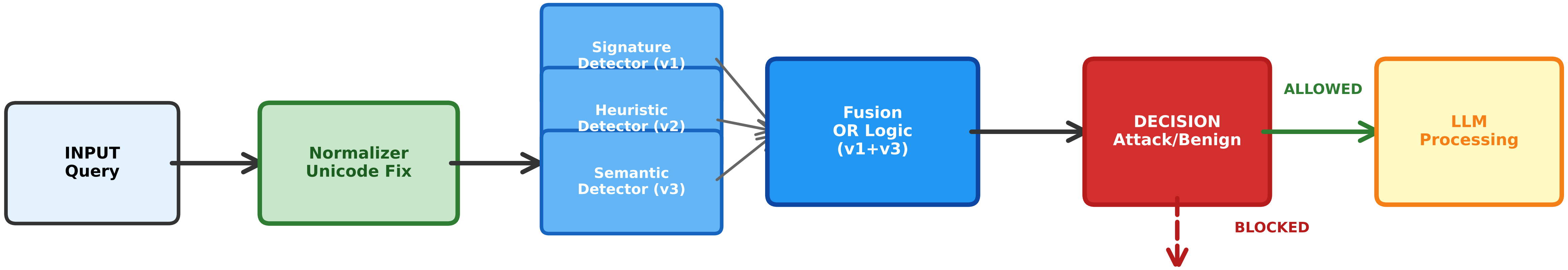
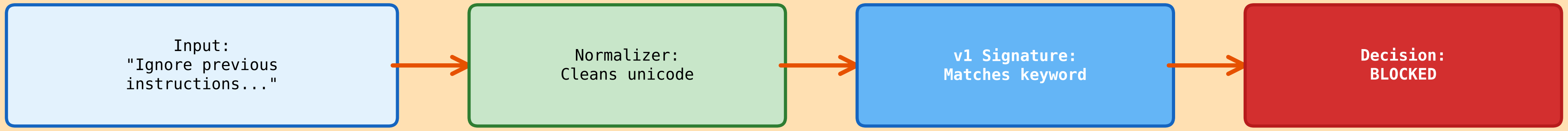
Prompt Injection Detection Pipeline Architecture

Input-Side Detection Before LLM Processing



Example: Attack Processing Flow



Performance: 87% TPR on known attacks | 0.77% FAR on benign | <1ms latency (GPU)

Production Configuration: Normalizer + v3

```
True Positive Rate (TPR): 87% on known attacks
False Alarm Rate (FAR): 0.77% on obfuscated benign
Latency: <1ms per sample (GPU-accelerated)
Complexity: ~1,200 lines of code
Deployment: Stateless, parallelizable
Dependencies: sentence-transformers, torch
```

Component Specifications

```
Signature Detector (v1): Keyword matching
  • 80% TPR, 0% FAR on Phase 1 attacks
  • Catches: plain, delimiter, role confusion

Semantic Detector (v3): Pattern analysis
  • 57% TPR, 0% FAR on Phase 1 attacks
  • Catches: formatting, semantic anomalies

Fusion (OR Logic): v1 OR v3
  • Combined: 87% TPR, 0% FAR
```

Key Design Principles

1. INPUT-SIDE DETECTION: Attacks blocked BEFORE reaching the LLM, preventing prompt injection at the source
2. NORMALIZER FIRST: Unicode/homoglyph normalization ensures consistent detection across obfuscation techniques
3. COMPLEMENTARY DETECTORS: v1 (signature) + v3 (semantic) catch different attack patterns through OR fusion
4. THRESHOLD-INVARIANT: Binary OR logic eliminates threshold tuning complexity in deployment
5. PRODUCTION-READY: <1ms latency with GPU acceleration, stateless architecture

Legend: □ Input 🟩 Normalizer 🟦 Detector 🟦 Fusion 🟥 Decision 🟨 LLM