The Logical Random Field

A Unified Theory of Statistical and Logical Reasoning

Greg Coppola

coppola.ai

Super Sonic Vibes

2025

Contents

1 Introduction **1**
1.1 Motivation 1
1.2 Contributions 1
1.3 Outline 1

2 The Logical Random Field **2**
2.1 Background 2
2.1.1 Random Fields 2
2.1.2 First-Order Logic 2
2.2 The Boolean Factor Graph 2
2.3 The Implication Graph 3
2.3.1 Predicate Abstraction 3
2.3.2 Quantified Implication Links 3
2.4 Inference 3
2.4.1 Belief Propagation 3
2.4.2 Complexity 3

3 The Syntactic Parsing Pipeline **4**
3.1 From Surface Form to Logical Form 4
3.2 A Key-Value Calculus 4
3.3 Dependency Parsing 4
3.4 Entity Resolution 5
3.5 Pipeline Architecture 5

4 Experiments **6**
4.1 Synthetic Logical Structures 6
4.1.1 Dating Universe 6
4.1.2 Training 6
4.1.3 Inference Results 6
4.2 Message Propagation Depth 6
4.3 Comparison to Neural Approaches 7

Chapter 1

Introduction

1.1 Motivation

Large language models have demonstrated remarkable capabilities in generating fluent text and performing a wide variety of tasks. However, they suffer from two fundamental limitations: *hallucinations* and *inconsistent reasoning*. We propose that these problems are directly related—a system that reasons consistently from its knowledge base cannot hallucinate, because every output must be justified by valid inference.

This book presents the *Logical Random Field* (LRF), a graphical model that unifies logical and probabilistic reasoning. The LRF is:

- **Generative**: Like language models, it can compress and generate data.
- **Consistent**: It maintains $P(x) + P(\neg x) = 1$ for all propositions.
- **Explainable**: Every inference can be traced through causal structure.
- **Efficient**: Belief propagation runs in $O(N \cdot 2^n)$ where n bounds factor size.

1.2 Contributions

This work makes the following contributions:

1. A unified model of logical and probabilistic reasoning based on random fields over boolean propositions.
2. A syntactic parsing pipeline that maps natural language to the logical forms consumed by the LRF.
3. Experimental validation showing convergence of iterative belief propagation on logical structures.
4. Complexity analysis demonstrating tractable inference for bounded factor graphs.

1.3 Outline

Chapter 2 presents the Logical Random Field formalism. Chapter 3 describes the syntactic pipeline from surface text to logical forms. Chapter 4 presents experimental results on synthetic and linguistic data.

Chapter 2

The Logical Random Field

2.1 Background

2.1.1 Random Fields

A random field defines a probability distribution over a set of random variables through local potential functions. Given variables $\{p_1, \ldots, p_N\}$, a distribution factorizes according to a factor graph G_F if:

$$P(p_1, \ldots, p_N) = Z^{-1} \prod_{\alpha \in F} \Psi_\alpha(\{p\}_\alpha) \tag{2.1}$$

where $\{p\}_\alpha$ are the variables in factor α and Z is the partition function.

2.1.2 First-Order Logic

First-order logic provides the expressive power to represent mathematics and science. Universal quantification and implication work together:

$$\forall x, \text{man}(x) \to \text{mortal}(x) \tag{2.2}$$

This single rule licenses unbounded inferences—the "infinite use of finite means."

2.2 The Boolean Factor Graph

The Logical Random Field restricts attention to *boolean* random fields, where each variable $p \in \{0, 1\}$. We further decompose factors into two types:

- **Conjunction factors** $\Psi_\wedge$: Deterministic AND gates
- **Disjunction factors** $\Psi_\vee$: Learned OR gates (log-linear)

This bipartite structure alternates between conjunction and disjunction, enabling efficient message passing.

Definition 2.1 (Conjunction Factor)**.**

$$\Psi_\wedge(g \mid p_1, \ldots, p_n) = \begin{cases} 1 & \textit{if } g = p_1 \wedge \cdots \wedge p_n \\ 0 & \textit{otherwise} \end{cases} \tag{2.3}$$

Definition 2.2 (Disjunction Factor)**.** *For statistical inference, we model disjunction with a log-linear potential:*

$$\Psi_\vee(p \mid g_1, \ldots, g_n) = \exp\left(\sum_{i=1}^{n} w \cdot \phi(p, g_i)\right) \tag{2.4}$$

2.3 The Implication Graph

2.3.1 Predicate Abstraction

A *proposition* is a fully grounded predicate with no free variables:

$$p = (\text{LIKE}, \{\text{SUBJ} : c_{\text{jack}}, \text{OBJ} : c_{\text{jill}}\}) \tag{2.5}$$

A *predicate* contains open roles with variables:

$$q = (\text{LIKE}, \{\text{SUBJ} : x_{\text{jack}}, \text{OBJ} : x_{\text{jill}}\}) \tag{2.6}$$

2.3.2 Quantified Implication Links

Statistical quantification generalizes logical $\forall$:

$$\Psi[x_{\text{jack}}, x_{\text{jill}}] : \text{like}(x_{\text{jack}}, x_{\text{jill}}) \to \text{date}(x_{\text{jack}}, x_{\text{jill}}) \tag{2.7}$$

The weight Ψ is learned from data, capturing the probabilistic strength of implication.

2.4 Inference

2.4.1 Belief Propagation

We employ iterative belief propagation with π (forward) and λ (backward) messages:

$$\pi(z) = \sum_{a_1,\ldots,a_n} P(z \mid a_1, \ldots, a_n) \prod_i \pi_z(a_i) \tag{2.8}$$

$$\lambda(z) = \prod_c \lambda_c(z) \tag{2.9}$$

The posterior is computed as:

$$P(z \mid \text{evidence}) = \alpha \cdot \lambda(z) \cdot \pi(z) \tag{2.10}$$

2.4.2 Complexity

While general graphical model inference is $\Omega(2^N)$, our boolean decomposition yields $O(N \cdot 2^n)$ per iteration, where n bounds the factor arity. Further optimizations via Noisy-OR models may reduce this to $O(N \cdot n)$.

Chapter 3

The Syntactic Parsing Pipeline

3.1 From Surface Form to Logical Form

The Logical Random Field operates over propositions—but natural language arrives as strings. This chapter describes the pipeline from surface text to the key-value logical forms consumed by the LRF.

3.2 A Key-Value Calculus

Traditional first-order logic imposes arbitrary positional order on arguments:

$$\text{wrote}_{\text{arg0:PER,arg1:BOOK}}(c_{\text{Shakespeare}}, c_{\text{Macbeth}}) \tag{3.1}$$

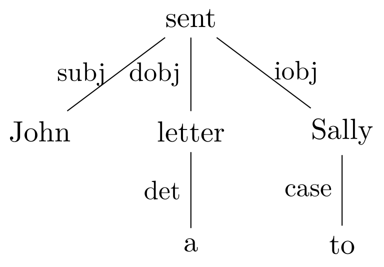
We adopt a key-value formalism closer to dependency structure:

$$(\textsc{wrote}, \{\textsc{arg0} : c_{\text{Shakespeare}}, \textsc{arg1} : c_{\text{Macbeth}}\}) \tag{3.2}$$

This eliminates spurious ordering while preserving semantic content.

3.3 Dependency Parsing

Given a sentence like "John sent a letter to Sally," dependency parsing yields:



From this we extract:

$$(\textsc{send}, \{\textsc{subj} : \text{John}, \textsc{dobj} : \text{letter}, \textsc{iobj} : \text{Sally}\}) \tag{3.3}$$

3.4 Entity Resolution

Constants reference specific entities with types:

$$c_{\text{usa}} = \text{constant}(\text{USA}, \textsc{country}) \tag{3.4}$$

Entity resolution maps surface mentions to canonical constants in the knowledge base.

3.5 Pipeline Architecture

The full pipeline consists of:

1. **Tokenization**: Segment text into tokens
2. **Dependency Parsing**: Extract labeled dependency structure
3. **Semantic Role Labeling**: Map dependencies to semantic roles
4. **Entity Linking**: Resolve mentions to constants
5. **Proposition Construction**: Build key-value logical forms

Each stage admits both neural and symbolic implementations, with the LRF providing a consistent target representation.

Chapter 4

Experiments

4.1 Synthetic Logical Structures

4.1.1 Dating Universe

We investigate a bipartite graph with entities of type x_{jack} and x_{jill}. The generative process:

- $P(\text{lonely}(x_{\text{jack}})) = 0.3$
- $P(\text{exciting}(x_{\text{jill}})) = 0.6$
- $\text{like}(x_{\text{jack}}, x_{\text{jill}}) \Leftrightarrow \text{lonely}(x_{\text{jack}}) \vee \text{exciting}(x_{\text{jill}})$
- $P(\text{like}(x_{\text{jill}}, x_{\text{jack}})) = 0.4$
- $\text{date}(x_{\text{jack}}, x_{\text{jill}}) \Leftrightarrow \text{like}(x_{\text{jack}}, x_{\text{jill}}) \wedge \text{like}(x_{\text{jill}}, x_{\text{jack}})$

4.1.2 Training

We train on 4096 synthetic examples using stochastic gradient descent. The learned model recovers the generative probabilities with small error due to optimization noise.

4.1.3 Inference Results

Theorem 4.1 (Forward Inference Convergence)**.** *When evidence is set at root nodes, beliefs propagate forward through the network in a single iteration.*

Theorem 4.2 (Backward Inference Convergence)**.** *When evidence is set at leaf nodes, beliefs propagate backward at rate $O(d)$ iterations for depth d.*

4.2 Message Propagation Depth

We test belief propagation over a chain of unary predicates $\alpha_0, \ldots, \alpha_N$ with $N = 10$:

$$\alpha_i(x) = \alpha_{i-1}(x) \quad \text{for } i \geq 1 \tag{4.1}$$

Results confirm:

- Forward propagation from α_0: Converges in 1 iteration
- Backward propagation from α_N: Converges in $2N$ iterations (accounting for intermediate conjunction nodes)

4.3 Comparison to Neural Approaches

Unlike neural language models, the Logical Random Field:

1. Maintains probabilistic consistency: $P(x) + P(\neg x) = 1$
2. Provides interpretable inference traces
3. Cannot hallucinate—every output follows from valid reasoning
4. Supports both forward (predictive) and backward (diagnostic) inference

The tradeoff is that the LRF requires structured logical forms rather than operating directly on text. The syntactic pipeline (Chapter 3) bridges this gap.

Bibliography