Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Adapter Based Fine-Tuning of Pre-Trained Multilingual Language Models for Code-Mixed and Code-Switched Text Classification

Himashi Rathnayake (himashi.17@cse.mrt.ac.lk)
University of Moratuwa
Janani Sumanapala
University of Moratuwa
Raveesha Rukshani
University of Moratuwa
Surangika Ranathunga
University of Moratuwa

Research Article

Keywords: code-switching, code-mixing, text classification, Sinhala, XLM-R, adapter

Posted Date: May 10th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1564359/v2

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Adapter Based Fine-Tuning of Pre-Trained Multilingual Language Models for Code-Mixed and Code-Switched Text Classification

Himashi Rathnayake[1*†], Janani Sumanapala[1†], Raveesha Rukshani[1†] and Surangika Ranathunga[1]

[1*]Department of Computer Science and Engineering, University of Moratuwa, Katubedda, 10400, Sri Lanka.

*Corresponding author(s). E-mail(s): himashi.17@cse.mrt.ac.lk;
Contributing authors: jananisudeeptha.17@cse.mrt.ac.lk;
raveesha.17@cse.mrt.ac.lk; surangika@cse.mrt.ac.lk;
†These authors contributed equally to this work.

Abstract

Code-mixing and code-switching (CMCS) are frequent features in online conversations. Classification of such text is challenging if one of the languages is low-resourced. Fine-tuning pre-trained multilingual language models (PMLMs) is a promising avenue for code-mixed text classification. In this paper, we explore adapter-based fine-tuning of PMLMs for CMCS text classification. We introduce sequential and parallel stacking of adapters, continuous fine-tuning of adapters, and training adapters without freezing the original model as novel techniques with respect to single-task CMCS text classification. We also present a newly annotated dataset for the classification of Sinhala-English code-mixed and code-switched text data, where Sinhala is a low-resourced language. Our dataset of 10000 user comments has been manually annotated for five classification tasks: sentiment analysis, humor detection, hate speech detection, language identification, and aspects identification, thus making it the first publicly available Sinhala-English CMCS dataset with the largest number of annotation types. In addition to

this dataset, we also carried out experiments on our proposed techniques with Kannada-English and Hindi-English datasets. These experiments confirm that our adapter-based PMLM fine-tuning techniques outperform, or are on par with the basic fine-tuning of PMLM models.

Keywords: code-switching, code-mixing, text classification, Sinhala, XLM-R, adapter

1 Introduction

Switching from one language to another and mixing two languages into a single conversation is a common feature of modern communication systems. This is typically seen on social media platforms like Facebook, Twitter, and Instagram, as well as cross-platform messaging applications like Whatsapp and Viber. This is due to the fact that these platforms have a worldwide user base and people from all over the world bringing their own cultures and backgrounds to the mix (King and Abney, 2013).

The embedding of linguistic components such as words, phrases, and morphemes from one language into an utterance from another language is referred to as code-mixing (Gundapu and Mamidi, 2018). In simple terms, Code-mixing is the practice of borrowing words from one language and adapting them to another without affecting the topic. The juxtaposition of two grammatical systems or subsystems within the same conversation is referred to as code-switching (Gundapu and Mamidi, 2018). An example of Sinhala-English code-mixed and code-switched data is given in Table 1.

Code-mixing and code-switching (CMCS) are crucial for effective advertising, providing customer assistance, and gathering user feedback on products (Agarwal et al, 2017). In these cases, limiting communication to a single language may be challenging if one party is not fluent in that language.

Natural Language Processing (NLP) faces a significant challenge when dealing with CMCS data, as tools developed for a single language might underperform in the context of CMCS data. Handling of code-mixed and code-switched (CMCS) data is difficult because of the lack of annotated datasets, a significant number of unobserved constructions created by combining the syntax and lexicon of two or more languages, and a large number of possible CMCS combinations (Yadav and Chakraborty, 2020). This problem is exacerbated in the context of low-resource languages (LRLs), where datasets are known to be even more scarce and NLP tools are sub-optimal.

Pre-trained multilingual language models (PMLMs) such as mBERT (Libovický et al, 2019) and XLM-R (Conneau et al, 2020) have attained state-of-the-art performance in most of the text classification tasks (Conneau et al, 2020), including code-mixed data classification (Khanuja et al, 2020). In previous work, the PMLMs were mostly used with some basic fine-tuning and hyperparameter optimization (Antoun et al, 2020; Huertas García et al, 2021).

Table 1 Social media language variations of Sinhala-English CMCS data

Type	Example
Purely written in English	Telecom has the best internet facility in Sri Lanka
Purely written in Sinhala	ශ්‍රී ලංකාවේ හොදම අන්තර්ජාල පහසුකම තියෙන්නෙ ටෙලිකොම් එකෙ
Written in a english pronunciation using unicodes (Code-mixed)	ටෙලිකොම් හැස් ද බෙස්ට් ඉන්ටෙර්නෙට් ෆැසිලිටි ඉන් ශ්‍රී ලංකා
Written in a sinhala pronunciation using English characters (Code-mixed)	Sri lankave hodama antharjala paha-sukama thiyenne telecom eke
English characters with Unicode (Code-mixed)	ශ්‍රී ලංකාවේ best ම internet facility එක තියෙන්නෙ ටෙලිකොම් එකෙ
Code-switched	Mata eka labuna, Thank you very much for it.

In this paper, we apply the adapter-based PMLM fine-tuning strategy for CMSC text classification. Adapters are neural modules that add a small number of new parameters to a model (Pfeiffer et al, 2020a). During the training phase, the original model parameters are frozen, and only the newly introduced adapter parameters are fine-tuned. Adapters can be either task-specific or language-specific: language adapters are trained to learn specific language representations, whereas task adapters are trained to learn specific task representations.

Unlike previous research that used adapters for text classification (Pfeiffer et al, 2020b; Wang et al, 2021; Friedman et al, 2021), we are focusing on CMCS data, which is a mix of multiple languages. Thus, we use different combinations and fine-tuning strategies of both language and task adapters. Specifically, we introduce three ways of using adapters in CMCS data classification: (1) sequential and parallel stacking of language adapters followed by a single task adapter, (2) continuous fine-tuning of task adapters on different pre-trained language models (PLMs), and (3) training task adapters without freezing the original model parameters.

Our solutions are validated on three datasets: a Sinhala-English CMCS dataset newly compiled by us, as well as Kannada-English (Hande et al, 2021a) and Hindi-English CMCS datasets which are publicly available. Our Sinhala-English dataset has been annotated for sentiment, humor, hate speech, language ID, and aspect classification considering all CMCS phenomena given in Table 1. Thus this dataset can be used for five classification tasks, which makes this dataset support the largest number of classification tasks, compared to the other datasets for CMCS presented in previous work (Given in Table 2). In this research, we experimented with this dataset for the first four tasks. Further, we experimented with sentiment analysis and hate speech detection tasks in the Kannada-English dataset, as well as humor detection and language identification in the Hindi-English dataset.

Experiment results on the XLM-R PMLM show that our adapter-based fine-tuning strategies either outperform or are on par with basic fine-tuning for

all the datasets we used. Our third approach yielded the best results on average across all experiments. Given that Sinhala and Kannada are heavily under-represented in XLM-R (meaning that a relatively smaller dataset was used for these languages during XLM-R fine-tuning), its performance on Sinhala and Kannada CMCS data is impressive - we believe that this is due to the XLM-R model being able to learn a strong cross-lingual signal from the CMCS data, which can further strengthen up its cross-lingual representation. Thus such PMLMs should be considered a viable option even for LRLs such as Sinhala and Kannada.
This paper makes the following contributions:

1. We presented three adapter-based fine-tuning strategies on PMLMs for CMCS data classification
2. A Sinhala-English CMCS dataset annotated for five different tasks. Compared to the existing datasets, this CMCS dataset has the largest number of task annotations. To the best of our knowledge, this is also the first annotated dataset with Sinhala-English CMCS humor and hate speech classifications.
3. This is the first systematic study of the classification of Sinhala-English CMCS text.

Our dataset, code, and the trained models are publicly available[1].

2 Related Work

2.1 CMCS Text Classification

For classifying CMCS data, Machine Learning approaches such as Logistic Regression, Support Vector Machines, Multinomial Naive Bayes, K-Nearest Neighbors, Decision Trees, and Random Forest have been used by early research (Chakravarthi et al, 2022). Later, Deep Learning techniques such as CNN, LSTM, and BiLSTM (Kamble and Joshi, 2018) have been widely used. Most recently, fine-tuning pre-trained monolingual models such as BERT and multilingual models such as XLM-R and mBERT (Hande et al, 2021b) have been used. PLMs outperformed other Deep Learning and Machine Learning techniques in some studies (Chakravarthi et al, 2020; Aguilar et al, 2020; Khanuja et al, 2020), while they performed poorly in some others (Kazhuparambil and Kaushik, 2020). But according to Kazhuparambil and Kaushik (2020), PLMs can be made the top-performing models for CMCS data classification by optimizing hyper-parameters.

2.2 Annotated Corpora for CMCS Text Classification

CMCS can appear in various forms, including code-switching, inter-sentential, and intra-sentential code-mixing, and texts written in both Latin and native scripts. CMCS text classification corpora have been mainly created with

[1] https://github.com/HimashiRathnayake/CMCS-Text-Classification

respect to Indian languages such as Hindi-English (Bohra et al, 2018), Telugu-English (Gundapu and Mamidi, 2018), Tamil-English, Kannada-English, and Malayalam-English (Chakravarthi et al, 2022), while there are some corpora for other CMCS languages such as Sinhala-English (Smith and Thayasivam, 2019), Spanish-English (Vilares et al, 2016) and Arabic-English (Sabty et al, 2019). Except for the dataset created by Chakravarthi et al (2022), others have removed the text written in native script and considered only a limited type of code-mixing levels. Chakravarthi et al (2022) have considered all types of code-mixing phenomena including inter-sentential and intra-sentential code-mixing, including texts written in both Tamil and Latin scripts. However, most of the studies have annotated their datasets for limited types of classification tasks as given in Table 2.

Table 2 Datasets with Types of Classification Tasks

Paper	Annotated Tasks	Language Pair	CMCS Type
Smith and Thayasivam (2019)	Language identification	Sinhala-English	Latin Script
Bohra et al (2018)	Language identification, Hate speech detection	Hindi-English	Latin Script
Gundapu and Mamidi (2018)	Language identification, POS	Telugu-English	Latin Script
Chakravarthi et al (2022)	Sentiment analysis, Offensive language identification	Tamil-English, Kannada-English, Malayalam-English	All types of code-mixing phenomena
Khandelwal et al (2019)	Humor detection, Language identification	Hindi-English	Latin Script
Swami et al (2018)	Language Identification, Sarcasm Detection	Hindi-English	Latin Script
Chathuranga and Ranathunga (2021)	Sentiment Classification, Aspect Extraction	Sinhala-English	Latin Script

Note: 'Latin Script' indicates that the study has only considered texts written in English alphabetical characters.

In the literature, positive, negative, mixed, neutral, and 'not in intended language' tags have generally been used for CMCS Sentiment Analysis (Chakravarthi et al, 2022), while CMCS humor detection task uses a binary tag scheme to indicate whether a text is humorous or non-humorous (Khandelwal et al, 2019). For CMCS hate speech detection, some studies used a binary tag scheme (Bohra et al, 2018), while others used a tag scheme containing hate, abusive, and not offensive tags (Mathur et al, 2018a). On the other hand, aspects are always domain-specific (Ousidhoum et al, 2019). For CMCS language identification, most research used a tag scheme with tags for

corresponding two languages with few other tags to represent Named Entities, URLs, and punctuation marks (Ansari et al, 2021). But none of them considered separate tags to identify the hybrid mixing[2] of two languages.

2.3 Adapter-based Fine-tuning of PLMs

While fine-tuning PLMs is widely used in NLP, fully fine-tuning those models for a specific task is time-consuming as millions, if not billions, of parameters must be learned. Sharing and storing those models is equally challenging. To address these concerns, "Adapters"(Houlsby et al, 2019) were introduced as a parameter-efficient fine-tuning strategy. Also that fine-tuning strategy achieves performance comparable to fully fine-tuning on most tasks. Adapters are small learned bottleneck layers that are inserted within each layer of a PLM and are updated during fine-tuning while the rest of the model remains fixed. There are two popular adapter architectures: *Houlsby Adapter* (Houlsby et al, 2019) and *Pfeiffer Adapter* (Pfeiffer et al, 2020b). Both types of adapters are implemented on the Transformer architecture (Vaswani et al, 2017). Houlsby adapters have two down and up-projections within each transformer layer, whereas the Pfeiffer adapter has one down- and up-projection. Figure 1 visualizes the transformer layers with these adapter layers in comparison to standard transformer layers. However, Pfeiffer et al (2020b) showed that there is no significant difference in performance between the model architectures.

Adapters can be further classified into two types: task adapters and language adapters. Task adapters (Houlsby et al, 2019) are trained to learn a specific task representation. For example, for a sentiment analysis task, we can train a task adapter with sentiment annotated data to learn the sentiment classification representation. Language adapters (Pfeiffer et al, 2020b) are trained to learn a specific language representation. Language adapters, unlike task adapters, are usually not used alone. They are generally used with a task adapter for cross-lingual transfer learning tasks. When training on a downstream task such as sentiment analysis, the task adapter is stacked on top of the source language adapter for cross-lingual transfer learning. The source language adapter is replaced with the target language adapter and evaluated during inference time for zero-shot cross-lingual transfer capabilities (Pfeiffer et al, 2020b).

Also, there is a possibility to combine multiple adapters using stacking (sequential) and parallel composition blocks[3]. Figures 2 and 3 visualize these stacking and parallel adapter compositions, respectively. To facilitate the cross-lingual transfer of PMLMs, Pfeiffer et al (2020b) used a stacking adapter composition where a task adapter was stacked on top of a language adapter. Wang et al (2021) proposed the Entropy Minimized Ensemble of Adapters (EMEA) method, where they stacked an ensemble of multiple related language adapters with a task adapter. Parallel processing of adapters was first used by

[2]Write the pronunciation of one language using other language, e.g.: hodama - Written in a sinhala pronunciation using English characters, Translation: best

[3]https://docs.adapterhub.ml/adapter_composition.html

Rücklé et al (2021). AdapterHub (Pfeiffer et al, 2020a) is an open-source[4], easy-to-use, and extensible adapter training and sharing framework that supports both of these adapter types as well as different adapter architectures.

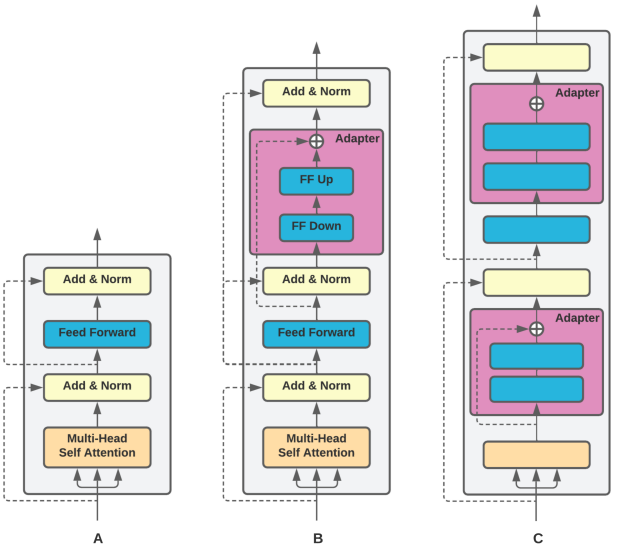

Fig. 1 A - Transformer layer without adapters, B - Transformer layer with a Pfeiffer adapter, C - Tranformer layer with a Houlsby adapter (Pfeiffer et al, 2020a; Rücklé, 2021)

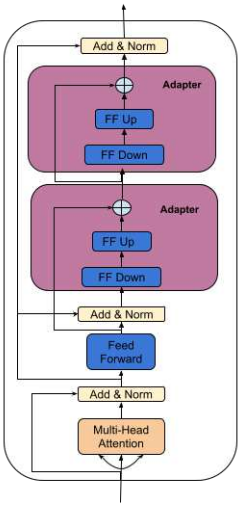

Fig. 2 Stacking adapter composition (Ünal and Dağ, 2022)

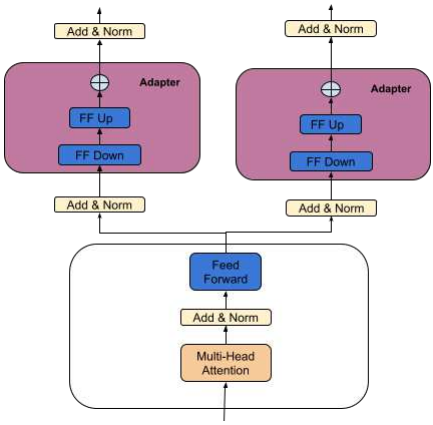

Fig. 3 Parallel adapter composition[a]

[a] https://docs.adapterhub.ml/adapter_composition.html

[4] https://github.com/Adapter-Hub/adapter-transformers

3 Methodology

3.1 Vanilla Fine-tuning of PMLMs

The most successful PMLMs are trained in the Transformer architecture (Vaswani et al, 2017). Encoder-based ones, such as mBERT and XLM-R, are commonly used for classification. These have been trained with a variety of languages and self-supervised objectives such as Masked Language Modeling. As a result, they have to be fine-tuned separately for the downstream task. Vanilla Fine-tuning, also known as *basic fine-tuning*, is the most common method for training them in downstream tasks. During fine-tuning, PMLM weights are copied and fine-tuned with task-specific data to learn the task representation.

3.2 Basic Adapter-based Fine-tuning

For a downstream task, a task adapter introduces new and randomly initialized adapter parameters in addition to the initial parameters of the original model. During fine-tuning, the newly introduced adapter parameters are trained, while keeping the original PLM parameters fixed to learn the specific task representation. Since we are implementing multiple classification tasks, we trained separate task adapters on the PMLM, XLM-R to learn each classification task representation. It was decided to train adapters with both Pfeiffer and Houlsby configurations at the beginning, and continue with the best-performed adapter configuration for further experiments.

3.3 Stacking Language Adapters

As mentioned in Section 2.3, usually language adapters are used for cross-lingual knowledge transfer, where the source language adapter can simply be replaced with the target language adapter to perform a zero-shot transfer to the target language (Pfeiffer et al, 2020b). Therefore, most of the previous works used language adapters to work with one language at a time, so there is no need for multiple language adapters. The exception is Wang et al (2021), who used an ensemble of related language adapters to adapt a PMLM to languages unseen by the model. However, CMCS data contains multiple languages. Thus, as our first technique, we stack language adapters corresponding to the languages that are present in our datasets (which will be referred to as *contributing language* adapters in this paper), followed by a task adapter for the corresponding classification task. In contrast to previous works (Pfeiffer et al (2020b) and Wang et al (2021)), we experimented with stacking multiple contributing language adapters in two different ways:

- Sequential stacking (Figure 4) - We stacked multiple language adapters sequentially, which means one language adapter on top of the other to learn representation specific to the languages in CMCS data. This stack is followed by a task adapter to learn the classification task.

- Parallel stacking (Figure 5) - We used a parallel setup for multiple language adapters. This parallel stack is followed by a task adapter. For that, we used the parallel adapter composition introduced by Rücklé et al (2021). While Rücklé et al (2021) used this technique for parallel multi-task inference, we use the same idea to enable parallel inference of language adapters.

Adapter modules that are trained in about 50 languages to capture language-specific knowledge are available at AdapterHub[5]. Those adapter modules can be applied to any downstream task to capture specific language knowledge as pre-trained language adapters. To continue with these experiments, first, we selected available pre-trained language adapters from AdapterHub for the languages relevant to our experiment. For the languages not available in AdapterHub, we trained new adapters.

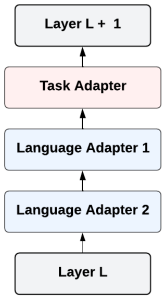

Fig. 4 Sequential language adapter stack followed by a single task adapter

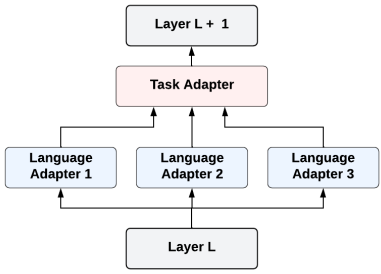

Fig. 5 Parallel language adapter stack followed by a single task adapter

3.4 Continuous Fine-tuning

Usually, a task adapter is trained on a single PLM. But in this experiment, we used CMCS data that contains multiple languages. Therefore we trained a task adapter on multiple PLMs specialized in each language included in the CMCS data. To train the adapters, first, we used two PLMs: PLM specialized in language 1 (Lang 1 PLM eg: SinBERT for the Sinhala language) and PLM specialized in language 2 (Lang 2 PLM eg: BERT for the English Language). Then we also used a PMLM which is specialized in both languages (eg: XLM-R which is pre-trained in both Sinhala and English languages).

In this way, we tried out different training orders of PLMs and with different combinations as shown in Figures 6 and 7.

[5] https://adapterhub.ml/

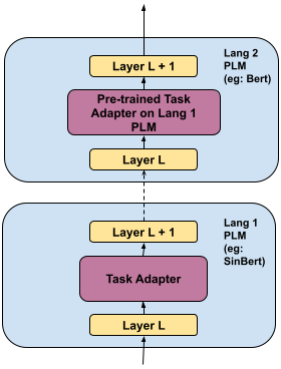

Fig. 6 Continuous Fine-tuning Combination 1 - TA(PLM1) → TA(PLM2)

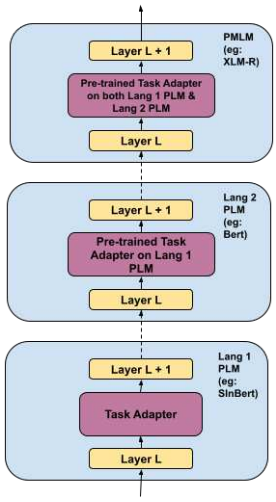

Fig. 7 Continuous Fine-tuning Combination 2 - TA(PLM1) → TA(PLM2) → $TA(PMLM)$

3.5 Training Adapters without Freezing the PMLM (Adapters + PMLM Training)

As mentioned in Section 3.2, the usual practice is to train adapters without training the original model parameters (Freezing the PMLM). Recently, Friedman et al (2021) jointly trained the PLM with the adapters. When the model no longer improves the validation accuracy (which is used to compare model performance), the PLM model parameters are frozen and adapter training is continued. Here, we adopt Friedman et al (2021)'s solution for multi-task adapters into a single task adapter. Here, we trained all of the parameters, which included both the original model parameters and the newly introduced parameters by the adapters. In our case, we used macro-F1 score instead of accuracy to compare model performance as it gives a better interpretation of results in the context of imbalanced datasets as explained in Section 7.2.
We also experimented with combining each of the aforementioned techniques.

4 CMCS Datasets

This section presents the three CMCS datasets used in this study.

4.1 Kannada-English Dataset

This dataset has been annotated at the sentence level for sentiment analysis and hate-speech detection (Hande et al, 2021a). This dataset contains all phenomena of code-mixing and code-switching between Kannada and English. But the tag scheme used in this dataset has more number of tags for each classification compared to our Sinhala-English CMCS dataset. Results of the

Table 3 Tag Schemes of Datasets

Task	Kannada-English	Hindi-English	Sinhala-English
Hate Speech Detection	not-Kannada Not-offensive Offensive-Targeted-Insult-Individual Offensive-Targeted-Insult-Group Offensive-Targeted-Insult-Other	-	Hate-Inducing Abusive Not offensive
Humor Detection	-	0 1	Humorous Non-humorous
Language Identification	-	Hi En Ot	Sinhala English Eng-Sin Sin-Eng NameEntity Mixed Symbol
Sentiment Analysis	Positive Negative not-Kannada Mixed unknown	-	Positive Negative Neutral Conflict

paper (Hande et al, 2021a) have been reported for both single-task learning and Multi-task learning with different PLMs, including PMLMs.

4.2 Hindi-English Dataset

This dataset has been annotated at both sentence-level and word level; humor detection at sentence-level as a binary classification and language identification at word level with three different tags.[6]. This dataset consists of Hindi-English CMCS data that has been written in Latin script.

The tags used in each dataset are listed in Table 3.

4.3 The Annotated Sinhala-English CMCS Corpus

This corpus is newly compiled by us.

4.3.1 Data Collection and pre-Processing

A raw data set that consists of 465314 social media comments was obtained from previous research (Chathuranga and Ranathunga, 2021). First, 15000 comments were randomly selected and Tamil[7] mixed comments were filtered out manually since there were only about 0.02% Tamil mixed comments. After ignoring noisy data instances such as one-character comments and integer comments, 10,000 comments were randomly selected for the annotation.

[6] https://github.com/amsuhane/Humour-Detection-in-English-Hindi-Code-Mixed-Text
[7] The other official language in Sri Lanka.

Since the data has been borrowed from social media comments, they contain identities such as names of persons, names of organizations, and contact numbers. To obey the ethics and rights, an anonymization scheme was designed to perform on the dataset before making it available to the public. The proposed anonymization scheme is given in Table 11.

4.3.2 Data Annotation

The selected 10,000 comments were annotated according to the tagging scheme given in Tables 4, 5, 6, 7, and 8.

Table 4 Sentiment Analysis Tag Scheme

Tag	Definition	Example
Positive	When the commentator is hopeful or confident, and focusing on the positive aspect of a situation rather than the negative aspect.	I enjoyed it a lot.
Negative	When the commentator is pessimistic about a situation or experience is unpleasant or depressing.	slow network
Neutral	When the comment is lacking in sentiment or the commentator does not express a thing as good or bad.	Please send your contact number
Conflict	When the commentator uses the same comment to describe something as good and something as bad.	Okkoma hodai signal nathi eka thamai lokuma gataluwa..

Note: Followed the annotation scheme proposed by Senevirathne et al (2020).

Table 5 Humor Detection Tag Scheme

Tag	Definition	Example
Humorous	When the comment includes anecdotes, irony, fantasy, jokes and insults.	Meka hoyalama mage oluwath ridenawa yku.
Non-humorous	When the comment does not contain any amusement.	Very disappointed about your service

Note: Followed the annotation scheme proposed by Khandelwal et al (2019).

Language Identification was tagged at the word level, and other tasks are tagged at the sentence level. Aspect is multi-label and other tags are single-label.

The dataset was annotated by four annotators. To evaluate the agreement among annotators, Fliess Kappa was calculated for single label tags while the multi-label tagging of aspects was measured by calculating Krippendorff's alpha. The results of those calculated values are shown in Table 9. All the values are above 0.6, interpreting that there exists a substantial inter-annotator agreement between the annotators.

Table 6 Hate Speech Detection Tag Scheme

Tag	Definition	Example
Hate-inducing	Offending, threatening or insulting and individual or group of individuals based on their religion, race, occupation, caste, sexual orientation, gender or membership in a stereotyped community.	Get lost Company A executives
Abusive	Motive to cause hurt by using slurs that are made up of derogatory phrases. (language that is offensive but not hate-inducing)	Thawa enava methana reddak katha karanna kalakanni haththa
Not offensive	No offensive content at all	our router doesn't work

Note: Followed the annotation scheme proposed by Mathur et al (2018b).

Table 7 Aspect Extraction Tag Scheme

Tag	Definition	Example
Billing or Price	Mentioned any fact or issue regarding the prices of services or products	Ape bill awe na ne
Customer Service	Mentioned any fact or issue regarding the customer service	worst customer service
Data	Mentioned any fact or issue regarding the data such as data usage	Masekata hambena data tika balan iddi iwara wenawa.
Network	Mentioned any fact or issue regarding the network, such as network quality or call drops	We don't have stable network in our area....
Package	Mentioned any fact or issue regarding the data or voice mobile packages	There are packages named as unlimited but all packages have speed limits.
Service or Product	Mentioned any fact or issue regarding the services or products	Mage sim eken credit snd krnn bene. Insufficient to snd crdt kyl wtnw
None	None of the above aspects applicable	Congratulations our boys

Note: Followed the annotation schema proposed by Chathuranga and Ranathunga (2021), which uses data particular to the telecommunication domain.

4.3.3 Dataset Statistics

Tag set distribution shown in Figures 8, 9, 11, and 10 indicates the dataset is imbalanced. We used some techniques to handle this imbalance, which is described in Section 7.

The Code-Mixing Index (CMI) proposed by Das and Gambäck (2014) is used to measure the level of mixing between Sinhala and English languages in the created dataset. Our dataset received a value of 11.52 for "CMI-All" (considering all sentences) and a value of 23.77 for "CMI-CS" (only considering code-switched sentences). The calculation for the CMI is given in appendix A.

Table 8 Language Identification Tag Scheme

Tag	Definition	Example
Sinhala	Sinhala words written in Sinhala script	සල්ලි
English	English words written in Latin script	network
Sin-Eng	Sinhala words written in Latin script	salli
Eng-Sin	English words written in Sinhala script	නෙට්වර්ක්
Mixed	Words written in both Sinhala script and Latin script or words that have combined features of two languages	Customersla, ඩේටත්, SIGNALලුත්
NameEntity	Include Name Entities(Names and Numbers)	7812981, Maradana, Srilanka
Symbol	Punctuation marks, emojis and Special characters.	?, #, ...

Note: Used an extended version of the annotation scheme proposed by Smith and Thayasivam (2019). We have introduced separate tags to identify the hybrid mixing of two languages.

Table 9 Inter-Annotator Agreement Evaluation

Evaluation matrix	Classification	Value
Fleiss Kappa	Humor	0.7146
	Hate Speech	0.7492
	Sentiment	0.7898
	Language ID	0.9349
Kripendorff's Alpha	Aspect	0.6441

A higher CMI value indicates a higher level of mixing between the languages, whereas CMI = 0 indicates no code-mixing. A comparison of the CMI of our dataset with other related datasets in the LinCE benchmark (Aguilar et al, 2020) is given in Table 10 and according to the comparison, our dataset has a significant level of mixing.

Table 10 Comparison of CMI in different corpora

Dataset	Language Pair	CMI-All	CMI-CS
Molina et al (2016)	Spanish-English	8.29	21.86
Solorio et al (2014)	Nepali-English	19.85	25.75
Mave et al (2018)	Hindi-English	10.14	22.68
Molina et al (2016)	Modern Standard Arabic-Egyptian Arabic	2.82	23.89
Our dataset	Sinhala-English	11.59	23.77

5 Adapter Training

- Training Task Adapters. A task adapter was trained for each classification task, using the dataset. For example, using the Sinhala-English dataset, we

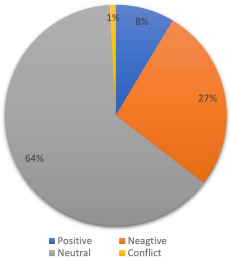

Fig. 8 Sentiment Tag Distribution

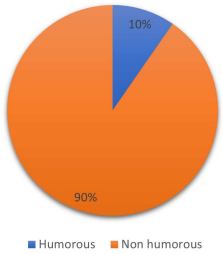

Fig. 9 Humor Tag Distribution

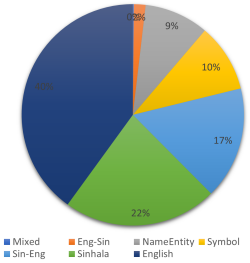

Fig. 10 Language ID Tag Distribution

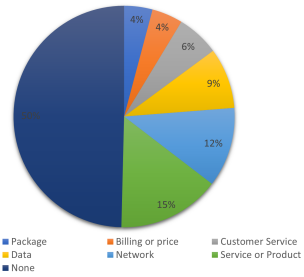

Fig. 11 Aspect Tag Distribution

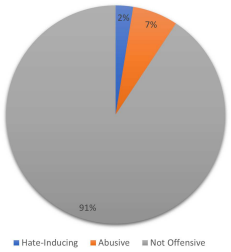

Fig. 12 Hate speech Tag Distribution

Table 11 Anonymization Scheme

Data Type	Anonymization Technique	Before	After
Telephone Numbers	Data Swapping	0112345678	0118567324[1]
Personal Names	Pseudonymization	Sahan Randima	Mark Spencer[1]
Company Names	Pseudonymization	Dialog	Company A[1]

[1]These are just examples and the values shown in the 'After' column have no resemblance to the replacements done in the corpus.

trained task adapters for all four classification tasks: sentiment analysis, humor detection, hate speech detection, and language identification. The same was done for other CMCS datasets.

- Training language adapters

  - We trained two language adapters, one for Sinhala (Si) using Senevirathne et al (2020)'s Sinhala dataset and the other for Sinhala-English CMCS (Si-En) using our newly created Sinhala-English CMCS training dataset. Language adapters were not used in experiments on Kannada-English and Hindi-English CMCS datasets.

- Using pre-trained language adapters
  - We used a pre-trained English language adapter (En) trained for XLM-R which is available in AdapterHub[8]

- Using PLMs for continuous fine-tuning
  - For Sinhala-English, we used English BERT (Kenton and Toutanova, 2019), SinBERT (a pre-trained RoBERTA model for Sinhala) with XLM-R.
  - For Kannada-English and Hindi-English CMCS datasets, we used English BERT (Kenton and Toutanova, 2019), IndicBERT (Kakwani et al, 2020) and XLM-R. IndicBERT has been trained in 12 Indian languages, including Hindi and Kannada. IndicBERT was used because there is no specific PLM for Hindi or Kannada that supports adapters[9].

6 Baseline Implementation

Long Short-Term Memory (LSTM), Bi-directional Long Short-Term Memory (BiLSTM), and Capsule Networks proposed by (Senevirathne et al, 2020) are used as the baseline models for sentiment analysis task because they reported the best results for Sinhala sentiment analysis. The same baselines are used for hate speech detection and humor detection tasks for Sinhala-English CMCS. However, for Sinhala-English (and Hindi-English) language identification, a two-layer bi-directional LSTM model proposed by Toftrup et al (2021) was used as the baseline. The same LSTM and BiLSTM models were applied to Kannada-English and Hindi-English classification tasks.

LSTM: The input layer, the embed layer, the two dropout layers to prevent overfitting, the LSTM layer, and the two solid layers with the Softmax function to predict the relevant label in a given sentence are the basic components of the LSTM model.

BiLSTM: The input layer, the embedding layer, the bidirectional LSTM layer, the time-distributed dense layer, the flatten layer, and the dense layer with the softmax activation function are the basic components of the BiLSTM model.

Two-layer BiLSTM: First, all the characters in the input string are replaced by vector embedding. In each subsequent step, LSTM obtains the character embedding and hidden layer representation. From left to right the output of one character in the LSTM layer is combined with the layer from

[8] https://adapterhub.ml/adapters/ukp/xlm-roberta-base-en-wiki_pfeiffer/

[9] There is only one Hindi BERT model available, but it is an Electra model. Electra does not currently support adapters.

right to left. The concatenated vector is similar to the first LSTM layer but does not share the parameters. After that, the concatenated vectors pass through a single linear layer, producing a distribution over all the supported languages.

Capsule Network: The three main capsule layers, which initiate with a convolutional layer are the main components of this model. Each capsule in the model is instantiated with 16 dimensional parameters. Also, each capsule layer has 16 filters.

XLM-R: XLM-R model is a transformer-based multilingual masked language model that has been pre-trained on text in 100 languages including Sinhala, Kannada, Hindi, and English, and has reported state-of-the-art results for cross-lingual classification (Conneau et al, 2020). In this research, the XLM-R model taken from HuggingFace[10] was initialized with a sequence classification head on top. Then this is fine-tuned with the CMCS dataset for each classification task separately.

7 Evaluation

7.1 Data Preparation

To overcome the issue of dataset imbalance, some oversampling approaches such as Random Oversampling (ROS) and Synthetic Minority Oversampling Technique (SMOTE)(Chawla et al, 2002) with different sampling ratios were explored as a pre-processing step. In this paper, we present the results of the best oversampling technique. Note that these oversampling techniques have been performed only on Sinhala-English, as the other two datasets are free of the data imbalance issue.

7.2 Experiment Setup

Table 12 contains the hyper-parameters of the experiments we customized; other hyperparameters are default values. For LSTM, BiLSTM, and capsule network models, fastText word embedding with 300 dimensions is used as the embedding layer, and categorical cross-entropy is used as the loss function. In those models, results are interpreted using five-fold cross-validation. For interpreting the results of XLM-R and adapters, the experiment was run a few times with different random states and the average was taken. We also used early stopping in these experiments. Precision, Recall, and F1-Score are given in macro averages as they give equal weights for each class and give correct interpretation of results in the context of imbalanced data. All the experiments in this paper are carried out using the Google Colab environment[11].

7.3 Results

The results of experiments carried out are given in Tables 13 and 14.

[10] https://huggingface.co
[11] https://colab.research.google.com/

Table 12 Hyper-parameter details

	Sentiment		Humor		Hate speech		Language ID	
Experiment	Optimizer	LR	Optimizer	LR	Optimizer	LR	Optimizer	LR
LSTM	Adadelta	0.95	Adadelta	0.95	Adamax	0.002		
BiLSTM	Adam	0.001	Adam	0.001	Adamax	0.002		
Capsule Network	Adamax	0.002	Adamax	0.002	Adam	0.001		
2-Layer BiLSTM							Adam	0.1
XLM-R	AdamW	5e-5	AdamW	5e-6	AdamW	5e-5	AdamW	5e-5
Adapters	AdamW	5e-4	AdamW	5e-4	AdamW	5e-4	AdamW	5e-4
Model+Adapters	AdamW	2e-5	AdamW	2e-5	AdamW	2e-5	AdamW	2e-5

Note: LR refers to the learning rate

Table 13 Results of Sinhala-English CMCS Experiments

		Sentiment				Humor				Hate speech				Language ID			
No	Experiment	Ac.	Pr.	Re.	F1	Ac.	Pr.	Re.	F1	Ac.	Pr.	Re.	F1	Ac.	Pr.	Re.	F1
1	LSTM	75	54	45	48	88	66	69	67	91	60	55	57				
2	BiLSTM	67	43	46	45	88	67	69	68	90	61	58	59				
3	Capsule Network	74	51	42	44	87	63	65	64	87	53	59	55				
4	2-Layer BiLSTM													89	73	81	73
5	XLM-R Basic Fine-tuning	78	53	51	54	94	85	74	79	93	72	73	72	97	97	97	97
6	TA train on XLM-R	81	56	54	54	94	83	77	79	92	70	73	71	97	97	97	97
7	Sequential Stacking of LAs with TA	79	54	53	53	94	87	74	79	93	71	75	73	97	97	97	97
8	Parallel Stacking of LAs with TA	81	55	55	54	94	87	75	79	93	72	76	**74**	97	97	97	97
9	TA Continuous Fine-tuning	80	54	55	54	94	85	74	79	93	74	72	73	97	97	97	97
10	Train Model + TAs	80	54	55	**55**	94	83	77	**80**	94	77	71	**74**	97	97	97	97
11	Experiment No 7 + No 10	80	54	56	**55**	95	93	74	**80**	94	73	75	**74**	97	97	97	97
12	Experiment No 8 + No 10	81	55	54	54	95	89	75	**80**	94	73	74	**74**	97	97	97	97
13	Experiment No 9 + No 10	79	54	54	54	95	92	72	79	93	71	75	73	97	97	97	97

Note: In all hate speech experiments, ROS with a sampling ratio of 1:0.25:0.25 was used to over-sample the training dataset. In humor experiments, only in LSTM and Capsule network models ROS with a sampling ratio of 1:0.5 was only used in LSTM and Capsule network models in the humor experiments. All other experiments performed better without oversampling. TAs refer to task adapters whereas LAs refer to language adapters. TA Continuous Fine-tuning training performed in the order of Bert, SinBERT, XLM-R. Stacking LAs composition contains En, Si, Si-En LAs.

7.4 Baseline Results Analysis

According to the results shown in Tables 13 and 14, the fine-tuned XLM-R has outperformed the other Deep Learning models used in each task. When it comes to oversampling techniques, only the ROS technique improved the results; SMOTE did not. Even ROS did not perform well across all tasks and models. It improves the results of the Sinhala-English hate speech task for each model and it only improves the results of the Sinhala-English humor detection task for LSTM and capsule network models.

7.5 Adapter based Fine-Tuning Results

According to the results shown in Tables 13 and 14, simply training task adapters result in a performance drop when compared to basic XLM-R fine-tuning in some cases while performing on par in others. However, both the

Table 14 Results of Experiments for Kannada-English and Hindi-English datasets

		Sentiment				Humor				Hate speech				Language ID			
No	Experiment	Ac.	Pr.	Re.	F1	Ac.	Pr.	Re.	F1	Ac.	Pr.	Re.	F1	Ac.	Pr.	Re.	F1
1	LSTM	61	38	36	36	57	57	57	57	53	45	41	41				
2	BiLSTM	50	46	44	44	57	57	57	57	66	41	39	39				
3	2-Layer BiLSTM													95	91	93	91
4	XLM-R Basic Fine-tuning	58	49	47	47	70	70	70	70	68	46	44	**44**	97	97	95	96
5	TA train on XLM-R	57	41	41	41	61	63	62	61	59	44	43	42	98	97	96	96
6	TA Continuous Fine-tuning	57	43	43	42	61	63	61	60	66	36	31	29	98	97	96	96
7	Train Model + TAs	65	53	53	**51**	72	73	72	**72**	73	58	46	**44**	99	98	99	**98**

Note: The results of Sentiment and Hate speech are reported for Kannada-English whereas the results of Humor and Language ID are reported for the Hindi-English dataset. TAs refers to task adapters. TA Continuous Fine-tuning was performed in the order of Bert, IndicBert, XLM-R.

Pfeiffer and Houlsby task adapters were unable to outperform the XLM-R model's basic fine-tuning results. Pfeiffer et al (2020a) and Pfeiffer et al (2021) made a similar observation. Furthermore, there was no significant difference in performance between the Pfeiffer and Houlsby configurations. Thus, in this paper, we only reported the results of the XLM-R model with Pfeiffer adapters, which we used for subsequent adapter experiments.

Stacking language adapters with task adapters outperformed basic adapter-based fine-tuning in most of the experiments. We experimented with the sequential and parallel stacking of various language adapter combinations, such as stacking a single language adapter alone, stacking two language adapters, and stacking all three language adapters in different orders with the task adapter. Among them, En+Si+Si-En performed best for Sinhala-English CMCS. That may be because those language adapters were able to add language knowledge of Sinhala, English, and Sinhala-English CMCS to the model. Furthermore, parallel stacking of language adapters performed better than sequential stacking. Despite outperforming basic adapter-based fine-tuning, this technique outperforms XLM-R only in the hate speech task. Therefore we did not apply this technique to Kannada-English and Hindi-English CMCS classifications.

Our continuous fine-tuning approach improved the results of some tasks while providing on par performance to XLM-R in others. For Sinhala-English CMCS classification, we obtained the best results for continuous fine-tuning in the order of BERT, SinBERT, and XLM-R. This technique, however, did not perform well with Hindi-English and Kannada-English datasets. That could be because we could not find a PLM trained specifically for Hindi or Kannada with an MLM objective.

Adapter fine-tuning without freezing the model gave us the best results in all four tasks for all Sinhala-English, Kannada-English, and Hindi-English classifications. But freezing the model and further training the adapters only did not improve ours in contrast to Friedman et al (2021).

Finally, combining the aforementioned techniques for Sinhala-English CMCS data did not further improve the results. Therefore we did not test them with Kannada-English or Hindi-English data.

7.6 Misclassified Sinhala-English CMCS Text

We carried out an error analysis on the Sinhala-English dataset. In the sentiment analysis task, each sentence is classified into four different classes. The sentences that are intended to be classified into the 'conflict' class contain both positive and negative sentiments in a single sentence. Moreover, there were only a few data samples for this class. Also, some sentences carry a negative or positive sentiment even though those sentences do not explicitly contain positive or negative polarity words that the models learn from. This may have made it difficult for the model to figure out the positivity or the negativity of a sentence. Therefore even the best performing model predicted these types of sentences inaccurately. Some examples are shown in Table 15.

Table 15 Sentiment Analysis Misclassified Sentences

Sentence	English Translation	Label	Prediction
E unata 078ta 3g tiyanawa 072ta apita 3g nhe..	But 078 have 3g, for our 072 doesn't have 3g	Conflict	Negative
She finds cooking is a meditation n always entertain me with her own recepies.	She finds cooking is a meditation and always entertains me with her own recipes.	Positive	Negative

Detecting humor and hate speech is a challenging task since it requires a large amount of external knowledge, such as language and common sense insights. With the small number of samples for the positive classes such as Humorous, Abusive, and Hate-Inducing (Even the random oversampling performed, only duplicates the already existing examples) the dataset covers only a small amount of those insights. Therefore, even the best performing model, XLM-R, was unable to detect humor and hate speech correctly in some sentences as seen in Tables 16 and 17.

Table 16 Humor Detection Misclassified Sentences

Sentence	English Translation	Label	Prediction
Mama masa 3 idala illane August iwara wanna kalin dennam kiuwa thama na.	I've been asking for three months and they said they'd give it before the end of August, but not yet.	Non-humorous	Humorous
ira newath hada newath kellonam nannema na	Even though sun doesn't rise, moon doesn't rise, girls will not change.	Humorous	Non-humorous

There are many ambiguous words when it comes to Sinhala-English CMCS. Some of the words are present in both languages, however the meaning of each word varies greatly between the two. In particular, when typing Sinhala,

Table 17 Hate Speech Detection Misclassified Sentences

Sentence	English Translation	Label	Prediction
slt ekata siya parakata wada cl kala et hadanne ne...	Called SLT more than a hundred times, but could not fix it.	Not offensive	Hate-Inducing
owa dila monawa karannada	What to do with them?	Not offensive	Abusive

people tend to use the characters “k” and “i” at the ends of the numbers. This results in misclassified words, as shown in Table 18.

Table 18 Language Identification Misclassified Words

Word	Meaning	Label	Prediction
one	Even though this is an English term for number 1, here it is used in Sinhala to refer ’want’	Sin-Eng	English
4k	Even though this refers to 4000 in practice, here it refers to ’exactly 4’ in Sinhala	Sin-Eng	NameEntity

8 Conclusion and Future Work

In this research, we experimented with the recently introduced light-weight fine-tuning strategy i.e. adapters in different ways with PLMs. We introduced a continuous fine-tuning strategy for adapters and we experimented with stacking language adapter compositions with single task adapters and training adapters without freezing the model. Our results showed that XLM-R basic fine-tuning outperformed the other deep learning techniques in CMCS data classification. Proposed adapter-based fine-tuning strategies further improve the results of XLM-R basic fine-tuning, and training adapters without freezing the model produced the best results for CMCS data. A comprehensive dataset annotated with the sentiment, humor, hate speech, aspect, and language id on Sinhala-English CMCS data was introduced. We intend to apply different improvements to XLM-R and adapters and develop a multi-task model for classifying all the tasks for a given dataset in the future.

References

Agarwal P, Sharma A, Grover J, et al (2017) I may talk in english but gaali toh hindi mein hi denge: A study of english-hindi code-switching and swearing pattern on social networks. In: 2017 9th International Conference on Communication Systems and Networks (COMSNETS), IEEE, pp 554–557

Aguilar G, Kar S, Solorio T (2020) Lince: A centralized benchmark for linguistic code-switching evaluation. In: Proceedings of the 12th Language Resources and Evaluation Conference, pp 1803–1813

Ansari MZ, Beg M, Ahmad T, et al (2021) Language identification of hindi-english tweets using code-mixed bert. arXiv preprint arXiv:210701202

Antoun W, Baly F, Achour R, et al (2020) State of the art models for fake news detection tasks. In: 2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT), IEEE, pp 519–524

Bohra A, Vijay D, Singh V, et al (2018) A dataset of hindi-english code-mixed social media text for hate speech detection. NAACL HLT 2018 p 36

Chakravarthi BR, Jose N, Suryawanshi S, et al (2020) A sentiment analysis dataset for code-mixed malayalam-english. In: Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL), pp 177–184

Chakravarthi BR, Priyadharshini R, Muralidaran V, et al (2022) Dravidiancodemix: Sentiment analysis and offensive language identification dataset for dravidian languages in code-mixed text. Language Resources and Evaluation pp 1–42

Chathuranga S, Ranathunga S (2021) Classification of code-mixed text using capsule networks. In: Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021), pp 256–263

Chawla NV, Bowyer KW, Hall LO, et al (2002) Smote: synthetic minority oversampling technique. Journal of artificial intelligence research 16:321–357

Conneau A, Khandelwal K, Goyal N, et al (2020) Unsupervised cross-lingual representation learning at scale. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp 8440–8451

Das A, Gambäck B (2014) Identifying languages at the word level in code-mixed indian social media text. In: Proceedings of the 11th International Conference on Natural Language Processing, pp 378–387

Friedman D, Dodge B, Chen D (2021) Single-dataset experts for multi-dataset question answering. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp 6128–6137

Gundapu S, Mamidi R (2018) Word level language identification in english telugu code mixed data. In: Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation

Hande A, Hegde SU, Priyadharshini R, et al (2021a) Benchmarking multi-task learning for sentiment analysis and offensive language identification in under-resourced dravidian languages. arXiv preprint arXiv:210803867

Hande A, Puranik K, Yasaswini K, et al (2021b) Offensive language identification in low-resourced code-mixed dravidian languages using pseudo-labeling. arXiv preprint arXiv:210812177

Houlsby N, Giurgiu A, Jastrzebski S, et al (2019) Parameter-efficient transfer learning for nlp. In: International Conference on Machine Learning, PMLR, pp 2790–2799

Huertas García Á, et al (2021) Automatic information search for countering covid-19 misinformation through semantic similarity. Master's thesis

Kakwani D, Kunchukuttan A, Golla S, et al (2020) Indicnlpsuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for indian languages. In: Findings of the Association for Computational Linguistics: EMNLP 2020, pp 4948–4961

Kamble S, Joshi A (2018) Hate speech detection from code-mixed hindi-english tweets using deep learning models. arXiv preprint arXiv:181105145

Kazhuparambil S, Kaushik A (2020) Cooking is all about people: Comment classification on cookery channels using bert and classification models (malayalam-english mix-code). arXiv preprint arXiv:200704249

Kenton JDMWC, Toutanova LK (2019) Bert: Pre-training of deep bidirectional transformers for language understanding. Universal Language Model Fine-tuning for Text Classification p 278

Khandelwal A, Swami S, Akthar SS, et al (2019) Humor detection in english-hindi code-mixed social media content: Corpus and baseline system. In: 11th International Conference on Language Resources and Evaluation, LREC 2018, European Language Resources Association (ELRA), pp 1203–1207

Khanuja S, Dandapat S, Srinivasan A, et al (2020) Gluecos: An evaluation benchmark for code-switched nlp. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp 3575–3585

King B, Abney S (2013) 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, pp 1110–1119

Libovickỳ J, Rosa R, Fraser A (2019) How language-neutral is multilingual bert? arXiv preprint arXiv:191103310

Mathur P, Sawhney R, Ayyar M, et al (2018a) Did you offend me? classification of offensive tweets in hinglish language. In: Proceedings of the 2nd Workshop on Abusive Language Online (ALW2), pp 138–148

Mathur P, Shah RR, Sawhney R, et al (2018b) Detecting offensive tweets in hindi-english code-switched language. ACL 2018 p 18

Mave D, Maharjan S, Solorio T (2018) Language identification and analysis of code-switched social media text. ACL 2018 p 51

Molina G, Rey-Villamizar N, Solorio T, et al (2016) Overview for the second shared task on language identification in code-switched data. EMNLP 2016 p 40

Ousidhoum N, Lin Z, Zhang H, et al (2019) Multilingual and multi-aspect hate speech analysis. In: EMNLP/IJCNLP (1)

Pfeiffer J, Rücklé A, Poth C, et al (2020a) Adapterhub: A framework for adapting transformers. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp 46–54

Pfeiffer J, Vulić I, Gurevych I, et al (2020b) Mad-x: An adapter-based framework for multi-task cross-lingual transfer. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp 7654–7673

Pfeiffer J, Kamath A, Rücklé A, et al (2021) Adapterfusion: Non-destructive task composition for transfer learning. In: Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, pp 487–503

Rücklé A (2021) Representation learning and learning from limited labeled data for community question answering

Rücklé A, Geigle G, Glockner M, et al (2021) Adapterdrop: On the efficiency of adapters in transformers. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp 7930–7946

Sabty C, Elmahdy M, Abdennadher S (2019) Named entity recognition on arabic-english code-mixed data. In: 2019 IEEE 13th International Conference on Semantic Computing (ICSC), IEEE Computer Society, pp 93–97

Senevirathne L, Demotte P, Karunanayake B, et al (2020) Sentiment analysis for sinhala language using deep learning techniques. arXiv preprint arXiv:201107280

Smith I, Thayasivam U (2019) Language detection in sinhala-english code-mixed data. In: 2019 International Conference on Asian Language Processing (IALP), IEEE, pp 228–233

Solorio T, Blair E, Maharjan S, et al (2014) Overview for the first shared task on language identification in code-switched data. EMNLP 2014 p 62

Swami S, Khandelwal A, Singh V, et al (2018) A corpus of english-hindi code-mixed tweets for sarcasm detection. Scanning Electron Microsc Meet at

Toftrup M, Sørensen SA, Ciosici MR, et al (2021) A reproduction of apple's bi-directional lstm models for language identification in short strings. In: Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop, pp 36–42

Ünal U, Dağ H (2022) Anomalyadapters: Parameter-efficient multi-anomaly task detection. IEEE Access

Vaswani A, Shazeer N, Parmar N, et al (2017) Attention is all you need. Advances in neural information processing systems 30

Vilares D, Alonso MA, Gómez-Rodríguez C (2016) En-es-cs: An english-spanish code-switching twitter corpus for multilingual sentiment analysis. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), pp 4149–4153

Wang X, Tsvetkov Y, Ruder S, et al (2021) Efficient test time adapter ensembling for low-resource language varieties. In: Findings of the Association for Computational Linguistics: EMNLP 2021, pp 730–737

Yadav S, Chakraborty T (2020) Unsupervised sentiment analysis for code-mixed data. arXiv preprint arXiv:200111384

Declarations

Ethical Approval and Consent to participate

Ethical approval is not applicable. Data annotators were informed about the task prior to the work, and were paid according to the institution approved rates.

Consent for publication

We would like to confirm that this work is original and that has not been published elsewhere, nor is it currently under consideration for publication elsewhere. All the authors give their consent to publish the manuscript.

Availability of data and materials

The dataset, code, and the trained models used in this paper are publicly available[12].

Competing interests

The four authors of this paper have no conflict of interest to disclose.

Funding

Senate Research Committee (SRC) Grant of University of Moratuwa, Sri Lanka

Authors' contributions

Surangika Ranathunga - research idea, and supervision. Manuscript reviewing. Himashi Rathnayake, Janani Sumanapala and Raveesha Rukshani - literature review, system implementation and evaluation. Prepared the initial manuscript.

Acknowledgments

The authors would like to acknowledge the SRC funding.

Authors' information

Affiliations

Department of Computer Science and Engineering, University of Moratuwa, Katubedda, 10400, Sri Lanka
Himashi Rathnayake1, Janani Sumanapala, Raveesha Rukshani and Surangika Ranathunga

Corresponding author

Correspondence to Himashi Rathnayake

Appendix A CMI Calculation

The following equation can be used to calculate CMI at the utterance level Das and Gambäck (2014).

$$CMI = \begin{cases} 100 \times [1 - \frac{max(w_i)}{n-u}] & : n > u \\ 0 & : n = u \end{cases}$$

In this equation, w_i is the highest number of words present from any language in CMCS data, n is the total number of tokens and u is the number of

[12] https://github.com/HimashiRathnayake/CMCS-Text-Classification

tokens with language-independent tags. In our Sinhala-English corpus, we considered Mixed, Symbol, and NamedEntity tags as the language-independent tags.

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- AdapterBasedFineTuningofPreTrainedMultilingualLanguageModelsforCodeMixedandCodeSwitchedTextClassification.zip