

Evaluating the Performance of the No Language Left Behind Encoder for Sentiment Classification Across Multiple Indonesian Regional Languages

Zul Akhyar¹, Zahnur², Kikye Martiwi Sukiakhy^{3*}, Zulfan⁴, Nazaruddin⁵
^{1,2,3,4,5} Departmen Informatika, Universitas Syiah Kuala, Banda Aceh, Indonesia

ARTICLE INFO

Article history:
Initial submission
01-04-2026
Received in revised form
17-05-2026
Accepted 10-06-2026
Available online 30-09-2026

Keywords:
*NLLB, Sentiment
Classification, Regional
Languages, Fine-Tuning*

DOI:
<https://doi.org/10.59356/smart-techno.v8i2.213>

ABSTRACT

This study aims to evaluate the performance of the encoder from the No Language Left Behind (NLLB) model for sentiment classification tasks across several Indonesian regional languages. Originally developed for machine translation, the NLLB model is explored for its ability to generate contextual text representations that are relevant to sentiment classification. The dataset used in this study is NusaX, which comprises 12 languages, including Indonesian, English, and 10 Indonesian regional languages. Two training approaches were employed: fine-tuning, in which all model parameters were updated using sentiment classification data, and partial fine-tuning, in which only the upper layers were updated while the embedding layer was frozen to preserve the original lexical representations. Training was conducted using the AdamW optimization algorithm with CrossEntropyLoss as the loss function and mean pooling as the feature aggregation mechanism. Model performance was evaluated using accuracy and macro F1-score metrics at both the multilingual and per-language levels. The results indicate that the two approaches achieved comparable performance, with an accuracy of 81%

and a macro F1-score of 80% on the multilingual dataset. Per-language analysis further revealed that the model performed better on languages that had been included in the original NLLB pretraining data, such as Acehnese, Balinese, Banjarese, Minangkabau, Javanese, and Sundanese, achieving accuracy scores ranging from 79% to 86%. In contrast, several languages that were not covered during NLLB pretraining, including Ngaju, Madurese, and Toba Batak, exhibited slightly lower performance, with accuracy scores ranging from 70% to 78%. These findings demonstrate that the NLLB encoder possesses strong adaptation capabilities for text classification tasks, even in the context of low-resource regional languages.

1. INTRODUCTION

Advances in Natural Language Processing (NLP) have driven the development of various Transformer-based models capable of understanding and generating text with high accuracy (Sucholutsky et al., 2019). Models such as Bidirectional Encoder Representations from Transformers (BERT) (Ishak et al., 2025), Cross-lingual Language Model-RoBERTa (XLM-R) (Tsalitsatun et al., 2024), and Text-to-Text Transfer Transformer (T5) have been widely applied to a variety of NLP tasks, including machine translation and text classification. However, these models are generally optimized for English or a limited number of high-resource languages with abundant online data. For instance, IndoBERT was specifically developed for Indonesian by training a BERT-based model on a large Indonesian corpus to improve performance on NLP tasks in the Indonesian language.

In contrast, many Indonesian regional languages remain underrepresented in NLP research due to the scarcity of labeled datasets and limited research attention. As a result, existing NLP technologies are often unable to effectively support these languages, thereby widening the digital divide in language technology development and access to NLP applications.

To address this challenge, NusaX was introduced as a multilingual dataset covering ten Indonesian regional languages. In addition, Meta AI developed No Language Left Behind (NLLB), a multilingual model designed to support translation across more than 200 languages, including many low-resource languages. Among these are several Indonesian regional languages, such as Acehnese, Balinese, Banjarese, Buginese, Minangkabau, Javanese, and Sundanese.

NLLB employs a Transformer-based encoder-decoder architecture, where the encoder transforms input text into numerical representations, while the decoder generates text in the target language. Although NLLB was originally developed as a Neural Machine Translation (NMT) model, its encoder has the potential to be utilized for other NLP tasks, including text classification. This is consistent with the work of Ni et al. (2022), who used the encoder component of the T5 model another encoder-decoder Transformer architecture for text classification tasks. Furthermore, Eriguchi et al. (2018) demonstrated that the encoder component of an encoder-decoder NMT model based on Recurrent Neural Networks (RNNs) can be effectively applied to text classification tasks. Such applications are typically enabled through transfer learning approaches, including fine-tuning and freezing selected encoder parameters during training.

Motivated by these findings, this study evaluates the performance of the encoder component of the NLLB model for sentiment classification in Indonesian regional languages using the NusaX dataset. In this context, sentiment classification refers to the task of categorizing text into sentiment classes such as positive, neutral, and negative. The study compares two training strategies: full fine-tuning, in which all encoder parameters are updated during training, and partial fine-tuning, in which only a subset of encoder parameters is updated while the remaining parameters are frozen. The performance of these approaches is evaluated using accuracy and macro F1-score metrics to determine which method is more effective for multilingual sentiment classification in Indonesian regional languages.

2. LITERATURE REVIEW

Natural Language Processing (NLP)

The rapid advancement of Natural Language Processing (NLP) in recent years has been largely driven by breakthroughs in Deep Learning, particularly through the introduction of Transformer architectures and Large Language Models (LLMs). These technologies have significantly improved performance across a wide range of NLP tasks, including information extraction, text classification, sentiment analysis, and machine translation.

NusaX Dataset

NusaX is a collection of datasets for Indonesian regional languages developed to support NLP research on local languages in Indonesia. The dataset was introduced to provide high-quality labeled data for two NLP tasks: sentiment classification and machine translation. One of the main strengths of NusaX is its broad linguistic coverage, encompassing 12 languages, including Indonesian, English, and ten Indonesian regional languages: Acehnese, Balinese, Banjarese, Buginese, Madurese, Minangkabau, Javanese, Ngaju, Sundanese, and Toba Batak.

The dataset contains a total of 12,000 samples, distributed across the 12 languages, with 1,000 samples per language. The training set (Training_all) constitutes the largest portion of the dataset, accounting for 50% of the data (6,000 samples). This is followed by the testing set (Testing_all), which comprises 40% of the data (4,800 samples). Finally, the validation set (Validation_all) represents the remaining 10% (1,200 samples). Therefore, the overall data split ratio is 50:40:10 for the training, testing, and validation sets, respectively. For each language, the 1,000 text samples are further divided into training, validation, and testing subsets. All samples are annotated for sentiment classification tasks using three sentiment labels: positive, neutral, and negative. The detailed distribution of data across subsets and sentiment labels is presented in Table 1.

Table 1. Sentiment Label Distribution Across Languages in the NusaX Dataset

Subset	Negatif	Netral	Positif	Total	Persentase
Train	192	119	189	500	50%
Valid	38	24	38	100	10%
Test	153	96	151	400	40%
Total	383	239	378	1000	100%

Transformer

The Transformer architecture consists of two main components: the encoder and the decoder. The encoder is responsible for processing the input text and converting it into numerical representations that capture the semantic meaning of the text. The decoder receives these representations from the encoder and utilizes them to generate outputs, such as translated text or summarized content. Before being processed by either the encoder or decoder, each token in the input text is transformed into a vector representation through an embedding layer. Embedding serves as the initial stage that converts tokens into token embeddings, which are fixed-dimensional vectors containing fundamental semantic information about the corresponding tokens. Since the Transformer architecture does not inherently model sequential order as Recurrent Neural Networks (RNNs) do, it incorporates positional embeddings to encode the positions of tokens within a sequence. The sum of the token embeddings and positional embeddings forms the initial input to both the encoder and decoder components.

No Language Left Behind (NLLB)

The No Language Left Behind (NLLB) model is built upon a Transformer-based encoder decoder architecture. However, during its development, NLLB was implemented in two variants: the Dense Transformer, which follows the standard Transformer architecture, and the Mixture of Experts (MoE) Transformer, which employs a specialized architecture with multiple expert networks. These two variants differ in their approaches to parameter utilization and model computation during the learning and inference processes.

Transfer Learning

In the field of Natural Language Processing (NLP), transfer learning can generally be categorized into two main types. The first is Transfer Learning, in which the task remains the same while the domain or language differs, as commonly observed in domain adaptation and cross-lingual learning scenarios. The second is Inductive Transfer Learning, in which the source and target tasks are different. Inductive Transfer Learning encompasses approaches such as Multi-Task Learning, where a single model is trained to perform multiple tasks simultaneously, and Sequential Transfer Learning (Zulfa et al., 2025). In the sequential approach, a model is first pre-trained on a large-scale dataset and subsequently adapted or fine-tuned for a specific downstream task. This strategy enables the model to leverage knowledge acquired during pre-training and transfer it effectively to new tasks with limited labeled data.

Mean Pooling

Mean pooling is an aggregation method used to transform token-level representations generated by attention-based models, such as Transformers, into a sentence-level representation. This technique is commonly employed in the development of Sentence Transformers to ensure that padding tokens are excluded from the averaging process, thereby producing more meaningful sentence embeddings.

Let:

$H = \{\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_l\}$ denote the set of token representations, where $\mathbf{h}_i \in \mathbb{R}^d$

$M = \{m_1, m_2, \dots, m_l\}$ denote the attention mask, where $m_i \in \{0, 1\}$

l represent the sequence length, d represent the dimensionality of the vector representation and $i = \{1, 2, 3, \dots, l\}$ denote the token index within the sequence.

The masked mean pooling operation is defined as follows:

$$\text{Mean Pooling}(H, M) = \frac{\sum_{i=1}^l m_i h_i}{\sum_{i=1}^l m_i} \quad (1)$$

This formulation computes the sum of the representations corresponding to valid tokens (i.e., tokens for which $m_i = 1$) and divides it by the total number of valid tokens. Consequently, padding tokens with $m_i = 0$ do not contribute to the aggregated representation, ensuring that the resulting sentence embedding accurately reflects the semantic content of the input text.

Valuation Metrics

In the context of sentiment classification, evaluation metrics are used to measure the performance of a model in categorizing text into positive, neutral, or negative sentiment classes. In this study, accuracy and macro F1-score are employed as the primary evaluation metrics (Cipta et al., 2024).

1. Accuracy

Accuracy measures the proportion of correctly classified instances relative to the total number of instances in the dataset. It provides an overall indication of the model's classification performance.

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Ins}} \quad (2)$$

2. Precision, Recall and F1-score

To compute the macro F1-score, the values of precision, recall, and F1-score are first calculated for each sentiment class (positive, neutral, and negative) as follows:

$$\text{Precision} = \frac{TP_i}{TP_i + FP_i} \quad (3)$$

$$\text{Recall} = \frac{TP_i}{TP_i + FN_i} \quad (4)$$

$$\text{F1-score} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \quad (5)$$

where:

- i : denotes the sentiment class, where $i \in \{\text{positif}, \text{netral}, \text{negatif}\}$
- TP_i : represents the number of correctly predicted instances belonging to class i
- FP_i : represents the number of instances incorrectly classified as class i
- FN_i : represents the number of instances belonging to class i that were incorrectly classified as another class (False Negatives)

3. Macro F1-score

The Macro F1-score is the arithmetic mean of the F1-scores computed for all classes, giving equal weight to each class regardless of the number of instances in that class. This metric is particularly useful for evaluating model performance on datasets with imbalanced class distributions.

$$\text{Macro F1-Score} = \frac{1}{N} \sum_{i=1}^N \text{F1 Score}_i \quad (6)$$

where:

i : denotes the class, where $i \in \{\text{positif, netral, negatif}\}$

N : represents the total number of classes; and

F1-score_i : denotes the F1-score of class i

3. METHOD

The overall research workflow is illustrated in the following diagram:

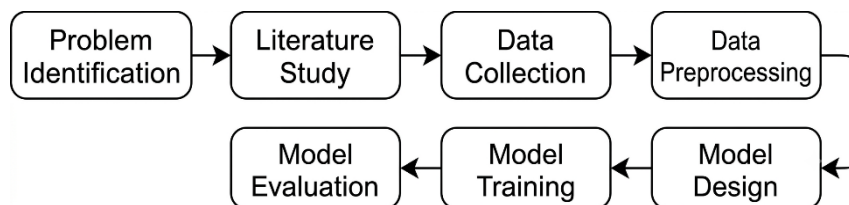


Figure 1. Research Workflow

Problem Identification

Although the No Language Left Behind (NLLB) model supports several Indonesian regional languages, it was primarily developed for machine translation tasks. Its potential for other NLP applications, such as sentiment classification, has not yet been thoroughly explored.

Literature Review

Relevant literature was collected from various sources to gain a comprehensive understanding of the NLLB architecture and the potential use of its encoder component for NLP tasks such as text classification. This review also examined previous studies demonstrating that encoders from Neural Machine Translation (NMT) models can be effectively utilized for text classification through transfer learning approaches.

Data Collection

The dataset used in this study is the sentiment analysis subset of the NusaX dataset collection. Since the data for each language are provided separately and the proposed model is designed as a multilingual classification model trained simultaneously on all 12 languages (multilingual training), a data integration process was required. Specifically, the datasets from each language were concatenated into a unified dataset for each data split, namely the training, validation, and testing sets. This process ensured that the multilingual model could be trained and evaluated using a consolidated dataset containing samples from all supported languages.

Data Preprocessing

The data preprocessing stage in this study consisted of the following steps:

1. Label Mapping

Sentiment labels originally represented as text, namely positive, neutral, and negative, were converted into numerical values (0, 1, and 2) to enable processing by the classification model.

2. Tokenization

Tokenization was performed using the NLLB tokenizer, which also handled truncation of sequences exceeding the maximum input length and padding to ensure uniform input lengths across samples. The output of the tokenization process consisted of input IDs representing individual tokens and an attention mask used to distinguish actual tokens (1) from padding tokens (0).

3. Tensor Dataset Construction

The tokenized outputs and numerical labels were subsequently organized into a `TensorDataset` using utilities provided by the PyTorch framework. This tensor-based representation serves as the standard data format for efficient processing and training on Graphics Processing Units (GPUs).

4. DataLoader Preparation

A `DataLoader` was employed to divide the dataset into batches and randomly shuffle the data during training. This process improves training efficiency and helps enhance the model's generalization capability by exposing it to data in varying orders across training iterations.

Model Design

Two main approaches were implemented in this study: fine-tuning and partial fine-tuning. In the fine-tuning approach, all parameters of the NLLB encoder were updated during training along with a classification head consisting of a fully connected (FC) layer, which maps the encoder-generated vector representations to the corresponding sentiment classes.

Model Training

Model training was conducted separately for the two approaches: fine-tuning and partial fine-tuning, where the embedding layer was frozen in the latter approach. All training processes were executed on a Graphics Processing Unit (GPU) using the Google Colaboratory platform to accelerate optimization. The implementation was carried out using the PyTorch framework.

Model Evaluation

The performance of the proposed models was evaluated using two standard metrics for text classification tasks: accuracy and macro F1-score. These metrics were employed to assess the effectiveness of the training approaches and to provide a comprehensive evaluation of the models' classification performance.

4. RESULT AND DISCUSSION

Dataset Collection

At this stage, the NusaX Sentiment dataset was utilized, which was obtained from the IndoNLP GitHub repository. The dataset covers 12 languages, including Indonesian, English, and 10 Indonesian regional languages: Acehnese, Banjarese, Balinese, Buginese, Minangkabau, Javanese, Sundanese, Madurese, Ngaju, and Toba Batak.

Data Preprocessing

The data preprocessing stage consisted of the following steps:

1. Label Mapping

This stage was performed to convert sentiment labels originally represented as text (positive, neutral, and negative) into numerical representations that could be processed by the model. The labels were mapped as follows: negative = 0, neutral = 1, and positive = 2.

2. Tokenization

Tokenization was carried out using the tokenizer provided by the NLLB-200-distilled-600M model, which has been pretrained and is capable of recognizing multiple Indonesian regional languages. This process transforms the input text into token representations that can be processed by the model.

3. Tensor Dataset Construction

After tokenization, the processed data were converted into tensor format to enable efficient processing within the PyTorch framework. This stage combined three main components input IDs, attention masks, and labels into a data structure known as a `TensorDataset`.

4. Data Loader Preparation

This process generated the train_dataloader, val_dataloader, and test_dataloader. Each data loader consisted of multiple mini-batches, with each mini-batch containing up to 32 data samples. This batching strategy facilitates efficient model training and evaluation.

Model Design

In this study, two sentiment classification models were developed based on the encoder component of the NLLB-200-distilled-600M model. Each model was equipped with an additional classification layer and implemented using two different training strategies: fine-tuning and partial fine-tuning.

1. Model 1 - Fine-Tuning

The first model was designed using the fine-tuning approach, in which all components of the model including the embedding layer, encoder, and classification head were included in the parameter update process. Consequently, all model parameters were optimized during training, allowing the pretrained representations to be fully adapted to the sentiment classification task.

Layer (type:depth-idx)	Param #
NllbClassifier	--
~H2M100Encoder: 1-1	--
~H2M100ScaledWordEmbedding: 2-1	262,354,944
~H2M100SinusoidalPositionalEmbedding: 2-2	--
~ModuleList: 2-3	--
~H2M100EncoderLayer: 3-1	12,596,224
~H2M100EncoderLayer: 3-2	12,596,224
~H2M100EncoderLayer: 3-3	12,596,224
~H2M100EncoderLayer: 3-4	12,596,224
~H2M100EncoderLayer: 3-5	12,596,224
~H2M100EncoderLayer: 3-6	12,596,224
~H2M100EncoderLayer: 3-7	12,596,224
~H2M100EncoderLayer: 3-8	12,596,224
~H2M100EncoderLayer: 3-9	12,596,224
~H2M100EncoderLayer: 3-10	12,596,224
~H2M100EncoderLayer: 3-11	12,596,224
~H2M100EncoderLayer: 3-12	12,596,224
~LayerNorm: 2-4	2,048
~Dropout: 1-2	--
~Linear: 1-3	3,075
Total params: 413,514,755	
Trainable params: 413,514,755	
Non-trainable params: 0	

Figure 2. Model 1 Using the Fine-Tuning Approach

Based on the model summary shown in Figure 2, the number of total parameters and trainable parameters is identical, indicating that no part of the model was frozen during training. With this design, all components of the model contribute to the learning process on the multilingual sentiment dataset. This allows the representations generated by the encoder to be fully adapted to the characteristics of the regional languages and the sentiment classification task considered in this study, thereby enabling optimal task-specific feature learning.

2. Model 2 - Partial Fine-Tuning

The second model was designed using a partial fine-tuning approach, in which the embedding layer was frozen, while the encoder and classification head remained trainable. As illustrated in Figure 3, the parameters belonging to the embedding layer are indicated by the values shown in parentheses and are entirely categorized as non-trainable parameters. This value corresponds to the total number of embedding parameters, confirming that this component was not updated during the training process. Meanwhile, the remaining components, including the encoder and classification head, continued to be optimized during training, resulting in approximately 151 million trainable parameters. This design allows the model to preserve the pretrained lexical representations learned by the embedding layer while adapting the higher-level representations to the multilingual sentiment classification task.

```

Layer (type:depth-ids)                               Param #
-----
NllbClassifier                                         --
├─H2M1000ScaledDotProductAttention: 2-1              (262,354,944)
├─H2M1005InpositionalEmbedding: 2-2                 --
├─ModuleList: 2-3                                     --
│   ├─H2M1000EncoderLayer: 3-1                      12,596,224
│   ├─H2M1000EncoderLayer: 3-2                      12,596,224
│   ├─H2M1000EncoderLayer: 3-3                      12,596,224
│   ├─H2M1000EncoderLayer: 3-4                      12,596,224
│   ├─H2M1000EncoderLayer: 3-5                      12,596,224
│   ├─H2M1000EncoderLayer: 3-6                      12,596,224
│   ├─H2M1000EncoderLayer: 3-7                      12,596,224
│   ├─H2M1000EncoderLayer: 3-8                      12,596,224
│   ├─H2M1000EncoderLayer: 3-9                      12,596,224
│   ├─H2M1000EncoderLayer: 3-10                     12,596,224
│   ├─H2M1000EncoderLayer: 3-11                     12,596,224
│   └─H2M1000EncoderLayer: 3-12                     12,596,224
└─LayerNorm: 2-4                                     2,048
Dropout: 1-2                                         --
Linear: 1-3                                         3,875
-----
Total params: 413,514,755
Trainable params: 151,159,811
Non-trainable params: 262,354,944

```

Figure 3. Model 2 Using the Partial Fine-Tuning Approach

This design ensures that the fundamental representations learned during the pretraining phase of the NLLB model are preserved, while the upper layers remain adaptable to the sentiment data used in this study. By freezing the embedding layer and updating only the encoder and classification head, the model can retain its original lexical knowledge while learning task-specific features relevant to multilingual sentiment classification.

Model Training

This section presents the training results obtained from the two models developed in this study, namely the model employing the full fine-tuning approach and the model employing the partial fine-tuning approach.

1. Training Results of Model 1 (Fine-Tuning)

The first model was trained using the full fine-tuning approach, in which all model parameters were updated throughout the training process. Training was conducted for a maximum of 10 epochs, with an early stopping mechanism applied using a patience value of 3 to prevent overfitting and improve training efficiency.

Table 2. Training History of Model 1

Epoch	Training loss	Validation loss	Training accuracy	Validation accuracy	Early stopping
1	0.7878	0.6808	0.66	0.72	Save
2	0.4959	0.5856	0.81	0.77	Save
3	0.3528	0.5770	0.87	0.78	Save (Best)
4	0.2565	0.5993	0.91	0.79	1/3
5	0.1858	0.6738	0.93	0.79	2/3
6	0.1336	0.7804	0.95	0.80	3/3

Based on the results presented in Table 2, the training loss decreased consistently across epochs, indicating that the learning process progressed effectively. Meanwhile, the validation loss reached its lowest value of 0.5770 at Epoch 3 before increasing in subsequent epochs.

The early stopping mechanism employed in this study monitored the model's performance on the validation set and automatically saved the best model weights whenever a lower validation loss was achieved. After Epoch 3, the validation loss no longer improved,

prompting the early stopping mechanism to begin counting the predefined patience period of three epochs. Since no improvement in validation performance was observed for three consecutive epochs (Epochs 4-6), the training process was automatically terminated to prevent overfitting and preserve the model parameters corresponding to the best validation performance.

2. Training Results of Model 2 (Partial Fine-Tuning)

Model 2 was developed using a partial fine-tuning approach, in which only a subset of the encoder parameters was updated during training, while the embedding layer remained frozen. The training process was conducted for a maximum of 10 epochs with an early stopping mechanism configured with a patience value of 3.

Table 3. Training History of Model 2

Epoch	Training loss	Validation loss	Training accuracy	Validation accuracy	Early stopping
1	0.7758	0.6527	0.67	0.74	Save
2	0.4790	0.5924	0.82	0.78	Save (Best)
3	0.3482	0.5944	0.87	0.78	1/3
4	0.2529	0.6626	0.91	0.79	2/3
5	0.1856	0.7364	0.93	0.79	3/3

Based on the results presented in Table 3, the training loss decreased consistently across epochs, indicating a stable learning process. In contrast, the validation loss reached its minimum value at Epoch 2 before increasing in subsequent epochs, suggesting the onset of overfitting after that point. The early stopping mechanism subsequently terminated the training process after no improvement in validation performance was observed for the specified patience period. The best model was automatically saved at Epoch 2, when the validation loss achieved its lowest value. Therefore, although the training process continued until the stopping criterion was met, the model used for subsequent evaluation was the checkpoint saved at Epoch 2, corresponding to the optimal validation performance.

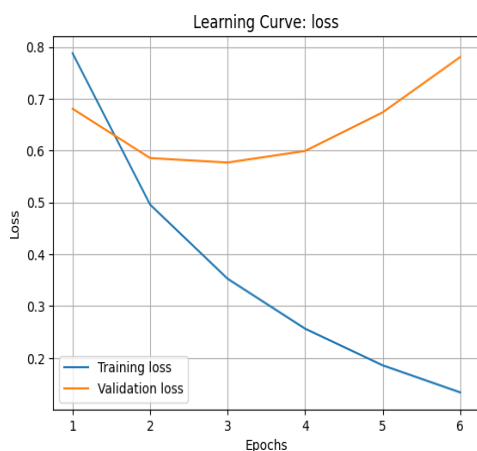


Figure 4. Learning Curves of Model 1 Training

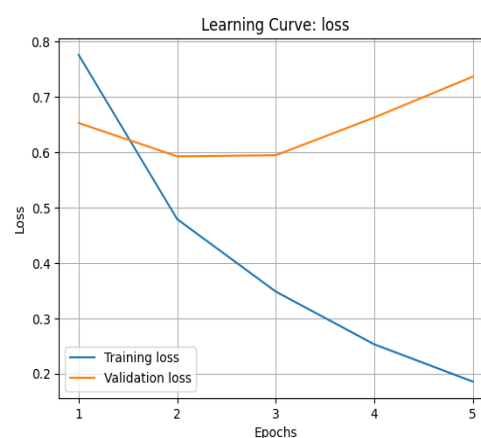


Figure 5. Learning Curves of Model 2 Training

Figure 4 presents the learning curve of Model 1, which facilitates the diagnosis of the training process by illustrating the changes in training loss and validation loss across epochs.

As shown in the figure, the training loss decreases consistently throughout the training process, indicating that the model successfully learns patterns from the training data. Meanwhile, the validation loss reaches its minimum value at Epoch 3, which is consistent with the results reported in Table 2. After this point, the validation loss begins to increase while the training loss continues to decrease, suggesting the onset of overfitting if training were to continue. Therefore, the learning curve demonstrates that the early stopping mechanism with a patience value of 3 effectively prevents overfitting and ensures that training terminates at the point of optimal generalization.

Figure 5 illustrates the learning curve of Model 2, showing the progression of training loss and validation loss during the training process. The figure indicates that the training loss decreases consistently as the number of epochs increases, demonstrating that the model gradually improves its fit to the training data. However, a different pattern is observed for the validation loss, which decreases until Epoch 2 and then begins to increase in subsequent epochs.

The increase in validation loss after Epoch 2 suggests that the model starts to lose its ability to generalize to unseen data. The early stopping mechanism with a patience value of 3 allows training to continue until Epoch 5 to verify whether the increase in validation loss is merely a temporary fluctuation. After three consecutive epochs without improvement in validation loss, the training process is automatically terminated, and the best-performing model is retained from Epoch 2, where the validation loss reached its minimum value of 0.5924.

Model Evaluation

The evaluation process was conducted using accuracy and macro F1-score, which are widely adopted metrics for multi-class classification tasks, particularly when the class distribution is imbalanced. These metrics provide a comprehensive assessment of model performance by measuring both the overall classification accuracy and the balance between precision and recall across all sentiment classes.

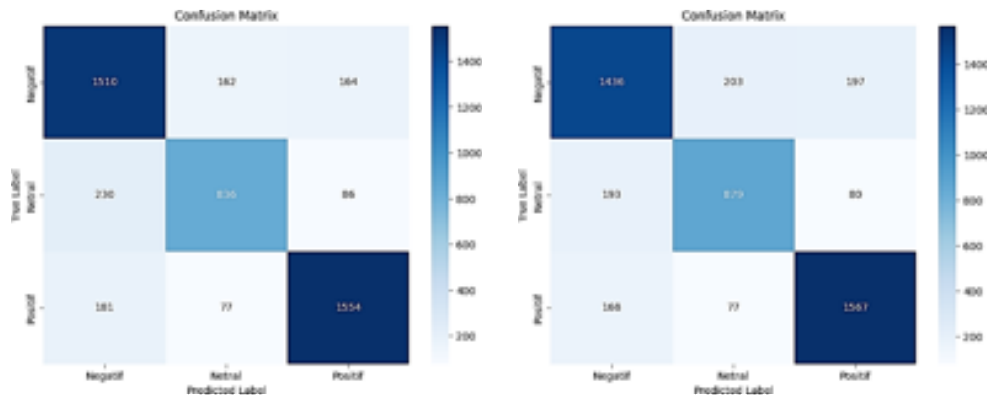
1. Evaluation on the Multilingual Dataset

The evaluation was conducted to measure the performance of both models in performing sentiment classification on the test set consisting of 4,800 samples from 12 languages.

Table 4. Evaluation Results of the Models on the Multilingual Dataset

Model	Accuracy(%)	Macro-F1(%)
Fine-tuning (model 1)	81	80
Partial fine-tuning (model 2)	81	80

Based on the results presented in Table 4, both models achieved comparable performance on the multilingual dataset, obtaining an accuracy of 81% and a macro F1-score of 80%. These results indicate that both the full fine-tuning and partial fine-tuning approaches are capable of generalizing effectively to the test data encompassing multiple languages.



(a) Confusion matrix model 1

(b) Confusion matrix model 2

Figure 6. Confusion Matrices of the Models on the Multilingual Dataset

The classification results on the multilingual test set are visualized in Figure 6, which presents the confusion matrices for the fine-tuning model (a) and the partial fine-tuning model (b). These visualizations provide a more detailed view of the classification outcomes and enable the analysis of misclassification patterns among the sentiment classes.

2. Language-Specific Model Evaluation

A further evaluation was conducted to assess model performance on each language individually. The purpose of this analysis was to examine the extent to which both models are able to generalize across the different languages included in the dataset. The per-language evaluation results are presented in Table 5, which compares the performance of the fine-tuning and partial fine-tuning models across the 12 languages contained in the NusaX Sentiment dataset. This analysis provides insights into the effectiveness of each training approach for individual languages and highlights potential variations in performance across linguistic contexts.

Table 5. Language-Specific Evaluation Results

Language	Fine-tuning		Partial fine-tuning	
	Accuracy (%)	Macro-F1 (%)	Accuracy (%)	Macro-F1 (%)
Aceh	83	82	81	81
Bali	79	79	79	78
Banjar	84	83	83	82
Bugis	78	76	75	74
Batak toba	70	70	71	71
Inggris	88	87	89	88
Indonesia	88	87	87	86
Jawa	83	83	84	84
Madura	78	77	78	77
Minangkabau	81	80	83	82
Ngaju	76	75	75	74
Sunda	86	85	85	85

Average Across Regional Languages	79,8	79	79,4	78,8
Overall Average	1,2	80,3	80,8	80,2

Overall, both approaches demonstrated relatively comparable performance. The full fine-tuning model achieved an average accuracy of 81.2% and a macro F1-score of 80.3%, whereas the partial fine-tuning model obtained an accuracy of 80.8% and a macro F1-score of 80.2%. The similarity of these results indicates that freezing part of the encoder does not lead to a substantial degradation in performance for the multilingual sentiment classification task.

When considering only the regional languages (excluding Indonesian and English), the average performance of both models decreased slightly compared to the overall results. For the full fine-tuning approach, the average accuracy across regional languages reached 79.8%, with a macro F1-score of 79.0%. Meanwhile, the partial fine-tuning approach achieved an average accuracy of 79.4% and a macro F1-score of 78.8%. These findings suggest that the linguistic complexity and structural diversity among regional languages continue to pose challenges to the model's generalization capability.

In terms of performance distribution, the highest results were achieved on languages that were included in the original NLLB pretraining corpus, namely Acehnese, Balinese, Banjarese, Buginese, Minangkabau, Javanese, and Sundanese. These languages generally attained macro F1-scores ranging from 79% to 85%, indicating that the representations learned during NLLB pretraining can be effectively transferred to sentiment classification tasks. In contrast, languages that were not included in the NLLB training set, such as Toba Batak, Madurese, and Ngaju, exhibited slightly lower performance, with macro F1-scores ranging from approximately 71% to 77%.

Comparative Analysis

Based on the evaluation results on the combined dataset, both models demonstrated relatively comparable performance. The fine-tuning model achieved an accuracy of 81% and a macro-F1 score of 80%, while the partial fine-tuning model attained the same results, with an accuracy of 81% and a macro-F1 score of 80%. These findings indicate that the NLLB encoder possesses strong cross-lingual representation capabilities, enabling both full and partial training strategies to effectively leverage the semantic knowledge acquired during the pre-training phase.

A language-level analysis reveals that model performance was generally higher for languages included in the NLLB training coverage, such as Acehnese, Balinese, Banjarese, Minangkabau, Javanese, and Sundanese, with accuracy ranging from 79% to 86%. Across all regional languages, the fine-tuning model achieved an average accuracy of 79.8% and a macro-F1 score of 79%, whereas the partial fine-tuning model obtained an average accuracy of 79.4% and a macro-F1 score of 78.8%. In contrast, several languages that were likely underrepresented or not included in the original NLLB training corpus, such as Ngaju, Madurese, and Toba Batak, exhibited slightly lower performance, with accuracy ranging from 70% to 78%. This suggests that the model's transfer capability remains somewhat limited for languages outside its original training coverage.

Overall, the results indicate that both approaches provide comparable performance for multilingual sentiment classification tasks. Nevertheless, the fine-tuning approach offers greater flexibility and adaptation potential, whereas partial fine-tuning is computationally more efficient because it requires updating fewer parameters. Therefore, the choice between these approaches can be tailored to the available computational resources and data availability in different deployment scenarios.

5. CONCLUSION

Based on the results and analyses conducted in this study, the following conclusions can be drawn:

1. The evaluation results demonstrate that the NLLB encoder is capable of generating effective representations for texts across multiple languages, including both the national language and regional languages. When evaluated on the combined multilingual dataset, the proposed model achieved strong performance, with an accuracy of 81% and a macro-F1 score of 80%, indicating the encoder's ability to generalize effectively across diverse languages.
2. The language-specific evaluation revealed that the highest performance was achieved for languages that were included in the NLLB training coverage, such as Acehnese, Balinese, Banjarese, Buginese, Minangkabau, Javanese, and Sundanese, with accuracy ranging from approximately 79% to 86%. These findings suggest that the transfer capabilities of the NLLB encoder are more effective for languages that were represented during the pre-training phase.
3. Overall, the findings of this study highlight the significant potential of the NLLB encoder for sentiment classification tasks involving Indonesian regional languages. Furthermore, the results emphasize the importance of selecting an appropriate fine-tuning strategy based on the characteristics of the available data and the computational resources allocated for training.

6. RECOMMENDATIONS

Based on the findings of this study, several recommendations can be proposed for future research:

1. The dataset used in this study remains limited in size. Future research is therefore encouraged to expand the training data by collecting additional samples from authentic sources, such as social media platforms, local news outlets, and other relevant media. Furthermore, data augmentation techniques, including translation-based approaches (e.g., back-translation) and generative methods (e.g., synthetic text generation), may serve as effective alternatives for enriching the training corpus and improving model performance.
2. Although both fine-tuning and partial fine-tuning approaches yielded promising results, there remains substantial room for exploring alternative training strategies. Future studies may investigate the use of learning rate scheduling, more adaptive layer-freezing strategies, and additional regularization techniques, such as dropout and weight decay, to enhance training stability and mitigate overfitting.
3. The model developed in this study has strong potential for practical applications in sentiment analysis of Indonesian regional languages, particularly in areas such as public opinion monitoring and social research conducted in local languages. In addition, the model can serve as a base model for data augmentation processes in future studies, facilitating the generation of additional training data and potentially improving the performance of sentiment classification systems for regional-language texts.

7. REFERENCES

- Akula, B., Barrault, L., Gonzalez, G. M., Hansanti, P., & Hoffman, J. (2022). No language left behind: Scaling human-centered machine translation. *Meta AI*. <https://arxiv.org/abs/2207.04672>
- Cipta, S. P., Zulfadhilah, M., Naparin, H., & Syapotro, U. (2024). Analisis sentimen terhadap aplikasi Parak Acil Online berdasarkan ulasan masyarakat menggunakan metode Support Vector Machine (SVM). 7(5), 1042–1049.
- Dewi, S., & Eka, B. (2025). Pengukuran kemiripan kalimat bahasa Indonesia menggunakan representasi word embedding FastText. 2(2), 20–29.

- Eriguchi, A. (2019). Incorporating source-side phrase structures into neural machine translation. *Computational Linguistics*. <https://doi.org/10.1162/COLI>
- Ishak, S. I., Arnilia, O., Widodo, T., Gunawan, I. N., & Bisma, A. (2025). Analisis sentimen terhadap pemerintahan Prabowo–Gibran menggunakan IndoBERT dan LDA. 7(2), 72–82. <https://doi.org/10.37905/jji.v1i2.34895>
- Raffel, C., Roberts, A., Matena, M., & Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21, 1–67.
- Sucholutsky, I., Narayan, A., & Schonlau, M. (2019). Pay attention and you won't lose it: A deep learning approach to sequence imputation. *PeerJ Computer Science*. <https://doi.org/10.7717/peerj-cs.210>
- Suhaida, D., Wulan, A. P., Rosanti, R., & Dianna, D. (2024). Evaluasi efektivitas model klasifikasi sentimen untuk analisis opini publik terhadap kebijakan lingkungan berdasarkan data media sosial berbahasa Indonesia. 2.
- Tsalitsatun, A., Yunitarini, R., & Nawawi, S. (2024). Pembelajaran pendidikan agama Islam melalui sistem pencarian artikel berbasis Bidirectional Encoder Representations from Transformers (BERT). 4(2), 187–195.
- Utami, N. W., Purnama, I. N., & Prayoga, I. M. A. (2024). Sentiment analysis system of Bali tourism using Naive Bayes algorithm and web framework. *Jurnal Info Sains: Informatika dan Sains*, 14(3), 275–282.
- Zulfa, I., Studi, P., Informatika, T., Teknik, F., & Gajah, U. (2025). Sistem pendataan mahasiswa dan dosen berprestasi menggunakan metode Weighted Product PROMETHEE. 8(6), 3354–3366.