Multimodal Sentiment Analysis of Tamil and Malayalam D2

Abhinav Patil, Sam Briggs, Tara Wueger, D. D. O'Connell
Department of Linguistics, University of Washington
Seattle, WA
{abhinavp, briggs3, taraw28, danieloc}@uw.edu

Abstract

Sentiment analysis consists of categorizing a unit of language about a certain topic according to the author's attitude towards that topic. Our task is the classification of multi-modal data – text, audio, and video data – disseminated in the Dravidian languages Tamil and Malayalam into 5 separate sentiment classes: highly negative, negative, neutral, positive, and highly positive. We implement and evaluate a Multinomial Naive Bayes baseline model, as well as a finetuned XLM RoBERTA model for comparison. We found that both the baseline and the Neural Network have the same performance as a naive Majority Class Classifier baseline. Finally, we perform error analysis of the two models.

1 Introduction

Sentiment analysis consists of categorizing a unit of language about a certain topic according to the author's attitude towards that topic. This task can be constructed in many ways, e.g. as a binary classification task (i.e., categorizing language in two classes, Positive and Negative sentiment), as an ordinal regression problem, or, as in our case, as a multi-class classification problem. Sentiment analysis has a rich history and many methods have been used (Cui et al., 2023).

2 Task Description

We will participate in the "Multimodal Abusive Language Detection and Sentiment Analysis: DravidianLangTech@RANLP 2023" shared task hosted on CodaLab (B. et al., 2023). [1]

This shared task has two subtasks: abusive language detection in Tamil, and sentiment analysis in both Tamil and Malayalam (members of the Dravidian language family). Both tasks are multimodal, consisting of videos paired with files containing just their audio tracks as well as (sometimes partial) text transcripts. For the purposes of this class, we will be working on the second subtask of sentiment analysis in Tamil and Malayalam.

2.1 Primary Task: Sentiment Analysis of Text Data

Our task is to categorize text written in either Tamil or Malayalam into 5 different categories: Highly Negative, Negative, Neutral, Positive, or Highly Positive. The data is ordinal, that is, the categories are discrete and there exists a total order over them, but the distances between the categories are taken to be unknown or not well-defined. In our initial attempts, we tried formulating the task both in terms of ordinal regression and in terms of multiclass classification.

Previous work on sentiment analysis on Dravidian languages has been done using code-switched data (English and either Tamil or Malayalam) (Chakravarthi et al., 2021). Our data is not code-switched, and only contains one language. Like many tasks in Natural Language Processing, different Neural Network architectures have been used to perform sentiment analysis (Habimana et al., 2019).

2.2 Adaptation Task: Sentiment Analysis of Multimodal Data: Text, Audio, and Video

For our adaptation task, we will continue to work on sentiment analysis; however, we will be including audio and video data in addition to the text data from the primary task. This task will also be multilingual, using data from both Tamil and Malayalam. The same five categories will be used in the adaptation task as in the primary task.

Work on analyzing and indexing videos has been summarized in Snoek and Worring (2005). Additionally, Dimitri (2022) contains an extensive

[1] The official citation has yet to be released as of April 26, 2023.

overview of multimodal integration methods and their applications, including for the purpose of sentiment analysis.

2.3 Data set

2.3.1 Description

The shared task organizers have provided, in total, 52 Tamil samples and 70 Malayalam samples. Each sample consists of a video containing speech in the given language, the corresponding audio file, and a (sometimes partial) transcript of the speech in the audio. From an initial analysis of a subset of the data samples, with the help of Google Translate, we tentatively believe the samples are all movie reviews, though this has not been stated by the task organizers explicitly.

The shared task organizers permit the use of other data sources and pretrained models as long as they are named and cited properly. We are still exploring the possibility of using other data; at the present time, it is unlikely we will do so. We are, however, likely to use pretrained models in some way.

2.3.2 Split

The task organizers have provided a train and dev split over the data set, which is further subdivided by language (Tamil and Malayalam). The 52 Tamil samples are split 42/10 train/dev while the Malayalam samples are split 60/10.

However, instead of using the official split, we combined the train and dev data into a single train set, and then used k-fold validation as we explain later.

For evaluation, an official evaluation dataset will be provided in the coming weeks by the organizers.

The official data can be found in Google Drive folder linked below:

- Tamil train data
- Tamil dev data
- Malayalam train data
- Malayalam dev data

2.4 Evaluation

The organizers of the competition have stated that they will use an F1 metric for evaluation purposes. They have not release evaluation tools or scripts at this time, but suggest teams use Sklearn's classification report function (which lists precision, recall, F1, and a confusion matrix) in evaluating performance, suggesting that they may use the same function in evaluation tools (whether eventually released or not). We intend to follow their advice in this matter.

We note that F1 is a problematic metric for ordinal data, as it is ambivalent to the ordering of the class labels; for example, given a review whose true label is "highly positive," an F1 score would equally penalize a model classifying it as "positive" as "highly negative." However, since this is the metric of choice for the task organizers, we will use it as our baseline evaluation metric (over the official "dev" data, which is our test data). Nevertheless, we are currently exploring other metrics we can use, both for evaluation of our end-to-end system, and for use as a loss metric when training our model(s), which would more accurately capture the inter-class ordering of our labels.

3 System Overview

As you see from figure TODO: **??**, there are three main stages to our system pipeline, namely preprocessing, classifying, and evaluating. First we preprocess the data. To preprocess the data we tokenize the data and then vectorize the data using various methods to get document feature vectors. Second, we classify each document into the 5 different sentiment categories. To classify, we first train various models using the training data, and then use the various trained models to predict the sentiment of specific documents using K-fold Cross validation. Lastly, we evaluate the performance of each model using various classification evaluation metrics.

4 Approach

4.1 Database Object

4.2 Baseline

4.2.1 Preprocessing

During preprocessing for the baseline, we first tokenized the data by whitespace. We then removed any tokens containing punctuation or numbers, as well as stop words. We used the list of stop words provided by `spaCy`[2] for both Tamil and Malayalam.

[2]We used spaCy v3.*. The Tamil and Malayalam language models can be found here.

4.2.2 TF-IDF Vectors

We then create one TF-IDF vector per document in the data set. To create the TF-IDF vectors, we run each document through the TF-IDF vectorizer provided by `sklearn`.[3]

To calculate TF-IDF, we used the unsmoothed TF-IDF provided by `sklearn`. Given a document set D with n documents, a document $d \in D$, and a term t with document frequency $df(t)$, we calculate TF-IDF for term t as follows:

$$\text{TFIDF}(t, d, D) = tf(t, d) \cdot \text{idf}(t) \quad (1)$$

$$tf(t, d) = count(t) \in d \quad (2)$$

$$\text{idf}(t, D) = \log\left(\frac{n}{df(t)}\right) + 1 \quad (3)$$

4.2.3 Multinomial Naive Bayes Classifier

We used a multinomial Naive Bayes classifier as our baseline. To build our model, we train the multinomial Naive Bayes classifier on the TF-IDF feature vectors from section 4.2.2. To run Naive Bayes, we used `sklearn`.

4.3 Finetuned RoBERTa Model

We also finetuned the XLM RoBERTa model (available from HuggingFace) on our task. We tried training across a variety of hyperparameter settings: we varied the number of epochs from 5 to as many as 1000, we tried freezing or unfreezing various layers at various points during training, and we tried training with both a multiclass classifier head and a regression head. However, in terms of actual accuracy over the k-fold validation sets, results did not vary.

5 Results

5.1 Evaluation

To evaluate our classifiers, we pooled the train and dev into one large training set, and then ran k-fold Cross-Validation. This allowed us to train our models on more training data, as there was not a lot of training data provided. For k-fold Cross-Validation, we used 4 folds. To be able to compare the different models that we created, we used a deterministic algorithm to shuffle the data. This ensures that the same four train and dev splits were used for all the models, making the performance of our different models comparable.

[3]We used sklearn v1.*. `Sklearn`'s TF-IDF vectorizer documentation can be found here

For k-fold Cross-Validation, we report the per-fold weighted f_1 score, the per-fold weighted accuracy, and the per-fold weighted precision. We also report the pooled weighted f_1 score, pooled weighted accuracy, and the pooled weighted precision, by taking the averages of the respective per-fold scores over all four folds.

Table 1: Zeroth fold scores for Tamil

	Precis.	Recall	F1	Support
Highly neg.	0.00	0.00	0.00	1
Negative	0.00	0.00	0.00	1
Neutral	0.00	0.00	0.00	2
Positive	0.64	1.00	0.78	9
Highly pos.	0.00	0.00	0.00	1
Accuracy			0.64	14
Macro avg.	0.13	0.20	0.16	14
Weighted avg.	0.41	0.64	0.50	14

Table 2: First fold scores for Tamil

	Precis.	Recall	F1	Support
Highly neg.	0.00	0.00	0.00	1
Negative	0.00	0.00	0.00	1
Neutral	0.00	0.00	0.00	2
Positive	0.62	1.00	0.76	8
Highly pos.	0.00	0.00	0.00	2
Accuracy			0.57	14
Macro avg.	0.11	0.20	0.15	14
Weighted avg.	0.33	0.57	0.42	14

Table 3: Second and third fold scores for Tamil

	Precis.	Recall	F1	Support
Highly neg.	0.00	0.00	0.00	1
Negative	0.00	0.00	0.00	1
Neutral	0.00	0.00	0.00	1
Positive	0.62	1.00	0.76	8
Highly pos.	0.00	0.00	0.00	2
Accuracy			0.62	13
Macro avg.	0.12	0.20	0.15	13
Weighted avg.	0.38	0.62	0.47	13

5.2 Baseline

The multinomial Naive Bayes classifier assigns each document the POSITIVE label.

Table 4: Pooled scores for Tamil

	Precis.	Recall	F1	Support
Highly neg.	0.00	0.00	0.00	4
Negative	0.00	0.00	0.00	4
Neutral	0.00	0.00	0.00	6
Positive	0.61	1.00	0.76	33
Highly pos.	0.00	0.00	0.00	7
Accuracy			0.61	54
Macro avg.	0.12	0.20	0.15	54
Weighted avg.	0.37	0.61	0.46	54

Table 5: All scores for Malayalam

	Precis.	Recall	F1	Support
Highly neg.	0.00	0.00	0.00	0
Negative	0.00	0.00	0.00	10
Neutral	0.00	0.00	0.00	6
Positive	0.60	1.00	0.75	36
Highly pos.	0.00	0.00	0.00	7
Accuracy			0.60	60
Macro avg.	0.12	0.20	0.15	60
Weighted avg.	0.36	0.60	0.40	60

5.3 XLM RoBERTa Model

All permutations of our finetuned XLM RoBERTa model performed the same as the NB classifier, assigning each document the POSITIVE label.

6 Discussion

We believe that the reason both the baseline and actual models result in POSITIVE labels for all documents is due to the fact that the amount of data available to train on is fairly small and most of the gold labels are POSITIVE.

In the future we will look at ways to mitigate this problem. We plan on looking at oversampling techniques such as the Synthetic Minority Oversampling Technique (SMOTE) to deal with the class imbalance. While training our models, we largely focused on accuracy, but because of the class imbalance, moving forward, we intend to focus on macro F1 score instead. For example, in the process of finetuning the XLM RoBERTa models, for certain hyperparameter settings, the macro F1 was higher than we reported above, but this came at the cost of accuracy. For that reason, we initially discarded these models as inferior. However, we intend to revisit that choice with a focus on maximizing macro F1 instead.

7 Ethical Considerations

TODO: Ethical Considerations.

Sentiment Analysis is used in... This has implications because...

8 Conclusion

References

Premjith B., Sowmya V., Joyithish Lal. G., Bharathi Raja Chakravarthi, K. Nandhini, Rajeswari Natarajan, Abirami Murugappan, and Bharathi B. 2023. Multimodal Abusive Language Detection and Sentiment Analysis:DravidianLangTech@RANLP 2023.

Bharathi Raja Chakravarthi, Ruba Priyadharshini, Vigneshwaran Muralidaran, Shardul Suryawanshi, Navya Jose, Elizabeth Sherly, and John P. McCrae. 2021. Overview of the track on sentiment analysis for dravidian languages in code-mixed text. In *Proceedings of the 12th Annual Meeting of the Forum for Information Retrieval Evaluation*, FIRE '20, page 21–24, New York, NY, USA. Association for Computing Machinery.

Jingfeng Cui, Zhaoxia Wang, Seng-Beng Ho, and Erik Cambria. 2023. Survey on sentiment analysis: evolution of research methods and topics. *Artificial Intelligence Review*.

Giovanna Maria Dimitri. 2022. A short survey on deep learning for multimodal integration: Applications, future perspectives and challenges. *Computers*, 11(11).

Olivier Habimana, Yuhua Li, Ruixuan Li, Xiwu Gu, and Ge Yu. 2019. Sentiment analysis using deep learning approaches: an overview. *Science China Information Sciences*, 63(1):111102.

Cees GM Snoek and Marcel Worring. 2005. Multimodal video indexing: A review of the state-of-the-art. *Multimedia tools and applications*, 25:5–35.

A Appendices