Application of Neural Machine Translation with Attention Mechanism for Translation of Indonesian to Seram Language (Geser)

Abdul Wahid Rukua¹, Yopi Andry Lesnussa², Dorteus Lodewyik Rahakbauw³*, Berny Pebo Tomasouw⁴

¹²³⁴ Mathematic Program Study, Faculty of Mathematics and Natural Science, Universitas Pattimura
Jln. Ir. M. Putuhenna, Ambon, 97233, Indonesia

Corresponding author's e-mail: ³* lodewyik@gmail.com

ABSTRACT

The Seram language (Geser) is one of the regional languages in Kabupaten Seram Bagian Timur of Maluku Province which has been classified by the Language Office as an endangered language. This study uses the Neural Machine Translation (NMT) method in an effort to preserve the Seram (Geser) language. The NMT method has proven to be effective compared to SMT in overcoming the challenges of language translation by using the attention mechanism to improve translation accuracy. The data used in this study were obtained through interviews of 3538 parallel corpus, 255 Indonesian vocabularies and 269 Seram (Geser) vocabularies. The result showed that using 708 test data without Out-of Vocabulary (OOV) the BLUE Score was 0.90518992895191 or 90.518%.

This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution-NonCommercial 4.0 International License. Editor of PIJMath, Pattimura University

1 How to cite this article:
© 2023 by the Author(s)

e-mail: pijmath.journal@mail.unpatti.ac.id
Homepage https://ojs3.unpatti.ac.id/index.php/pijmath

53
1. Introduction

Humans need communication to live and one of the communication tools is language [1]. Language is a form of crystallization of civilizational values and has an important role in directing the movement of civilization [2]. With the development of language studies, there is an increase in understanding and level of thought in a civilization [3]. Therefore, it is very important to learn a language in order to understand the culture, values and social background of the people who speak the language [4].

Cultural and linguistic diversity has long been one of Indonesia's wealth [5]. There is no exact information regarding the total number of languages spoken in the world, but it is estimated that about ten percent of them originate from Indonesia [6]. There are 726 languages in Indonesia [7], or according to the Language Development and Bookkeeping Agency of the Ministry of Education and Culture in 2019 there are 718 languages from 1340 ethnicities and tribes in Indonesia. This ethnic and cultural diversity creates linguistic phenomena such as bilingualism and language transformation until language extinction occurs [8].

The inability of languages in the eastern part of Indonesia to adapt to the dynamics of the times has made these languages vulnerable to extinction [9]. In the Maluku Province, Maluku Utara and Papua there are 11 out of 71 languages that have been declared extinct [10]. Meanwhile, according to the Maluku Provincial Language Office, it states that there are 22 languages in Maluku, one of which is the Seram language (Geser) [11].

Machine Translation is a technology that automatically converts from one language to another [12]. Recently, machine translation has achieved significant success [13]. Statistical Machine Learning (SMT) that has dominated machine learning research over the last few decades has been taken in its place by NMT (Neural Machine Translation) [14]. From a study of 50 million to 10 million paired sentences, NMT is superior to SMT [15].

NMT (Neural Machine Translation) is an end-to-end learning method for translating automatically [16]. The main form of NMT consists of two main components: (1) The encoder calculates the representation c of each input sequence (2) Decoder that guesses word by word using information from c which is formulated as follows:

$$\log p(y|x) = \sum_{j=1}^{M} \log p(y_j|y_{<j}, c)$$

(1)

To model this equation, an RNN (Recurrent Neural Network) architecture can be used. The main components required in the BLSTM architecture are the same as NMT in general with a slight difference, namely: (1) Encoder (2) Global Attention (3) Local Attention (4) Decoder [20].

2. Research Methods

2.1 Neural Machine Translation

The main form of NMT consists of two main components: (1) Encoder calculates the representation c of each input sequence (2) Decoder which guesses word by word using the information from c which is formulated as follows:

$$\log p(y|x) = \sum_{j=1}^{M} \log p(y_j|y_{<j}, c)$$

To model this equation, an RNN (Recurrent Neural Network) architecture can be used. The main components required in the BLSTM architecture are the same as NMT in general with a slight difference, namely: (1) Encoder (2) Global Attention (3) Local Attention (4) Decoder [20].

2.2 Research Subject

The subject of this research is a parallel corpus of Indonesian to Seram language (Geser) dialect spoken by people in Kabupaten Seram Bagian Timur (SBT), Provinsi Maluku.

2.3 Data Collection Techniques

The research data collection used interview method to obtain the data needed in the research. Researchers collected Geser language sentences and their translations by conducting interviews with 10 respondents (Geser language speakers).

2.4 The Tools Used

The tools used in this study are divided into several parts as follows:

- **a.** NMT training platform: the platform used is Google Colab which is available for free and paid on the internet.
- **b.** Hardware: the hardware used is 1 unit of 4.00 GB RAM computer with Intel ® Core ™ i3-6100 U Processor with CPU @ 2.30 Hz and built-in GPU from Google Colab, namely NVIDIA Tesla K80 which can only be used in 12 hours per day.
- **c.** Deep Learning Framework: The framework used is Tensorflow.
2.5 Conceptual Framework

Figure 1 illustrates the stages of NMT research starting from the data collection stage to the conclusion stage.

2.6 Data Preprocessing

Before training, the dataset will be prepared in order to maximize the results of the training. This process consists of Data Sharing, Batching, Cleaning, Tokenizing and Text Vectorization, Word Dictionary Creation, and Padding.

2.7 NMT System Design

The Neural Machine Translation (NMT) architecture used is Bidirectional Encoder-Decoder RNN for Geser Language to Indonesian translation. The process consists of parallel corpus input, pre-processing, model initialization, training, model evaluation, parameter tuning, best model and calculate BLEU Score.
3. Results And Discussion

The data obtained from the interviews amounted to 3538 parallel corpus, 255 Indonesian vocabularies and 269 Slang vocabularies which were used as the basis for testing the developed Neural Machine Translation (NMT) system.

3.1 Parallel Corpus Description

The parallel corpus contains pairs of original sentences to translated sentences consisting of several sentence types, namely: Verb Phrases and Verb Clauses. Here is an example of a parallel corpus of Indonesian – Geser language:

<table>
<thead>
<tr>
<th>No</th>
<th>Indonesia</th>
<th>Geser</th>
<th>Phrases / Clauses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Belajar Bahasa</td>
<td>Balajar Minak</td>
<td>Verb Phrase</td>
</tr>
<tr>
<td>2</td>
<td>Mengambil Pena</td>
<td>Na Kalam</td>
<td>Verb Phrase</td>
</tr>
<tr>
<td>3</td>
<td>Saya makan roti</td>
<td>Aku ka barot</td>
<td>Clause Verb</td>
</tr>
<tr>
<td>4</td>
<td>Ibu pergi ke pasar</td>
<td>Nina na tagi bua pasar</td>
<td>Clause Verb</td>
</tr>
<tr>
<td>5</td>
<td>Adik sedang belajar aljabar</td>
<td>Toi na balajar aljabar liwa</td>
<td>Clause Verb</td>
</tr>
</tbody>
</table>

| Dataset Total | 3538 |

Source: Primary data of interview results

The following is a visualization of the word length distribution in the Indonesian – Geser Language parallel corpus:
On Figure 3, showing the difference in linguistic structure between Indonesian and Seram language (Geser) is very striking between these two languages. This is in accordance with what is shown in Table 1 where the grammar, words, vocabulary, syntax, and sentence structure of these two languages are different.

3.2 Research Data Preparation

Before training there are several data preprocessing steps that must be done as follows:

1. Splitting Dataset

   In this process the dataset is divided into two parts, namely for training and validation. This process is carried out using the library from Scikit-learn where the dataset is divided into 60% for training, 20% for validation and 20% for testing shown in Figure 4 as follows:

2. Batching

   Effective batching is 32 and 64. In this process, batching will be done with the Tensorflow Library. Here is the syntax:

   ```python
   BATCH_SIZE = 32
   train_raw = tf.data.Dataset.from_tensor_slices((context_raw, target_raw)).batch(BATCH_SIZE)
   val_raw = tf.data.Dataset.from_tensor_slices((context_raw, target_raw)).batch(BATCH_SIZE)
   ```

3. Data Cleaning

   In this process, the data is cleaned from errors such as typos, missing or duplicate data, and invalid or meaningless data. The data cleaning process is done using the following syntax:

   ```python
   def tf_lower_and_split_punct(text):
       text = tf_text.normalize_utf8(text, 'NFKD')
       text = tf.strings.lower(text)
       text = tf.strings.regex_replace(text, r'[\^ a-z0-9.?!,¿\-+=]', '')
   ```
4. Word Dictionary Creation
In this process the dataset is sorted as unique words or tokens of a sentence. The following is the word dictionary
generation syntax:

```python
context_vocab = np.array(context_text_processor.get_vocabulary())
tokens = context_vocab[example_tokens[0].numpy()].join(tokens)
```

5. Padding
In this process, datasets have different lengths. When padding is applied, a number of "padding" tokens will be added
to the input sequence, so that all input sequences have the same length.

```python
train_ds = train_raw.map(process_text, tf.data.AUTOTUNE)
val_ds = val_raw.map(process_text, tf.data.AUTOTUNE)
```

3.3 NMT Training

1. Training Simulation
In this study, several training simulations will be tried to evaluate the performance of the model. The following
hyperparamater will be used to obtain the best model:

<table>
<thead>
<tr>
<th>No</th>
<th>Activation Function</th>
<th>Unit</th>
<th>Batch Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>RMSprop</td>
<td>128</td>
<td>32</td>
</tr>
<tr>
<td>2</td>
<td>Adagrad</td>
<td>256</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Adam</td>
<td>300</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>512</td>
<td></td>
</tr>
</tbody>
</table>

**Table 2. Hyperparameter**

<table>
<thead>
<tr>
<th>Activation Function</th>
<th>Unit</th>
<th>Loss</th>
<th>Masked Acc</th>
<th>Masked Loss</th>
<th>Val Loss</th>
<th>Val Masked Acc</th>
<th>Val Masked Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMSprop</td>
<td>128</td>
<td>0.0456</td>
<td>0.9908</td>
<td>0.0456</td>
<td>0.1129</td>
<td>0.9797</td>
<td>0.1169</td>
</tr>
<tr>
<td>RMSprop</td>
<td>256</td>
<td>0.0650</td>
<td>0.9849</td>
<td>0.0650</td>
<td>0.1447</td>
<td>0.9708</td>
<td>0.1145</td>
</tr>
<tr>
<td>RMSprop</td>
<td>300</td>
<td>0.0481</td>
<td>0.9919</td>
<td>0.0481</td>
<td>0.1889</td>
<td>0.9587</td>
<td>0.1868</td>
</tr>
<tr>
<td>RMSprop</td>
<td>512</td>
<td>0.0955</td>
<td>0.9790</td>
<td>0.0955</td>
<td>0.2002</td>
<td>0.9529</td>
<td>0.2046</td>
</tr>
<tr>
<td>Adagrad</td>
<td>128</td>
<td>2.7273</td>
<td>0.4257</td>
<td>2.7273</td>
<td>2.7629</td>
<td>0.4409</td>
<td>2.7756</td>
</tr>
<tr>
<td>Adagrad</td>
<td>256</td>
<td>2.3587</td>
<td>0.4978</td>
<td>2.3587</td>
<td>2.4114</td>
<td>0.4894</td>
<td>2.4230</td>
</tr>
<tr>
<td>Adagrad</td>
<td>300</td>
<td>2.2671</td>
<td>0.5110</td>
<td>2.2671</td>
<td>2.3164</td>
<td>0.5005</td>
<td>2.3250</td>
</tr>
<tr>
<td>Adagrad</td>
<td>512</td>
<td>1.9104</td>
<td>0.5678</td>
<td>1.9104</td>
<td>1.9756</td>
<td>0.5493</td>
<td>1.9824</td>
</tr>
<tr>
<td>Adam</td>
<td>128</td>
<td>0.0158</td>
<td>0.9976</td>
<td>0.0158</td>
<td>0.0939</td>
<td>0.986</td>
<td>0.0912</td>
</tr>
<tr>
<td>Adam</td>
<td>256</td>
<td>0.0335</td>
<td>0.9937</td>
<td>0.0335</td>
<td>0.0947</td>
<td>0.9859</td>
<td>0.0927</td>
</tr>
<tr>
<td>Adam</td>
<td>300</td>
<td>0.0372</td>
<td>0.9935</td>
<td>0.0372</td>
<td>0.0981</td>
<td>0.9848</td>
<td>0.0952</td>
</tr>
<tr>
<td>Adam</td>
<td>512</td>
<td>0.0955</td>
<td>0.9790</td>
<td>0.0955</td>
<td>0.2002</td>
<td>0.9529</td>
<td>0.2046</td>
</tr>
</tbody>
</table>

**Table 3. Research Results**
On Table 3, shows that Adam's activation function with 128 units performs very well compared to others.

![Figure 5. Accuracy and Loss Graph](image)

On Figure 5, The left part shows the loss graph and the right part shows the accuracy graph of the model. Based on this information, it can be concluded that the model learned the patterns in the data well in the first few epochs, experienced significant improvement in the 10th epoch, and reached a relatively stable level of accuracy and loss at around epoch 40. The model also did not experience overfitting, which indicates its ability to generalize to new data.

On Figure 6, showed that the engine gave high attention scores for the words "ayah" and "bahasa", and this resulted in an excellent translation.

2. BLUE Score Testing

In the training simulation conducted, testing was carried out by trying to translate Indonesian sentences into Seram Language (Slide) machines without OOV (Output-out vocabulary). In the Blue Score test conducted using 708 test data, the BLUE Score value is 0.90518992895191 or 90%, this shows that the model has a very good accuracy rate. The following are the results of the machine and human translation comparison shown in Table 4, as follows:
Table 4. Comparison of Machine and Human Translation

<table>
<thead>
<tr>
<th>No</th>
<th>Indonesia</th>
<th>Machine Translation</th>
<th>Human Translation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Mengambil Mangkuk</td>
<td>Na basi kua</td>
<td>Na basi kua</td>
</tr>
<tr>
<td>2</td>
<td>Adik Belajar Bahasa Ambon</td>
<td>Toi na balajar manuk ambon</td>
<td>Toi na balajar manuk ambon</td>
</tr>
<tr>
<td>3</td>
<td>Aku Makan Mangga</td>
<td>Aku Ka Ayai</td>
<td>Aku Ka Ayai</td>
</tr>
<tr>
<td>4</td>
<td>Membeli Anggur</td>
<td>Fas Anggur a</td>
<td>Fas Anggur a</td>
</tr>
</tbody>
</table>

3. Application of Website-based NMT

Interface design plays an important role in improving the effectiveness of Neural Machine Translation (NMT) models. The use of an appropriate interface can enhance the user's ability to interact with the model more effectively, as well as improve the performance of the model. As a result, much research has been done to develop an optimal interface design for NMT models.

The development of this website uses the best model from NMT training results stored in tensorflow tf-lite and then deployed using the Flask framework. The following is an image of the NMT Interface Design for Indonesian to Seram language translation (Geser).

Figure 7. Interface NMT Design
4. Conclusions

From the results of research conducted using 708 test data, the BLUE Score value is 0.90518992859191 or 90% for test data without Out-of-Vocabulary (OOV).

References


