applied sciences

MDPI

Article

Neural Network-Based Bilingual Lexicon Induction for Indonesian Ethnic Languages

Kartika Resiandi [1], Yohei Murakami [1] and Arbi Haza Nasution [2,*]

[1] Faculty of Information Science and Engineering, Ritsumeikan University, Kusatsu 525-8577, Shiga, Japan; yohei@fc.ritsumei.ac.jp (Y.M.)
[2] Department of Informatics Engineering, Universitas Islam Riau, Pekanbaru 28284, Riau, Indonesia
* Correspondence: arbi@eng.uir.ac.id

Abstract: Indonesia has a variety of ethnic languages, most of which belong to the same language family: the Austronesian languages. Due to the shared language family, words in Indonesian ethnic languages are very similar. However, previous research suggests that these Indonesian ethnic languages are endangered. Thus, to prevent that, we propose the creation of a bilingual dictionary between ethnic languages, using a neural network approach to extract transformation rules, employing character-level embedding and the Bi-LSTM method in a sequence-to-sequence model. The model has an encoder and decoder. The encoder reads the input sequence character by character, generates context, and then extracts a summary of the input. The decoder produces an output sequence wherein each character at each timestep, as well as the subsequent character output, are influenced by the previous character. The first experiment focuses on Indonesian and Minangkabau languages with 10,277 word pairs. To evaluate the model's performance, five-fold cross-validation was used. The character-level seq2seq method (Bi-LSTM as an encoder and LSTM as a decoder) with an average precision of 83.92% outperformed the SentencePiece byte pair encoding (vocab size of 33) with an average precision of 79.56%. Furthermore, to evaluate the performance of the neural network model in finding the pattern, a rule-based approach was conducted as the baseline. The neural network approach obtained 542 more correct translations compared to the baseline. We implemented the best setting (character-level embedding with Bi-LSTM as the encoder and LSTM as the decoder) for four other Indonesian ethnic languages: Malay, Palembang, Javanese, and Sundanese. These have half the size of input dictionaries. The average precision scores for these languages are 65.08%, 62.52%, 59.69%, and 58.46%, respectively. This shows that the neural network approach can identify transformation patterns of the Indonesian language to closely related languages (such as Malay and Palembang) better than distantly related languages (such as Javanese and Sundanese).

Keywords: natural language processing; low-resource language; Indonesian ethnic languages; bilingual lexicon induction; sequence-to-sequence model

Citation: Resiandi, K.; Murakami, Y.; Nasution, A.H. Neural Network-Based Bilingual Lexicon Induction for Indonesian Ethnic Languages. *Appl. Sci.* **2023**, *13*, 8666. https://doi.org/10.3390/app13158666

Academic Editor: Rocco Zaccagnino

Received: 12 May 2023
Revised: 16 July 2023
Accepted: 25 July 2023
Published: 27 July 2023

1. Introduction

Indonesia's riches extend beyond its natural resources, such as minerals, vegetation, and fauna. The archipelago is highly diversified, with a variety of ethnic languages in Indonesia, which are mostly part of the Austronesian language family. Since prehistoric times, Indonesian ethnic languages have evolved, resulting in a different language for each ethnic group [1]. Based on the similarity matrix utilizing the ASJP database [2], most Indonesian ethnic languages are closely related and similar.

Currently, the extinction of ethnic languages in Indonesia is a pressing issue that has attracted the attention of scholars, particularly linguists. The Summer Institute of Linguistics has stated that these local languages are endangered and may cease to be spoken in Indonesia. Therefore, we started the Indonesia Language Sphere project, which aims to create comprehensive bilingual dictionaries between the ethnic languages using a

neural network and crowdsourcing approach, in order to conserve local languages on the verge of extinction [3].

We expect that this project will expand the vocabulary of ethnic languages and that more people will learn and use them. The generated bilingual dictionary can be transformed into a bilingual dictionary service, and further integrated with the Google Translate service as a composite service on the Language Grid to enable a pivot-based hybrid machine translation, bridging the gap between high-resource languages and low-resource languages [4,5].

Indonesian ethnic languages are low-resource languages with a limited amount of language resources, such as bilingual dictionaries. We chose Minangkabau, Malay, Palembang, Javanese, and Sundanese as the languages to implement the proposed method in this study due to the availability of the bilingual dictionaries obtained from the results of our previous study [6]. Moreover, the Indonesian and Minangkabau languages have significant lexical similarities; thus, we presume they have several phonetic transformation rules, from Indonesian to Minangkabau and vice versa. For example, there appears to be a rule in Indonesian and Minangkabau where the last phoneme "a" in Indonesian tends to turn "o" in Minangkabau, while the middle phoneme "ia" appears to turn "i". There are many more patterns in these languages. Although this rule is not always applicable, it can help to predict a rough translation as a preliminary translation. To the best of our knowledge, no previous studies have explored approaches to bilingual lexicon induction using neural network targeting transformation rules between closely related languages. This study proposes a neural network-based bilingual lexicon induction for Indonesian ethnic languages with the following research goals:

- Model a neural network-based bilingual lexicon induction between Indonesian and Minangkabau, using long short-term memory (LSTM).
- Evaluate how well the model detects the transformation rules between Indonesian and Minangkabau.
- Apply the model to Malay, Palembang, Javanese, and Sundanese languages.

2. Bilingual Lexicon Induction

Creating a bilingual dictionary is the first crucial step toward enriching low-resource languages. Particularly for closely related languages, it has been shown that the constraint-based approach helps induce bilingual lexicons from two bilingual dictionaries via the pivot language [7,8]. However, implementing the constraint-based approach on a large scale to create multiple bilingual dictionaries is still challenging, particularly in determining the constraint-based approach's execution order to reduce the total costs. Plan optimization using the Markov decision process is essential when composing the order of creation for bilingual dictionaries, considering the methods and their costs [6,9].

Heyman et al. [10] proposed a method for transforming bilingual lexical induction into a binary classification task in the biomedical domain for English to Dutch. They created a classifier that predicts whether a pair of words translates using character and word level, employing the LSTM method. Their study showed that character-level representations successfully induce bilingual lexicons in the biomedical domain.

Zhang et al. [11] presented a character-level sequence-to-sequence learning method, RNNembed, for English-to-Chinese translation. Specifically, the recurrent neural network (RNN) is embedded into an encoder–decoder framework and generates character-level sequence representation as input. The dimension of the input feature space can be significantly reduced and eliminates the need to handle unknown or rare words in sequences. Experimental results demonstrate that the proposed approach achieves a translation performance that is comparable—or close—to conventional word-based and phrase-based systems.

Feng et al. [12] proposed a cross-lingual feature extraction (CFE) method to learn the cross-lingual features from monolingual corpora for low-resource UBLI, enabling representations of words with the same meaning to be leveraged by the initialization step. By integrating cross-lingual representations with pre-trained word embeddings in

a fully unsupervised initialization on UBLI, the proposed method outperforms existing state-of-the-art methods on low-resource language pairs.

Addressing the poor alignment of Chinese–Uyghur cross-language word embeddings due to significant morphological differences, Aysa et al. [13] proposed a multilingual morphological analyzer based on a morpheme sequence combined with neural network cross-language word embedding vector mapping, and used for Chinese–Uyghur bilingual dictionary extraction. They used robust morpheme segmentation and stemming of bilingual text data to obtain excellent and meaningful word semantic features. Using a small number of Chinese–Uyghur parallel seed dictionaries as weakly supervised signals, respectively, they mapped multilingual word or morpheme vectors to a unified vector space. Experimental results show that the morpheme sequence-based method for the Chinese–Uyghur dictionary induction task significantly improved the accuracy of dictionary alignment compared to the word-based model. Aysa et al. [14] actively explored the resource construction and granularity optimization of minority low-resource languages and learned cross-language word embeddings without the supervision of parallel data. A Chinese–Uyghur bilingual dictionary extraction method is proposed based on the neural network cross-language word embedding vector technology and the multilingual morphological analyzer. Experiments showed that the morpheme sequence-based approach significantly improved compared to the baseline model of the word sequence.

3. Materials and Methods

3.1. A Neural Network Approach

We introduce a neural network approach to extract transformation rules or patterns from the Indonesian to Minangkabau language [15]. The first approach uses character-level one-hot embedding, where words will be separated as characters, and each vector has the same length size adjusted by total characters. Then, the sequence-to-sequence (seq2seq) model, which has two RNN encoders and decoders, is utilized. Bi-LSTM as the encoder and LSTM as the decoder were used in this research. The Bi-LSTM encoder processes the word in the source language (Indonesian), character by character, and produces a representation of the inputted words. The LSTM decoder takes the output of the encoder as the input and produces a representation, character by character, in the target language (Minangkabau). Similar to the first method, the second method employs a sequence-to-sequence model. The distinction is in the inputted words, which are tokenized using SentencePiece with byte pair encodings for the input to the encoder–decoder in the sequence-to-sequence model. The tokenization involves splitting the words into chunks of characters.

The secondary data were obtained from Nasution et al. [2] and Koto and Koto [16] with a total of 13,761 translation pairs. Pre-processing the data was completed by deleting duplicate word pairs and constructing an array of word pairs in the form of a data-type dictionary given by Python. In this case, various Indonesian to Minangkabau word pairings have several meanings. A dictionary is made up of a set of key-value pairs. Each key-value pair corresponds to a certain value Baidalina et al. [17]. The data were validated by Minangkabau native speakers after the duplicate data were removed. As a result, there were 10,278 translation pairs in the complete set of data. The model's performance was evaluated using five-fold cross-validation with precision as the evaluation metric. Precision measures the extent to which the positive predictions from a classification model are correct. Precision calculates the percentage of true positive (TP) predictions compared to the total positive, i.e., true positive (TP) + false positive (FP) predictions made by the model, as shown in Equation (1).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{1}$$

3.2. Long Short-Term Memory (LSTM)

Long short-term memory (LSTM) is an upgraded recurrent neural network (RNN) that is used to overcome the problem of vanishing and exploding gradients [18]. LSTM addresses the problem of long-term RNN reliance, where RNNs are unable to predict input data stored in long-term memory but can make more accurate predictions based on current information. The LSTM architecture can store large amounts of data for lengthy periods of time. They are applied to time-series data processing, forecasting, and categorization. Memory cells and gate units are key components of the LSTM architecture. Forget gate, input gate, and output gate are the three types of gates in an LSTM. Figure 1 illustrates the structure of the LSTM model. The library used was tf.keras.layers.LSTM.

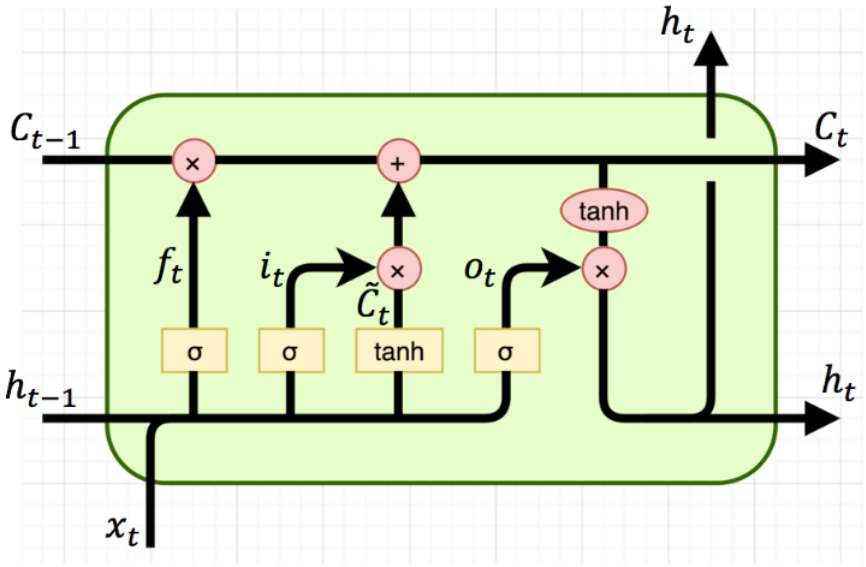


Figure 1. Unit structure of the LSTM.

Cell memory tracks the dependencies between components in the input sequence. New values that enter the cell state are handled by the input gate. The LSTM unit utilizes a forget gate to select the value that remains in the cell state. The value in the cell state that remains will be sent to the output gate, where the LSTM activation function, also known as the logistic sigmoid function, will be used to start the calculation. The tanh and sigma symbols represent the types of activation functions employed in the neural network's training layers.

A sigmoid gate, which restricts how much information may pass through, and allows information to flow through it unmodified, is another essential feature of LSTM. The outputs of the sigmoid layer, which vary from zero to one, specify how much of each component should be permitted to pass. The Equation (2) that controls the LSTM flow is as follows:

$$
\begin{aligned}
f_t &= \sigma(w_f \cdot [h_{t-1}, x_t] + b_f \\
i_t &= \sigma(w_i \cdot [h_{t-1}, x_t] + b_i \\
C_t &= \tanh(w_c \cdot [h_{t-1}, x_t] + b_c \\
C_t &= f_t \times C_{t-1} + i_t \star C_t \\
o_t &= \sigma(w_o \cdot [h_{t-1}, x_t] + b_o \\
h_t &= o_t \times \tanh C_t
\end{aligned}
\tag{2}
$$

where

o_t	: at time t, output gate
i_t	: at time t, input gate
h_t	: output at time t
f_t	: forget gate, at time t
x_t	: input at time t
σ	: sigmoid function
C_t	: the state of the cell at time t
w_o, w_f, w_i, w_c	: weights that have been trained
b_c, b_i, b_f	: trained biases

3.3. Bidirectional Long Short-Term Memory (Bi-LSTM)

RNN's advantage is in the reliance on coding inputs. However, LSTM's advantage is in resolving RNN's long-term issues. Improvements are made with Bi-RNN because only one direction of the previous contextual information can be used by LSTM and RNN [19].

As a result of the advantages of each technique, the LSTM form is kept in the cell memory, and Bi-RNN can process information from the previous and following contexts, resulting in Bi-LSTM [19]. Bi-LSTM can leverage contextual information and generate two separate sequences from the LSTM output vector. Each time step's output is a mixture of the two output vectors from both directions [20]. Figure 2 depicts the combination of LSTM and Bi-RNN. The library used was tf.keras.layers.Bidirectional.

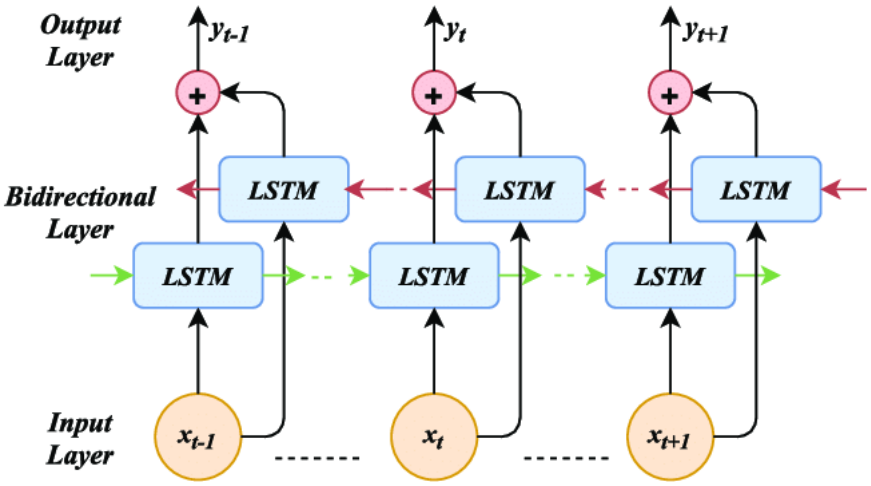


Figure 2. Bi-LSTM architecture.

3.4. Character-Level Sequence-to-Sequence Model

Figure 3 shows the Seq2Seq model considered in this study with a two-layered Bi-LSTM encoder and LSTM decoder. The encoder's functions are to read the input sequence (character by character), build context, and extract a summary of the input. The decoder will provide an output sequence in which the previous character affects every character in each time step as well as the next character that emerges. The marker /<eos/> denotes the end of a sentence, and it will determine when we stop predicting the following character in a series [21].

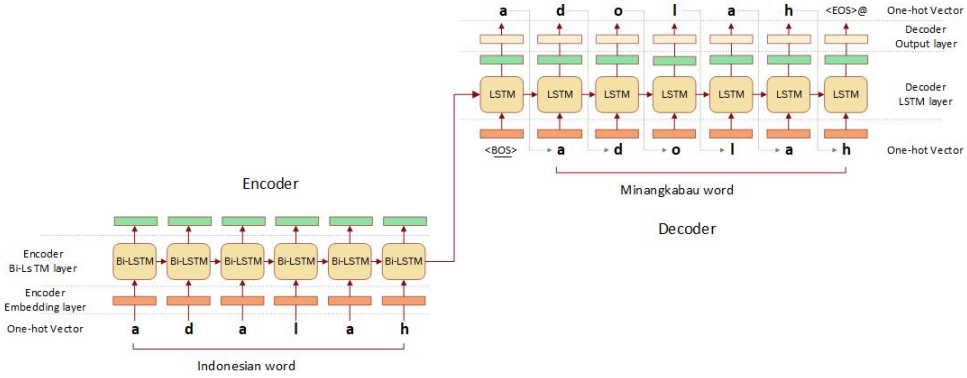


Figure 3. Character-level sequence-to-sequence model.

Following the construction of the encoder–decoder network architecture in this typical end-to-end framework, a training approach may be utilized to obtain an optimal word pair translation model and keep the character order, referred to as a cell state or memory cell. Since the horizontal line across the bottom of the diagram is in the source and target words, the input (Indonesia) and output (Minangkabau) sequence must be treated in time order. For the Indonesian language, 28 input tokens were used, and 31 output tokens were used for the Minangkabau language.

3.5. SentencePiece Sequence-to-Sequence with Byte Pair Encoding (BPE)

The second method we present is SentencePiece (as subword tokenization). According to Kudo [22], subword tokenization implements SentencePiece, subword-nmt, and WordPiece model features. Subword vocabulary is built by using the BPE segmentation method to train a SentencePiece tokenization model, which divides words into chunks of characters based on vocabulary size to make pattern detection easier.

BPE was added to our research methodology because Indonesian ethnic languages now utilize an alphabet script established by the Dutch, despite having original traditional scripts. Dutch people appeared to assign chunks of alphabets to phonemes of Indonesian ethnic languages when teaching the alphabet to them [1]. As a result, most Indonesian ethnic languages can use the same or similar tokens.

Furthermore, with each phonetic development, languages belonging to the same language family descended from the same proto-language. As a result, we assume a phonetic-based strategy is preferable to a character-based method. The number of words to be processed into tokenization is known as the vocabulary size, which in this case, refers to the number of most frequently occurring characters, including symbols like </unk>, and whitespace. We employed a wide range of vocabulary sizes.

In the character-level sequence-to-sequence model shown in Figure 3, both input and output are split, character by character. As for the SentencePiece sequence-to-sequence model with BPE, as shown in Figure 4, both input and output are split by the BPE method, after initially setting the vocabulary size for each language.

BPE constructs a base vocabulary comprising all symbols found in the set of unique words, and then learns merge rules to combine two symbols from the base vocabulary to create a new symbol. It continues to do this until the vocabulary has grown to the required size. The BPE algorithm replaces the data byte pairs that occur most frequently with a new byte until the data can no longer be compressed because no byte pair occurs most frequently. The steps in the training procedure are as follows [23]:

(1) Gather a huge amount of training data.
(2) Determine the vocabulary size.
(3) Identify the end of a word, add an identifier (</w>) to the end of each word, and then calculate the word frequency in the text.
(4) Calculate the character frequency after dividing the word into characters.

(5) Count the frequency of consecutive byte pairs from the character tokens for a predetermined number of rounds and combine the most frequently occurring byte pairings.
(6) Repeat step 5 until the required number of merging operations has been performed or the specified vocabulary size is reached.

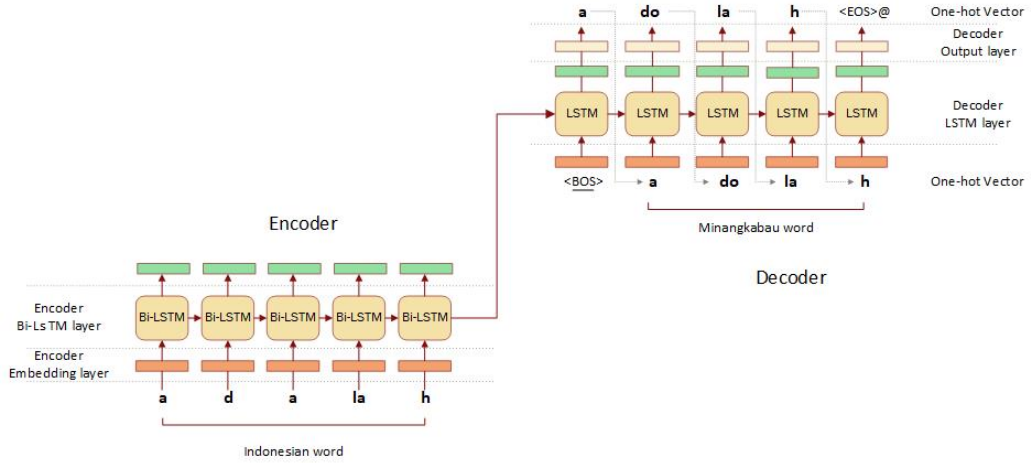


Figure 4. SentencePiece sequence-to-sequence model with BPE.

The input text is treated as a sequence of Unicode characters by SentencePiece. Whitespace is also treated like any other symbol. SentencePiece expressly handles whitespace as a fundamental token by first escaping it with the meta symbol "___" (U + 2581) [22]. Meanwhile, the '\n' symbol is the end of a string. The results of the chunks of characters from the BPE will vary when utilizing a larger vocab size.

Excluding alphabets, the vocabularies obtained from BPE 40 and 100 are summarized in Tables 1 and 2. Overall, the vocabulary numbers in Indonesia and Minangkabau are the same (7 and 68, respectively). As shown in Table 1, more character pieces are obtained if larger vocabulary sizes are used. The alphabet following the "_" symbol is a portion of the characters from the beginning of the term in the vocabulary. For example, as indicated in bold in Table 1, the difference between the Minangkabau character pieces, **sa** and _ **sa**, is that **sa** indicates that the character is not at the beginning of the word. Tokenization results are presented in Table 2, which shows the words in Minangkabau and Indonesian turned into pieces of characters from BPE. The tokenization with vocab size = 40 is almost the same as character-based tokenization because vocab size = 40 is nearly the same as the number of alphabets.

Table 1. Vocabularies obtained from BPE Indonesian–Minangkabau.

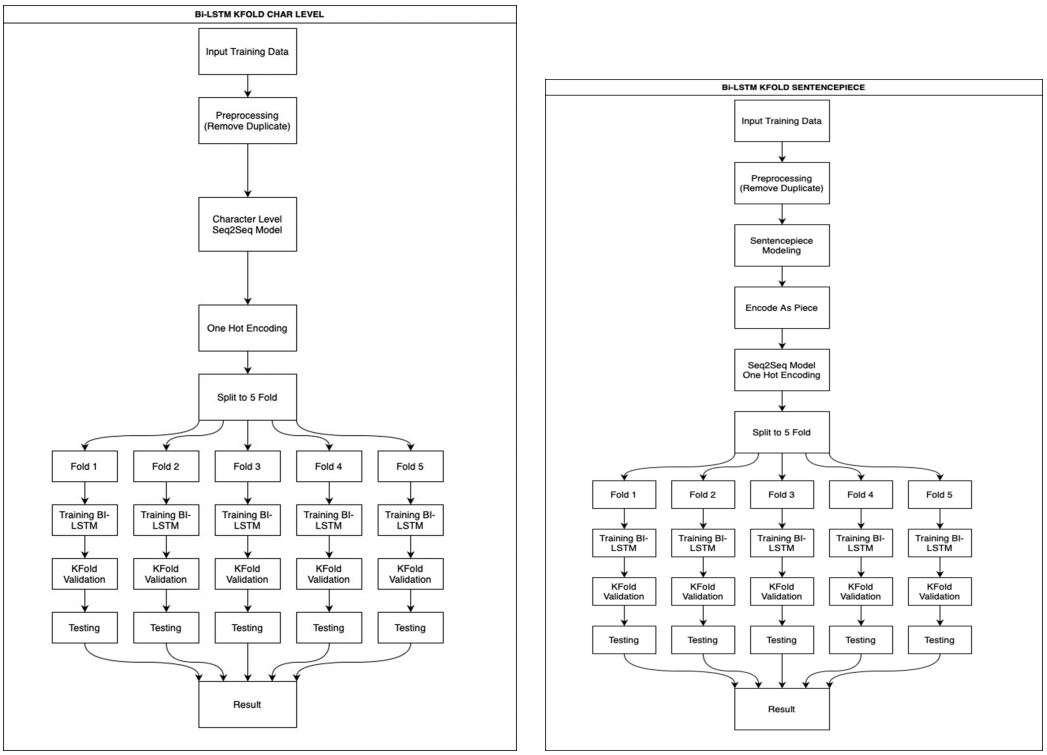
Language	Vocab Size = 40	Vocab Size = 100
Indonesian	an, ng, nya, ta, kan, _di, _men,	an, ng, kan, ta, _di, la, nya, ra, da, si, _ke, _ber, ti, ba, li, ga, ri, ja, er, tu, bu, _se, at, in, _men, ma, sa, _per, ka, en, di, wa, ku, _meng, ya, na, _me, _pen, te, mp, ca, _p, _ter, ru, du, _mem, de, pa, or,un, ar, ju, is, _ka, bi, _ko, _ma, re, on, _ba, _pe, _pem, tan, pu, gu, al, ran, asi
Minangkabau	an, ang, _pa, _di, _ma, _ba, ng	an, ng, _di, _ba, ra, si, la,_pa, nyo, _ka, ta, da, ang, _ma, ik, kan, li, ri, ti, ak, tu, ka, _**sa**, _man, ja, ah, _ta, bu, ga, ek, in, ba, ku, **sa**, ma, su, di, ru, ya, _a, mp, _pan, to, wa, pa, ca, ran, du, ro, lu, tan, lo, mba, angan, ju, bi, pu, re, han, en, te, do, de, ko, gu, gi, _mam

Table 2. Example of tokenization BPE with different vocabulary sizes, Indonesian–Minangkabau.

Vocab Size = 40		Vocab Size = 100	
Indonesian	**Minangkabau**	**Indonesian**	**Minangkabau**
_,y,a,ng	_,n,an,'\n'	_,ya,ng	_,n,an,'\n'
_,p,a,d,a	_,pa,d,o,'\n'	_,pa,da	_,pa,do,'\n'
_,a,d,a,la,h	_,a,d,o,l,a,h,'\n'	_,a,da,la,h	_,a,do,la,h,'\n'
_,s,e,g,e,ra	_,s,a,g,i,r,o,'\n'	_,se,ge,ra	_,se,ge,ra,'\n'
_,d,a,s,a,r,nya	_,d,a,s,a,n,y,o,'\n'	_,da,sa,r,nya	_,da,sa,nyo,'\n'

3.6. Experiment Design

This study has three research goals. To reach the first goal, two models were used to find translation word pairs; they will be examined using bidirectional long short-term memory and long short-term memory, according to previous research [10]. Figure 5 shows the comparison between the two models, which are character-level and SentencePiece with BPE. We utilized the parameters selected for both models in Table 3.



(**a**) Character level. (**b**) SentencePiece with BPE

Figure 5. Experimental design for the two models.

Table 3. Model's parameter.

Character Level and SentencePiece with BPE		
Parameter	**Bi-LSTM**	**LSTM**
Embedding Size	512	512
Epoch	120	120
Batch Size	64	64

In this case, the implemented learning rate schedule technique was the learning rate decay; we chose an initial learning rate, then reduced it progressively according to a scheduler. We set the learning rate to 0.001, and then the learning rate decreased by 1% for every epoch above the 15th. A slower learning rate may allow the model to acquire a

more optimal or even globally optimal set of weights, but it will take much longer to train the model. The 10,278 translation pairs are split into 8221 training sets and 2056 test sets.

Languages evolve from each generation of language users and learners, which suggests that arbitrary aspects of linguistic structure may result from general learning and processing biases derived from thought processes, perception factors, cognitive limitations, and pragmatics [24]. Therefore, after determining the best neural network-based bilingual lexicon induction model, the second research goal is obtained by evaluating how well the model detects the transformation rules between Indonesian and Minangkabau. We define a rule-based approach as a baseline by involving Minangkabau language experts to provide the transformation patterns of the Minangkabau language to the Indonesian language. Lastly, to reach the third research goal, we apply the model to Malay, Palembang, Javanese, and Sundanese languages.

4. Results

4.1. Neural Network Performance

This study uses two scenarios to find the optimal seq2seq model with the best performance. When comparing the character-level and SentencePiece approaches with the seq2seq model, the character-level seq2seq method generates a more accurate translation of word pairs.

As shown in Tables 4 and 5, the results demonstrate that character-level tokenization, as opposed to BPE tokenization, is more suitable for translating word-to-word. The vocabulary size has minimum and maximum values. The minimum value necessary for this experiment's data is 33. The experiment was run seven times with different vocabulary sizes; the maximum vocabulary size used was 300. When utilizing a minimal vocabulary size in BPE, it indicates that the number of tokens is approximately the same as the character-level-based method. However, as shown in Table 5, the tokenization outcomes from the source and target pairs will vary more as the vocabulary size increases, which has an impact on the BPE performance outcomes. This shows that because the vector length is shortened, the data are likely to be less informative, making it more difficult for the model to recognize. In general, the larger the vocabulary size, the higher the results due to the data being word-to-word pair translations instead of sentence-to-sentence.

Table 4. Evaluation of the character-level model.

Method	K-Fold Cross-Validation Indonesian–Minangkabau					
	K = 1	K = 2	K = 3	K = 4	K = 5	Average Precision
Bi-LSTM (encoder), LSTM (decoder)	84.72	83.70	83.31	83.60	84.30	83.92
LSTM (encoder–decoder)	76.79	74.56	77.82	78.21	75.87	76.65

Table 5. Evaluation of SentencePiece with the BPE model.

Vocab Size	K-Fold Cross-Validation Indonesian–Minangkabau					
	K = 1	K = 2	K = 3	K = 4	K = 5	Average Precision
33	79.96	76.55	78.84	**81.71**	80.78	**79.56**
35	76.11	76.89	79.42	74.31	80.73	77.49
40	72.12	72.88	75.23	75.99	71.64	73.59
50	67.12	62.15	66.97	67.41	64.29	65.58
80	58.73	59.32	53.35	54.12	56.47	56.39
100	49.36	48.24	49.46	49.70	48.78	49.10
300	34.85	34.93	30.31	35.76	36.19	34.40

Figure 6 illustrates that the character-level method has a shorter vector length (29) compared to the SentencePiece with the BPE method (300) when representing the same word, i.e., "*adolah*". As shown in Table 5, the larger the vocabulary size, the lower the translational accuracy results. For an example of the Minangkabau word **adolah**, if the vocabulary size = 300, the number of tokens decreases, while the lengths of the vectors representing the tokens become longer because the vectors need more expression power.

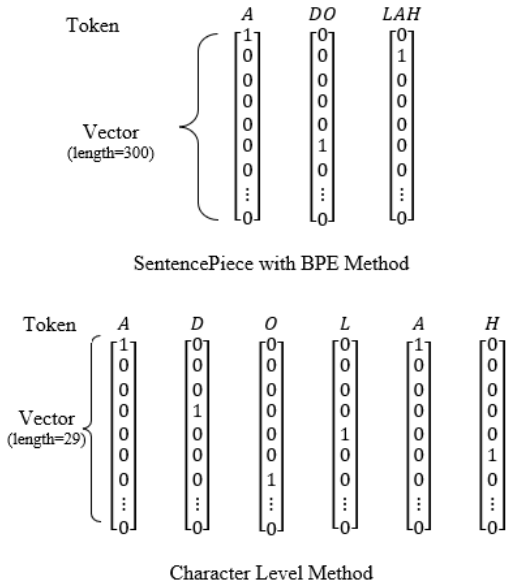


Figure 6. Comparison between SentencePiece with BPE and the character-level method.

4.2. Pattern Recall

To evaluate the neural network-based approach (character-level Bi-LSTM), we define a rule-based approach as the baseline. Minangkabau language experts provided the transformation patterns of the Indonesian language to the Minangkabau language. The steps to generate translation using the rule-based approach are as follows:

(1) Remove the translation pairs, where the source word and the target word are identical from the 2056 translation pair candidates, making 1262 translation pairs.
(2) Following the transformation pattern from the expert, define the transformation rules by regular expressions.
(3) Use the transformation rules that have been determined with all source words and replace rule matches with a string.

There are 34 transformation patterns for the Indonesian to Minangkabau language, as shown in Table 6. However, there are exceptions for pattern numbers 21–27, where the patterns can be grouped by changing all first characters "e" to "a" and all first characters "er" to "a". The group can be created using regular expressions ('([aiueo]*) e') and ('([aiueo]*)er').

The rule-based approach can generate 475 of the 1262 translation pairings, whereas the remaining 787 translation pairs cannot be produced since they have no pattern or have double or multiple patterns. Figure 7 describes the rule-based results. There are several factors affecting the poor performance of the rule-based approach. The rule-based approach is unable to identify two or more patterns in a single word, as shown in Table 7. Only a single pattern can be generated via a rule-based approach. The rule-based approach can be enhanced by identifying multiple rules on a single word, which can potentially increase recall. However, the more rules applied to a single word, the higher the risk of low precision. This thread-off could be explored in future work. Moreover, the current rules listed in Table 6 can be enriched by Minangkabau native speakers to improve their performance.

Table 6. List of patterns in Indonesian and Minangkabau.

	Pattern	Indonesian	Minangkabau
1	Ending **uk** to **uak**	Rus**uk**	Rus**uak**
2	Ending **a** to **o**	Sam**a**	Sam**o**
3	Ending **ik** to **iak**	Bat**ik**	Bat**iak**
4	Ending **ing** to **iang**	Bal**ing**	Bal**iang**
5	Remove last character	Tuka**r**	Tuka
6	Ending **as** to **eh**	Pan**as**	Pan**eh**
7	Ending **uh** to **uah**	Pen**uh**	Pen**uah**
8	Ending **ut** to **uik**	La**ut**	Lau**ik**
9	Ending **ung** to **uang**	Pat**ung**	Pat**uang**
10	Ending **ap** to **ok**	At**ap**	At**ok**
11	Ending **it** to **ik**	Kul**it**	Kul**ik**
12	Ending **is** to **ih**	Lap**is**	Lap**ih**
13	Ending **up** to **uik**	Hid**up**	Hid**uik**
14	Ending **ul** to **ua**	Puk**ul**	Puk**ua**
15	Ending **kan** to **an**	Arahk**an**	arah**an**
16	Ending **a** to **ok**	Jik**a**	Jik**ok**
17	Ending **ur** to **ua**	Kab**ur**	Kab**ua**
18	Ending **t** to **ik**	Gia**t**	Gia**ik**
19	Beginning **meng** to **ma**	**Meng**adu	**Ma**adu
20	Beginning **meng** to **mang**	**Meng**aku	**Man**gaku
21	Beginning **Ber** to **Ba**	**Ber**lari	**Ba**lari
22	Beginning **Per** to **Pa**	**Per**jalanan	**Pa**jalanan
23	Beginning **Pe** to **Pa**	**Pe**nyabar	**Pa**nyaba
24	Beginning **Se** to **Sa**	**Se**irama	**Sa**irama
25	Beginning **Re** to **Ra**	**Re**tak	**Ra**tak
26	Beginning **Te** to **Ta**	**Te**pian	**Ta**pian
27	Beginning **Ter** to **Ta**	**Ter**makan	**Ta**makan
28	Ending **ir** to **ia**	Kinc**ir**	Kinc**ia**
29	Ending **at** to **ek**	Kering**at**	Kering**ek**
30	Ending **d** to **ik**	Jasa**d**	Jasa**ik**
31	Ending **id** to **ik**	Mur**id**	Mur**ik**
32	Ending **ih** to **iah**	Gig**ih**	Gig**iah**
33	Ending **us** to **uih**	Ar**us**	Ar**uih**
34	Ending **il** to **ia**	Has**il**	Has**ia**

Table 7. Example words with double or multiple patterns.

	Pattern	Indonesian	Minangkabau
1	**Me** to **ma**, **kan** to **an**	**Me**resmi**kan**	**Ma**resmi**an**
2	**Pe** to **Pa**, **ih** to **iah**	**Pe**mili**h**	**Pa**mili**ah**
3	**Ke** to **Ka**, **Ing** to **Iang**, **Kan** to **An**	**Ke**ringka**n**	**Ka**riangan

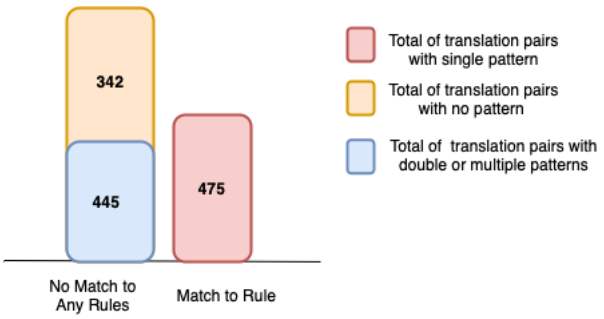


Figure 7. Rule-based result.

We compare the results with the best neural network model, which is the character-level model with Bi-LSTM as the encoder and LSTM as the decoder. We conduct the comparison by examining some of the potential outcomes from the test data as shown in Table 8. As shown in Table 8, there are 414 correct translations obtained by both the neural network approach and rule-based approach, which means that the neural network approach successfully identifies the transformation patterns of the Indonesian language to the Minangkabau language. Moreover, the neural network approach successfully obtains 603 correct translation pairs, which have no pattern or have multiple patterns that cannot be handled by the rule-based approach. Furthermore, only 61 translation pairs are not recognized by the neural network approach.

Table 8. Comparisons between the neural network approach and rule-based approach.

Method	Result	Neural	
		Correct	Wrong
Rule	Correct	414	61
	Wrong	603	184

4.3. Neural Network Performances for Other Ethnic Languages

The best neural network model, which is the character-level model with Bi-LSTM as the encoder and LSTM as the decoder, was utilized in additional experiments to create Indonesian–Malay, Indonesian–Palembang, Indonesian–Javanese, and Indonesian–Sundanese bilingual dictionaries. Unfortunately, we only have small secondary data obtained from Nasution et al. [2], as shown in Table 9. The source code and the bilingual dictionaries are available online (https://github.com/arbihazanst/neural-network-lexicon-induction, accessed date: 24 July 2023).

Table 9. Dataset for all ethnic languages.

Language Pair	#Translation Pair	#Training Set	#Test Set
Indonesian–Minangkabau	13,761	11,008	2753
Indonesian–Malay	5229	4183	1046
Indonesian–Palembang	5098	4078	1020
Indonesian–Javanese	4778	3822	956
Indonesian–Sundanese	5045	4036	1009

The Indonesian–Malay dictionary has 5229 translation pairs, divided into 80% in the training set (4183), and 20% in the test set (1046); the number of tokens in Indonesia is 27 characters, and in Malay is 30 characters. Table 10 shows the performance results of the character-level model for the Indonesian–Malay experiment. The Indonesian–Palembang dictionary has 5098 translation pairs, divided into 80% in the training set (4078), and 20% in the test set (1020); the number of tokens in Indonesia is 28 characters, and in Palembang is 29 characters. Table 11 shows the performance results of the character-level model of the Indonesian–Palembang experiment. The Indonesian–Javanese dictionary has 4778 translation pairs, divided into 80% in the training set (3822) and 20% in the test set (956); the number of tokens in Indonesia is 27 characters, and in Javanese is 32 characters. Table 12 shows the performance results of the character-level model of the Indonesian–Javanese experiment. Finally, the Indonesian–Sundanese dictionary has 5045 translation pairs, divided into 80% in the training set (4036) and 20% in the test set (1009); the number of tokens in Indonesia is 26 characters, and in Sundanese is 33 characters. Table 13 shows the performance results of the character-level model for the Indonesian–Sundanese experiment.

Table 10. Evaluation of the character-level model in Indonesian–Malay.

Method	K-Fold Cross-Validation Indonesian–Malay					
	K = 1	K = 2	K = 3	K = 4	K = 5	Average Precision
Bi-LSTM (encoder), LSTM (decoder)	64.72	66.15	65.20	65.96	63.38	65.08

Table 11. Evaluation of the character-level model in Indonesian–Palembang.

Method	K-Fold Cross-Validation Indonesian–Palembang					
	K = 1	K = 2	K = 3	K = 4	K = 5	Average Precision
Bi-LSTM (encoder), LSTM (decoder)	63.82	62.45	63.23	60.29	62.84	62.52

Table 12. Evaluation of the character-level model in Indonesian–Javanese.

Method	K-Fold Cross-Validation Indonesian–Javanese					
	K = 1	K = 2	K = 3	K = 4	K = 5	Average Precision
Bi-LSTM (encoder), LSTM (decoder)	62.02	59.30	59.62	55.34	61.08	59.69

Table 13. Evaluation of the character-level model in Indonesian–Sundanese.

Method	K-Fold Cross-Validation Indonesian–Sundanese					
	K = 1	K = 2	K = 3	K = 4	K = 5	Average Precision
Bi-LSTM (encoder), LSTM (decoder)	57.77	58.47	59.36	59.26	57.48	58.46

Based on the results, we can see the average precision numbers for Indonesian–Malay and Indonesian–Palembang (65.08 and 62.52, respectively) are higher than the average precision numbers for Indonesian–Javanese and Indonesian–Sundanese (59.69 and 58.46, respectively). This shows that the neural network approach can identify transformation patterns of the Indonesian language to closely related languages (such as Malay and Palembang) better than distantly related languages (such as Javanese and Sundanese).

5. Conclusions

In order to obtain word-to-word translation pairs, the experiment shows that the neural network approach utilizing a sequence-to-sequence model is more able to extract Indonesian–Minangkabau language transformation patterns with a distinct number of tokens based on a character basis. The character-level seq2seq method (Bi-LSTM as an encoder and LSTM as a decoder) with an average precision of 83.92% outperforms the SentencePiece byte pair encoding (with a vocab size of 33), with an average precision of 79.56%. Furthermore, to evaluate the performance of the neural network model in finding the pattern, a rule-based approach was conducted as the baseline. The neural network approach obtained 542 more correct translations compared to the baseline. We implemented the best settings (character-level embedding with the Bi-LSTM as the encoder and LSTM as the decoder) for four other Indonesian ethnic languages (Malay, Palembang, Javanese, and Sundanese), with average precisions of 65.08%, 62.52%, 59.69%, and 58.46%, respectively. This shows that the neural network approach can identify transformation patterns of the Indonesian language to closely related languages (such as Malay and Palembang) better than distantly related languages (such as Javanese and Sundanese). The generated neural

network–bilingual lexicon induction model can be improved by increasing the size of the bilingual dictionaries so that the model can learn more translation patterns in Minangkabau, Malay, Palembang, Javanese, Sundanese, and other Indonesian ethnic languages.

Author Contributions: Conceptualization, Y.M. and K.R.; methodology, Y.M. and K.R.; software, A.H.N.; validation, Y.M. and A.H.N.; formal analysis, Y.M. and A.H.N.; investigation, Y.M. and A.H.N.; resources, K.R.; data curation, K.R. and A.H.N.; writing—original draft preparation, K.R.; writing—review and editing, A.H.N.; visualization, K.R. and A.H.N.; supervision, Y.M.; funding acquisition, Y.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially supported by a Grant-in-Aid for Scientific Research (B) (21H03561, 2021–2024) and a Grant-in-Aid Young Scientists (A) (17H04706, 2017–2020) from the Japan Society for the Promotion of Science (JSPS).

Data Availability Statement: The source code and the bilingual dictionaries are available online (https://github.com/arbihazanst/neural-network-lexicon-induction, accessed date: 24 July 2023).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ASJP	Automated Similarity Judgment Program
Bi-LSTM	bidirectional long short-term memory
BPE	byte pair encoding
LSTM	long short-term memory
RNN	recurrent neural network
seq2seq	sequence-to-sequence

References

1. Paauw, S. One land, one nation, one language: An analysis of Indonesia's national language policy. *Univ. Rochester Work. Pap. Lang. Sci.* **2009**, *5*, 2–16.
2. Nasution, A.H.; Murakami, Y.; Ishida, T. Generating similarity cluster of Indonesian languages with semi-supervised clustering. *Int. J. Electr. Comput. Eng. (IJECE)* **2019**, *9*, 531–538. [CrossRef]
3. Murakami, Y. Indonesia Language Sphere: An ecosystem for dictionary development for low-resource languages. *J. Phys. Conf. Ser.* **2019**, *1192*, 012001. [CrossRef]
4. Nasution, A.H.; Syafitri, N.; Setiawan, P.R.; Suryani, D. Pivot-based hybrid machine translation to support multilingual communication. In Proceedings of the 2017 International Conference on Culture and Computing (Culture and Computing), Kyoto, Japan, 10–12 September 2017; pp. 147–148.
5. Nasution, A.H. Pivot-based hybrid machine translation to support multilingual communication for closely related languages. *World Trans. Eng. Technol. Educ.* **2018**, *16*, 167–172.
6. Nasution, A.H.; Murakami, Y.; Ishida, T. Plan Optimization to Bilingual Dictionary Induction for Low-resource Language Families. *Trans. Asian Low-Resour. Lang. Inf. Process.* **2021**, *20*, 1–28. [CrossRef]
7. Nasution, A.H.; Murakami, Y.; Ishida, T. Constraint-based bilingual lexicon induction for closely related languages. In Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), Portorož, Slovenia, 23–28 May 2016; pp. 3291–3298.
8. Nasution, A.H.; Murakami, Y.; Ishida, T. A generalized constraint approach to bilingual dictionary induction for low-resource language families. *ACM Trans. Asian Low-Resour. Lang. Inf. Process. (TALLIP)* **2017**, *17*, 1–29.
9. Nasution, A.H.; Murakami, Y.; Ishida, T. Plan optimization for creating bilingual dictionaries of low-resource languages. In Proceedings of the 2017 International Conference on Culture and Computing (Culture and Computing), Kyoto, Japan, 10–12 September 2017; pp. 35–41.
10. Heyman, G.; Vulić, I.; Moens, M.F. A deep learning approach to bilingual lexicon induction in the biomedical domain. *BMC Bioinform.* **2018**, *19*. [CrossRef] [PubMed]
11. Zhang, H.; Li, J.; Ji, Y.; Yue, H. A character-level sequence-to-sequence method for subtitle learning. In Proceedings of the 2016 IEEE 14th International Conference on Industrial Informatics (INDIN), Poitiers, France, 19–21 July 2016. [CrossRef]
12. Feng, Z.; Cao, H.; Zhao, T.; Wang, W.; Peng, W. Cross-lingual Feature Extraction from Monolingual Corpora for Low-resource Unsupervised Bilingual Lexicon Induction. In Proceedings of the 29th International Conference on Computational Linguistics, Gyeongju, Republic of Korea, 12–17 October 2022; pp. 5278–5287.

13. Aysa, A.; Ablimit, M.; Yilahun, H.; Hamdulla, A. Chinese-Uyghur Bilingual Lexicon Induction Based on Morpheme Sequence and Weak Supervision. In Proceedings of the 2022 3rd International Conference on Pattern Recognition and Machine Learning (PRML), Chengdu, China, 22–24 July 2022; pp. 357–363.
14. Aysa, A.; Ablimit, M.; Yilahun, H.; Hamdulla, A. Sub-word based unsupervised bilingual dictionary induction for Chinese-Uyghur. In Proceedings of the 2022 International Conference on Asian Language Processing (IALP), Singapore, 27–28 October 2022; pp. 476–481.
15. Resiandi, K.; Murakami, Y.; Nasution, A.H. A Neural Network Approach to Create Minangkabau-Indonesia Bilingual Dictionary. In Proceedings of the the 1st Annual Meeting of the ELRA/ISCA Special Interest Group on Under-Resourced Languages, Marseille, France, 24–25 June 2022; European Language Resources Association: Marseille, France, 2022; pp. 122–128.
16. Koto, F.; Koto, I. Towards Computational Linguistics in Minangkabau Language: Studies on Sentiment Analysis and Machine Translation. In Proceedings of the 34th Pacific Asia Conference on Language, Information and Computation, Hanoi, Vietnam, 24–26 October 2020; pp. 138–148.
17. Baidalina, A.R; Boranbayev, S.A. Programming data structure algorithms in Python. *Bull. Ser. Phys. Math. Sci.* **2021**, *73*, 134–141. [CrossRef]
18. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
19. Schuster, M.; Paliwal, K. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.* **1997**, *45*, 2673–2681. [CrossRef]
20. Yulita, I.N.; Fanany, M.I.; Arymuthy, A.M. Bi-directional Long Short-Term Memory using Quantized data of Deep Belief Networks for Sleep Stage Classification. *Procedia Comput. Sci.* **2017**, *116*, 530–538. [CrossRef]
21. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to sequence learning with neural networks. *arXiv* **2014**, arXiv:1409.3215.
22. Kudo, T. Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates. *arXiv* **2018**, arXiv:1804.10959.
23. Sennrich, R.; Haddow, B.; Birch, A. Neural Machine Translation of Rare Words with Subword Units. *arXiv* **2015**, arXiv:1508.07909.
24. Ranaldi, L.; Pucci, G. Knowing knowledge: Epistemological study of knowledge in transformers. *Appl. Sci.* **2023**, *13*, 677. [CrossRef]