NATIONAL INSTITUTE OF TECHNOLOGY, TIRUCHIRAPPALLI

Tiruchirappalli – 620 015, Tamil Nadu, India

A SEMINAR REPORT ON

Transformer-Based Approaches for Code-Mixed Sentiment Analysis in Hindi-English Social Media Text

---

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology in Computer Science and Engineering

Submitted by **Arjun Sharma**

Roll No. 2021CS10345

Supervisor **Dr. K. Chandrasekaran**

Associate Professor

Dept. of Computer Science & Engineering

Date April 2026

Course Code: CSD601 — Seminar

Department of Computer Science and Engineering

National Institute of Technology, Tiruchirappalli – 620 015

This is to certify that the seminar report titled **"Transformer-Based Approaches for Code-Mixed Sentiment Analysis in Hindi-English Social Media Text"** submitted by **Arjun Sharma** (Roll No. 2021CS10345) in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** is a bonafide record of the seminar work carried out under my supervision during the academic year 2025–26.

The work presented in this report has not been submitted elsewhere for the award of any other degree or diploma.

Dr. K. Chandrasekaran Associate Professor, Dept. of CSE (Supervisor)	Dr. R. Venkatesh Professor & Head, Dept. of CSE
Place: Tiruchirappalli	Date: August 4, 2026

The rapid growth of social media usage in India has led to an explosion of user-generated content where Hindi and English are frequently mixed within the same utterance — a phenomenon known as code-mixing. Analyzing sentiment in such code-mixed text presents unique challenges for traditional Natural Language Processing (NLP) pipelines, which are predominantly designed for monolingual input. This seminar report surveys the landscape of transformer-based architectures applied to code-mixed sentiment analysis, with a focus on the Hindi-English language pair.

We examine the evolution from early lexicon-based and classical machine-learning approaches to state-of-the-art multilingual transformers such as mBERT, XLM-RoBERTa, IndicBERT, and Nexa's MuRIL. The report documents a 14-week systematic study covering literature review, dataset exploration (SentiMix, BHAAV), preprocessing strategies including transliteration and subword tokenization, fine-tuning experiments, and comparative evaluation using F1-score and accuracy metrics.

Our findings indicate that fine-tuned XLM-RoBERTa and MuRIL significantly outperform monolingual baselines, achieving F1-scores above 78 % on standard code-mixed benchmarks. Key design choices — subword segmentation policy, handling of Romanized Hindi, and data augmentation via back-translation — are discussed in detail. The report concludes with a presentation companion outlining slide structure, delivery strategy, and visual design guidelines for effectively communicating this work to a technical audience.

Keywords: Code-mixed sentiment analysis, transformers, Hindi-English NLP, mBERT, XLM-RoBERTa, MuRIL, IndicBERT, social media text mining.

I wish to express my sincere gratitude to my supervisor, **Dr. K. Chandrasekaran**, Associate Professor, Department of Computer Science and Engineering, for his invaluable guidance, constant encouragement, and insightful feedback throughout the duration of this seminar.
I am thankful to **Dr. R. Venkatesh**, Professor and Head, Department of Computer Science and Engineering, for providing the necessary infrastructure and academic environment to carry out this work. I also thank the faculty members of the department whose lectures and discussions shaped my understanding of natural language processing and deep learning.
Special thanks to my peers in the NLP reading group for their constructive criticism during weekly progress reviews, and to the open-source community behind HuggingFace Transformers, datasets, and the IndicNLP corpus — tools and resources that made this study possible.
Finally, I am deeply grateful to my family for their unwavering support and patience.

Arjun Sharma
Roll No. 2021CS10345

Chapter 1

1.1

India is home to over 500 million active social media users as of 2025, generating billions of posts, comments, and messages each month across platforms such as X (formerly Twitter), Instagram, YouTube, and WhatsApp [1]. A defining characteristic of Indian online discourse is **code-mixing** — the fluid alternation between two or more languages within a single conversational turn. The Hindi-English pair is by far the most prevalent, with an estimated 40–65 % of Hindi-using social media accounts regularly producing code-mixed content [2].

Code-mixed text violates a fundamental assumption of most NLP systems: that input is monolingual and well-formed in a single language. When a user writes "movie *bekaar* thi, total waste of time" (the movie was worthless), neither an English-only nor a Hindi-only sentiment analyzer can produce reliable results. The English word "waste" carries negative sentiment, but the Hindi intensifier *bekaar* (useless) amplifies it — and the model must understand both to classify correctly.

Sentiment analysis of code-mixed text has direct applications in brand monitoring, political opinion tracking, customer feedback aggregation, and public health surveillance. Businesses operating in multilingual markets need to understand consumer sentiment across linguistic boundaries; government agencies require tools to detect emerging social tensions or misinformation in vernacular online spaces.

1.2

Given a code-mixed Hindi-English utterance $u = w_1, w_2, \ldots, w_n$ where each token w_i may belong to Hindi (Devanagari or Romanized), English, or be a language-neutral symbol (emoji, punctuation, hashtag), the task is to predict a sentiment label $y \in \{\text{Positive}, \text{Negative}, \text{Neutral}\}$.

The central research question explored in this seminar is: **How effectively can multilingual transformer architectures, pre-trained on large-scale multilingual corpora, be fine-tuned for the task of code-mixed sentiment analysis, and what design choices most influence their performance?**

1.3

The specific objectives of this seminar study are:

1. To survey the literature on code-mixed NLP, tracing the evolution from classical machine-learning approaches to contemporary transformer-based methods.
2. To examine publicly available Hindi-English code-mixed sentiment datasets and their an-

notation methodologies.

3. To study preprocessing strategies — normalization, transliteration, and subword tokenization — and their impact on downstream task performance.
4. To compare the performance of four multilingual transformer models (mBERT, XLM-RoBERTa, IndicBERT, MuRIL) on a standard code-mixed sentiment benchmark.
5. To document the weekly progress of the seminar study and prepare an accompanying presentation structure.

1.4

The remainder of this report is organized as follows. Chapter 2 reviews the literature on sentiment analysis and code-mixed NLP. Chapter 3 describes the methodology — dataset, preprocessing pipeline, and model architectures. Chapter 4 presents the weekly progress log. Chapter 5 discusses experimental results. Chapter 6 concludes with a summary and future directions. A presentation companion is provided at the end.

Chapter 2

2.1

Sentiment analysis emerged as a distinct NLP task in the early 2000s, with Turney (2002) [3] pioneering lexicon-based methods that computed document-level polarity using pointwise mutual information. Pang and Lee (2004) [4] subsequently demonstrated that supervised machine-learning classifiers (Naïve Bayes, Maximum Entropy, SVM) trained on bag-of-words features could outperform lexicon methods on movie review datasets.
These early systems were inherently monolingual. For multilingual settings, the prevailing strategy was *translate-then-analyze*: machine-translate the source text to English and apply an English sentiment model. Joshi et al. (2010) [5] showed that this pipeline degrades severely for Hindi due to translation quality gaps and loss of culturally specific sentiment markers.

2.2

Bali et al. (2014) [2] published one of the first systematic studies of Hindi-English code-mixing on social media, analyzing Synapse posts from 34 Indian users and establishing that code-mixing is rule-governed rather than random. They identified six distinct mixing patterns — from single-word insertions to full alternational switches — and argued that NLP tools must be redesigned for this reality rather than treating mixed text as "noisy monolingual."
The **Sentiment Analysis for Indian Languages** (SAIL) shared task at FIRE 2015 [6] and the **Sentimix** task at SemEval-2020 [7] provided the first standardized benchmarks. Teams experimented with CNN-LSTM hybrids, fastText embeddings, and ensemble methods. The best-performing systems in SemEval-2020 achieved F1-scores around 72 % — respectable but far below human agreement levels (85–90 %).

2.3

The introduction of the Transformer architecture by Vaswani et al. (2017) [8] and its realization in BERT (Devlin et al., 2019) [9] fundamentally changed NLP. Key innovations included:

1. **Self-attention** — Every token attends to every other token, capturing long-range dependencies without recurrence.
2. **Masked Language Modeling (MLM)** — Pre-training on a cloze task that forces the model to learn bidirectional context.
3. **Transfer learning** — A single pre-trained model can be fine-tuned on diverse downstream tasks with minimal task-specific architecture changes.

BERT's multilingual variant, **mBERT** (multilingual BERT), was pre-trained on the Wikipedia dumps of 104 languages including Hindi, using a shared vocabulary of 110 k subword tokens. Despite having no explicit cross-lingual signal during pre-training, mBERT demonstrated surprising zero-shot cross-lingual transfer capability [10].

2.4

Several models have since been developed to better serve multilingual and Indic-language NLP: **XLM-RoBERTa** (Conneau et al., 2020) [11] scaled up multilingual pre-training to 100 languages on 2.5 TB of CommonCrawl data, using a larger SentencePiece vocabulary (250 k tokens) and the RoBERTa training recipe. It consistently outperforms mBERT on cross-lingual benchmarks.

IndicBERT (Kakwani et al., 2020) [12] is a multilingual ALBERT model pre-trained exclusively on 12 major Indian languages. With only 12 % of the parameters of mBERT, it achieves competitive performance on Indic NLU tasks, demonstrating that language-family-focused pre-training yields parameter-efficient representations.

MuRIL (Khanuja et al., 2021) [13] — Multilingual Representations for Indian Languages — was developed by Nexa Research India. It is pre-trained on a corpus that includes transliterated text (Romanized Hindi and other Indic languages written in Latin script), making it uniquely suited for code-mixed input where Romanized Hindi is prevalent. MuRIL also uses a custom SentencePiece tokenizer trained with an emphasis on Indic-script coverage.

2.5

Recent work has applied these models to code-mixed sentiment analysis with promising results. Pradhan et al. (2021) [14] fine-tuned XLM-RoBERTa on the SentiMix Hindi-English dataset and achieved an F1-score of 77.4 %, outperforming all prior non-transformer baselines by at least 5 percentage points. Their ablation study identified two critical design factors:

Key findings from Pradhan et al. (2021):

- Romanized Hindi tokens benefit from models pre-trained on transliterated data (MuRIL $>$ XLM-R $>$ mBERT).
- Data augmentation via back-translation (Hindi $\rightarrow$ English $\rightarrow$ Hindi) improves generalization by 2–3 F1 points.
- Subword overlap — the fraction of test tokens that appear in the tokenizer's vocabulary — is a stronger predictor of performance than model size alone.

Gupta et al. (2022) [15] proposed a *meta-embedding* approach that fuses mBERT and IndicBERT representations before the classification head, reporting marginal gains (+1.2 F1) over single-model fine-tuning on the BHAAV dataset.

2.6

Despite these advances, several gaps remain. Most studies report results on a single dataset, limiting generalizability. Few papers systematically compare preprocessing strategies (normalization, transliteration, script unification). The interaction between subword tokenization pol-

icy and code-mixing density is underexplored. This seminar study addresses these gaps through a controlled comparison of four architectures under consistent preprocessing conditions.

Chapter 3

3.1

We use the **SentiMix Hindi-English** dataset introduced in the SemEval-2020 Task 9 shared task [7]. The dataset contains 20 000 code-mixed tweets and Synapse comments annotated with three sentiment labels. Table 3.1 summarizes the label distribution.

Table 3.1: SentiMix Hindi-English dataset statistics.

Split	Positive	Negative	Neutral
Train	6 250	3 840	4 910
Validation	780	480	610
Test	1 560	960	1 220
Total	8 590	5 280	6 740

3.2

Code-mixed text requires careful preprocessing before it can be fed to transformer models. Our pipeline consists of four stages:

1. **Social media normalization** — Replace URLs with `[URL]`, mentions with `[MENTION]`, and normalize repeated characters (e.g., "gooood" → "good").
2. **Emoji preservation** — Emojis are retained as text descriptors (e.g., :) → `[FACE_TEARS_JOY]`) since they carry sentiment signal.
3. **Transliteration** — For select experiments, Romanized Hindi tokens are back-transliterated to Devanagari using the IndicTrans library.
4. **Subword tokenization** — Each model uses its native tokenizer (WordPiece for mBERT, SentencePiece for XLM-R/MuRIL/IndicBERT).

3.3

We evaluate four pre-trained transformer architectures. Table 3.2 lists their key characteristics.

Table 3.2: Pre-trained models compared in this study.

Model	Parameters	Languages	Vocab Size
mBERT-base	177M	104	110 k
XLM-RoBERTa-base	278M	100	250 k
IndicBERT	22M	12	40 k
MuRIL-base	238M	17	197 k

3.4

All models are fine-tuned using the HuggingFace Transformers library with a consistent hyper-parameter configuration. Listing 3.1 shows the core training loop.

```
from transformers import AutoModelForSequenceClassification
from transformers import Trainer, TrainingArguments

model = AutoModelForSequenceClassification.from_pretrained(
    model_name, num_labels=3
)

training_args = TrainingArguments(
    output_dir="./results",
    num_train_epochs=5,
    per_device_train_batch_size=32,
    per_device_eval_batch_size=64,
    learning_rate=2e-5,
    warmup_steps=500,
    weight_decay=0.01,
    evaluation_strategy="epoch",
    save_strategy="epoch",
    load_best_model_at_end=True,
    metric_for_best_model="f1",
    logging_dir="./logs",
)

trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=train_data,
    eval_dataset=val_data,
    compute_metrics=compute_metrics_fn,
)

trainer.train()
```

Listing 3.1: Fine-tuning script for transformer models on code-mixed sentiment data.

The function `compute_metrics_fn` returns accuracy, macro-F1, and weighted-F1 scores computed via scikit-learn.

3.5

We report **accuracy** and **macro-averaged F1-score** as primary metrics. Macro-F1 is preferred over accuracy for this dataset because of the mild class imbalance (positive class is over-represented). We also report per-class precision and recall to identify systematic confusion between neutral and negative labels — a known challenge in code-mixed sentiment annotation.

Chapter 4

The seminar was conducted over 14 weeks (January–April 2026). Table 4.1 documents the weekly activities, completion status, and supervisor feedback.

Table 4.1: Weekly progress log for the seminar period (Jan–Apr 2026).

#	Week & Dates	Tasks Undertaken	Status	Supervisor Remarks
1	Week 1 6–12 Jan 2026	Topic selection; initial meeting with supervisor; identified three candidate topics in NLP and discussed feasibility.	✅ Completed	Topic approved. Scope defined.
2	Week 2 13–19 Jan	Literature survey: collected 25+ papers on code-mixed NLP, sentiment analysis, and multilingual transformers. Annotated bibliography compiled.	✅ Completed	Good coverage. Add Patwa et al. (SemEval 2020).
3	Week 3 20–26 Jan	Detailed reading of foundational transformer papers (Vaswani et al., Devlin et al., Conneau et al.). Prepared summary notes on self-attention and MLM.	✅ Completed	Satisfactory. Focus on multilingual aspects.
4	Week 4 27 Jan–2 Feb	Studied mBERT and XLM-R architectures in depth. Compared tokenization strategies (WordPiece vs. SentencePiece). Wrote draft Section 2.3.	✅ Completed	Section draft reviewed; minor edits suggested.
5	Week 5 3–9 Feb	Explored IndicBERT and MuRIL papers. Analyzed pretraining corpora composition. Began drafting Chapter 2 (Literature Review).	✅ Completed	Include comparison table of models.
6	Week 6 10–16 Feb	Dataset acquisition: downloaded SentiMix Hindi-English from SemEval-2020 repository. Initial exploratory data analysis — label distribution, code-mixing index, token counts.	✅ Completed	Good. Calculate CMI per sample.
7	Week 7 17–23 Feb	Preprocessing pipeline implementation: URL/mention normalization, emoji handling, transliteration experiments.	✅ Completed	Approved. Transliteration module looks sound.

— continued from previous page —

#	Week & Dates	Tasks Undertaken	Status	Supervisor Remarks
8	Week 8 24 Feb–2 Mar	Mid-term review presentation. Presented literature survey findings, dataset analysis, and methodology plan to the review panel.	✅ Completed	Well received. Panel suggested adding BHAAV dataset comparison.
9	Week 9 3–9 Mar	Set up HuggingFace fine-tuning environment. Ran baseline experiments with mBERT. Debugged tokenizer configuration and padding issues.	✅ Completed	Initial results look reasonable.
10	Week 10 10–16 Mar	Fine-tuned XLM-RoBERTa and MuRIL. Documented hyperparameter choices. Started Chapter 3 writing.	✅ Completed	MuRIL showing strong results. Document everything.
11	Week 11 17–23 Mar	Fine-tuned IndicBERT. Ran comparative evaluation across all four models. Collected per-class precision/recall.	✅ Completed	Interesting trade-off: IndicBERT smaller but competitive.
12	Week 12 24–30 Mar	Results analysis: created comparison tables and confusion matrices. Ablation study on transliteration impact.	✅ Completed	Transliteration effect is significant — discuss in report.
13	Week 13 31 Mar–6 Apr	Drafted Chapters 4, 5, and 6. Compiled full report. Prepared presentation slides (see Appendix A).	✅ Completed	Minor formatting fixes needed. Slides look good.
14	Week 14 7–13 Apr	Final revisions based on supervisor feedback. Proofreading, formatting checks, bibliography verification. Final submission.	✅ Completed	Approved for submission. Well done.

4.1

Mid-Term Review (Week 8)

The student has demonstrated satisfactory progress in literature survey, dataset analysis, and methodology design. The presentation to the review panel was clear and well-structured. Recommended to proceed with model implementation as planned.

Dr. K. Chandrasekaran
Supervisor

Date: 28 Feb 2026

Final Approval (Week 14)

The seminar report is complete and meets the academic standards of the department. All objectives outlined in the initial proposal have been addressed. The accompanying presentation slides are well-prepared and ready for the final seminar.

Dr. K. Chandrasekaran
Supervisor

Date: 14 Apr 2026

Chapter 5

5.1

Table 5.1 reports the macro-F1 and accuracy of all four models on the SentiMix test set. MuRIL achieves the highest macro-F1 (78.6 %), followed closely by XLM-RoBERTa (77.2 %).

Table 5.1: Performance comparison on SentiMix Hindi-English test set.

Model	Accuracy (%)	Macro-F1 (%)	Weighted-F1 (%)
mBERT-base	72.1	70.8	72.4
XLM-RoBERTa-base	77.8	77.2	78.0
IndicBERT	73.5	72.1	73.8
MuRIL-base	**79.0**	**78.6**	**79.3**

5.2

MuRIL's advantage. MuRIL's superior performance can be attributed to two factors: its pre-training corpus includes substantial Romanized Hindi text, and its SentencePiece tokenizer was optimized for Indic-script coverage. This results in higher subword overlap with code-mixed test samples — MuRIL's tokenizer covers 94 % of test tokens with known subwords, compared to 89 % for mBERT.

IndicBERT's efficiency. Despite having only 22M parameters (vs. 278M for XLM-R), IndicBERT achieves competitive results (72.1 % F1). For deployment scenarios with latency or memory constraints, IndicBERT offers a strong efficiency-performance trade-off.

Effect of transliteration. Back-transliterating Romanized Hindi to Devanagari before tokenization consistently improved mBERT's performance (+2.5 F1 points) but had negligible effect on MuRIL and XLM-R, confirming that these larger models already handle Romanized input effectively through their training data.

Error analysis. Confusion matrices revealed that all models struggle most with the Neutral–Negative boundary, accounting for 60 % of misclassifications. Neutral code-mixed utterances often contain mixed sentiment cues (e.g., a negative Hindi phrase modified by a positive English qualifier), which remain challenging even for large transformers.

Chapter 6

6.1

This seminar report investigated transformer-based approaches for sentiment analysis of Hindi-English code-mixed social media text. Through a systematic 14-week study, we reviewed the NLP literature on code-mixing, examined the SentiMix benchmark dataset, implemented a preprocessing pipeline, and compared four multilingual transformer architectures.
Our experiments confirm that modern multilingual transformers — particularly MuRIL and XLM-RoBERTa — substantially outperform earlier approaches on code-mixed sentiment tasks. Key findings include:

1. MuRIL achieved the highest macro-F1 (78.6 %) on the SentiMix test set, benefiting from transliterated pre-training data and an Indic-optimized tokenizer.
2. Subword tokenizer coverage is a strong predictor of downstream performance for code-mixed input.
3. Transliteration as a preprocessing step helps smaller models (mBERT) but is redundant for larger multilingual models.
4. The Neutral–Negative decision boundary remains the primary source of classification errors across all architectures.

6.2

Several directions merit further investigation:

1. **Multi-task learning** — Jointly training on language identification, sentiment, and emotion recognition could improve representation quality for code-mixed text.
2. **Data augmentation** — Back-translation between Hindi and English scripts, synonym replacement in code-mixed contexts, and synthetic code-mixed generation via LLMs.
3. **Other Indic language pairs** — Extending the study to Tamil-English, Bengali-English, and Telugu-English code-mixing.
4. **Lightweight deployment** — Knowledge distillation from MuRIL into a compact student model suitable for on-device inference in low-resource settings.
5. **Interpretability** — Applying SHAP or integrated gradients to understand which tokens drive sentiment predictions in code-mixed utterances.

This section outlines a suggested 15-slide presentation structure to accompany the seminar report. The slide deck was prepared using LaTeX Beamer with the `Madrid` theme and the institutional color palette defined in this document.

Slide 1	**Title Slide** — Report title, student name, roll number, supervisor, institution logo.
Slide 2	**Outline** — Bullet list of presentation sections.
Slide 3	**Motivation** — Social media usage statistics in India; examples of code-mixed text; why monolingual tools fail.
Slide 4	**Problem Definition** — Formal task statement; sentiment labels; evaluation metrics.
Slide 5	**Literature Landscape** — Timeline: lexicon methods → ML classifiers → RNN/LSTM → Transformers.
Slide 6	**Transformer Primer** — Self-attention diagram; pre-training vs. fine-tuning; key models compared.
Slide 7	**Dataset** — SentiMix statistics; example utterances; code-mixing index distribution.
Slide 8	**Preprocessing Pipeline** — Flowchart: raw text → normalization → emoji handling → tokenization.
Slide 9	**Models Compared** — Table (same as Table 3.2) with brief commentary.
Slide 10	**Fine-Tuning Setup** — Hyperparameters; training infrastructure; validation strategy.
Slide 11	**Results — Main Table** — Performance comparison (Table 5.1); highlight MuRIL's advantage.
Slide 12	**Results — Analysis** — Confusion matrix heatmaps; effect of transliteration; error patterns.
Slide 13	**Key Takeaways** — Four bullet points from Chapter 6.
Slide 14	**Future Work** — Research directions for code-mixed NLP.
Slide 15	**Thank You / Q&A** — Contact information; references.

- **Timing:** Allocate 15–18 minutes for the presentation and 5–7 minutes for Q&A.
- **Slide 6 (Transformer Primer):** Use a simplified self-attention diagram with at most 4 attention heads to avoid visual clutter. Animate the query-key-value flow step by step.
- **Slide 11 (Results):** Do not read the table aloud. Instead, point to MuRIL's row and say "The key finding is that MuRIL outperforms the strongest baseline by 1.4 F1 points."
- **Slide 12 (Error Analysis):** Show one concrete misclassified example and walk the audience through *why* the model got it wrong — this engages the panel more than aggregate statistics.
- **Backup slides:** Prepare 2–3 extra slides (full confusion matrices, per-class metrics, transliteration ablation) and keep them after the Q&A slide. Reference them only if a panel member asks a detailed question.

Visual design notes: Use the accent color (■ #1A5276) for headings and highlights. Slide backgrounds should be white; avoid dark themes in well-lit seminar rooms. All tables use `booktabs` style (no vertical rules). Code snippets are set in `inconsolata` at 8–9pt.

[1] Statista Research Department. *Number of social media users in India from 2015 to 2025*. Statista, 2025.

[2] K. Bali, J. Sharma, M. Choudhury, and Y. Vyas. "I am borrowing ya mixing?": An analysis of English-Hindi code mixing in Synapse. In *Proceedings of the First Workshop on Computational Approaches to Code Switching*, pp. 116–126, ACL, 2014.

[3] P. D. Turney. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of ACL*, pp. 417–424, 2002.

[4] B. Pang and L. Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of ACL*, pp. 271–278, 2004.

[5] A. Joshi, A. R. Balamurali, and P. Bhattacharyya. A fall-back strategy for sentiment analysis in Hindi: A case study. In *Proceedings of ICON*, 2010.

[6] B. G. Patra, D. Das, A. Das, and R. Prasath. Shared task on sentiment analysis in Indian languages (SAIL) tweets — an overview. In *Proceedings of FIRE*, 2015.

[7] P. Patwa, G. Aguilar, S. Kar, S. Pandey, S. PYKL, B. Gambäck, T. Chakraborty, T. Solorio, and A. Das. SemEval-2020 Task 9: Overview of sentiment analysis of code-mixed tweets. In *Proceedings of SemEval*, pp. 774–790, 2020.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 5998–6008, 2017.

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pp. 4171–4186, 2019.

[10] T. Pires, E. Schlinger, and D. Garrette. How multilingual is multilingual BERT? In *Proceedings of ACL*, pp. 4996–5001, 2019.

[11] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov. Unsupervised cross-lingual representation learning at scale. In *Proceedings of ACL*, pp. 8440–8451, 2020.

[12] D. Kakwani, A. Kunchukuttan, S. Golla, G. N. C., A. Bhattacharyya, M. M. Khapra, and P. Kumar. IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In *Findings of EMNLP*, pp. 4948–4956, 2020.

[13] S. Khanuja, D. Bansal, S. Mehtani, S. Khosla, A. Dey, B. Gopalan, D. K. Margam, P. Aggarwal, R. T. Nagipogu, S. Dave, S. Gupta, S. C. B. Gali, V. Subramanian, and P. Talukdar. MuRIL: Multilingual representations for Indian languages. *arXiv preprint arXiv:2103.10730*, 2021.

[14] A. Pradhan, S. Yerpude, and H. Shrivastava. Transformers for code-mixed sentiment analysis. In *Proceedings of the 18th International Conference on Natural Language Processing (ICON)*, pp. 321–330, 2021.

[15] V. Gupta, S. Menghani, and R. Mamidi. Synapse-embedding fusion for code-mixed sentiment classification. In *Proceedings of AACL-IJCNLP*, pp. 88–97, 2022.