Span Extraction Aided Improved Code-mixed Sentiment Classification

Ramaneswaran S
Vellore Institute of Technology
s.ramaneswaran2000@gmail.com

Sean Benhur
PSG College of Arts & Science
seanbenhur@gmail.com

Sreyan Ghosh
University of Maryland, College Park
sreyang@umd.edu

Abstract

Sentiment classification is a fundamental NLP task of detecting the sentiment polarity of a given text. In this paper we show how solving sentiment span extraction as an auxiliary task can help improve final sentiment classification performance in a low-resource code-mixed setup. To be precise, we don't solve a simple multi-task learning objective, but rather design a unified transformer framework that exploits the bidirectional connection between the two tasks simultaneously. To facilitate research in this direction we release gold-standard human-annotated sentiment span extraction dataset for Tamil-english code-switched texts. Extensive experiments and strong baselines show that our proposed approach outperforms sentiment and span prediction by 1.27% and 2.78% respectively when compared to the best performing MTL baseline. We also establish the generalizability of our approach on the Twitter Sentiment Extraction dataset. We make our code and data publicly available on GitHub [1].

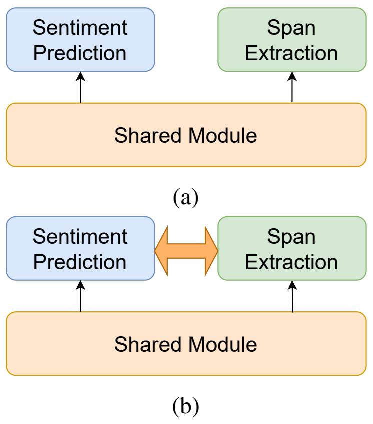

Figure 1: (a) Multi-task learning setup with parameter-sharing and joint learning of sentiment prediction and span extraction tasks (b) Our approach which establishes bi-directional connection to explicitly model the mutual interactions between the tasks

1 Introduction

With the rapid growth of social media networks and the democratization of internet technology, massive amounts of text-based user-generated content is being produced everyday. It is essential to understand the opinion and sentiment of users from these textual posts. In the past decade the NLP research community has made several advancements in the field of language based sentiment analysis. However most of these advances are in high-resource languages like English. In contrast there are limited resources for sentiment analysis for Indian languages.

In the context of the Indian sub-continent, the user-generated content on social media is unique because is not in any one particular language, rather a single utterance may consist of words, phrases and phonemes from multiple different languages. This phenomenon is code-mixing and is widely observed in multi-lingual communities such as India. Although recent advances have been made in developing sentiment analysis text corpora and methods for Indian languages such as Hindi and Bengali there has been little progress for truly low-resourced languages such as Tamil, a Dravidian language which is spoken by well over 70 million people worldwide. (Chakravarthi et al., 2020) is a seminal work on creating corpora for sentiment classification of Tamil-English code-mixed text.

While sentiment classification is well researched; sentiment span extraction on the other hand (Lai et al., 2020) is a rather new NLP task which involves the extraction of supporting phrases from text in the form of a sequence of contiguous words, which reflect the sentiment of the sentence. These support phrases can be used to further perform fine-grained analysis of the sentiment to understand the opinion and feelings of the user. Similar approach has also been applied to toxicity analysis from text (Ghosh and Kumar, 2021).

[1] https://github.com/ramaneswaran/code mixed_sentiment_span_extraction

In this paper we present our hypothesis that solving sentiment span extraction as an additional task can help the model learn better semantic representations of the text which in-turn will improve sentiment classification. We explore this hypothesis for code-mixed Tamil texts. Firstly we develop a novel Tamil-English code-mixed sentiment extraction dataset to support the task of sentiment span extraction. We obtain this dataset by extending the DravidianCodemix dataset (Chakravarthi et al., 2022) by adding gold-standard human-annotated sentiment span labels to it. To the best of our knowledge this is the first code-mixed sentiment span extraction dataset. The proposed dataset will facilitate further research in this direction and helps improve sentiment classification performance in a low-resource setting in a language spoken by millions around the globe where annotated data is scarce.

Secondly we experiment with various single-task learning and multi-task learning models to evaluate our hypothesis. Further inspired from (Qin et al., 2021) we explore a methodology based on transformer architecture which explicitly models the interactions between the two tasks of sentiment prediction and sentiment span extraction in a unified framework (Refer to Fig. 1b . Extensive experiments and ablation study establish the efficacy of this proposed approach, we also demonstrate that this model generalizes well to a similar English dataset for sentiment analysis. To the best of our knowledge, we are the first to explore the modelling of the two tasks together for improving performance on sentiment classification. Moreover, our framework performs better than the generic multi-task learning setup which acts as one of our baselines.

To summarize, the following are our main contributions

- We propose a novel dataset consisting of 2152 user-generated comments along with gold-standard human-annotated sentiment-span labels.

- We propose a unified sentiment prediction and span extraction framework based on transformer architecture

- Through empirical analysis we establish our proposed method's superiority over strong baselines.

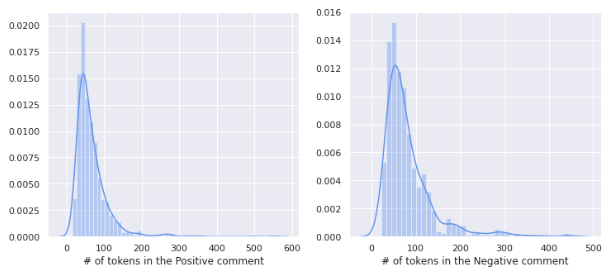

Figure 2: Length of the Positive and Negative comments

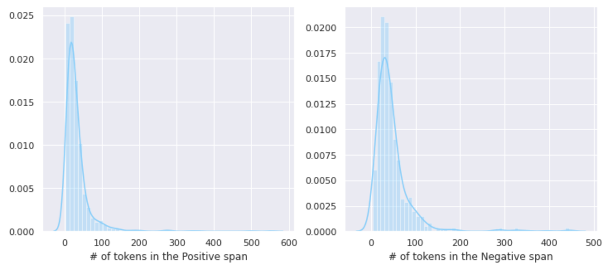

Figure 3: Length of the Positive and Negative spans

- We demonstrate the generalizability of our proposed approach to similar sentiment classification datasets.

2 Related Work

2.1 Sentiment Classification

Sentiment analysis and sentiment classification are widely explored problems in the area of Natural Language Processing. Detecting sentiments in texts helps in identifying its polarity which in turn helps understanding people's opinion.This has been widely employed in e-commerce sites (Agarap, 2018; Hoang et al., 2019) and social media networks (Samuels and Mcgonical, 2020; Aho and Ullman, 1972). With the growing number of users and user-generated content, social media networks are considered a rich data source for this task. Sentiment classification in social media is also critical in tackling mental health problems of its users (Saifullah et al., 2021).

Although most of the advances in sentiment analysis have been in high-resource languages there has been a growing interest and recent progress in low resource and codemixed sentiment analysis. (Patwa et al., 2020) used Twitter to extract the text from users and construct a codemixed corpus for Spanglish and Hinglish. (Kaur et al., 2019) used Youtube to extract hinglish comments from cooking videos and use that to analyze the polarity of the viewers. DravidianCodemix (Chakravarthi et al., 2022) is a recent work that developed sentiment classification corpora for truly low-resourced dravidian languages such as Tamil. As emphasized in (Chakravarthi et al., 2022) it takes lot of effort to obtain and annotate code-mixed sentiment data hence there is a need to make effort to explore and utilize the potential in existing resources.

2.1.1 Sentiment Span Extraction

Sentiment span extraction itself has been less explored in literature.(Pavlopoulos et al., 2021) released dataset of 10k samples for English language.Kaggle hosted a competition for sentiment extraction, the data released from the competition, Sentiment Text Extraction [2] consists of English tweets labelled under three categories- Positive, Negative and Neutral. The task here was to extract the span given the sentiment of the text as input.

2.1.2 MultiTask Learning

MultiTask Learning (Caruana, 1993) have been used for in Machine Learning across the task in Natural Language Processing and Computer Vision, It originates from the idea of learning multiple tasks helps the model to exploit the predictive features of one task to the other task helping in gaining the perfomance. (Barnes et al., 2021) used Multi Task Learning with Attention and LSTM layers for the task of improving the sentiment detection model by using an additional auxillary task of Negation detection. MultiTask Learning also has been widely used in conversational dialogue systems for the task of jointly training the Intent Detection and Slot Tagging tasks. (S et al., 2022) used a Jointly trained pretrained transformers model for the task of Intent Detection and Slot Tagging for Tamil Conversational Dialogues.

3 Dataset

3.1 Data Collection

We extend the DravidianCodemix dataset (Chakravarthi et al., 2022) by adding gold-standard human-annotated sentiment span labels to it. The dataset consists of code-mixed YouTube comments in Tamil-English, Malayalam-English and Kannada-English for the tasks of Sentiment Detection and Offensive Language Identification. It is annotated in a five class setting with classes, Positive state, Negative state, Neutral state and Mixed Feelings.

[2]https://www.kaggle.com/competitions/tweet-sentiment-extraction/

# of unique tokens in a comment	11322
# of unique tokens in a substring	8267
# of unique native Tamil tokens	3435
# of unique romanized Tamil tokens	7885
Avg # of tokens in positive comment	67.07
Avg # of tokens in positive span	32.56
Avg # of tokens in negative comment	82.75
Avg # of tokens in negative span	46.78

Table 1: Corpus analysis of our proposed dataset

Since our goal is to build sentiment span extraction dataset for Tamil-English, we only use the Tamil-English subset. We randomly sample 4935 comments from the Tamil-English subset for the annotation. We only use the comments that were labelled as Positive, Negative and Neutral and discard other labels for the annotation purposes

3.2 Human Annotation

The proposed dataset was completely annotated by human experts who are native speakers of Tamil and who are fluent in English. We hired three annotators who are master's student and native Tamil speaker. We explained the concept of Positive, Negative and Neutral sentiments and provided examples for each. We also explained the concept of code-mixing, based on our interactions with the annotators we found that they also use code-mixing in their daily conversations. We didn't collect any information of annotators other than their education details and known languages. Since YouTube comments may contain comments that are profane in nature, we inform annotators that the comments contain words that are profane, offensive and vulgar in nature. The annotators are given the liberty to withdraw from the annotation, if they feel the necessity.

To aid the annotation effort, we created a custom tool that provides the annotators with an easy to-use interface for annotation. Each annotator was assigned random batches of comments and they worked independently in their own schedule. The annotators were asked to follow the annotation guidelines given below.

- Extract the phrases from the comment which support the sentiment expressed.

- If the comment does not express any sentiment, do not highlight any span.

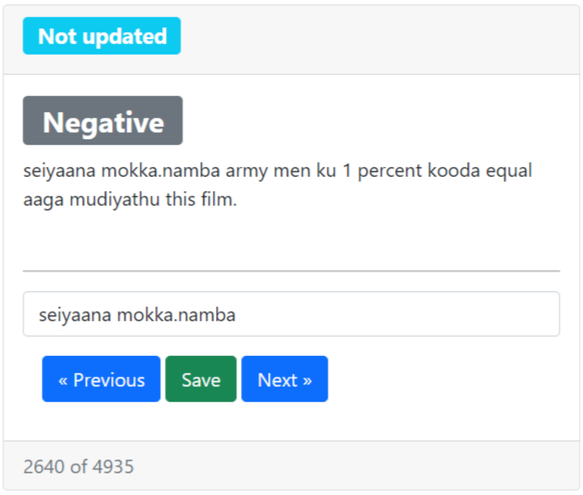
Not updated
Negative
seiyaana mokka.namba army men ku 1 percent kooda equal aaga mudiyathu this film.
seiyaana mokka.namba
« Previous
Save
Next »
2640 of 4935

Figure 4: Interface of the annotation tool used during annotation

- If the entire comment expresses a single sentiment, highlight the whole comment
- If there are any emoji characters that expresses the sentiment, highlight that emoji as a span

Dry Run: We first conducted a dry run to ensure the uniformity in annotation and to check whether the spans are annotated correctly. We took a subset of 100 comments and asked the three annotators to annotate each of the comment independently. After this annotation, we computed the Cohen's for annotated tokens. The Inter-Annotater agreement k value is 0.60

Final Run: The annotations obtained in the dry run were evaluated and the annotators were given feedback on mistakes they made and any doubts they had. Once the annotators were confident that they understood the annotation process we proceeded with the final annotation. At the end of annotation we got 2152 samples. The majority of the 3 annotations were then used as the final annotation. At the end of the annotation and after removing wrong samples, we got 2152 samples. We took the majority of the three annotations as our final annotation.

3.3 Corpus Analysis

Table 1 contains the summary statistics of the dataset. From the table we can infer that the average length of the Negative comments is higher than the average length of the Positive comment. The dataset consists of comments written both on native script and roman script comments labelled as Positive, Negative and Neutral. The final dataset consists of 875 Positive, 679 Negative and 598 Neutral comments. Due to the codemixing nature of the dataset, it consists of Tamil comments written in both Native script and Roman script. We used the langid[3] framework to find the original language of the word based on the nature of the script and found there are 7885 unique English tokens and 3435 unique Tamil tokens. We can also note that, there are some comments that is written entirely on English and some comments that are written entirely on Tamil. The dataset was split into Train, Dev and Test sets in the ratio of 80:10:10. Our dataset is released as CSV files.

4 Proposed Approach

In this section we describe our proposed approach. It takes as input a piece of text x and predicts the sentiment of the x and the span within x that display this sentiment.

Fig 5 depicts the architecture of the proposed model. It consists of a text encoder that provides contextual representation of x at both sentence and word level. It then uses a Task Interaction Module (TIM) to learn the interactions between the two tasks of sentiment classification and sentiment span extraction.

4.1 Text Encoder

Given a piece of text x consisting of n tokens $[x_1, x_2, ...x_n]$ we encode it using a transformer based text encoder. We use the word-level representations $H = [h_1, h_2, ...h_n]$ obtained from the last hidden layer.

4.2 Task Interaction Module

The Task Interaction Module (TIM) is utilized to learn the inter-dependencies between the task of sentiment classification and sentiment span extraction.

Each encoder block in TIM consists of the following two components; a label attention layer that produces explicit sentiment and span representations; a co-attention mechanism to model the mutual interactions between the two tasks.

4.2.1 Label Attention Layer

We utilize label attention over the sentiment and span labels to produce explicit sentiment and span representations. These representations are then fed

[3]https://github.com/saffsd/langid.py

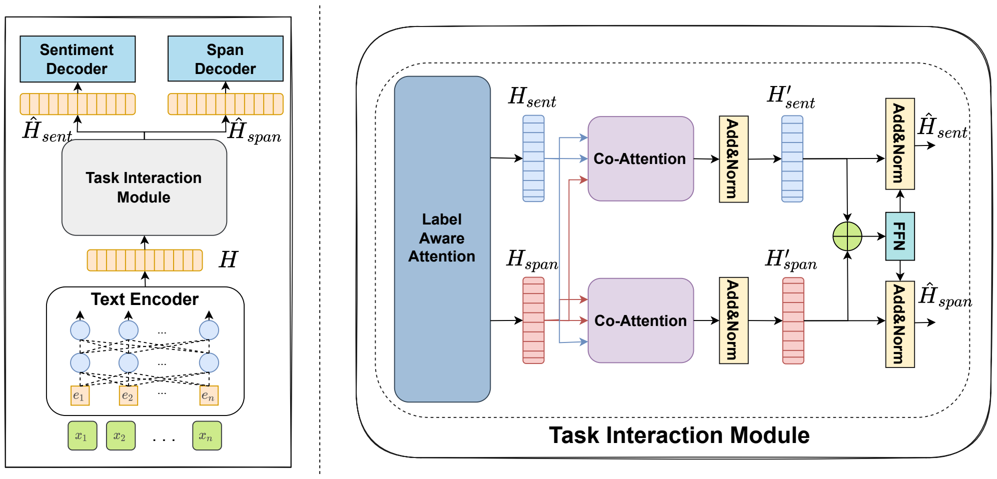

Figure 5: The architecture of our proposed model (left). It uses a transformer based Task Interaction Module (left) to explicitly model the mutual interactions between the two tasks.

into the co-attention layer to capture the mutual interactions. We use the parameters of the fully-connected sentiment and span decoders as sentiment and span embeddings matrices ($W^{sent} \in R^{d\times 3}$ and $W^{span} \in R^{d\times 2}$); as they can be considered to be label distribution in a sense.

We use $H \in R^{n\times d}$ and $W^v \in R^{n\times |v|}$ ($v \in$ sent or span) to obtain the explicit representation H_v as follows

$$A = softmax(HW^v) \tag{1}$$

$$H_v = H + AW^v \tag{2}$$

Here $sent$ represents sentiment and $span$ represents span. We finally obtain the explicit sentiment and span representations H_{sent} and H_{span}, which capture the sentiment and span semantic information respectively.

4.2.2 Co-Attention Layer

H_{sent} and H_{span} are next passed through a co-attention mechanism to model the mutual interactions between the two tasks of sentiment and span prediction. Through this mechanism we get sentiment representations updated with guidance from span representation and vice versa. This establishes a bi-directional connection between the two tasks.

We use linear projections on H_{sent} and H_{span} to generate the query (Q_{sent} , Q_{span}),key (K_{sent} , K_{span}) and value (V_{sent} , V_{span}) vectors respectively.

To incorporate span information in sentiment representation it is necessary to align sentiment with its closely related spans. We use Q_{sent} as query and K_{span}, V_{span} as key and value vectors respectively. We then get span-aware sentiment representation H'_{sent} as follows

$$C_{sent} = softmax\left(\frac{Q_{sent}K^T_{span}}{\sqrt{d_k}}\right)V_{span} \tag{3}$$

$$H'_{sent} = LayerNorm(H_{sent} + c_{sent}) \tag{4}$$

In a similar fashion we obtain sentiment-guided span representation by treating K_{sent}, V_{sent} as key and value vectors and Q_{span} as query vector. Through the co-attention layer we obtain H'_{sent} and H'_{span} which can be considered to be span-guided sentiment representation and sentiment-guided span representation respectively.

We extend the feed-forward network layer from a vanilla transformer encoder block to implicitly fuse sentiment and span information. We concatenate H'_{sent} and H'_{span} to combine the sentiment and span information.

$$H_{ss} = H'_{sent} + H'_{span} \tag{5}$$

Then we follow (Zhang and Wang, 2016) to use word features for each token, which is formulated as

$$h_{(f,t)} = h^{t-1}_{ss} h^{t}_{ss} h^{t+1}_{ss} \tag{6}$$

Finally we use feed-forward networks to fuse the

Type	Sentiment Classification					Span Prediction		
	Text Encoder	**Accuracy**	**F1**	**Precision**	**Recall**	**F1**	**Exact Match**	**Jaccard Sim.**
STL	BERT	59.72%	55.85%	57.16%	57.52%	52.16%	8.33%	50.84%
	MBERT	62.96%	61.3%	61.97%	61.28%	54.51%	7.41%	45.68%
	MURIL	63.01%	61.87%	62.24%	62.84%	54.81%	8.33%	50.12%
MTL	BERT	61.11%	59.93%	60.14%	59.90%	53.01%	8.80%	51.32%
	MBERT	62.96%	62.22%	62.97%	62.12%	57.80%	9.72%	49.64%
	MURIL †	63.43%	62.85%	65.22%	63.05%	57.23%	8.33%	49.25%
OURS	BERT	61.57%	62.19%	64.27%	62.34%	53.81%	9.72%	52.37%
	MBERT	64.81%	62.80%	64.02%	63.16%	58.83%	9.72%	50.39%
	MURIL *	**65.74%**	**64.12%**	**67.41%**	**64.28%**	**59.94%**	**11.11%**	**52.43%**
$\Delta_{(\star-\dagger)\times 100}(\%)$		$\uparrow$ 2.31%	$\uparrow$ 1.27%	$\uparrow$ 3.39%	$\uparrow$ 1.23%	$\uparrow$ 2.71%	$\uparrow$ 2.78%	$\uparrow$ 3.18%

Table 2: Comparison of different approaches on our dataset. The last row shows the absolute improvement of our approach over the MTL approach with the MURIL as text encoder.

sentiment and span information.

$$FFN(H_{(f,t)}) = max(0, H_{(f,t)}W_1 + b + 1)W_2 + b_2 \quad (7)$$

$$\hat{H}_{sent} = LayerNorm(H'_{sent} + FFN(H_{(f,t)}) \quad (8)$$

$$\hat{H}_{span} = LayerNorm(H'_{span} + FFN(H_{(f,t)}) \quad (9)$$

Here $H_{(f,t)} = (h^1_{(f,t)}, h^2_{(f,t)}...h^t_{(f,t)})$; $\hat{H_{sent}}$ and $\hat{H_{span}}$ are the final updated sentiment and span representations that align the corresponding span and sentiment features respectively.

4.2.3 Decoder For Sentiment And Span Prediction

We utilize two decoder heads to get the final predictions, one head each for sentiment prediction and sentiment span extraction task respectively.

Sentiment Prediction We apply max-pooling operation on $\hat{H}_{sent}$ to obtain sentence representation c which is used for sentiment prediction.

$$\hat{y}^{sent} = softmax(W^{sent}c + b_{sent}) \quad (10)$$

Span Classification We pass $\hat{H}_{span}$ through feed-forward networks to obtain the start and end position as follows

$$\hat{y}^{span} = softmax(W^{span}\hat{H}_{span} + b_{span}) \quad (11)$$

5 Experimental Results

In this section we present the results (averaged over 5 independent runs) on our test set and perform comparative analysis followed qualitative and error analysis. For comparison we use the following standard metrics - accuracy, macro averaged F1 score, precision, recall for sentiment prediction task and F1 score, exact match and jaccard similarity for the span prediction task.

5.1 Baselines And Compared Methods

We compare our approach with single-task learning (STL) architectures and multi-task learning (MTL) architectures.

1. **Single-Task Learning** In this setup we separately train two transformer based text encoders, one for sentiment prediction and one for span extraction.

2. **Multi-Task Learning** In this setup we train a transformer based text encoder jointly for sentiment prediction and span extraction

In both STL and MTL setup we use the pooled representation corresponding to the [CLS] token as sentence representation for sentiment prediction and use the token level representations from the last hidden layer for span extraction.

Text Encoders We experiment with three text encoders. The first one is the BERT(Devlin et al., 2018) base model, since the dataset is codemixed we also experiment with MBERT and MURIL(Khanuja et al., 2021) which are multilingual models based on BERT architecture. Both MBERT and MURIL are trained on english and Tamil text corpus and specifically MURIL is trained on a romanized Tamil corpus.

5.2 Main Results

Table 2 depicts the results obtained via different approaches and text encoders on our dataset.

Sentiment Classification					Span Extraction		
Text Encoder	**Accuracy**	**F1**	**Precision**	**Recall**	**F1**	**Exact Match**	**Jaccard Sim.**
No Label Attention	62.50%	62.36%	64.36%	62.57%	54.11%	9.26%	52.95%
Self Attention Mechanism	64.35%	62.83%	63.41%	62.83%	54.76%	11.11%	53.56%
Sentiment To Span Connection	64.22%	60.88%	64.22%	61.37%	50.20%	9.26%	49.36%
Span To Sentiment Connection	64.43%	61.06%	61.39%	61.28%	53.24%	9.72%	51.30%

Table 3: Each key component in the proposed approach contributes to overall performance. Replacing or removing a component results in a drop in performance.

Type	**Sentiment Classification**					**Span Extraction**		
	Text Encoder	**Accuracy**	**F1**	**Precision**	**Recall**	**F1**	**Exact Match**	**Jaccard Sim.**
STL	BERT	75.01%	75.41%	75.85%	74.91%	48.48%	16.62%	44.58%
MTL	BERT †	76.47%	76.61%	78.92%	75.63%	49.97%	19.07%	45.70%
OURS	**BERT** *	**78.33%**	**78.63%**	**78.97%**	**78.36%**	**54.86%**	**20.16%**	**50.76%**
$\Delta_{(\star-\dagger)\times 100}(\%)$		↑ 1.86%	↑ 2.02%	↑ 0.06%	↑ 2.73%	↑ 4.89%	↑ 1.09%	↑ 5.06%

Table 4: Comparison of different approaches on the Twitter Sentiment Extraction dataset. The last row shows the absolute improvement of our approach over the MTL approach.

We experiment with three different text encoders, we observe that among these MURIL performs better than MBERT and BERT in both STL and MTL setup, moreover when MURIL is used as text encoder in our approach it acheives the best performance for our dataset.

We observe that models trained in MTL setup perform better than STL models for all the text encoders across all the metrics, this indicates that jointly learning the tasks of sentiment prediction and span extraction can mutually enhance performance.

MTL can be seen as considering the mutual-interaction between the two tasks via parameter sharing and joint optimization, however our approach out-performs MTL setup with their respective text encoders. Moreover when compared to the best MTL setup which is MTL MURIL, our approach with MURIL text encoder performs better.

5.3 Ablation Study

In this section we study the efficacy of the key components present in our approach. Table 3 shows the sentiment prediction and span prediction results using our approach on our dataset. We modify the key components in our proposed approach to investigate their contribution to the performance.

We drop the label attention layer and replace H_{sent} and H_{span} with H. From Table 3 we observe that this leads to a drop in performance. This demonstrates the usefulness of using label information to generate explicit sentiment and span representations.

We replace the co-attention mechanism in TIM with the vanilla self-attention mechanism. This change means that there is no explicit interaction between the two tasks. From Table 3 we notice that this leads to a drop in performance justifying the use co-attention mechanism. While self-attention only implicitly models the interaction between the sentiment and span tasks, co-attention can explicitly consider the cross-impact between the two.

We restrict the bi-directional flow of information so that the information can either from from sentiment to span or span to sentiment. We implement this by using only one type of information representation as queries to attend to the other information. In table we refer to this as Sentiment To Span and Span To Sentiment. From Table 3 we observe that such a unidirectional flow of information leads to a performance drop. We can conclude that modelling the mutual interaction between the sentiment prediction and span prediction task can enhance the performance in a mutual way.

5.4 Generalizability

In this section we establish the generalizability of our proposed approach by experimenting on the Twitter Sentiment Extraction dataset from Kaggle. The original task for this dataset is to extract the sentiment span given the sentiment, however we repurpose it for our task of joint sentiment prediction and span extraction. Since the span labels are not present in the test set, we split the original train set into a 80/20 split and perform our testing on the unseen 20 split while the training and validation is done on the 80 split.

Table 4 shows the results of STL, MTL and our proposed appoach on the dataset. We observe that MTL setup performs better that the STL setup indicating that jointly optimizing for the two tasks of sentiment prediction and span extraction is mutually beneficial. Our approach shows improvement over the MTL setup across all the metrics thus demonstrating the capability of our proposed model to generalize to other sentiment prediction datasets.

6 Conclusion and Future Work

In this work we explore the use of sentiment span cues towards improving code-mixed sentiment prediction. We first curate a novel manually annotated dataset to support code-mixed sentiment span extraction. We then propose a novel methodology based on transformer architecture to explicitly model the the mutual interactions between sentiment prediction and sentiment span extraction tasks. Empirical evaluation along with an extensive ablation study suggests the efficacy of our proposed model and its design choices. We also establish the generalizability of the proposed model by demonstrating its performance on the Twitter Sentiment Extraction dataset.

References

Abien Fred Agarap. 2018. Statistical analysis on e-commerce reviews, with sentiment classification using bidirectional recurrent neural network (rnn).

Alfred V. Aho and Jeffrey D. Ullman. 1972. *The Theory of Parsing, Translation and Compiling*, volume 1. Prentice-Hall, Englewood Cliffs, NJ.

Jeremy Barnes, Erik Velldal, and Lilja Øvrelid. 2021. Improving sentiment analysis with multi-task learning of negation. *Natural Language Engineering*, 27(2):249–269.

Richard Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 41–48. Morgan Kaufmann.

Bharathi Raja Chakravarthi, Vigneshwaran Muralidaran, Ruba Priyadharshini, and John Philip McCrae. 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)*, pages 202–210, Marseille, France. European Language Resources association.

Bharathi Raja Chakravarthi, Ruba Priyadharshini, Vigneshwaran Muralidaran, Navya Jose, Shardul Suryawanshi, Elizabeth Sherly, and John P. McCrae. 2022. Dravidiancodemix: Sentiment analysis and offensive language identification dataset for dravidian languages in code-mixed text. *ArXiv*, abs/2106.09460.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Sreyan Ghosh and Sonal Kumar. 2021. Cisco at semeval-2021 task 5: What's toxic?: Leveraging transformers for multiple toxic span extraction from online comments. *arXiv preprint arXiv:2105.13959*.

Suong N. Hoang, Linh V. Nguyen, Tai Huynh, and Vuong T. Pham. 2019. An efficient model for sentiment analysis of electronic product reviews in vietnamese.

Gagandeep Kaur, Abhishek Kaushik, and Shubham Sharma. 2019. Cooking is creating emotion: A study on hinglish sentiments of youtube cookery channels using semi-supervised approach. *Big Data and Cognitive Computing*, 3(3).

Simran Khanuja, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip Kumar Margam, Pooja Aggarwal, Rajiv Teja Nagipogu, Shachi Dave, Shruti Gupta, Subhash Chandra Bose Gali, Vish Subramanian, and Partha Talukdar. 2021. Muril: Multilingual representations for indian languages.

Shanrou Lai, Zichen Yu, and Hanyue Wang. 2020. Text sentiment support phrases extraction based on roberta. In *2020 2nd International Conference on Applied Machine Learning (ICAML)*, pages 232–237.

Parth Patwa, Gustavo Aguilar, Sudipta Kar, Suraj Pandey, Srinivas PYKL, Björn Gambäck, Tanmoy Chakraborty, Thamar Solorio, and Amitava Das. 2020. SemEval-2020 task 9: Overview of sentiment analysis of code-mixed tweets. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 774–790, Barcelona (online). International Committee for Computational Linguistics.

John Pavlopoulos, Jeffrey Sorensen, Léo Laugier, and Ion Androutsopoulos. 2021. SemEval-2021 task 5: Toxic spans detection. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 59–69, Online. Association for Computational Linguistics.

Libo Qin, Tailu Liu, Wanxiang Che, Bingbing Kang, Sendong Zhao, and Ting Liu. 2021. A co-interactive transformer for joint slot filling and intent detection. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8193–8197. IEEE.

Ramaneswaran S, Sanchit Vijay, and Kathiravan Srinivasan. 2022. TamilATIS: Dataset for task-oriented dialog in Tamil. In *Proceedings of the Second Workshop on Speech and Language Technologies for Dravidian Languages*, pages 25–32, Dublin, Ireland. Association for Computational Linguistics.

Shoffan Saifullah, Yuli Fauziyah, and Agus Sasmito Aribowo. 2021. Comparison of machine learning for sentiment analysis in detecting anxiety based on social media data. *Jurnal Informatika*, 15(1):45.

Antony Samuels and John Mcgonical. 2020. Sentiment analysis on social media content.

Xiaodong Zhang and Houfeng Wang. 2016. A joint model of intent determination and slot filling for spoken language understanding. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, page 2993–2999. AAAI Press.