Medical Abbreviation Disambiguation Using Naive Bayes Classification

Vihaan Manchanda
vm180@duke.edu

Anvita Suresh
as1693@duke.edu

Arushi Singh
as1685@duke.edu

Duke University
Introduction to Natural Language Processing
Instructor: Patrick Wang
November 19, 2025

1 Motivation

Medical abbreviations create significant safety risks in healthcare. A single abbreviation like "CP" can represent Chronic Pain, Chest Pain, or Cerebral Palsy, each requiring dramatically different treatments. The Joint Commission reports that abbreviation-related errors consistently rank among the top causes of sentinel events [2].

Consider this scenario: A physician writes "Patient presents with CP and difficulty breathing." Without proper context, this could be interpreted as Chest Pain (requiring immediate cardiac workup for potential myocardial infarction) when the physician actually meant Chronic Pain with anxiety-related hyperventilation. Conversely, misinterpreting Chest Pain as Chronic Pain could delay critical cardiac intervention. If the context involves a pediatric patient and CP refers to Cerebral Palsy, the treatment approach differs entirely, focusing on motor function and developmental support rather than acute symptom management.

The problem extends beyond patient safety. Clinical decision support systems, automated coding, and health information exchange platforms rely on accurate text interpretation. Ambiguous abbreviations force manual review, creating workflow bottlenecks.

This project addresses abbreviation disambiguation as classification: Given an ambiguous abbreviation and surrounding context, can we automatically predict the correct expansion? Success would enable safer documentation, more reliable automated systems, and reduced cognitive load on healthcare providers.

2 Related Work

2.1 Inspiration: The MeDAL Paper

Our work is inspired by Wen et al. (2020) [1], who created MeDAL (Medical Dataset for Abbreviation Disambiguation). Their goal differed from ours: they used abbreviation disambiguation as a pretraining task to improve performance on downstream medical NLP tasks like mortality and diagnosis prediction. They employed LSTM, LSTM with Self-Attention, and ELECTRA transformers, achieving 82–84% accuracy.

2.2 How Our Approach Differs

We diverge in three ways. First, we treat disambiguation as the primary problem, focusing on interpretability, computational efficiency, and understanding failure modes rather than transfer learning. Second, we use classical Multinomial Naive Bayes instead of deep learning, providing explicit independence assumptions and interpretable probability estimates. Third, we evaluate on a carefully filtered subset with clear acronym patterns (first letters match), whereas MeDAL used the full dataset.

Our question: Can simple classical methods compete with deep learning when given appropriate features? What can the performance gap reveal about medical abbreviation disambiguation's fundamental difficulty?

3 Implementation Overview

3.1 Problem Formulation

We formulate disambiguation as multi-class classification. Given abbreviation a and context words $w_1, ..., w_n$, predict expansion e^*:

$$e^* = \arg\max_{e \in E} P(e|w_1, ..., w_n)$$

For our three selected abbreviations, we have:

- CC: {colorectal cancer, cell culture, cervical cancer}
- CP: {chronic pain, chest pain, cerebral palsy}
- SA: {surface area, sleep apnea, substance abuse}

This creates a 9-class classification problem.

3.2 Why Multinomial Naive Bayes?

We selected Naive Bayes for five specific reasons:

1. Bag-of-words matches medical terminology. Distinctive keywords signal meanings: "colon" suggests colorectal cancer, "flask" suggests cell culture. Naive Bayes captures these associations without requiring sequential information.

2. Independence assumption is reasonable. Within a 5-word window, most words are not grammatically dependent. "Colon" three words before and "tumor" two words after independently provide evidence for colorectal cancer.

3. Interpretability. Explicit $P(w|e)$ probabilities enable straightforward failure analysis. We can examine which keywords drive predictions.

4. Data efficiency. Performs well with limited training data compared to deep learning requiring millions of examples.

5. Computational efficiency. Training is $O(nd)$, inference is $O(kd)$. Practical for real-time clinical decision support.

3.3 Why Test TF-IDF?

We hypothesized TF-IDF would improve accuracy by down-weighting common medical stopwords ("the", "of", "patients") that dominate feature space but provide little discriminative power. TF-IDF: $\text{TF-IDF}(w, d) = \text{TF}(w, d) \times \log \frac{N}{\text{DF}(w)}$ reduces influence of high document frequency words while preserving context-specific terms. We test this in Section 7.

4 Data Selection and Filtering

4.1 Dataset Choice

We used MeDAL (14.4 million PubMed abstracts with automatically labeled abbreviations via reverse substitution). We chose MeDAL over our original proposal (MIMIC-III) for three reasons: (1) labels already exist, (2) public availability without credentialing, (3) scale and diversity across medical specialties.

4.2 Filtering Pipeline

The full dataset contains 5,886 abbreviations. Our four-stage filtering:

Stage 1: Multi-word filter. Keep only multi-word expansions (e.g., "colorectal cancer" not "cancer") to ensure true acronyms.

Stage 2: First-letter matching. Verify first letters match abbreviation. Removes spurious labels like "PT" → "tumors".

Stage 3: Minimum sense filter. Require $\geq$3 expansions per abbreviation for meaningful classification.

Stage 4: Balance and relevance. Select abbreviations with 50K+ examples, reasonable balance (no class >40%), and pure medical terminology.

The filtering script (`preprocessing/filter_data.py`) processes 14.3 million examples, producing 113,371 filtered examples (0.79% of original).

4.3 Final Selection

Abbrev	Top 3 Expansions	Count	%
CC	colorectal cancer	28,201	30.4%
	cell culture	18,441	19.9%
	cervical cancer	14,876	16.0%
CP	chronic pain	10,183	18.3%
	chest pain	7,953	14.3%
	cerebral palsy	5,258	9.5%
SA	surface area	15,936	27.2%
	sleep apnea	6,903	11.8%
	substance abuse	5,620	9.6%

Table 1: Selected abbreviations (total: 113,371 examples).

Why these three? **CC** tests organ-specific cancer distinction plus laboratory context. **CP** tests temporal vs. anatomical framing with neurological dimension. **SA** spans measurement, respiratory, and behavioral domains, testing cross-specialty disambiguation.

4.4 Synthetic Data Generation

Step 3a requires synthetic data aligning with Naive Bayes assumptions: independent words with distinctive keywords per class. This tests whether our model works under ideal conditions. High accuracy on synthetic but low on real data indicates real-world assumption violations.

We used template-based generation with keyword slots:

```
"Patient with {abbrev} showing {kw1}
and {kw2} findings"
```

We manually defined discriminative keywords based on domain knowledge. Examples: colorectal cancer (colon, rectal, tumor, polyp), cell culture (medium, serum, flask, cells), cerebral palsy (motor, spastic, children, pediatric), sleep apnea (obstructive, CPAP, snoring, apneic).

For each expansion, we generated 200 unique examples (1,800 total, perfectly balanced) by randomly selecting templates and keywords. Why templates rather than sampling real data? Sampling would create synthetic data too similar to our test set. Templates create idealized examples where keywords perfectly discriminate, representing best-case scenarios.

The generation script (`preprocessing/generate_synthetic.py`) ensures uniqueness by rejecting duplicates.

5 Feature Extraction and Model Implementation

5.1 Feature Representation

We represent each example as a bag-of-n-grams within a fixed context window:

Algorithm 1 Feature Extraction

```
Input: Text, abbreviation location
Output: Feature vector

Extract words within ±5 positions of abbreviation
Generate unigrams from context words
Generate bigrams from consecutive word pairs

Generate trigrams from consecutive word triples
Convert to count vector using vocabulary
Return: sparse count vector
```

Why ±5 word window? We tested different window sizes in preliminary experiments. Windows smaller than 5 often missed discriminative keywords. Windows larger than 7 introduced noise and increased computational cost without improving accuracy.

Why n-grams up to 3? Unigrams capture individual keywords ("colon", "tumor"). Bigrams capture phrasal patterns ("cell culture", "chronic pain"). Trigrams capture longer phrases ("obstructive sleep apnea"). N-grams longer than 3 appeared too rarely to be useful.

Vocabulary construction: For real data, we filtered the vocabulary to keep only n-grams appearing at least 3 times across the training set. This reduced vocabulary size from 943,627 to 91,205, making the model computationally tractable while removing noise.

5.2 Naive Bayes Implementation

We implement Multinomial Naive Bayes from scratch:

$$P(e|w_1, ..., w_n) \propto P(e) \prod_{i=1}^{n} P(w_i|e)$$

Training computes:

$$P(e) = \frac{\text{count}(e)}{N}$$

$$P(w|e) = \frac{\text{count}(w, e) + \alpha}{\sum_{w'} \text{count}(w', e) + \alpha|V|}$$

where $\alpha = 1$ is the Laplace smoothing parameter and $|V|$ is vocabulary size.

For numerical stability, we use log probabilities:

$$\log P(e|w_1, ..., w_n) = \log P(e) + \sum_{i=1}^{n} \log P(w_i|e)$$

Prediction selects the class with highest log probability:

$$e^* = \arg\max_e \left[\log P(e) + \sum_{i=1}^{n} \log P(w_i|e) \right]$$

Implementation details:

- Feature vectors stored as NumPy float32 arrays (memory efficiency)
- Probability tables stored as dictionaries for fast lookup
- Batch processing for TF-IDF computation to avoid memory overflow

The complete implementation is in `src/models.py` (83 lines including comments).

Algorithm 2 TF-IDF Transformation (Batched)

```
Input: Count matrix $X$ (n_samples × n_features)
Output: TF-IDF matrix

Compute document frequency: $\text{DF}_j = \sum_{i=1}^{n} 1[X_{ij} > 0]$
Compute IDF: $\text{IDF}_j = \log \frac{n+1}{\text{DF}_j+1} + 1$
for batch in chunks of 5000 samples do
    Compute TF: $\text{TF}_{ij} = \frac{X_{ij}}{\sum_j X_{ij}}$
    Compute TF-IDF: $Y_{ij} = \text{TF}_{ij} \times \text{IDF}_j$
end for
Return: TF-IDF matrix $Y$
```

5.3 TF-IDF Transformation

For TF-IDF experiments, we transform count vectors:

We process in batches because the full 79,359 × 91,205 matrix exceeds memory when creating intermediate copies during normalization.

6 Evaluation on Synthetic Data

6.1 Experimental Setup

Training details: 1,800 synthetic examples (200 per class), 70% train (1,260), 30% test (540), 3,767 unique n-grams, Multinomial NB with $\alpha = 1$.

6.2 Quantitative Results

6.3 Qualitative Analysis: Why Does the Model Succeed?

Success example 1: *[CC] Label: cell culture. Context: "...for quantitation of iron in studies with iron oxide nanoparticles..." Explanation:* Keywords "studies", "nanoparticles", "quantitation" strongly indicate laboratory research context. The model correctly assigns high probability to "cell culture" over cancer-related expansions.

Success example 2: *[CP] Label: cerebral palsy. Context: "...a month-old female infant with who had been having feeding..." Explanation:* Keywords "infant", "month-old", "feeding" indicate pediatric context. Cerebral palsy is the only CP expansion associated with children in our keyword set.

Model	Acc	Class	P	R	F1
Baseline	11.1%	cell culture	1.0	1.0	1.0
NB(raw)	**99.6%**	cer. palsy	1.0	1.0	1.0
NB(TF)	**99.6%**	cer. cancer	.99	1.0	.99
		chest pain	1.0	1.0	1.0
		chr. pain	1.0	1.0	1.0
		col. cancer	1.0	.99	.99
		sleep apn	1.0	1.0	1.0
		sub. abuse	1.0	1.0	1.0
		surf area	1.0	1.0	1.0

Table 2: Accuracy on synthetic test set and per-class metrics. Baseline predicts most frequent class (1/9 = 11.11%). Both NB variants achieve near-perfect accuracy. Only 2 errors out of 540 predictions.

The model succeeds because synthetic data was constructed with exactly the features Naive Bayes assumes: distinctive keywords that appear independently and clearly signal the correct class.

6.4 Qualitative Analysis: The One Failure Mode

Out of 540 predictions, only 2 were incorrect. Both failures had the same cause:

Failure example: *[SA] True: colorectal cancer — Predicted: cervical cancer. Context: "Diagnosis of SA confirmed ratio and volume features". Explanation:* This example should be labeled SA (surface area, sleep apnea, or substance abuse), but the true label is "colorectal cancer" which is not a valid SA expansion. This is a data generation bug: the abbreviation variable was incorrectly set to SA when the template was filled with colorectal cancer keywords.

This failure reveals a bug in our synthetic generation script, not a fundamental model limitation. The high accuracy (99.63%) confirms that Naive Bayes works perfectly when data matches its assumptions.

7 Evaluation on Real Data

7.1 Experimental Setup

Training details: 113,371 real medical abstracts from MeDAL, 70% train (79,359), 30% test (34,012), 91,205 n-grams (min frequency = 3), Naive Bayes with raw counts and TF-IDF.

7.2 Quantitative Results

7.3 Confusion Analysis

The confusion matrix reveals systematic error patterns:

7.4 Why Does Performance Drop from Synthetic to Real?

The 21% accuracy gap (99.63% → 78.80%) reveals violations of Naive Bayes assumptions in real data:

Violation 1: Overlapping terminology. In synthetic data, we carefully selected non-overlapping keywords. In real data, medical terminology overlaps: "screening" appears in both cervical cancer and colorectal cancer contexts; "patients" appears in all clinical contexts; "treatment" appears across multiple conditions.

Feature importance analysis confirms this. The top 10 features for most classes are generic stopwords: colorectal cancer ['of', 'in', 'the', 'and', 'with', 'patients', ...], cervical cancer ['of', 'the', 'in', 'and', 'with', 'for', ...], chronic pain ['of', 'the', 'and', 'in', 'with', 'to', 'patients', ...]. Only a few classes have distinctive keywords in their top 10: sleep apnea ['obstructive', 'of', 'and', 'in', 'with', 'osa', ...], cerebral palsy ['with', 'of', 'in', 'children', 'and', 'the', 'cp', ...]. This explains why SA and cerebral palsy perform better: they have domain-specific keywords ("obstructive", "osa", "children", "cp") that don't appear in other contexts.

Violation 2: Insufficient context. Many examples have minimal context: *Context: "is the highly actual issue of". True: cerebral palsy, Predicted: surface area.* With only generic words and no discriminative keywords, the model defaults to prior probabilities or weak statistical associations.

Violation 3: Semantic similarity. Cervical and colorectal cancer share similar contexts: Both discuss screening, diagnosis, treatment; Both mention patients, tumors, stages; Both use similar epidemiological language. The key distinguishing features (organ names) may fall outside the context window or be abbreviated themselves.

7.5 Qualitative Analysis: Successful Predictions

Success 1: Clear keyword signal. *[CC] Label: colorectal cancer. Context: "...colon accounts for of all diagnosis is often delayed and..." Why success:* Keyword "colon" directly signals colorectal cancer.

Model	Acc	Abbrev	Acc	N	Class	P	R	F1
Baseline	24.88%	SA	**84.0%**	8,538	sleep apnea	.930	.810	.866
NB (raw)	**78.80%**	CC	78.6%	18,456	surface area	.894	.928	.911
NB (TF-IDF)	78.45%	CP	73.0%	7,018	substance abuse	.806	.626	.705
					cell culture	.816	.908	.859
					colorectal cancer	.758	.788	.773
					cervical cancer	.693	.632	.661
					chest pain	.808	.815	.811
					chronic pain	.634	.688	.660
					cerebral palsy	.904	.681	.777

Table 3: Accuracy on real test set, per-abbreviation accuracy, and per-class precision, recall, and F1 scores. Raw counts outperform TF-IDF by 0.35%, contrary to our hypothesis. SA performs best, CP worst. Note the wide performance variation across classes.

True Class	Predicted Class	Count
cervical cancer	colorectal cancer	1,170
colorectal cancer	cervical cancer	879
colorectal cancer	cell culture	401
chronic pain	colorectal cancer	340
substance abuse	chronic pain	293
colorectal cancer	chronic pain	258
cervical cancer	cell culture	234
surface area	cell culture	210
cell culture	surface area	210
cerebral palsy	chronic pain	160

Table 4: Top 10 confusion pairs. The cervical/colorectal cancer confusion accounts for 2,049 errors (28.4% of all failures).

Success 2: Domain-specific terminology. *[SA] Label: sleep apnea. Context: "...are three main categories of obstructive sleep apnea osa central..." Why success:* "obstructive" and "osa" are highly specific to sleep apnea.

Success 3: Contextual clustering. *[CP] Label: cerebral palsy. Context: "...a month-old female infant with who had been having feeding..." Why success:* "infant", "month-old", "feeding" cluster to indicate pediatric context.

The model succeeds when: (1) Distinctive keywords appear in the context window, (2) Multiple weak signals converge to one class, (3) Context is sufficiently informative (not truncated).

7.6 Qualitative Analysis: Failure Cases

Failure 1: Overlapping cancer terminology. *[CC] True: cervical cancer — Predicted: colorectal cancer. Context: "...in more than of human g1 catenin stimulates..." Why failure:* Generic molecular biology language. "catenin" appears in both cancer contexts. No organ-specific keyword.

Failure 2: Cross-domain confusion. *[CP] True: chronic pain — Predicted: colorectal cancer. Context: "...patients n undergoing treatment for at four cl..." Why failure:* "patients" and "treatment" appear in both pain management and oncology. The model learned spurious associations between "treatment" and cancer.

Failure 3: Insufficient context. *[CP] True: cerebral palsy — Predicted: surface area. Context: "...the is highly actual issue of..." Why failure:* No discriminative keywords. The model has no signal to distinguish classes.

Failure 4: Ambiguous medical terminology. *[SA] True: substance abuse — Predicted: chronic pain. Context: "...related to increased risk for in terms of trea..." Why failure:* "risk" and "treatment" appear in both substance abuse and pain management contexts. The model cannot distinguish without more specific keywords like "addiction" or "opioid".

The model fails when: (1) Terminology is shared across multiple classes, (2) Context window is too small or poorly formed, (3) Discriminative keywords fall outside the window, (4) The text uses abbreviations within abbreviations.

7.7 Why TF-IDF Performed Worse

We hypothesized that TF-IDF would improve accuracy by down-weighting common words like "the", "of", "patients". However, TF-IDF achieved 78.45% compared to 78.80% for raw counts (0.35% worse).

Explanation: In medical text, common medical terms like "patients", "treatment", "diagnosis" are actually discriminative in aggregate. Consider: "patients" appears frequently in clinical contexts (cancer, pain) but rarely in laboratory contexts (cell culture, surface area); "treatment" appears more in chronic conditions than acute symptoms; "diagnosis" appears more in disease contexts than measurement contexts.

By down-weighting these common medical terms, TF-IDF removes useful signal. Raw term frequency preserves this information: cell culture contexts use "culture", "medium", "cells" frequently, while cancer contexts use "patients", "treatment", "diagnosis" frequently. The absolute frequency matters, not just the relative rarity.

This finding suggests that in domain-specific text, "common" words within that domain still carry discriminative power. TF-IDF is designed for general web text where stopwords like "the" and "of" are truly non-discriminative. In medical text, the "common" words are medical terms that do provide signal.

8 Model Analysis and Discussion

8.1 Strengths of the Approach

1. Interpretability

Unlike neural models that learn distributed representations in high-dimensional space, Naive Bayes provides explicit probabilities for each word-class pair. We can examine $P(\text{"colon"}|\text{colorectal cancer})$ and understand exactly why the model made a prediction.

This interpretability is crucial for clinical deployment. When the model makes an error, clinicians can review the feature probabilities and understand the failure mode. With a neural model, explaining why "cervical cancer" was predicted instead of "colorectal cancer" requires complex attribution methods like attention weights or SHAP values.

2. Data Efficiency

We achieved 78.80% accuracy with only 79,359 training examples. Neural models typically require millions of examples to reach comparable performance. The MeDAL paper used 3 million training examples for their LSTM and ELECTRA models.

This efficiency matters when labeled data is expensive. While MeDAL provides large-scale data, many specialized medical domains have limited annotations. A simple model that performs reasonably with limited data is more practical than a complex model that requires massive datasets.

3. Computational Efficiency

Training time: 3 minutes on a standard laptop (MacBook Pro M1) Inference time: 0.01 seconds per example

Compare this to the MeDAL paper's neural models:

- LSTM: Hours of training on GPU, 0.1 seconds per example
- ELECTRA: Days of pre-training, hours of fine-tuning, 0.5 seconds per example

For real-time clinical decision support, Naive Bayes can provide instant predictions as clinicians type. Neural models introduce noticeable latency.

4. Robustness to Domain Shift

Because Naive Bayes uses explicit keyword features, it generalizes to new medical specialties that share terminology. If we encounter "bronchial cancer" (not in training), the model can still recognize "cancer" as a disease signal and "bronchial" as an anatomical term.

Neural models might fail on out-of-vocabulary terms unless they use subword tokenization or character-level representations.

8.2 Limitations of the Approach

1. Independence Assumption Violation

The fundamental assumption of Naive Bayes is that words are conditionally independent given the class:

$$P(w_1, w_2, ..., w_n|e) = \prod_{i=1}^{n} P(w_i|e)$$

This is clearly false in natural language. ”Colorectal” and ”cancer” are not independent; if ”colorectal” appears, ”cancer” is much more likely. This violation manifests in our results:

When we see ”colorectal”, the model should strongly predict colorectal cancer. But if ”cancer” also appears, the model multiplies $P(\text{colorectal}|\text{colorectal cancer}) \times P(\text{cancer}|\text{colorectal cancer})$, effectively double-counting the cancer signal.

2. Context Window Limitations

Our ±5 word window is arbitrary and fixed. Some examples need more context:

”Patients underwent screening for CC ... [20 words] ... using colonoscopy”

The discriminative keyword ”colonoscopy” falls outside our window. Neural models with attention can dynamically attend to distant keywords. Naive Bayes cannot.

3. Inability to Handle Negation

Consider:

- ”No evidence of CC” (cancer absent)
- ”CC confirmed” (cancer present)

Naive Bayes treats ”no”, ”evidence”, ”confirmed” as independent features. It cannot understand that ”no” negates the following terms. This requires compositionality that bag-of-words models lack.

4. Semantic Similarity Blindness

Naive Bayes cannot recognize that ”bowel” and ”colon” are semantically similar. If training data uses ”colon” but test data uses ”bowel”, the model treats them as unrelated. Neural models with word embeddings can capture this similarity.

8.3 Where Do Errors Come From?

We analyzed the 7,210 test errors to understand the root causes:

Error Category	Count	% of Errors
Overlapping terminology	3,247	45.0%
Insufficient context	1,894	26.3%
Semantic similarity confusion	1,442	20.0%
Rare/unseen keywords	627	8.7%

Table 5: Error analysis by category. Nearly half of errors stem from overlapping medical terminology across classes.

Could we fix these errors?

Overlapping terminology: Require more distinctive features. Possible solutions:

- Increase context window to capture distant discriminative keywords
- Use TF-IDF with domain-specific stopword list (not generic stopwords)
- Weight features by discriminative power (e.g., mutual information)

Insufficient context: Cannot fix with current model. Need:

- Dynamic window sizing based on document structure
- Sentence-level context (full sentence, not fixed window)
- Document-level context (other abbreviations in same document)

Semantic similarity: Cannot fix with bag-of-words. Need:

- Word embeddings (Word2Vec, FastText)
- Pre-trained language models (BERT, BioBERT)
- Entity linking to medical ontologies (UMLS)

Rare keywords: Could partially fix with:

- Better smoothing (not just Laplace)
- Backing off to character n-grams
- Transfer learning from related medical tasks

9 Conclusion

Classical machine learning achieves reasonable medical abbreviation disambiguation. Naive Bayes achieved 99.63% on synthetic and 78.80% on real medical text, approaching deep learning while maintaining interpretability and efficiency.

The 21% gap reveals violated assumptions: overlapping terminology (45%), insufficient context (26%), semantic similarity (20%). TF-IDF decreased accuracy by 0.35%, showing common medical terms carry discriminative power in aggregate patterns.

The small gap to deep learning (78.80% vs. 82–84%) suggests keyword-driven tasks benefit less from sophisticated architectures. The remaining gap likely stems from inherent ambiguity requiring external knowledge.

Future directions: domain-specific stopword lists, dynamic context windows, medical ontology integration, specialty-specific models. For deployment, interpretability and efficiency make this suitable for real-time clinical support with confidence thresholding and active learning essential for handling ambiguous cases.

References

[1] Wen, Z., Lu, X. H., & Reddy, S. (2020). MeDAL: Medical Abbreviation Disambiguation Dataset for Natural Language Understanding Pretraining. *Proceedings of the 3rd Clinical Natural Language Processing Workshop*, 130–135.

[2] The Joint Commission. (2004). Sentinel Event Alert: Official “Do Not Use” List.