Neural Network Assignment - 2

Information Retrieval

Submitted to
Dr. Syed Khaldoon Khurshid

Submitted By

Shahzaib Irfan 2021-CS-07

University of Engineering and Technology
Lahore, Pakistan

Contents

1 Introduction

1.1 Project Overview

This project develops an advanced Tech Trend Information Retrieval System using a sophisticated Python-based framework with Streamlit. The system leverages cutting-edge techniques in semantic search, neural networks, and information retrieval to provide intelligent and context-aware technology trend insights.

1.2 Objectives

The main objectives of this project are:

- Implement an intelligent semantic search mechanism for technology articles
- Develop a neural network-based relevance ranking system
- Create an interactive web application for technology trend discovery
- Demonstrate advanced machine learning and natural language processing techniques

2 System Design

2.1 System Architecture

The system comprises multiple interconnected components:

1. **Semantic Understanding Module**
    - Maintains a semantic dictionary of technology terms
    - Performs query expansion and semantic mapping
2. **Neural Network Inference Engine**
    - Implements a custom neural network for relevance scoring
    - Handles forward propagation and prediction
    - Manages model training and feature extraction
3. **Information Retrieval Processor**

- Processes user queries
- Performs semantic understanding
- Retrieves and ranks relevant articles

4. **Interactive Web Interface**
   - Built using Streamlit
   - Allows dynamic query processing
   - Provides interactive result visualization

3 Implementation

3.1 Libraries and Tools

The project utilizes the following Python libraries:

- **Streamlit**: Web application framework
- **Math**: Mathematical computations
- **Random**: Weight initialization

3.2 Core Components and Algorithms

3.2.1 Neural Network Implementation

The `NeuralNetwork` class represents a custom neural network with key functionalities:

Listing 1: Neural Network Class

```
class NeuralNetwork:
    def __init__(self, input_size=4, hidden_size=5,
        output_size=1):
        # Network architecture initialization
        # Random weight and bias generation
```

Key Features:

- Supports configurable input, hidden, and output layer sizes
- Uses sigmoid activation function
- Implements forward propagation
- Supports training with backpropagation

3.2.2 Semantic Understanding

The `semantic_understanding()` method creates a semantic map of technology terms:

Listing 2: Semantic Understanding Method

```
def semantic_understanding(self, keywords):
    semantic_map = {}
    for word in keywords:
        related_terms = self.semantic_dictionary.get(word,
            [])
        semantic_map[word] = related_terms
    return semantic_map
```

Key Capabilities:

- Expands query terms with related semantic concepts
- Uses a predefined semantic dictionary
- Enables context-aware search

3.2.3 Query Expansion and Feature Extraction

The `query_expansion()` and `extract_features()` methods enhance query processing:

Listing 3: Query Expansion and Feature Extraction

```
def query_expansion(self, keywords, semantic_map):
    expanded_terms = set(keywords)
    for word in keywords:
        expanded_terms.update(semantic_map.get(word, []))
    return list(expanded_terms)

def extract_features(self, expanded_terms):
    return [
        len(expanded_terms) / 10.0,
        len(set(expanded_terms)) / len(expanded_terms),
        sum(len(term) for term in expanded_terms) / len(
            expanded_terms),
        len([term for term in expanded_terms if any(t in term
            for t in self.semantic_dictionary)])
    ]
```

Key Features:

- Semantically enriches initial query terms
- Generates numerical features for neural network prediction
- Captures query complexity and semantic richness

3.2.4 Search and Retrieval

The `search_and_retrieve()` method ranks articles based on neural network predictions:

Listing 4: Search and Retrieval Method

```
def search_and_retrieve(self, expanded_terms):
    features = self.extract_features(expanded_terms)

    matching_articles = []
    for article in self.articles:
        if any(term.lower() in article.get('content', '').
           lower() for term in expanded_terms):
            relevance_score = self.nn.predict(features)[0]
            matching_articles.append({
                'article': article,
                'relevance_score': relevance_score
            })

    return sorted(matching_articles, key=lambda x: x['
       relevance_score'], reverse=True)
```

Key Capabilities:

- Matches articles against expanded query terms
- Uses neural network to predict article relevance
- Ranks and returns most relevant articles

4 Results and Evaluation

4.1 Model Performance

- Successfully implemented a neural network-based relevance ranking system
- Developed a semantic search mechanism
- Created an interactive Streamlit web interface
- Demonstrated advanced query expansion techniques

5 Conclusion

The project successfully developed an intelligent Tech Trend Information Retrieval System, showcasing advanced techniques in:

- Neural network-based relevance scoring
- Semantic query expansion
- Interactive information retrieval

5.1 Future Enhancements

- Integrate more sophisticated natural language processing models
- Expand semantic dictionary with machine learning
- Implement more advanced feature extraction techniques
- Add support for more complex query structures
- Enhance neural network architecture for improved predictions