

Example LinkedIn Post — “Teaching My iOS App to Understand Language”

The Challenge

Ever wondered how Siri or ChatGPT understands *what* you mean instead of just reading *what* you type?

That curiosity pushed me into Apple’s **Natural Language** and **Core ML** frameworks. My goal: build a Swift app that can interpret text, classify its tone, and respond intelligently — all on-device.

What I Built

I developed a small **sentiment-aware iOS prototype** where users can input a phrase, and the app instantly detects whether it’s positive, neutral, or negative. Behind the scenes, it uses **NLTagger** and **NLModel** from Apple’s Natural Language framework, combined with a lightweight **Core ML sentiment model**.

The key challenge was ensuring everything worked *locally* — no external API calls — to preserve privacy and real-time performance.

What I Learned

Working with Apple’s Natural Language tools taught me that language understanding is less about syntax and more about *context*.

I learned to:





- Fine-tune Core ML models using **Create ML**
- Implement tokenization and lemmatization for cleaner analysis
- Handle multilingual text gracefully without crashing the pipeline

Even something as simple as detecting sarcasm or ambiguity reminded me how much nuance there is in human language.

Tech Stack & Architecture

This prototype runs on **Swift 6.2** with **MVVM-C architecture** to separate logic from UI.

Frameworks included:

-  **Core ML + Create ML** for model training
 -  **Natural Language (NLTagger, NLModel)** for inference
 -  **SwiftUI + Combine** for UI updates
 -  **On-device inference** for privacy and speed
-

Visual Snippet

I've attached a short clip showing the real-time sentiment analysis — the background color of the text field changes dynamically based on detected emotion. Small touch, big feedback loop. 🧐

Reflection

This project made me appreciate how powerful Apple's ecosystem is for AI — not because it's flashy, but because it prioritizes privacy and performance.

The hardest part wasn't training the model — it was designing the experience so users *trust* the results.

Next, I plan to expand this into a **multimodal analyzer** that combines voice tone (from **Speech Framework**) and text sentiment for richer context.

Call to Action

If you're experimenting with Core ML, Natural Language, or on-device AI — let's connect!

You can check out the code and model setup here: github.com/yourrepo

Would love feedback from other iOS devs pushing ML to the edge. 🚀