Aspect-Based Sentiment Analysis Project

Course: Applied Computer Science – Data & AI 6
Team Members: Filip Zekovich and Ryan Mattew

---

1. Introduction

Sentiment analysis has long served as a critical subfield of Natural Language Processing (NLP), enabling automated systems to understand public opinion, customer feedback, and emotional tone within text. Traditional sentiment classification approaches typically operate at the document or sentence level, producing a single sentiment label (positive, neutral, negative). However, real-world texts—such as product reviews or social media posts—often express multiple opinions about different aspects of a subject (e.g., "The food was great, but the service was slow").

This more granular perspective has given rise to **Aspect-Based Sentiment Analysis (ABSA)**, which aims to identify sentiments associated with specific aspects or attributes of an entity rather than producing an overall sentiment score.

Project Objectives

The purpose of this project is to perform a comparative evaluation of three ABSA implementations:

1. **LexiconABSA** - A rule-based approach using spaCy and VADER
2. **ML_ABSA** - A transformer-based approach using DeBERTa
3. **LLMABSA** - An LLM-based approach using local Ollama (Gemma 3)

The project's primary objectives are:

- To implement and compare three fundamentally different approaches to ABSA
- To analyze methodological and architectural differences between rule-based, transformer, and LLM-driven sentiment approaches
- To evaluate practical trade-offs in terms of accuracy, speed, resource requirements, and use-case suitability
- To provide recommendations for deployment scenarios based on empirical results

---

2. Methodology

2.1 LexiconABSA (Rule-Based Approach)

Implementation Details:

The LexiconABSA implementation combines linguistic analysis with sentiment lexicons:

- **Aspect Extraction:** Uses spaCy's dependency parsing to extract noun chunks and nouns as aspect candidates
- **Opinion Detection:** Identifies opinion words (adjectives, adverbs, verbs) related to aspects through dependency relations
- **Sentiment Scoring:** Applies VADER (Valence Aware Dictionary and sEntiment Reasoner) to score sentiment polarity
- **Negation Handling:** Detects negation patterns (e.g., "not good") and inverts sentiment accordingly

Key Features:

- Extracts aspects using linguistic rules (POS tags, dependency parsing)
- Context window analysis when direct opinion words aren't found
- Handles modifiers and hedge words ("could", "should") with confidence adjustments
- Fully deterministic and interpretable

Advantages:

- Fast inference (~7-10ms per review)
- No GPU required
- Interpretable results
- No training data needed

Limitations:

- Extracts many irrelevant aspects (pronouns: "I", "it", "he")
- Struggles with implicit sentiment
- Cannot handle sarcasm or humor
- Poor at aggregating multi-aspect sentiment

2.2 ML_ABSA (Transformer-Based Approach)

Implementation Details:

The ML_ABSA implementation uses a pre-trained transformer model fine-tuned for ABSA:

- **Model:** `yangheng/deberta-v3-base-absa-v1.1` from Hugging Face
- **Architecture:** DeBERTa-v3 (Decoding-enhanced BERT with disentangled attention)
- **Aspect Extraction:** Uses spaCy to extract candidate aspects (noun chunks, top 12)
- **Sentiment Classification:** For each aspect, constructs input `[TEXT] [SEP] [ASPECT]` and classifies sentiment
- **Output:** Returns sentiment labels (positive/negative/neutral) with confidence scores from model probabilities

Key Features:

- State-of-the-art transformer architecture
- Contextual understanding through attention mechanisms
- GPU acceleration support (automatic detection)
- Model size: ~500MB

Advantages:

- Most accurate sentiment classification across test categories
- Handles complex multi-aspect reviews effectively
- Understands implicit sentiment and context
- Reasonable inference speed (~30-50ms per review)

Limitations:

- Still extracts irrelevant aspects (pronouns, generic nouns)
- Requires more computational resources than lexicon-based
- Less interpretable than rule-based approaches
- May over-predict sentiment for neutral terms based on overall review tone

---

2.3 LLMABSA (LLM-Based Approach)

Implementation Details:

The LLMABSA implementation leverages local large language models through Ollama:

- **Model:** Gemma 3 (1B parameters) running locally via Ollama
- **Prompting Strategy:** Few-shot prompting with examples
- **System Prompt:** Instructs the model to extract aspect-sentiment pairs in JSON format
- **Output Parsing:** Extracts and validates JSON response, with fallback mechanisms
- **Temperature:** 0.3 for consistent, deterministic outputs

Few-Shot Examples Provided:

```
Input: "The pizza was delicious but the service was terrible."
Output: [
  {"aspect": "pizza", "sentiment": "positive", "confidence": 0.95},
  {"aspect": "service", "sentiment": "negative", "confidence": 0.92}
]
```

Key Features:

- Zero-shot/few-shot learning without training data
- Natural language understanding through pre-trained knowledge
- Structured JSON output
- Fallback to simple aspect extraction if LLM fails

Advantages:

- Best aspect extraction quality (filters pronouns, consolidates terms)
- Handles edge cases (sarcasm, humor, implicit sentiment)
- Strong contextual understanding
- Can identify abstract aspects ("dog's health", "online shopping experience")
- Most human-like interpretation

Limitations:

- Significantly slower (~400-600ms per review, 10-60x slower than transformer)
- Requires Ollama service running locally
- Higher resource consumption
- Occasional duplicate aspects or hallucinations
- Less consistent output format than deterministic methods

2.4 Unified API Design

All three implementations conform to a single interface:

```python
class ABSAAnalyzer:
    def analyze(self, text: str) -> List[AspectSentiment]:
        """Analyze text and return aspect-sentiment pairs."""
        raise NotImplementedError

@dataclass
class AspectSentiment:
    aspect: str                      # The aspect mentioned
    sentiment: str                    # "positive", "negative", or "neutral"
    confidence: float                 # Confidence score 0.0 to 1.0
    text_span: Tuple[int, int] = None  # Character positions (optional)
```

This design enables:

- Easy comparison between implementations
- Swappable analyzers for different use cases
- Consistent evaluation metrics
- Extensibility for future implementations

---

3. Experimental Setup

3.1 Dataset

Source: Amazon Dog Food Reviews (Kaggle)
URL: https://www.kaggle.com/datasets/unwrangle/amazon-reviews-for-dog-food-product

Dataset Composition:

- **Total Reviews:** 30 curated samples
- **Simple Cases:** 10 reviews with clear, straightforward sentiment
- **Complex Cases:** 10 reviews with multiple aspects and mixed sentiment
- **Edge Cases:** 10 reviews testing boundary conditions

Dataset Characteristics:

Category	Count	Avg Length (words)	Challenges
Simple Cases	10	~30	Clear sentiment, single/few aspects
Complex Cases	10	~450	Multi-aspect, mixed sentiment, long text
Edge Cases	10	~80	Sarcasm, typos, ambiguity, formatting issues

Edge Case Types:

- Sarcasm/humor: "Haven't heard complaints from my dog"
- Typos: "Our dog love it" (grammar error)
- Ambiguity: "Great" (no explicit aspect)
- Formatting: "CANCELATION isNOThappening!!!"
- Implicit negatives: "makes me question freshness"

3.2 Environment

Hardware:

- CPU: x86_64 architecture
- RAM: 16 GB
- GPU: Optional (CUDA support for ML_ABSA)

Software:

- Python: 3.8+
- Operating System: Linux
- Jupyter Notebook for experiments

Key Dependencies:

```
spacy==3.8.7
en-core-web-sm==3.8.0
vaderSentiment==3.3.2
torch==2.9.0
transformers==4.57.1
ollama
```

Ollama Setup:

- Service: Ollama daemon running locally
- Model: gemma3:1b (pulled and loaded)

3.3 Evaluation Metrics

Qualitative Analysis:

- Aspect extraction quality (relevance, noise level)
- Sentiment accuracy on simple vs. complex cases
- Handling of edge cases (sarcasm, typos, ambiguity)
- Output consistency and format

Quantitative Metrics:

- **Inference Speed:** Average time per review (milliseconds)
- **Aspect Quality:** Proportion of relevant vs. irrelevant aspects extracted
- **Sentiment Patterns:** Distribution of positive/negative/neutral classifications
- **Edge Case Performance:** Success rate on challenging samples

Testing Procedure:

1. Run all three analyzers on identical dataset
2. Benchmark speed over 5 runs on simple cases (10 reviews)
3. Manually evaluate aspect quality and sentiment accuracy
4. Document failure modes and edge case handling

---

4. Results and Analysis

4.1 Quantitative Results

Speed Benchmark (Simple Cases - 10 Reviews, 5 Runs Average):

Method	Time per Review	Relative Speed	Resource Usage
Lexicon-Based	7-10 ms	1x (baseline)	CPU only, minimal RAM
Transformer-Based	30-50 ms	3-5x slower	GPU optional, ~500MB model
LLM-Based	400-600 ms	40-60x	CPU/GPU, ~2GB RAM, Ollama

		slower	service

Key Findings:

- Lexicon is fastest, suitable for real-time applications (<10ms requirement)
- Transformer provides good balance (acceptable for batch processing)
- LLM is significantly slower but offers best quality

4.2 Qualitative Comparison

4.2.1 Simple Cases

Example: "Best dog food ever"

Method	Extracted Aspects	Sentiment	Observations
Lexicon	"Best dog food", "dog", "food"	All POSITIVE	Over-extraction (redundant aspects)
Transformer	"Best dog food", "dog", "food"	All POSITIVE	Same over-extraction pattern
LLM	"dog food"	POSITIVE	Clean consolidation, best quality

Example: "Tried to change over to this brand, and my dog just doesn't like it and won't eat it."

Method	Extracted Aspects	Sentiment	Observations
Lexicon	"this brand", "my dog", "it", "brand", "dog"	Mixed (some NEUTRAL)	Handles negation, but noisy
Transformer	"this brand", "my dog", "it", "brand", "dog"	Mostly NEGATIVE	Better sentiment consistency
LLM	"brand", "dog"	NEGATIVE	Cleanest extraction

Pattern: All methods perform well on explicit sentiment, but LLM provides cleanest aspect extraction.

4.2.2 Complex Cases

Example: "My pup likes this food. The problem is ....The best by or expiration date was not present and there was a small hole in the bag so I don't know the quality so I threw it out and bought a bag from my pet store. Very disappointing and expensive."

Lexicon Results:

- Extracts: "My pup", "this food", "The problem", "a small hole", "the bag", "I", "the quality", "it", "a bag", "my pet store"
- Sentiments: Mixed (often incorrect - "The problem" → POSITIVE with 0.64 conf)
- Issues: Cannot aggregate conflicting sentiments properly

Transformer Results:

- Extracts: Similar aspects to Lexicon

- Sentiments: Better accuracy ("The problem" → NEGATIVE, "a small hole" → NEGATIVE, "the quality" → NEGATIVE)
- Strengths: Understands implicit negative sentiment ("makes me question")

LLM Results:

- Extracts: "pup" (POSITIVE), "food" (NEGATIVE), "quality" (NEGATIVE), "price" (NEGATIVE)
- Sentiments: Correctly identifies mixed sentiment and consolidates related aspects
- Strengths: Understands "disappointing and expensive" applies to purchase experience

Key Finding: Transformer and LLM significantly outperform Lexicon on complex, multi-aspect reviews.

4.2.3 Edge Cases

Example: "Haven't heard any complaints from my dog so I guess it's delicious." (Sarcasm/Humor)

Method	Extracted Aspects	Sentiment	Accuracy
Lexicon	"any complaints", "my dog", "I", "it"	Mostly NEUTRAL	□ Misses sarcasm
Transformer	Similar aspects	Mixed (some POSITIVE for "it")	□ Partial understanding
LLM	"dog"	POSITIVE	□ Correctly interprets humor

Example: "Great" (Single word, ambiguous)

Method	Extracted Aspects	Sentiment	Handling
Lexicon	(no aspects found)	N/A	Cannot process
Transformer	(no aspects found)	N/A	Cannot process
LLM	"overall"	POSITIVE	□ Infers general sentiment

Example: "dog does not like food . CANCELATION isNOThappening !!! STOP SENDING" (Formatting issues)

Method	Handling	Observations
Lexicon	Extracts "dog", "food", "CANCELATION"	Handles negation, struggles with formatting
Transformer	Similar extraction	Better sentiment classification despite formatting
LLM	Extracts "dog" (NEGATIVE)	Most robust to formatting issues

Key Finding: LLM excels at edge cases (sarcasm, ambiguity, formatting), while Lexicon and Transformer struggle.

4.3 Aspect Extraction Quality

Relevant vs. Irrelevant Aspects (Observed Trends):

Method	Relevant Aspects	Irrelevant Aspects (Pronouns/Generic)	Quality Score
Lexicon	High count	Very High ("I", "it", "he", "she", "that", "this")	Low
Transformer	High count	High (same pronouns, plus "years", "days")	Medium
LLM	Moderate count	Very Low (filters pronouns, consolidates)	High

Example Comparison - "My dogs love this. I have two dogs that won't eat anything else."

- **Lexicon:** Extracts 8 aspects including "I", "that", "anything"
- **Transformer:** Extracts 8 aspects, same noise
- **LLM:** Extracts 2 aspects: "dogs" (with duplicate sentiment entries - occasional issue)

Conclusion: LLM provides superior aspect quality by filtering noise and consolidating related terms.

4.4 Sentiment Accuracy Patterns

Observed Performance by Category:

Category	Lexicon Accuracy	Transformer Accuracy	LLM Accuracy
Simple Cases (Explicit Sentiment)	Good	Very Good	Excellent
Complex Cases (Multiple Aspects)	Poor	Good	Very Good
Edge Cases (Sarcasm, Ambiguity)	Poor	Fair	Excellent

Specific Observations:

1. **Explicit Sentiment (e.g., "loves it", "terrible"):**
   - All methods perform well
   - LLM slightly better at consolidation
2. **Implicit Sentiment (e.g., "makes me question freshness"):**
   - Lexicon: Fails (often marks as NEUTRAL or incorrectly POSITIVE)
   - Transformer: Good (correctly identifies NEGATIVE)
   - LLM: Excellent (best contextual understanding)
3. **Multi-Aspect Mixed Sentiment:**

    - Lexicon: Poor aggregation (conflicting signals confuse scoring)
    - Transformer: Good (handles most cases correctly)
    - LLM: Excellent (cleanly separates aspects and sentiments)

4. **Negations:**

    - Lexicon: Good on simple negations ("doesn't like")
    - Transformer: Very good (contextual understanding)
    - LLM: Excellent (handles complex negation patterns)

4.6 Resource and Cost Analysis

Method	Model Size	RAM Usage	GPU Requirement	Deployment Cost
Lexicon	~50MB (spaCy model)	<500MB	None	Very Low
Transformer	~500MB	~2GB	Optional (10x speedup)	Low-Medium
LLM	~2GB (Gemma 3 1B)	~4GB	Optional	Medium-High

---

5. Discussion

5.1 Strengths and Weaknesses Summary

Lexicon-Based (spaCy + VADER)

Strengths:

- Fast inference suitable for high-volume processing
- Interpretable results based on linguistic rules
- No GPU or external services required
- Works well on simple, clear sentiment expressions
- andles straightforward negations

Weaknesses:

- Extracts too many irrelevant aspects (pronouns, generic terms)
- Struggles with implicit sentiment and context
- Cannot handle sarcasm or humor
- Poor at aggregating multi-aspect sentiment
- Often defaults to neutral when uncertain

Best For:

- High-throughput systems (millions of reviews/day)
- Real-time processing (<10ms requirement)
- Resource-constrained environments
- Simple reviews with straightforward language
- Baseline comparisons and quick prototyping

Transformer-Based (DeBERTa)

Strengths:

- Most accurate sentiment classification across categories
- Handles complex multi-aspect reviews effectively
- Understands implicit sentiment and context
- Reasonable inference speed for production use
- Works across various sentiment expressions
- Better aspect filtering compared to lexicon approach

Weaknesses:

- Still extracts many irrelevant aspects (pronouns, generic nouns)
- Over-predicts sentiment for neutral terms
- Can misclassify neutral aspects based on overall review sentiment
- Requires more computational resources
- Less interpretable than rule-based approaches
- Model size considerations (~500MB)

Best For:

- **Recommended for most production use cases**
- E-commerce review analysis and aggregation
- Customer feedback processing at scale
- Batch processing of thousands to millions of reviews
- Systems with GPU access (cloud deployment)
- Cross-domain applications

LLM-Based (Gemma 3)

Strengths:

- Best aspect extraction quality (filters irrelevant terms)
- Extracts meaningful, consolidated aspects
- Handles edge cases excellently (sarcasm, humor, implicit sentiment)
- Strong contextual understanding across long, complex reviews
- Can identify abstract aspects
- Most human-like interpretation
- Handles ambiguous cases ("Great" → infers "overall")

Weaknesses:

- Significantly slower (10-60x slower than transformer)
- Requires Ollama service running locally with model downloaded
- Occasional duplicate aspects with different sentiments
- Less consistent output format than deterministic methods
- Higher resource consumption
- Sometimes consolidates too much (misses sub-aspects)
- May hallucinate aspects not explicitly mentioned

Best For:

- Detailed feedback analysis and market research
- Low-volume, high-value reviews (B2B customer feedback)
- Research applications requiring nuanced understanding
- Cases where aspect quality justifies computational cost
- Analysis of sarcastic, humorous, or complex reviews
- Exploratory analysis of new product categories

6. Conclusion

6.1 Key Findings

This comparative study demonstrates that **model sophistication directly correlates with performance** in Aspect-Based Sentiment Analysis tasks, with clear trade-offs in speed, cost, and implementation complexity:

1. **Lexicon-Based (LexiconABSA):**
    - Offers excellent **speed** (~7-10ms) and **interpretability**
    - Sufficient for simple, high-volume scenarios
    - Limited by inability to handle context, sarcasm, and implicit sentiment
    - Best for **real-time, resource-constrained** applications
2. **Transformer-Based (ML_ABSA):**
    - Provides **best balance** of accuracy and speed (~30-50ms)
    - **Recommended for most production deployments**
    - Handles 80-90% of reviews accurately (estimated from qualitative analysis)
    - Suitable for **batch processing and general e-commerce** applications
3. **LLM-Based (LLMABSA):**
    - Achieves **highest quality** aspect extraction and sentiment understanding
    - **10-60x slower** than transformer approach (~400-600ms)
    - Excels at **edge cases, sarcasm, and nuanced sentiment**
    - Best for **low-volume, high-value** analysis and research

---