 0$, the stable form $\max(x, y) + \beta \log\left(1 + e^{-|x-y|_{\beta}^{\downarrow}}\right)$ approaches $\max(x, y)$ as the correction term vanishes. However, for very small β (e.g., $\beta < 10^{-}6$), the computation $e^{-|x-y|_{\beta}^{\downarrow}}$ can overflow if $|x - y|$ is not extremely small. In practice, when $\beta < 10^{-}6$, it's safe to use the hard maximum $\max(x, y)$ directly, as the correction term is negligible compared to floating-point precision.

2. **Very large arguments**: When x or y are very large (e.g., > 100), the stable form $\max(x, y) + \beta \log\left(1 + e^{-|x-y|_{\beta}^{\downarrow}}\right)$ remains stable, but care must be taken when $|x - y|_{\beta}^{\downarrow}$ is very large. In this case, $e^{-|x-y|_{\beta}^{\downarrow}} \to 0$, and the correction term approaches $\beta \log(1) = 0$, so the result is simply $\max(x, y)$. This is the correct behavior, but implementations should handle the case where $|x - y|_{\beta}^{\downarrow} > 50$ by short-circuiting to $\max(x, y)$ to avoid unnecessary computation.

3. **Very close arguments**: When $x \approx y$ (within machine epsilon), the stable form $\max(x, y) + \beta \log\left(1 + e^{-|x-y|_{\beta}^{\downarrow}}\right)$ correctly handles the case, but the correction term $\beta \log\left(1 + e^{-|x-y|_{\beta}^{\downarrow}}\right)$ approaches $\beta \log(2)$ as $|x - y| \to 0$. This means that when $x = y$ exactly, the log-sum-exp gives $x + \beta \log(2)$, which is slightly larger than x. This is the correct behavior (the smooth maximum is always $\geq \max(x, y)$), but it's important to understand that log-sum-exp never exactly equals the hard maximum, even when $x = y$.

4. **High-dimensional aggregation**: When computing log-sum-exp over many values $x_1, ..., x_n$, the stable form generalizes to: $\text{lse}(x_1, ..., x_n) = \max_i x_i + \log\left(\sum_i e^{x_i - \max_j x_j}\right)$. For box intersection with many boxes, this becomes computationally expensive. In practice, when $n > 100$, approximate methods (e.g., sampling a subset of boxes) or hierarchical aggregation may be preferable to exact computation.

5. **Gradient computation**: The gradient of log-sum-exp is the softmax: $\frac{\partial}{\partial x_i} \text{lse}(x_1, ..., x_n) = \frac{e^{x_i}}{\sum_j} e^{x_j} = \text{softmax}_{i(x_1, ..., x_n)}$. This means that gradients are well-behaved and bounded between 0 and 1, ensuring stable backpropagation. However, when some x_i are much larger than others, the softmax becomes very peaked (approaching a one-hot vector), and gradients for smaller values become very small (approaching zero). This can cause vanishing gradients for boxes that are "far" from the maximum. In practice, temperature annealing (starting with larger β and decreasing) helps mitigate this issue by keeping gradients well-distributed early in training.