**Intelligent Personal Injury Law Chatbot Technical Report: Algorithms and Functional Architecture**

**Prepared by:** Wajahat Khan  
**Date:** 29th April 2025

### **1. Introduction**

This report presents a professional overview of the architecture, logic, and algorithms implemented in two chatbot modules: Chatbot.py and EnhancedLawFirmChatbot.py. These modules collectively power an intelligent chatbot designed to assist users in discovering, comparing, and obtaining recommendations about personal injury law firms across five Canadian provinces.

### **2. System Overview**

The chatbot is designed to interpret natural language queries from users and provide contextually appropriate responses based on a structured JSON knowledge base of law firms.

#### **Key Features:**

* **Recommendations:** Suggest firms based on location, specialization, and case type.
* **Content Search & Discovery:** Retrieve metadata such as websites, traffic, and LinkedIn profiles.
* **Opinion & Analysis:** Offer comparative insights across provinces or categories.

### **3. Chatbot.py – Core Functionalities**

The Chatbot.py module establishes the base LawFirmChatbot class, which is responsible for loading, parsing, and managing the law firm data.

#### **3.1 Data Handling:**

* Data is imported from JSON format, where each firm has fields like name, province, monthly\_visits, website, and rank.
* The module normalizes data for case-insensitive comparison and search.

#### **3.2 Basic Query Resolution:**

* Province-based filtering is implemented using conditional list comprehensions.
* If a firm name or province keyword is found in the query, matching results are returned.

#### **3.3 Limitations:**

* Lacks deeper NLP processing.
* Cannot understand multi-intent queries or perform comparative evaluations.

### **4. EnhancedLawFirmChatbot.py – Advanced Logic and NLP Integration**

This subclass enhances the core logic by integrating more intelligent query handling and broader use of pattern recognition, token classification, and ranking.

#### **4.1 Recommendation Algorithm:**

* Filters are applied based on query classification: **province, specialization**, or **traffic** level.
* Implements semantic matching for legal specializations (e.g., workplace injury, car accidents).
* Ranking is based on **rank** and **monthly\_visits**.

**Sample Logic:**

if specialization:

results = [f for f in results if specialization in f["description"].lower()]

#### **4.2 Content Search & Discovery:**

* Implements fuzzy matching for firm names and metadata.
* Supports keyword and phrase matching for fields like **LinkedIn, monthly\_visits**, etc.
* Supports retrieval by metadata class (e.g., "high-traffic firms").

**Sample Logic:**

if "traffic" in query:

return [f for f in self.firms if "25K" in f["monthly\_visits"] or "50K" in f["monthly\_visits"]]

#### **4.3 Opinion & Analysis Algorithm:**

* Aggregates and compares provinces using **Counter** and statistical analysis.
* Computes average ranks and total firm counts by province.

**Sample Logic:**

from collections import Counter

counts = Counter([f["province"] for f in self.firms])

#### **4.4 Response Handling:**

* Natural language responses are templated for clarity.
* Includes fallback suggestions for vague or unsupported queries.

### **5. Evaluation Criteria Fulfillment**

| **Objective** | **Fulfilled** | **Details** |
| --- | --- | --- |
| Recommendations | Yes | Province, case type, specialization supported |
| Content Search & Discovery | Yes | Metadata querying fully supported |
| Opinion & Comparative Analysis | Yes | Multi-province comparison logic implemented |
| Good Interface Integration | Yes | Province cards, user input, firm display |

### 

### **6. Conclusion**

The combination of **Chatbot.py** and **EnhancedLawFirmChatbot.py** successfully meets the requirements of an intelligent assistant in the domain of personal injury law. The bot effectively handles structured queries, offers valuable firm recommendations, enables detailed metadata discovery, and provides analytical comparisons across Canadian provinces.