Journal of Data, Information and Management (2022) 4:137–165
https://doi.org/10.1007/s42488-022-00070-w

ORIGINAL ARTICLE

Multi-task learning in under-resourced Dravidian languages

Adeep Hande[1] · Siddhanth U. Hegde[2] · Bharathi Raja Chakravarthi[3]

Received: 20 January 2022 / Accepted: 13 May 2022 / Published online: 13 June 2022

Abstract
It is challenging to obtain extensive annotated data for under-resourced languages, so we investigate whether it is beneficial to train models using multi-task learning. Sentiment analysis and offensive language identification share similar discourse properties. The selection of these tasks is motivated by the lack of large labelled data for user-generated code-mixed datasets. This paper works with code-mixed YouTube comments for Tamil, Malayalam, and Kannada languages. Our framework is applicable to other sequence classification problems irrespective to the size of the datasets. Experiments show that our multi-task learning model can achieve high results compared to single-task learning while reducing the time and space constraints required to train the models on individual tasks. Analysis of fine-tuned models indicates the preference of multi-task learning over single task learning resulting in a higher weighted F1 score on all three languages. We apply two multi-task learning approaches to three Dravidian languages, Kannada, Malayalam, and Tamil. Maximum scores on Kannada and Malayalam were achieved by mBERT subjected to cross entropy loss and with an approach of hard parameter sharing. Best scores on Tamil was achieved by DistilBERT subjected to cross entropy loss with soft parameter sharing as the architecture type. For the tasks of sentiment analysis and offensive language identification, the best performing model scored a weighted F1-Score of (66.8%, 90.5%), (59%, 70%) and (62.1%,75.3%) for Kannada, Malayalam and Tamil on sentiment analysis and offensive language identification respectively.

Keywords Code-mixing · Sentiment analysis · Offensive language identification · Multi-task learning · Under-resourced languages · Dravidian languages

1 Introduction

The internet is today an essential for accessing information and communicating with others. The easier access of the internet has resulted in majority of the population using social media to amplify their communication and connections throughout the world. Social media platforms such as Facebook, YouTube, and Twitter have paved the way to express any sentiment about the content posted by its users in any language(Severyn et al. 2014; Clarke and Grieve 2017; Tian et al. 2017). These texts are informal as they are written in a spoken tone that does not follow strict grammatical rules. Understanding social media content is lately attracting much attention from the Natural Language Processing (NLP) community (Barman et al. 2014), owing to the growing amount of data and active users. Sentiment Analysis (SA) refers to the method to extract subjectivity and polarity from text (Pang and Lee 2008). The lack of moderation on social media has resulted in the persistent use of offensive language towards other users on their platforms. Offensive language identification (OLI) is the task of identifying whether a comment contains offensive language or not. Traditionally, the classification of SA or OLI is usually attempted in a monolingual and single task. Traditional approaches to this text classification problems are pretty helpful for high resourced languages. However, traditional

✉ Bharathi Raja Chakravarthi
bharathi.raja@insight-centre.org

Adeep Hande
adeeph18c@iiitt.ac.in

Siddhanth U. Hegde
siddhanthhegde227@gmail.com

[1] Indian Institute of Information Technology Tiruchirappalli, Tiruchirappalli, Tamil Nadu, India

[2] University Visvesvaraya College of Engineering, Bangalore University, Bangalore, India

[3] Insight SFI Research Centre for Data Analytics, National University of Ireland Galway, Galway, Ireland

approaches fail for languages with limited resources, and they also fail on code-mixed text (Chakravarthi et al. 2020).

Multi-Task Learning (MTL) is a practical approach for utilising shared characteristics of tasks to improve system performances (Caruana 1997). In MTL, the objective is to utilise learning multiple tasks simultaneously to improve the performance of the system (Martínez Alonso and Plank 2017). Since SA and OLI are both essentially sequence classification tasks, this inspired us to do MTL. To utilise MTL on Dravidian languages, we experimented with several recent pretrained transformer-based natural language models on Tamil (ISO 639-3: tam), Malayalam (ISO 639-3: mal), and Kannada (ISO 639-3:kan).

Kannada and Malayalam are among the Dravidian languages predominantly spoken in South India and are also official languages in the states of Karnataka and Kerala, respectively (Reddy and Sharoff 2011). The Tamil language has official status in Tamil Nadu of India and countries like Sri Lanka, Singapore, Malaysia and other parts of the world. Dravidian languages are morphologically rich; along with code-mixing, it becomes even more challenging to process these languages (Bhat 2012), and they are under-resourced (Prabhu et al. 2020).

We propose a method to perform MTL to address two main challenges arising when creating a system for user-generated comments from social media. The challenges are:

1. **Code-mixing:** In multilingual countries like India, Sri Lanka and Singapore, the speakers are likely to be polyglottic and often switch between multiple languages. This phenomenon is called code-mixing and these code-mixed texts are even written in non-native scripts (Das and Gambäck 2014; Bali et al. 2014; Chakravarthi et al. 2020). Code-mixing can be referred to as a blend of two or more languages in a single sentence or conversation, which is prevalent in the social media platforms such as Facebook and YouTube.
2. **Scarcity of Data:** Although there is an enormous number of speakers for Tamil, Kannada, and Malayalam, these languages are extensively considered as under-resourced languages. One of the main reasons is the lack of user-generated content extracted from social media applications. One of the ways to tackle the problem is to annotate the data on several tasks.

We address 1 by using pretrained multilingual natural language models and 2 by using the MTL approach.

The rest of the paper is organised as follows. Section 2 shows previous work on SA, OLI , and MTL in NLP. Section 3 consists of a detailed description of the datasets for our purposes. Section 4 talks about the proposed models for single-task models and their loss functions. Popular MTL frameworks are described in Section 4.2. We describe the experimental results and analysis in Section 5.4 and conclude our work in Section 6.

2 Related Work

In this section, we briefly review previous relevant work related to (i) Sentiment Analysis, (ii) Offensive language identification and finally,(iii) MTL in NLP.

2.1 Sentiment Analysis

Sentiment analysis is one of the leading research domains devoted to analyse people's sentiments and opinions on any given entity. Due to its broader applications, there has been a plethora of research performed in several languages. However, the same is not true for the Dravidian languages. As stated earlier, there is a significant lack of data for conducting experiments on code-mixed data in the Dravidian Languages. SA is one of the most prominent downstream tasks of NLP as it is essential for obtaining people's opinion, which has several business applications in the E-commerce market.

A dataset was created as a part of a shared task on SA of Indian Languages (SAIL), which consisted of around 1,663 code-mixed tweets extracted from Twitter in Tamil (Patra et al. 2015), where SentiWordNet outperformed all of the other systems(Phani et al. 2016)(Das and Bandyopadhyay 2010). There has been a plethora of research performed on several downstream tasks in other languages, primarily due to the abundance of user-generated data in social media, which has developed an interest in people's opinions and emotions with respect to a specific target. Existing research is relatively low due to the lack of data in code-mixing. SA is one of the downstream tasks that are performed on any natural language model. The largely available crowd-sourced data on social media applications such as Twitter and YouTube have resulted in developing several code-mixed datasets for the task.

To our knowledge, very few Kannada-English code-mixed datasets exist on SA. A Kannada-English code-mixed dataset for the emotion prediction was created (Appidi et al. 2020). A probabilistic approach was employed to classify Parts of Speech (POS) tags (Sowmya Lakshmi and Shambhavi 2017). Several research pursuits have been worked upon SA in Tamil (Se et al. 2016; Thilagavathi and Krishnakumari 2016). A recursive neural network approach was opted to improve the accuracy of texts in Tamil (Padmamala and Prema 2017). A dynamic mode decomposition (DMD) method with random mapping was developed for SA on the SAIL 2015 dataset (Kumar et al. 2020). A Lexicon based approach was employed along with several feature representation approaches for analysing the sentiments on Tamil

texts (Thavareesan and Mahesan 2019). When it comes to Malayalam, several supervised Machine Learning and rule-based approaches were used to analyse the sentiments of the Malayalam movie reviews (Nair et al. 2015; 2014; Soumya and Pramod 2020). A fuzzy logic-based hybrid approach was also used to analyse the movie reviews in Malayalam (Anagha et al. 2015).

2.2 Offensive Language Identification

A surge in the popularity of social media platforms has resulted in the rise of trolling, aggression, hostile, and abusive language, which is a concerning issue pertaining to the positive/ negative impacts a message can imply on an individual or groups of people (Tontodimamma et al. 2021; del Arco et al. 2021). This issue has led several researchers to work on identifying offensive language/posts from social media for moderating content on social media platforms to promote positivity (Chakravarthi 2020). Offensive Language can be defined as any text entailing certain forms of unacceptable language, which may include insults, threats, or bad words (del Arco et al. 2021). In comparison, hate speech seems indistinguishable to offensive language. The former aims to detect 'abusive' words, which are considered a type of degradation (Nobata et al. 2016; Djuric et al. 2015).

There are several ways to detect offensive language. A supervised learning technique was used (Dadvar et al. 2013), which was based on three decisive factors: content based, cyberbullying based, and user based features to tackle cyberbullying. A multi-level classification system was developed which extracts features at different conceptual levels and applies pattern recognition for detecting flames (rants, taunts, and squalid phases) in sentences (Razavi et al. 2010). It can also be detected by ferreting out offensive and toxic spans in the texts. A toxic span detecting system was developed by leveraging token classification, and span prediction techniques that are based on BERT (Chhablani et al. 2021). MUDES, a multilingual system to detect offensive spans in texts was developed (Ranasinghe and Zampieri 2021). Several systems were developed to identify offensive language as a part of shared tasks conducted to stimulate research in this domain for Arabic, Danish, English, Greek, and Turkish (Zampieri et al. 2019; Zampieri et al. 2020). Consequentially, several NLP researchers have worked on developing systems to detect hate speech on social media (Kumar et al. 2020; Zampieri et al. 2019). However, most of the work done on OLI is language-specific, focusing on monolingual users in lieu of multilingual users, which entails code-mixed text on its social media users (Bali et al. 2014).

For code-mixed sentences, certain researchers analysed the existing state-of-the-art hate speech detection on Hindi-English (Rani et al. 2020), while other researchers compared the existing pretrained embeddings for CNN networks (Banerjee et al. 2020). When it comes to OLI, several systems were developed as a part of shared task for OLI in Dravidian Languages (Chakravarthi et al. 2021; Ghanghor et al. 2021; Yasaswini et al. 2021) and in Indo-European languages (Mandl et al. 2020).

2.3 Multi-Task Learning

The main objective of the multi-task learning (MTL) model is to improve the learning of a model for a given task by utilising the knowledge encompassed in other tasks, where all or a subset of tasks are related (Zhang and Yang 2018). The essence of MTL is that solving many tasks together provides a shared inductive bias that leads to more robust and generalisable system (Changpinyo et al. 2018). It has long been studied in the domain of machine learning. It has applications on neural networks in the NLP domain (Caruana 1997). However, to the best of our knowledge, MTL models have not been developed for the Dravidian languages yet.

There is a considerable variety and unique model designs in the field of computer vision since the requirements in the domain of computer vision is vast. Tasks like image classification, object localisation, object segmentation (Mou and Zhu 2018) and object tracking are widespread. Additionally, multi-tasking in videos for real-time applications (Ouyang et al. 2019), play a vital role in day to day life. Multi-task models have also been used for medical images (Zhai et al. 2020).

MTL models are usually designed with shared encoders which can be customised for the preferred tasks. A multi-scale approach was used to combine LSTM and CNN for better results as it adds up the benefits of both CNN (nearby sentiments) and LSTM (sentiments which are further away) (Jin et al. 2020). Attention-based mechanisms can also be used in the encoder, such as multi-layer bidirectional transformer encoder and knowledge encoder which injects knowledge into the language expressions (Zhang et al. 2020). Not only LSTM, even Gated Recurrent Units (GRU) can be used as a part of multi-task model based on specific tasks; although GRUs have much simpler architecture than the LSTMs, they have the capabilities to forecast futuristic values based on the previous values (Zhang et al. 2020). Such model preferences can be opted to the domain of NLP in order to make the model lighter and faster. The most recent transformation includes using multi-tasks over semi-supervised learning (Li et al. 2020), by stacking recurrent neural networks and utilising a sliding window algorithm the sentiments are transferred on to the next item. An empirical study on MTL models for varieties of biomedical and clinical NLP tasks on BERT was proposed (Peng et al. 2020).

MTL models for the domain of NLP are less in number. The ideas which were applied to other domains such as Computer Vision, Time Series, semi-supervised learning

Table 1 Classwise distribution of the datasets for Kannada, Malayalam, and Tamil

Sentiment analysis			Offensive language identification	
SL.No	Class	Distribution	Class	Distribution
Kannada				
1	Positive	3,291	Not Offensive	4,121
2	Negative	1,481	Offensive Untargeted	274
3	Mixed Feelings	678	Offensive Targeted Individual	624
4	Neutral	820	Offensive Targeted Group	411
5	Other Language	1,003	Offensive Targetd Others	145
6	–	–	Other Language	1698
	Total	7,273	Total	7,273
Tamil				
1	Positive	24,501	Not Offensive	31,366
2	Negative	5,190	Offensive Untargeted	3,594
3	Mixed Feelings	4,852	Offensive Targeted Individual	2,928
4	Neutral	6,748	Offensive Targeted Group	3,110
5	Other Language	2,058	Offensive Targeted Others	582
6	–	–	Other Language	1,769
	Total	43,349	Total	43,349
Malayalam				
1	Positive	5,565	Not Ofensive	11,357
2	Negative	1,394	Offensive Untargeted	171
3	Mixed Feelings	794	Offensive Targeted Individual	179
4	Neutral	4,063	Offensive Targeted Group	113
5	Other Language	955	Other Language	951
	Total	12,771	Total	12,771

can be acquired, and models could be improved in NLP. As such models are rarely seen, this paper introduces MTL for SA and OLI incorporated from other domains of deep learning.

3 Dataset

We make use of the multilingual dataset, DravidianCodeMix[1] (Chakravarthi et al. 2021), consisting of over 60,000 manually annotated YouTube comments. The dataset comprises of sentences from 3 code-mixed Dravidian languages, Kannada (Hande et al. 2020), Malayalam (Chakravarthi et al. 2020), and Tamil (Chakravarthi et al. 2020; Chakravarthi et al. 2021). Each comment is annotated for the tasks of Sentiment Analysis and offensive language identification. The code-mixed Kannada dataset consists of 7,273 comments, while the corresponding Malayalam and Tamil code-mixed datasets consist of 12,711 and 43,349 comments, respectively. After removing repetitive sentences, the class-wise distribution of the datasets are specified in Table 1 which are to be split into train, validation, and test sets.

The class labels in the dataset are as follows: **Sentiment Analysis:**

- **Positive state:** Comment contains an explicit or implicit clue in the text suggesting that the speaker is in a positive state.
- **Negative state:** Comment contains an explicit or implicit clue in the text suggesting that the speaker is in a negative state.
- **Mixed feelings:** Comment contains an explicit or implicit clue in both positive and negative feeling.
- **Neutral state:** Comment does not contain an explicit or implicit indicator of the speaker's emotional state.
- **Not in intended language:** For Kannada if the sentence does not contain Kannada written in Kannada script or Latin script then it is not Kannada.

Offensive language identification:

- **Not Offensive**: Comment does not contain offence or profanity.
- **Offensive Untargeted**: Comment contain offence or profanity without any target. These are comments which contain unacceptable language that does not target anyone.

[1] https://github.com/bharathichezhiyan/DravidianCodeMix-Dataset

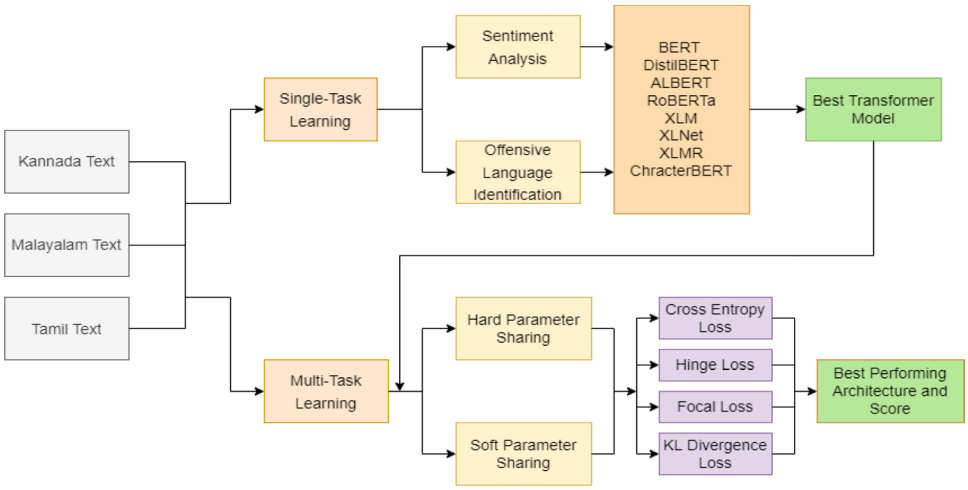


Fig. 1 Outline of the methodology

- **Offensive Targeted Individual**: Comment contains offence or profanity which targets the individual.
- **Offensive Targeted Group**: Comment contains offence or profanity which targets the group.
- **Offensive Targeted Other**: Comment contains offence or profanity which does not belong to any of the previous two categories(e.g., a situation, an issue, an organization or an event).
- **Not in indented language**: Comment not in the Kannada language.

The overall class types are similar in all languages The code-mixed datasets of Kannada and Tamil consist of six classes in OLI , while Malayalam consists of five classes. There is an absence of Offensive Language Others (OTO) class in the Malayalam dataset.

4 Methodology

We explore the suitability of several NLP models on the task of Sequence Classification. Several pretrained multilingual transformer models are investigated to find the better fit for the code-mixed datasets of Tamil, Kannada, and Malayalam. For our purpose, we define Single Task Learning (STL) Models, as when we train the transformer-based models on both of the tasks separately. In this section, we discuss several pretrained transformer-based models that have been used for both STL and MTL. Figure 1 represents the outline of the approaches undertaken.

4.1 Transformer-Based Models

Recurrent models such as LSTMs, GRUs fail to achieve state of the art results due to longer sequence lengths, owing to memory limitations while batching. While factorization and conditional computational approaches (Shazeer et al. 2017; Kuchaiev and Ginsburg 2017) have improved the efficiency of the model, yet the underlying issue of computing it sequentially persists. To overcome this, Transformer was proposed (Vaswani et al. 2017), an architecture that completely shuns recurrence and restores to attention mechanisms. It is found that adapting to an architecture with attention mechanisms proves to be much more efficient than recurrent architectures. The transformer block follows a stacked encoder-decoder architecture with multi-headed attention and feed forward layers. Self-attention is an attention mechanism relating distinct arrangements of a single sequence to compute a representation of a given sequence.

Scaled dot-product attention is mathematically computed using 3 vectors from each of the encoder's input vectors, *Query, Key* and *Value* vectors. *Key* and *Value* assume dimensions d_k and d_v respectively. A softmax function is applied on the dot product of queries and keys in order to compute the weights of the values. Practically, the attention function is computed simultaneously on a set of queries and then stacked into a matrix Q. In practice, the attention function is computed on a set of queries simultaneously, being packed into a matrix Q. The *Keys* and *Values* are packed into matrices K and V. The matrix of outputs is computed as follows:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \quad (1)$$

The above dot product attention is preferred over additive attention owing to its practical efficiency in both space and time complexities. Self attention is computed several times in transformer's architecture, thus it is referred to as multi-head Attention. This approach collectively attends information from different representations at different positions.

Consider the phrase - ***Ask Powerful Questions***. In order to calculate the self attention of the first word *Ask*, the

scores for all words in the phrase with respect to *Ask* is to be computed, which then determines the importance of other words when certain words are being encoded into the input sequence. The scores are divided by $8(2^3)$ which is the square root of the dimension of the key vector. The score of the first word is calculated using dot product attention , with the dot product of the query vector q_1 with keys k_1,k_2.k_3 of all words in the input sentence. The scores are then normalized using the softmax activation. The normalized scores are then multiplied by vectors v_1,v_2,v_3 and summed up to obtain self-attention vector z_1. It is then passed to feed forward network as input. The vectors for the other words are calculated in a similar way in dot product attention.

The normalized scores are then multiplied by vectors v_1,v_2,v_3 and summed up to obtain self-attention vector z_1. It is then passed to feed forward network as input. The vectors for the other words are calculated in a similar way in dot product attention. Softmax is an activation function that transforms the vector of numbers into a vector of probabilities. We use softmax function when we want a discrete variable representation of the probability distribution over *n* possible values. It is a generalization of sigmoid activation function which is used to represent the probability distribution over a binary variable. Softmax activation function over K classes is represented as follows:

$$softmax(z) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{2}$$

Whereas, sigmoid is a binary representation of softmax activation function. It is given by:

$$sigmoid(x) = \frac{e^x}{e^x + 1} \tag{3}$$

Along with attention sub-layers in the transformer block, a fully connected feed forward network (FN) persists in each of the layers in its encoder and decoder. It consists of a ReLU activation in between two positions.

$$FN(x) = \max(0, xW_1, b_1)W_2 + b_2 \tag{4}$$

ReLU, a rectified linear activation function, is a piece-wise linear function that will output the input directly if positive, else it will output it as zero. ReLU overcomes the vanishing gradient problem, thus allowing models to learn faster and perform better. It is mathematically computed as follows:

$$y = \max(0, x) \tag{5}$$

While attention mechanism is a major improvement over recurrence based seq2seq models, it does have its own limitations. As attention can only deal with fixed-length strings,

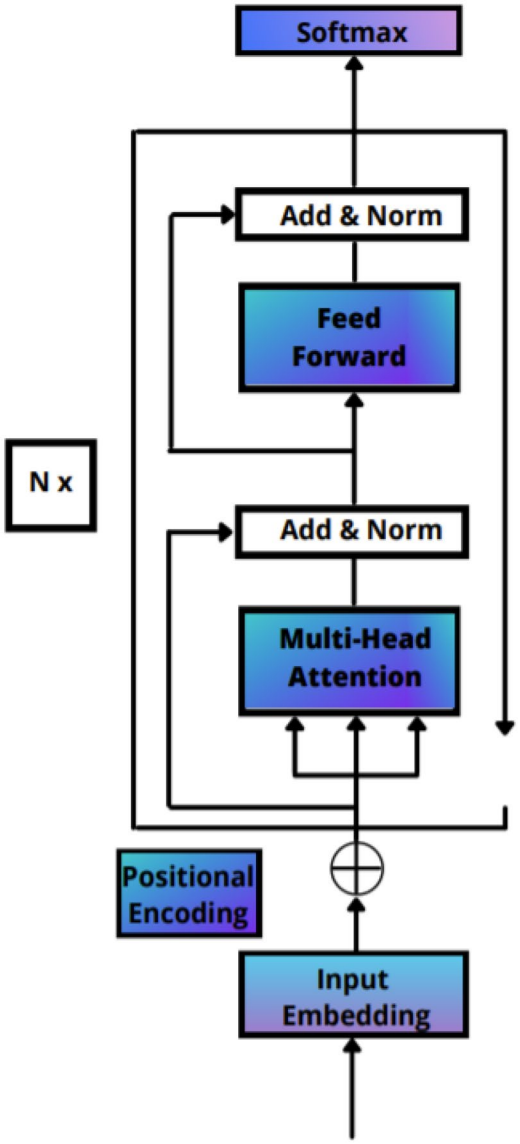


Fig. 2 A transformer architecture by Vaswani et al. (2017)

the text has to be split into a number of chunks before feeding them into the inputs. The chunking inadvertently results in context fragmentation. This means that a sentence, if split in the middle will result in significant loss of context. It would mean that the text is split without considering sentence or any other semantic boundary.

Due to the absence of recurrence or convolutional layers, some information about the tokens' relative or absolute position must be fed for the succession of sequence order in the model. Thus *Positional Encodings* is added to the input embeddings as shown in Fig. 2. Most of the state of the art language models assume the transformer block as its fundamental building block in each layer.

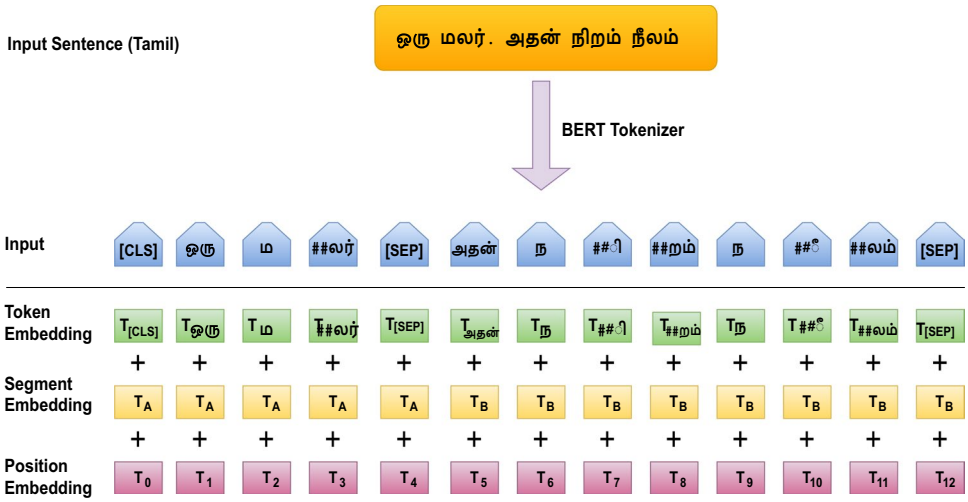


Fig. 3 Illustration of BERT input representations. The Tamil sentence translates to "This is a Flower. It is blue in colour." The input embeddings are the sum of token, segmentation and positional embeddings (Devlin et al. 2019)

4.1.1 BERT

The computer vision researchers have repeatedly demonstrated the advantages of transfer learning, pretraining a neural network model on a known task (Deng et al. 2009).

Bidirectional Encoder Representations from Transformers (BERT), a language representation model is designed to pretrain deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers (Devlin et al. 2019). It has been trained on eleven NLP tasks.

BERT is pretrained on two unsupervised tasks:

- Masked Language Modeling (MLM):
  Standard LMs (Radford 2018) can only be trained left-to-right or right-to-left. However, training it bidirectionally might allow the word to accidentally spot itself. To pretrain deep representations, a certain percentage of the input tokens are masked arbitrarily, and then predict those masked tokens, and are referred to as "Masked Language Modeling", which was previously stated as *Cloze* (Taylor 1953). Before feeding word sequences into BERT, 15% of the words in each sequence are replaced with a [MASK] token. The model then attempts to predict the initial value of the masked words, supported by the context provided by the opposite non-masked words within the sequence.
- Next Sentence Prediction (NSP):
  One of the main drawbacks of LM is its inability to capture the connection between two sentences directly. However, important downstream tasks such as Question Answering (QA) and Natural Language Interference (NLI) are supported with the relationships between two sentences. To overcome this obstacle pretraining for a *Binarized Next Sentence Prediction* task is performed. Specifically, when choosing the sentences A and B for every pretraining example, 50% of the time B is that the actual next sentence that follows A (labeled as *Is Next*),and 50% of the time it is a random sentence from the corpus (labeled as *Not Next*). In Fig. 3, the input consist of two Tamil sentences. After tokenizing the sentences, we observe that there are 13 input tokens inclusive of the [CLS] and [SEP] tokens. Tokens T_1, T_2, ..., T_13 represent the positional embeddings while T_A and T_B represent the tokens for whether a given sentence follows the other.

We use the sequence tagging specific inputs and outputs into BERT and fine-tune all parameters end-to-end. At the input, sentence A and sentence B from pretraining are equivalent to a degenerate text pair in sequence tagging. The [CLS] representation is fed into an output layer for classification as observed in Fig. 4.

4.1.2 DistilBERT

DistilBERT was proposed (Sanh et al. 2019) as a method to pretrain a smaller language representation model which serves for the same purpose as other large-scale models like BERT. DistilBERT is 60% faster and has 40% fewer parameters than BERT. It also preserves 97% of BERT performance in several downstream tasks by leveraging the knowledge distillation approach. Knowledge distillation is a refining approach in which a smaller model - the student - is trained to recreate the performance of a larger model - the teacher. DistilBERT is trained on a triple loss with student initialization.

Triple loss is a linear combination of the distillation loss L_{ce} with the supervised training loss, the Masked Language Modeling loss L_{mlm}, and a cosine embedding loss ($L_{\cos}$) which coordinates the direction of the student and teacher hidden state vectors, where

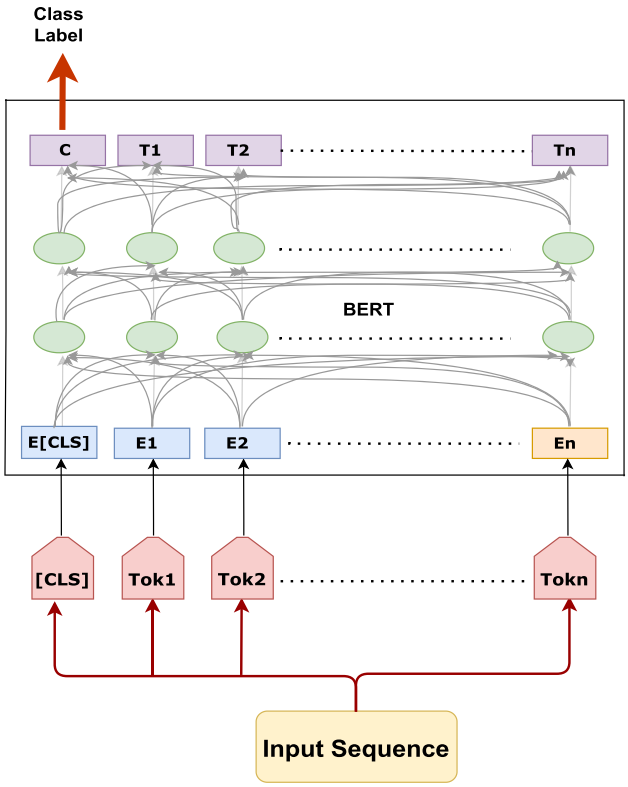


Fig. 4 Illustration of Fine-Tuning BERT on single sentence classification Tasks like SA and OLI (Devlin et al. 2019)

$$L_{ce} = \sum_i t_i * \log(s_i) \tag{6}$$

where ti (resp. si) is a probability estimated by the teacher (resp. the student) (Sanh et al. 2019).

We use a pretrained multilingual DistilBERT model from the huggingface transformer library *distilbert-base-multilingual-cased* for our purpose, which is distilled from the mBERT model checkpoint.

4.1.3 ALBERT

Present State of The Art (SoTA) LMs consist of hundreds of millions if not billions of parameters. In order to scale the models, we would be restricted by the memory limitations of compute hardware such as GPUs or TPUs. It is also found that increasing the number of hidden layers in the BERT-large model (340M parameters) can lead to worse performance. Several attempts of parameter reduction techniques to reduce the size of models without affecting their performance. Thus ALBERT: A lite BERT for self-supervised learning of language representations was proposed. ALBERT (Lan et al. 2019) overcomes the large memory consumption by incorporating several memory reduction techniques, Factorized Embedding Parameterization,

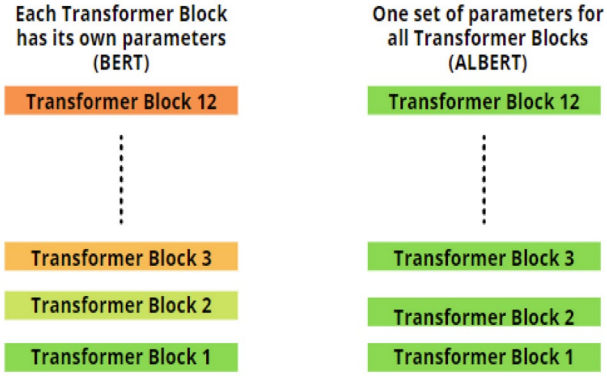


Fig. 5 No shared parameters in BERT vs cross-layer parameter sharing in ALBERT

cross-layer parameter sharing, and Sentence ordering Objectives.

– Factorized Embedding Parameterization

It is learned that WordPiece embeddings are designed to learn context independent representations, but, hidden layer embeddings are designed to learn context dependent representations. BERT heavily relies on learning context dependent representations with the hidden layers. It is found that embedding matrix E must scale with Hidden layers H, and thus, this results in models having billions of parameters. However, these parameters are rarely updated during training indicating that there are insufficient useful parameters. Thus, ALBERT has designed a parameter reduction method to reduce memory consumption by changing the result of the original embedding parameter P (the product of the vocabulary size V and the hidden layer size H).

$$V * H = P \rightarrow V * E + E * H = P \tag{7}$$

E represents the size of the low-dimensional embedding space. In BERT, E = H. While in ALBERT, H $\gg$ E, so the number of parameters will be greatly reduced.

– Cross-layer parameter sharing

ALBERT aims to elevate parameter efficiency by sharing all parameters, across all layers. Hence, feed forward network and attention parameters are shared across all layers as shown in Fig. 5.

– Sentence Order Prediction (SOP)

ALBERT uses masked language modeling, as similar to BERT, using up to 3 word masking (max(n_gram) =3). ALBERT also uses SOP for computing inter-sentence coherence loss. Consider two sentences being used in the same document. The positive test case is that the sentences are in a correct order, while the negative test case states that the sentences are not in a proper order. SOP results in the model learning finer-grained distinc-

tions about coherence properties, while additionally solving the Next Sentence Prediction (NSP) task to a rational degree. Thus, ALBERT models are more effective in improving downstream tasks' performance for multisentence encoding tasks.

By incorporating these features and loss functions, ALBERT requires much fewer parameters in contrast to the base and large versions of BERT models proposed earlier in Devlin et al. (2019), without hurting its performance.

4.1.4 RoBERTa

RoBERTa (Liu et al. 2019) is a robustly optimised BERT based model. The difference is within the masking technique. BERT performs masking once during the information processing, which is essentially a static mask, the resulting model tends to see the same form of masks in numerous training phases. RoBERTa was designed to form a dynamic mask within itself, which generated masking pattern changes every time the input sequence is fed in, thus, playing a critical role during pretraining. The encoding used was Byte-Pair encoding (BPE) (Sennrich et al. 2016), which may be a hybrid encoding between character and word level encoding that allows easy handling of huge text corpora, meaning it relies on subwords instead of full words.The model was made to predict the words using an auxiliary NSP loss. Even BERT was trained on this loss and was observed that without this loss, pretraining would hurt the performance, with significant degradation of results on QNLI, MNLI, and SQuAD.

4.1.5 XLM

The XLM model was proposed in Cross-lingual Language Model pretraining (Dai et al. 2019a). This model uses a shared vocabulary for different languages. For tokenizing the text corpus, Byte-Pair encoding (BPE) was used. Causal Language Modelling (CLM) was designed to maximize the probability of a token x_t to appear at the t th position in a given sequence. Masked Language Modelling (MLM) is when we maximize the probability of a given masked token x_t to appear at the t th position in a given sequence. Both CLM and MLM perform well on monolingual data. Therefore, XLM model used a translation language modelling. The sequences are taken from the translation data and randomly masks tokens from the source as well as from the target sentence. Similar to other transformer-based monolingual models, XLM was fine tuned on XLNI dataset for obtaining the cross-lingual classification. Downstream tasks on which this was evaluated on were tasks like Cross Lingual Classification, Neural Machine Translation (NMT) and LMs for low resource languages.

4.1.6 XLNet

BERT performed extremely well on almost every language modelling task. It was a revolutionary model as it could be fine-tuned for any downstream task. But even this came with a few flaws of its own. BERT was built in such a manner that it replaces random words in the sentences with a special [MASK] token and attempts to predict what the original word was. XLNet (Yang et al. 2019) pointed out certain major issues during this process. The [MASK] token which is used in the training would not appear during fine-tuning the model and for other downstream tasks. However, this approach could further create problems failing to replace [MASK] tokens at the end of pretraining. Moreover, the model finds it difficult to train when there are no [MASK] tokens in the input sentence. BERT also generates the predictions independently, meaning it does not care about the dependencies of its predictions.

XLNet uses auto regressive model (AR) language modeling which aims to estimate the probability distribution of a text corpus and without using the [MASK] token and parallel independent predictions. It is achieved through the AR modelling as it provides a reasonable way to express the product rule of factorizing the joint probability of the predicted tokens. XLNet uses a particular type of language modelling called the "permutation language modelling" in which the tokens are predicted for a particular sentence in random order rather than sequential order. The model is forced to learn bidirectional model dependencies between all combinations of the input. It is significant to note that it permutes only the factorization order and not the sequence order, and it is rearranged and brought back to the original form using the positional embedding. For sequence classification, the model is fine tuned for sentence classifier and it does not predict the tokens but predicts the the sentiment according to the embedding. The architecture of the XLNet uses transformer XL as its baseline (Dai et al. 2019b). The transformer adds recurrence to the segment level instead of the word level. Hence, fine-tuning is carried out by caching the hidden states of the previous states and passing them as keys or values when processing the current sequence. The transformer also uses the notion of relative embedding instead of positional embedding by encoding the relative distance between the words.

4.1.7 XLM-RoBERTa

XLM-RoBERTa was proposed as an unsupervised cross-lingual representation approach (Conneau et al. 2020), and it significantly outperformed multilingual BERT on a variety of cross-lingual benchmarks. XLM-R was trained on Wikipedia data of 100 languages and fine-tuned on different downstream tasks for evaluation and inference. The

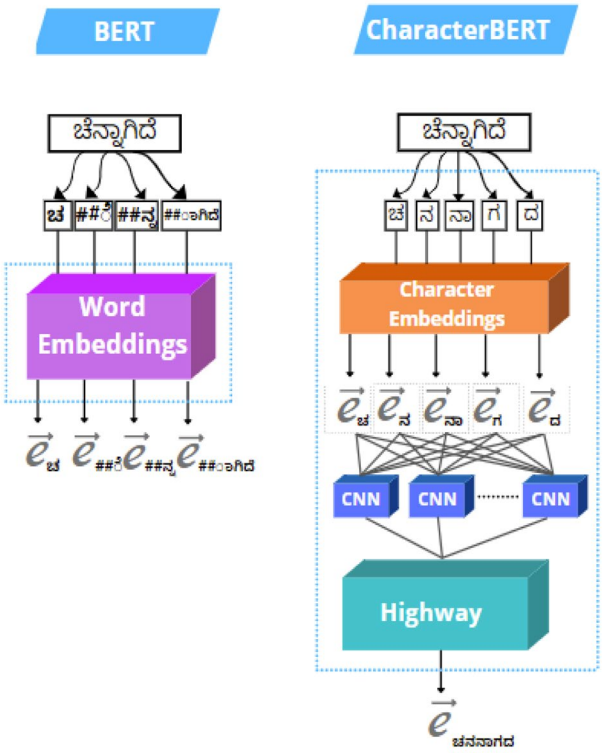


Fig. 6 Comparison of the context-independent representation systems and tokenization approach in BERT and CharacterBERT (El Boukkouri et al. 2020)

XNLI dataset was used for machine translation from English to other languages and vice-versa. It was checked on Named Entity Recognition (NER) and Cross Lingual Question Answering. It achieved great results on the standard GLUE benchmark and achieved state of the art results in several tasks.

4.1.8 CharacterBERT

The success of BERT has led many language representation models to adapt to the transformers architecture as their main building block, consequently inheriting the *wordpiece* tokenization. This could result in an intricate model that focuses on subword rather than word, for specialized domains. Hence, CharacterBERT, a new variant of BERT that completely drops the *wordpiece* system and uses a Character-CNN module instead to represent entire words by consulting their characters (El Boukkouri et al. 2020) as shown in Fig. 6.

The characterBERT is based on "base-uncased" version of BERT (L = 12, H = 768, A = 12, Total Parameters 109.5M). The subsequent CharacterBERT architecture has 104.6M parameters. Usage of character-CNN results in a smaller overall model, in spite of using a complex character module,mfor BERT's *wordpiece* matrix possesses 30K X 768-D vectors, while CharacterBERT utilizes 16-D character embeddings with the majority of small-sized CNNs. Seven 1-D CNNs with the following filters are used: [1, 32], [2, 32], [3, 64], [4, 128], [5, 256], [6, 512] and [7, 1024]. The Kannada word *chennagide* written in latin script, which can be translated as *Great*. Figure 6 compares how CharacterBERT focuses more on the entire word rather than subwords as observed in BERT.

4.2 Multi-Task Learning (MTL)

We generally focus more on optimizing a particular metric to score well on a certain benchmark. Thus, we train on a single model or an ensemble of models to perform our desired task. These are then fine-tuned until we no longer see any major increase in its performance. However, the downside to this is that we lose out on much of the information which could have helped our model to train better. If there are multiple related tasks, we could devise an MTL model which can further elevate its performance in multiple tasks by learning from shared representations between related tasks. We aim to achieve better F1-Scores by leveraging this approach. Since both SA and OLI are sequence classification tasks, we believe that an MTL model trained on these two related tasks will achieve better results, in comparison to their performances when trained separately on a model to predict a single task. Most of the methodologies of MTL in NLP are based on the success of MTL models in Computer Vision. We use two novel approaches for MTL in NLP, which are discussed in the subsequent sections.

4.2.1 Hard Parameter Sharing

This is one of the most common approaches to MTL. Hard parameter sharing of hidden layers (Caruana 1997) is applied by sharing the hidden layers between the tasks, while keeping the task-specific output layers separate. Hard parameter sharing is represented in Fig. 7. As both of the tasks are very similar, one of the tasks of the model is supposed to learn from the representations of the other task. The more are the number of tasks that we are learning simultaneously, the more the representations the model will capture and reduce the chances of overfitting in our model (Ruder 2017). As shown in Fig. 7, the model will have two outputs. We would be using different loss functions to optimize the performance of the models.

A loss function takes the (output, target) a pair of inputs, and computes a value that estimates how far away the output is from the target. To back propagate the errors, we first clear the gradients to prevent the accumulation of gradients in each epoch. In this method, we back propagate the losses separately and the optimizer updates the parameters for each

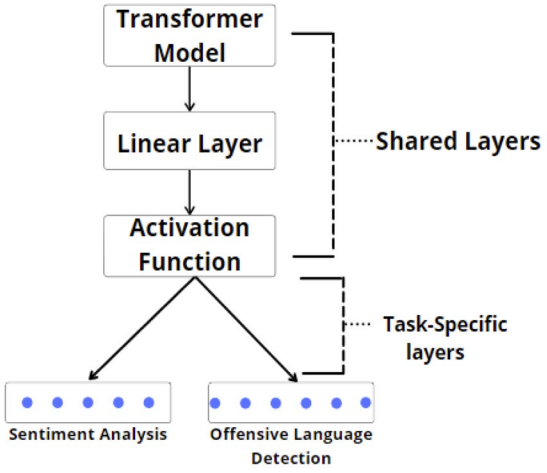


Fig. 7 Hard parameter sharing for sentiment analysis and offensive language identification

of the tasks separately. After the loss function calculates the losses, we add the losses of the two tasks and the optimizer then updates the parameters for both tasks simultaneously, where $Loss_{SA}$ and $Loss_{OLI}$ are losses for SA and OLI.

$$Loss = Loss_{SA} + Loss_{OLI} \tag{8}$$

4.2.2 Soft Parameter Sharing

This is another approach of MTL where constrained layers are added to support resemblance among related parameters. Unlike hard parameter sharing, each task has its own model, and it learns for each task to regularize the distances between the different models' parameters in order to encourage the parameter to be similar. Unlike hard parameter sharing, this approach gives more adaptability to the tasks by loosely coupling the representations of the shared space. We would be using Frobenious norm for our experiments as the additional loss term (Fig. 8).

1. Additional Loss Term

   This approach of soft parameter sharing is to impose similarities between corresponding parameters by augmenting the loss function with an additional loss term, as we have to learn 2 tasks. Let *Wij* denote the task j for the i^{th} layer.

$$L = l + \sum_{S(L)} \lambda_i \mid\mid W_i^{(S)} - W_i^{(O)} \mid\mid_{(F)}^2 \tag{9}$$

   Where l is the original loss function for both of the tasks and ||.||$F2$ is the squared Frobenious norm. A similar approach was used for low resource dependency parsing, by employing a cross-lingual parameter sharing whose learning objective involved regularization using Frobenious norm (Duong et al. 2015).

2. Trace Norm

   We intend to penalize trace norm resulting from stacking *WiS* and *WiO*. A penalty encourages estimation of shrinking coefficient estimates thus resulting in dimension reduction (Yang and Hospedales 2017). The trace norm can be defined as the sum of singular values

$$|| \mathbf{W} ||= \sum\nolimits_i \sigma_i \tag{10}$$

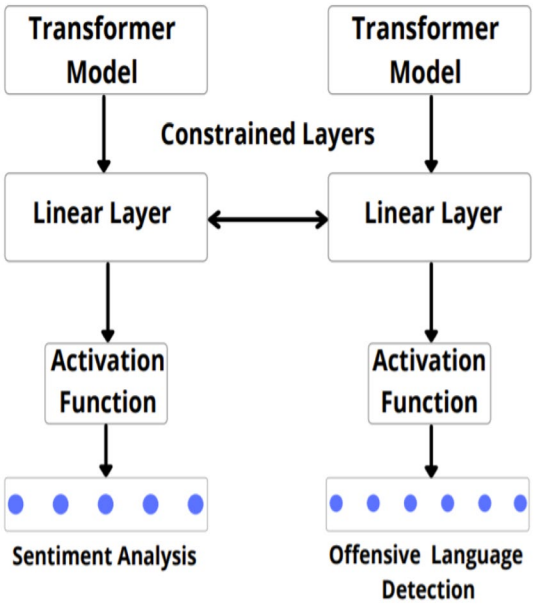


Fig. 8 Soft parameter sharing for sentiment analysis and offensive language identification

4.3 Loss Functions

Loss function is a technique of evaluating the effectiveness of specific algorithm models on the given data. If the predictions are deviating from expected results, the purpose of a loss function is to output a high value. The output of the loss function decreases when the predictions start to match the expected results. With the help of an optimization function, a loss function comprehends to reduce the error in the predictions. In this section, we will be discussing several loss functions that are going to be used in our experiments.

4.3.1 Cross Entropy Loss

This is one of the most commonly used loss functions for classification tasks. It quantifies from the study of information theory and entropy and is calculated as the difference of the probability distributions for a given random variable. Entropy can be defined as the number of entities required for the transmission of a randomly selected event in a probability distribution. "Cross Entropy is the average number of bits needed to encode data from a source coming from a distribution p when we use a model q" (Murphy 2012). The divination of this definition can be conveyed if we consider a target as an underlying probability distribution P and an approximation of the target probability distribution as Q. Let the cross-entropy between two probability distributions, Q from P, be stated as H(P,Q). It is calculated as follows:

$$H(P,Q) = \sum_x P(x)\log(Q(x)) \tag{11}$$

For the sake of multi-label classification, it is computed as follows:

$$L_{CE} = \sum_i^C t_i \log(s_i) \tag{12}$$

Where t_i and s_i are the ground truths for each class i in C.

Considering Class Imbalance As specified earlier we observe that the datasets have class imbalances and are not equally distributed as shown in Table 1. Thus we consider the weights of each class and pass the weights as a tensor to the parameter while computing the loss. Class Weights give inverse class weights to penalize the underrepresented class for class imbalance.

$$W_i = 1 - \frac{W_i}{\sum_i^C W_i} \tag{13}$$

where C is the number of classes. The resultant tensor is $[w_1,w_2,...,w_{c-1},w_c]$

4.3.2 Multi-class Hinge Loss

Hinge Loss function was initially proposed as an alternative to cross-entropy for binary classification. It was primarily developed for use with SVM models. The targets values are in the set {-1, 1}. The main purpose of hinge loss is to maximize the decision boundary between two groups that are to be discriminated, i.e, a binary classification problem. For linear classifiers w,x, hinge loss is computed as follows

$$L(w,(x,y)) = max\{0, 1 - y\{w,x\}\} \tag{14}$$

Squared Hinge Loss function was developed as an extension, which computes the square of the score hinge loss. It smoothens the surface of the error function, thus making it easier for computational purposes.

Categorical Hinge Loss or multiclass hinge loss is computed as follows, "For a prediction y, take all values unequal to t, and compute the loss. Eventually, sum them all together to find the multiclass hinge loss" (Weston and Watkins 1999; Zhang et al. 2014; Rakhlin 2016; Shalev-Shwartz and Ben-David 2014). The regular targets are computed into categorical data and the loss function is calculated as follows

$$l(y) = \sum_{y \neq t} max(0, 1 + W_y X - W_t X) \tag{15}$$

4.3.3 Focal Loss

The Focal Loss was initially proposed for dense object detection task as an alternative to CE loss. It enables training highly accurate dense object detectors for highly imbalanced datasets (Lin et al. 2017). For imbalanced text datasets, it handles class imbalance for by calculating the ratio of each class and thus assigning weights based on whether it is hard or soft examples (inliers) (Tula et al. 2021; Ma et al. 2020).

In CE, easily classified examples incur a loss with non-trivial magnitude. We define focal loss by adding a modulating factor loss to the cross entropy loss with a tunable focusing parameter $\gamma \geq 0$ (Lin et al. 2017) as:

$$FL(p_t) = -(1 - p_t)^{\gamma} \log(p_t) \tag{16}$$

where $p \in [0,1]$ is the estimated probability of the model for a given class. When $\gamma = 0$, the above formula represents CE. If $\gamma > 1$, the modulating factor assists in reducing the loss for easily classified examples ($p_t \geq 0.5$) resulting in more number of corrections of misclassified examples.

4.3.4 Kullback Leibler Divergence Loss

The Kullback-Leibler Divergence (KLD) measures the difference in the probability distribution of two random variables (Kullback and Leibler 1951). Theoretically, a KLD score of 0 indicates that the two distributions are identical.

It is defined as "the average number of extra bits needed to encode the data, due to the fact that we used the distribution q to encode the data instead of the true distribution p" (Murphy 2012), it is also known as relative entropy. It is generally used for models that have complex functions such as Variational Autoencoders (VAEs) for text generation rather than simple multi-class classification tasks (Prokhorov et al. 2019; Asperti and Trentin 2020). Consider two random distributions, P and Q. The main intuition behind KLD is that if the probability for an event from Q is small, but that

Table 2 Class-wise distribution of the test sets of Kannada, Malayalam, and Tamil datasets

Sentiment analysis	Kannada	Malayalam	Tamil	Offensive language identification	Kannada	Malayalam	Tamil
Positive	334	544	2,426	Not Offensive	407	1,134	3,148
Negative	164	135	529	Offensive Untargeted	27	28	336
Mixed feelings	63	81	465	Offensive Targeted Individual	82	16	316
Neutral	80	424	702	Offensive Targeted Group	44	10	305
Other language	87	94	213	Offensive Targeted Others	14	-	52
–	–	–	–	Other Language	154	90	178
Total	728	1,278	4,335		728	1,278	4,335

of P is large, then there exists a large divergence. It is used to compute the divergence between discrete and continuous probability distributions (Brownlee 2019). It is computed as follows

$$D_{KL}(p||q) = \sum_{i=1}^{N} p(x_i) \log \frac{p(x_i)}{q(x_i)} \tag{17}$$

5 Experiments and Evaluation

5.1 Experiments Setup

We use the pretrained models available on Huggingface Transformers[2] (Wolf et al. 2020), while fine-tuning the model on a python-based deep learning framework, Pytorch[3] (Paszke et al. 2019) and Scikit-Learn[4] (Pedregosa et al. 2011) for evaluating the performance of the models. For MTL experiments, we take the naive weighted sum by assigning equal weights to the losses (Eigen and Fergus 2015). We have experimented with various hyperparameters that are listed as shown in Table 7. The experiments were conducted by dividing the dataset into three parts: 80% for training, 10% for validation, and 10% for testing. The class-wise support of the test is shown in Fig. 2 The experiments of all models were conducted on Google Colab (Bisong 2019).

- Epoch: As the pretrained LMs consist of hundreds of Millions of parameters, we limit training up to 5 epochs, owing to memory limitations (Pan and Yang 2009).
- Batch Size: Different Batch sizes were used among [16, 32, 64].
- Optimizer: we use Adam optimizer with weight decay (Loshchilov and Hutter 2019).
- Loss weights: We give equal importance to two tasks and add losses as shown in Formula (8).

5.2 Text Preprocessing

Preprocessing is one of the important steps in any NLP systems, especially for text classification tasks (Uysal and Gunal 2014). Comments on YouTube are grammatically inconsistent in nature, and thus contain emojis, mentions, hashtags, elongated words, expressions and urls which makes the process of tokenization quite streneous (del Arco et al. 2021). To overcome these challenges we follow these steps:

- Converting the comments to lower case.
- The emojis are replaced by the words that the emoji represents like happy, sad, among other emotions depicted by emojis. As emojis mainly depict the intention of a user, it would be imperative to replace them with their meanings to pick up their cues. As majority of the models are pretrained only on unlabelled text, we feel that it would be necessary.
- URLs and other links are replaced by the word, 'URL'.
- Multiple spaces in a sentence and other special characters are removed as they do not contribute significantly in the overall intention of a comment.

5.3 Transfer Learning Fine-Tuning

We have exhaustively implemented several pretrained language models by fine-tuning them for text classification. All of the models we use are pretrained on large corpora consisting of unlabelled text. As we are dealing with code-mixed text, it would be interesting to see the performance of the models in Kannada, Malayalam, and Tamil, as all models are pretrained on either monolingual or multilingual corpora, it is imperative that the fine-tuned models could have a difficulty to classify code-mixed sentences. For the optimizer, we leverage weight decay in Adam optimizer (AdamW), by decoupling weight decay from the gradient update (Loshchilov and Hutter 2019; Kingma and Ba 2014).

[2] https://huggingface.co/transformers/index.html

[3] https://pytorch.org/

[4] https://scikit-learn.org/

The primary step is to use the pretrained tokenizer to first cleave the word into tokens. Then, we add the special tokens needed for sentence classification ([CLS] at the first position, and [SEP] at the end of the sentence as shown in Fig.3). In the figure, tokens T1, T2, ..., Tn represent the cleaved tokens obtained after tokenizing. After special tokens are added, the tokenizer replaces each token with its id from the embedding table which is a component we obtain from the pretrained model.

BERT (Devlin et al. 2019) was originally pretrained on English texts, and was later extended for mbert (Pires et al. 2019), which is a language model pretrained on the Wikipedia dumps of the top 104 languages. mBERT consists of 512 input tokens, output being represented as a 768 dimensional vector and 12-attention heads. It is worth noting that we have used both multilingual and monolingual models (pretrained in English), to analyse the improvements, as we are dealing with code-mixed texts. There are two models of mBERT that are available, for our task, we use the BERT-base, Multilingual Cased checkpoint[5]. In MTL, we use distilbert-base-multilingual-cased for Tamil, while bert-base-multilingual-cased on Malayalam and Kannada datasets, based on their performances on STL on the DravidianCodeMix dataset.

5.4 Results Analysis

This section entails a comprehensive analysis of the capabilities of several pretrained LMs. We have employed the popular metrics in NLP tasks, including Precision (P), Recall (R), F1-Score (F), weighted average, and macro-average. F1-Score is the harmonic average of recall and precision, taking values between 0 and 1. The metrics are computed as follows,

$$Precision(c) = \frac{TP}{TP + FP} \tag{18}$$

$$Recall(c) = \frac{TP}{TP + FN} \tag{19}$$

$$F1 - Score(c) = \frac{2 * P(c) * R(c)}{P(c) + R(c)} \tag{20}$$

Where, c is the number of classes:

- TP—True Positive examples are predicted to be positive and are positive;
- TN—True Negative examples are predicted to be negative and are negative;
- FP—False Positive examples are predicted to be positive but are negative;
- FN—False Negative examples are predicted to be negative but are positive.

The weighted-Average and Macro-Average are computed as follows:

$$Precision_{M-Avg} = \frac{\sum_i^c TP_i}{\sum_i^c (TP_i + FP_i)} \tag{21}$$

$$Recall_{M-Avg} = \frac{\sum_i^c TP_i}{\sum_i^c (TP_i + FN_i)} \tag{22}$$

We have evaluated the performance of the models on various metrics such as precision, recall, and F1-Score. Accuracy gives more importance to the TP and TN, while disregarding FN and FP. F1 score is the harmonic average of precision and recall. Therefore, this score takes both FP and FN into account. Due to the persistence of class imbalance in our datasets, we use weighted F1-Score as the evaluation metric. The benefits of MTL is multifold as these two tasks are related to each other. MTL offers several advantages such as improved data efficiency, reduces overfitting while faster learning by leveraging auxillary representations (Ruder 2017). As a result, MTL allows us to have a single shared model in lieu of training independent models per task (Dobrescu et al. 2020). Hard parameter sharing involves the practice of sharing model weights between multiple tasks, as they are trained jointly to minimize multiple losses. However, soft parameter sharing involves all individual task-specific models to have different weights which would add the distance between the different task specific models that would have to be optimized (Crawshaw 2020). We intend to experiment with MTL to achieve a slight improvement in the performance of the model along with reduced time and space constraints in hard parameter sharing (Table 2).

Apart from the pretrained LMs shown in Tables 3, 4, and 5, we have tried other domain specific LMs. IndicBERT (Kakwani et al. 2020) is a fastText based word embeddings and ALBERT-based LM trained on 11 Indian languages along with English, and is a multilingual model similar to mBERT (Pires et al. 2019). IndicBERT was pretrained on IndicCorp (Kakwani et al. 2020), the sentences of which were trained using a sentence piece tokenizer (Kudo and Richardson 2018). However, when IndicBERT was fine-tuned for sentiment analysis and offensive language identification, it was found that the model performed very poorly, despite being pretrained on 12 languages, inclusive of Kannada, Malayalam, Tamil, and English. We believe that one of the main reasons for its poor performance, despite being pretrained on a large corpus, has to do with the architecture

[5] https://github.com/google-research/bert/blob/master/multilingual.md

Table 3 Precision, Recall, and F1-Scores of STL approach for Kannada

Models	BERT	DistilBERT	ALBERT	RoBERTa	XLM	XLNet	XLM-R	CharBERT
Precision (Sentiment Analysis)								
Positive	0.701	0.692	0.634	0.667	0.681	0.711	0.567	0.631
Negative	0.573	0.501	0.556	0.120	0.552	0.571	0.520	0.689
Mixed Feelings	0.329	0.443	0.0	0.162	0.266	0.292	0.0	0.248
Neutral	0.371	0.551	0.350	0.264	0.568	0.498	0.467	0.602
Other Language	0.564	0.192	0.496	0.012	0.498	0.508	0.073	0.503
Macro Average	0.509	0.471	0.427	0.245	0.461	0.441	0.325	0.452
Weighted Average	0.592	0.564	0.536	0.362	0.519	0.587	0.418	0.542
Precision (offensive language identification)								
Not Offensive	0.781	0.778	0.715	0.679	0.759	0.719	0.735	0.749
Offensive Untargeted	0.0	0.123	0.0	0.0	0.0	0.0	0.0	0.00
Offensive Targeted Individual	0.452	0.461	0.654	0.769	0.798	0.641	0.085	0.721
Offensive Targeted Group	0.0	0.391	0.0	0.571	0.371	0.0	0.0	0.428
Offensive Targeted Others	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Other Language	0.704	0.731	0.702	0.748	0.678	0.712	0.008	0.678
Macro Average	0.321	0.408	0.345	0.461	0.431	0.345	0.138	0.431
Weighted Average	0.651	0.691	0.622	0.656	0.678	0.625	0.425	0.669
Recall (Sentiment Analysis)								
Positive	0.742	0.753	0.793	0.667	0.663	0.649	0.759	0.867
Negative	0.581	0.442	0.524	0.467	0.676	0.778	0.604	0.532
Mixed Feelings	0.100	0.229	0.0	0.074	0.108	0.061	0.0	0.051
Neutral	0.493	0.581	0.263	0.144	0.376	0.401	0.073	0.343
Other Language	0.590	0.209	0.678	0.006	0.608	0.702	0.117	0.618
Macro Average	0.501	0.441	0.452	0.271	0.508	0.521	0.311	0.479
Weighted Average	0.602	0.572	0.592	0.361	0.628	0.612	0.470	0.628
Recall (offensive language identification)								
Not Offensive	0.842	0.813	0.875	0.917	0.858	0.708	0.746	0.864
Offensive Untargeted	0.0	0.041	0.0	0.0	0.0	0.0	0.0	0.0
Offensive Targeted Individual	0.533	0.629	0.415	0.484	0.599	0.610	0.025	0.653
Offensive Targeted Group	0.0	0.251	0.0	0.089	0.229	0.0	0.0	0.303
Offensive Targeted Others	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Other Language	0.741	0.748	0.812	0.601	0.768	0.872	0.038	0.731
Macro Average	0.379	0.412	0.350	0.349	0.408	0.365	0.135	0.409
Weighted Average	0.719	0.709	0.707	0.696	0.714	0.706	0.418	0.718
F1-Score (Sentiment Analysis)								
Positive	0.721	0.723	0.705	0.665	0.714	0.681	0.649	0.729
Negative	0.573	0.473	0.583	0.190	0.662	0.663	0.559	0.608
Mixed Feelings	0.153	0.301	0.0	0.102	0.182	0.102	0.0	0.089
Neutral	0.424	0.558	0.300	0.187	.449	0.443	0.126	0.431
Other Language	0.581	0.201	0.573	0.008	0.548	0.591	0.090	0.609
Macro Average	0.493	0.453	0.432	0.230	0.498	0.491	0.285	0.489
Weighted Average	0.591	0.561	0.556	0.349	0.595	0.589	0.418	0.594
F1-Score (offensive language identification)								
Not Offensive	0.811	0.801	0.787	0.780	0.792	0.788	0.741	0.801
Offensive Untargeted	0.0	0.061	0.0	0.0	0.0	0.0	0.0	0.0
Offensive Targeted Individual	0.529	0.534	0.507	0.594	0.685	0.625	0.039	0.649
Offensive Targeted Group	0.0	0.299	0.0	0.154	0.285	0.0	0.0	0.348
Offensive Targeted Others	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Other Language	0.743	0.741	0.753	0.667	0.725	0.710	0.013	0.698
Macro Average	0.351	0.421	0.341	0.366	0.419	0.354	0.132	0.418
Weighted Average	0.686	0.698	0.656	0.651	0.685	0.661	0.416	0.680

of the model which was also encountered by Puranik et al. (2021). Even though IndicBERT was pretrained in a multilingual setting, it followed the architecture of ALBERT using the standard masked language modeling objective. We believe that this is due to the cross-layer parameter sharing that hinders its performance in a multilingual setting (when its pretrained on more than one language). Consequentially, ALBERT tends to focus on the overall spatial complexity and training time (Lan et al. 2019).

We have also experimented with Multilingual Representations for Indian Languages (MuRIL), a pretrained LM that was pretrained on Indian languages as similar to IndicBERT but differs in terms of pretraining strategy and the corpus used (Khanuja et al. 2021). It follows the architecture of BERT-base encoder model, however it is trained on two objectives, masked language modeling, and transliterated language modeling (TLM). TLM leverages parallel data unlike the former training objective. However, in spite of having a TLM objective, the model performed worse than BERT base. which was the base encoder model for MuRIL. Hence, we have not tabulated the classification report of both MuRIL and IndicBERT. We will be analysing the performance of models on different datasets separately.

5.4.1 Kannada

Tables 3 and 6 presents the classification report of the models on Kannada dataset. We opted to train the STL with CE Loss, which is a popular choice among loss functions for classification instances (De Boer et al. 2005). We observe that mBERT (multilingual-bert) achieves the highest weighted F1-Score among other models as observed in Table 3.

mBERT achieved a 0.591 for sentiment analysis and 0.686 for offensive language identification. In spite of achieving the highest weighted average of F1-Scores of the classes, we can observe that the some classes have performed very poorly with 0.0 as their F1-Score. In offensive language identification, we can see that most of the models perform poorly on 3 classes, *Offensive Untargeted*, *Offensive Targeted Group*, and *Offensive Targeted Others*. One of the main reasons could be due to the less support of these classes in the test set. *Offensive Untargeted* has a support of 27, while *offensive Targeted Group* has 44 and *Offensive Targeted others* has a mere 14 out of the total test set having a support of 728. As the ratio of the majority class to the minority class is very severe (407:14), which is essentially the reason why the weighted F1-Scores of *Not Offensive* and *Offensive Targeted Individual* are much higher in comparison to the low-resourced classes. Among monolingual LMs, CharacterBERT performs at par with mBERT. It is interesting to note that we use *general-character-bert*[6], a LM that was pretrained only on English Wikipedia and OpenWebText (Liu and Curran 2006). We believe the approach employed in characterBERT, of attending to the characters to attain a single word embedding as opposed to the process of tokenization in BERT, as illustrated in Fig. 6. This is one of the reasons why CharacterBERT outperforms other monolingual models such as RoBERTa, ALBERT, and XLNet. Despite the superiority of other models that are pretrained on more data and better strategies.

To our surprise, we observe that XLM-RoBERTa, a multilingual LM pretrained on the top 100 languages, performed the worst among the models. One of the main reasons for the low performance of XLM-RoBERTa (-20%) could be on the account of low support of the test set. However, due to the nature of the language, it would be hard to extract more data. Another reason for the poor performance is based on the architecture of RoBERTa models, upon which the XLM-R model is based. Unlike the conventional wordpiece (Schuster and Nakajima 2012) and Unigram tokenizer (Kudo 2018), RoBERTa uses a Byte-level BPE tokenizer (Sennrich et al. 2016) for its tokenization process. However, BPE tends to have a poor morphological alignment with the original text (Jain et al. 2020). As Dravidian Languages are morphologically rich (Tanwar and Majumder 2020), this approach results in poor performance on sentiment analysis and offensive language identification. DistilBERT scores more than the other multilingual models such as XLM and XLM-R, in spite of having very few parameters in comparison to large LMs such as XLM-R base (66M vs 270M). However, among all models, mBERT had the best overall scores in both tasks. Hence, we used BERT to perform the MTL models by training them for hard parameter sharing and soft parameter sharing.

We have fine-tuned BERT for these loss functions as it outperformed other pretrained LMs for CEs. We have employed four loss functions. The highest weighted F1-Score for sentiment analysis was 0.602, which was achieved when mBERT was trained in soft parameter sharing strategy with CE as the loss function. There is an increase of 0.101 in contrast to STL. However, the output of the second task scored 0.672, which is lower than the STL score (-0.140 F1-Score). As we take the naive sum of the losses when training in a multi-task setting, the losses of one task suppresses the other task, thus, only one task majorly benefiting from the approach (Maninis and Radosavovic 2019). However, it is worth noting that the approach did achieve a competitive weighted F1-Score, if not greater than the former. The highest F1-Score achieved for offensive language identification was 0.700, which was achieved by training mBERT in hard parameter sharing strategy, with CE as its loss function. However, the same issue of supression

[6] https://github.com/helboukkouri/character-bert#pretrained-models

Table 4 Precision, Recall, and F1-Scores of STL approach for Malayalam

Models	BERT	DistilBERT	ALBERT	RoBERTa	XLM	XLNet	XLM-R	CharBERT
Precision (Sentiment Analysis)								
Positive	0.704	0.657	0.446	0.615	0.713	0.695	0.750	0.673
Negative	0.343	0.0	0.0	0.512	0.536	0.521	0.494	0.623
Mixed Feelings	0.0	0.0	0.0	0.175	0.521	0.571	0.562	0.447
Neutral	0.664	0.600	0.333	0.712	0.765	0.591	0.751	0.679
Other Language	0.750	0.750	0.622	0.744	0.735	0.814	0.706	0.787
Macro Average	0.492	0.402	0.280	0.552	0.654	0.639	0.653	0.642
weighted Average	0.612	0.533	0.346	0.618	0.701	0.643	0.708	0.664
Precision (offensive language identification)								
Not Offensive	0.939	0.947	0.933	0.946	0.929	0.951	0.716	0.934
Offensive Untargeted	0.0	0.529	0.0	0.400	0.0	0.0	0.0	1.000
Offensive Targeted Individual	0.0	0.222	0.0	0.0	0.0	0.027	0.0	0.0
Offensive Targeted Group	0.0	0.0	0.0	0.0	0.0	0.028	0.0	0.0
Other Language	0.717	0.716	0.744	0.807	0.756	0.045	0.829	0.724
Macro Average	0.331	0.483	0.335	0.431	0.337	0.210	0.353	0.532
weighted Average	0.884	0.905	0.880	0.904	0.878	0.847	0.885	0.901
Recall (Sentiment Analysis)								
Positive	0.801	0.826	0.963	0.829	0.827	0.697	0.776	0.814
Negative	0.274	0.0	0.0	0.304	0.437	0.363	0.630	0.244
Mixed Feelings	0.0	0.0	0.0	0.136	0.309	0.147	0.233	0.159
Neutral	0.719	0.717	0.009	0.531	0.505	0.559	0.596	0.712
Other Language	0.742	0.717	0.596	0.681	0.566	0.411	0.566	0.528
Macro Average	0.506	0.452	0.314	0.496	0.509	0.495	0.503	0.502
weighted Average	0.663	0.642	0.457	0.620	0.608	0.640	0.609	0.621
Recall (offensive language identification)								
Not Offensive	0.983	0.966	0.981	0.976	0.983	0.648	0.935	0.979
Offensive Untargeted	0.0	0.321	0.0	0.286	0.0	0.0	0.0	0.071
Offensive Targeted Individual	0.0	0.125	0.0	0.0	0.0	0.125	0.0	0.0
Offensive Targeted Group	0.0	0.0	0.0	0.0	0.0	0.100	0.0	0.0
Other Language	0.789	0.756	0.711	0.789	0.656	0.156	0.815	0.700
Macro Average	0.353	0.434	0.338	0.410	0.328	0.206	0.359	0.350
Weighted Average	0.922	0.919	0.920	0.928	0.919	0.588	0.886	0.919
F1-Score (Sentiment Analysis)								
Positive	0.750	0.732	0.609	0.706	0.666	0.666	0.662	0.637
Negative	0.305	0.0	0.0	0.381	0.482	0.428	0.554	0.351
Mixed Feelings	0.0	0.0	0.0	0.153	0.388	0.145	0.219	0.128
Neutral	0.691	0.654	0.018	0.608	0.634	0.665	0.622	0.655
Other Language	0.742	0.733	0.609	0.711	0.591	0.627	0.605	0.598
Macro Average	0.497	0.424	0.247	0.512	0.624	0.481	0.488	0.492
Weighted Average	0.635	0.581	0.310	0.605	0.623	0.630	0.603	0.624
F1-Score (offensive language identification)								
Not Offensive	0.958	0.956	0.956	0.961	0.955	0.771	0.962	0.956
Offensive Untargeted	0.0	0.400	0.0	0.333	0.0	0.0	0.0	0.133
Offensive Targeted Individual	0.0	0.160	0.0	0.0	0.0	0.044	0.0	0.0
Offensive Targeted Group	0.0	0.0	0.0	0.0	0.0	0.043	0.0	0.0
Other Language	0.751	0.685	0.727	0.798	0.702	0.070	0.836	0.712
Macro Average	0.342	0.440	0.337	0.418	0.332	0.186	0.356	0.360
Weighted Average	0.902	0.901	0.900	0.900	0.897	0.690	0.898	0.901

Table 5 Precision, Recall, and F1-Scores of STL approach for Tamil

Models	BERT	DistilBERT	ALBERT	RoBERTa	XLM	XLNet	XLM-R	CharBERT
Precision (Sentiment Analysis)								
Positive	0.713	0.740	0.664	0.702	0.689	0.686	0.714	0.676
Negative	0.413	0.417	0.393	0.392	0.462	0.444	0.399	0.505
Mixed Feelings	0.411	0.329	0.436	0.345	0.440	0.419	0.397	0.307
Neutral	0.444	0.472	0.464	0.465	0.534	0.423	0.526	0.540
Other Language	0.713	0.621	0.667	0.630	0.675	0.660	0.517	0.543
Macro Average	0.520	0.516	0.525	0.507	0.560	0.527	0.511	0.514
Weighted Average	0.596	0.607	0.574	0.584	0.609	0.584	0.602	0.587
Precision (offensive language identification)								
Not Offensive	0.797	0.860	0.796	0.826	0.815	0.837	0.853	0.839
Offensive Untargeted	0.294	0.385	0.284	0.438	0.428	0.370	0.409	0.368
Offensive Targeted Individual	0.0	0.371	0.0	0.384	0.382	0.377	0.409	0.349
Offensive Targeted Group	0.0	0.318	0.0	0.297	0.423	0.329	0.454	0.323
Offensive Targeted Others	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Other Language	0.840	0.829	0.727	0.857	0.780	0.660	0.806	0.780
Macro Average	0.322	0.461	0.301	0.467	0.472	0.429	0.458	0.451
Weighted Average	0.636	0.738	0.630	0.718	0.715	0.714	0.725	0.719
Recall (Sentiment Analysis)								
Positive	0.854	0.805	0.882	0.827	0.888	0.863	0.831	0.891
Negative	0.391	0.401	0.265	0.427	0.329	0.323	0.440	0.274
Mixed Feelings	0.129	0.228	0.131	0.123	0.174	0.095	0.174	0.144
Neutral	0.373	0.442	0.322	0.365	0.366	0.386	0.363	0.298
Other Language	0.554	0.601	0.404	0.545	0.526	0.474	0.582	0.620
Macro Average	0.460	0.495	0.401	0.457	0.457	0.428	0.478	0.445
Weighted Average	0.627	0.625	0.612	0.614	0.641	0.618	0.625	0.626
Recall (offensive language identification)								
Not Offensive	0.964	0.887	0.947	0.935	0.946	0.910	0.915	0.919
Offensive Untargeted	0.345	0.408	0.384	0.339	0.432	0.426	0.479	0.414
Offensive Targeted Individual	0.0	0.288	0.0	0.266	0.082	0.275	0.326	0.288
Offensive Targeted Group	0.0	0.351	0.0	0.134	0.154	0.092	0.210	0.102
Offensive Targeted Others	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Other Language	0.618	0.708	0.539	0.742	0.719	0.775	0.715	0.685
Macro Average	0.321	0.440	0.312	0.403	0.389	0.413	0.439	0.401
Weighted Average	0.752	0.750	0.740	0.765	0.767	0.752	0.741	0.755
F1-Score (Sentiment Analysis)								
Positive	0.777	0.771	0.758	0.759	0.776	0.765	0.768	0.769
Negative	0.402	0.409	0.316	0.409	0.384	0.374	0.419	0.355
Mixed Feelings	0.196	0.269	0.202	0.181	0.250	0.154	0.242	0.196
Neutral	0.406	0.456	0.380	0.409	0.434	0.404	0.430	0.384
Other Language	0.586	0.611	0.503	0.584	0.591	0.552	0.552	0.579
Macro Average	0.473	0.503	0.432	0.469	0.487	0.450	0.489	0.457
Weighted Average	0.600	0.614	0.571	0.589	0.607	0.583	0.609	0.585
F1-Score (offensive language identification)								
Not Offensive	0.873	0.873	0.865	0.877	0.876	0.872	0.882	0.877
Offensive Untargeted	0.317	0.396	0.326	0.383	0.430	0.396	0.441	0.387
Offensive Targeted Individual	0.0	0.324	0.0	0.314	0.135	0.318	0.362	0.315
Offensive Targeted Group	0.0	0.334	0.0	0.185	0.226	0.144	0.287	0.155
Offensive Targeted Others	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Other Language	0.712	0.764	0.619	0.795	0.749	0.713	0.737	0.751
Macro Average	0.317	0.448	0.302	0.426	0.403	0.407	0.430	0.414
Weighted Average	0.688	0.743	0.679	0.735	0.726	0.727	0.734	0.731

Table 6 MTL results on the Kannada Dataset

	Hard parameter sharing				Soft parameter sharing			
Losses	Cross Entropy	Hinge Loss	Focal Loss	KLD	CE	HL	FL	KLD
Precision (Sentiment Analysis)								
Positive	0.683	0.673	0.666	0.459	0.713	0.534	0.679	0.459
Negative	0.677	0.685	0.651	0.0	0.596	0.733	0.631	0.0
Mixed Feelings	0.0	0.0	0.0	0.0	0.143	0.0	0.0	0.0
Neutral	0.5	0.700	0.479	0.0	0.491	0.577	0.492	0.0
Other Language	0.442	0.477	0.459	0.0	0.538	0.703	0.454	0.0
Macro Average	0.460	0.507	0.451	0.092	0.496	0.509	0.451	0.092
Weighted Average	0.574	0.597	0.560	0.210	0.592	0.558	0.562	0.210
Precision (offensive language identification)								
Not Offensive	0.802	0.755	0.764	0.559	0.727	0.707	0.796	0.559
Offensive Untargeted	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Offensive Targeted Individual	0.640	0.625	0.588	0.0	0.641	0.838	0.718	0.0
Offensive Targeted Group	0.171	0.0	0.0	0.0	0.250	0.0	0.132	0.0
Offensive Targeted Others	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Other Language	0.703	0.700	0.709	0.0	0.730	0.716	0.682	0.0
Macro Average	0.680	0.348	0.344	0.093	0.391	0.377	0.388	0.093
Weighted Average	0.680	0.642	0.644	0.313	0.648	0.641	0.678	0.313
Recall (Sentiment Analysis)								
Positive	0.787	0.790	0.781	1.0	0.746	0.967	0.772	1.0
Negative	0.689	0.689	0.659	0.0	0.774	0.268	0.689	0.0
Mixed Feelings	0.0	0.0	0.0	0.0	0.016	0.0	0.0	0.0
Unknown State	0.237	0.287	0.322	0.0	0.325	0.188	0.375	0.0
Other Language	0.701	0.828	0.644	0.0	0.655	0.299	0.563	0.0
Macro Average	0.483	0.496	0.474	0.200	0.503	0.344	0.480	0.200
Weighted Average	0.626	0.636	0.615	0.459	0.632	0.560	0.618	0.459
Recall (offensive language identification)								
Not Offensive	0.828	0.857	0.853	1.0	0.882	0.870	0.794	1.0
Offensive Untargeted	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Offensive Targeted Individual	0.585	0.671	0.610	0.0	0.610	0.378	0.622	0.0
Offensive Targeted Group	0.159	0.0	0.0	0.0	0.023	0.0	0.159	0.0
Offensive Targeted Others	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Other Language	0.877	0.818	0.870	0.0	0.721	0.883	0.877	0.0
Macro Average	0.408	0.391	0.389	0.167	0.373	0.355	0.409	0.167
Weighted Average	0.724	0.728	0.729	0.559	0.716	0.716	0.709	0.559
F1-Score (Sentiment Analysis)								
Positive	0.732	0.727	0.719	0.629	0.729	0.688	0.723	0.629
Negative	0.683	0.687	0.655	0.0	0.674	0.393	0.659	0.0
Mixed Feelings	0.0	0.0	0.0	0.0	0.029	0.0	0.0	0.0
Neutral	0.322	0.280	0.359	0.0	0.391	0.283	0.426	0.0
Other Language	0.542	0.605	0.536	0.0	0.591	0.419	0.603	0.0
Macro Average	0.456	0.460	0.454	0.126	0.483	0.357	0.462	0.126
Weighted Average	0.590	0.591	0.581	0.289	0.602	0.485	0.587	0.289
F1-Score (offensivelanguage identification)								
Not Offensive	0.815	0.803	0.806	0.717	0.797	0.780	0.795	0.717
Offensive Untargeted	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Offensive Targeted Individual	0.611	0.647	0.610	0.0	0.625	0.521	0.667	0.0
Offensive Targeted Group	0.165	0.0	0.0	0.0	0.042	0.0	0.144	0.0
Offensive Targeted Others	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Other Language	0.780	0.759	0.781	0.0	0.725	0.791	0.767	0.0
Macro Average	0.395	0.368	0.364	0.120	0.365	0.349	0.395	0.120
Weighted Average	0.700	0.683	0.683	0.401	0.672	0.662	0.690	0.401

Table 7 Various Hyper-parameters used for our experiments

Hyper-parameters	Characteristics
Shared layers	STL: Pooler output from pretrained LM, 1 Linear layer (hidden_size,128) with a ReLU activation function, output layer with (128, n_classes) for the tasks. MTL: same characteristics as STL
Task specific layers	MTL: 1 linear layer of shape (128, n_classes) depending on the number of labels in the datasets.
Output layers	STL: 1 Linear layer each for Sentiment task ([5] neurons each) and Offensive task [6], [5], [6] neurons for Kannada, Malayalam, and Tamil respectively. MTL: 2 Linear Layers for both tasks, ([5,6], [5,5], [5,6] output neurons for Kannada, Malayalam, and Tamil respectively)
Loss	[CE, KLD, HL, FL]
Epoch	5
Batch size	[16, 32, 64]
Optimizer	AdamW (Loshchilov and Hutter 2019)
Dropout	0.4 (Srivastava et al. 2014)
Loss weights	[1, 1] (All tasks are treated equally)

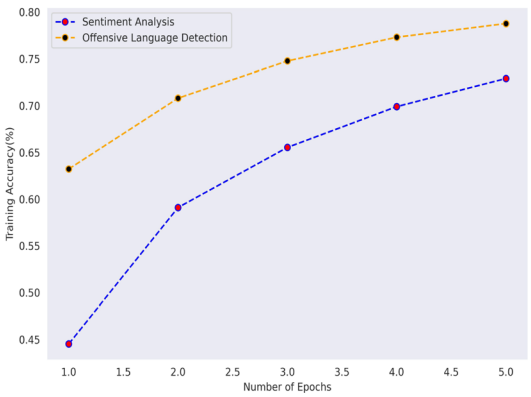
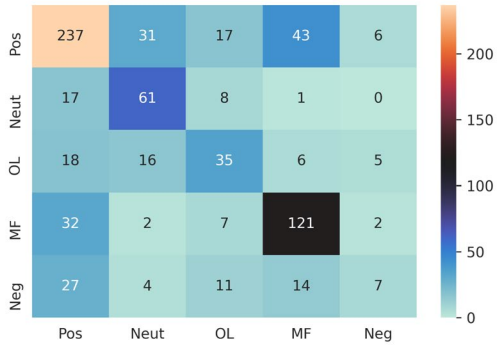
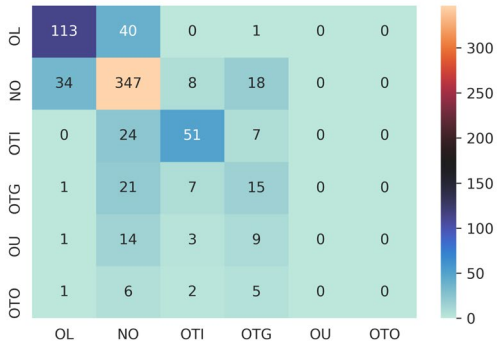


Fig. 9 Train accuracy during MTL



(a) Pos: Positive, Neut: Neutral, OL: Other Language, MF: Mixed Feelings, Neg: Negative



(b) NO: Not Offensive, OL: Other Language, OU: Offensive Untargeted, OTG: Offensive Targeted Group, OTI: Offensive Targeted Individual,OTO: Offensive Targeted Others

Fig. 10 Heatmap of confusion matrix for the best performing model on the Kannada Dataset

of performance on the other task was observed as it scored 0.590. It can be observed that only two of the classes scored 0.0 in MTL in contrast to three in STL. When both models were trained with KLD as its loss function, the performance was abysmal, and it was only able to classify a single class in the respective tasks (Positive and Not Offensive) among all. When trained on Hinge Loss and Focal Loss, we observe that they attend to the class imbalance. Hence, we observe that MTL frameworks tend to perform slightly better than when treated as individual tasks.

Figure 9 represents the training accuracy of sentiment analysis and offensive language identification in an MTL scenario. It can be observed from the graph that there is a

Table 8 MTL results on the Malayalam Dataset CE: Cross Entropy Loss, HL: Multi-class Hingle Loss, FL: Focal Loss, KLD: Kullback Leibler Divergence

	Hard parameter sharing				Soft parameter sharing			
Losses	Cross Entropy	Hinge Loss	Focal Loss	KLD	CE	HL	FL	KLD
Precision (Sentiment Analysis)								
Positive	0.726	0.448	0.453	0.426	0.450	0.620	0.660	0.426
Negative	0.453	0.0	0.0	0.0	0.0	0.0	0.521	0.0
Mixed Feelings	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Neutral	0.709	0.0	0.0	0.0	0.0	0.595	0.743	0.0
Other Language	0.745	0.653	0.613	0.0	0.616	0.724	0.758	0.0
Macro Average	0.527	0.220	0.213	0.085	0.213	0.388	0.536	0.085
Weighted Average	0.647	0.239	0.238	0.181	0.237	0.515	0.638	0.181
Precision (offensive language identification)								
Not Offensive	0.939	0.930	0.930	0.887	0.933	0.933	0.939	0.887
Offensive Untargeted	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Offensive Targeted Individual	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Offensive Targeted Group	0.0	0.0		0.0	0.0	0.0	0.0	0.0
Other Language	0.750	0.635	0.585	0.0	0.714	0.780	0.789	0.0
Macro Average	0.338	0.313	0.303	0.177	0.330	0.343	0.345	0.177
Weighted Average	0.886	0.870	0.866	0.787	0.879	0.883	0.888	0.787
Recall (Sentiment Analysis)								
Positive	0.800	0.972	0.965	1.000	0.965	0.801	0.871	1.000
Negative	0.496	0.0	0.0	0.0	0.0	0.0	0.370	0.0
Mixed Feelings	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Neutral	0.724	0.0	0.0	0.0	0.0	0.670	0.653	0.0
Other Language	0.777	0.681	0.777	0.0	0.734	0.755	0.734	0.0
Macro Average	0.559	0.331	0.348	0.200	0.340	0.445	0.526	0.200
Weighted Average	0.690	0.464	0.468	0.426	0.465	0.619	0.681	0.426
Recall (offensive language identification)								
Not Offensive	0.979	0.969	0.961	1.000	0.977	0.984	0.983	1.000
Offensive Untargeted	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Offensive Targeted Individual	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Offensive Targeted Group	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Other Language	0.800	0.678	0.689	0.0	0.722	0.711	0.789	0.0
Macro Average	0.356	0.329	0.330	0.200	0.340	0.339	0.354	0.200
Weighted Average	0.925	0.908	0.901	0.887	0.918	0.923	0.928	0.887
F1-Score (Sentiment Analysis)								
Positive	0.761	0.614	0.617	0.597	0.614	0.699	0.751	0.597
Negative	0.473	0.0	0.0	0.0	0.0	0.0	0.433	0.0
Mixed Feelings	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Neutral	0.716	0.0	0.0	0.0	0.0	0.630	0.695	0.0
Other Language	0.760	0.667	0.685	0.0	0.670	0.740	0.746	0.0
Macro Average	0.542	0.256	0.260	0.119	0.257	0.414	0.525	0.119
Weighted Average	0.668	0.310	0.313	0.254	0.311	0.561	0.651	0.254
F1-Score (offensive language identification)								
Not Offensive	0.959	0.949	0.945	0.940	0.955	0.958	0.960	0.940
Offensive Untargeted	0.0	0.0	0.0	0.0	0.0	0.0	0.00.350	0.0
Offensive Targeted Individual	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Offensive Targeted Group	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Other Language	0.774	0.656	0.633	0.0	0.718	0.744	0.789	0.0
Macro Average	0.347	0.321	0.316	0.188	0.335	0.340	0.350	0.188
Weighted Average	0.905	0.888	0.883	0.834	0.898	0.902	0.908	0.834

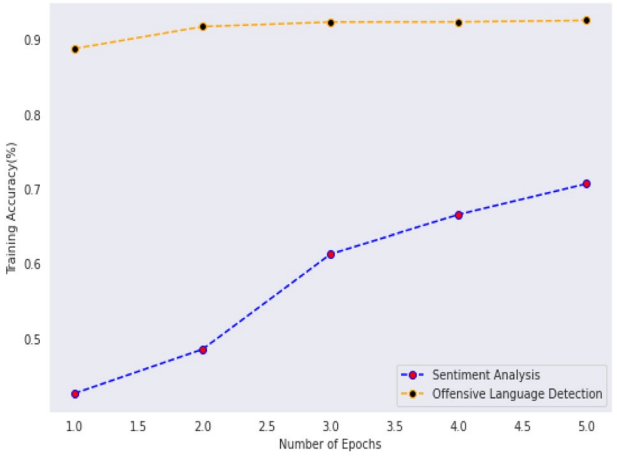
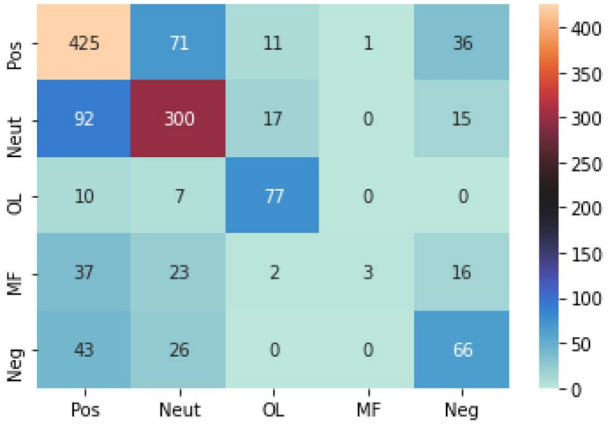
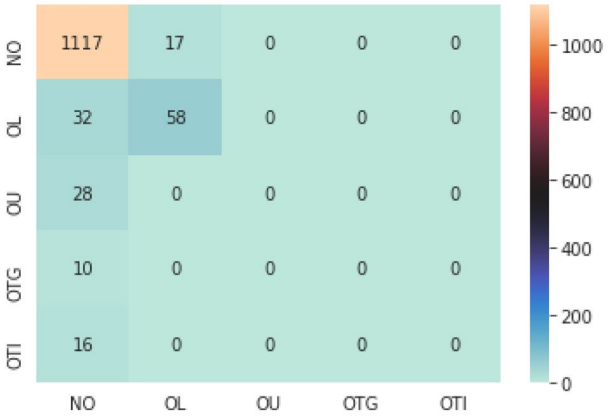


Fig. 11 Train Accuracy during MTL of mBERT on the Malayalam Dataset



(a) Pos: Positive, Neut: Neutral, OL: Other Language, MF: Mixed Feelings, Neg: Negative



(b) NO: Not Offensive, OL: Other Language, OU: Offensive Untargeted, OTG: Offensive Targeted Group, OTI: Offensive Targeted Individual

Fig. 12 Heatmap of confusion matrix for the best performing model on the Malayalam Dataset

bigger jump for sentiment analysis after the first epoch in contrast to offensive language identification, where there is a steady increase in the accuracy. Figure 10a and b exhibit the confusion matrices of the best performing model for the respective tasks. For sentiment analysis, we observe that a lot of labels are being misclassified. The trend of misclassification can also be observed in offensive language identification, where most of the samples are being misclassified into four classes (instead of 6), with the absence of samples being classified into OTO and OU.The samples of OTO and OU are being misclassified into other classes, which is mainly due to the low support of these classes in the test set (Table 7).

5.4.2 Malayalam

Tables 4 and 8 illustrate the classification report of the models on STL, and MTL respectively. It can be observed that among STL, fine-tuning mBERT with CE gave a weighted F1-Score of 0.635 for sentiment analysis and 0.902 for offensive language identification. DistilBERT outperforms BERT on offensive language identification, however, it scores poorly on sentiment analysis. DistilBERT reduces the occurrence of misclassification, as it is able to classify four out of the five classes separately, unlike mBERT, which correctly classifies two out of five classes in offensive language identification. Even though XLM has higher class-wise F1-Scores for both tasks, when trained on an MTL setting, it performed very poorly. This could be due to the pretraining strategy of XLM, as it was only pretrained on Wikipedia, which does not improve the performance on low-resource languages. Added to that, acquiring parallel data for TLM during pretraining could be very challenging (Grégoire and Langlais 2018). Several classes have scored 0.0 for class-wise weighted F1-Scores due to less support on the test set (Class Imbalance). Since mBERT is the strongest model among all, we experiment it for MTL.

We observe that mBERT fine-tuned on hard parameter sharing strategy with CE as its loss, achieved the highest score of 0.668 (+3.3% from STL) in sentiment analysis, while its secondary task scored 0.905 (+0.03%). Despite not being the best weighted F1-Score, it outperformed the STL result of mBERT. We observe that when mBERT is trained using soft parameter sharing with Focal Loss being its loss function, it attains the highest weighted F1-Score for offensive language identification (+0.6%), while scoring 0.651 (+1.6%) on sentiment analysis. We also observe that KLD's performance is appalling and could be due to the perseverance of the loss function. KLD is likely to treat the following multi-class classification problem as a regression

Table 9 MTL on the tamil dataset

	Hard parameter sharing				Soft parameter sharing			
Losses	Cross Entropy	Hinge Loss	Focal Loss	KLD	CE	HL	FL	KLD
Precision (Sentiment Analysis)								
Positive	0.721	0.703	0.729	0.560	0.710	0.686	0.717	0.676
Negative	0.446	0.467	0.449	0.500	0.430	0.444	0.428	0.505
Mixed Feelings	0.443	0.396	0.389	0.428	0.370	0.419	0.292	0.307
Neutral	0.501	0.523	0.464	0.0	0.540	0.599	0.511	0.540
Other Language	0.669	0.695	0.667	0.0	0.663	0.721	0.570	0.543
Macro Average	0.556	0.557	0.537	0.112	0.543	0.583	0.504	0.514
Weighted Average	0.619	0.612	0.613	0.313	0.610	0.614	0.596	0.587
Precision(offensive language identification)								
Not Offensive	0.844	0.805	0.843	0.726	0.862	0.821	0.870	0.839
Offensive Untargeted	0.408	0.4310	0.370	0.0	0.392	0.438	0.395	0.368
Offensive Targeted Individual	0.341	0.462	0.335	0.0	0.371	0.418	0.318	0.349
Offensive Targeted Group	0.353	0.373	0.317	0.0	0.410	0.420	0.327	0.323
Offensive Targeted Others	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Other Language	0.823	0.834	0.828	0.0	0.761	0.856	0.850	0.830
Macro Average	0.461	0.485	0.449	0.121	0.466	0.492	0.460	0.451
Weighted Average	0.728	0.713	0.721	0.527	0.744	0.726	0.744	0.719
Recall (Sentiment Analysis)								
Positive	0.844	0.867	0.830	1.0	0.857	0.928	0.831	1.0
Negative	0.461	0.433	0.465	0	0.429	0.278	0.342	0
Mixed Feelings	0.168	0.155	0.155	0	0.168	0.159	0.241	0
Unknown State	0.420	0.372	0.433	0	0.366	0.289	0.356	0
Other Language	0.568	0.568	0.587	0	0.601	0.559	0.615	0
Macro Average	0.492	0.479	0.494	0.200	0.484	0.443	0.477	0.200
Weighted Average	0.643	0.643	0.637	0.560	0.639	0.645	0.620	0.560
Recall (offensive language identification)								
Not Offensive	0.927	0.959	0.919	1.000	0.909	0.945	0.881	1.000
Offensive Untargeted	0.443	0.357	0.414	0	0.485	0.375	0.435	0
Offensive Targeted Individual	0.196	0.114	0.212	0	0.269	0.177	0.323	0
Offensive Targeted Group	0.200	0.102	0.174	0	0.252	0.190	0.325	0
Offensive Targeted Others	0.0	0.0	0.0	0	0.0	0.945	0.0	0.0
Other Language	0.730	0.708	0.730	0	0.787	0.736	0.730	0
Macro Average	0.416	0.373	0.408	0.167	0.450	0.404	0.449	0.167
Weighted Average	0.766	0.769	0.757	0.726	0.767	0.772	0.750	0.726
F1-Score (Sentiment Analysis)								
Positive	0.778	0.777	0.777	0.718	0.777	0.778	0.770	0.718
Negative	0.454	0.449	0.457	0.0	0.430	0.357	0.380	0
Mixed Feelings	0.243	0.223	0.222	0.0	0.231	0.232	0.264	0
Neutral	0.457	0.435	0.450	0.0	0.436	0.390	0.420	0
Other Language	0.614	0.625	0.616	0.0	0.631	0.630	0.591	0
Macro Average	0.509	0.502	0.504	0.144	0.501	0.477	0.485	0.144
Weighted Average	0.621	0.614	0.625	0.402	0.614	0.598	0.603	0.402
F1-Score (offensive language identification)								
Not Offensive	0.883	0.875	0.879	0.841	0.885	0.879	0.876	0.841
Offensive Untargeted	0.425	0.393	0.390	0	0.434	0.404	0.414	0
Offensive Targeted Individual	0.249	0.183	0.260	0	0.312	0.249	0.320	0
Offensive Targeted Group	0.255	0.160	0.225	0	0.312	0.262	0.326	0
Offensive Targeted Others	0.0	0.0	0.000	0	0.0	0.0	0	0
Other Language	0.774	0.766	0.776	0	0.773	0.792	0.785	0
Macro Average	0.431	0.396	0.422	0.140	0.453	0.431	0.453	0.140
Weighted Average	0.742	0.729	0.745	0.611	0.753	0.739	0.746	0.611

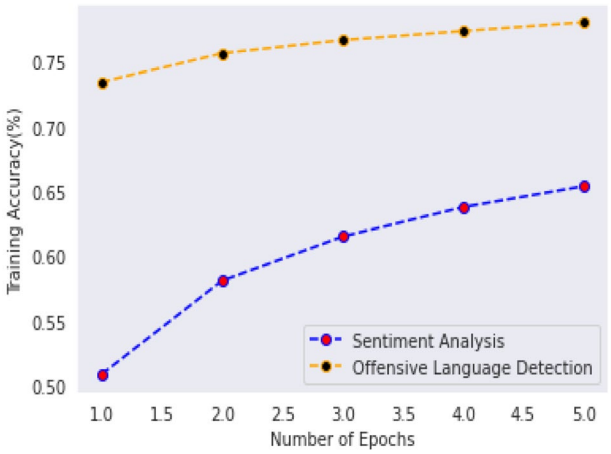
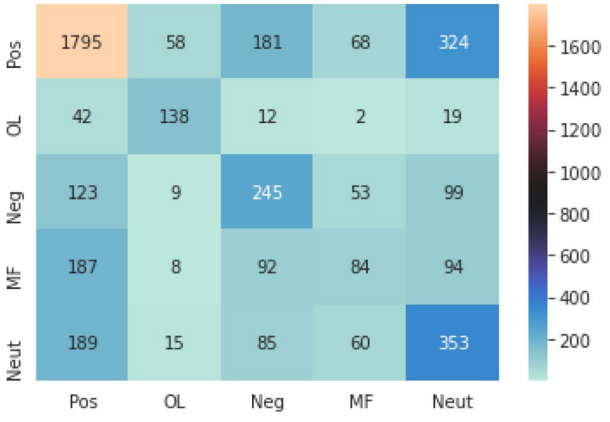
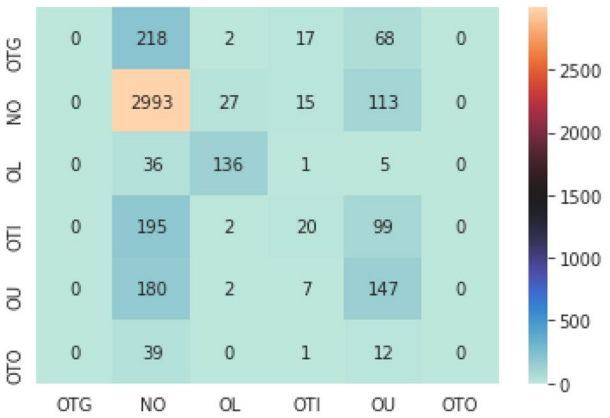


Fig. 13 Train Accuracy during MTL of DistilmBERT on the Tamil Dataset



(a) Pos: Positive, Neut: Neutral, OL: Other Language, MF: Mixed Feelings, Neg: Negative



(b) NO: Not Offensive, OL: Other Language, OU: Offensive Untargeted, OTG: Offensive Targeted Group, OTI: Offensive Targeted Individual

Fig. 14 Heatmap of confusion matrix for the best performing model on the Tamil Dataset

problem, hence predicting a single class that has the highest support. However, it has been previously used for classification problems (Nakov et al. 2016), but in our case, the loss performs poorer than the performance of the algorithms that serve as the baseline (Chakravarthi et al. 2020). It is to be noted that the increase in performance is more for sentiment analysis in contrast to offensive language identification. The time required to train, along with the memory constraints, are among the reasons why we opt for MTL.

Figure 11 represents the training accuracy of both tasks in the best performing model. After the first epoch, we see that there is a steady increase in both of the tasks. However, after the third epoch, we do not see any improvement in the performance of both of the tasks. Figure 12 reports the confusion matrices of the best performing models (in MTL). For sentiment analysis, we observe that most of the labels are misclassified for the samples of Mixed Feelings, which is mainly due to the low support in the training and test set as observed in Fig. 12a. For offensive language identification, we observe that the model tends to classify all of the samples in the test set into 2 classes, NO, and OL. As stated previously, low support of the other classes during training is why the model misclassifies its labels.

5.4.3 Tamil

Table 2 gives an insight into the support of the classes on the test set. We observe that the support of Offensive Targeted Others is quite low in contrast to Not Offensive (52:3148). From Table 5, DistilmBERT is the best performing model among all models. In spite of being a smaller, distilled form of BERT, it performs better than the parent model. Even though it was suggested that DistilBERT retains 97% of BERT's accuracy, it has performed better than BERT previously (Maslej-Krešň'akov'a et al. 2020), mainly due to the custom triple loss function and fewer parameters (Lan et al. 2019). The weighted average of F1-Score for sentiment analysis is 0.614 and 0.743 for offensive language identification, all trained with CE. Even though BERT achieves competitive performance, three out of six classes have a weighted average F1-Score of 0.0, due to misclassification. Hence, we train DistilmBERT on MTL. We observe that XLM has also performed better than BERT and XLM-RoBERTa.

It can be observed from Table 9 that DistilmBERT, when trained using hard parameter sharing strategy on CE loss yields the best results for offensive language identification with a weighted F1-Score of 0.753 (+1.0%), and 0.614 on sentiment analysis (+/-0.0%). However, the best score for offensive language identification was achieved by training DistilmBERT on soft parameter sharing strategy with Focal Loss. It achieved a score of 0.625 (+1.1%) on sentiment

analysis and 0.745 (+0.2%) on offensive language identification. It is to be noted that training on both CE and FL achieves better results than the best scores set forth by the STL model. The Hinge Loss function helps achieve better results for precision and recall. During training, the accuracy curve of sentiment analysis is increasing at a greater rate in contrast to offensive language identification, as displayed in Fig. 13. Figure 14 display the class-wise predictions by DistilmBERT trained on hard parameter sharing with CE. Most of the samples of NO class are predicted correctly, with the majority of the rest being mislabeled as OU. Due to low support in the training and test set, no samples of OTO have been predicted correctly. Most of the classes are incorrectly predicted as OU, which is mainly due to the close interconnection among the classes. In sentiment analysis, it can be observed that a significant number of samples of *Positive* and *Negative* classes have been predicted as *Neutral*. This is also observed in *Mixed Feeling* (MF), where most of the samples have been misclassified as *Positive* and *Negative*. The misclassification is mainly due to the nature of the class, as *Mixed Feelings (MF)* can be either of the two classes.

6 Conclusion

Despite of the rising popularity of social media, the lack of code-mixed data in Dravidian languages has motivated us to develop MTL frameworks for sentiment analysis and offensive language identification in three code-mixed Dravidian corpora, namely, Kannada, Malayalam, and Tamil. The proposed approach of fine-tuning multilingual BERT to a hard parameter sharing with cross entropy loss yields the best performance for both of the tasks in Kannada and Malayalam, achieving competitive scores in contrast to the performance when its counterparts are treated as separate tasks. For Tamil, our approach of fine-tuning multilingual distilBERT in soft parameter sharing with cross entropy loss scores better than the other models, mainly due to its triple loss function employed during pretraining. The performance of these models highlights the advantages of using an MTL model to attend to two tasks at a time, mainly reducing the time required to train the models while additionally reducing the space complexities required to train them separately. For future work, we intend to use uncertainty weighting to calculate the impact of one loss function on the other during MTL.

Funding Open Access funding provided by the IReL Consortium

References

Anagha M, Kumar RR, Sreetha K, Raj PR (2015) Fuzzy logic based hybrid approach for sentiment analysis of malayalam movie reviews. In: 2015 IEEE International conference on signal processing, informatics, communication and energy systems (SPICES), IEEE, pp 1–4

Appidi AR, Srirangam VK, Suhas D, Shrivastava M (2020) Creation of corpus and analysis in code-mixed Kannada-English Twitter data for emotion prediction. In: Proceedings of the 28th international conference on computational linguistics, International Committee on Computational Linguistics, Barcelona, Spain (Online), pp 6703–6709. https://www.aclweb.org/anthology/2020.coling-main.587

del Arco FMP, Molina-González MD, Ureña-López LA, Martín-Valdivia MT (2021) Comparing pre-trained language models for spanish hate speech detection. Expert Syst Appl 166(114):120. https://doi.org/10.1016/j.eswa.2020.114120

Asperti A, Trentin M (2020) Balancing reconstruction error and kullback-leibler divergence in variational autoencoders. IEEE Access 8:199,440–199,448. https://doi.org/10.1109/ACCESS.2020.3034828

Bali K, Sharma J, Choudhury M, Vyas Y (2014) "I am borrowing ya mixing ?" an analysis of English-Hindi code mixing in Facebook. In: Proceedings of the First workshop on computational approaches to code switching, Association for Computational Linguistics, Doha, Qatar, pp 116–126. https://doi.org/10.3115/v1/W14-3914, https://www.aclweb.org/anthology/W14-3914

Banerjee S, Chakravarthi BR, McCrae JP (2020) Comparison of pretrained embeddings to identify hate speech in indian code-mixed text. In: 2020 2Nd international conference on advances in computing, communication control and networking (ICACCCN), IEEE, pp 21–25

Barman U, Das A, Wagner J, Foster J (2014) Code mixing: A challenge for language identification in the language of social media. In: Proceedings of the First workshop on computational approaches to code switching, Association for Computational Linguistics, Doha, Qatar, pp 13–23. https://doi.org/10.3115/v1/W14-3902, https://www.aclweb.org/anthology/W14-3902

Bhat S (2012) Morpheme segmentation for Kannada standing on the shoulder of giants. In: Proceedings of the 3rd workshop on south and southeast asian natural language processing, The COLING 2012 Organizing Committee, Mumbai, India, pp 79–94. https://www.aclweb.org/anthology/W12-5007

Bisong E (2019) Google Colaboratory. Apress, Berkeley, CA, pp 59–64. https://doi.org/10.1007/978-1-4842-4470-8_7

Brownlee J (2019) How to calculate the kl divergence for machine learning. Available at http://machinelearningmastery.com/divergence-between-probability-distributions/l

Caruana R (1997) Multitask learning. Mach Learn 28(1):41–75. https://doi.org/10.1023/A:1007379606734

Chakravarthi BR (2020) HopeEDI: A multilingual hope speech detection dataset for equality, diversity, and inclusion. In:

Proceedings of the Third workshop on computational modeling of people's opinions, personality, and emotion's in social media, Association for Computational Linguistics, Barcelona, Spain (Online), pp 41–53. https://www.aclweb.org/anthology/2020.peoples-1.5

Chakravarthi BR, Jose N, Suryawanshi S, Sherly E, McCrae JP (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), European Language Resources association, Marseille, France, pp 177–184. https://www.aclweb.org/anthology/2020.sltu-1.25

Chakravarthi BR, Muralidaran V, Priyadharshini R, McCrae JP (2020) Corpus creation for sentiment analysis in code-mixed Tamil-English text. 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), European Language Resources association, Marseille, France, pp 202–210. https://www.aclweb.org/anthology/2020.sltu-1.28

Chakravarthi BR, Priyadharshini R, Jose NMAK, Mandl T, Kumaresan PK, Ponnusamy RVH, McCrae John Philip Sherly E (2021) Findings of the shared task on Offensive Language Identification in Tamil, Malayalam, and Kannada. In: Proceedings of the First workshop on speech and language technologies for dravidian languages. Association for Computational Linguistics

Chakravarthi BR, Priyadharshini R, Muralidaran V, Jose N, Suryawanshi S, Sherly E, McCrae JP (2021) Dravidiancodemix: Sentiment analysis and offensive language identification dataset for dravidian languages in code-mixed text. Language Resources and Evaluation

Changpinyo S, Hu H, Sha F (2018) Multi-task learning for sequence tagging: An empirical study. In: Proceedings of the 27th international conference on computational linguistics, Association for Computational Linguistics, Santa Fe, New Mexico, USA, pp 2965–2977. https://www.aclweb.org/anthology/C18-1251

Chhablani G, Bhartia Y, Sharma A, Pandey H, Suthaharan S (2021) Nlrg at semeval-2021 task 5: Toxic spans detection leveraging bert-based token classification and span prediction techniques

Clarke I, Grieve J (2017) Dimensions of abusive language on Twitter. In: Proceedings of the First workshop on abusive language online, Association for Computational Linguistics, Vancouver, BC, Canada, pp 1–10. https://doi.org/10.18653/v1/W17-3001, https://www.aclweb.org/anthology/W17-3001

Conneau A, Khandelwal K, Goyal N, Chaudhary V, Wenzek G, Guzmán F, Grave E, Ott M, Zettlemoyer L, Stoyanov V (2020) Unsupervised cross-lingual representation learning at scale. In: Proceedings of the 58th annual meeting of the association for computational linguistics, Association for Computational Linguistics, Online, pp 8440–8451. https://doi.org/10.18653/v1/2020.acl-main.747, https://www.aclweb.org/anthology/2020.acl-main.747

Crawshaw M (2020) Multi-task learning with deep neural networks: A survey. arXiv:2009.09796

Dadvar M, Trieschnigg D, Ordelman R, de Jong F (2013) Improving cyberbullying detection with user context. In: Serdyukov P, Braslavski P, Kuznetsov SO, Kamps J., Rüger S, Agichtein E, Segalovich I, Yilmaz E (eds) Advances in information retrieval. Springer, Berlin, Heidelberg, pp 693–696

Dai Z, Yang Z, Yang Y, Carbonell J, Le Q, Salakhutdinov R (2019) Transformer-XL: Attentive language models beyond a fixed-length context. In: Proceedings of the 57th annual meeting of the association for computational linguistics, Association for Computational Linguistics, Florence, Italy, pp 2978–2988. https://doi.org/10.18653/v1/P19-1285, https://www.aclweb.org/anthology/P19-1285

Dai Z, Yang Z, Yang Y, Carbonell JG, Le QV, Salakhutdinov R (2019) Transformer-xl: Attentive language models beyond a fixed-length context. arXiv:abs/1901.02860. 1901.02860

Das A, Bandyopadhyay S (2010) SentiWordNet for Indian languages. In: Proceedings of the Eighth workshop on asian language resouces, Coling 2010 Organizing Committee, Beijing, China, pp 56–63. https://www.aclweb.org/anthology/W10-3208

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, NLP Association of India, Goa, India, pp 378–387. https://www.aclweb.org/anthology/W14-5152

De Boer PT, Kroese DP, Mannor S, Rubinstein RY (2005) A tutorial on the cross-entropy method. Annals of Operations Research 134(1):19–67

Deng J, Dong W, Socher R, Li L (2009) Kai Li, Li fei-fei: imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on computer vision and pattern recognition, pp 248–255. https://doi.org/10.1109/CVPR.2009.5206848

Devlin J, Chang MW, Lee K, Toutanova K (2019) 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), Association for Computational Linguistics, Minneapolis, Minnesota, pp 4171–4186. https://doi.org/10.18653/v1/N19-1423, https://www.aclweb.org/anthology/N19-1423

Djuric N, Zhou J, Morris R, Grbovic M, Radosavljevic V, Bhamidipati N (2015) Hate speech detection with comment embeddings. In: Proceedings of the 24th international conference on world wide web, pp 29–30

Dobrescu A, Giuffrida MV, Tsaftaris SA (2020) Doing more with less: a multitask deep learning approach in plant phenotyping. Frontiers in plant science 11

Duong L, Cohn T, Bird S, Cook P (2015) Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (Volume 2: Short Papers), Association for Computational Linguistics, Beijing, China, pp 845–850. https://doi.org/10.3115/v1/P15-2139, https://www.aclweb.org/anthology/P15-2139

Eigen D, Fergus R (2015) Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In: 2015 IEEE International conference on computer vision (ICCV), pp 2650–2658

El Boukkouri H, Ferret O, Lavergne T, Noji H, Zweigenbaum P, Tsujii J (2020) CharacterBERT: Reconciling ELMo and BERT for word-level open-vocabulary representations from characters. In: Proceedings of the 28th international conference on computational linguistics, International Committee on Computational Linguistics, Barcelona, Spain (Online), pp 6903–6915. https://www.aclweb.org/anthology/2020.coling-main.609

Ghanghor NK, Krishnamurthy P, Thavareesan S, Priyadharshini R (2021) Chakravarthi, B.R.: IIITK@dravidianlangtech-EACL2021: Offensive Language Identification and Meme Classification in Tamil, Malayalam and Kannada. In: Proceedings of the First workshop on speech and language technologies for dravidian languages. Association for Computational Linguistics, Online

Grégoire F, Langlais P (2018) Extracting parallel sentences with bidirectional recurrent neural networks to improve machine translation. In: Proceedings of the 27th international conference on computational linguistics, Association for Computational Linguistics, Santa Fe, New Mexico, USA, pp 1442–1453. https://www.aclweb.org/anthology/C18-1122

Hande A, Priyadharshini R, Chakravarthi BR (2020) KanCMD: Kannada CodeMixed dataset for sentiment analysis and offensive language detection. In: Proceedings of the Third workshop on computational modeling of people's opinions, personality, and emotion's in social media, Association for Computational Linguistics, Barcelona, Spain (Online), pp 54–63. https://www.aclweb.org/anthology/2020.peoples-1.6

Jain K, Deshpande A, Shridhar K, Laumann F, Dash A (2020) Indictransformers: An analysis of transformer language models for indian languages

Jin N, Wu J, Ma X, Yan K, Mo Y (2020) Multi-task learning model based on multi-scale cnn and lstm for sentiment classification. IEEE Access 8:77,060–77,072. https://doi.org/10.1109/ACCESS.2020.2989428

Kakwani D, Kunchukuttan A, Golla SNCG, Bhattacharyya A, Khapra MM, Kumar P (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, Association for Computational Linguistics, Online, pp 4948–4961. https://doi.org/10.18653/v1/2020.findings-emnlp.445, https://www.aclweb.org/anthology/2020.findings-emnlp.445

Khanuja S, Bansal D, Mehtani S, Khosla S, Dey A, Gopalan B, Margam DK, Aggarwal P, Nagipogu RT, Dave S et al (2021) Muril:, Multilingual representations for indian languages. arXiv:2103.10730

Kingma DP, Ba J (2014) Adam:, A method for stochastic optimization. arXiv:1412.6980

Kuchaiev O, Ginsburg B (2017) Factorization tricks for LSTM networks. arXiv:abs/1703.10722

Kudo T (2018) Subword regularization: Improving neural network translation models with multiple subword candidates. In: Proceedings of the 56th annual meeting of the association for computational linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Melbourne, Australia, pp 66–75. https://doi.org/10.18653/v1/P18-1007, https://www.aclweb.org/anthology/P18-1007

Kudo T, Richardson J (2018) SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In: Proceedings of the 2018 conference on empirical methods in natural language processing: system demonstrations, Association for Computational Linguistics, Brussels, Belgium, pp 66–71. https://doi.org/10.18653/v1/D18-2012, https://www.aclweb.org/anthology/D18-2012

Kullback S, Leibler RA (1951) On information and sufficiency. Ann Math Statist 22(1):79–86. https://doi.org/10.1214/aoms/1177729694

Kumar R, Ojha AK, Lahiri B, Zampieri M, Malmasi S, Murdock V, Kadar D (eds) (2020) Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying. European Language Resources Association (ELRA), Marseille, France. https://www.aclweb.org/anthology/2020.trac-1.0

Kumar SS, Kumar MA, Soman K, Poornachandran P (2020) Dynamic mode-based feature with random mapping for sentiment analysis. In: Intelligent systems, technologies and applications, Springer, pp 1–15

Lan Z, Chen M, Goodman S, Gimpel K, Sharma P, Soricut R (2019) ALBERT: A lite BERT for self-supervised learning of language representations. arXiv:abs/1909.11942

Li N, Chow CY, Zhang JD (2020) Seml: a semi-supervised multi-task learning framework for aspect-based sentiment analysis. IEEE Access 8:189,287–189,297. https://doi.org/10.1109/ACCESS.2020.3031665

Lin TY, Goyal P, Girshick R, He K, Dollar P (2017) Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV)

Liu V, Curran JR (2006) Web text corpus for natural language processing. In: 11th Conference of the European chapter of the association for computational linguistics. Association for Computational Linguistics, Trento, Italy. https://www.aclweb.org/anthology/E06-1030

Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, Levy O, Lewis M, Zettlemoyer L, Stoyanov V (2019) Roberta: A robustly optimized BERT pretraining approach. arXiv:abs/1907.11692

Loshchilov I, Hutter F (2019) Decoupled weight decay regularization. In: International conference on learning representations. https://openreview.net/forum?id=Bkg6RiCqY7

Ma Y, Zhao L, Hao J (2020) XLP at SemEval-2020 task 9: Cross-lingual models with focal loss for sentiment analysis of code-mixing language. In: Proceedings of the Fourteenth workshop on semantic evaluation, International Committee for Computational Linguistics, Barcelona (online), pp 975–980. https://www.aclweb.org/anthology/2020.semeval-1.126

Mandl T, Modha S, Kumar MA, Chakravarthi BR (2020) Overview of the hasoc track at fire 2020: Hate speech and offensive language identification in tamil, malayalam, hindi, english and german. In: Forum for information retrieval evaluation, FIRE 2020. https://doi.org/10.1145/3441501.3441517. Association for Computing Machinery, New York, NY, USA, pp 29–32

Maninis K, Radosavovic I (2019) Kokkinos, I.: Attentive single-tasking of multiple tasks. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp 1851–1860

Martínez Alonso H, Plank B (2017) When is multitask learning effective? semantic sequence prediction under varying data conditions. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers, Association for Computational Linguistics, Valencia, Spain, pp 44–53. https://www.aclweb.org/anthology/E17-1005

Maslej-Krešň'akov'a V, Sarnovsk'y M, Butka P, Machov'a K (2020) Comparison of deep learning models and various text pre-processing techniques for the toxic comments classification. Appl Sci 10 (23):8631

Mou L, Zhu XX (2018) Vehicle instance segmentation from aerial image and video using a multitask learning residual fully convolutional network. IEEE Trans Geosci Remote Sens 56(11):6699–6711. https://doi.org/10.1109/TGRS.2018.2841808

Murphy KP (2012) Machine learning: a probabilistic perspective. The MIT Press, London, England

Nair DS, Jayan JP, Rajeev R, Sherly E (2015) Sentiment analysis of malayalam film review using machine learning techniques. In: 2015 International conference on advances in computing, communications and informatics (ICACCI), IEEE, pp 2381–2384

Nair DS, Jayan JP, Rajeev RR, Sherly E (2014) Sentima - sentiment extraction for malayalam. In: 2014 International conference on advances in computing, communications and informatics (ICACCI), pp 1719–1723. https://doi.org/10.1109/ICACCI.2014.6968548

Nakov P, Ritter A, Rosenthal S, Sebastiani F, Stoyanov V (2016) SemEval-2016 task 4: Sentiment analysis in Twitter. In: Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016), Association for Computational Linguistics, San Diego, California, pp 1–18. https://doi.org/10.18653/v1/S16-1001, https://www.aclweb.org/anthology/S16-1001

Nobata C, Tetreault J, Thomas A, Mehdad Y, Chang Y (2016) Abusive language detection in online user content. In: Proceedings of the 25th international conference on world wide web, pp 145–153

Ouyang X, Xu S, Zhang C, Zhou P, Yang Y, Liu G, Li X (2019) A 3d-cnn and lstm based multi-task learning architecture for action recognition. IEEE Access 7:40,757–40,770. https://doi.org/10.1109/ACCESS.2019.2906654

Padmamala R, Prema V (2017) Sentiment analysis of online tamil contents using recursive neural network models approach for tamil language. In: 2017 IEEE International conference on smart technologies and management for computing, communication, controls, energy and materials (ICSTM), pp 28–31. https://doi.org/10.1109/ICSTM.2017.8089122

Pan SJ, Yang Q (2009) A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering 22(10):1345–1359

Pang B, Lee L (2008) Opinion mining and sentiment analysis. Found Trends Inf Retr 2 (1–2):1–135. https://doi.org/10.1561/1500000011

Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L, Desmaison A, Köpf A., Yang E, DeVito Z, Raison M, Tejani A, Chilamkurthy S, Steiner B, Fang L, Bai J, Chintala S (2019) Pytorch: an imperative style, high-performance deep learning library. In: NeurIPS

Patra BG, Das D, Das A, Prasath R (2015) Shared task on sentiment analysis in indian languages (sail) tweets - an overview. In: Prasath R, Vuppala AK, Kathirvalavakumar T (eds) Mining intelligence and knowledge exploration. Springer International Publishing, Cham, pp 650–655

Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Louppe G, Prettenhofer P, Weiss R, Weiss RJ, VanderPlas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: Machine learning in python. J Mach Learn Res 12:2825–2830

Peng Y, Chen Q, Lu Z (2020) An empirical study of multi-task learning on BERT for biomedical text mining. In: Proceedings of the 19th SIGBioMed workshop on biomedical language processing, Association for Computational Linguistics, Online, pp 205–214. https://doi.org/10.18653/v1/2020.bionlp-1.22, https://www.aclweb.org/anthology/2020.bionlp-1.22

Phani S, Lahiri S, Biswas A (2016) Sentiment analysis of tweets in three Indian languages. In: Proceedings of the 6th workshop on south and southeast asian natural language processing (WSSANLP2016), The COLING 2016 Organizing Committee, Osaka, Japan, pp 93–102. https://www.aclweb.org/anthology/W16-3710

Pires T, Schlinger E, Garrette D (2019) How multilingual is multilingual BERT? In: Proceedings of the 57th annual meeting of the association for computational linguistics, Association for Computational Linguistics, Florence, Italy, pp 4996–5001. https://doi.org/10.18653/v1/P19-1493, https://www.aclweb.org/anthology/P19-1493

Prabhu S, Narayan U, Debnath ASS, Shrivastava M (2020) Detection and annotation of events in Kannada. In: 16th Joint ACL - ISO workshop on interoperable semantic annotation PROCEEDINGS, European Language Resources Association, Marseille, pp 88–93. https://www.aclweb.org/anthology/2020.isa-1.10

Prokhorov V, Shareghi E, Li Y, Pilehvar MT, Collier N (2019) On the importance of the Kullback-Leibler divergence term in variational autoencoders for text generation. In: Proceedings of the 3rd workshop on neural generation and translation, Association for Computational Linguistics, Hong Kong, pp 118–127. https://doi.org/10.18653/v1/D19-5612, https://www.aclweb.org/anthology/D19-5612

Puranik K, Hande A, Priyadharshini R, Thavareesan S, Chakravarthi BR (2021) IIITT@LT-EDI-EACL2021-Hope Speech detection: There is always hope in Transformers. In: Proceedings of the First workshop on language technology for equality, diversity and inclusion. Association for Computational Linguistics

Radford A (2018) Improving language understanding by generative pre-training

Rakhlin A (2016) MIT Online Methods in Machine Learning 6.883, Lecture Notes: Multiclass and multilabel problems. http://www.mit.edu/rakhlin/6.883/lectures/lecture05.pdf. Last visited on 2021/02/08

Ranasinghe T, Zampieri M (2021) Mudes: Multilingual detection of offensive spans

Rani P, Suryawanshi S, Goswami K, Chakravarthi BR, Fransen T, McCrae JP (2020) A comparative study of different state-of-the-art hate speech detection methods in hindi-english code-mixed data. In: Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying, pp 42–48

Razavi AH, Inkpen D, Uritsky S, Matwin S (2010) Offensive language detection using multi-level classification. In: Canadian conference on artificial intelligence, Springer, pp 16–27

Reddy S, Sharoff S (2011) Cross language POS taggers (and other tools) for Indian languages: An experiment with Kannada using Telugu resources. In: Proceedings of the Fifth international workshop on cross lingual information access, Asian Federation of Natural Language Processing, Chiang Mai, Thailand, pp 11–19. https://www.aclweb.org/anthology/W11-3603

Ruder S (2017) An overview of multi-task learning in deep neural networks. arXiv:abs/1706.05098

Sanh V, Debut L, Chaumond J, Wolf T (2019) Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv:abs/1910.01108

Schuster M, Nakajima K (2012) Japanese and korean voice search. In: 2012 IEEE International conference on acoustics, speech and signal processing (ICASSP), pp 5149–5152. https://doi.org/10.1109/ICASSP.2012.6289079

Se S, Vinayakumar R, Kumar MA, Soman K (2016) Predicting the sentimental reviews in tamil movie using machine learning algorithms. Indian J Sci Technol 9(45):1–5

Sennrich R, Haddow B, Birch A (2016) Neural machine translation of rare words with subword units. In: Proceedings of the 54th annual meeting of the association for computational linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Berlin, Germany, pp 1715–1725. https://doi.org/10.18653/v1/P16-1162, https://www.aclweb.org/anthology/P16-1162

Severyn A, Moschitti A, Uryupina O, Plank B, Filippova K (2014) Opinion mining on YouTube. In: Proceedings of the 52nd annual meeting of the association for computational linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Baltimore, Marylandm, pp 1252–1261. https://doi.org/10.3115/v1/P14-1118, https://www.aclweb.org/anthology/P14-1118

Shalev-Shwartz S, Ben-David S (2014) Understanding machine learning: from theory to algorithms. Cambridge University Press, New York

Shazeer N, Mirhoseini A, Maziarz K, Davis A, Le Q, Hinton G, Dean J (2017) Outrageously large neural networks: The sparsely-gated mixture-of-experts layer

Soumya S, Pramod K (2020) Sentiment analysis of malayalam tweets using machine learning techniques. ICT Express 6(4):300–305

Sowmya Lakshmi BS, Shambhavi BR (2017) An automatic language identification system for code-mixed english-kannada social media text. In: 2017 2Nd international conference on computational systems and information technology for sustainable solution (CSITSS), pp. 1–5

Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958

Tanwar A, Majumder P (2020) Translating morphologically rich indian languages under zero-resource conditions. ACM Trans. Asian Low-Resour. Lang. Inf. Process 19(6). https://doi.org/10.1145/3407912

Taylor WL (1953) "Cloze procedure": a new tool for measuring readability. Journalism & Mass Communication Quarterly 30:415–433

Thavareesan S, Mahesan S (2019) Sentiment analysis in tamil texts: a study on machine learning techniques and feature representation. In: 2019 14Th conference on industrial and information systems (ICIIS), pp 320–325. https://doi.org/10.1109/ICIIS47346.2019.9063341

Thilagavathi R, Krishnakumari K (2016) Tamil english language sentiment analysis system. International Journal of Engineering Research & Technology (IJERT) 4:114–118

Tian Y, Galery T, Dulcinati G, Molimpakis E, Sun C (2017) Facebook sentiment: Reactions and emojis. In: Proceedings of the Fifth international workshop on natural language processing for social media, Association for Computational Linguistics, Valencia, Spain, pp 11–16. https://doi.org/10.18653/v1/W17-1102, https://www.aclweb.org/anthology/W17-1102

Tontodimamma A, Nissi E, Sarra A, Fontanella L (2021) Thirty years of research into hate speech: topics of interest and their evolution. Scientometrics 126(1):157–179

Tula D, Potluri P, Ms S, Doddapaneni S, Sahu P, Sukumaran R, Patwa P (2021) Bitions@DravidianLangTech-EACL2021: Ensemble of multilingual language models with pseudo labeling for offence detection in Dravidian languages. In: Proceedings of the First workshop on speech and language technologies for dravidian languages, Association for Computational Linguistics, Kyiv, pp 291–299. https://www.aclweb.org/anthology/2021.dravidianlangtech-1.42

Uysal AK, Gunal S (2014) The impact of preprocessing on text classification. Inform Process Manage 50(1):104–112. https://doi.org/10.1016/j.ipm.2013.08.006

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need

Weston J, Watkins C (1999) Support vector machines for multi-class pattern recognition. In: ESANN

Wolf T, Debut L, Sanh V, Chaumond J, Delangue C, Moi A, Cistac P, Rault T, Louf R, Funtowicz M, Davison J, Shleifer S, von Platen P, Ma C, Jernite Y, Plu J, Xu C, Le Scao T, Gugger S, Drame M, Lhoest Q, Rush A (2020) Transformers: State-of-the-art natural language processing. In: Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations. https://doi.org/10.18653/v1/2020.emnlp-demos.6, https://www.aclweb.org/anthology/2020.emnlp-demos.6. Association for Computational Linguistics, Online, pp 38–45

Yang Y, Hospedales TM (2017) Trace norm regularised deep multi-task learning. arXiv:abs/1606.04038

Yang Z, Dai Z, Yang Y, Carbonell JG, Salakhutdinov R, Le QV (2019) Xlnet: Generalized autoregressive pretraining for language understanding. arXiv:abs/1906.08237

Yasaswini K, Puranik K, Hande A, Priyadharshini R, Thavareesan S, Chakravarthi BR (2021) IIITT@dravidianlangtech-EACL2021: Transfer learning for offensive language detection in dravidian languages. In: Proceedings of the First workshop on speech and language technologies for dravidian languages, Association for Computational Linguistics

Zampieri M, Malmasi S, Nakov P, Rosenthal S, Farra N, Kumar R (2019) SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval). In: Proceedings of the 13th international workshop on semantic evaluation, Association for Computational Linguistics, Minneapolis, Minnesota, USA, pp 75–86. https://doi.org/10.18653/v1/S19-2010, https://www.aclweb.org/anthology/S19-2010

Zampieri M, Nakov P, Rosenthal S, Atanasova P, Karadzhov G, Mubarak H, Derczynski L, Pitenis Z, Çöltekin Ç (2020) SemEval-2020 task 12: Multilingual offensive language identification in social media (OffensEval 2020). In: Proceedings of the Fourteenth workshop on semantic evaluation, International Committee for Computational Linguistics, Barcelona (online), pp 1425–1447. https://www.aclweb.org/anthology/2020.semeval-1.188

Zhai P, Tao Y, Chen H, Cai T, Li J (2020) Multi-task learning for lung nodule classification on chest ct. IEEE Access 8:180,317–180,327. https://doi.org/10.1109/ACCESS.2020.3027812

Zhang H, Sun S, Hu Y, Liu J, Guo Y (2020) Sentiment classification for chinese text based on interactive multitask learning. IEEE Access 8:129,626–129,635. https://doi.org/10.1109/ACCESS.2020.3007889

Zhang K, Wu L, Zhu Z, Deng J (2020) A multitask learning model for traffic flow and speed forecasting. IEEE Access 8:80,707–80,715. https://doi.org/10.1109/ACCESS.2020.2990958

Zhang Y, Yang Q (2018) A survey on multi-task learning

Zhang Z, Chen C, Dai G, Li WJ, Yeung DY (2014) Multicategory large margin classification methods: Hinge losses vs. coherence functions. Artif Intell 215:55–78. https://doi.org/10.1016/j.artint.2014.06.002