WiDS 2025–26 Midterm Report

Prashant Bothra
24B0334

1. Introduction

Natural Language Processing (NLP) has evolved rapidly from rule-based and statistical methods to deep learning–based architectures capable of understanding context, semantics, and logical relationships in text. The goal of this midterm work was to build a **strong conceptual foundation** in modern NLP while also gaining **hands-on exposure** to implementing models used in real-world applications such as sentiment analysis and natural language inference (NLI).

This report documents my learning journey across three major stages:

1. **Foundations of text representation and sentiment analysis**
2. **Contextual embeddings and the BERT architecture**
3. **Task-specific modeling and fine-tuning pretrained transformers**

The work combines theoretical understanding with practical coding, emphasizing *why* modern models work and *how* they are applied.

---

2. Week 1: Word Embeddings and Sentiment Analysis

2.1 Limitations of Traditional Text Representations

Early NLP systems relied on representations such as **one-hot encoding** and **TF-IDF**, which suffer from two major problems:

- High dimensionality and sparsity
- No notion of semantic similarity

For example, the words *“good”* and *“excellent”* are orthogonal in one-hot space despite having similar meanings.

This motivated the shift toward **dense vector representations**, known as **word embeddings**.

2.2 Word Embeddings: Concept and Intuition

Word embeddings map words into a continuous vector space where semantic relationships are captured geometrically. Models like **Word2Vec** and **GloVe** learn embeddings such that:

- Words appearing in similar contexts have similar vectors
- Semantic relationships can be represented as vector arithmetic
  (e.g., king − man + woman ≈ queen)

However, these embeddings are **static**:
each word has only one vector, regardless of context.

This creates problems with **polysemy**, for example:

- *"bank"* (financial institution)
- *"bank"* (river bank)

2.3 Sentiment Analysis as a Classification Problem

Sentiment analysis involves mapping a text sequence to a discrete label set, typically:

- Positive
- Negative

Using the IMDb movie reviews dataset, the task becomes a supervised learning problem:

f:X→Yf : X \rightarrow Yf:X→Y

where XXX is a variable-length text sequence and YYY is a sentiment label.

2.4 Preprocessing Pipeline

Before feeding text into neural networks, several preprocessing steps are required:

1. **Tokenization** – splitting text into words or subwords
2. **Vocabulary construction** – limiting rare words to control model size
3. **Padding and truncation** – standardizing sequence lengths for batching

This pipeline highlights a key challenge in NLP: models must be invariant to sentence length while remaining sensitive to word order.

---

3. Week 2: From Static to Contextual Embeddings — BERT

3.1 Why Static Embeddings Are Insufficient

Static embeddings treat word meaning as a constant lookup. In contrast, **human language is contextual**, and meaning depends heavily on surrounding words.

BERT reframes embeddings as **functions**, not lookups:

$$v_i = f(w_i \mid context)$$

This enables different representations for the same word in different sentences.

3.2 BERT: Bidirectional Encoder Representations from Transformers

BERT introduced two key innovations:

1. **Bidirectionality**
   Unlike left-to-right language models, BERT reads the entire sentence at once, allowing every token to attend to every other token.

2. **Denoising-based pretraining**
   Instead of predicting the next word, BERT predicts masked words using full context.

3.3 Input Representation in BERT

Each input token embedding is a sum of:

- **Token embeddings** (WordPiece subwords)
- **Segment embeddings** (Sentence A or B)
- **Positional embeddings** (learned position information)

This allows BERT to handle both single-sentence and sentence-pair tasks such as NLI.

3.4 Pretraining Objectives

BERT is trained using two objectives:

Masked Language Modeling (MLM)

- Randomly mask 15% of tokens
- Predict masked tokens using surrounding context
- Encourages deep semantic understanding

Next Sentence Prediction (NSP)

- Predict whether two sentences logically follow each other
- Especially useful for tasks involving sentence relationships

---

3.5 Transformer Self-Attention

The self-attention mechanism allows each token to weigh the importance of other tokens dynamically. This enables:

- Long-range dependency modeling
- Parallel computation

- Superior context handling compared to RNNs

---

4. Week 3: Task-Specific Models and Fine-Tuning

4.1 CNNs for Sentiment Analysis

Convolutional Neural Networks (CNNs) can be applied to text by treating sentences as 1D signals. Key ideas include:

- Filters acting as **n-gram detectors**
- **Max-over-time pooling** to capture the most salient features
- Multi-channel embeddings (static + trainable)

CNNs are particularly effective for sentiment analysis because sentiment-bearing phrases are often sparse but strong.

4.2 Natural Language Inference (NLI)

NLI is a more complex task involving two sentences:

- Premise
- Hypothesis

The goal is to classify their relationship as:

- Entailment
- Contradiction
- Neutral

This task requires **interaction modeling**, not just independent sentence understanding.

4.3 Attention-Based NLI Models

Attention mechanisms explicitly align words in the premise with words in the hypothesis. This enables:

- Soft alignment between related words
- Local comparison of aligned phrases
- Global aggregation of relational evidence

These models demonstrated that attention alone can outperform recurrent architectures.

4.4 Fine-Tuning BERT for NLI

Instead of designing task-specific architectures from scratch, modern NLP uses **transfer learning**:

1. Start with a pretrained BERT model
2. Add a lightweight classification head
3. Fine-tune all parameters on the task dataset

The `[CLS]` token acts as a global sequence representation, making BERT especially effective for classification tasks.

Fine-tuning requires:

- Very small learning rates
- Short training schedules
- Careful optimization to avoid catastrophic forgetting

---

5. Key Learnings and Takeaways

Through this midterm work, I gained:

- A strong understanding of **text representation evolution**
- Conceptual clarity on **why contextual embeddings outperform static ones**
- Practical experience with **CNNs, attention mechanisms, and transformers**
- Insight into **transfer learning and fine-tuning strategies**
- Appreciation for the balance between theory, architecture, and optimization