Prompt Engineering Using GPT for Word-Level Code-Mixed Language Identification in Low-Resource Dravidian Languages

Aniket Deroy[1,*,†], Subhankar Maity[1]

[1]IIT Kharagpur, Kharagpur, India

Abstract

Language Identification (LI) is crucial for various natural language processing tasks, serving as a foundational step in applications such as sentiment analysis, machine translation, and information retrieval. In multilingual societies like India, particularly among the youth engaging on social media, text often exhibits code-mixing, blending local languages with English at different linguistic levels. This phenomenon presents formidable challenges for LI systems, especially when languages intermingle within single words. Dravidian languages, prevalent in southern India, possess rich morphological structures yet suffer from under-representation in digital platforms, leading to the adoption of Roman or hybrid scripts for communication. This paper introduces a prompt based method for a shared task aimed at addressing word-level LI challenges in Dravidian languages. In this work, we leveraged GPT-3.5 Turbo to understand whether the large language models are able to classify words into correct categories correctly. Our findings show that the results on the Kannada dataset consistently outperformed the Tamil dataset across most metrics, indicating a higher accuracy and reliability in identifying and categorizing Kannada language instances. In contrast, the results on the Tamil dataset showed moderate performance, particularly needing improvement across all metrics.

Keywords

GPT, Word level identification, Classification, Low-resource Languages, Prompt Engineering

1. Introduction

Language Identification (LI) [1] is a fundamental task in natural language processing (NLP) that involves determining the language(s) present in a given text. This task is pivotal for numerous applications such as sentiment analysis, machine translation, information retrieval, and natural language understanding. Accurate LI becomes particularly challenging in multilingual societies where texts often exhibit code-mixing, a phenomenon where multiple languages co-occur within the same discourse, ranging from phrases to individual words.

In the context of India, a country renowned for its linguistic diversity [2], social media platforms reflect a vibrant mix of languages. Among the youth, in particular, there is a prevalent use of code-mixed text that blends local languages from the Dravidian language family with English. Dravidian languages, spoken predominantly in southern India, including languages like Kannada, Tamil, Malayalam, and Tulu, are characterized by rich morphological structures and diverse linguistic features. However, despite their significance, these languages face technological challenges, such as inadequate digital representation and script variations, which complicate language processing tasks like LI.

This paper focuses on addressing the specific challenges of word-level LI in Dravidian languages, leveraging the unique linguistic characteristics and code-mixed nature prevalent in social media and digital communications. We introduce a prompt engineering based method aimed at advancing LI capabilities in these languages by experimenting at different temperature values. By doing so, we aim to contribute to the broader goal of enhancing NLP tools for under-resourced languages, ultimately facilitating more accurate and inclusive language processing technologies.

Forum for Information Retrieval Evaluation, December 12-15, 2024, India

[*] Corresponding author.

roydanik18@kgpian.iitkgp.ac.in (A. Deroy); subhankar.ai@kgpian.iitkgp.ac.in (S. Maity)

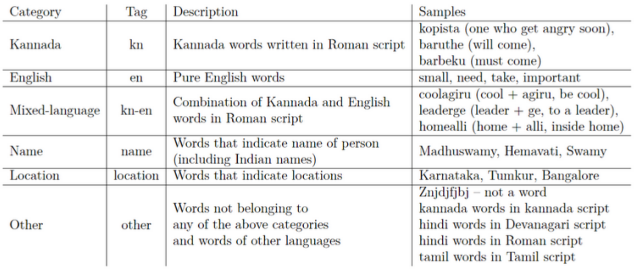
0000-0001-7190-5040 (A. Deroy); 0009-0001-1358-9534 (S. Maity)

An example of the dataset structure and word categories (adapted from-https://sites.google.com/view/coli-dravidian-2024/datasets?authuser=0) for the task is shown in Figure 1.

To the best of our knowledge, there is no work which explores unsupervised approaches for language identification. In this work, we leveraged GPT-3.5 Turbo [3] to understand whether the large language models is able to correctly classify words into correct categories. We experiment with GPT at different temperature values namely 0.7, 0.8, and 0.9.

GPT models are trained on large corpora from the internet, but the availability of high-quality data in Dravidian languages is limited compared to more widely spoken languages like English, Spanish, or Chinese. This means that GPT might not have been exposed to as much diverse or extensive data in these languages. Dravidian languages use distinct scripts (e.g., Tamil script for Tamil, Kannada script for Kannada). Moreover, code-mixing (where Dravidian languages are mixed with English or Hindi, often using the Roman script) is common on social media and informal communications. GPT's ability to handle code-mixed text varies and may not be as robust as its handling of pure English text.

Based on our experiments we observe that for Tamil and Kannada, GPT models have significant room for improvement.



Category	Tag	Description	Samples
Kannada	kn	Kannada words written in Roman script	kopista (one who get angry soon), baruthe (will come), barbeku (must come)
English	en	Pure English words	small, need, take, important
Mixed-language	kn-en	Combination of Kannada and English words in Roman script	coolagiru (cool + agiru, be cool), leaderge (leader + ge, to a leader), homealli (home + alli, inside home)
Name	name	Words that indicate name of person (including Indian names)	Madhuswamy, Hemavati, Swamy
Location	location	Words that indicate locations	Karnataka, Tumkur, Bangalore
Other	other	Words not belonging to any of the above categories and words of other languages	Znjdjfjbj – not a word kannada words in kannada script hindi words in Devanagari script hindi words in Roman script tamil words in Tamil script

Figure 1: Dataset structure and word categories for the task

2. Related Work

Language Identification (LI) [4, 5, 6] has been a crucial area of research within Natural Language Processing (NLP) due to its foundational role in various applications such as sentiment analysis, machine translation, and information retrieval. Traditional LI approaches [7, 8, 9, 10, 11] have primarily focused on monolingual or bilingual sentences, where clear boundaries between languages are assumed. However, these methods often struggle in multilingual and code-mixed environments, especially in regions like India, where linguistic diversity [12, 13, 14, 15, 16, 17] is high and social media usage reflects complex language practices.

Code-mixing [18, 19] presents unique challenges for LI systems. Early research in code-mixing focused on language pairs like English-Spanish or Hindi-English, where code-mixed texts predominantly used Roman scripts. Notable works explored the linguistic features of code-switched texts and highlighted the difficulties in segmenting and identifying languages at the word level. Similarly, Hindi-English code-mixed social media text, emphasizes the necessity for specialized LI models capable of handling intra-word language switches.

Dravidian languages [20] have been relatively underexplored in the context of LI, primarily due to the scarcity of annotated datasets and the complex morphological characteristics inherent to these languages. Previous efforts have developed initial datasets and models for LI in Dravidian languages; however, these models often fall short in handling code-mixed text, where Roman or hybrid scripts are employed. The Dravidian-CodeMix shared task aimed to address some of these gaps by introducing datasets for Tamil, Malayalam, and Kannada, which included code-mixed instances. Yet, the performance of models on these datasets indicated significant room for improvement, particularly in distinguishing between closely related languages and dialects.

Large Language Models (LLMs) [21, 22, 23, 24] like GPT-3 have shown promise in various NLP tasks, including LI. Previous works have demonstrated the capability of GPT-3 in performing zero-shot and few-shot learning, making it a potentially powerful tool for LI in resource-constrained settings. However, the application of LLMs [25, 26, 27, 28] to code-mixed and morphologically rich languages remains underexplored. Recent studies, have started to explore the use of transformers and pre-trained models for multilingual LI, but the effectiveness of these models in code-mixed Dravidian languages, particularly at the word level, requires further investigation.

Our work builds upon these existing efforts by focusing on a prompt-based method using GPT-3.5 Turbo to address word-level LI challenges in Dravidian languages. Unlike previous approaches, we leverage the linguistic diversity and code-mixed nature of the datasets to enhance the robustness of LI systems in detecting and classifying under-resourced languages. This study contributes to the growing body of research by providing a prompt engineering based method for Kannada, Tamil and evaluating the performance of advanced LLMs in this complex linguistic landscape.

3. Dataset

This shared task (adapted from https://sites.google.com/view/coli-dravidian-2024/datasets?authuser=0) consists of four distinct datasets [29, 30, 31, 32, 33, 34, 35, 36, 33, 37]:

1. **Tulu Dataset:** This dataset is composed of 7,171 code-mixed sentences gathered from YouTube videos. These sentences have been cleaned to remove non-textual elements and transliterated into Roman script. The dataset contains a total of 36,002 words, which are organized into six categories: ’English’, ’Kannada’, ’Tulu’, ’Location’, ’Name’, and ’Mixed-language’. The dynamic and context-specific nature of mixed-language words presents notable challenges for processing.
2. **Kannada Dataset:** This Kannada dataset contains 14,847 tokens in Roman script and is divided into six categories: ’English’, ’Kannada’, ’Name’, ’Mixed-Language’, ’Other’, and ’Location’. The primary goal of the dataset is to improve techniques for language identification and classification, particularly for Kannada-English code-mixed texts.
3. **Tamil Dataset:** The Tamil dataset comprises 17,568 tokens, created using a methodology similar to that employed for the Kannada and Tulu datasets. It is divided into six categories and is designed to facilitate a range of NLP tasks tailored to the Tamil language.
4. **Malayalam Dataset:** This dataset consists of 25,035 tokens classified into 7 categories: ’Number’, ’Mixed’, ’English’, ’Location’, ’Name’, ’sym’ (for sentence boundaries), and ’Malayalam’. This dataset offers extensive coverage for NLP tasks and includes the ’Number’ category for numerical values, akin to the structure of the other provided datasets.

We participated in shared tasks based on two languages, namely, *Kannada* and *Tamil.* The test dataset size for Kannada is 2502 words. The test dataset size for Tamil is 2024 words.

4. Task Definition

The goal of this task is to classify individual words from a code-mixed text into predefined categories or classes. The words should be classified into the following categories:

- ***English***: Words or phrases that are in the English language (e.g., hello, book, run).
- ***Dravidian***: Words or phrases that are in the Kannada language or Tamil language.
- ***Mixed***: Words or phrases that mix English, Kannada, or Tamil or combine elements from both languages.
- ***Name***: Proper nouns, including names of people, organizations, etc. (e.g., John, Infosys).
- ***Location***: Names of places, such as cities, countries, or landmarks (e.g., Bangalore, India, Taj Mahal).
- ***Symbol***: Symbols or punctuation marks used in the text (e.g., *, =, #, ;).
- ***Other***: Words or elements that do not fit into the above categories or are ambiguous.

5. Methodology

5.1. Why Prompting?

Prompting [38] to solve a word-level classification problem often arises from the need to accurately identify and categorize individual words within texts that exhibit code-mixing or multilingual content. Next we discuss the reasons why the problem of language identification is tried via prompting through GPT-3.5 Turbo:

- **Code-Mixing in Texts:** In multilingual societies or digital platforms, texts frequently mix languages, such as local languages with English [39]. Understanding which language each word belongs to is essential for applications like sentiment analysis, machine translation, and information retrieval.
- **Accuracy in Language Processing:** For effective natural language processing (NLP), identifying the language of each word enhances the accuracy of subsequent tasks [40]. It ensures that language-specific models or algorithms are applied correctly.
- **Contextual Understanding:** Words in code-mixed texts can change meaning based on the language they are derived from [41]. Accurate language identification at the word level aids in preserving context and meaning during NLP tasks.
- **Challenges and Innovation:** Word-level classification poses challenges due to the intricacies of code-mixed languages, where words may seamlessly blend multiple languages or scripts [42]. Addressing these challenges fosters innovation in NLP methodologies and technologies.

In summary, prompting to solve word-level classification problems stems from the practical need to accurately handle code-mixed languages and optimize language-specific processing in diverse linguistic contexts.

5.2. Prompt Engineering-Based Approach

We used the GPT-3.5 Turbo model via prompting[1] to solve the classification task in Zero-shot mode. After the prompt is provided to the LLM, the following steps occur internally while generating the output. GPT-3.5 Turbo follows a decoder-only architecture. So based on [3, 26, 43], we list these steps, summarizing the prompting approach using GPT-3.5 Turbo. The set of steps [26, 43] for GPT-3.5 Turbo [3] is as follows:

Step 1: Tokenization

- **Prompt:** $X = [x_1, x_2, \dots, x_n]$
- The input text (prompt) is first tokenized into smaller units called tokens. These tokens are often subwords or characters, depending on the model's design.
- **Tokenized Input:** $T = [t_1, t_2, \dots, t_m]$

Step 2: Embedding

- Each token is converted into a high-dimensional vector (embedding) using an embedding matrix E.
- **Embedding Matrix:** $E \in \mathbb{R}^{|V| \times d}$, where $|V|$ is the size of the vocabulary and d is the embedding dimension.
- **Embedded Tokens:** $T_{\text{emb}} = [E(t_1), E(t_2), \dots, E(t_m)]$

Step 3: Positional Encoding

[1] https://platform.openai.com/docs/models/gpt-3-5-turbo

- Since the model processes sequences, it adds positional information to the embeddings to capture the order of tokens.
- **Positional Encoding:** $P(t_i)$
- **Input to the Model:** $Z = T_{\text{emb}} + P$

Step 4: Attention Mechanism (Transformer Architecture)

- **Attention Score Calculation:** The model computes attention scores to determine the importance of each token relative to others in the sequence.
- **Attention Formula:**
$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$
- where Q (query), K (key), and V (value) are linear transformations of the input Z.
- This attention mechanism is applied multiple times through multi-head attention, allowing the model to focus on different parts of the sequence simultaneously.

Step 5: Feedforward Neural Networks

- The output of the attention mechanism is passed through feedforward neural networks, which apply non-linear transformations.
- **Feedforward Layer:**
$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{2}$$
- where W_1, W_2 are weight matrices and b_1, b_2 are biases.

Step 6: Stacking Layers

- Multiple layers of attention and feedforward networks are stacked, each with its own set of parameters. This forms the ”deep” in deep learning.
- **Layer Output:**
$$H^{(l)} = \text{LayerNorm}(Z^{(l)} + \text{Attention}(Q^{(l)}, K^{(l)}, V^{(l)})) \tag{3}$$
$$Z^{(l+1)} = \text{LayerNorm}(H^{(l)} + \text{FFN}(H^{(l)})) \tag{4}$$

Step 7: Output Generation

- The final output of the stacked layers is a sequence of vectors.
- These vectors are projected back into the token space using a softmax layer to predict the next token or word in the sequence.
- **Softmax Function:**
$$P(y_i|X) = \frac{\exp(Z_i)}{\sum_{j=1}^{|V|} \exp(Z_j)} \tag{5}$$
- where Z_i is the logit corresponding to token i in the vocabulary.
- The model generates the next token in the sequence based on the probability distribution, and the process repeats until the end of the output sequence is reached.

Step 8: Decoding

- The predicted tokens are then decoded back into text, forming the final output.
- **Output Text:** $Y = [y_1, y_2, \ldots, y_k]$

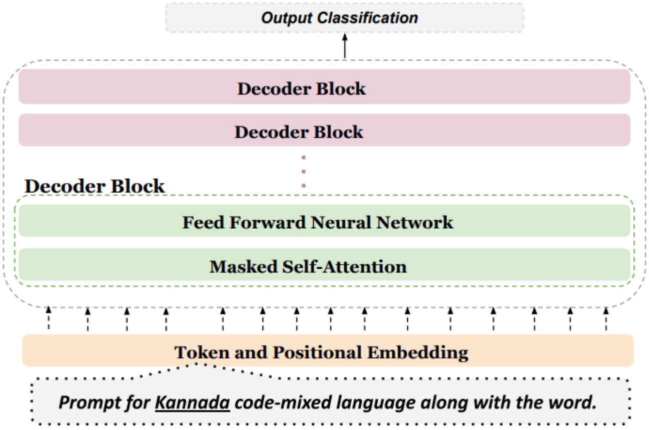

Output Classification
Decoder Block
Decoder Block
Decoder Block
Feed Forward Neural Network
Masked Self-Attention
Token and Positional Embedding
Prompt for Kannada code-mixed language along with the word.

Figure 2: An overview of GPT-3.5 Turbo for Kannada code-mixed language classification.

We used the following prompt for Kannada language for the purpose of classification: ”*Please identify which category the word is in English, Kannada, Mixed, Name, Location, Symbol and Other. Please state en, kn, mixed, name, location, sym and other. The word is <Word>.*” The figure representing the methodology is shown in Figure 2.

We used the following prompt for Tamil language for the purpose of classification: ”*Please identify which category the word is in English, Tamil, Mixed, Name, Location, Symbol and Other. Please state en, tm, tmen, name, Location, sym and Other. The word is <Word>.*” The figure representing the methodology is shown in Figure 3.

Corresponding to the two distinct prompts (for Kannada and Tamil) the two distinct figures are stated (Figure 2 and Figure 3).

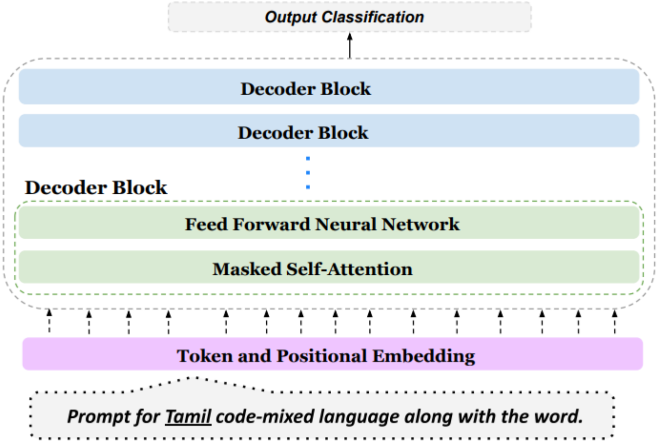

Output Classification
Decoder Block
Decoder Block
Decoder Block
Feed Forward Neural Network
Masked Self-Attention
Token and Positional Embedding
Prompt for Tamil code-mixed language along with the word.

Figure 3: An overview of GPT-3.5 Turbo for Tamil code-mixed language classification.

6. Results

Table 1 presents metrics comparing the performance of two language identification models, one for Tamil and the other for Kannada. Our Team ranked in the 10th position for both tasks. Here’s a detailed discussion of each metric.

Next, we discuss the metric values as well as their corresponding analysis to explain the results in the Table 1. For Tamil language, the macro F1 score is 0.3312. This suggests that the model achieves a balanced performance in terms of precision and recall for Tamil language identification. However, it indicates there is room for improvement in correctly identifying both positive and negative instances.

Table 1
Comparison of various metrics for Word level identification in code-mixed languages in two languages-Tamil, Kannada. The team name is **TextTitans** and the username is **roydanik18**. All metric values have been reported in this table.

Metric	Tamil	Kannada
Macro F1	0.3312	0.4493
Macro Precision	0.3259	0.5474
Macro Recall	0.3657	0.4241
Weighted F1	0.7022	0.6725
Weighted Precision	0.7559	0.7191
Weighted Recall	0.6689	0.6994
Accuracy	0.6689	0.6994

For Kannada, the macro F1 score is 0.4493. This score is higher compared to Tamil, indicating a better overall balance between precision and recall for Kannada language identification. The model for Kannada performs better in correctly classifying instances across the dataset.

For Tamil, the macro precision score is 0.3259. For Kannada, the macro precision score is 0.5474. This score indicates a higher accuracy in positive predictions for Kannada compared to Tamil, suggesting better precision in correctly identifying Kannada instances.

For Tamil, the macro recall score is 0.3657. The macro recall score is 0.4241 for Kannada. This score indicates a slightly higher ability to identify Kannada instances correctly compared to Tamil.

For Tamil, the weighted F1 score is 0.7022. This metric considers the F1 score weighted by the number of samples in each class, indicating a solid overall performance for Tamil language identification. For Kannada, the weighted F1 score is 0.6725. This indicates a slightly lower weighted F1 score compared to Tamil, suggesting a nuanced performance when considering class distribution.

For Tamil, the weighted precision score is 0.7559. This metric reflects the precision of the model when adjusted for the distribution of samples across Tamil language classes. For Kannada, the weighted precision score is 0.7191. This score indicates a slightly lower weighted precision compared to Tamil, reflecting the model's ability to accurately predict positive instances in Kannada.

For Tamil, the weighted recall score is 0.6689. This metric demonstrates the model's ability to identify all positive instances within the Tamil language classes when considering class distribution. For Kannada, the weighted recall score is 0.6994. This score indicates a slightly higher ability to correctly identify positive instances within Kannada language classes compared to Tamil.

For Tamil, the accuracy score is 0.6689. This metric measures the overall correctness of the model's predictions for Tamil language identification. For Kannada, the accuracy score is 0.6994. This indicates a slightly higher overall correctness in predictions for Kannada compared to Tamil.

The metrics highlight differences in performance between the Tamil and Kannada language identification models across various evaluation criteria. These metrics provide insights into the strengths and areas for improvement in both models, guiding further optimizations and enhancements for accurate language identification tasks in practical applications.

The weighted precision, recall, and f1-scores being higher than the macro precision, recall, and f1-scores shows that dataset likely has an imbalance, with some classes having many more samples than others. The weighted F1 score takes this into account by giving more importance to the performance on larger classes. The model is performing well on the classes that contribute the most to the overall accuracy. This could mean that it is effectively identifying the majority classes but may struggle with minority classes for both languages.

Weighted precision being higher than weighted recall suggests that the model performs better on the more frequent classes in the dataset. This means it is more effective at correctly identifying positive instances for these majority classes for both datasets. For Kannada dataset, a higher macro precision than recall may suggest that the model is conservative in its positive predictions, prioritizing accuracy over completeness. For Tamil dataset, a higher macro recall than precision suggests that while the model is effective at capturing relevant instances, it may not be very reliable in its predictions.

7. Conclusion

In this study, we investigated the effectiveness of language identification models for Tamil and Kannada using the advanced capabilities of GPT-3.5 Turbo via prompting. Language identification is a crucial preliminary step in various natural language processing applications, including sentiment analysis, machine translation, and information retrieval. Our research focused on evaluating and comparing the performance of these models across multiple metrics: macro F1 score, macro precision, macro recall, weighted F1, weighted precision, weighted recall, and accuracy. The results reveal notable distinctions between the Tamil and Kannada models. Kannada consistently demonstrated superior performance across most metrics. This indicates that the GPT for Kannada effectively identifies and categorizes Kannada language instances with greater accuracy and reliability. Conversely, while the Tamil model exhibited moderate performance, there remains room for improvement, particularly in precision and recall metrics.

The methodology employed in this research leveraged GPT-3.5 Turbo via prompting, harnessing its natural language processing capabilities to handle code-mixed texts and diverse linguistic patterns prevalent in real-world applications. This approach allowed for comprehensive evaluation under varying linguistic contexts, ensuring robustness and applicability in multilingual environments.

Moving forward, further refinements in model training and dataset augmentation could enhance the performance of language identification systems for both Tamil and Kannada. Future research efforts may focus on incorporating additional linguistic features, optimizing model architectures, and expanding datasets to include more diverse linguistic variations and challenges. In conclusion, this study underscores the importance of tailored approaches in language identification, particularly in multilingual settings like India where linguistic diversity is prominent. By advancing the capabilities of language identification models through innovative methodologies such as GPT-3.5 Turbo via prompting, we contribute to the broader goal of improving language processing technologies for diverse and under-resourced languages, fostering more accurate and inclusive natural language understanding systems. Future work would focus on improving the prompts to improve accuracy on the language identification task.

Declaration on Generative AI

During the preparation of this work, the author(s) used ChatGPT in order to: Drafting content, Grammar and spelling check, etc. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] T. Jauhiainen, H. Jauhiainen, K. Linden, Automatic language identification using word embeddings and normalized log probabilities, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018, pp. 1146–1153.

[2] A. Mandal, et al., Multilingual language identification based on recurrent neural networks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, 2019, pp. 3345–3352.

[3] T. B. Brown, Language models are few-shot learners, arXiv preprint ArXiv:2005.14165 (2020).

[4] T. Jauhiainen, H. Jauhiainen, K. Linden, A survey on automatic language identification in written texts, in: Journal of Artificial Intelligence Research, volume 65, 2019, pp. 675–782.

[5] Y. Muthusamy, R. A. Cole, B. T. Oshika, Automatic language identification: A review/tutorial, in: IEEE Signal Processing Magazine, volume 11, 1994, pp. 33–41.

[6] J. Tiedemann, News from opus-a collection of multilingual parallel corpora with tools and interfaces, in: Recent advances in natural language processing (vol. 5), 2009, pp. 237–248.

[7] M. Zampieri, B. Gebre, S. Malmasi, A system for tweet normalization and part-of-speech tagging of non-standard italian, Proceedings of the first workshop on noisy user-generated text (2014) 61–70.

[8] S. Malmasi, M. Dras, Discriminating between similar languages and dialects using crfs and svms, in: Proceedings of the International Conference Recent Advances in Natural Language Processing, 2015, pp. 140–147.

[9] B. King, S. Abney, Labeling the languages of words in mixed-language documents using weakly supervised methods, in: Proceedings of the 2013 conference of the North American chapter of the association for computational linguistics: human language technologies, 1996, pp. 1110–1119.

[10] V. Singh, J. Lal, A. Sharma, et al., Automatic language identification system using machine learning techniques: A review, Journal of Ambient Intelligence and Humanized Computing 9 (2018) 417–425.

[11] S. Zwarts, P. McNamee, Proceedings of the vardial workshop series on variation of languages in dialects and varieties, in: Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial), 2017, pp. 9–18.

[12] J. Tiedemann, Automatic identification of cognates and false friends in bilingual wordlists, in: Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1, 2003, pp. 116–119.

[13] T. Jauhiainen, H. Jauhiainen, K. Linden, Automatic detection of compound words in multiple languages, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, 2017, pp. 2047–2052.

[14] A. Mandal, A. Das, P. Pakray, Automatic language identification based on lexical and syntactic features, in: Proceedings of the 6th International Conference on Computer Applications in Biotechnology, 2015, pp. 213–221.

[15] R. Gamba, A. Das, Comparing the level of code-switching in corpora, in: Proceedings of the 10th edition of the Language Resources and Evaluation Conference, 2016, pp. 14–20.

[16] B. King, S. Abney, Labeling the languages of words in mixed-language documents using weakly supervised methods, in: Proceedings of the 2013 conference of the North American chapter of the association for computational linguistics: human language technologies, 2014, pp. 1110–1119.

[17] P. Molaei, et al., Cross-language identification of dravidian languages using transformer models, Proceedings of the Workshop on Computational Approaches to Linguistic Code-Switching (2020) 45–52.

[18] M. Zampieri, S. Malmasi, Y. Scherrer, Predicting the language of informal code-switched text, in: Proceedings of the 6th Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2019), 2019, pp. 135–144.

[19] I. Bardaji, et al., Language identification using cross-lingual word embeddings, Natural Language Engineering 18 (2012) 515–531.

[20] B. R. Chakravarthi, R. Priyadharshini, J. Jose, P. Kumaresan, S. Muralidaran, Findings of the shared task on sentiment analysis for dravidian languages in code-mixed text, in: Proceedings of the first workshop on speech and language technologies for Dravidian languages, 2021, pp. 133–139.

[21] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, Language models are unsupervised multitask learners, in: OpenAI Blog, volume 1, 2019.

[22] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, Journal of Machine Learning Research 21 (2020) 1–67.

[23] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 2019, pp. 4171–4186.

[24] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov,

Roberta: A robustly optimized bert pretraining approach, in: arXiv preprint arXiv:1907.11692, 2019.

[25] W. X. Zhao, K. Zhou, J. Li, X. Tang, J. J. Wang, J. Liu, T. Wang, Y. Bao, J.-R. Wen, A survey of large language models, in: arXiv preprint arXiv:2303.18223, 2023.

[26] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017) 5998–6008.

[27] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, Fine-tuning gpt-2 for human-like text generation, in: arXiv preprint arXiv:1907.11692, 2019.

[28] R. Zellers, A. Holtzman, H. Rashkin, Y. Bisk, A. Farhadi, F. Roesner, Y. Choi, Defending against neural fake news, in: Advances in Neural Information Processing Systems, volume 32, 2019, pp. 9054–9065.

[29] A. Hegde, M. D. Anusha, S. Coelho, H. L. Shashirekha, B. R. Chakravarthi, Corpus creation for sentiment analysis in code-mixed tulu text, in: Proceedings of the 1st Annual Meeting of the ELRA/ISCA Special Interest Group on Under-Resourced Languages, 2022, pp. 33–40.

[30] S. H. Lakshmaiah, F. Balouchzahi, M. D. Anusha, G. Sidorov, Coli-machine learning approaches for code-mixed language identification at the word level in kannada-english texts, Acta Polytechnica Hungarica 19 (2022).

[31] A. Hegde, F. Balouchzahi, S. Coelho, S. HL, H. A. Nayel, S. Butt, Coli@ fire2023: Findings of word-level language identification in code-mixed tulu text, in: Proceedings of the 15th Annual Meeting of the Forum for Information Retrieval Evaluation, 2023, pp. 25–26.

[32] A. Hegde, F. Balouchzahi, S. Coelho, H. Shashirekha, H. A. Nayel, S. Butt, Overview of coli-tunglish: Word-level language identification in code-mixed tulu text at fire 2023., in: FIRE (Working Notes), 2023, pp. 179–190.

[33] F. Balouchzahi, S. Butt, A. Hegde, N. Ashraf, H. Shashirekha, G. Sidorov, A. Gelbukh, Overview of coli-kanglish: Word Level Language Identification in Code-mixed Kannada-English Texts at Icon 2022, in: Proceedings of the 19th International Conference on Natural Language Processing (ICON): Shared Task on Word Level Language Identification in Code-mixed Kannada-English Texts, 2022, pp. 38–45.

[34] A. Hegde, F. Balouchzahi, S. Coelho, S. H L, H. A. Nayel, S. Butt, Coli@fire2023: Findings of word-level language identification in code-mixed tulu text, FIRE '23, Association for Computing Machinery, New York, NY, USA, 2024, p. 25–26. URL: https://doi.org/10.1145/3632754.3633075. doi:10.1145/3632754.3633075.

[35] S. Hosahalli Lakshmaiah, F. Balouchzahi, A. Mudoor Devadas, G. Sidorov, CoLI-Machine Learning Approaches for Code-mixed Language Identification at the Word Level in Kannada-English Texts, acta polytechnica hungarica (2022).

[36] A. Hegde, F. Balouchzahi, S. Butt, S. Coelho, K. G, H. S Kumar, S. D, S. Hosahalli Lakshmaiah, A. Agrawal, Overview of CoLI-Dravidian: Word-level Code-mixed Language Identification in Dravidian Languages, in: Forum for Information Retrieval Evaluation FIRE - 2024, 2024.

[37] F. Balouchzahi, S. Butt, A. Hegde, N. Ashraf, S. Hosahalli Lakshmaiah, G. Sidorov, A. Gelbukh, Overview of CoLI-Kanglish: Word Level Language Identification in Code-mixed Kannada-English Texts at ICON 2022, in: 19th International Conference on Natural Language Processing Proceedings, 2022.

[38] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, G. Neubig, Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing, ACM Computing Surveys 55 (2023) 1–35.

[39] C. Lee, Multilingualism online, Routledge, 2016.

[40] K. Chowdhary, K. Chowdhary, Natural language processing, Fundamentals of artificial intelligence (2020) 603–649.

[41] G. Takawane, A. Phaltankar, V. Patwardhan, A. Patil, R. Joshi, M. S. Takalikar, Language augmentation approach for code-mixed text classification, Natural Language Processing Journal 5 (2023) 100042.

[42] A. Mangla, R. K. Bansal, S. Bansal, Code-mixing and code-switching on social media text: A brief

survey, in: 2023 IEEE International Conference on Computer Vision and Machine Intelligence (CVMI), IEEE, 2023, pp. 1–5.

[43] A. Radford, Improving language understanding by generative pre-training (2018).