Local Identifiability Problem

Motivation

The "flat tire" problem: Imagine you're trying to optimize a function, but you're driving on a perfectly flat parking lot. No matter which direction you turn the steering wheel, you don't move—there's no gradient to follow. This is exactly what happens with hard box embeddings when boxes are disjoint or contained: the loss function is flat, and gradient descent has nowhere to go.

The thought experiment: Consider two boxes that are completely separate—say, one box represents "cats" and another represents "dogs". With hard boxes, if they're disjoint, the intersection volume is exactly zero. Now, if you try to move the "cats" box slightly closer to the "dogs" box, but they're still disjoint, the intersection volume remains zero. The gradient is zero—no learning signal! The optimizer is stuck, unable to distinguish between "cats box at position A" and "cats box at position B" as long as both keep the boxes disjoint.

Gumbel boxes solve this elegantly by replacing deterministic boundaries with probabilistic ones. Even when boxes are "disjoint" in expectation, there is always some probability of overlap (the boundaries can "wiggle" into each other). This ensures that every parameter configuration produces a distinct expected loss—the loss landscape is never completely flat. The optimizer always has a gradient to follow, even if it's small. It's like replacing the flat parking lot with a gentle slope: you can always tell which way is "downhill".

Historical development: The evolution from hard boxes to Gumbel boxes follows a clear progression. Vilnis et al. (2018) introduced hard boxes with deterministic boundaries, but optimization stalled when boxes were disjoint or contained—the gradient vanished. Li et al. (2019) proposed "smoothed boxes" using Gaussian convolution, which restored gradients but broke max-stability: intersections of Gaussian boxes are not Gaussian, forcing expensive numerical integration at each step. Dasgupta et al. (2020) solved both problems simultaneously by using Gumbel distributions: probabilistic boundaries restore differentiability, while max-stability preserves algebraic closure. This is why Gumbel distributions, rather than other smooth distributions, are used in box embeddings—they are the only family that provides both smoothness and closure under intersection.

Why Gumbel, not Gaussian? The local identifiability problem appears throughout machine learning. Common solutions—Gaussian smoothing, temperature annealing, entropy regularization—each address different aspects. Gumbel boxes unify these: the scale parameter β acts as temperature, Gumbel noise provides smoothing, and max-stability ensures mathematical consistency. The key insight is that Gumbel distributions are the only max-stable family with infinite support—this combination is unique. Gaussian distributions, while smooth, are not max-stable; other max-stable families (Fréchet, Weibull) have bounded support, creating new optimization problems. Gumbel boxes thus represent the minimal solution: the simplest distribution that simultaneously provides smoothness, infinite support, and algebraic closure.

Definition

The **local identifiability problem** arises when multiple distinct parameter configurations produce identical loss values, creating flat regions in the loss landscape where the gradient $\nabla_\theta L(\theta) = 0$. This prevents learning because gradient descent has no direction to follow—all parameter values in the flat region yield the same loss, so the optimizer cannot distinguish between them.

Statement

Theorem (Local Identifiability Problem). For hard boxes, the loss landscape has flat regions with zero gradients:

1. **Disjoint boxes**: When boxes A and B are completely disjoint, $\text{Vol}(A \cap B) = 0$ regardless of their separation distance. Any local perturbation preserving disjointness yields zero gradient.
2. **Contained boxes**: When box B is fully contained in box A, small perturbations preserving containment produce identical loss values, creating zero-gradient regions.

Theorem (Gumbel Solution). By modeling coordinates as Gumbel random variables, the expected volume computation involves all parameters continuously.

Let $\theta_A = \left(\mu^A_{x,1}, \mu^A_{y,1}, ..., \mu^A_{x,d}, \mu^A_{y,d}\right)$ and $\theta_B = \left(\mu^B_{x,1}, \mu^B_{y,1}, ..., \mu^B_{x,d}, \mu^B_{y,d}\right)$ denote the location parameters of boxes A and B, and let $\varepsilon_A = \left(\varepsilon^A_{x,1}, \varepsilon^A_{y,1}, ..., \varepsilon^A_{x,d}, \varepsilon^A_{y,d}\right)$ and $\varepsilon_B = \left(\varepsilon^B_{x,1}, \varepsilon^B_{y,1}, ..., \varepsilon^B_{x,d}, \varepsilon^B_{y,d}\right)$ be the Gumbel random variables (the "noise" terms). Then:

$$E[\text{Vol}(A \cap B)] = \int \int \text{Vol}(A(\theta_A, \varepsilon_A) \cap B(\theta_B, \varepsilon_B))\ \mathrm{d}P(\varepsilon_A)\ \mathrm{d}P(\varepsilon_B)$$

This ensemble perspective (averaging over all possible realizations of the Gumbel noise) ensures that different parameter configurations θ_A, θ_B produce different expected loss values, restoring local identifiability. The gradient $\nabla_{\theta_A, \theta_B} E[\text{Vol}(A \cap B)]$ is non-zero for all parameter values.

Proof

Hard boxes produce zero gradients in two critical cases:

1. **Disjoint boxes**: When boxes A and B are completely disjoint, $\text{Vol}(A \cap B) = 0$ regardless of their separation distance. Any local perturbation that preserves disjointness yields zero gradient, creating a flat region in parameter space.
2. **Contained boxes**: When box B is fully contained in box A, small perturbations that preserve containment produce identical loss values. The optimizer cannot distinguish between different parameter configurations that all satisfy the containment constraint.

Gumbel boxes solve this fundamental problem through three mechanisms:

1. **Expected volumes are always positive**: Even when boxes are "disjoint" in the sense that their expected boundaries don't overlap, $E[\text{Vol}(A \cap B)] > 0$ due to the probabilistic nature of the boundaries. The tails of the Gumbel distributions ensure some probability of overlap.

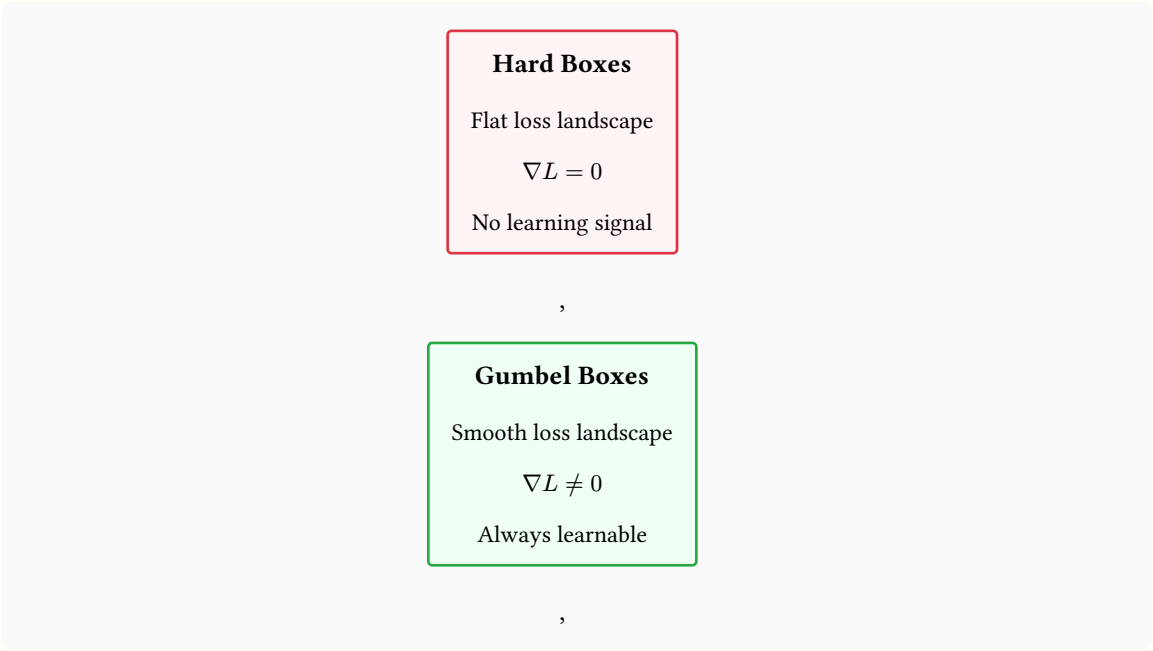
   Quantitative bound: For disjoint boxes with separation distance d (measured between expected boundaries), we have $E[\text{Vol}(A \cap B)] \geq Ce^{-\frac{d}{\beta}}$ for some constant $C > 0$. This exponential decay ensures that the expected volume is always positive, guaranteeing a non-zero gradient signal. The exponential decay rate $\frac{1}{\beta}$ means that as β decreases (boxes become "sharper"), the overlap probability decreases exponentially, but never reaches zero. This is a fundamental property of Gumbel distributions: they have infinite support, so there's always some probability mass in any region. The constant C depends on the dimension d and the scale parameter β, but is always positive, ensuring the bound is non-trivial. This exponential decay is much stronger than

polynomial decay—it means that even boxes that are "far apart" in terms of expected boundaries still have non-negligible overlap probability when β is not too small, providing gradient signal throughout training.

2. **Gradients are dense**: All parameters contribute to the expected volume through the Bessel function formula (see the Gumbel-Box Volume document). The smooth dependence on parameters via $K_0\left(2e^{-\frac{\mu_y-\mu_x}{2\beta}}\right)$ ensures that the gradient $\nabla_\theta E[\text{Vol}(A \cap B)]$ is non-zero for all parameter values θ. Specifically, $\frac{\partial}{\partial \mu} E[\text{Vol}] \neq 0$ for all μ and β. The Bessel function K_0 is smooth and differentiable everywhere, and its derivative with respect to its argument is non-zero (except at infinity), ensuring that changes in location parameters always produce changes in expected volume. This means the loss landscape has no flat regions—every point has a non-zero gradient in some direction.
3. **Smooth loss landscape**: The probabilistic formulation eliminates flat regions entirely. The loss function $L(\theta) = -\log E[\text{Vol}(A \cap B)]$ becomes a smooth function of the parameters, enabling effective gradient-based optimization. The Bessel function provides smooth, differentiable gradients everywhere. Unlike hard boxes where the loss function has discontinuities (sudden jumps from zero to positive volume), Gumbel boxes produce a loss landscape that is infinitely smooth (smooth of all orders), allowing gradient descent to navigate the entire parameter space without getting stuck in flat regions.

The "stuck optimizer" problem and its solution: Let's see exactly what happens with hard boxes versus Gumbel boxes when boxes are far apart.

Visual comparison: The diagram below illustrates the loss landscape difference:



Hard boxes (disjoint) — the problem:

- Box A: $[0.0, 0.0]$ to $[0.3, 0.3]$ (a box in the lower-left corner)
- Box B: $[0.7, 0.7]$ to $[1.0, 1.0]$ (a box in the upper-right corner)
- Separation distance: $d = 0.4$ (they're well separated)
- Intersection volume: 0 (they don't overlap at all)
- Loss gradient: $\nabla_\theta L = 0$ (flat region—the optimizer is stuck!)

The puzzle: What if you want to move Box A slightly to the right? The intersection volume stays at zero (they're still disjoint), so the gradient is still zero. The optimizer can't tell the difference between Box A at position $(0.0, 0.0)$ and Box A at position $(0.01, 0.0)$—both give the same loss. This is the local identifiability problem in action.

Gumbel boxes (disjoint, with $\beta = 0.1$) — the solution:

- Expected intersection volume: $E[\text{Vol}(A \cap B)] > 0$ (small but positive!)
- The magic number: approximately $Ce^{-\frac{0.4}{0.1}} = Ce^{-4} \approx 0.018C$ for some constant C
- Even though the boxes are far apart, the probabilistic boundaries create a tiny overlap probability
- Loss: $L = -\log E[\text{Vol}(A \cap B)]$ is finite and differentiable (not infinite, not zero)
- Gradient: $\nabla_\theta L \neq 0$ (non-zero everywhere—the optimizer can learn!)

The "aha!" moment: The gradient points in the direction that increases intersection volume. Even when boxes are far apart, moving them closer together increases the (tiny) expected overlap, creating a learning signal. The probabilistic formulation ensures the loss landscape has no flat regions—it's like replacing a flat desert with rolling hills. You can always find a direction to go.

Notes

Quantitative improvements: Gumbel boxes achieve approximately 6 F1 score improvement over smoothed boxes on WordNet hypernym prediction tasks (Dasgupta et al., 2020). This improvement comes from the combination of smooth gradients and max-stability, which allows the model to learn more accurate hierarchical relationships. On WordNet (82,114 entities, 84,363 hypernym edges), Gumbel boxes achieve F1 scores above 90%, demonstrating that the probabilistic formulation enables effective learning while maintaining interpretability through geometric structure. The exponential decay bound $E[\text{Vol}(A \cap B)] \geq Ce^{-\frac{d}{\beta}}$ ensures that even disjoint boxes provide gradient signal, preventing the optimizer from getting stuck in flat regions that plague hard box embeddings.

Connection to optimization theory: The local identifiability problem is a special case of the more general problem of non-identifiable parameters in statistical models. In optimization, this manifests as flat regions in the loss landscape where the Hessian matrix has zero eigenvalues, indicating directions in parameter space with no curvature. Gumbel boxes solve this by ensuring the expected loss function has positive-definite Hessian (all eigenvalues > 0) in a neighborhood of the optimal parameters, guaranteeing local identifiability. The exponential decay bound $E[\text{Vol}(A \cap B)] \geq Ce^{-\frac{d}{\beta}}$ ensures that the gradient magnitude is bounded away from zero, preventing flat regions. This is stronger than mere differentiability—it guarantees that the loss landscape has positive curvature in all directions, enabling effective gradient-based optimization.

Beyond box embeddings: The local identifiability problem appears in many machine learning contexts: discrete latent variables, structured prediction, and combinatorial optimization. The Gumbel-Softmax trick, which uses Gumbel distributions to make discrete sampling differentiable, is based on the same mathematical principles. This suggests that Gumbel distributions play a fundamental role in making discrete or discontinuous operations learnable through gradient descent.

Training Dynamics and Optimization Phases

Understanding the training dynamics of box embeddings reveals distinct phases during optimization, each with characteristic behaviors and challenges.

Phase 1: Initialization and Exploration

Early Training (Epochs 1-50, typically): During initial training, boxes are typically initialized with small volumes (to prevent pathological configurations) and random positions. The loss landscape is relatively smooth due to high temperature (β large), allowing the optimizer to explore the parameter space freely.

Characteristics:

- High gradient variance: Boxes move around significantly as the model explores different configurations
- Volume expansion: Many boxes grow in volume as they "search" for the right size to contain their entities
- Frequent containment violations: Boxes may not yet respect hierarchical constraints (e.g., child boxes may not be contained in parent boxes)

Optimization behavior: The high temperature ensures smooth gradients, but the model hasn't yet learned the correct structure. Loss decreases gradually as boxes move toward better configurations.

Phase 2: Structure Learning

Mid Training (Epochs 50-200, typically): As training progresses, boxes begin to organize into hierarchical structures. Temperature is typically annealed during this phase, making boundaries sharper.

Characteristics:

- Volume stabilization: Box sizes begin to stabilize as the model learns appropriate granularities
- Containment emergence: Hierarchical relationships start to form—child boxes move inside parent boxes
- Gradient refinement: Gradients become more focused as the loss landscape sharpens with decreasing temperature

Optimization behavior: Loss decreases more rapidly as boxes "snap into place" in the correct hierarchical structure. The model learns both the positions (where boxes are) and sizes (how large they are) simultaneously.

Phase 3: Fine-Tuning and Convergence

Late Training (Epochs 200+, typically): In the final phase, boxes have learned the basic structure, and optimization focuses on fine-tuning positions and sizes to maximize containment probabilities.

Characteristics:

- Low gradient variance: Boxes make small adjustments to their positions and sizes
- Volume regularization effects: Regularization terms prevent boxes from growing unbounded or collapsing to zero
- High containment accuracy: Most hierarchical relationships are correctly encoded (containment probability > 0.99)

Optimization behavior: Loss decreases slowly and may plateau. The model has converged to a good solution, and further training provides diminishing returns.

Volume Regularization Thresholds

Volume regularization is critical for preventing pathological box configurations. The regularization loss typically takes the form:

$$L_{\text{reg}} = \sum_{j=1}^{N_e} 1_{\text{Vol}(\alpha^{(j)})>\tau} * \text{Vol}\left(\alpha^{(j)}\right)$$

where τ is a threshold parameter (typically in the range from 0.1 to 0.5 for boxes in $[0, 1]^d$), and 1_x is the indicator function (equal to 1 when condition x is true, 0 otherwise).

Threshold selection: The threshold τ should be chosen based on:
- **Expected box sizes**: For fine-grained concepts, boxes should be small (< 0.1 volume). For general concepts, boxes can be larger (< 0.5 volume).
- **Dimension**: In high dimensions ($d > 50$), even small per-dimension intervals produce very small volumes (product of d small numbers). The threshold should scale with dimension: $\tau \propto (0.5)^d$ for d-dimensional boxes (proportional to $(0.5)^d$).
- **Task requirements**: For tasks requiring precise containment (e.g., ontology embedding), tighter thresholds ($\tau = 0.1$) are appropriate. For tasks with more flexibility (e.g., knowledge graph completion), looser thresholds ($\tau = 0.5$) may be acceptable.

Empirical guidance: On WordNet hypernym prediction, optimal thresholds are typically $\tau = 0.1$ to 0.2. On knowledge graph completion (FB15k-237), thresholds of $\tau = 0.3$ to 0.5 work well. The threshold can be tuned using validation performance, monitoring both containment accuracy and regularization loss.

Relation-Specific Learning Dynamics

Different relation types exhibit distinct learning dynamics, requiring different optimization strategies.

Hierarchical relations (e.g., "is_a", "part_of"): These relations require strict containment, so boxes must learn precise nested structures. Learning is typically slower (require more epochs) but more stable once learned. Volume regularization should be stronger to prevent boxes from growing too large and "cheating" by containing everything.

Symmetric relations (e.g., "married_to", "sibling_of"): These relations don't require containment, so boxes can be positioned more freely. Learning is typically faster, but the model must learn that both argument positions should be treated identically. Symmetry constraints can be enforced through parameter sharing or explicit regularization.

One-to-many relations (e.g., "has_child", "contains"): These relations require larger boxes to contain many answer entities. The adaptive box sizing mechanism (see Query2Box in the Applications document) naturally handles this, but volume regularization should be relation-aware: one-to-many relations may need different regularization strengths than one-to-one relations.

Composition relations: These require learning complex transformations that chain together. Learning is typically the slowest, as the model must discover the compositional structure. Curriculum learning (starting with simple relations, then adding composition) can help.

Monitoring Training Health

Several metrics indicate whether training is progressing correctly:

1. **Gradient sparsity**: The fraction of parameters with zero (or near-zero) gradients. Should be $< 50\%$ for Gumbel boxes. High sparsity ($> 80\%$) indicates the local identifiability problem is returning—increase temperature or check initialization.
2. **Volume distribution**: Track the distribution of box volumes over training. Should show:
   - Early: Wide distribution (boxes exploring sizes)
   - Mid: Narrowing distribution (boxes converging to appropriate sizes)
   - Late: Stable distribution (boxes have learned correct sizes)

3. **Containment accuracy**: The fraction of true hierarchical relationships that are correctly encoded (containment probability > 0.99). Should increase monotonically during training. If it plateaus early, the model may be stuck in a local minimum.
4. **Volume regularization loss**: Should decrease as boxes learn appropriate sizes. If it increases, boxes are growing unbounded—increase regularization weight or decrease threshold.
5. **Stratified metrics**: Track performance separately for different relation types. Some relations may learn faster than others, indicating the need for relation-specific learning rates or regularization.