Journal of Engineering Science and Technology
Vol. 17, No. 1 (2022) 0730 - 0744
© School of Engineering, Taylor's University

A HYBRID CNN-BILSTM MODEL FOR DRUG NAMED ENTITY RECOGNITION

DHOMAS HATTA FUDHOLI*, ROYAN ABIDA N. NAYOAN, AHMAD FATHAN HIDAYATULLAH, DEDE BRAHMA ARIANTO

Department of Informatics, Universitas Islam Indonesia
Jl. Kaliurang Km. 14.5, 55584, Sleman, Yogyakarta, Indonesia
*Corresponding Author: hatta.fudholi@uii.ac.id

Abstract

Named Entity Recognition (NER) has been proven to be successful and useful in many domains in extracting real-world entities. Under the medical domain, this study aims to extract drug name entities. To extract drug name entities, drug product, ingredient, and drug dose are used as categories to classify extracted entities. 18,075 sentences containing drug-related information have been collected from articles written in Bahasa Indonesia. Our main contribution lies in presenting the drug NER model and evaluation of the CNN-BiLSTM architecture in the Indonesian language. We used a hybrid Convolutional Neural Network - Bidirectional Long-Short Term Memory (CNN-BiLSTM) deep learning architecture model that automatically detects word- and character-level features. We trained the architecture with six different hyper-parameter sets to find the best model. Based on the experiments, the best achievement was obtained by one of the models in term of its F1 score. The model uses 2 layers of CNN with a kernel size of 7, a CNN filter of 50, a single LSTM layer with 200 hidden units, and additional chunk tag-based feature. The model achieved an f1-score of 0.892, a precision of 0.881, and a recall of 0.903.

Keywords: CNN-BiLSTM, Deep learning, Drug, Named entity recognition.

1. Introduction

An abundant amount of accessible data through the internet is bound to have less reliable information and is caused by difficulties in extracting valuable information. Information extraction is necessary to obtain essential and useful things from data. A subtask of information extraction, Named Entity Recognition (NER), is applied to extract useful information from a huge amount of data [1, 2]. NER is defined as the process of assigning unique identifiers to extracted entities based on domain knowledge [3, 4]. NER task recognizes unsigned words into categories in lexical analysis, information extraction and retrieval, intelligent question and answering, and other natural language processing tasks [5].

Named Entity Recognition (NER) has been proven to be successful and useful in extracting real-world entities, resulting in the proposal of several research methods on many various domains. NER will identify named entities within a text and classify them into predetermined categories, like persons (PER), organizations (ORG), locations (LOC), and miscellaneous names (MISC) [6]. Here is an example of NER: [ORG U.N.] official [PER Ekeus] heads for [LOC Baghdad]. Each term in the sentence is assigned to a specific category: U.N. is assigned as an organizations (ORG) category, Ekeus as a person (PER), and Baghdad as a location (LOC) [6]. Due to different research that is carried out by various scientists, unique identifiers and categories for the extracted entities may vary under different domains.

Many studies have been carried out by scientists in extracting named entities in the health-care domain. Various machine learning models and deep learning models are designed to improve the quality and performance of the present clinical NER model. A previous clinical NER work carried out by Luo et al. [7] tested different deep learning models and even combined different individual models, including BiLSTM-CRF, BiLSTM-CNN-CRF, BiLSTM+CNN-CRF, CNN-CRF, and Lattice LSTM, to extract Chinese clinical named entities. The named entities are anatomy, symptom, independent symptom, drug, and operation. They used the “BIOES” tag as a representation of their entity. Their work applied character embedding as a feature processing. The result shows that the ensemble model built by combining these models’ results with majority voting reached the highest f-score of 93.16%. This model has a promising result and achieves an f-score of higher than 80% under “strict” criteria and higher than 90% in “relaxed” criteria.

Deep Learning technologies and architectures have been widely used to model NER in various datasets. Some recent works in the NER area which use a well-known dataset, CoNLL03, sourced from Reuters news, are [8-11]. All of them use RNN-based architecture, including LSTM and GRU. Chiu and Nichols [3] proposed a hybrid BiLSTM and CNN architecture with word and character-level features. They also experimented by using different types of lexicons, word embeddings, features, and models. This research also used the “BIOES” tag to represent the entities. This research experimented with different word embedding and lexicon on a different dataset. Implementing Deep Learning architecture from their works is very promising by giving higher than 90% accuracy on CoNLL03.

Liu et al. [12] researched drugs NER with various approaches that classify named entities into four different categories. Their Drug Name Recognition (DNR) identifies unstructured medical texts and automatically extracts and classifies drug name entities into their predefined classes. Each type of the four different approaches used in this research, namely dictionary-based, rule-based, machine learning-based, and hybrid approaches, have its advantages. They also show the experiment on different types of texts. A hybrid approach using machine-learning and dictionary outperformed the other approaches and achieved the highest f-score of 92.54%.

Krallinger et al. [13] studied drugs and chemical extraction using Gold Standard data which was previously carefully prepared by specially trained chemists to label the data manually. They applied various methods to the data to obtain the best performance. They also show the experiment on different techniques, preprocessing, post-preprocessing, dictionary lookup, word-level features, lookup features, and document features on CEM. They evaluated two crucial aspects: the Chemical Document Indexing (CDI) task and the Chemical Entity Mention (CEM) recognition. CDI covered the indexing of the documents with chemicals, while CEM is responsible for finding the exact mentions of chemicals in the text. Using machine learning methods, they achieved the highest f-score of 87.39% on CEM tasks and 88.20% on CDI tasks.

Batbaatar and Ryu [14] studied Health-Related Named Entity Recognition. They use a healthcare-domain ontology to extract health-related entities. Their model can identify diseases, drugs, symptoms for biosurveillance, related properties and activities, and adverse drug events. The research uses the data taken from user messages on Twitter that contained one medical entity in UMLS. The features that are used are word embedding, character embedding, and POS tagging. The model they employed, the BiLSTM-CRF model, achieved a precision of 93.99%, recall of 73.31%, and F1-score of 81.77% for disease or syndrome HNER; a precision of 90.83%, recall of 81.98%, and F1-score of 87.52% for a sign or symptom HNER; and a precision of 94.85%, recall of 73.47%, and F1-score of 84.51% for pharmacologic substance named entities.

A clinical named entity recognition carried out by Wu et al. [15] investigated two different deep learning models, Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN), to extract named entities. They investigated RNN and CNN architectures by comparing them with three baselines Conditional Random Fields (CRF) models and two recent clinical NER systems. The corpus used are i2b2 2010 and MIMIC II. They also show the comparison of all their machine learning model with different features. The features are word embedding, linguistic, discourse, and many others. The RNN model with word embedding gives a higher f-score result of 85.94% and outperformed other supervised and unsupervised NER systems. Yepes and Mackinlay [16] also carried out a medical NER which uses a sequence-to-sequence neural network. This research uses Twitter data to perform named entity extraction and classify them into three categories: disease, symptom, and pharmacological substance. Their model outperformed the previous study, which also used the same Twitter dataset.

In this work, we aim to predict drug-related named entities such as the drug products, ingredients, and drug dose from health articles in Bahasa Indonesia. Our main contribution lies in presenting the model and evaluation of the BiLSTM-CNN architecture on Indonesian language datasets. The syntax or arrangement of the words in the Indonesian language is different from the available English-based model. We propose a deep learning architecture using a hybrid Convolutional Neural Network and Bidirectional Long-Short Term Memory (CNN-BiLSTM). We fine-tuned the model in both CNN (layer, kernel size, and filter number) and BiLSTM (layer and hidden units) architecture to get the best Drug NER model.

2. Methods

In this section, we elaborate on the methods used in our study. Our work consists of several stages, including collecting drug-related data, data preprocessing, POS tagging, entity tagging, training and tuning the model, and evaluation. In the first stage, we collected the data by using various health-care websites to get a list of drugs and get specific information on each drug through Google snippets. The texts we obtained are then preprocessed to get a cleaner version of the texts. Since entity tagging works on top of POS tagging, POS tagging is done before proceeding to the next stage. We train the model using the tagged entities and tune them to obtain the best model in extracting drug named entities. In the last step, we evaluate our models by using f-measure.

2.1. Data collection

To obtain the data, we identify several Indonesian health care websites that provide a list of drug names. The web-accessible URLs of drug name sources are listed in Table 1. A list of drug names from the websites is used as keywords for the next step. After the list of drug names is obtained, we retrieved the more detailed information of the drug by using Google snippets. To get Google to provide a short snippet, we use a simple keyword of “kandungan” which means “the content of” and then followed by the drug’s name. As an example, a keyword of “kandungan panadol” gave us a short snippet of “Panadol merah bermanfaat untuk meredakan sakit kepala dan sakit gigi. Tiap tablet Panadol Extra mengandung 500 mg paracetamol dan 65 mg kafein.”. A total of 18,075 sentences containing drug information has been successfully retrieved.

Table 1. List of the web-accessible URLs.

Website	URL
Klik Dokter	https://www.klikdokter.com/obat
Sehatq	https://www.sehatq.com/obat
Hello Sehat	https://hellosehat.com/obatan-suplemen/obat/

2.2. Data preprocessing

Data preprocessing is a process of preparing the raw data and making it suitable for training a machine learning model [17]. In this work, we perform various form of preprocessing tasks, to name a few: (i) removing date, since date is insignificant information in drug name entities recognition; (ii) removing punctuations (e.g., “setiap kemasan Acepress mengandung zat aktif sebagai berikut : Captopril 25 mg / tablet.” to “setiap kemasan Acepress mengandung sebagai berikut Captopril 25 mg tablet”); (iii) removing non-ASCII characters such as “Â” and “â€” which often appear in our obtained texts; (iv) replacing some measurement abbreviations to its original unit of measurement (e.g., “mg” to “miligram); and (v) removing stopwords. Preprocessing tasks must be done sequentially and cannot be done randomly. Therefore, the preprocessing arrangement in different case might be different depending on the cleaning needs.

2.3. POS (Part of Speech) Tagging

Part of Speech (POS) tagging is an essential part of NER. POS tagging will assign each word in the corpus to its POS tag based on the definition and context. POS tagging aims to categorize word classes into nouns, adjectives, verbs, etc., [18]. In this work, the POS tagging task is performed using the tool provided by Kumparan's NLP Services, which provides the POS tagging function for Bahasa Indonesia that's publicly available [19]. Table 2 shows the POS tag sets used in Kumparan's NLP Services.

Table 2. POS tag sets.

Tag	Description	Tag	Description
ADV	Adverbs. Includes adverb, modal, and auxiliary verb	PR	Pronoun
CC	Coordinating conjunction	RP	Particle
DT	Determiner/article	SC	Subordinating Conjunction
FW	Foreign word	UH	Symbols and Punctuations
IN	Preposition	VB	Verb
JJ	Adjective	WH	Question words
NEG	Negation	ADJP	Adjective Phrase
NN	Noun	DP	Date Phrase
NNP	Proper Noun	NP	Noun Phrase
NUM	Number	VP	Verb Phrase
NUMBVP	Number Phrase	PR	Pronoun

2.4. Entity tagging

This stage allows a particular word that fits the grammar criteria to be tagged as an entity. The entities used in this research are (MED) to indicate the name of the medicine, (INGRD) as an ingredient of the medicine, (DOSE) as a specific amount or dose of medication, and (O) for the word that has no labelled entity.

In this study, we also perform both chunking and chinking to extract some important information from text. Chunking is performed after POS-tag as this requires POS-tags as inputs to extract phrases. A standard set of Chunk tags usually used is Noun Phrase (NP) and Verb Phrase (VP). Regular expression rules, which are typically known as "grammar", are used to define chunk patterns. We describe two chunk patterns to extract drug ingredients, four chunk patterns to extract the drug's product, and a chunk pattern to extract the drug's dose. Chunk patterns in this research are defined by following the patterns often used in the writing of Indonesian medicinal texts we obtained. An example of grammar is as follows: {<NUM><NN|NUM><VB>*}. Chunk pattern started with NUM tag followed by NN or NUM tag, followed by zero or more VB tags helps in tagging drug DOSE. In extracting dose, we also performed chinking to remove the selected extracted word after chunking. An example of chinking is as follows: }<NUM><NN.*|NUM><VB>{. Chink pattern started with NUM tag followed by zero or more NN tags or a NUM tag, followed by VB tag will remove the extracted words. In our defined grammar, a chunk pattern is followed by a chink pattern. A chunk pattern is started with the character "{" and ended with "}". A chink pattern is started with the character "}" and ended with "{". Fig. 1 shows the grammar used to tag entities in this research.

```
Grammar = r"""
       INGRD: {<VB><NN><NNP>*}
               }<VB><NN>{
       INGRD: {<NNP>*<SC>*<VB><IN><NNP>*}
               }<SC>*<VB><IN><NNP>*{
       INGRD: {<VB>*<NN>*<SC>*<VB><NNP>*}
               }<VB>*<NN>*<SC>*<VB>{
       INGRD: {<VB>*<NN>*<SC>*<VB><NN><JJ><NNP>?}
               }<VB>*<NN>*<SC>*<VB>{
       MED: {<VB><IN><NN>?<NNP>*}
               }<VB><IN><NN>?{
       MED: {<NN.*>?<VB><VB>}
                 }<VB><VB>{
       MED: {<NNP>*<ADV><IN><VB>}
               }<ADV><IN><VB>{
       MED: {<NNP>*<NN>*<VB><NN><SC><VB>}
               }<VB><NN><SC><VB>{
       DOSE: {<NUM><NN|NUM><VB>*}
               }<NUM><NN.*|NUM><VB>{
       INGRD: {(<NNP><FW|NNP>*<DOSE>)|
                 (<FW><NNP|FW>*<NN><DOSE>)}
               }<DOSE>{
       MED: {<NNP>+?<NUM><NNP|NUM|NN>}
       MED: {<NNP|NN|FW>*<VB><DOSE>}
             }<VB><DOSE>{
       MED: {<NN.*|FW>*<VB><INGRD>}
               }<VB><INGRD>{
       MED: {<NN.*>*<VB><NUM>}
             }<VB><NUM>{
       INGRD: {<MED><VB><DOSE><NN.*|FW>*}

          }<MED><VB><DOSE>{                """
```

Fig. 1. Chunk patterns.

Furthermore, we compare two different model sets by removing the chunk tag on the other set. The first set uses four different feature field data formats: the word, part-of-speech tag, chunk tag, and named entity tag. The second set contains only three different feature fields: the word, part-of-speech tag, and named entity tag. Below is an example of the first set data format (to note that the word "mengandung" means contains):

Acepress	NNP	B-NP	B-MED
mengandung	VB	B-VB	O
Captopril	NNP	B-NP	B-INGRD
12,5	NUM	O	B-DOSE
miligram	NN	B-NP	I-DOSE
tablet	NN	I-NP	O

The above data format contains four fields: the word, part-of-speech tag, chunk tag, and named entity tag. B-I-O tagging is commonly used as tagging format in a chunking task. (B) tag is assigned at the beginning of the phrase. (I) is assigned at the inside. (O) is assigned to other phrases that are not defined in grammar rules [20]. An explanation of B-I-O tagging in the above data format is as follows: The word “Acepress” is tagged as B-MED since Acepress is the first word (beginning word) that’s recognised as a medication product. The word “mengandung” is tagged as O because it isn’t recognised in the grammar rules. The word “Catopril” is tagged ad B-INGRD since it is the beginning word recognised as a drug ingredient. “12,5 miligram” is tagged as B-DOSE and I-DOSE since both words are recognised as drug’s dose based on the grammar rules. The word “12.5” is tagged as B-DOSE because it is recognised as the beginning of the drug’s dose and followed with “miligram” which is then tagged as I-DOSE because it is located inside the phrase. The last word “tablet” is tagged as O since the grammar rules do not recognise it.

2.5. CNN-BiLSTM model

Convolutional neural networks (CNN) have been used in modelling character-level features. Previous research carried out by Santos & Guimarães [21] has employed CNNs and successfully extract character-level features in NER. For each word, convolution layer and max layer were applied to extract character-level features. Our model uses the pretrained Word2Vec Indonesian word embeddings, which were trained on Wikipedia vocabulary and has vector length equals to 50. The final network model used in this research is shown in Fig. 2.

CNN-BiLSTM model starts with an input layer in the workflow. This model uses two dropouts and three embeddings: case embedding, words embedding, and character embedding. The features and dimensions are as shown in Fig. 2. This layer composes artificial input neurons and brings the initial data into the system for further processing by subsequent layers of artificial neurons. The input layer is then followed by a character embedding layer to create dense and meaningful representations of high-dimensional categorical data of the character. A dropout layer is applied to reduce overfitting. The next layer is a 1D convolutional layer and a max pooling layer that extracts character-level feature from each word. In the max pooling layer, flatten operation is applied to transform the dimensional matrix of features into a vector that can be fed into a fully connected neural network. Another dropout layer is added and followed by a concatenate layer. The layer concatenates a list of inputs: word embedding, case embedding, and the extracted character-level feature. Word embedding maps the words from the vocabulary to vectors. The case embedding maps several different cases to vectors. The concatenated lists are then fed into a bidirectional long-short term memory (BiLSTM) network and followed by the last layer, which is a dense layer with softmax activation to produce a probability distribution. Time distributed layer is applied in several layers in this model. This wrapper allows to apply a layer to every temporal slice of an input.

Figure 3 shows how BiLSTM tagging works. It shows the multiple tables lookup word-level feature vectors [3]. In this research, words and characters are transformed into continuous vector representations by using lookup tables. The results are then concatenated and fed into our BiLSTM network. We use a convolutional neural network to extract character-level features, which previously has been successfully applied to Spanish and Portuguese NER [21]. The CNN shown in Fig. 4 extracts a fixed-length feature vector from character-level features. Convolution and max layer are used to extract the new feature vector, which is the per-character feature vectors like character embeddings. Several special PADDING characters are used to pad the words on both sizes. The padding depends on the CNN’s window size.

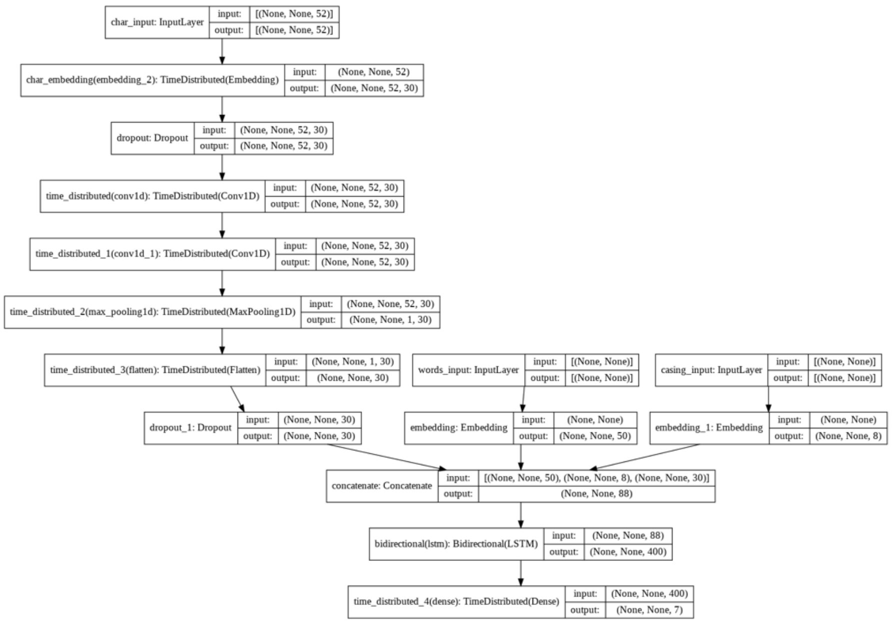

char_input: InputLayer | input: [(None, None, 52)] | output: [(None, None, 52)]
char_embedding(embedding_2): TimeDistributed(Embedding) | input: (None, None, 52) | output: (None, None, 52, 30)
dropout: Dropout | input: (None, None, 52, 30) | output: (None, None, 52, 30)
time_distributed(conv1d): TimeDistributed(Conv1D) | input: (None, None, 52, 30) | output: (None, None, 52, 30)
time_distributed_1(conv1d_1): TimeDistributed(Conv1D) | input: (None, None, 52, 30) | output: (None, None, 52, 30)
time_distributed_2(max_pooling1d): TimeDistributed(MaxPooling1D) | input: (None, None, 52, 30) | output: (None, None, 1, 30)
time_distributed_3(flatten): TimeDistributed(Flatten) | input: (None, None, 1, 30) | output: (None, None, 30)
words_input: InputLayer | input: [(None, None)] | output: [(None, None)]
casing_input: InputLayer | input: [(None, None)] | output: [(None, None)]
dropout_1: Dropout | input: (None, None, 30) | output: (None, None, 30)
embedding: Embedding | input: (None, None) | output: (None, None, 50)
embedding_1: Embedding | input: (None, None) | output: (None, None, 8)
concatenate: Concatenate | input: [(None, None, 50), (None, None, 8), (None, None, 30)] | output: (None, None, 88)
bidirectional(lstm): Bidirectional(LSTM) | input: (None, None, 88) | output: (None, None, 400)
time_distributed_4(dense): TimeDistributed(Dense) | input: (None, None, 400) | output: (None, None, 7)

Fig. 2. CNN-BiLSTM model architecture.

The lookup table is initialized randomly with the values drawn from a uniform distribution between -0.5 and 0.5 which will output a character embedding of 25 dimensions. The character set has all the unique characters inside the dataset and the special tokens that are PADDING and UNKNOWN. The PADDING token is used for the CNN, and the UNKNOWN token is for the other characters.

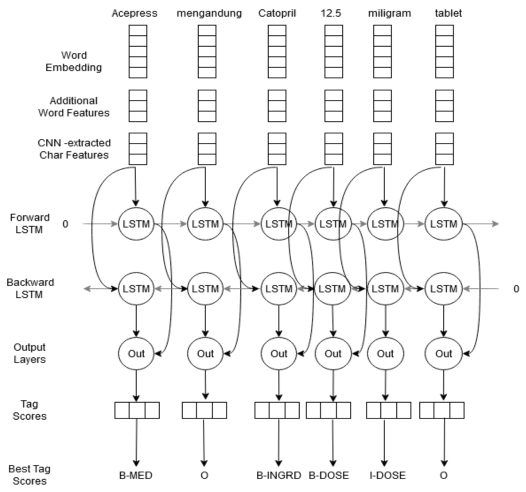


Fig. 3. BiLSTM-CNN for tagging drug entities.

Additional word-level features use Collobert's method, which uses a separate lookup table to add a capitalization feature with the following options: allCaps, upperInitial, lowercase, mixedCaps, noinfo [22]. The additional character-level features use a lookup table that outputs a 4-dimensional vector representing the type of the character (upper case, lower case, punctuation, other).

In this research, stacked BiLSTM with LSTM units transform word features into named entity tag scores. The features that are extracted from each word are then fed into a BiLSTM network. Each output of each time step is then decoded into two vectors of log-probabilities for each tag. The final output is the added two vectors. The LSTM is initialized with zero vectors.

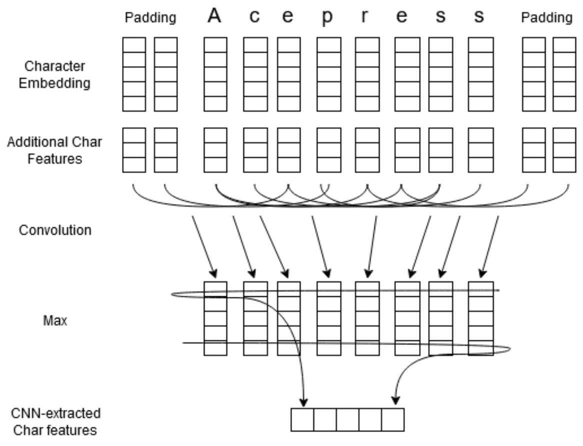


Fig. 4. CNN in extracting character features.

2.6. Evaluation

The performance in this task is measured with F_β=1 rate as stated in Eq. (1). With β=1, the F1 score is the weighted average of Precision and Recall. Precision is a measure of named entities found in the corpus that has been classified correctly by the learning model. A recall is a measure of the relevant results of named-entities retrieved by the NER model [6].

$$F_\beta = \frac{(\beta^2+1)*precision*recall}{(\beta^2*precision*recall)} \tag{1}$$

3. Results and Discussions

3.1. Data distribution

This study divided the datasets into three different parts: training set, validation set and test set. Previous studies show that if the dataset is divided as follows, 20-30% of the data is used as a testing set and 70%-80% of the data as a training set, the model can obtain the best result [23]. For this research, the train set and the test set are 70% and 30% of the total sentences, and the validation set uses 30% of the test set. Tables 3 and 4 illustrate the overview of the sizes of the data and the number of named entities per dataset. Here, the tags used are MED for the name of the medicine, INGRD for ingredients, DOSE fore Dosage, and O for Other. The O tag here has the proportion of being higher than the other tag. This is normal because in the collected sentences, there are many words used to explain what the medicine is for, or to explain how many times the medicine should be taken. Hence, these unrelated words are tagged as O and have high proportion.

Table 3. Number of sentences and tokens in each data file.

Drugs Data	Sentences	Tokens
Training set	8857	292527
Validation set	3795	75266
Testing set	5423	103802

Table 4. Number of named entities per dataset.

Drugs Data	MED	INGRD	DOSE	OTHER
Training set	13022	16248	8131	255126
Validation set	4415	7530	1372	53148
Testing set	5691	10126	10132	74243

3.2. Hyper-parameter setting

Differs from [3], in this research, we applied bucketing approach to speed up the training process. This approach improves the training speed by clustering sequences into buckets by their length, thus finding a compromise between data structuring and shuffling [24]. Bucketing is most suitable for training tasks such as speech recognition, language modelling, and other machine learning tasks using GRU and LSTM. In all scenarios, we also applied dropout layers to reduce overfitting [25]. As suggested by a previous study in [26], a dropout of 0.5 is selected to accompany the CNN and the LSTM layer. A dropout will ignore selected neurons randomly to help and prevent overfitting. All scenarios run in 50 epochs and used Nadam optimizer to compile the model. We performed hyper-parameter optimization and selected the best settings using our number choose. There are six different models that will be trained using different CNN Kernel Size, CNN Filter number, CNN Layers number, LSTM Unit number and LSTM layers number. Table 5 shows the different combination of the hyperparameters. The six models will be trained twice, the first batch of training will use chunk tag, while the second batch does not.

Table 5. Hyper-parameter used for experiments.

Model	Hyper-parameter				
	CNN Kernel Size	CNN Filter	CNN Layers	LSTM Unit	LSTM Layers
1	3	30	1	100	1
2	3	30	2	100	1
3	3	50	2	200	1
4	7	50	1	200	1
5	7	50	1	200	2
6	7	30	2	200	1

3.3. Model training result

To summarize the training result, we present all trained models' precision, recall, and F1 score of the train set and test set in Table 6. In Table 6, we can see that there are two blocks of training results. The first block contains the training results of the different models mentioned in Table 5 that use the chunk tag feature, while the other block does not. Our analysis through the experiment shows that the chuck tag is an essential feature, which achieves a better model's performance. Our experiment shows better overall scores for the models that use chunk tag than those that do not. From the table, we can see that, overall, all model setup gives score higher than 0.82 in all aspects (precision, recall, and F1). The best model with the highest F1 score in the test set is model 6a with the chunk tag feature. The model achieves 0.881 in precision, 0.903 in recall, and 0.892 in F1.

We give a performance comparison of our model with other existing work in the field of medical NER. The results are shown in Table 7. From the table, we can see that our model is as good or better than the other model. However, while the study in [7] seems to have higher scores overall, we need to note that we have a specific study domain that is different than the previous research. This specific domain is related to the localization of natural language structure which has been implied as the main ideas in Introduction section.

Table 6. Precision, Recall, and F1-score on train set and test set.

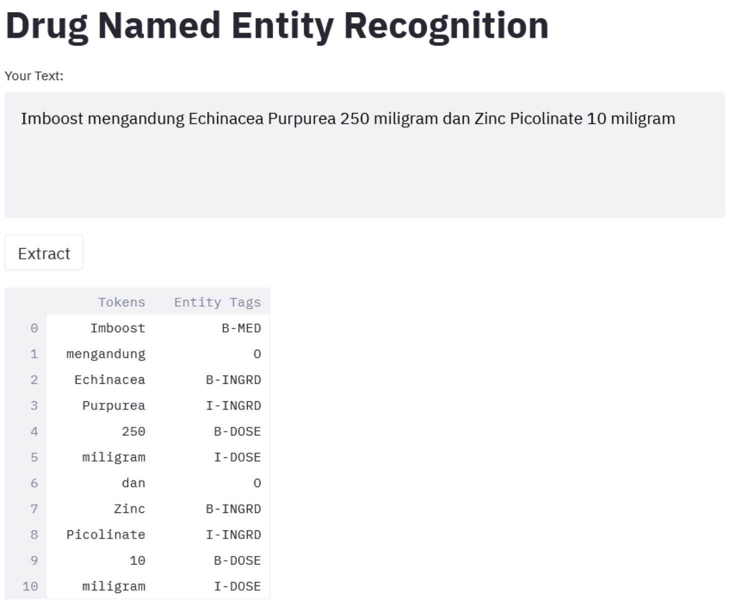
Model	Dataset	Precision	Recall	F_{β}=1	Chunk Tag
1a	Train set	0.880	0.892	0.886	Yes
	Test set	0.888	0.893	0.891	
2a	Train set	0.845	0.867	0.855	
	Test set	0.846	0.870	0.858	
3a	Train set	0.879	0.893	0.886	
	Test set	0.884	0.898	0.891	
4a	Train set	0.890	0.884	0.887	
	Test set	0.894	0.888	0.891	
5a	Train set	0.850	0.888	0.869	
	Test set	0.858	0.892	0.875	
6a	Train set	0.870	0.897	0.883	
	Test set	**0.881**	**0.903**	**0.892**	
1b	Train set	0.823	0.856	0.840	No
	Test set	0.825	0.855	0.840	
2b	Train set	0.834	0.874	0.853	
	Test set	0.845	0.878	0.861	
3b	Train set	0.883	0.897	0.890	
	Test set	0.884	0.898	0.891	
4b	Train set	0.874	0.862	0.868	
	Test set	0.870	0.852	0.861	
5b	Train set	0.883	0.874	0.879	
	Test set	0.875	0.863	0.869	
6b	Train set	0.869	0.861	0.865	
	Test set	0.871	0.868	0.870	

Table 7. Performance comparison with other existing work in medical NER

Model	Dataset	Precision	Recall
BiLSTM-CRF [14]	Twitter Data	93.22%	76.25%
RNN + Word Embedding [15]	i2b2 2010	-	-
Ensemble Method in “relaxed” criteria [7]	CCKS-2018 CNER	95.47%	94.92%
Machine learning techniques based on conditional random fields (CRFs) [13]	CHEMDNER for CEM (Chemical Entity mention Recognition)	89.09%	87.39%
Our Hybrid CNN-BiLSTM Model (Model 6a)	**Set of drug description sentences in Bahasa Indonesia**	**88.1%**	**90.3%**

3.4. Implementation result

As shown by Table 6, model 6a that uses a chunk tag, 2 layers of CNN, and a single layer LSTM obtained the highest F1-score and recall on the test set. Therefore, we choose this model to be implemented as a proof of concept. Inferencing the drug NER model in the real world has been done through our Drug NER app shown in Fig. 5.



Drug Named Entity Recognition

Your Text:

Imboost mengandung Echinacea Purpurea 250 miligram dan Zinc Picolinate 10 miligram

	Tokens	Entity Tags
0	Imboost	B-MED
1	mengandung	O
2	Echinacea	B-INGRD
3	Purpurea	I-INGRD
4	250	B-DOSE
5	miligram	I-DOSE
6	dan	O
7	Zinc	B-INGRD
8	Picolinate	I-INGRD
9	10	B-DOSE
10	miligram	I-DOSE

Fig. 5. CNN-BiLSTM application implementation

Drug NER was tested with different sentence' structures given in Table 8. There are four different structures of sentences written in Bahasa Indonesia that are taken as an inferencing test. The test shows the model generalization performance to do the drug NER task. Sentence #1 is an example of a simple active sentence structure: "Vilapon mengandung metoclopramide HCL untuk mengobati gangguan saluran pencernaan" (Vilapon contains metoclopramide HCL to treat digestive system disorders). Sentence #2 is an active sentence structure where a medicine has and information of multiple ingredients with their dose: "Hufagrip mengandung Paracetamol 500 mg, Ephedrin HCL 5 mg, Cholpeniramin Maleat 2 mg, Glyceril Guaiacolat 50 mg." (Hufagrip contains Paracetamol 500 mg, Ephedrin HCL 5 mg, Cholpeniramin Maleate 2 mg, Glyceril Guaiacolat 50 mg.). Sentence #3 is a passive sentence structure where a medicine with its ingredient is written closely in the sentence: "Furosemid yang dikandung dalam Edemin berguna untuk mengatasi edema…" (Furosemide contained in Edemin is useful for treating edema ...). Sentence #4 is another form of passive sentence structure: "Ethambutol yang terkandung dalam Kalbutol diindikasikan untuk mengobati penyakit tuberkulosis sistemik." (Ethambutol contained in Kalbutol is indicated to treat systemic tuberculosis.). In all four sentences, the drug NER model performs really well and can detect drug-related named-entity correctly.

Table 8. Different NER inferencing examples.

#	Text	NER
1	Vilapon mengandung metoclopramide HCL untuk mengobati gangguan saluran pencernaan.	[('Vilapon', 'B-MED'), ('mengandung', 'O'), ('metoclopramide', 'B-INGRD'), ('HCL', 'I-INGRD'), ('untuk', 'O'), ('mengobati', 'O'), ('gangguan', 'O'), ('saluran', 'O'), ('pencernaan', 'O'), ('.', 'O')]
2	Hufagrip mengandung Paracetamol 500 mg, Ephedrin HCL 5 mg, Cholpeniramin Maleat 2 mg, Glyceril Guaiacolat 50 mg.	[('Hufagrip', 'B-MED'), ('mengandung', 'O'), ('Paracetamol', 'B-INGRD'), ('500', 'B-DOSE'), ('mg', 'I-DOSE'), (',', 'O'), ('Ephedrin', 'B-INGRD'), ('HCL', 'I-INGRD'), ('5', 'B-DOSE'), ('mg', 'I-DOSE'), (',', 'O'), ('Cholpeniramin', 'B-INGRD'), ('Maleat', 'I-INGRD'), ('2', 'B-DOSE'), ('mg', 'I-DOSE'), (',', 'O'), ('Glyceril', 'B-INGRD'), ('Guaiacolat', 'I-INGRD'), ('50', 'B-DOSE'), ('mg', 'I-DOSE'), ('.', 'O')]
3	Furosemid yang dikandung dalam Edemin berguna untuk mengatasi edema…	[('Furosemid', 'B-INGRD'), ('yang', 'O'), ('dikandung', 'O'), ('dalam', 'O'), ('Edemin', 'B-MED'), ('berguna', 'O'), ('untuk', 'O'), ('mengatasi', 'O'), ('edema', 'O'), …]
4	Ethambutol yang terkandung dalam Kalbutol diindikasikan untuk mengobati penyakit tuberkulosis sistemik.	[('Ethambutol', 'B-INGRD'), ('yang', 'O'), ('terkandung', 'O'), ('dalam', 'O'), ('Kalbutol', 'B-MED'), ('diindikasikan', 'O'), ('untuk', 'O'), ('mengobati', 'O'), ('penyakit', 'O'), ('tuberkulosis', 'O'), ('sistemik', 'O'), ('.', 'O')]

4. Conclusion

In this work, an automatic name entity recognition for the drug has been successfully conducted. We focused on extracting three different entities, including (MED) to indicate the name of the medicine, (INGRD) as the ingredient of the medicine, and (DOSE) as a specific amount or weight of medication. Our main contribution is presenting the model and its evaluation using CNN-BiLSTM architecture on Indonesian language datasets. We used a deep learning architecture using a hybrid Convolutional Neural Network - Bidirectional Long-Short Term Memory (CNN-BiLSTM) model that automatically detects word- and character-level features to identify the said three name entities. We trained the architecture with six different hyper-parameter sets to find the best model. Based on the experiments, the best achievement was obtained by model 6a with f1-score of 0.892. In model 6a, we applied 2 layers of CNN with kernel size of 7, CNN filter of 50 and a single LSTM layer with 200 hidden units. In addition, we also found that applying a chunk tag could improve the model performance.

This research mainly focused on synthetic drugs which have the ingredients and dose. For future work, though this drug NER only focuses on synthetic drugs, it is also possible to extend this research and cover extracting important entities on herbal drugs. To improve the model, generating high quality tagged Indonesian medicinal data is critical. Defining rules cannot be too specific or too general. They will affect the model’s precision and recall. It is also possible to add a few more categories to classify extracted entities such as adverse drug reactions and drug side effects.

References

1. Grishman, R.; and Sundheim, B. (1996). Message understanding conference-6: A brief history. *Proceedings of the* 16*th Conference on Computational Linguistics*, Volume 1. Pennsylvania, USA, 466-471.
2. Zhao, W.; Gao, L.; and Liu, A. (2018). Programming foundations for scientific big data analytics. *Scientific Programming*, Special Issue, Volume 2018, Article ID 2707604.
3. Chiu, J.P.C.; and Nichols, E. (2016). Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, 4: 357-370.
4. Lample, G.; Ballesteros, M.; Subramanian, S.; Kawakami, K.; and Dyer, C. (2016). Neural architectures for named entity recognition. *Proceedings of the* 2016 *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, San Diego, California, 260-270.
5. Huang, H.; Wang, H.; and Jin, D. (2018). A low-cost named entity recognition research based on active learning. *Scientific Programming*, Special Issue, Volume 2018, Article ID 1890683.
6. Sang, E.F.T.K.; and De Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-Independent named entity recognition. *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL* 2003, 142-147.
7. Luo, L.; Li, N.; Li, S.; Yang, Z.; and Lin, H. (2018). DUTIR at the CCKS-2018 Task1: A neural network ensemble approach for Chinese clinical named entity recognition. *Proceedings of the Evaluation Tasks at the China Conference on Knowledge Graph and Semantic Computing* (*CCKS-Tasks* 2018), Tianjin, China, 7-12.
8. Luo, Y.; Xiao, F.; and Zhao, H. (2020). Hierarchical contextualized representation for named entity recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(5), 8441-8448.
9. Baevski, A.; Edunov, S.; Liu, Y.; Zettlemoyer, L.; and Auli, M. (2019). Cloze-driven pretraining of self-attention networks. *Proceedings of the* 2019 *Conference on Empirical Methods in Natural Language Processing and the* 9*th International Joint Conference on Natural Language Processing* (*EMNLP-IJCNLP*), Hong Kong, China, 5360-5369.
10. Jiang, Y.; Hu, C.; Xiao, T.; Zhang, C.; and Zhu, J. (2019). Improved differentiable architecture search for language modeling and named entity recognition. *Proceedings of the* 2019 *Conference on Empirical Methods in Natural Language Processing and the* 9*th International Joint Conference on Natural Language Processing* (*EMNLP-IJCNLP*), Hong Kong, China, 3585-3590.
11. Liu, Y.; Meng, F.; Zhang, J.; Xu, J.; Chen, Y.; and Zhou, J. (2019). GCDT: A global context enhanced deep transition architecture for sequence labeling. *Proceedings of the* 57*th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, 2431-2441.
12. Liu, S.; Tang, B.; Chen, Q.; and Wang, X. (2015). Drug name recognition: Approaches and resources. *Information*, 6(4), 790-810.

13. Krallinger, M.; Leitner, F.; Rabal, O.; Vazquez, M.; Oyarzabal, J.; and Valencia, A. (2015). CHEMDNER: The drugs and chemical names extraction challenge. *Journal of Cheminformatics*, 7, Article number: S1.
14. Batbaatar, E.; and Ryu, K.H. (2019). Ontology-based healthcare named entity recognition from twitter messages using a recurrent neural network approach. *International Journal of Environmental Research and Public Health*, 16(19), 3628.
15. Wu, Y.; Jiang, M.; Xu, J.; Zhi, D.; and Xu, H. (2017). Clinical named entity recognition using deep learning models. *AMIA Annual Symposium Proceedings*, 2017, 1812-1819.
16. Yepes, A.J.; and MacKinlay, A. (2016). NER for medical entities in Twitter using sequence to sequence neural networks. *Proceedings of the Australasian Language Technology Association Workshop* 2016, 138-142.
17. Hidayatullah, A.F.; and Ma'arif, M.R. (2017). Pre-processing tasks in Indonesian Twitter messages. *Journal of Physics: Conference Series*, 801(1), 012072.
18. Kumawat, D.; and Jain, V. (2015). POS tagging approaches: A comparison. *International Journal of Computer Applications*, 118(6), 32-38.
19. Juwantara, Z.; Eddy, F.; Sasmita, D.H.; Khoolish, T.N.; and Aldiyansyah, B. (2021). Kumparan NLP Library. Retreived 21 June, 2021 from https://zenodo.org/record/4556870#.Yf8tUupBzIU
20. Eftimov, T.; Koroušić Seljak, B.; and Korošec, P. (2017). A rule-based named-entity recognition method for knowledge extraction of evidence-based dietary recommendations. *PloS one*, 12(6), e0179488.
21. Santos, C.N.D.; and Guimaraes, V. (2015). Boosting named entity recognition with neural character embeddings. *Proceedings of the Fifth Named Entity Workshop, joint with* 53*rd ACL and the* 7*th IJCNLP*, Beijing, China, 25-33.
22. Collobert, R.; and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. *Proceedings of ICML* '08*: Proceedings of the* 25*th International Conference on Machine Learning*, 160-167.
23. Gholamy, A.; Kreinovich, V.; and Kosheleva, O. (2018). Why 70 / 30 or 80 / 20 relation between training and testing sets : A pedagogical explanation. *Departmental Technical Reports* (*CS*), 1209.
24. Khomenko, V.; Shyshkov, O.; Radyvonenko, O.; and Bokhan, K. (2016). Accelerating recurrent neural network training using sequence bucketing and multi-GPU data parallelization. 2016 *IEEE First International Conference on Data Stream Mining & Processing* (*DSMP*), 100-103.
25. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1), 1929-1958.
26. Zhang, Y.; and Wallace, B.C. (2015). A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *Proceedings of the The* 8*th International Joint Conference on Natural Language Processing*, Taipei, Taiwan, 253-263.