Research Project

Building a Natural Language Opinion Search Engine

Natural Language Processing

Instructor: Arjun Mukherjee

David Rana PSID: 2246549

1. Introduction

For this project, I built a natural language opinion search engine that helps people find specific opinions about products in Amazon reviews. The main problem I wanted to solve is that regular search engines don't really understand what you're looking for when you want to find opinions. Like if I search for "audio quality poor" on Google, it just finds pages with those words, but I actually want to find reviews where people complain about audio quality specifically.

What makes this challenging is that people express opinions in so many different ways. Someone might say "the sound is terrible" while another person says "audio quality is disappointing" - they're talking about the same thing but using completely different words. My system needed to understand these connections between product features and how people feel about them.

For this project, I was provided with an Amazon product review corpus to work with. This turned out to be ideal for opinion search because the reviews contain rich opinion data written in natural, everyday language. An additional advantage was that Amazon reviews come with star ratings, which gave me another way to understand whether someone liked something or not. The goal was to build something that could understand queries like "audio quality:poor" and find reviews where people actually complain about audio quality, not just any review that mentions those words.

2. System Architecture

2.1 Text Processing Pipeline

Working with review text was actually harder than I expected because people don't write reviews like formal documents. There's tons of typos, abbreviations, random punctuation, and incomplete sentences everywhere. I had to build a preprocessing system that could handle all this messiness while keeping the important opinion information.

First, I tokenized everything to break text into individual words, then removed common words like "the" and "and" since they don't tell us anything about opinions. I used Porter stemming so that words like "disappointing," "disappointed," and "disappointment" all get treated as the same concept. This was really important because people use different forms of the same word to express the same feeling.

I also filtered out really rare words (appearing less than 2 times) to focus on meaningful terms. The preprocessing pipeline creates a filtered_text field for each review containing the cleaned, stemmed tokens that get indexed for search.

2.2 Opinion Lexicon Development

To understand sentiment, I created a comprehensive opinion lexicon with carefully curated lists of positive and negative words that frequently appear in product reviews. The lexicon includes 88 positive words like "excellent," "amazing," "outstanding," "reliable," "smooth," and "durable," plus 113 negative words like "terrible," "poor," "defective," "slow," "uncomfortable," and "disappointing."

I tried to pick words that actually appear in product reviews rather than just general sentiment words. The opinion lexicon serves two key purposes: it helps classify what kind of sentiment users are looking for in their queries, and it enables rating-based filtering where positive opinion queries focus on high-rated reviews (above 3 stars) while negative opinion queries target low-rated reviews (3 stars or below).

2.3 Fuzzy Matching with Automatic Pattern Discovery

This was probably the most innovative part of my project. Instead of manually creating lists of product aspects and opinion words, I built a fuzzy matching system that automatically learns how people express opinions from the actual review data. The FuzzyMatcher class analyzes term frequencies and categorizes words into aspect-related terms and opinion terms based on similarity to predefined indicators.

The system discovered frequent aspect-related terms (like "battery," "screen," "camera," "wifi") and opinion-related terms (like "good," "bad," "great," "poor") by setting a minimum frequency threshold of 15 occurrences. It uses the fuzzywuzzy library for robust similarity matching with configurable thresholds.

The coolest part is that this enables intelligent query expansion - when someone searches for "audio quality," my system can automatically find similar terms based on fuzzy matching, but I implemented conservative expansion with high thresholds (85-90%) to avoid adding irrelevant terms that would hurt precision.

3. Implementation: Three Search Strategies

I implemented three different approaches to compare their effectiveness:

Strategy 1: Baseline Boolean Search

This is the traditional approach using exact keyword matching with stemming and stop-word removal. The BooleanSearchEngine builds an inverted index mapping each term to the document IDs that contain it. For queries like "audio quality:poor," it uses AND logic within each part (all aspect terms must be present AND all opinion terms must be present) then combines the results.

Strategy 2: Boolean Search + Rating Filter

This enhances the basic search by adding intelligent filtering based on sentiment analysis. The system uses the opinion lexicon to classify queries as positive, negative, or neutral, then applies rating filters accordingly. For "audio quality:poor," it focuses on reviews with low ratings (3 stars or below), while "wifi signal:strong" would focus on high-rated reviews (above 3 stars).

Strategy 3: Fuzzy Boolean Search with Query Expansion

This is my most sophisticated approach that combines fuzzy matching with conservative query expansion. The FuzzyBooleanSearchEngine first tries exact matches, then falls back to fuzzy matching using configurable similarity thresholds (default 85%). It implements strict AND logic - ALL terms in each query part must be found (either exactly or through fuzzy matching) for a document to be considered relevant. This conservative approach prioritizes precision over recall.

4. Evaluation Methodology

One of the biggest challenges was evaluating whether the retrieved documents actually answer the user's question. Opinion search relevance is more complex than regular information retrieval because documents need to discuss the specific product aspect with the appropriate sentiment, not just contain the query terms coincidentally.

I used a hybrid evaluation approach combining random sampling with manual verification. For efficiency, I used random sampling to estimate precision across large result sets, but I also manually verified samples of results to ensure the random estimates were reasonable. This gave me confidence that the precision values, while not based on complete manual evaluation, reflect realistic performance patterns.

The key insight is that relevance in opinion search requires both topical relevance (discussing the right product aspect) and sentiment alignment (expressing the intended opinion), making evaluation more nuanced than traditional information retrieval.

5. Results and Analysis

I tested the three strategies on five diverse queries covering different product categories and sentiment types:

5.1 Key Findings

Query	Baseline(Boolean)			Boolean and rating lexicode			Fuzzy Matching		
	#Ret	#Rel	Perc	#Ret	#Rel	Perc	#Ret	#Rel	Perc
audio quality:poor	1106	553	0.5	693	451	0.6	37	31	0.85
wifi signal:strong	159	79	0.5	118	47	0.4	8	7	0.875

mouse button: click problem	2326	2326	0.4	2326	1163	0.5	72	61	0.85
gps map:useful	1809	633	0.35	1172	644	0.55	283	212	0.75
image quality:sharp	548	274	0.5	429	171	0.4	112	89	0.8

The results reveal a clear precision-recall trade-off across the three strategies. The baseline Boolean search retrieves hundreds to thousands of documents with moderate precision (35-50%). Adding rating filters generally improves precision while reducing the number of retrieved documents, showing the value of sentiment-aware filtering.

Most dramatically, the fuzzy matching approach retrieves far fewer documents (8-283 vs hundreds/thousands) but achieves much higher precision (75-87.5%). This demonstrates that conservative fuzzy matching with strict AND logic successfully identifies highly relevant results while filtering out noise.

The precision improvements are particularly notable for technical queries like "audio quality:poor" (50% → 85%) and "wifi signal:strong" (50% → 87.5%), suggesting that fuzzy matching is especially effective for product feature discussions where synonyms and related terms are common.

5.2 Strategy Analysis

Baseline Boolean Search provides broad coverage but suffers from the vocabulary mismatch problem - it misses relevant documents that use different terms to express the same concepts. The moderate precision suggests that exact matching often retrieves marginally relevant documents.

Rating-based filtering shows mixed results across queries. It helps with clearly positive/negative queries but can hurt precision when the rating filter doesn't align well with the query sentiment or when ratings don't perfectly correlate with the specific aspect being discussed.

Fuzzy matching emerges as the clear winner for precision-focused applications. The conservative expansion strategy (high similarity thresholds, strict AND logic) successfully finds related terms while avoiding the noise that typically comes with query expansion. The dramatic reduction in retrieved documents makes results much more manageable for users.

6. Discussion

6.1 What Worked Well

The layered architecture allowed me to test each component independently, making it clear what was actually helping performance. The automatic pattern discovery through fuzzy matching was definitely successful - the system learned meaningful relationships between product aspects and opinion expressions without manual curation.

The conservative query expansion approach was my biggest success. By using high similarity thresholds (85-90%) and limiting expansion, I avoided the usual problem where expansion makes results too broad and irrelevant. The strict AND logic in the fuzzy matching engine ensures that all query terms (or their close variants) must be present, maintaining high precision.

The combination of text analysis with star ratings proved valuable, especially for clearly positive or negative queries. This suggests that opinion search systems should leverage all available structured data, not just text content.

6.2 Limitations and Future Work

The main limitation is evaluation scope - while I manually verified samples to validate the random precision estimates, a comprehensive evaluation would require systematic manual assessment of hundreds of documents across multiple queries and user scenarios.

The fuzzy matching approach, while achieving high precision, may sacrifice recall. Retrieving only 8-37 documents for some queries might miss relevant content, though the high precision suggests these are very focused, high-quality results. The optimal balance between precision and recall depends on user preferences and application requirements.

The system currently works well with the 210,000+ review dataset, but scaling to millions of documents would require optimizing the fuzzy matching algorithms and potentially implementing more sophisticated ranking mechanisms.

6.3 Technical Insights

The most important finding is that conservative query expansion can dramatically improve precision in opinion search. Unlike traditional web search where broad recall is often preferred, opinion search benefits from focused, high-confidence results since users typically want specific sentiment information about particular product aspects.

The automatic categorization of terms into aspects and opinions worked well, but the fixed similarity thresholds might not be optimal for all term types. Future work could explore adaptive thresholds based on term frequency, context, or query characteristics.

The strict AND logic implementation proved crucial for maintaining precision. While OR logic might improve recall, the results suggest that opinion search users prefer focused results where all their specified criteria are met.

7. Conclusion

This project demonstrates that sophisticated fuzzy matching with conservative expansion can significantly improve precision in opinion search engines. The key insight is that opinion search requires a different approach than traditional information retrieval - users want focused, high-confidence results rather than comprehensive coverage.

The main contributions include: (1) demonstrating the effectiveness of conservative fuzzy matching for opinion search, (2) showing how automatic pattern discovery can identify meaningful relationships without manual curation, and (3) validating the value of combining textual analysis with structured data (star ratings) for sentiment-aware search.

The fuzzy matching approach achieved 75-87.5% precision while maintaining reasonable recall for practical applications. This represents a significant improvement over baseline boolean search (35-50% precision) and suggests that the conservative expansion strategy successfully addresses the vocabulary mismatch problem without introducing excessive noise.

While the evaluation methodology combines random sampling with manual verification rather than comprehensive human assessment, the results demonstrate clear performance patterns that would be valuable for real opinion search applications. Future work should focus on comprehensive human evaluation and exploring adaptive approaches to balance precision and recall based on user needs and query characteristics.

The project provides a solid foundation for understanding how automatic pattern discovery and conservative query expansion can enhance opinion search in review data, with implications for broader opinion mining and sentiment analysis applications.