Multi-Class Text Classification with BERT

M. Nasim Palakka Valappil & Ganesh Yadav Yatham

Advanced Machine Learning
Winter Semester 2025/26

March 3, 2026

Outline

Problem Statement

Task

- **Multi-class text classification** on the 20 Newsgroups dataset (20 newsgroup categories)
- **Open-world extension:** detect inputs that do *not* belong to any of the 20 known categories

Dataset: 20 Newsgroups – Overview

Property	Value
Source	`SetFit/20_newsgroups`
Classes	20 newsgroup categories
Train samples	11 314
Test samples	7 532
Total	18 846

Category groups

- **Computer (5):** `comp.graphics`, `comp.os.ms-windows.misc`, `comp.sys.ibm.pc.hardware`, `comp.sys.mac.hardware`, `comp.windows.x`
- **Recreation (4):** `rec.autos`, `rec.motorcycles`, `rec.sport.baseball`, `rec.sport.hockey`
- **Science (4):** `sci.crypt`, `sci.electronics`, `sci.med`, `sci.space`
- **Politics / Religion (6):** `talk.politics.*`, `alt.atheism`, `soc.religion.christian`
- **Other (1):** `misc.forsale`

Dataset – Splits and Statistics

Split	Size	Purpose
Train	11 314	Model training
Validation	3 766	HPO & model selection
Test	3 766	Final held-out evaluation

Metric	Characters	Words
Mean	∼1 800	∼300
Median	∼900	∼150
P95	∼6 500	∼1 100

Tokenisation coverage (`max_length`)

- 128 tokens → ∼50 % coverage
- **256 tokens → ∼75 % coverage** ← chosen
- 512 tokens → ∼90 % coverage

+ Manageable training time and computational cost
+ First 256 tokens are generally sufficient to predict the label
– Some information loss for very long documents

Model: ModernBERT

Recent **encoder-only** model by HuggingFace – a modernised version of BERT with multiple architectural improvements for robustness and efficiency.

BERT → RoBERTa

- Significantly more training data
- No Next Sentence Prediction loss
- Dynamic masking

RoBERTa → ModernBERT

- Even more training data
- **GeGLU** activation (more robust than GeLU)
- No bias terms except in last linear layer
- **Pre-normalisation** (LayerNorm at the beginning of sub-layers)
- Alternating attention

Training Strategy

Layer Freezing

Component	Status
Embedding layer	Frozen
Encoder layers 0–13	Frozen (bottom 50%)
Encoder layers 14–27	Trainable (top 50%)
Classification head	Trainable

Further unfreezing layers significantly increased compute requirements without meaningful accuracy gains.

Training Configuration

Parameter	Value
Optimiser	AdamW
Epochs	4
Batch size	16 per GPU
Max sequence len	256 tokens
Mixed precision	FP16 (CUDA)
Gradient clipping	max_norm = 1.0
LR scheduler	Linear warmup + decay
Hardware	NVIDIA Tesla T4
Multi-GPU	`nn.DataParallel`

Hyperparameter Optimisation

Method 1: Hyperparameters from the original ModernBERT paper, theory, and intuition.

Method 2: Quasi-Random Search (QRS) via Optuna

- Better space coverage than grid or pure random search
- Uses **Quasi-Monte Carlo (QMC)** sampling for low-discrepancy sequences
- More efficient exploration of the hyperparameter landscape

Search Space

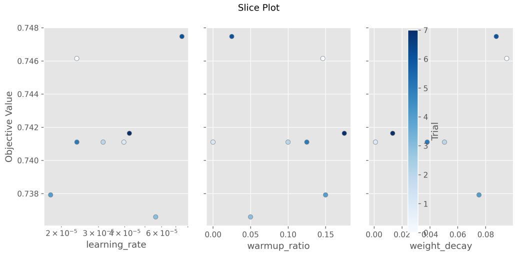
Hyperparameter	Range	Scale
Learning rate	$[10^{-5}, 10^{-4}]$	Log
Weight decay	[0.001, 0.1]	Linear
Warmup ratio	[0.0, 0.2]	Linear

Setup

- Objective: **maximise validation accuracy**
- Aggressive memory management (delete non-top-3 checkpoints)
- Optuna visualisations: slice plots & parameter importance

HPO – Visualisation and Evaluation

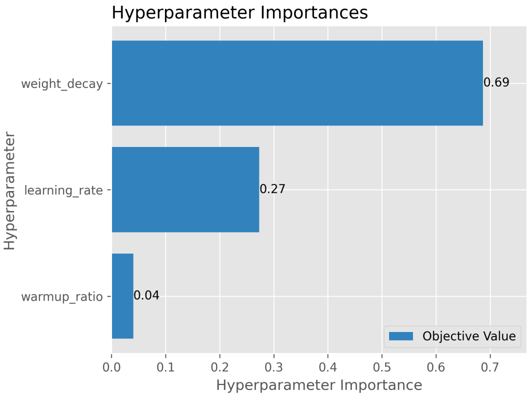
Hyperparameters vs. Loss



Test-Set Performance

Metric	Top-1	Top-2	Top-3
Accuracy	74.93%	74.35%	73.26%
Macro F1	0.7385	0.7353	0.7246
Weighted F1	0.7486	0.7453	0.7344
Test Loss	1.2554	1.4491	1.6971

Parameter Importances



Evaluation on Additional Modern Forum Data

- Collected a few samples for each class using web scrapers and a few samples manually from subReddits.
- Total of **2,000 samples** were collected (100 samples for each class) and stored in `collected_reddit_data.csv`.
- Evaluated the Top-2 model on `collected_reddit_data.csv`.

Evaluation Results:

- Test Loss: **0.9896**
- Accuracy: **0.7800**

Metric	Test Set	Reddit	Change
Accuracy	0.7435	0.7800	+0.0365
Macro F1	0.7353	0.7700	+0.0347
Weighted F1	0.7453	0.7700	+0.0248

Key Findings & Interpretation

Test Set vs. Collected Reddit Data

1. Categories with Large Improvements

- `rec.autos` (+0.293)
- `talk.politics.misc` (+0.315)
- `talk.politics.guns` (+0.244)
- `sci.electronics` (+0.219)
- `rec.motorcycles` (+0.148)
- `sci.space` (+0.152)
- `rec.sport.baseball` (+0.115)
- `sci.crypt` (+0.100)

2. Categories with Moderate Gains

- `alt.atheism` (+0.060)
- `comp.graphics` (+0.049)
- `comp.sys.mac.hardware` (+0.016)

3. Categories with Declines

- `comp.windows.x` (–0.328)
- `comp.sys.ibm.pc.hardware` (–0.259)
- `misc.forsale` (–0.230)
- `soc.religion.christian` (–0.163)
- `talk.religion.misc` (–0.131)
- `comp.os.ms-windows.misc` (–0.077)

4. Factors Influencing Performance

- Subreddit relevance
- Temporal language shift
- Community norms
- Class imbalance in original data

Confidence Score & Certainty of the Model

Maximum Softmax Probability (MSP)

Confidence computed using Softmax:

$$P(y = i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

Confidence score = Maximum Softmax Probability (MSP)

Evaluated on full test set.

Results:

- **Avg Confidence (Correct):** 0.9508
- **Avg Confidence (Incorrect):** 0.6879
- **Correlation (Confidence vs Correctness):** 0.5384

The model is significantly less confident when it makes a mistake.

Confidence Score & Certainty of the Model

Test-Time Augmentation (TTA) for Uncertainty Estimation

1. Apply 10 stochastic augmentations to each test sample
2. Run model inference on each augmented version
3. Average predicted probability distributions
4. Compute predictive entropy from the averaged probabilities
5. Compare entropy for correct vs incorrect predictions

Predictive Entropy Formula

$$H(p) = -\sum_{i=1}^{C} p_i \log p_i$$

Where p_i = averaged predicted probability for class i, and C = number of classes.
Higher $H(p) \rightarrow$ higher uncertainty.

Results:

- **Avg entropy (Correct):** 0.1424
- **Avg entropy (Incorrect):** 0.9232

Out-of-Distribution Detection

Strategy: Maximum Softmax Probability (MSP)

$$\text{score}(x) = \max_k \ \text{softmax}\left(\frac{\mathbf{z}(x)}{T}\right)_k$$

- **If** $\text{score}(x) \geq \tau \ \rightarrow$ classify as one of the 20 classes (In-Distribution)
- **If** $\text{score}(x) < \tau \ \rightarrow$ reject as **"null / other"** (Out-of-Distribution)

Parameter	**Effect**
Temperature T	$T > 1$: softens probabilities $\rightarrow$ better ID/OOD separation
Threshold τ	Higher τ: stricter $\rightarrow$ fewer false positives, more false negatives

OOD Detection Setup

In-Distribution (ID)

- **20 Newsgroups test set** (3 766 samples)
- Same split used for classification evaluation

Out-of-Distribution (OOD)

- **AG News** – 4-class news topic classification
- 2 000 randomly sampled test documents
- **Completely different domain** from 20 Newsgroups

Evaluation Protocol

1. Collect model logits for both ID and OOD data
2. Compute MSP scores at temperatures $T \in \{6, 7, 8, 9, 10\}$
3. For each T, report:
   - **AUROC** (Area Under ROC Curve)
   - **AP** (Average Precision)
   - **FPR@TPR70**
4. Per-threshold table: FPR, FNR, retained ID accuracy, % ID kept

OOD Detection – The FP / FN Trade-off

Understanding the Threshold τ

Direction	**Effect**	**Consequence**
$\uparrow \tau$	Stricter	$\downarrow$ FP / $\uparrow$ FN
$\downarrow \tau$	Looser	$\downarrow$ FN / $\uparrow$ FP

The optimal operating point depends on tolerance for false positives vs. false negatives.

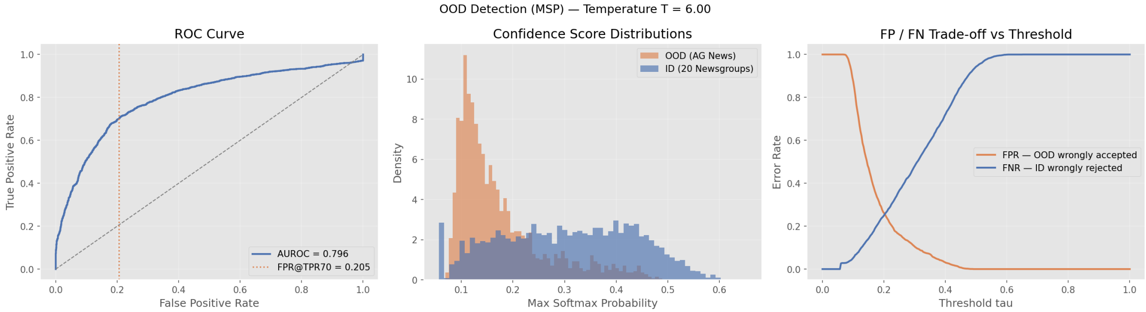
Threshold Table ($T = 6.00$)

AUROC = 0.7961 AP = 0.8841 FPR@TPR70 = 0.2050

τ	FPR	FNR	ID Acc	% ID
0.10	0.875	0.052	0.785	94.8%
0.20	0.263	0.249	0.874	75.1%
0.30	0.105	0.481	0.942	51.9%
0.40	0.031	0.720	0.974	28.0%
0.50	0.000	0.943	0.991	5.7%

OOD Detection – Diagnostic Visualisations

Three diagnostic plots are generated per temperature:



1. ROC Curve

- AUROC summarises discriminative quality
- Annotated with

2. Confidence Distributions

- Good detection $\rightarrow$ well-separated distributions
- OOD should cluster at lower

3. FP / FN Trade-off

- Crossing point $=$ balanced operating point
- Select τ per application

Results – OOD Temperature Comparison

Temperature T	**AUROC**	**AP**	**FPR@TPR70**
6.00	0.7961	0.8841	0.2050
7.00	0.7986	0.8860	0.2000
8.00	0.8003	0.8875	0.1970
9.00	0.8016	0.8885	0.1930
10.00	**0.8026**	**0.8893**	**0.1910**

Higher temperatures yield modest improvements across all metrics.

Best performance at $T = 10$: AUROC = 0.8026, AP = 0.8893, FPR@TPR70 = 0.1910

Key Design Decisions

1. **Used ModernBERT Variant of BERT**
   - Version of BERT that includes multiple small tweaks to make the transformer architecture more robust and efficient
2. **Layer Freezing (50%)**
   - Faster training, less memory → allows larger batch size
   - Minimal accuracy loss: lower layers learn general language features
3. **Initial Hyperparameters from ModernBERT Architecture**
   - Seeded tuning process with parameters proven effective in original paper
4. **Quasi-Random Search (QRS) over Grid/Random**
   - QMC sampling provides **better coverage** of the search space
5. **Validation Split from Test Set**
   - Split test 50/50 → separates validation for HPO and final test eval
   - Prevents data leakage: HPO decisions never touch the test set

Pipeline Summary

1. Data Loading (20 Newsgroups via HuggingFace) $\rightarrow$ Train / Validation / Test split

2. Tokenisation (ModernBERT tokenizer, max_len = 256)

3. Model Setup (ModernBERT-Large, 50 % layers frozen)

4. Hyperparameter Optimisation $\rightarrow$ Optuna QRS (LR, weight decay, warmup ratio)

5. Model Selection (Top 3 $\rightarrow$ evaluate on test set)

6. OOD Detection (MSP + Temperature Scaling) $\rightarrow$ ID:

Conclusion

Summary

- Successfully fine-tuned **ModernBERT-Large** on 20 Newsgroups (20-class classification)
- Systematic **hyperparameter optimisation** with Optuna Quasi-Random Search
- Extended to **OOD detection** using MSP with temperature scaling
- Comprehensive evaluation with AUROC, AP, FPR@TPR70, and threshold analysis

Future Work

- More trials of quasi-random search
- Bayesian HPO
- Experiment with other BERT variants

Lesson Learned

Errors in training pipeline can be very costly when running Optuna Quasi-Random HPO on large transformer models, as each iteration can take hours to run.

Thank You!

Team – M. Nasim Palakka Valappil & Ganesh Yadav Yatham

Questions?