Named Entity Recognition of Tourist Destinations Reviews in the Indonesian Language

Muhammad Fakhri Despawida Aulia Putra¹, Ahmad Fathan Hidayatullah², Adityo Permana Wibowo³, Kartika Rizqi Nastiti⁴

¹Undergraduate Program in Informatics, Universitas Islam Indonesia
Jl. Kaliurang km 14.5 Sleman, Yogyakarta
muhammadfakhri335@gmail.com

²Department of Informatics, Universitas Islam Indonesia
Jl. Kaliurang km 14.5 Sleman, Yogyakarta
fathan@uii.ac.id

³Information Systems Study Program, Universitas Teknologi Yogyakarta
Jl. Siliwangi (Ringroad Utara), Jombor, Sleman, Yogyakarta
adityopw@uty.ac.id

⁴Master of Computer Science Program, Universitas Gadjah Mada
Sekip Utara Bulaksumur Sleman, Yogyakarta
kartikarizqinastiti@mail.ugm.ac.id

Abstract—Tourists typically rely on reviews of destinations when seeking information about tourist attractions. However, several studies have found that locating the specific information they need can be challenging. Named Entity Recognition (NER) is a technique used to identify entities within the text. This research aims to develop a NER model using Bidirectional Long Short-Term Memory (BLSTM) to identify and evaluate entities in tourism destination reviews. This study utilized 2010 reviews of various tourism destinations in Indonesia and segmented them into 116,564 tokens. These tokens were categorized based on their attributes, including the destination's name, location, and facilities. Tokens that could not be classified within existing categories were labeled outside (O). In this study, we applied several hyperparameter scenarios using BLSTM. Based on the experiments, we obtained the best average F1 score of 94.3%.

Keywords—NER, BLSTM, tourism reviews, Bahasa Indonesia

I. INTRODUCTION

Indonesia is one of the countries with the most attractive tourist destinations. There are many destinations to visit in Indonesia: natural tourism, education, cuisine, shopping, and culture. As technology develops, much information is scattered about tourism destinations on the internet. This makes it easier for tourists to get complete information about the tourist destinations they want to visit [1]. Nonetheless, not all information found on the internet is pertinent to tourists. It can be time-consuming to filter through irrelevant data to obtain valuable information about their desired destinations. To acquire detailed knowledge about a location's offerings, tourists must peruse reviews that provide specific insights about cities, types of tourism destinations, and facilities. Consequently, the process of extracting pertinent information from existing data becomes crucial [2].

One of the information extraction tasks that can be performed is named entity recognition (NER). NER is the process of extracting information from a structured or unstructured document to identify the name of a person, place, organization, or company [3]. NER has been widely implemented in various fields to identify multiple named entities from a document.

This study aims to address the challenges mentioned above by utilizing NER in tourism destination reviews. In addition, the proposed approach aims to speed up obtaining relevant information for tourists. Specifically, the study focuses on identifying various entities, such as the names of tourism destinations, locations, and facilities. The Bidirectional Long Short-Term Memory (BLSTM) algorithm is utilized to build the NER model. BLSTM was chosen for its efficiency in performing NER.

This paper is structured as follows: Section 2 outlines the related work, while section 3 details the research methodology used in this study. Section 4 describes the experiment and results. Finally, section 5 provides the conclusion and future work of this study.

II. RELATED WORK

This section describes some previous research about NER and its application in tourism. Saputro et al. [4] proposed a semi-supervised learning algorithm called Yet
Another Two Step Idea (YATSI) combined with the Naïve Bayes Classifier for entity recognition in the tourism domain. Experiments showed that the system obtained an accuracy of 70.43% and F-measure of 69%.

Vijay and Srihar [5] applied a conditional random field (CRF) algorithm to extract entities in tourism data from Wikipedia and TripAdvisor. The results showed that the CRF model could achieve an F1 score of 83%. A study by [6] also utilized CRF with various features to extract tourist attraction entities from 92 English tourism articles. Their experiments presented that adding previous and next tag information could improve model performance with an F1 score of 95.75%. Zahra et al. [7] extracted tourist attraction entities using BLSTM-CRF. They used the same dataset as [6] and achieved an average F1 score of 75.25%.

Since the emergence of Bidirectional Encoder Representations from Transformers (BERT) proposed by [8], many studies have leveraged it in many NLP applications, including NER. In Xue et al. [9], BERT was incorporated with BLSTM and CRF to develop a named entity recognition system for Chinese tourism. Their proposed model attained the highest F1 score in identifying the organization, location, and thing entities with 71.13%, 91.46%, and 91.39%, respectively.

In Chantrapornchai and Tunsakul [10], BERT was implemented with SpaCy [11]. They extracted tourism-related name entities from Traveloka, TripAdvisor, and Hotels.com using SpaCy and BERT. From the experimental results, BERT outperformed SpaCy with more than 99% accuracy, while SpaCy obtained an accuracy of 95%. Bouabdallaoui et al. [12] built the NER system using data from the TripAdvisor forum on tourism in Morocco. The researchers used three transformer methods: BERT, RoBERTa, and XLM-RoBERTa. Of the three methods, BERT produced the best performance with an F1 score of 70.5%.

### III. RESEARCH METHODOLOGY

#### A. Data Collection

This study used datasets from prior research conducted by Nayoan et al. [1]. The data contained reviews of several tourism destinations in Indonesia scraped from the TripAdvisor web page. The total number of data is 2010 randomly selected reviews. The review data used in this study were entirely in Indonesian.

#### B. Preprocessing

The next stage was preprocessing, which aimed to clean the datasets from irrelevant elements. In this study, the preprocessing steps were deleting URLs, deleting non-ASCII characters, removing mentions and hashtags, adding spaces after periods (.), adding spaces after commas, removing excess spaces, and tokenization.

#### C. Data Labeling

The data labeling process in this study was done manually. Two annotators who understand Indonesian were employed to label the data. In this research, Cohen’s kappa [13] is utilized to measure the level of concurrence between two annotators. The equation for determining Cohen’s kappa (K) is presented below.

$$K = \frac{Pr(a) - Pr(e)}{1 - Pr(e)}$$  \hspace{1cm} (1)

From equation 1, Pr(a) represents the frequency of agreed labels, and Pr(e) represents the probability of agreement when observing data randomly. Cohen’s kappa (K) ranges from -1 to 1 and indicates the level of difference between the labels selected by annotators. A K value of 0 indicates agreement as if they were guessing randomly, and a K value closer to 1 implies a better dataset. In this study, the K value of was 0.996, which indicates our dataset is quite good.

The type of entity category used was divided into three, the name of the tourism place, the name of the tourism location, and the facilities. The tourism place name category (WIS) was used to identify the name entities of tourist attractions such as Mount Merapi, Parangtritis Beach, Ijen Crater, and Borobudur Temple. The location name category (LOC) indicates the location of the place or city of the tourist attraction, for example, Yogyakarta, Bandung, Bali, Lombok, and Central Java. At the same time, the category of facilities (FAS) was used to identify the services provided at tourist attractions, such as prayer rooms, bathrooms, toilets, and parking.

In this study, the labeling process referred to the Beginning Inside Outside (BIO) format because the identified entity contained several tokens [14]. They were labeled with BIO format identified entities based on their word order. Entities in the first order were labeled ‘B’ (Beginning). If the entity contained more than one word, the word in the second-order was given the label ‘I’ (Inside). The terms that did not include entities would be labeled ‘O’ (Outside). Overall, seven labels were used in the study, namely B-WIS, I-WIS, B-LOC, I-LOC, B-FAS, I-FAS, and O. Table I is an example of entity labeling carried out in this study. The results of data labeling can be seen in Table II.

**Table I: Example of Entity Labeling**

<table>
<thead>
<tr>
<th>Entity Example</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pantai Parangtritis</td>
<td>Pantai: B-WIS</td>
</tr>
<tr>
<td></td>
<td>Parangtritis: I-WIS</td>
</tr>
<tr>
<td>Provinsi Jawa Tengah</td>
<td>Provinsi: B-LOC</td>
</tr>
<tr>
<td></td>
<td>Jawa: I-LOC</td>
</tr>
<tr>
<td></td>
<td>Tengah: I-LOC</td>
</tr>
<tr>
<td>Kamar mandi</td>
<td>Kamar: B-FAS</td>
</tr>
<tr>
<td></td>
<td>Mandi: I-FAS</td>
</tr>
</tbody>
</table>
D. NER Modeling

This section describes a deep learning architecture using BLSTM in our study. Figure 1 illustrates the proposed BLSTM architecture. The architecture consists of input, embedding, BLSTM, dropout, and dense layers.

![BLSTM Architecture](image)

The input layer in the neural network consists of artificial input neurons. It brings the initial data (in this case, text) into the system for further processing by the next artificial neuron layer. The next layer is the embedding layer which functions as a feature extractor from the input data in text. Feature extraction aims to obtain important information from the sample representing the data. This study used Word2vec to extract features. Applying Word2vec for NER has been shown to improve performance [15]. To build the Word2vec, we applied two Word2vec algorithms, skip-gram, and continuous bag-of-words (CBOW). In this study, the two algorithms were used to compare which one could give better results for NER.

The next layer is BLSTM, a development of the long short-term memory (LSTM) algorithm that aims to improve classification performance on sequential data such as text [16]. The BLSTM algorithm performs sequential input data training using two LSTM layers, forward and backward [17]. Furthermore, a dropout layer is added to this architecture to prevent overfitting during the training process [18]. The last layer in this architecture is the dense layer which predicts the system output.

<table>
<thead>
<tr>
<th>Label</th>
<th>Number of Tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td>B-WIS</td>
<td>5533</td>
</tr>
<tr>
<td>I-WIS</td>
<td>2058</td>
</tr>
<tr>
<td>B-LOC</td>
<td>1396</td>
</tr>
<tr>
<td>I-LOC</td>
<td>110</td>
</tr>
<tr>
<td>B-FAS</td>
<td>1901</td>
</tr>
<tr>
<td>I-FAS</td>
<td>241</td>
</tr>
<tr>
<td>O</td>
<td>105325</td>
</tr>
<tr>
<td>Total tokens</td>
<td>116564</td>
</tr>
</tbody>
</table>

**E. Model Evaluation**

The model evaluation aims to measure the performance of the NER model that has been built. The F1 score was chosen as an evaluation measure in this study. The F1 score was considered more suitable than using the accuracy due to the class imbalanced [19]. The F1-score is the harmonic mean of precision and recall. The following are equations to calculate precision, recall, and F1 score.

\[
\text{Precision} = \frac{TP}{TP + FP}
\]

\[
\text{Recall} = \frac{TP}{TP + FN}
\]

\[
F1 \text{ score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}
\]

From the formula above, TP (True Positive) stands for the positive values that were correctly predicted. TN (True Negative) describes the negative values that were correctly predicted. FP (False Positive) represents the value of wrongly positive predictions. FN (false negative) describes the value of wrong negative predictions.

**IV. EXPERIMENTS AND RESULTS**

A. Experimental Setup

In our experiments, we applied several scenarios by applying different hyperparameter values to obtain the best NER model. Table III shows the initial scenario with the parameter values used as a reference in the experiment. In this initial scenario, we configured several parameters: dropout, Word2vec algorithm, LSTM unit, batch, epoch, and optimization function. We followed the previous studies to assign the values for each parameter. We set the dropout value to 0.5 as one of the parameters in the LSTM layer to avoid overfitting [20]. We selected Skip-gram to build the Word2vec representation due to its excellent performance [19]. For the LSTM unit, batch, epoch, and optimization function values, we set them to 128, 32, 30, and Adam [7], respectively. Table III presents the initial scenario parameters in our experiments.

**TABLE III**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dropout</td>
<td>0.5</td>
</tr>
<tr>
<td>Word2vec algorithm</td>
<td>Skip-gram</td>
</tr>
<tr>
<td>LSTM unit</td>
<td>128</td>
</tr>
<tr>
<td>Batch</td>
<td>32</td>
</tr>
<tr>
<td>Epoch</td>
<td>30</td>
</tr>
<tr>
<td>Optimization function</td>
<td>Adam</td>
</tr>
</tbody>
</table>
TABLE IV
SCENARIO SUMMARY

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Hyperparameter</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Learning rate</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0001</td>
</tr>
<tr>
<td>2</td>
<td>Dropout</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.50</td>
</tr>
<tr>
<td>3</td>
<td>Word2vec algorithm</td>
<td>Skip-gram</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CBOW</td>
</tr>
<tr>
<td>4</td>
<td>Batch</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td></td>
<td>64</td>
</tr>
<tr>
<td>5</td>
<td>LSTM unit</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
</tr>
<tr>
<td>6</td>
<td>Epoch</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td></td>
<td>40</td>
</tr>
<tr>
<td>7</td>
<td>Optimization function</td>
<td>Adam</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Nadam</td>
</tr>
</tbody>
</table>

In this study, we conducted seven scenarios to compare hyperparameters used in the neural network model. TABLE IV provides a summary of the scenarios investigated. Scenario 1 evaluated the impact of varying the learning rate on model performance, with three values tested: 0.01, 0.001, and 0.0001. A lower learning rate can potentially yield optimal results but requires a longer learning time. The default learning rate value in Adam's optimization is 0.001. In scenario 2, we investigated the number of dropouts needed to find the most appropriate value. Scenario 3 aimed to compare the Word2vec performance between CBOW and Skip-gram. Scenario 4 tested the impact of varying batch counts. In scenario 5, we compared the number of LSTM units used. Scenario 6 evaluated the impact of the number of epochs. Finally, we compared the optimization functions used in scenario seven, namely Adam and Nadam optimizers.

B. Results and Discussion

In this sub-section, we provide the results of each scenario experiment. In Scenario 1, a comparison of the learning rates was carried out; it was found that of the three learning rates used, the learning rate that had the best performance based on the F1 score obtained was the learning rate with a value of 0.01, which was 94.2%, as shown in Table V.

TABLE V
RESULTS OF SCENARIO 1

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Learning Rate</th>
<th>F1 score</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.01</td>
<td>99.5 %</td>
<td>83.6 %</td>
</tr>
<tr>
<td></td>
<td>0.001</td>
<td>99.4 %</td>
<td>80.7 %</td>
</tr>
</tbody>
</table>

TABLE VI presents the results of scenario 2. We found that applying dropout can improve the average F1 score. In addition, when we set the dropout value to 0.5, we obtained the best average F1 score of 94.2%.

TABLE VI
RESULTS OF SCENARIO 2

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Dropout</th>
<th>F1 score</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>-</td>
<td>99.5 %</td>
<td>99.4 %</td>
</tr>
<tr>
<td></td>
<td>0.25</td>
<td>99.4 %</td>
<td>82.6 %</td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>99.5 %</td>
<td>83.6 %</td>
</tr>
</tbody>
</table>

In Scenario 3, using Skip-gram algorithm to build Word2vec produced slightly better average F1 score value compared to CBOW. Table VII provides the results of Scenario 3.

TABLE VII
RESULTS OF SCENARIO 3

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Word2vec</th>
<th>F1 score</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>CBOW</td>
<td>99.5 %</td>
<td>99.5%</td>
</tr>
<tr>
<td></td>
<td>Skip-gram</td>
<td>99.5 %</td>
<td>83.6 %</td>
</tr>
</tbody>
</table>

Table VIII shows the result comparison for two batch sizes. It is shown that the batch value of 32 gives a better average F1 score of 94.1% Kesalahan! Sumber referensi tidak ditemukan.

TABLE VIII
RESULTS OF SCENARIO 4

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Batch</th>
<th>F1 score</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>32</td>
<td>99.5%</td>
<td>83.6%</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>99.5%</td>
<td>83.1%</td>
</tr>
</tbody>
</table>

As shown in Table IX, we compared the number of units for the LSTM layer. Based on our experiments, LSTM units of 128 produced a slightly better average F1 score value.

TABLE IX
RESULTS OF SCENARIO 5

<table>
<thead>
<tr>
<th>Scenario</th>
<th>LSTM</th>
<th>F1 score</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>128</td>
<td>99.5%</td>
<td>83.6%</td>
</tr>
</tbody>
</table>
In Scenario 6, a comparison of the epoch sizes used was carried out. As shown in Table X, we observed that an epoch of 40 gives better 0.1% than an epoch of 30 with an average F1 score of 94.3%.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Epoch</th>
<th>F1 score</th>
<th>Average F1 score</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Micro</td>
<td>Macro</td>
</tr>
<tr>
<td>6</td>
<td>30</td>
<td>99.5%</td>
<td>83.6%</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>99.5%</td>
<td>83.8%</td>
</tr>
</tbody>
</table>

In Scenario 7, we performed a comparison for the optimization function using Adam and Nadam. Our experiments showed that the Adam optimization function provides a better average F1 score of 93.7%. Table XI presents the results of Scenario 7.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Optimization</th>
<th>F1 score</th>
<th>Average F1 score</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Micro</td>
<td>Macro</td>
</tr>
<tr>
<td>7</td>
<td>Adam</td>
<td>99.5%</td>
<td>83.8%</td>
</tr>
<tr>
<td></td>
<td>Nadam</td>
<td>99.5%</td>
<td>82.4%</td>
</tr>
</tbody>
</table>

We obtained the best hyperparameter results based on the seven scenarios we conducted. Table XII presents the best value for each hyperparameter. Therefore, our NER model used a learning rate of 0.01, a dropout of 0.5, the Skip-gram algorithm for building Word2vec representation, a batch size of 32, the number of LSTM units as 128, an epoch of 40, and Adam optimization function. The optimal hyperparameter configuration was determined upon comparing the scenarios, yielding a model with the highest average F1 score of 94.3%.

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learning rate</td>
<td>0.01</td>
</tr>
<tr>
<td>Dropout</td>
<td>0.5</td>
</tr>
<tr>
<td>Word2vec algorithm</td>
<td>Skip-gram</td>
</tr>
<tr>
<td>Batch</td>
<td>32</td>
</tr>
<tr>
<td>LSTM unit</td>
<td>128</td>
</tr>
<tr>
<td>Epoch</td>
<td>40</td>
</tr>
<tr>
<td>Optimization function</td>
<td>Adam</td>
</tr>
</tbody>
</table>

Figure 2 shows the confusion matrix of the NER model based on the best hyperparameter values. We can see a few errors in determining the name entities. For example, in Figure 3, the second word in the phrase ‘Nusa Dua’ is incorrectly detected as I-LOC. This error may happen because the term ‘dua’ can sometimes be identified as a number depending on the context.

V. CONCLUSION AND FUTURE WORKS

In this study, we have developed a named entity recognition model for the tourist domain using BLSTM. To build the model, we performed some scenarios to get the best hyperparameters. As a result, we obtained the best average F1 score of 94.3% by comparing the scenarios. The current study involved a manual selection of the optimal hyperparameters, but automated hyperparameter tuning is recommended for future investigations. Furthermore, utilizing pre-trained language models such as BERT can be explored as an alternative approach.
REFERENCES


