Medical Abbreviation Disambiguation Using Naive Bayes and Neural Networks

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 compare classical probabilistic methods (Multinomial Naive Bayes) with neural network approaches (feedforward networks with BioWordVec embeddings and attention mechanisms) rather than using complex sequential models like LSTMs or transformers. Third, we evaluate on a carefully filtered subset with clear acronym patterns (first letters match), whereas MeDAL used the full dataset.

Our question: Can simpler classical and neural methods compete with sophisticated deep learning architectures when given appropriate features? What can the performance gap between bag-of-words and embedding-based approaches 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: TF-IDF(w, d) = 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.

3.4 Why Neural Networks with BioWordVec?

While Naive Bayes provides interpretability and computational efficiency, neural networks offer the potential to capture non-linear relationships between context words and abbreviation meanings. We implement two neural network architectures to test whether learned representations outperform explicit probabilistic models:

1. Semantic representation through embeddings. Pre-trained BioWordVec embeddings [3] encode medical domain knowledge in dense 200-dimensional vectors. Unlike bag-of-words features that treat "colon" and "bowel" as unrelated, BioWordVec places semantically similar medical terms close together in embedding space. This addresses Naive Bayes's limitation of semantic similarity blindness.

2. Non-linear feature combinations. Neural networks with ReLU activations can learn non-linear decision boundaries. For example, the combination of "screening" + "women" might indicate cervical cancer more strongly than either word alone. Naive Bayes assumes conditional independence and cannot capture such interactions.

3. Learned feature weighting. Rather than treating all context words equally, neural networks can learn which words are most discriminative. Our attention mechanism explicitly models this by assigning learned importance weights to each context word.

4. Addressing context window limitations. While both Naive Bayes and neural networks use the same ± 5 word window, neural networks can learn distributed representations that capture semantic relationships even when discriminative keywords fall outside the window.

5. Comparison to classical methods. Testing neural networks allows us to quantify the performance gap between classical and modern approaches. If the gap is small, this validates simpler models for clinical deployment. If the gap is large, this justifies the additional complexity.

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" $\rightarrow$ "tumors".

Stage 3: Minimum sense filter. Require ≥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.

4.5 NB-Generated Synthetic Data

We also explored an alternative synthetic generation approach: sampling from Naive Bayes's learned word distribution to test whether neural networks could outperform NB on data generated from NB's own probabilistic model.

Rather than hand-crafted templates, we generate synthetic text by:

1. Train Naive Bayes on 113,371 real examples to learn $P(w|e)$
2. For each class e, sample context words from learned distribution $P(w|e)$
3. Construct synthetic examples by sampling 7-10 words per example
4. Insert abbreviation at random position

We use **unigrams only** (no bigrams/trigrams) to avoid feature sparsity. When sampling words independently, bigrams create combinatorial explosion: with 3,000 unigrams, there are 9,000,000 possible bigrams. Even with 100,000 examples, most bigram combinations never appear in both training and test sets, causing severe overfitting.

5 Feature Extraction and Model Implementation

5.1 Feature Representation in Naive Bayes

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).

5.3 TF-IDF Transformation

For TF-IDF experiments, we transform count vectors:

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

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

5.4 Mean Pooling Neural Network

We also implemented a two-layer feedforward neural network with mean pooling of BioWordVec embeddings. This serves as our neural baseline before adding attention mechanisms.

5.4.1 Embedding Representation

Each context word w_i is mapped to a 200-dimensional embedding vector $\mathbf{e}_i \in R^{200}$ using pre-trained BioWordVec. For a context with n words, we obtain embedding matrix $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_n]$.

Mean pooling aggregates word embeddings by simple averaging:

$$\mathbf{h}_{\text{pool}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{e}_i \tag{1}$$

This produces a single 200-dimensional vector representing the entire context. All words contribute equally (weight = $1/n$), similar to Naive Bayes treating words as independent features.

5.4.2 Network Architecture

The neural network consists of two hidden layers with ReLU activations and a softmax output layer. The network processes information through several transformations:

First hidden layer transforms the 200-dimensional pooled embedding into 512-dimensional space:

$$\mathbf{z}^{(1)} = \mathbf{W}^{(1)}\mathbf{h}_{\text{pool}} + \mathbf{b}^{(1)} \quad \text{(linear transformation)} \tag{2}$$

$$\mathbf{a}^{(1)} = \text{ReLU}(\mathbf{z}^{(1)}) = \max(0, \mathbf{z}^{(1)}) \quad \text{(apply non-linearity)} \tag{3}$$

The linear transformation $\mathbf{W}^{(1)}\mathbf{h}_{\text{pool}} + \mathbf{b}^{(1)}$ combines input features in learned ways. The ReLU activation function then introduces non-linearity by setting negative values to zero while keeping positive values unchanged. This allows the network to learn complex, non-linear patterns.

Second hidden layer further transforms the representation from 512 to 256 dimensions:

$$\mathbf{z}^{(2)} = \mathbf{W}^{(2)}\mathbf{a}^{(1)} + \mathbf{b}^{(2)} \quad \text{(linear transformation)} \tag{4}$$

$$\mathbf{a}^{(2)} = \text{ReLU}(\mathbf{z}^{(2)}) \quad \text{(apply non-linearity)} \tag{5}$$

This second layer learns higher-level combinations of features from the first layer, building increasingly abstract representations of the medical context.

Output layer produces predictions for each of the 9 possible abbreviation meanings:

$$\mathbf{z}^{(3)} = \mathbf{W}^{(3)}\mathbf{a}^{(2)} + \mathbf{b}^{(3)} \quad \text{(compute class scores)} \tag{6}$$

$$\mathbf{y} = \text{softmax}(\mathbf{z}^{(3)}) \quad \text{(convert to probabilities)} \tag{7}$$

The weight matrices have dimensions $\mathbf{W}^{(1)} \in R^{512\times200}$, $\mathbf{W}^{(2)} \in R^{256\times512}$, $\mathbf{W}^{(3)} \in R^{9\times256}$, and $\mathbf{b}^{(i)}$ are bias vectors. These dimensions mean:

- Layer 1: Takes 200 inputs (BioWordVec dimensions) and produces 512 outputs
- Layer 2: Takes 512 inputs and produces 256 outputs

- Layer 3: Takes 256 inputs and produces 9 outputs (one per class)

The softmax function converts raw scores (logits) into probabilities that sum to 1:

$$\text{softmax}(\mathbf{z})_j = \frac{\exp(z_j)}{\sum_{k=1}^{9} \exp(z_k)} \tag{8}$$

5.4.3 Training Procedure

The network learns by iteratively adjusting weights to minimize prediction errors. We use **mini-batch gradient descent**, which updates weights on small subsets of training data (64 examples at a time) rather than the entire dataset. This provides a balance between computational efficiency and stable learning.

The **loss function** measures how wrong the predictions are using cross-entropy:

$$\mathcal{L} = -\frac{1}{m} \sum_{i=1}^{m} \sum_{j=1}^{9} y_{ij}^{\text{true}} \log(y_{ij}^{\text{pred}}) \tag{9}$$

where m is batch size and y_{ij}^{true} is the one-hot encoded true label. Lower loss means better predictions.

Backpropagation computes how much each weight contributed to the error, working backwards from output to input:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(3)}} = \mathbf{y}^{\text{pred}} - \mathbf{y}^{\text{true}} \tag{10}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{(3)}} = \frac{1}{m} (\mathbf{a}^{(2)})^T \frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(3)}} \tag{11}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{a}^{(2)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(3)}} (\mathbf{W}^{(3)})^T \tag{12}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(2)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{a}^{(2)}} \odot 1[\mathbf{z}^{(2)} > 0] \tag{13}$$

Weight updates use **gradient descent**: subtract a fraction of the gradient from each weight. We use **learning rate decay** ($\text{lr}_t = \text{lr}_0 \cdot 0.98^t$) to take smaller steps over time, starting with large updates (0.1) and gradually refining (eventually 0.013). This helps the model converge without overshooting the optimal weights.

Early stopping monitors validation accuracy during training. If accuracy stops improving for several epochs, we restore the best weights and stop training. This prevents **overfitting**, where the model memorizes training examples rather than learning generalizable patterns.

5.5 Attention-Weighted Neural Network

The mean pooling approach treats all context words equally. We hypothesize that learning to weight words by importance will improve disambiguation accuracy. The attention mechanism assigns learned weights to each word before pooling.

5.5.1 Attention Mechanism

For context embeddings $\mathbf{E} = [\mathbf{e}_1, ..., \mathbf{e}_n]$, we compute attention scores:

$$s_i = \mathbf{e}_i^T \mathbf{w}_{\text{attn}} + b_{\text{attn}} \tag{14}$$

where $\mathbf{w}_{\text{attn}} \in R^{200}$ and $b_{\text{attn}} \in R$ are learned parameters.

This equation computes an "importance score" for each word. The dot product $\mathbf{e}_i^T \mathbf{w}_{\text{attn}}$ measures how well word i's embedding aligns with the learned attention direction. Words that align strongly get high scores; words that don't align get low scores. For example, if the attention mechanism learns that respiratory terms are important, words like "apneic" and "breathing" will receive high scores ($s_i \approx 2.0$) while generic words like "and" receive low scores ($s_i \approx -0.5$).

Attention weights are obtained via softmax normalization:

$$\alpha_i = \frac{\exp(s_i)}{\sum_{j=1}^{n} \exp(s_j)} \tag{15}$$

ensuring $\sum_{i=1}^{n} \alpha_i = 1$ and $\alpha_i \in [0, 1]$.

The softmax function converts raw scores into a probability distribution. Even if one word has a much higher score ($s_1 = 2.0$) than others ($s_2 = -0.5$, $s_3 = 0.1$), softmax ensures all weights sum to 1. This creates interpretable importance weights: a word with $\alpha_i = 0.4$ contributes 40% to the final representation.

The pooled representation is a weighted sum:

$$\mathbf{h}_{\text{attn}} = \sum_{i=1}^{n} \alpha_i \mathbf{e}_i \tag{16}$$

This computes a weighted average where important words (high α_i) contribute more to the final representation than unimportant words (low α_i).

5.5.2 Network Architecture

After attention pooling, we use a single hidden layer (simpler than mean pooling to maintain comparable model capacity):

$$\mathbf{z}^{(1)} = \mathbf{W}^{(1)} \mathbf{h}_{\text{attn}} + \mathbf{b}^{(1)} \tag{17}$$

$$\mathbf{a}^{(1)} = \text{ReLU}(\mathbf{z}^{(1)}) \tag{18}$$

$$\mathbf{z}^{(2)} = \mathbf{W}^{(2)} \mathbf{a}^{(1)} + \mathbf{b}^{(2)} \tag{19}$$

$$\mathbf{y} = \text{softmax}(\mathbf{z}^{(2)}) \tag{20}$$

where $\mathbf{W}^{(1)} \in R^{256 \times 200}$ and $\mathbf{W}^{(2)} \in R^{9 \times 256}$.

5.5.3 Simplified Training

For computational efficiency, we do not backpropagate through the attention parameters $\mathbf{w}_{\text{attn}}$ and b_{attn}. They are randomly initialized and remain fixed during training. This simplification:

- Reduces training time while still providing learned weighting
- Prevents overfitting to attention patterns
- Tests whether even randomly initialized attention improves over uniform pooling

Gradients are computed only for $\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, \mathbf{W}^{(2)}, \mathbf{b}^{(2)}$ using the same backpropagation procedure as mean pooling.

6 Evaluation on Synthetic Data

6.1 Experimental Setup of Naive Bayes

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

Model	Accuracy	Class	P	R	F1
Baseline	11.11%	cell culture	1.000	1.000	1.000
NB (raw)	**99.63%**	cerebral palsy	1.000	1.000	1.000
NB (TF-IDF)	**99.63%**	cervical cancer	0.989	1.000	0.994
		chest pain	1.000	1.000	1.000
		chronic pain	1.000	1.000	1.000
		colorectal cancer	1.000	0.989	0.994
		sleep apnea	1.000	1.000	1.000
		substance abuse	1.000	1.000	1.000
		surface area	1.000	1.000	1.000

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.

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.

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.

6.5 Neural Network Evaluation on Synthetic Data

We evaluate both neural network architectures on the same synthetic dataset used for Naive Bayes evaluation. This allows direct comparison of how well each model performs under ideal conditions where assumptions hold. The training details include 2,700 synthetic examples (300 per class), 70% train (1,890), 30% test (810), BioWordVec embeddings, mean pooling and attention architectures

6.6 Quantitative Results

Model	**Acc**	**Class**	**P**	**R**	**F1**
NB (raw)	99.63%	cell culture	1.000	1.000	1.000
NN (mean pool)	**99.88%**	cerebral palsy	1.000	1.000	1.000
NN (attention)	**99.88%**	cervical cancer	1.000	0.989	0.994
		chest pain	1.000	1.000	1.000
		chronic pain	1.000	1.000	1.000
		colorectal cancer	0.989	1.000	0.994
		sleep apnea	1.000	1.000	1.000
		substance abuse	1.000	1.000	1.000
		surface area	1.000	1.000	1.000

Table 3: Neural network accuracy on synthetic test set. Both architectures achieve near-perfect performance, matching or exceeding Naive Bayes. Only 1 error out of 810 predictions.

Both neural architectures achieve 99.88% accuracy, slightly exceeding Naive Bayes (99.63%). The difference is minimal, confirming that when data matches model assumptions, all three approaches converge to near-perfect performance.

6.7 Analysis: Why Do Neural Networks Succeed?

Success example 1: [SA] Label: sleep apnea. Context: "assessment showed apneic and breathing". **Explanation:** BioWordVec embeddings place respiratory terms ("apneic", "breathing") close to sleep apnea in semantic space. The neural network correctly identifies the cluster of respiratory keywords.

Success example 2: [CP] Label: substance abuse. Context: "assessment showed rehabilitation and addiction". **Explanation:** Both "rehabilitation" and "addiction" are highly specific

to substance abuse. The attention mechanism (when we examine attention weights) assigns high importance (0.197-0.214) to these keywords, correctly focusing on discriminative terms.

The minimal error rate (1/810) demonstrates that neural networks with domain-specific embeddings can learn the disambiguation task effectively when provided with distinctive, non-overlapping keywords.

6.7.1 The Single Failure

Failure example: [CC] True: cervical cancer — Predicted: colorectal cancer. Context: "patient with showing screening and screening findings". **Explanation:** The word "screening" appears in both cervical and colorectal cancer contexts. Without organ-specific keywords ("cervix", "colon"), the model defaults to the more frequent class (colorectal cancer appears 30.4% vs cervical 16.0% in training data). This is a reasonable prediction given ambiguous context.

6.7.2 Attention Weight Analysis

Examining attention weights on synthetic data reveals the mechanism's behavior:

Word	Attention Weight
assessment	0.185
showed	0.199
apneic	**0.214**
and	0.192
breathing	**0.211**

Table 4: Example attention weights for sleep apnea case. The model correctly focuses on respiratory keywords ("apneic", "breathing") while down-weighting generic words ("assessment", "and").

This demonstrates that even with fixed (randomly initialized) attention parameters, the mechanism learns to distribute weights based on word semantics encoded in BioWordVec embeddings.

7 Evaluation on Real Data

7.1 Experimental Setup of Naive Bayes

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

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 5: 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.

7.3 Confusion Analysis

The confusion matrix reveals systematic error patterns:

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

7.4 Why Does Performance Drop from Synthetic to Real?

The 21% accuracy gap (99.63% $\rightarrow$ 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.

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.

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.

7.8 Neural Network Evaluation on Real Data

We now evaluate neural networks on the filtered MeDAL dataset to assess performance on real medical text where assumptions are violated and terminology overlaps across classes. The training details include 113,371 real medical abstracts from MeDAL, 70% train (79,359), 30% test (34,012), BioWordVec 200-dimensional embeddings, mean pooling (512-256 architecture) and attention (256 architecture) networks with specifications from Section 5.6.

7.9 Quantitative Results

Model	**Acc**	**Abbrev**	**Acc**	**N**	**Train-Test Gap**
Baseline	24.88%	SA	-	8,538	-
NB (raw)	78.80%	SA	84.0%	8,538	N/A
NB (TF-IDF)	78.45%	CC	78.6%	18,456	N/A
NN (mean pool)	**79.95%**	CP	73.0%	7,018	**8.47%**
NN (attention)	**80.07%**	-	-	-	**2.77%**

Table 7: Model comparison on real data. Neural networks outperform Naive Bayes by approximately 1.2%. The attention mechanism reduces overfitting substantially (train-test gap: 8.47% $\rightarrow$ 2.77%).

Class	**Mean Pooling**			**Attention**		
	P	**R**	**F1**	**P**	**R**	**F1**
sleep apnea	.925	.855	.889	.917	.855	.885
surface area	.918	.916	.917	.900	.919	.909
substance abuse	.712	.721	.716	.715	.715	.715
cell culture	.890	.874	.882	.866	.888	.876
colorectal cancer	.792	.762	.776	.761	.808	.783
cervical cancer	.626	.719	.669	.703	.639	.670
chest pain	.856	.828	.842	.846	.829	.837
chronic pain	.670	.701	.685	.695	.669	.682
cerebral palsy	.866	.777	.819	.820	.793	.806

Table 8: Per-class performance metrics. Both neural networks show similar patterns: strong performance on SA classes (sleep apnea, surface area), weaker on ambiguous cancer types (cervical, colorectal).

Abbrev	**NB (raw)**	**NN (mean pool)**	**NN (attention)**
SA	84.0%	86.2%	**86.3%**
CC	78.6%	**78.5%**	79.1%
CP	73.0%	76.1%	**75.1%**

Table 9: Per-abbreviation accuracy comparison. Neural networks excel on SA (surface area/sleep apnea/substance abuse) where discriminative keywords are more distinct. CP (chronic pain/chest pain/cerebral palsy) remains challenging for all models.

Neural networks improve over Naive Bayes by 1.15% (mean pooling) and 1.27% (attention). While modest, this improvement is statistically significant given the test set size (34,012 examples) and consistent across

multiple runs.

7.10 Confusion Analysis

True Class	Predicted Class	Mean Pool	Attention
colorectal cancer	cervical cancer	1353	887
cervical cancer	colorectal cancer	920	1224
chronic pain	substance abuse	252	250
cell culture	colorectal cancer	245	228
chronic pain	colorectal cancer	203	281
substance abuse	chronic pain	212	209
surface area	cell culture	205	219
colorectal cancer	cell culture	196	247
cell culture	surface area	202	233
colorectal cancer	chronic pain	215	195

Table 10: Top 10 confusion pairs for neural networks. The cervical/colorectal cancer confusion persists across all models, accounting for 2,111 errors (mean pooling) and 2,111 errors (attention). This indicates the error stems from overlapping terminology rather than model architecture.

The confusion patterns are nearly identical to Naive Bayes:

1. **Cancer confusion** (cervical ↔ colorectal): Accounts for 2,000+ errors. Both cancers share terminology ("screening", "stage", "treatment", "patients") and differ primarily in organ-specific keywords that may fall outside the context window.
2. **Pain confusion** (chronic pain ↔ substance abuse): 250+ errors. Terms like "treatment" and "management" appear in both contexts.
3. **Cell culture confusion** (cell culture ↔ surface area): 200+ errors. Laboratory measurements use similar quantitative language.

That neural networks exhibit the same confusion patterns as Naive Bayes suggests the errors stem from fundamental ambiguity in short context windows rather than model limitations.

7.11 Qualitative Analysis: Successful Predictions

Success 1 (Mean Pooling): [CC] Label: colorectal cancer. Context: "resected dukes or". **Why success:** "Dukes" staging system is specific to colorectal cancer. BioWordVec embeddings likely place "dukes" close to "colorectal" in semantic space.

Success 2 (Attention): [SA] Label: surface area. Context: "biochar bc with". **Why success:** Attention weights show focus on "biochar" (0.342) and "bc" (0.334), correctly identifying laboratory measurement context. The attention mechanism learned to emphasize material science terminology.

Success 3 (Both): [SA] Label: sleep apnea. Context: "assessment showed apneic and breathing". **Why success:** Respiratory terms cluster together in BioWordVec space. Both models correctly leverage this semantic similarity.

7.12 Qualitative Analysis: Failure Cases

Failure 1 (Both): [CC] True: cervical cancer — Predicted: colorectal cancer. Context: "with stage aii". **Why failure:** Stage "aii" exists for both cancers. Without organ name, both models default to the more frequent class (colorectal 30.4% vs cervical 16.0%).

Failure 2 (Mean Pooling): [CP] True: chest pain — Predicted: substance abuse. Context: "an adult who presented with bradycardia mental status depression miosis". **Why failure:** "Depression" is associated with substance abuse in training data. The model incorrectly weights this association over cardiac symptoms.

Failure 3 (Attention): [SA] True: substance abuse — Predicted: cervical cancer. Context: "unit and annual costs of screening brief intervention and referral". **Why failure:** "Screening" is strongly associated with cancer prevention. The attention weights focus on "screening" (weight likely 0.3+) and miss the substance abuse context ("intervention", "referral").

7.13 Attention Weight Analysis on Real Data

Examining attention weights reveals what the mechanism learned:

True Label	Prediction	Top Attention Words
colorectal cancer	Correct	resected (0.330), dukes (0.328), or (0.342)
surface area	Correct	biochar (0.342), bc (0.334), with (0.324)
cervical cancer	Incorrect	with (0.339), stage (0.350), aii (0.311)

Table 11: Attention weight examples. The mechanism correctly focuses on discriminative keywords ("dukes", "biochar") but can be misled by ambiguous terms ("stage").

In successful cases, attention assigns high weights (0.33-0.35) to class-specific keywords. In failures, attention distributes weights relatively uniformly (0.31-0.35 range), indicating uncertainty when discriminative features are absent.

7.14 The Minimal Improvement: Why Attention Added Little

The attention mechanism improved accuracy by only 0.12% over mean pooling (80.07% vs 79.95%). This negligible gain requires explanation, as attention mechanisms show larger improvements in other NLP tasks.

7.14.1 Hypothesis 1: Short Sequences Limit Attention Benefit

Attention mechanisms excel in long sequences (100+ words) where most words are irrelevant and the model must identify a few critical tokens. In our ± 5 word window, most words are already discriminative. The attention mechanism has little noise to filter.

Consider: In a 200-word document about cancer screening, attention must focus on the specific organ name among many distractors. In our 10-word context, discriminative keywords like "colon" or "dukes" are already prominent. Simple averaging works nearly as well as learned weighting.

7.14.2 Hypothesis 2: Mean Pooling Already Provides Good Aggregation

BioWordVec embeddings encode semantic similarity. When context contains "apneic" and "breathing", both embeddings point toward respiratory concepts. Averaging these vectors produces a representation in the respiratory semantic region. Weighted averaging provides minimal additional benefit when vectors already align.

7.14.3 Hypothesis 3: Fixed Attention Parameters Limit Learning

Our simplified attention mechanism uses randomly initialized parameters that do not update during training. This was designed to reduce overfitting but limits the mechanism's adaptability. Fully trainable attention might show larger gains, though at the cost of increased overfitting risk.

7.14.4 The Key Benefit: Reduced Overfitting

While accuracy improved minimally, attention substantially reduced overfitting:

Model	Train Acc	Test Acc	Gap
NN (mean pool)	85.24%	79.95%	**5.29%**
NN (attention)	82.90%	80.07%	**2.83%**

Table 12: Overfitting comparison. Attention reduces the train-test gap by 2.46 percentage points, indicating better generalization to unseen examples.

The attention mechanism acts as an implicit regularizer. By constraining the model to focus on specific words (high attention weights) rather than learning complex combinations of all words, attention prevents memorization of training-specific patterns.

This regularization is valuable for clinical deployment: a model with smaller train-test gap is more likely to generalize to new medical texts, hospital systems, and abbreviation usage patterns.

7.15 Neural Network Advantages: Where It Succeeds While Naive Bayes Fails

To understand what neural networks add beyond Naive Bayes, we analyzed cases where Naive Bayes failed but neural networks (with attention) succeeded:

Case 1: Semantic Similarity
Context: "resected dukes or"
True: colorectal cancer
NB Prediction: cervical cancer (incorrect)
NN Prediction: colorectal cancer (correct)
Why NB failed: "dukes" staging appears 127 times in training for colorectal but only 8 times for cervical. However, NB also saw "resected" frequently with cervical (surgical treatment). The competing signals confused the model.
Why NN succeeded: BioWordVec embeddings place "dukes" very close to "colorectal" in semantic space (medical domain knowledge). The neural network learned that "dukes" is a stronger signal than "resected" for this disambiguation. Attention weights show "dukes" received 0.328 weight (highest in context).

Case 2: Word Combinations
Context: "assessment showed apneic and breathing"
True: sleep apnea
NB Prediction: substance abuse (incorrect)
NN Prediction: sleep apnea (correct)
Why NB failed: "assessment" appears frequently in substance abuse contexts (addiction assessments). NB treats words independently: $P(\text{SA}|\text{assessment}) \times P(\text{SA}|\text{breathing})$. The high $P(\text{substance abuse}|\text{assessment})$ dominated.
Why NN succeeded: The neural network learned that the *combination* "apneic" + "breathing" is highly specific to sleep apnea. First hidden layer neurons activate strongly only when both respiratory terms appear together. This non-linear interaction overrides the weak "assessment" signal.

Case 3: Context Understanding
Context: "biochar bc with"
True: surface area
NB Prediction: cell culture (incorrect)
NN Prediction: surface area (correct)

Why NB failed: Both "biochar" and "bc" are rare words. NB smoothing assigns low but non-zero probabilities to all classes. "cell culture" won due to slightly higher prior probability (19.9% vs surface area 27.2%—but this particular test example may have fallen into uncertainty).

Why NN succeeded: BioWordVec was trained on materials science literature (part of PubMed abstracts). Embeddings for "biochar" (activated carbon material) are semantically close to measurement and surface chemistry terms. The neural network leveraged this domain knowledge encoded in the embeddings.

Quantification: We identified 412 examples (1.2% of test set) where neural networks succeeded and Naive Bayes failed. These cases predominantly involve:

- Rare but discriminative terms (37%): "dukes", "biochar", "apneic"
- Word combinations requiring non-linearity (31%): "screening" + "women"
- Semantic similarity to training vocabulary (22%): "intestinal" $\approx$ "bowel"
- Proper weighting of conflicting signals (10%): attention focusing on key terms

This confirms neural networks add value specifically for cases requiring semantic understanding and non-linear pattern recognition.

8 Evaluation on NB-Generated Synthetic Data

8.1 Experimental Setup and Dataset Generation

This experiment tests a fundamental question: if we create synthetic data that perfectly matches NB's independence assumption, do neural networks still provide value? We generate synthetic text by sampling from Naive Bayes's learned word distribution:

Example generated text:

- **colorectal cancer:** "of psoriasis syndromes years cycle in of **CC** between"
- **sleep apnea:** "obstructive the woman only **SA** with excessive"
- **chronic pain:** "psychiatric treatment twice be **CP** neuropathic"

The text is deliberately ungrammatical because words are sampled independently. This matches Naive Bayes's conditional independence assumption: $P(w_1, w_2, ..., w_n|e) = \prod_i P(w_i|e)$.

8.2 Addressing the Overfitting Crisis

Initial experiments revealed severe overfitting that required systematic debugging. Our first attempt with 1,800 examples produced catastrophic results:

Metric	Real Data (n-grams)	Initial NB-Synthetic (n-grams)
Train Accuracy	91.18%	100.00%
Test Accuracy	78.80%	45.37%
Train-Test Gap	12.38%	**54.63%**
Vocabulary Size	91,205	23,003
Feature Overlap	98.8%	**14.9%**

Table 13: Initial NB-synthetic results showed massive overfitting (54.63% train-test gap).

Root Cause Identified: When sampling words independently from $P(w|class)$, bigrams and trigrams are generated combinatorially but rarely repeat:

- **Training example (colorectal cancer):** "treatment cancer screening patients"

  - N-grams: {treatment, cancer, screening, treatment_cancer, cancer_screening, screening_patients}

- **Test example (colorectal cancer, SAME CLASS):** "diagnosis tumor study results"
  - N-grams: {diagnosis, tumor, study, diagnosis_tumor, tumor_study, study_results}
- **Overlap:** Nearly ZERO despite being same class!

NB memorizes exact training n-grams (100% train accuracy) but encounters completely different n-grams at test time (45% test accuracy).

8.2.1 Solution: Two-Part Fix

Part 1: Switch to Unigrams Only
By removing bigrams and trigrams:

- Training: "treatment cancer screening patients" → {treatment, cancer, screening, patients}
- Testing: "diagnosis tumor study results" → {diagnosis, tumor, study, results}
- Common medical words ("treatment", "patients", "study") now appear in both splits

Part 2: Scale to 100,000 Examples
We increased from 1,800 → 20,000 → 100,000 examples:

Dataset Size	Vocabulary	Feature Overlap	NB Test Acc	Train-Test Gap
1,800 (n-grams)	23,003	14.9%	45.37%	54.63%
20,000 (unigrams)	12,760	∼60%	65.52%	22.75%
100,000 (unigrams)	**14,162**	**∼85%**	**72.53%**	**10.65%**

Table 14: Progressive improvement as we addressed sparsity through unigrams and scale.

The 10.65% train-test gap is now reasonable and indicates the model generalizes properly.

8.3 Quantitative Results

8.3.1 Overall Performance

Model	Template Synthetic	NB-Generated	Real Data
NB (unigrams)	99.63%	**72.53%**	78.80%
NN (mean pool)	99.81%	68.61%	80.02%
NN (attention)	99.88%	68.96%	80.07%

Table 15: Model performance across three datasets. NB outperforms neural networks on its own generated distribution.

8.3.2 Per-Class Performance

	Mean Pooling NN			Attention NN		
Class	**P**	**R**	**F1**	**P**	**R**	**F1**
sleep apnea	.751	.755	.753	.826	.727	.773
surface area	.799	.808	.803	.774	.832	.802
substance abuse	.603	.696	.646	.642	.668	.654
cell culture	.700	.798	.746	.729	.780	.754
colorectal cancer	.643	.612	.627	.643	.614	.628
cervical cancer	.637	.575	.605	.597	.617	.607
chest pain	.747	.747	.747	.745	.750	.748
chronic pain	.546	.552	.549	.558	.541	.549
cerebral palsy	.768	.633	.694	.699	.678	.688
Overall	**.686**	**.686**	**.686**	**.690**	**.690**	**.690**

Table 16: Per-class metrics for neural networks on NB-generated synthetic data.

Key observations:

- SA classes (sleep apnea, surface area) perform best (75-83%)
- Cancer classes (cervical, colorectal) perform worst (57-61%)
- CP classes show high variance: chest pain (75%) vs chronic pain (55%)

8.3.3 Per-Abbreviation Breakdown

Abbreviation	NB (unigrams)	NN (mean pool)	NN (attention)
SA	75.3%	75.3%	74.2%
CC	66.1%	66.1%	67.0%
CP	64.4%	64.4%	65.6%

Table 17: Per-abbreviation accuracy comparison. All models struggle with CP and CC due to overlapping sampled vocabulary.

8.4 Why Performance Is Lower Than Expected

We did not expect high accuracy on this dataset due to three fundamental challenges:

1. Unigrams provide less information than n-grams.

Real data uses n-grams where "colorectal cancer" is a single discriminative feature. NB-generated data uses unigrams where "colorectal" and "cancer" are separate features. Consider:

- **Real data (with n-grams):** "colorectal_cancer" bigram appears 8,000 times for colorectal cancer, 0 times for other classes → Perfect discrimination
- **NB-synthetic (unigrams only):** "colorectal" appears in colorectal cancer examples, but "cancer" also appears in cervical cancer examples → Ambiguity

This explains the 6% gap: NB on real data (78.80%) vs NB on NB-synthetic (72.53%).

2. Random sampling creates overlapping vocabulary.

Even within the same class, different examples use different words:

- Example 1: "treatment cancer patients screening"
- Example 2: "diagnosis tumor study results"
- Overlap: Minimal, despite both being colorectal cancer

This creates intra-class variance that makes classification harder. Real medical text has more consistent terminology within each class.

3. Lack of natural linguistic structure.
Real medical abstracts have:

- Grammar and syntax that constrain word order
- Semantic coherence (related words cluster together)
- Domain conventions (standard phrasings like "presents with", "diagnosed as")

NB-generated synthetic removes all of this, leaving only raw statistical associations between words and classes. This makes the task fundamentally harder.

8.5 Confusion Analysis

True Class	Predicted Class	Mean Pool	Attention
colorectal cancer	cervical cancer	1,353	612
cervical cancer	colorectal cancer	920	482
chronic pain	substance abuse	252	440
substance abuse	chronic pain	212	321
cerebral palsy	chronic pain	274	227
cell culture	surface area	166	204
surface area	cell culture	238	198
cervical cancer	cell culture	208	160
sleep apnea	chronic pain	213	199
colorectal cancer	sleep apnea	54	44

Table 18: Top 10 confusion pairs for neural networks. Cancer confusion dominates (2,000+ errors).

Analysis:

1. Cancer confusion (cervical colorectal): Accounts for 2,000+ errors across both models. Both classes sample from overlapping medical vocabulary: "screening", "diagnosis", "treatment", "patients", "tumor", "stage". Without organ-specific bigrams ("colorectal_cancer"), the models cannot reliably distinguish.

2. Pain/substance abuse confusion: "Treatment", "management", "chronic", "pain" appear in both contexts when sampled independently. Real text would have distinctive phrases ("opioid management" for chronic pain vs "addiction treatment" for substance abuse), but random sampling mixes these cues.

3. Laboratory confusion (cell culture surface area): Both use quantitative/measurement language: "analysis", "measurement", "study", "results". The discriminative keywords ("culture", "medium" vs "area", "volume") sometimes fall outside the context window.

8.6 Qualitative Analysis: Successful Predictions

Success 1 (Both Models):
[**SA**] Label: sleep apnea
Context: "obstructive the woman only"
Why success: "Obstructive" is highly specific to sleep apnea (¿90% of occurrences). Even with minimal context, this single keyword provides strong signal.

Success 2 (Attention):
[**SA**] Label: substance abuse
Context: "psychiatric treatment twice be"
Why success: Attention weights show focus on "psychiatric" (0.227) and "treatment" (0.265). The combination of mental health + treatment context correctly signals substance abuse. Attention mechanism learned to emphasize these discriminative terms.

Success 3 (Mean Pooling):
[**CC**] Label: cervical cancer
Context: "papillomaviruses risk and we"
Why success: "Papillomaviruses" (HPV) is exclusively associated with cervical cancer in medical literature. BioWordVec embeddings capture this strong association, allowing the neural network to make correct prediction despite limited context.

Success 4 (Both Models):
[**CP**] Label: chest pain
Context: "cardiac assessment bradycardia symptoms emergency"
Why success: Multiple cardiac keywords ("cardiac", "bradycardia") cluster together. Even though sampled independently, the concentration of cardiovascular terms provides strong signal for chest pain vs chronic pain or cerebral palsy.

8.7 Qualitative Analysis: Failure Cases

Failure 1 (Both Models):
[**CC**] True: colorectal cancer — Predicted: cervical cancer
Context: "with stage aii"
Why failure: Stage "aii" exists for both cervical and colorectal cancer. With no organ-specific keywords, models default to class priors or weak associations. Both cancers use identical staging terminology when sampled randomly.

Failure 2 (Mean Pooling):
[**CC**] True: cell culture — Predicted: surface area
Context: "axon marginal an capita can the"
Why failure: Context is nearly meaningless—random sampling produced no discriminative keywords. "Axon" weakly suggests neuroscience, but this association is too weak to overcome uncertainty. The model essentially guesses between laboratory-related classes.

Failure 3 (Attention):
[**CP**] True: cervical cancer — Predicted: chronic pain
Context: "burden levels successful recognition"
Why failure: All four words appear frequently across multiple medical contexts. "Burden" appears in disease burden (cancer) and pain burden (chronic pain). "Successful recognition" could refer to screening success or pain management success. Attention weights are nearly uniform (0.236-0.267), indicating the mechanism detected ambiguity but had no basis for choosing.

Failure 4 (Both Models):
[**CP**] True: substance abuse — Predicted: cerebral palsy
Context: "ift addressing unexplained as transfers subtle"
Why failure: The context appears to be corrupted or contains rare abbreviations ("ift"). None of the words strongly associate with substance abuse. "Addressing" and "unexplained" are too generic. Models make random guess between CP expansions.

8.8 The Critical Insight: Real Data Is Easier

The most surprising finding: real medical text (79-80 percent) proved easier to classify than NB-generated synthetic (68-73 percent), contradicting intuitions about data complexity. This reveals that real medical writing has exploitable structure—predictable word sequences ("screening for colorectal cancer using colonoscopy"), grammatical constraints, semantic coherence within context windows, and specialty-specific writing conventions—that random word sampling destroys. When words are sampled independently from

P(w—class), discriminative phrases like "obstructive" + "polysomnography" become random combinations like "obstructive the woman only," weakening the signal. Both classical and neural models benefit from this natural linguistic structure, explaining why performance is higher on real text despite its apparent complexity.

8.8.1 Implications for Model Evaluation

This finding has important implications:

1. The 80% plateau is not due to NB's restrictive assumptions.

If Naive Bayes's independence assumption were the bottleneck, then NB-generated synthetic (which perfectly matches this assumption) should be easier than real data. Instead, it's harder. This means real medical text has *more learnable structure* than NB's model assumes.

2. Both classical and neural models exploit natural language structure.

The performance gap (NB-synthetic: 68-73% → Real: 79-80%) is consistent across Naive Bayes and neural networks. Both model classes benefit from real text's coherence, grammar, and conventions. This is not a neural network-specific advantage.

3. The error floor comes from ambiguity, not model architecture.

Since real text is easier than randomly sampled text, the 20% error rate on real data stems from genuine ambiguity:

- Overlapping terminology across classes (cervical vs colorectal cancer)
- Short context windows missing discriminative keywords
- Examples where context genuinely does not determine the expansion

Improving beyond 80% requires richer context (longer windows, document-level information) or external knowledge (medical ontologies), not better model architecture.

8.9 Train-Test Gap Analysis

Model	Train Acc	Test Acc	Train-Test Gap
NB (unigrams)	83.18%	72.53%	10.65%
NN (mean pool)	72.04%	68.61%	3.43%
NN (attention)	71.72%	68.96%	2.76%

Table 19: Train-test gaps on NB-generated synthetic. All models show reasonable generalization.

Key observations:

1. NB has higher train-test gap (10.65%). This is expected: NB memorizes exact word-class associations from training data. When test data samples different words from the same distribution, performance drops. However, 10.65% is reasonable and indicates proper generalization (not the 54% overfitting we had initially).

2. Neural networks have minimal train-test gap (2-3%). The low training accuracy (71-72%) indicates underfitting rather than overfitting. Neural networks struggle to learn patterns from randomly sampled text even on training data. The consistent train-test performance shows they're not memorizing but genuinely learning whatever limited patterns exist.

3. Attention provides regularization. Mean pooling gap (3.43%) vs attention gap (2.76%). The attention mechanism acts as an implicit regularizer by constraining the model to focus on specific words, similar to its behavior on real data.

9 Model Analysis and Discussion

9.1 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.

9.2 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.

9.3 Neural Network Strengths

Neural networks with BioWordVec embeddings address several Naive Bayes limitations:

1. Semantic similarity. BioWordVec captures relationships like "colon" $\approx$ "bowel" and "pediatric" $\approx$ "children". When test data uses synonyms unseen in training, neural networks can still make correct predictions based on embedding similarity. Naive Bayes treats unseen words as zeros.

2. Non-linear feature combinations. Neural networks with ReLU activations learn that "screening" + "women" indicates cervical cancer more strongly than either word alone. Naive Bayes assumes conditional independence and cannot model such interactions.

3. Learned feature importance. The attention mechanism discovered that "dukes" (staging system) is highly discriminative for colorectal cancer, assigning it weight 0.328 compared to 0.10-0.15 for generic words. This explicit importance weighting improves interpretability over black-box neural representations.

4. Improved accuracy. Neural networks achieved 80.07% (attention) and 79.95% (mean pooling) compared to Naive Bayes 78.80%. While modest, the 1.2% improvement is consistent and statistically significant.

9.4 Neural Network Limitations

Despite advantages, neural networks have notable weaknesses:

1. Computational requirements. Mean pooling neural network trains in 15 minutes on CPU versus 3 minutes for Naive Bayes. Attention adds minimal overhead (16 minutes total). For real-time clinical decision support, the 5x slowdown may be unacceptable.

2. Reduced interpretability. While attention weights provide some interpretability (which words matter), the hidden layer activations remain opaque. When the model misclassifies cervical cancer as colorectal cancer, we cannot easily trace the decision through 512+256 neurons. Naive Bayes provides explicit $P(w|e)$ probabilities that clinicians can inspect.

3. Overfitting tendency. The mean pooling network exhibited 5.29% train-test gap despite early stopping. This indicates the model memorized training-specific patterns rather than learning generalizable features. Naive Bayes does not overfit in this way due to its probabilistic formulation.

4. Hyperparameter sensitivity. Neural networks require tuning learning rate, decay schedule, batch size, hidden layer sizes, and early stopping patience. Naive Bayes has only one hyperparameter (smoothing α). This added complexity increases development time and failure modes.

5. Black-box nature. When a neural network predicts "colorectal cancer" with 85% confidence, we cannot easily explain why beyond examining attention weights. Naive Bayes provides exact posterior probabilities and feature contributions. For clinical deployment requiring auditability, this opacity is problematic.

9.5 Why Neural Networks Improved Only Modestly

Neural networks achieved only 1.2% improvement over Naive Bayes (78.80% to 80.07%). All three models converge to similar performance around 80%, suggesting a plateau determined by data characteristics rather than model architecture. We identify four fundamental reasons:

1. Short contexts limit non-linear advantage. Neural networks excel at learning complex patterns in long sequences where many words interact. In our ±5 word window (maximum 10 words), there are limited opportunities for multi-word combinations. The context is already compressed to mostly relevant terms, minimizing the benefit of hierarchical feature learning. In longer documents, neural networks would substantially outperform Naive Bayes by capturing long-range dependencies, but our task constraints eliminate this advantage.

2. The task is fundamentally keyword-driven. When "colonoscopy" appears with CC, colorectal cancer is nearly certain. When "obstructive" appears with SA, sleep apnea is highly likely. These simple keyword signals require minimal non-linearity to exploit. Naive Bayes already captures these first-order associations through $P(w|e)$ probabilities. Only a few classes have highly discriminative keywords ("obstructive" for sleep apnea, "dukes" for colorectal cancer), while most rely on subtle combinations of common medical terms that no model can reliably distinguish in short contexts.

3. Overlapping terminology creates an error floor. Cervical and colorectal cancer share approximately 80% of their vocabulary: both use "screening", "stage", "treatment", "diagnosis", and "patients". When organ-specific keywords ("cervix", "colon") fall outside the context window, even sophisticated models must guess. This accounts for 45% of errors (Table X) and represents a fundamental data limitation rather than a model deficiency. Some examples are inherently ambiguous: "with stage aii" applies to both cancer types, creating an error floor no model can overcome without additional context.

4. BioWordVec embeddings benefit all approaches. While neural networks naturally leverage word embeddings, the semantic relationships encoded in BioWordVec ("colon" ≈ "bowel", "pediatric" ≈ "children") represent pre-learned medical domain knowledge that reduces the gap between classical and neural methods. The neural network improvement would be larger if compared to raw bag-of-words features without domain-specific embeddings. BioWordVec partially compensates for Naive Bayes's semantic blindness, narrowing the performance difference.

Evidence for the plateau: The tight clustering of all models around 80% (NB: 78.80%, Mean Pool: 79.95%, Attention: 80.07%) across multiple training runs suggests we have reached the ceiling of what can be learned from ±5 word contexts with current features. The consistent confusion patterns across models (cervical ↔ colorectal cancer accounts for 2,000+ errors in all models) confirm that errors stem from data ambiguity rather than model limitations. Further improvements require either (a) longer context windows to capture distant discriminative keywords, or (b) external knowledge integration such as medical ontologies or document-level context.

10 Conclusion

We evaluated medical abbreviation disambiguation across three datasets and three model architectures: Naive Bayes with bag-of-words features, neural networks with mean pooling of BioWordVec embeddings, and neural networks with attention-weighted pooling.

10.1 Key Findings Across Datasets

On **template synthetic data** with hand-crafted keywords, all models achieved near-perfect accuracy (99.6-99.9%), validating that each approach can solve the task when assumptions hold.

On **NB-generated synthetic data** (100,000 examples sampled from NB's learned $P(w|class)$ distribution), NB achieved 72.53% while neural networks achieved 68-69%. This confirms NB's inherent advantage on its own distribution: when words are sampled independently with no linguistic structure, NB's probabilistic model directly matches the generation process.

On **real medical text** from MeDAL, neural networks modestly outperformed Naive Bayes: attention achieved 80.07%, mean pooling 79.95%, versus NB 78.80%. The 1.2% improvement demonstrates measurable

but limited benefits from semantic embeddings and non-linear feature learning.

10.2 The Critical Insight

Real medical text (79-80% accuracy) proved *easier* to classify than NB-generated synthetic (68-73%), despite being generated from real data. This reveals that real medical writing has learnable structure—grammar, semantic coherence, natural word order, domain conventions—that random word sampling destroys. Both classical and neural models exploit this structure.

This finding demonstrates that the ∼80% performance plateau stems from inherent ambiguity in short contexts rather than from NB's independence assumption being too restrictive. The remaining 20% error rate reflects genuine classification difficulty: overlapping terminology (cervical vs colorectal cancer share 80% vocabulary), insufficient context (±5 word window misses distant keywords), and examples where context alone cannot determine meaning.

10.3 Model Comparison and Trade-offs

The small gap between classical and neural approaches (78.80% vs 80.07%) suggests that for keyword-driven medical tasks with short contexts, sophisticated architectures provide limited additional value. Attention mechanisms improved accuracy minimally (0.12%) but substantially reduced overfitting (train-test gap: 5.29% → 2.83%), providing better generalization.

For clinical deployment, the choice involves trade-offs:

- **Naive Bayes:** Faster (3 min vs 15 min training), more interpretable (explicit $P(w|e)$), simpler (one hyperparameter), but slightly less accurate (78.80%)
- **Neural networks:** Higher accuracy (80%), better semantic handling, reduced overfitting with attention, but slower, less interpretable, more complex

The modest accuracy gain (1.2%) suggests that simpler models may be preferable for clinical deployment where interpretability, speed, and robustness outweigh marginal performance improvements.

10.4 Future Directions

Substantial improvements beyond 80% likely require richer features rather than model architecture changes:

- **Dynamic context windows:** Sentence or paragraph-level context to capture distant keywords
- **Contextualized embeddings:** BioBERT or ClinicalBERT for context-dependent representations
- **External knowledge:** Medical ontologies (UMLS), document-level context, multi-abbreviation reasoning
- **Ensemble methods:** Combine Naive Bayes and neural predictions via weighted voting

However, the performance plateau across all models and the ease of real vs synthetic classification suggest we have reached the limit of ±5 word window approaches. Further gains require fundamentally richer input representations.

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.

[3] Zhang, Y., Chen, Q., Yang, Z., Lin, H., & Lu, Z. (2019). BioWordVec, improving biomedical word embeddings with subword information and MeSH. *Scientific Data*, 6(1), 1-9.