



AI-Based Semantic Understanding of Code Mixed Tamil-English Text in Informal Digital Communication

Aditya Raman¹, Sushmita Kundu², Aditya Balaji³, G.Paavai Anand⁴

Student, Computer Science and Engineering, SRM Institute of Science and Technology¹

Student, Computer Science and Engineering, SRM Institute of Science and Technology²

Student, Computer Science and Engineering, SRM Institute of Science and Technology³

Assistant Professor Sr.G, Computer Science and Engineering, SRM Institute of Science and Technology⁴

Abstract: Tamil-English code-mixing, popularly known as “Tanglish,” is a pervasive mode of informal digital communication among Tamil speakers, frequently rendered in Romanized script across social media and messaging platforms. Existing natural language processing (NLP) systems, largely trained on monolingual or formally structured corpora, are not optimized to interpret the semantic, sentiment, and intent-level nuances of such hybrid, non-standardized text. This paper proposes a comparative research framework to evaluate the capability of different computational approaches—traditional NLP techniques, BERT-based transformer models, and large language models (LLMs)—in understanding Tamil-English code-mixed text, alongside a proposed preprocessing pipeline designed to normalize Romanized Tamil variants prior to model inference. We outline a methodology for constructing an annotated code-mixed dataset, detail the architecture of each comparative model, and define evaluation metrics spanning accuracy, F1-score, and semantic similarity. The study aims to establish whether targeted preprocessing can meaningfully improve semantic fidelity for low-resource, code-mixed language understanding, contributing toward AI systems that better preserve and process minority and regional language expression in digital spaces.

Keywords: code-mixing, Tanglish, Tamil-English NLP, Romanized Tamil, semantic preservation, low-resource language processing, transformer models

I. INTRODUCTION

Digital communication among Tamil speakers frequently blends Tamil and English within a single utterance, often transliterated into Roman script due to keyboard habits and platform constraints. For example, a typical informal message reads: “Enaku iniki class iruku, assignment submit pannanum.” Such code-mixed, Romanized text lacks a consistent spelling standard, formal grammar, or a single canonical script, and is largely absent from the large annotated corpora used to train mainstream NLP systems.

Most widely used language models are trained on monolingual, formally written text—English news corpora or Tamil Unicode text—and are not tuned to informal, Romanized, code-mixed input. This creates a measurable gap in digital accessibility and language technology support for a form of Tamil expression that is functionally dominant in everyday messaging yet remains under-modeled in NLP research. The problem is compounded by the fact that a single Tamil word may be Romanized in several visually distinct ways depending on the writer’s dialect, typing habits, or phonetic interpretation, so systems trained on one spelling convention frequently fail on another.

This paper is motivated by the following research question: Can an AI model correctly understand the meaning, sentiment, and intent of Tamil-English code-mixed/Romanized text? We propose a comparative methodology—spanning traditional NLP, BERT-based models, LLMs, and a novel preprocessing pipeline—along with a dataset construction plan and an evaluation protocol, to guide future empirical investigation of this question.

The contribution of this work is threefold: (1) a dataset construction plan for annotated Tamil-English code-mixed text with sentiment, intent, and gold-standard meaning labels; (2) a proposed Romanized-Tamil normalization pipeline intended to reduce spelling-induced sparsity before model inference; and (3) a four-way comparative evaluation protocol spanning traditional machine learning, transformer-based, and LLM-based approaches, assessed jointly on classification accuracy and semantic similarity rather than accuracy alone.

II. RELATED WORK

A. Multilingual and Cross-Lingual Transformer Models

Multilingual transformer models such as mBERT and XLM-R [1], [2] are pretrained jointly across many languages and demonstrate strong cross-lingual transfer on formally written text. However, these models are not specifically optimized for the lexical irregularity introduced by informal Romanization, and their subword tokenizers frequently fragment non



standard spellings in ways that reduce representational quality.

B. Transliteration and Text Normalization for Indic Languages

Toolkits such as the IndicNLP library [5] support script conversion and normalization for Indic languages, but are typically applied to formally written Unicode text rather than social-media-style Romanized input. Rule-based and edit distance-based normalization approaches have shown promise in related transliteration tasks, motivating their adaptation to the code-mixed setting addressed in this paper.

C. Code-Mixed Sentiment and Intent Analysis

Prior work on Dravidian code-mixed sentiment analysis, including shared tasks on Tamil-English and Malayalam English code-mixed data [4], [8], has largely focused on classification accuracy rather than meaning preservation, leaving semantic similarity between model output and human-intended meaning comparatively unexplored. Surveys of code mixing in NLP [6] further note the scarcity of annotated, intent-labeled code-mixed corpora for Dravidian languages, which limits the reproducibility and scale of prior comparative studies.

D. Large Language Models for Low-Resource Language Understanding

Large-scale pretrained language models, built on the transformer architecture [3] and refined through approaches such as RoBERTa-style pretraining [7], have demonstrated emergent capability on low-resource and code-mixed inputs even without task-specific fine-tuning. However, systematic comparison of such LLMs against dedicated code-mixed pipelines—particularly on semantic fidelity rather than surface-level classification—remains limited in the Tamil-English setting.

E. Summary of Identified Gaps

Taken together, prior work leaves three gaps that this paper addresses: (i) comparative evaluation of Tamil-English code-mixed understanding rarely spans traditional, transformer-based, and LLM-based approaches within a single, consistent protocol; (ii) evaluation has focused predominantly on classification accuracy, with semantic-similarity-based meaning preservation largely unmeasured; and (iii) no widely available preprocessing pipeline specifically targets the normalization of Romanized Tamil spelling variation as a precursor to model inference. The methodology proposed in Section IV is designed to close each of these gaps in turn, as summarized in Table I.

TABLE I IDENTIFIED RESEARCH GAPS AND CORRESPONDING CONTRIBUTIONS

Gap	Limitation in prior work	Addressed by (this paper)
(i)	Comparisons rarely span traditional, transformer-based, and LLM-based approaches under one protocol	Four-way comparative evaluation protocol (Section IV-C)
(ii)	Evaluation centred on classification accuracy; meaning preservation largely unmeasured	Semantic-similarity metrics against a gold-standard meaning (Section IV-D)
(iii)	No widely available pipeline targets Romanized Tamil spelling variation	Romanized Tamil normalization pipeline (Section IV B)

III. PROBLEM DEFINITION

We define the problem along three sub-tasks operating on informal, messaging-style Tanglish text: (1) semantic understanding—recovering the intended meaning of a code-mixed sentence; (2) sentiment classification—identifying the polarity expressed; and (3) intent detection—categorizing the communicative purpose (e.g., statement, request, question). The scope of this study is restricted to short-form, informal, Romanized Tamil-English text typical of chat and social media communication, excluding formal Tamil Unicode text and full-sentence machine translation, which are addressed by existing systems.

Formally, given a code-mixed input sentence S composed of interleaved Tamil and English tokens, the objective is to produce: a semantic representation $M(S)$ approximating the human-intended meaning; a sentiment label from {positive, negative, neutral}; and an intent label from a predefined communicative-intent taxonomy, as illustrated in Fig. 1. Success is measured by how closely $M(S)$ aligns with a human-annotated gold meaning, not solely by classification accuracy.

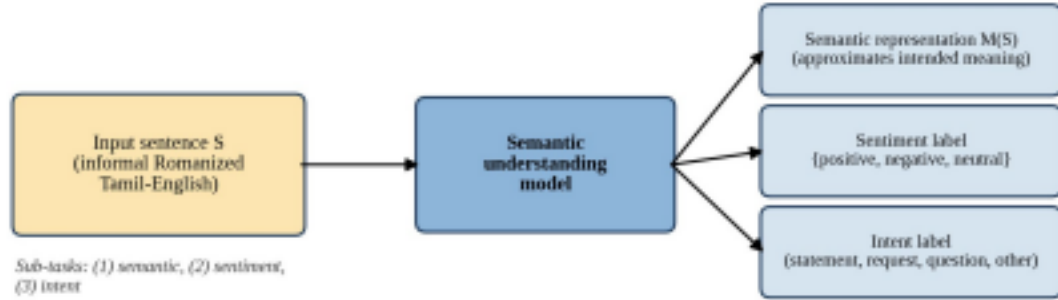


Fig. 1 Problem formulation: a code-mixed sentence S is mapped to a semantic representation M(S), a sentiment label, and an intent label.

IV. PROPOSED METHODOLOGY

A. Dataset Construction

A code-mixed corpus will be assembled from public social media posts, discussion forums, and synthetically generated conversational examples representative of everyday Tanglish usage [9]. Each sentence will be annotated with a sentiment label (positive, negative, neutral), an intent category (e.g., statement, request, question), and a human-verified gold standard meaning summary in English to support semantic similarity evaluation. Multiple annotators will label each sample, with inter-annotator agreement measured using Cohen's kappa to ensure label reliability. Annotation guidelines will explicitly address ambiguous Romanization variants and mixed-sentiment sentences to reduce labeling inconsistency. The annotation protocol will proceed in three passes (Fig. 2): an initial independent labeling pass by two annotators per sample, a reconciliation pass in which disagreements are discussed and resolved by a third senior annotator, and a final validation pass in which a random 10% subset is re-checked for consistency. Sentences on which annotators cannot reach agreement after reconciliation will be excluded from the gold evaluation set rather than resolved by majority vote, to avoid injecting artificial labels into the benchmark.

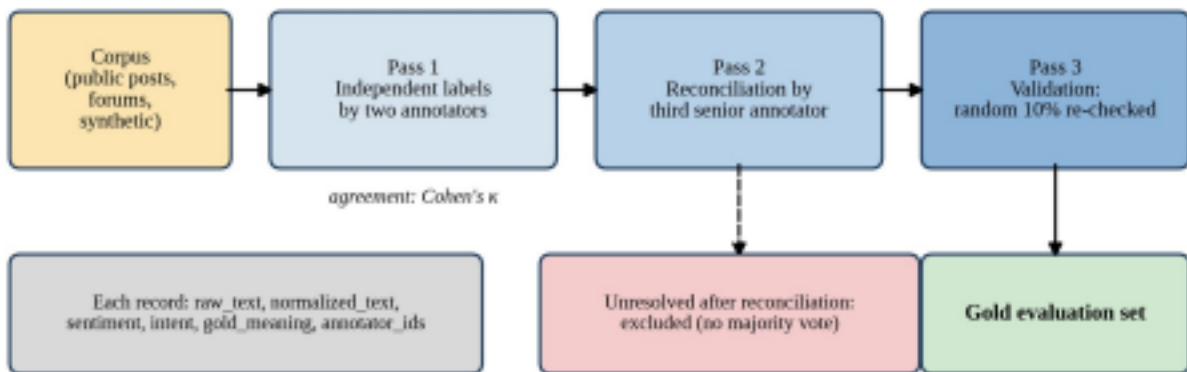


Fig. 2 Proposed three-pass annotation workflow and construction of the gold evaluation set.

TABLE II PROPOSED ANNOTATION SCHEMA

Field	Description
raw_text	Original Romanized code-mixed sentence as posted
normalized_text	Sentence after Romanized Tamil normalization
sentiment	One of: positive, negative, neutral
intent	Statement, request, question, or other
gold_meaning	Human-written English meaning summary
annotator_ids	Identifiers of annotators who labeled the sample



Each record in the resulting dataset will therefore carry both the raw and normalized forms of the sentence alongside its labels, allowing later researchers to evaluate models on either representation and to measure, in isolation, how much of any performance gain is attributable to normalization rather than to the underlying model architecture.

B. Preprocessing Pipeline (Proposed Approach)

- Word-level language identification to tag each token as Tamil or English within a sentence. □ Romanized Tamil normalization, clustering spelling variants of the same word into a canonical Romanized form using rule-based and edit-distance techniques.
- Transliteration of Tamil-tagged tokens into Tamil Unicode script using a transliteration model. □ Reconstruction of a normalized hybrid sentence as input for downstream models.

The overall pipeline is shown in Fig. 3. Fig. 4 illustrates the kind of spelling variation that the normalization step is intended to cluster into a single canonical form; the variants shown are illustrative examples rather than corpus statistics.

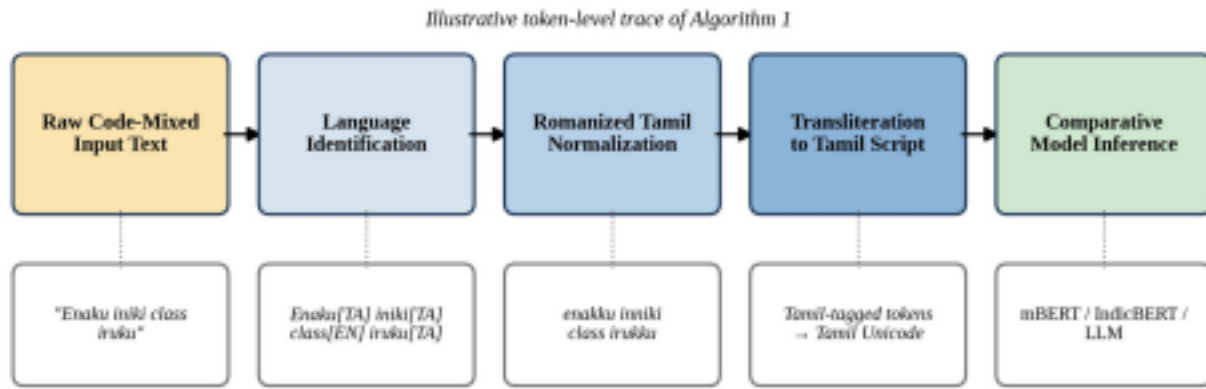


Fig.

3 Proposed preprocessing and inference pipeline for Tamil-English code-mixed text.

Algorithm 1 summarizes the proposed normalization procedure applied to each input sentence prior to model inference.

Algorithm 1: Romanized Tamil Normalization

Input: raw code-mixed sentence S

Output: normalized hybrid sentence S'

```

1: for each token  $t$  in  $S$  do
2:    $tag \leftarrow \text{LanguageIdentify}(t)$ 
3:   if  $tag = \text{Tamil}$  then
4:      $t' \leftarrow \text{ClusterVariant}(t)$  // edit-distance match to canonical form
5:      $t' \leftarrow \text{Transliterate}(t')$  // Roman  $\rightarrow$  Tamil script
6:   else
7:      $t' \leftarrow t$ 
8:   end if
9:   append  $t'$  to  $S'$ 
10: end for

```

Table III traces Algorithm 1 on the example sentence from Section I, whose intended meaning is “I have class today; I need to submit the assignment.”

TABLE III ILLUSTRATIVE WALK-THROUGH OF ALGORITHM 1

Token (raw)	Language tag	Canonical Roman form	Tamil script	Gloss
Enaku	Tamil	enaku	எனக்கு	to me
iniki	Tamil	inniki	இன்னிக்கு	today
class	English	class (unchanged)	—	class
iruku	Tamil	irukku	இருக்கு	is / have



assignment	English	assignment (unchanged)	—	assignment
submit	English	submit (unchanged)	—	submit
pannanum	Tamil	pannanum	பண்ணனும்	must do

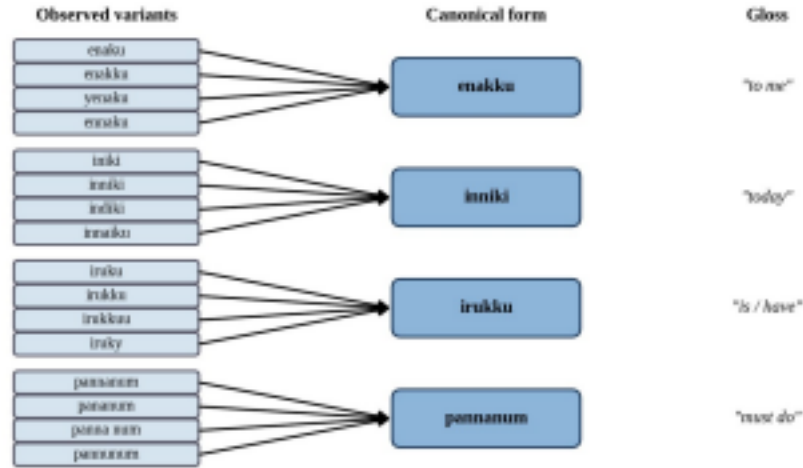


Fig. 4 Illustrative Romanized spelling variants of common Tamil words clustered into canonical forms.

C. Comparative Models

- *Traditional NLP baseline*: TF-IDF vectorization with classical classifiers (Naive Bayes, SVM) for sentiment and intent classification.
- *BERT-based model*: a multilingual transformer (e.g., mBERT or IndicBERT) fine-tuned directly on raw, unprocessed code-mixed text.
- *LLM-based approach*: a general-purpose large language model prompted in zero-/few-shot settings to extract meaning, sentiment, and intent.
- *Proposed approach*: a BERT-based model fine-tuned on text processed through the proposed normalization pipeline described in Section IV-B.

The four approaches and their data flow are depicted in Fig. 5 and compared in Table IV.

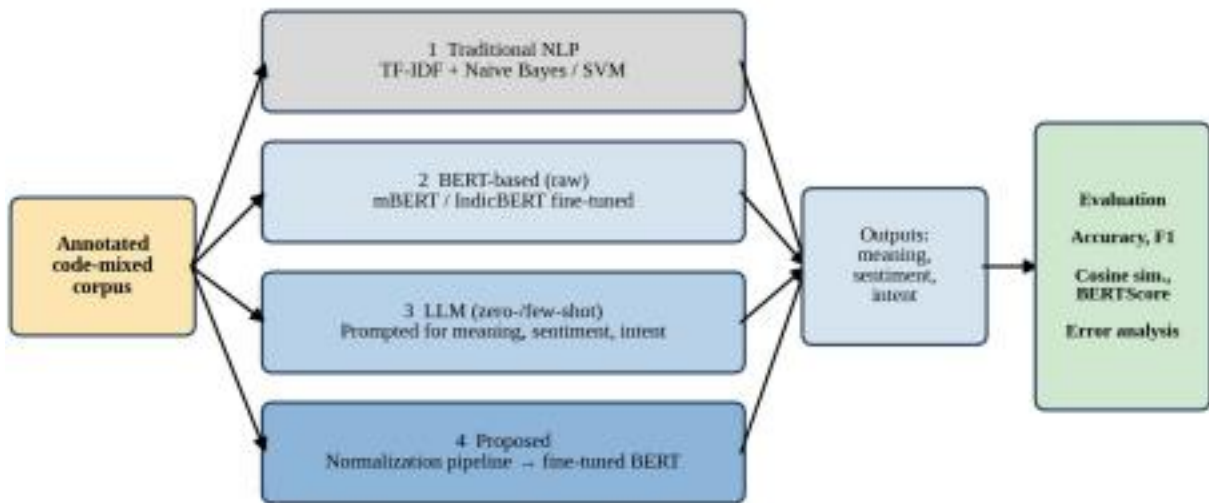


Fig. 5 Overview of the four-way comparative evaluation protocol.



TABLE IV COMPARATIVE OVERVIEW OF EVALUATED MODELS

Model	Approach	Expected strength
Traditional NLP	TF-IDF features with Naive Bayes / SVM classifiers	Fast, interpretable, low compute cost
BERT-based (raw)	mBERT / IndicBERT fine-tuned on unprocessed code-mixed text	Contextual embeddings without preprocessing overhead
LLM (zero-/few-shot)	General-purpose LLM prompted for meaning, sentiment, intent	Broad world knowledge and pretraining coverage
Proposed (Normalized + BERT)	Normalization pipeline followed by fine-tuned transformer	Reduced lexical sparsity, expected best semantic fidelity

D. Evaluation Metrics

Model performance will be assessed using accuracy and F1-score for the sentiment and intent classification tasks, and semantic similarity between model-inferred meaning and the human gold-standard summary, computed via sentence embedding cosine similarity and/or BERTScore (Table V). A qualitative error analysis will additionally examine code mixing-specific failure modes, such as transliteration ambiguity and conflicting sentiment cues across languages within a single sentence.

TABLE V EVALUATION METRICS BY TASK

Task	Metric	What it captures
Sentiment classification	Accuracy, F1-score	Label-level correctness
Intent detection	Accuracy, F1-score	Label-level correctness
Semantic understanding	Sentence-embedding cosine similarity and/or BERTScore against gold meaning	Meaning preservation
Error analysis (qualitative)	Manual inspection of failure cases	Transliteration ambiguity; conflicting cross-language sentiment cues

V. EXPECTED OUTCOMES AND DISCUSSION

A. Quantitative Expectations

We hypothesize that the proposed normalization pipeline will improve semantic similarity and classification performance relative to raw-input BERT and traditional baselines, as normalization is expected to reduce lexical sparsity introduced by inconsistent Romanization. LLMs are expected to perform competitively on meaning extraction given their broad pretraining exposure, but may still misinterpret culturally specific or highly informal phrasing without preprocessing support.

B. Qualitative Expectations

Beyond aggregate metrics, we expect qualitative differences across models: traditional baselines are likely to struggle with sentences containing mixed-language sentiment cues, while transformer-based and LLM approaches are expected to better capture pragmatic nuance once normalization reduces surface-form variability. Error analysis is expected to reveal systematic failure patterns tied to specific Romanization ambiguities rather than random noise.

C. Broader Impact

If validated, this framework would support more accurate sentiment analysis for customer feedback and social media monitoring, and more capable conversational AI for Tamil-speaking users, contributing to the digital preservation and accessibility of Tamil within AI-driven ecosystems. Such improvements are particularly relevant for minority and regional language communities, where mainstream AI tooling frequently underperforms relative to high-resource languages.



VI. LIMITATIONS

The proposed methodology is subject to several limitations. First, the scale and diversity of the annotated dataset will directly bound the generalizability of any empirical findings. Second, sentiment and intent annotation inherently involves subjective judgment, and residual annotator disagreement may affect evaluation reliability despite inter-annotator agreement checks. Third, the normalization pipeline is designed specifically for Tamil-English code-mixing and may require adaptation before it generalizes to other Dravidian-English language pairs. Fourth, LLM-based evaluation may be sensitive to prompt design, and results obtained under one prompting strategy may not generalize to another without further robustness testing.

VII. ETHICAL CONSIDERATIONS

Data collection from public social media sources will follow platform terms of service, and any personally identifiable information present in collected posts will be anonymized prior to annotation and release. Annotators will be briefed to handle potentially sensitive or offensive content according to a documented content-handling protocol. Because the dataset captures informal, community-specific language use, care will be taken to ensure that annotation guidelines do not penalize legitimate dialectal or regional variation as “noise,” so that the resulting models do not systematically disadvantage particular speaker groups within the Tamil-speaking community.

VIII. IMPLEMENTATION ROADMAP

Execution of the proposed methodology is planned across five phases, summarized in Table VI and Fig. 6, moving from dataset construction through comparative evaluation and public release of findings and resources.

TABLE VI PLANNED IMPLEMENTATION PHASES

Phase	Duration	Deliverable
1. Data Collection	Months 1–2	Raw code-mixed text corpus
2. Annotation	Months 2–3	Sentiment, intent, and gold meaning labels
3. Pipeline Build	Months 3–4	Normalization and transliteration pipeline
4. Model Training	Months 4–5	Four comparative model implementations
5. Evaluation & Release	Months 5–6	Metrics, error analysis, dataset release

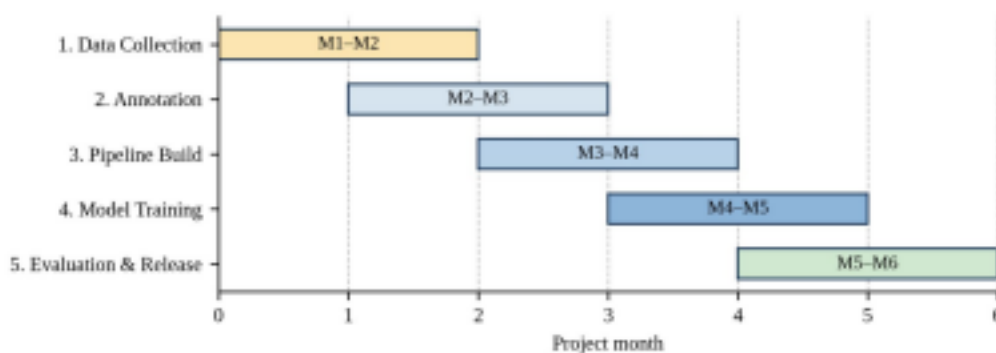


Fig. 6 Gantt view of the six-month implementation roadmap.

This phased plan is designed so that intermediate artifacts—particularly the annotated dataset and the normalization pipeline—remain independently reusable by other researchers working on Tamil-English or related Dravidian-English code-mixed language processing, even before the full comparative evaluation is complete.

IX. CONCLUSION AND FUTURE WORK

This paper proposes a structured methodology—comprising dataset construction, a Romanized-Tamil normalization pipeline, and a four-way model comparison—for evaluating AI-based semantic, sentiment, and intent understanding of Tamil-English code-mixed text. Future work includes implementing and empirically evaluating this framework, releasing the constructed dataset, extending the approach to voice-based code-mixed input, and generalizing the pipeline to other



Dravidian-English code-mixed pairs such as Telugu-English and Kannada-English.

X. REPRODUCIBILITY STATEMENT

To support reproducibility and reuse by other researchers, we intend to publicly release the annotated dataset, the annotation guidelines described in Section IV-A, and the implementation of the normalization pipeline described in Section IV-B upon completion of the empirical study. Model configurations and evaluation scripts for all four comparative approaches will also be documented so that the reported evaluation protocol can be independently replicated or extended to additional Dravidian-English code-mixed language pairs.

ACKNOWLEDGMENT

The authors thank the Department of Computer Science and Engineering, SRM Institute of Science and Technology, for their support.

REFERENCES

- [1]. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. NAACL-HLT*, 2019, pp. 4171–4186.
- [2]. A. Conneau *et al.*, "Unsupervised cross-lingual representation learning at scale," in *Proc. ACL*, 2020, pp. 8440–8451.
- [3]. A. Vaswani *et al.*, "Attention is all you need," in *Proc. NeurIPS*, 2017, pp. 5998–6008.
- [4]. B. R. Chakravarthi, N. Jose, S. Suryawanshi, E. Sherly, and J. P. McCrae, "A sentiment analysis dataset for code-mixed Malayalam-English," in *Proc. SLTU-CCURL*, 2020.
- [5]. A. Kunchukuttan, "The IndicNLP library," 2020. [Online]. Available: https://github.com/anoopkunchukuttan/indic_nlp_library
- [6]. S. Thara and P. Poornachandran, "Code-mixing: A brief survey," in *Proc. ICACCI*, 2018, pp. 2382–2388.
- [7]. Y. Liu *et al.*, "RoBERTa: A robustly optimized BERT pretraining approach," 2019, arXiv:1907.11692.
- [8]. B. R. Chakravarthi *et al.*, "Overview of the track on sentiment analysis for Dravidian languages in code-mixed text," in *Proc. FIRE*, 2020, pp. 21–24.
- [9]. T. Winata, A. Madotto, C. Wu, and P. Fung, "Code-switched language models using neural based synthetic data from parallel sentences," in *Proc. CoNLL*, 2019, pp. 271–280.