Applied Deep Learning: Final Project Report
Media Bias Detection with a Multi Task Deep Learning Model

Hedda Fiedler (12402607)

January 14, 2025

1 Problem Statement and Motivation

In today's digital age, the challenge of media bias detection has become increasingly critical. With the overwhelming amount of information available and the prevalence of cognitive biases, people find it difficult to critically analyze the media they consume. Recent research by (Rodrigo-Ginés, de Albornoz and Plaza, 2024; Menzner and Leidner, 2024; Horych et al., 2024; Spinde et al., 2022, 2024), highlights the growing importance of automated bias detection tools in supporting critical media consumption.

Deep Learning, specifically the Multi Task Learning (MTL) approach with transformer models, is particularly well-suited for media bias detection for several reasons:

1. **Complex Pattern Recognition**: Media bias manifests in subtle linguistic patterns that require understanding context and nuance. Deep learning models, especially transformer architectures like DistilBERT, excel at capturing these complex language patterns through self-attention mechanisms.
2. **Transfer Learning Benefits**: By using a pre-trained language model (DistilBERT), the solution leverages existing knowledge about language structure, making it more efficient than training from scratch.
3. **Multi-Task Advantage**: The MTL approach allows the model to learn shared representations across related tasks (like bias detection, sentiment analysis, and hate speech detection). This is particularly valuable because media bias often intersects with these related phenomena.

The specific problem addressed in this project is the development of a computationally efficient deep learning model for automated media bias detection that maintains high accuracy while being more accessible to deployment in resource-constrained environments. This builds on recent advances in the field, particularly the work of Horych et al. (2024) with MAGPIE and earlier contributions by Spinde et al. (2022). The project code is available at this GitHub repository.

2 Solution Approach

This project adapts the MAGPIE architecture to create a more computationally efficient solution while maintaining high accuracy in bias detection. The approach focuses on three key aspects:

- Replaced RoBERTa with DistilBERT as the backbone encoder, reducing model size while maintaining performance
- Reduced the number of datasets from the original MAGPIE implementation for (pre-)fine-tuning
- Implemented comprehensive error handling and logging mechanisms

2.1 Dataset Selection and Processing

I carefully selected 9 key datasets from the original 59 datasets of the Large Bias Mixture database (LBM) (see (Horych et al., 2024)) representing all different bias aspects, chosen for their size and relevance. The selection is visible in Table 1.

Table 1: Selected Datasets and Their Characteristics

Task Family	Dataset	Size	Task Type
News bias	BABE	3,672	Binary Classification
Subjective bias	CW_HARD	6,843	Binary Classification
Hate speech	MeTooMA	7,388	Binary Classification
Gender bias	MDGender	2,332	Binary Classification
Sentiment analysis	MPQA	5,508	Token-Level Classification
Emotionally	GoodNewsEveryone	4,428	Token-Level Classification
Group bias	StereotypeDataset	2,208	Binary Classification
Stance detection	GWSD	2,010	Multi-Class Classification

2.2 Training Pipeline Implementation

The training process was executed in three sequential phases, each building upon the results of the previous phase to achieve optimal model performance.

2.2.1 Phase 1: Pre-finetuning

In the initial phase, the model was trained on all tasks except BABE with the following configuration:

- 500 training steps with 50 warmup steps
- Gradient accumulation for training stability
- Multi-task learning across 8 datasets (see Table 1)

As shown in Figure 1, both training and validation losses demonstrate steady convergence, decreasing from approximately 0.9 to 0.5 over the training period, even though the different datasets showed very different performance. The parallel trajectories of these curves and their close tracking indicate effective learning without significant overfitting.

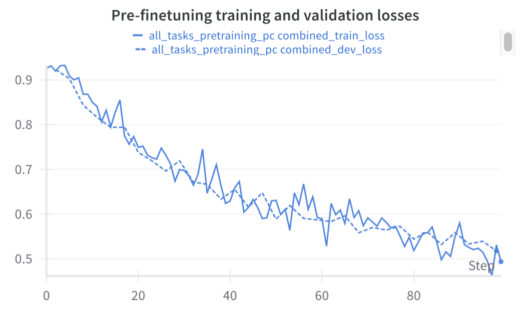

Figure 1: Pre-finetuning phase showing convergence of training and validation losses across all tasks except BABE.

2.2.2 Phase 2: Hyperparameter Optimization

Following pre-finetuning, I conducted a systematic grid search over three key hyperparameters:

- Dropout probability: range [0.1, 0.2, 0.3]
- Warmup steps: range [50, 100]
- Sub-batch size: range [16, 32, 64]

Figure 2 visualizes the relationships between these parameters and model performance. The optimal configuration achieved a maximum F1-score of 0.795 with:

- Dropout: 0.1 (indicating minimal regularization requirements)
- Warmup steps: 100 (enabling gradual learning rate adjustment)
- Batch size: 64 (providing stable gradient updates)

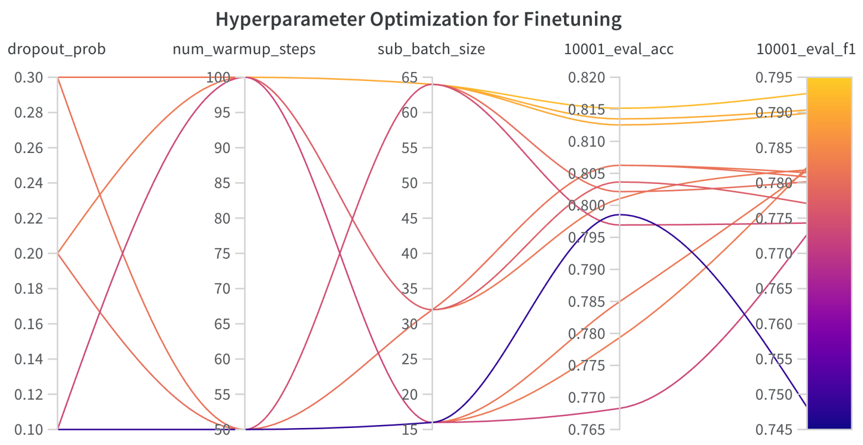

Figure 2: Parallel coordinates plot showing relationships between hyperparameters and model performance metrics.

2.2.3 Phase 3: Final Model Training

The final training phase involved finetuning on the BABE dataset using the optimal hyperparameters identified in Phase 2. To ensure robustness, training was repeated across 10 random seeds. Figure 3 reveals several key patterns:

- Training loss (dotted line) exhibits consistent convergence, stabilizing around 0.2
- Validation loss (solid line) shows higher values (0.4-0.6) with periodic fluctuations
- A persistent gap between training and validation losses indicates some degree of overfitting, though performance metrics remain strong
- Regular validation loss patterns suggest batch-specific sensitivity
- Consistent trends across random seeds demonstrate stable and reproducible training behavior

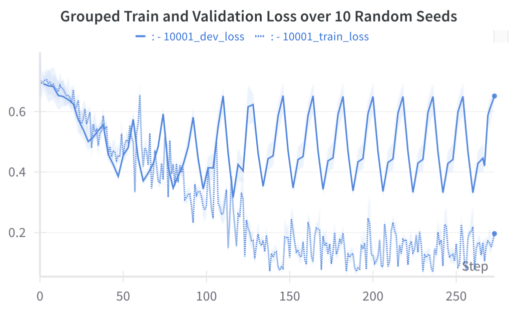

Figure 3: Final finetuning performance on BABE dataset averaged across 10 random seeds.

3 Results and Performance

This project targeted a Macro F1 Score of 0.78, based on the results achieved by Spinde et al. (2022) using a similar MTL approach. This target was chosen over MAGPIE's higher performance metrics due to the simplified architecture and reduced computational requirements.

Initial baseline testing achieved:

- Maximum F1 Score: 0.7143
- Significant variation across tasks
- Notable challenges with MeTooMA (F1 < 0.1) and MDGender (F1 < 0.35)

The final model achieved a maximum F1 Score of 0.7773, approaching my target of 0.78. This demonstrates that the simplified architecture successfully maintained comparable performance while reducing computational complexity.

Table 2: Final Model Performance Metrics

Metric	Mean	Std Dev	Min	Max
F1 Score	0.7587	0.0108	0.7415	0.7773
Accuracy	0.7772	0.0098	0.7604	0.7946
Loss	0.5120	0.0134	0.4920	0.5282

4 Model Deployment

The final model was uploaded to Huggingface and deployed as an interactive web application using Streamlit, making it accessible for practical use. The application[1] allows users to input text and receive bias detection results in real-time. The interface provides both overall bias scores and sentence-by-sentence analysis, making the model's predictions transparent and interpretable. This deployment phase demonstrates the practical applicability of the approach, as the model runs smoothly in a web-based environment.

5 Time Analysis

5.1 Planned vs. Actual Time Breakdown

Phase	**Planned (hours)**	**Actual (hours)**
Dataset Collection	2	6
Network Design & Building	15-20	43*
Training & Fine-tuning	20-25	21**
Application Development	15-20	8
Final Report	10	4
Presentation Preparation	5-10	2
Total	**67-87**	**84**

*Includes initial setup, first implementation, and reset

**Without Compute Time (about 52 h)

Table 3: Planned vs. Actual Time Comparison

The most significant time underestimation occurred in the network design and building phase, which took more than twice the planned time (43 hours vs. 15-20 hours planned). This underestimation stemmed primarily from three factors: limited software development experience, which necessitated multiple architecture revisions; the unexpected complexity of working with and adapting an existing codebase; and a steeper learning curve for proper code modularization than anticipated. To accommodate this additional time requirement, resources had to be reallocated from other project phases.

6 Main Take-Aways and Learnings

The following insights are drawn from the project:

Code Architecture Complexity

- Building modular, maintainable code requires significant upfront planning
- Understanding software engineering principles is crucial even for ML projects
- Working with existing codebases requires careful balance between simplification and maintaining functionality and efficiency

Technical Insights

- Hyperparameter tuning significantly improved model performance (F1 score from 0.71 to 0.78)

[1]Open the demo application here.

- Larger batch sizes (64) performed better than smaller ones for this task
- Some tasks (MeTooMA, MDGender) consistently underperformed in the pre finetuning step

Project Management Learnings

- Code architecture and setup took significantly more time than expected
- Limited time remained for detailed analysis of training dynamics

7 Future Improvements

If undertaking this project again, I would do the following aspects differently:

- Allocate more time for understanding the initial code architecture and setup and do this as the first step before starting with a generic project set up
- Invest in learning basic software engineering principles (e.g. modular code structure)
- Set up proper CI/CD pipeline from the start
- Allocate more time for analyzing task-specific performance variations and final training optimization

8 Conclusion

This project successfully demonstrated that a simplified multi-task learning approach can achieve comparable performance to more complex models in media bias detection. The final model reached the target metric while maintaining computational efficiency, providing a foundation for future development of accessible media bias detection tools.

References

Horych, Tomas, Martin Wessel, Jan Philip Wahle, Terry Ruas, Jerome Wassmuth, Andre Greiner-Petter, Akiko Aizawa, Bela Gipp and Timo Spinde. 2024. MAGPIE: Multi-Task Analysis of Media-Bias Generalization with Pre-Trained Identification of Expressions. In *"Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation"*.
URL: *https://media-bias-research.org/wp-content/uploads/2024/04/Horych2024a.pdf*

Menzner, Tim and Jochen L. Leidner. 2024. “BiasScanner: Automatic Detection and Classification of News Bias to Strengthen Democracy.”.
URL: *https://arxiv.org/abs/2407.10829*

Rodrigo-Ginés, Francisco-Javier, Jorge Carrillo de Albornoz and Laura Plaza. 2024. “A systematic review on media bias detection: What is media bias, how it is expressed, and how to detect it.” *Expert Systems with Applications* 237:121641.
URL: *https://www.sciencedirect.com/science/article/pii/S0957417423021437*

Spinde, Timo, Jan-David Krieger, Terry Ruas, Jelena Mitrović, Franz Götz-Hahn, Akiko Aizawa and Bela Gipp. 2022. *Exploiting Transformer-Based Multitask Learning for the Detection of Media Bias in News Articles*. Springer International Publishing p. 225–235.
URL: *http://dx.doi.org/10.1007/978-3-030-96957-8*$_2$0

Spinde, Timo, Smi Hinterreiter, Fabian Haak, Terry Ruas, Helge Giese, Norman Meuschke and Bela Gipp. 2024. “The Media Bias Taxonomy: A Systematic Literature Review on the Forms and Automated Detection of Media Bias.”.
URL: *https://arxiv.org/abs/2312.16148*