Figurative Language Understanding

Sentiment and Sarcasm Classification: single and multitask learning, domain shift and BERT optimization

Elena Nespolo
Politecnico di Torino
Torino, Italy
s345176@studenti.polito.it

Romeo Vercellone
Politecnico di Torino
Torino, Italy
s341967@studenti.polito.it

Giuseppe Mallo
Politecnico di Torino
Torino, Italy
s346884@studenti.polito.it

Carla Finocchiaro
Politecnico di Torino
Torino,Italy
s337024@studenti.polito.it

Alessandra Marchese
Politecnico di Torino
Torino, Italy
s349536@studenti.polito.it

***Abstract*—This report investigates sentiment analysis and sarcasm detection across English varieties using the BESSTIE benchmark, focusing on cross-variety generalization, domain shift, and computational efficiency. Both tasks are modeled as supervised binary classification problems, and we evaluate performance degradation when training and testing on different regional varieties.**

Starting from a BERT-based baseline with a linear classification head, we introduce three main extensions. First, we explore architectural variants at the classification head level, replacing the linear layer with convolutional and BiLSTM heads to better capture local and sequential patterns in contextual embeddings. We also propose a multitask CrossTalk configuration that jointly models sentiment and sarcasm through controlled representation sharing. To stabilize multitask optimization, we integrate PCGrad, which mitigates gradient conflicts between tasks.

Second, we analyze domain shift by training models on external datasets (Twitter Sentiment and Bicodemix) and evaluating zero-adaptation transfer to BESSTIE. This setting allows us to assess robustness under distributional mismatch and quantify the impact of linguistic variety and platform-specific differences on generalization.

Third, we incorporate a Sparse Attention mechanism based on cascading token and head pruning to study the trade-off between computational efficiency and predictive robustness.

Results show that increasing classification head complexity provides limited gains, while domain and variety shift remain the primary challenges, especially for sarcasm detection. Multitask learning improves joint modeling in specific settings but does not fully resolve cross-variety degradation. Finally, Sparse Attention highlights a delicate balance between efficiency and robustness, as aggressive pruning may remove subtle contextual cues essential for figurative language understanding. Code and experimental details are available at: https://github.com/elenanespolo/Sentiment_Sarcasm_Analysis.git.

I. Problem Statement

The analysis of sentiment and sarcasm on online platforms has become increasingly important for understanding public opinion, social dynamics, and user behaviour in real-world applications. Modelling these phenomena therefore represents a key challenge in the NLP field.

In this work, we focus on BESSTIE [1], a benchmark for sentiment and sarcasm classification with three varieties of English: Australian (en-AU), Indian (en-IN), and British (en-UK). Starting from strong pretrained representations based on BERT [2] and inspired by prior work showing the benefits of more expressive classification heads [3], we investigate whether architectural choices at the classification head level, enabling multitask settings as well, can improve performance in figurative language understanding. To further stabilize multitask optimization, we adopt gradient-based conflict mitigation techniques such as PCGrad [4].

Real-world scenarios often require models to operate on data distributions that differ from those observed during training. For this reason, we explicitly study domain shift by evaluating models trained on external datasets against BESSTIE. This analysis allows us to assess the robustness and generalization capabilities of both single-task and multitask architectures.

Finally, practical applications impose strong constraints on computational efficiency, especially when models are deployed in real-time settings. To address this aspect, we complement our performance analysis with an efficiency-oriented technique proposed by [5] based on token and attention head pruning, aiming to reduce model complexity while preserving predictive performance.

II. Methodology

A. Baseline Model: Linear Classification Head

As a baseline, we fine-tune a pre-trained BERT encoder coupled with a linear classification head, a widely adopted architecture for sequence-level classification tasks. Given an input sequence of tokens, BERT produces a sequence of contextualized token embeddings, and we denote H as the hidden dimension of the model. Following standard practice, we use the embedding corresponding to the special `[CLS]` token, $\mathbf{h}_{\text{CLS}}$, as a fixed-length representation of the entire sequence. This representation is passed through a dropout layer and then projected into the label space via a linear transformation:

$$\mathbf{y} = \mathbf{W}\mathbf{h}_{\text{CLS}} + \mathbf{b},$$

where $\mathbf{W} \in \mathbb{R}^{C \times H}$, $\mathbf{b} \in \mathbb{R}^{C}$, and C denotes the number of sentiment classes. The resulting logits $\mathbf{y}$ are used to compute the cross-entropy loss during training.

B. Extended Models

To better leverage the token-level representations produced by BERT, we explore alternative classification heads that explicitly model sequential and local compositional patterns in the embedding space.

1) BiLSTM-based Classification Head: We replace the linear classification head with a single-layer bidirectional Long Short-Term Memory (BiLSTM) network. The full sequence of BERT embeddings $\mathbf{H}$ is fed into the BiLSTM, which processes the sequence in both forward and backward directions. The hidden states from both directions are concatenated to form contextual representations $\mathbf{z}_t = [\overrightarrow{\mathbf{h}}_t; \overleftarrow{\mathbf{h}}_t]$.

For sequence-level classification, we use the representation corresponding to the first token position, $\mathbf{z}_1$, which captures information from the entire sequence. This vector is then passed through a linear layer to produce the final logits:

$$\mathbf{y} = \mathbf{W}\mathbf{z}_1 + \mathbf{b}.$$

This architecture allows the classifier to better model sequential structure and long-range dependencies compared to relying solely on the `[CLS]` embedding.

2) Convolutional Classification Head: We adopt a convolutional classification head that focuses on capturing local patterns in the sequence of BERT embeddings. The token-level representations are treated as a sequence of feature vectors, where the hidden dimension acts as the channel dimension. One-dimensional convolutional filters are applied over the token axis:

$$\mathbf{c}_t = f(\mathbf{W}_k * \mathbf{H}_{t:t+k-1} + \mathbf{b}_k),$$

where k is the kernel size, $*$ denotes the convolution operation, and $f(\cdot)$ is a non-linear activation function.

The feature maps are aggregated via global average pooling to obtain a fixed-size representation, which is then fed to a linear classifier to produce the final logits. This convolutional head captures local n-gram-like patterns in the contextual embedding space, complementing BERT's global representations.

3) CrossTalk Classification Head: To enable multitask learning, we introduce a classification head which jointly models sentiment and sarcasm prediction, motivated by the fact that each task can provide useful information to support the other.

A shared convolutional encoder extracts a common representation from BERT outputs, which is projected into task-specific spaces and fused through cross-talk layers before being fed to separate output classifiers.

To further stabilize multitask optimization, we adopt PCGrad from [4], a gradient surgery technique that mitigates conflicts between task gradients by projecting them onto the normal plane of each other. This reduces gradient interference and improves learning stability and overall performance.

C. Domain shift

Natural language is heterogeneous and strongly dependent on the context in which it is produced. Differences in communication settings, user populations, platforms, and stylistic conventions can lead to substantial variations in vocabulary, syntax, and sentence structure. As a consequence, models trained on data drawn from a specific domain often struggle to generalize when applied to texts originating from different sources.

Building upon the upper bounds established in the previous sections, we therefore address the challenge of domain shift and evaluate the robustness of our models in these conditions. Specifically, we assess the performance of our BERT-based models trained on external benchmarks and tested directly on BESSTIE, without any additional domain adaptation.

The external benchmarks considered in our experiments are the following:

- **X (Twitter) dataset:** a benchmark consisting of short, informal, and highly heterogeneous texts, which represents a challenging source domain due to its distinctive linguistic characteristics.
- **Bicodemix dataset:** a multilingual benchmark providing annotations for both sentiment and sarcasm. This property enables the evaluation of multitask learning scenarios while simultaneously studying the effects of domain shift.

Figure 1 reports the percentage of coverage of the top 1000 most common words per dataset of interest. en-UK and en-AU (inner-circle) are more similar to each other compared to en-IN (outer-circle), that enables easier knowledge-transfer. The other two datasets may pose an additional challenge due to the even lower coverage of common words.

D. Sparse Attention for efficient inference

The attention mechanism has become the computational bottleneck because of its quadratic complexity and low arithmetic intensity. BERT was originally designed to introduce architectural optimizations rather than relying solely on scaling model size to improve performance. With this in mind, we implement a way to reduce computational requirements by introducing cascading masking at the token and attention head level.

a) Cascade token pruning: The first algorithm aims to reduce the amount of tokens a certain layer of the model will use, effectively removing from the input certain tokens. The mask is obtained after each multi-head attention layer by computing how much each token (t_i) contributes in the self-attention scores ($a_s \in R^{H/64 \times L_0 \times L_0}$) and accumulate each of these contributions across heads (h') and output tokens (t_o):

$$s_{t_i} = \sum_{h'=0}^{n_h} \sum_{t_o=0}^{L_0} (a_s)_{h',t_o,t_i}$$

Defining as $n_h = H/D$ the number of attention heads used by the model and $D = 64$ the dimension of the QKV representations. Tokens with small cumulative importance scores s_{t_i} are masked away and will not be used in the next layers any more (the cascading part of the masking). We we decided to remove the tokens associated to the bottom $p_t\%$ of the cumulative important scores (alternately to keep the top $1 - p_t\%$).

b) Cascade head pruning: It has been observed that some attention heads may be redundant or have little influence on outputs. Instead of reducing the sequence length, head pruning reduces the feature length, removing these redundancies. The importance of each head is calculated on the output of the self-attention layer ($a_{out} \in R^{L_0 \times H}$). This masking is based

on the fact that heads that outputs small magnitude signals will have small impact on the output of the Dense layer that operates on the concatenation of each attention head result.
The a_{out} signal is reshaped to separate each chunk, the output of a single head ($E \in R^{n_h \times L_0 \times D}$), and accumulating all the magnitudes associated to that head (h_i):

$$s_{h_i} = \sum_{t_o=0}^{L_0} \sum_{d=0}^{D} |(E)_{h_i,t_o,d}|$$

We removed the heads associated to the bottom $p_h\%$ of the cumulative important scores.

III. Experiments and Results

In this section, we present the experimental evaluation of the proposed extensions on the BESSTIE benchmark. We describe the dataset, the experimental setup and a detailed analysis of the results for each extension.

A. Data description

BESSTIE is a benchmark combining data from two different domains: Google Places reviews and Reddit comments, and covering three varieties of English: Australian (en-AU), Indian (en-IN), and British (en-UK). The Google Places data is used exclusively for sentiment classification, whereas the Reddit data supports both sentiment and sarcasm detection.

X (Twitter) is a goldmine for understanding public opinion. Therefore, the **Twitter Sentiment Analysis Dataset** [6] allows researchers to analyse the sentiment behind tweets, whether they're expressing joy about a recent event, frustration over a political decision, or anything in between. This dataset helps to gauge the overall sentiment of Twitter users on various topics, such as video games, companies and many others.
Differently from BESSTIE, this dataset present a much higher amount of data but due to the nature of the informations, the samples are much shorter and only a fraction of the data contained is informative enough. In fact the mean textual length is less than 25 words, so we applied a filtering that excludes textual data if it's shorter than 40 words.
This new dataset is more aligned with BESSTIE in term of mean textual length with a new mean of about 50 words per sample. But it still present a significant shift in writing style enabling the observability of a domain shift.

[7] provides the **Bicodemix dataset**: a multilingual benchmark from which we focus exclusively on the English comments for our experiments. Data were collected from social media platforms in Southeast Asia, then refined and labelled with both sentiment and sarcasm annotations.
The comments in this dataset are generally short, with a mean comment length of 17, similar to the Twitter dataset. This brevity reflects typical user-generated content on social platforms, posing challenges for text understanding and classification models.
Despite its relatively small size, we include the Bicodemix dataset because it supports multitask learning by providing both sentiment and sarcasm labels.

B. Experimental Design

All experiments are conducted using the Google Colab platform, utilizing an NVIDIA T4 GPU to accelerate the training process. The implementation relies on Python 3, using PyTorch as the deep learning framework and the Hugging Face Transformers library for loading the pre-trained BERT models.

All models are fine-tuned end-to-end, updating both the BERT encoder and the classification head parameters. Training is performed using the Adam optimizer with a learning rate of 2×10^{-5} for 30 epochs and a batch size of 8. To address class imbalance, a weighted cross-entropy loss is employed, with class weights computed from the empirical label distribution of the training data. Model performance is evaluated using the macro-averaged F1-score, which ensures balanced evaluation across classes.

To assess cross-variety generalization, models are trained on data from a single regional variety of English and evaluated both on the same variety (in-variety) and on the remaining varieties (cross-variety). On the other hand, when performing domain shift, the models are trained on the external datasets and evaluated on each variety within the BESSTIE benchmark.

C. Execution Times

Table I and Table II report the average seconds per batch for each task and model variant, with all experiments performed using a fixed batch size of 8. For the trained-on-BESSTIE models, to ensure a robust estimate, the reported values were obtained via a two-step aggregation process: first, for each model configuration, we computed the average seconds per batch for each regional variety run that we obtain in a single epoch; subsequently, these per-variety metrics were averaged across all considered varieties to provide a single, representative timing estimate that enables a fair comparison of the computational overhead introduced by the different classification heads.

In terms of performance, the Linear Head proves to be the most efficient architecture. The introduction of the Convolutional Head results in a moderate increase in latency, while the LSTM incurs the highest computational cost, attributable to its sequential processing mechanism. A similar trend is observed in the multitask setting, where the addition of the *Cross Talk* head leads to a slight further increase in processing time compared to the single-task one.

Although Sparse Attention is designed to reduce the quadratic complexity of self-attention, we observe a slight increase in the average seconds per batch, shown in Table I. This overhead arises because our implementation operates only at the software level. Instead of true pruning, which avoids loading irrelevant values into memory, we apply masking, so attention scores are still fully computed and only zeroed afterward. In contrast, [5] introduces dedicated hardware support for activation-aware pruning, achieving measurable speed-ups.

D. Analysis of the results

Extension 1: Architectural Exploration.

The results for the three tasks highlight a consistent pattern

across domains and architectures.. For the Google sentiment results shown in Figure 2, the Linear Head achieves the best performance on en-AU, en-UK varieties, while more complex heads (CNN, LSTM) provide no clear advantage and may even slightly degrade performance. This suggests that, for standard English, BERT representations are already sufficiently expressive. However, for the Indian variety (en-IN), the CNN head improves cross-variety performance, indicating the presence of localized linguistic patterns that benefit from convolutional feature extraction.

For Reddit sentiment, whose results are shown in Figure 3, performance remains strongly variety-dependent, with the highest scores observed in in-variety settings. Unlike the Google domain, architectural extensions do not consistently improve cross-variety generalization, suggesting that the performance gap is driven by deeper linguistic and cultural differences rather than structural features that CNN or LSTM layers can capture.

A similar behavior emerges in Reddit sarcasm detection in Figure 4. While slightly higher in-variety scores are sometimes obtained with more complex heads, cross-variety performance remains low and comparable across architectures. Overall, these results further confirm that the primary bottleneck lies in the variety shift between linguistic varieties rather than in the expressiveness of the classification head. Consequently, most of the discriminative information appears to be already captured by BERT, with the classification head playing a secondary role in overall model performance.

Finally, the *CrossTalk* head achieves the strongest performance, although its evaluation is limited to Reddit data, the only domain annotated for both sentiment and sarcasm. As shown in Figure 5, F1-scores significantly improve compared to single-task settings, particularly for the en-UK variety, while en-AU and en-IN show more moderate gains. These results indicate that controlled information sharing between related tasks enhances performance, although it does not fully resolve the challenges introduced by linguistic variety shifts.

Extension 2: Domain Shift.

The Twitter Sentiment dataset enables inference only on sentiment, therefore only two thirds of BESSTIE can be used for performance evaluation. As shown in Figure 6, overall performance are impacted, while the LSTM head proves to be relatively robust to domain shift when trained on Twitter data, consistently achieving stronger results.

In contrast, the Bicodemix dataset allows training in both single-task and multi-task settings. For both configurations, all the proposed classification heads are evaluated. Sentiment classification performance on Bicodemix, as reported in Figure 7, is generally lower than that observed on the Twitter dataset. However, sarcasm detection results show a noticeable improvement. This suggests that the higher quality and more controlled annotation process of Bicodemix partially compensates for domain-related challenges.

Overall, performance on the Google review domain remains relatively stable across settings, whereas results on Reddit data are more variable and strongly dependent on the linguistic context on which the model was trained.

Extension 3: Sparse Attention for efficient inference.

Figure 8 reports both in-variety and cross-variety performance, allowing a direct comparison with the standard BERT + linear classifier baseline for the benchmark BESSTIE. The results obtained with BERT enhanced by Sparse Attention show task-dependent behaviour across domains. For the Google sentiment task, performance remains relatively stable in in-variety settings and only moderately degrades under cross-variety evaluation. This suggests that, in a more structured and lexically consistent dataset, pruning tokens and attention heads does not significantly compromise the distributed sentiment signal.

In contrast, Reddit shows a stronger performance degradation, especially under cross-variety evaluation. Reddit data is short, informal, and highly contextual, where subtle lexical and pragmatic cues are often crucial. The cascading token pruning mechanism may remove low-frequency but informative elements, reducing robustness which relies on sparse and context-dependent signals.

Training on Twitter sentiment instead causes clear degradation on Google reviews but not on Reddit sentiment, as presented in Figure 9. This may be due to structural similarities between Twitter and Reddit posts, despite differences in topics.

On the other hand, as shown in Figure 10, when applying spAtten to models trained on the Bicodemix dataset, a clear degradation is observed for the sentiment classification. In contrast, sarcasm detection remains almost unaffected, both in the single-task and multitask settings, achieving results comparable to those obtained without sparse attention. This suggests that in Bicodemix sarcasm-related cues are more robust to token and head pruning, while sentiment classification appears to depend more on global contextual information that is partially lost through sparsification.

IV. Conclusions

Results show that increasing classification head complexity provides limited gains, confirming that BERT representations already encode most discriminative features. Multitask learning with CrossTalk and PCGrad improves joint modeling, particularly in the Reddit domain, but does not fully mitigate cross-variety degradation. Domain shift emerges as the main challenge, especially for sarcasm detection, where linguistic and pragmatic differences significantly impact performance. Finally, Sparse Attention highlights the delicate balance between computational efficiency and robustness, as pruning may remove subtle but informative contextual cues. Overall, the study suggests that future improvements should prioritize domain adaptation and representation alignment over architectural expansion.

References

[1] D. Srirag, A. Joshi, J. Painter, and D. Kanojia, "BESSTIE: A Benchmark for Sentiment and Sarcasm Classification for Varieties of English."

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *North American Chapter of the Association for Computational Linguistics*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:52967399

[3] N. J. Prottasha, A. A. Sami, M. Kowsher, S. A. Murad, A. K. Bairagi, M. Masud, and M. Baz, "Transfer Learning for Sentiment Analysis Using BERT Based Supervised Fine-Tuning," *Sensors*, vol. 22, no. 11, p. 4157, May 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/11/4157

[4] T. Yu, S. Kumar, A. Gupta, S. Levine, K. Hausman, and C. Finn, "Gradient Surgery for Multi-Task Learning," Dec. 2020, arXiv:2001.06782 [cs]. [Online]. Available: http://arxiv.org/abs/2001.06782

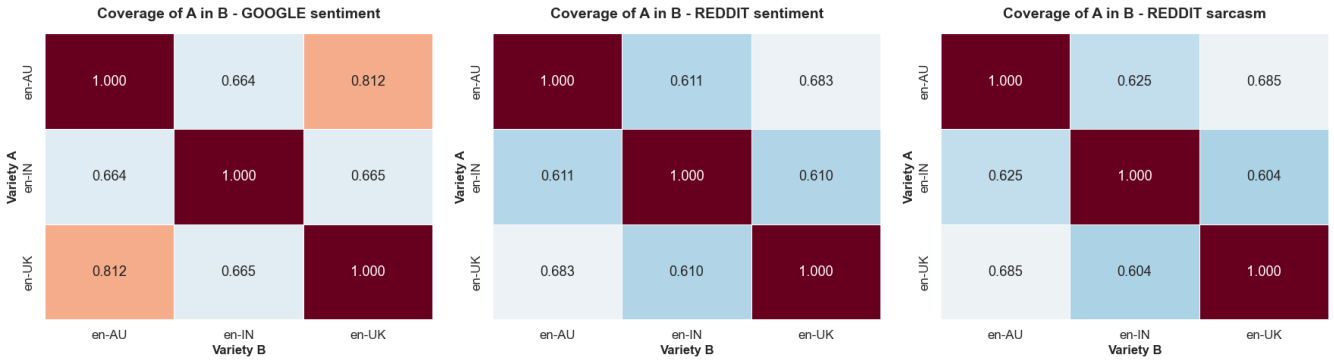
[5] H. Wang, Z. Zhang, and S. Han, "SpAtten: Efficient Sparse Attention Architecture with Cascade Token and Head Pruning," in *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, Feb. 2021, pp. 97–110, arXiv:2012.09852 [cs]. [Online]. Available: http://arxiv.org/abs/2012.09852

[6] GeeksforGeeks, "Dataset for sentiment analysis," https://www.geeksforgeeks.org/nlp/dataset-for-sentiment-analysis/#2-twitter-sentiment-analysis-dataset, 2024.

[7] M. S. Md Suhaimin, M. H. Ahmad Hijazi, and E. G. Moung, "Annotated dataset for sentiment analysis and sarcasm detection: Bilingual code-mixed English-Malay social media data in the public security domain," *Data in Brief*, vol. 55, p. 110663, Aug. 2024. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2352340924006309

APPENDIX

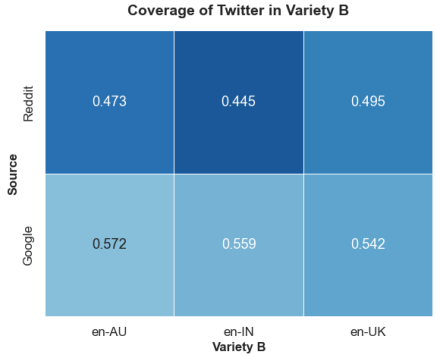
A. Benchmark Evaluation



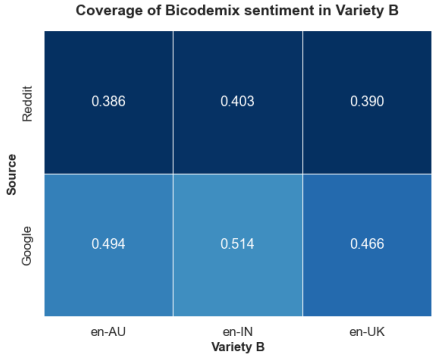
(a) BESSTIE Google sentiment

(b) BESSTIE Reddit sentiment

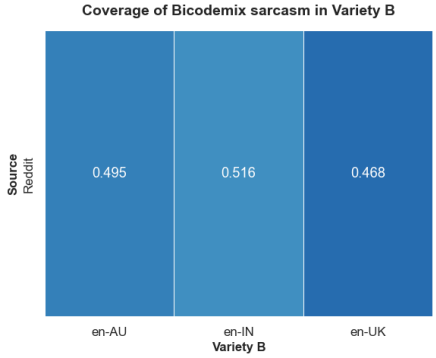
(c) BESSTIE Reddit sarcasm



(d) Twitter sentiment with BESSTIE



(e) Bicodemix sentiment with BESSTIE



(f) Bicodemix sarcasm with BESSTIE

Fig. 1: Percentage of top 1000 most frequent words shared across datasets. 1a to 1c uses as source the BESSTIE dataset, 1d uses X as source dataset, 1e and 1f uses Bicodemix as source dataset.

B. Execution Time Analysis

This section reports the average training time per batch for the proposed models, measured in seconds, in order to assess the computational overhead introduced by different classification heads, multitask learning, and sparse attention mechanisms.

TABLE I: Average training time per batch (in seconds) for single-task models.

Model	Google Sent.	Reddit Sent.	Reddit Sarc.	Twitter Sent.	Bic. Sent.	Bic. Sarc.
BERT + Linear Head	0.2799	0.2907	0.2837	0.2889	0.5054	0.5054
BERT + Convolutional Head	0.3116	0.3157	0.3037	0.2970	0.5269	0.5322
BERT + LSTM	0.3830	0.3885	0.3770	0.3541	0.6882	0.7043
BERT + spAtten + Linear Head	0.2809	0.3111	0.2902	0.3371	0.5779	0.6261

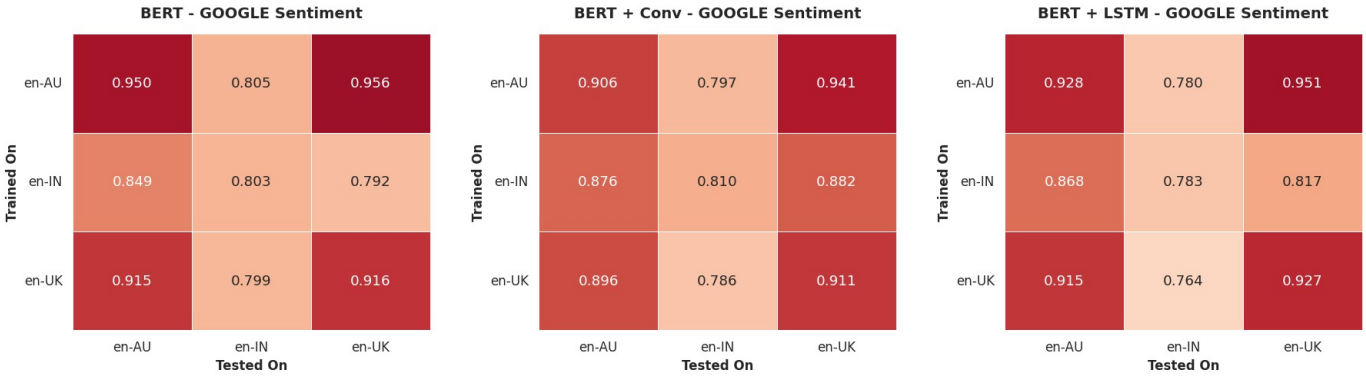
TABLE II: Average training time per batch (in seconds) for multitask models.

Model	BESSTIE Sent./Sarc.	Bicodemix Sent./Sarc.
BERT + Linear Head	0.3163	0.3441
BERT + spAtten + Linear Head	-	0.3161

C. Extended Experimental Analysis

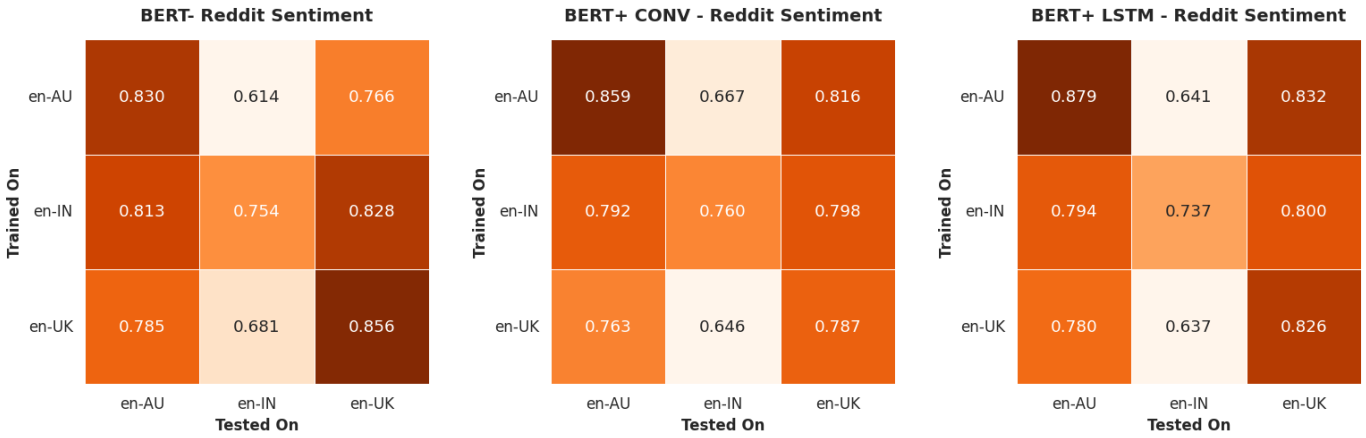
Here below are reported the complete set of F1-score heatmaps for all the models and experimental settings considered in this work and discussed in the main sections.

- Figure 2, 3, 4, and 5 correspond to **Extension 1**, which investigates the impact of different classification heads.
- Figure 6 and 7 report the results for **Extension 2**, focusing on cross-domain generalization under domain shift.
- Figure 8, 9, and 10 present the outcomes of **Extension 3**, which explores model efficiency through Sparse Attention mechanisms.



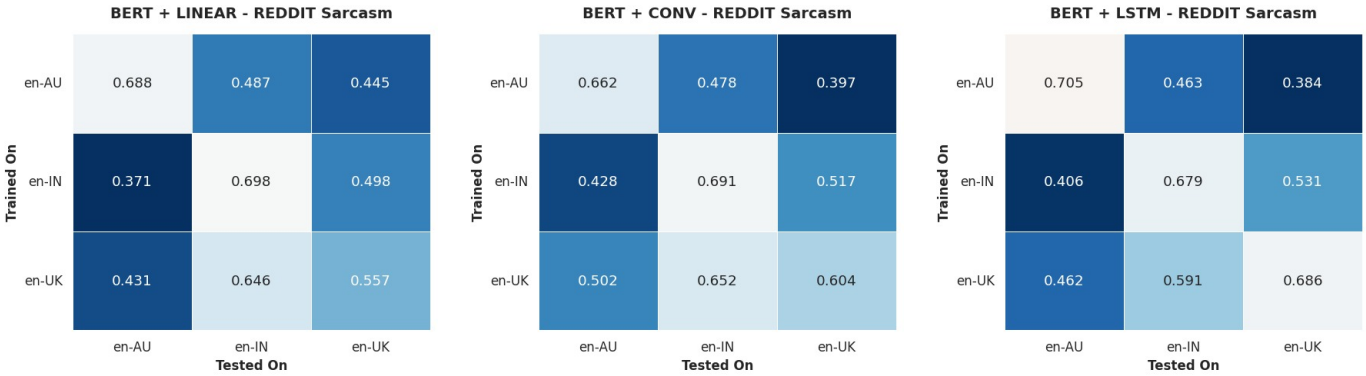
(a) BERT + Linear Head (b) BERT + Convolutional Head (c) BERT + LSTM Head

Fig. 2: Google Sentiment: Cross-variety performance heatmaps with F1-score.



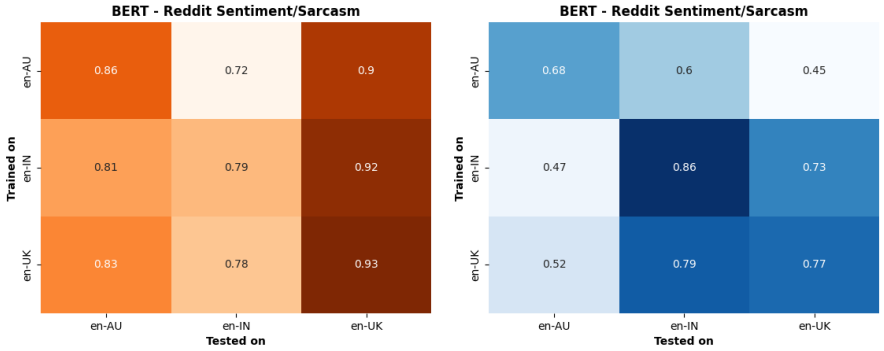
(a) BERT + Linear Head (b) BERT + Convolutional Head (c) BERT + LSTM Head

Fig. 3: Reddit Sentiment: Cross-variety performance heatmaps with F1-score.



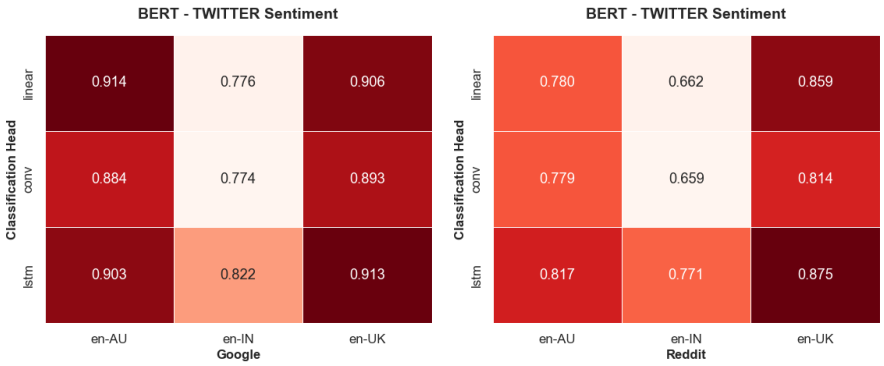
(a) BERT + Linear Head (b) BERT + Convolutional Head (c) BERT + LSTM Head

Fig. 4: Reddit Sarcasm: Cross-variety performance heatmaps with F1-score.



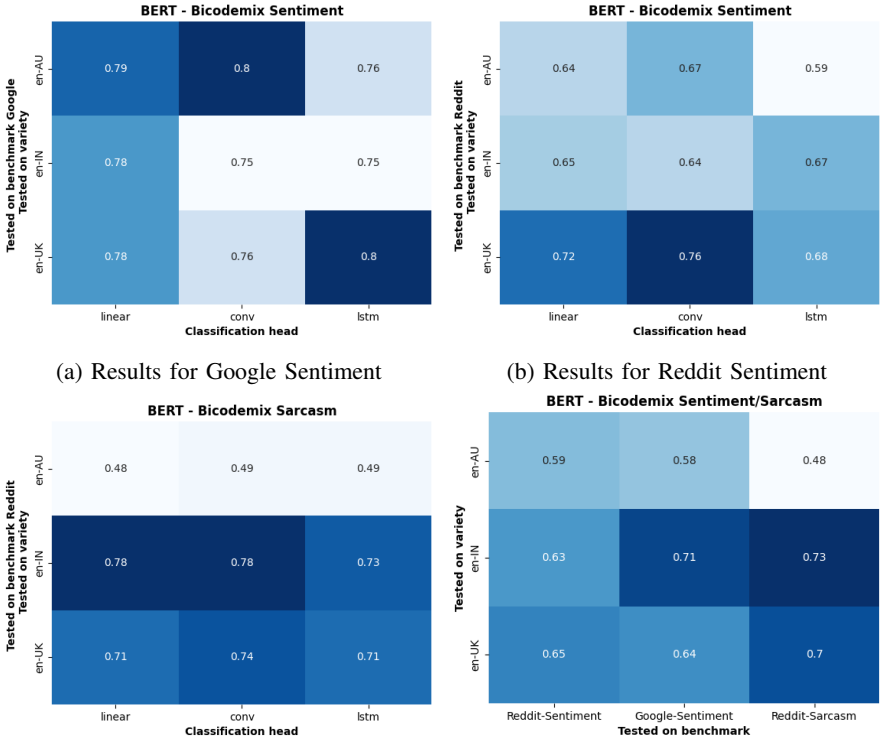
(a) Result for Reddit Sentiment (b) Result for Reddit Sarcasm

Fig. 5: Reddit multitask setting: Cross-variety performance heatmaps with F1-score of the BERT model with *Cross Talk* classification head trained on Reddit Sentiment/Sarcasm and tested separately for the two tasks.



(a) Results for Google Sentiment (b) Results for Reddit Sentiment

Fig. 6: Twitter Sentiment: Cross-domain performance heatmaps with F1-score of the BERT models with Linear, Convolutional, and LSTM classification heads trained on Twitter Sentiment and tested on Google Sentiment and Reddit Sentiment.



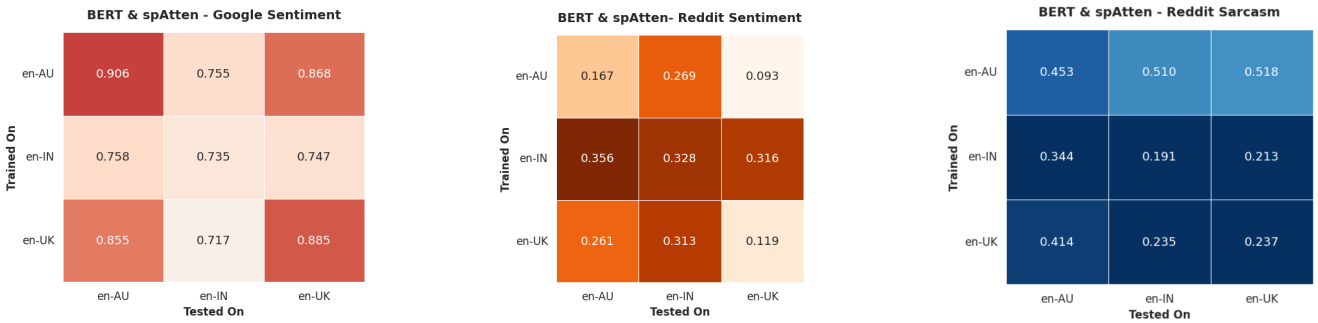
(a) Results for Google Sentiment

(b) Results for Reddit Sentiment

(c) Results for Reddit Sarcasm

(d) Results for both Google and Reddit

Fig. 7: Bicodemix single and multitask setting: Cross-domain performance heatmaps with F1-score of the BERT models: (a) and (b) with Linear, Convolutional, and LSTM classification heads trained on Bicodemix Sentiment and tested on Google Sentiment and Reddit Sentiment; (c) with Linear, Convolutional, and LSTM classification heads trained on Bicodemix Sarcasm and tested on Reddit Sarcasm; (d) with *Cross Talk* classification head trained on Bicodemix Sentiment/Sarcasm and tested on Google Sentiment, Reddit Sentiment and Reddit Sarcasm.



(a) Results for Google Sentiment

(b) Results for Reddit Sentiment

(c) Results for Reddit Sarcasm

Fig. 8: Cross-variety performance heatmaps with F1-score of the BERT models with Sparse Attention and Linear classification head for Google Sentiment, Reddit Sentiment and Reddit Sarcasm.

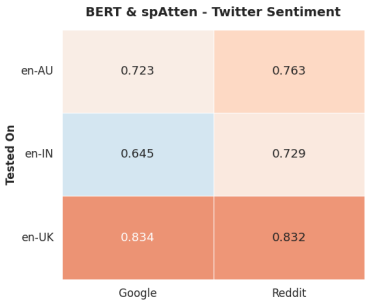
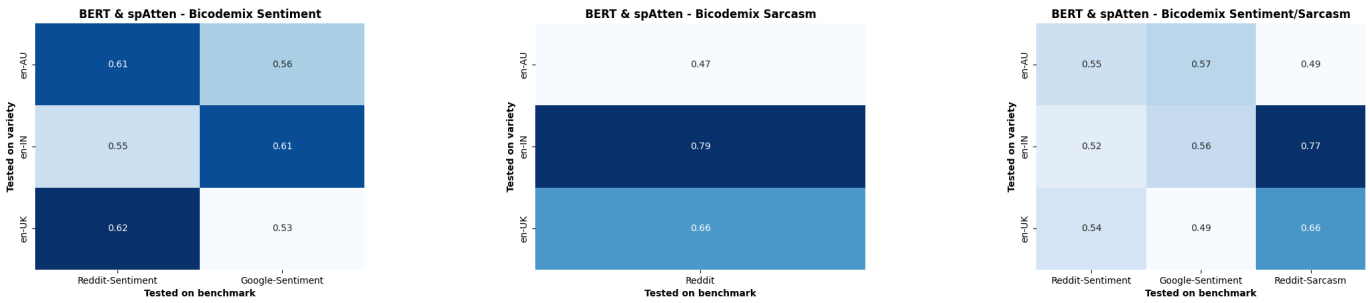


Fig. 9: Cross-domain performance heatmaps with F1-score of the BERT model with Sparse Attention and Linear classification head trained on Twitter Sentiment and tested on Google Sentiment and Reddit Sentiment.



(a) Bicodemix Sentiment (b) Bicodemix Sarcasm (c) Bicodemix Sentiment/Sarcasm

Fig. 10: Cross-domain performance heatmaps with F1-score of the BERT models with Sparse Attention and Linear classification head: (a) trained on Bicodemix Sentiment and tested on Google Sentiment and Reddit Sentiment; (b) trained on Bicodemix Sarcasm and tested Reddit Sarcasm; (c) trained on Bicodemix Sentiment/Sarcasm and tested on Google Sentiment, Reddit Sentiment and Reddit Sarcasm.

D. *Sparse Attention Example*

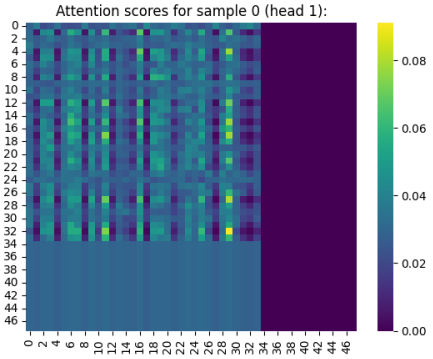
Example of an application of sparse attention on the input: ”The attention mechanism is becoming increasingly popular in Natural Language Processing (NLP) applications, showing superior performance than convolutional and recurrent architectures.”

In the first layer the tokenization used is: ’[CLS]’, ’the’, ’attention’, ’mechanism’, ’is’, ’becoming’, ’increasingly’, ’popular’, ’in’, ’natural’, ’language’, ’processing’, ’(’, ’nl’, ’##p’, ’)’, ’applications’, ’,’, ’showing’, ’superior’, ’performance’, ’than’, ’con’, ’##vo’, ’##lu’, ’##tion’, ’##al’, ’and’, ’rec’, ’##urrent’, ’architecture’, ’##s’, ’.’, ’[SEP]’.

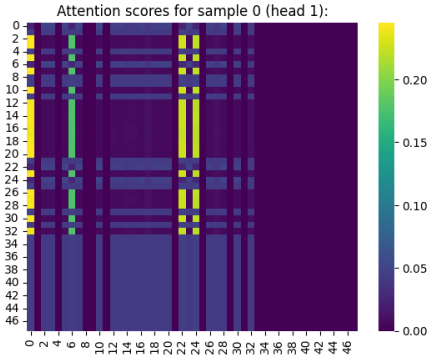
At the 6^{th} layer: ’[CLS]’, ’attention’, ’mechanism’, ’becoming’, ’increasingly’, ’popular’, ’language’, ’(’, ’nl’, ’##p’, ’)’, ’applications’, ’,’, ’showing’, ’superior’, ’performance’, ’con’, ’##vo’, ’##lu’, ’##al’, ’and’, ’rec’, ’architecture’, ’.’.

At the last layer: ’[CLS]’, ’attention’, ’mechanism’, ’increasingly’, ’popular’, ’language’, ’(’, ’nl’, ’##p’, ’)’, ’applications’, ’showing’, ’performance’, ’con’, ’##lu’, ’##al’, ’and’, ’architecture’.

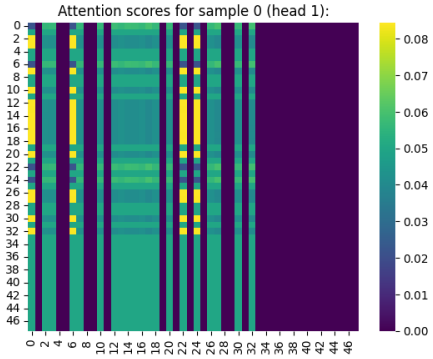
In Figure 11, the associated masks used internally by the model. Figures 11a-11c show which tokens are active during attention calculation: on the y-axis the query tokens, on the x-axis the key tokens. Sparse attention sets to 0 the contribution of value projection of masked tokens. Figure 11d shows the final head mask for each sample that describes which heads are and are not set to 0 in the output. It is important to note that each sample may generate different masking since it is activation-aware.



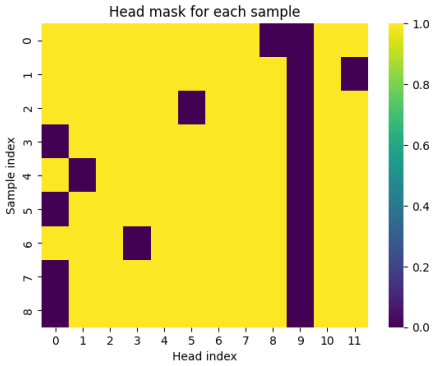
(a) Attention scores at the first layer



(b) Attention scores at the sixth layer



(c) Attention scores at the last layer



(d) Head mask in the last layer.

Fig. 11: Example of an application of sparse attention.