Building an Indonesian Rule-Based Part-of-Speech Tagger

Fam Rashel, Andry Luthfi, Arawinda Dinakaramani, and Ruli Manurung
Faculty of Computer Science
Universitas Indonesia
Depok, Indonesia
fam.rashel@ui.ac.id, andry.luthfi@ui.ac.id, ard51@ui.ac.id, maruli@cs.ui.ac.id

Abstract—**This paper describes work on a part-of-speech tagger for the Indonesian language by employing a rule-based approach. The system tokenizes documents while also considering multi-word expressions and recognizes named entities. It then applies tags to every token, starting from closed-class words to open-class words and disambiguates the tags based on a set of manually defined rules. The system currently obtains an accuracy of 79% on a manually tagged corpus of roughly 250.000 tokens.**

Keywords-part of speech tag; disambiguation rule; token

I. Introduction

Part-of-speech tag is a grammatical category for words, such as noun, verb, adjective, etc. Part-of-speech tagging is a process of assigning those part-of-speech tags to words in a text. Doing part-of-speech tagging manually would be a time-consuming duty. That is why automating the process is such a fundamental process in natural language processing.

Our part-of-speech tagger employs a rule-based approach. The system utilizes several language resources to collect all possible tags for the respective tokens. A set of disambiguation rules is then applied to decide the appropriate part-of-speech tag for a token.

In the *Background* section below, relevant works will be discussed. The probabilistic part-of-speech tagging described in [1] and the morphological analyzer described in [2] are several notable works. The *Language Resources* section describes the various language resources required by the system. Our system utilizes a dictionary for *Bahasa Indonesia*, the Indonesian language, as the main linguistic resource. From this Bahasa Indonesia Dictionary we built two purpose-specific dictionaries, a closed-class tagging dictionary and a multi-word expression dictionary. We also constructed a list of disambiguation rules which help the system to disambiguate ambiguous tokens. MorphInd [2] is a morphological analyzer employed by the system to annotate specific groups of token. In the *Rule-Based Tagging* section, we present the workflow of the system. Our system tokenizes the documents by considering the multi-word expressions. The closed-class words are tagged using the closed-class tagging dictionary while MorphInd analyzes the open-class words after securing the proper nouns using a named entity recognizer. Lastly, the system disambiguates every ambiguous token by executing all appropriate disambiguation rules. The *Experiments and Evaluation* section provides the result from several experiments conducted for the system.

II. Background

Cutting et al. define a rule-based part-of-speech tagger as a part-of-speech tagger that assigns tags to a token based on several manually created linguistic rules [3]. Besides that, there is probabilistic approach and transformational-based tagger. One of the most well-known taggers is the Brill tagger [4]. It has been adapted into several languages such as German, Hungarian, Swedish, etc.

Although there have been many studies on automation for part-of-speech tagging in languages such as English, there is still relatively little work for Bahasa Indonesia. Mistica et al. showed promising evidence that Indonesian word class labeling may be achieved through rules [5]. We attempt to extend this hypothesis by building a part-of-speech tagger for Bahasa Indonesia that employs a purely rule-based approach. Such an approach yields a model that is more linguistically principled and can be studied.

III. Language Resources

The system we built requires three language resources, a Bahasa Indonesia dictionary, morphological analyzer, and disambiguation rules. These resources provide important information for the system to decide accurate part-of-speech tag. Hereby is the explanation for each of them.

A. *Bahasa Indonesia Dictionary*

Bahasa Indonesia dictionary is our main language resource. We use *Kamus Besar Bahasa Indonesia* (KBBI) version 3 to extract the required information. From KBBI we managed to build closed-class tagging dictionary and multi-word expressions dictionary.

The **closed-class tagging dictionary** is a dictionary containing all tokens known as closed-class words. Every token in this dictionary is deemed to have an unambiguous tag. This dictionary is formatted as a tab separated file, for every token has its corresponding part-of-speech tag separated by tab. Figure 1 shows us a portion of the closed-class tagging dictionary.

dia	PRP
belum	NEG
atau	CC

Figure 1. Closed-class Tagging Dictionary

As can be seen in Figure 1, every line consists of a token and its corresponding part-of-speech tag. For example, the word "*dia*" has part-of-speech tag as personal pronoun (PRP), the word "*belum*" has part-of-speech tag as negation (NEG), and the word "*atau*" has part-of-speech tag as coordinating conjunction (CC).

The **multi-word expression dictionary** is a dictionary containing all Bahasa Indonesia multi-word expressions. Every token constructed from two or more words would be considered to be included in this dictionary. The system uses this multi-word expressions dictionary to recognize multi-word expressions when tokenizing the document. An excerpt of this dictionary is shown in Figure 2.

"atas nama"	noun
"balas dendam"	verb
"haru biru"	noun

Figure 2. Multi-word Expressions Dictionary

The word "*atas nama*" has part-of-speech tag "*noun*". The same principle applied for word "*balas dendam*" which has part-of-speech tag "*verb*", and the word "*haru biru*" has part-of-speech tag "*noun*".

B. *Morphological Analyzer*

For the open-class words, we use morphological analyzer to annotate the tag for each token. The system employs MorphInd to perform this task [2]. The system filters the outputs from MorphInd for only noun, verb and adjective part-of-speech tag. This means that MorphInd should only handle open-class words.

C. *Disambiguation Rules*

It is possible for a token to have more than one tag. It means that the token is ambiguous. The system provides a disambiguation feature in order to provide the suitable tag. To do this, the system requires a language resource containing disambiguation rules. This language resource contains rules of how to disambiguate a token by lookup for the neighboring token's tag. Figure 3 shows an example of a disambiguation rule.

```
<rule id="rule-4" tags="NN/NND">
   <premise grammar="-1:CD and +1:NN" output="NND"/>
   <premise grammar="-2:NN and -1:CD" output="NND"/>
   <premise output="NN"/>
</rule>
```

Figure 3. Example of disambiguation rule

Figure 3 demonstrates a disambiguation rule for NN and NND tags. This rule has id "*rule-4*" and three premises. The first premise states that if the previous token has "*CD*" tag and the next one has "*NND*" tag then the part-of-speech tag for the current tag is "*NND*". The second premise implies that if the previous token has "*CD*" tag and the second previous has "*NN*" tag then the part-of-speech tag for the current tag is also "*NND*". The last premise states that the current tag must be "*NN*". This premise is the base-case. The system would trace the premise with top-down approach. If the pattern of any premise matches the current context the output tag is issued.

IV. Rule-Based Tagging

Our POS Tagger adapts a rule-based approach. The system tokenizes documents while also considering multi-word expressions and recognizes name entities. It then gives tag for every token, starting from closed-class words to open-class words. Every ambiguous token will be disambiguated based on one or more defined rules. Overall, the system is implemented in five stages: MWE tokenization, Name Entity Recognizer, closed-class words & open-class words tagging, rule-based disambiguation, and resolver. Figure 4 shows us the overall process of how the system works.

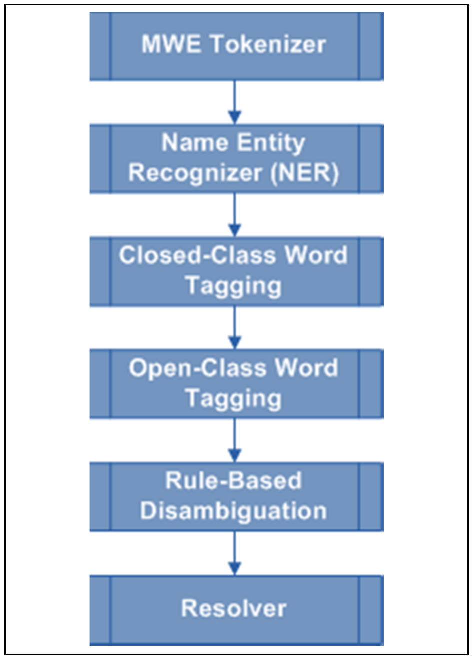

MWE Tokenizer
Name Entity Recognizer (NER)
Closed-Class Word Tagging
Open-Class Word Tagging
Rule-Based Disambiguation
Resolver

Figure 4. Rule-Based POS Tagger Overview

A. *MWE Tokenizer*

During this stage, tokenization is performed against the input document. The document is split using whitespaces then each token is considered as a multi-word expression (MWE) before finalizing the token. The system also annotates the part-of-speech tag for every token that is recognized as a multi-word expression.

Our system splits the document based on whitespaces but also considers special characters such as punctuation. We take several punctuations such as fullstops, exclamation marks, and question marks for sentences delimiters. Aside from the characters above, new lines are also considered as sentence delimiters.

First, the tokenizer builds a prefix tree from the multi-word expression dictionary. The prefix tree starts from the root. The root then has several children which is the list of first words from the phrase. It is then expanded with the second possible word from the phrase, and so on.

The system scans every token one by one and consults with the multi-word expressions prefix tree whether it is expandable or not. If it is expandable, then the system would consult the next token if it is the suitable word to form the phrase, and so on until the terminal word found or a non-suitable word is found at the next token. If the terminal word is found the system would pack those tokens into one token and give the corresponding part-of-speech tag. Otherwise, if non-suitable word is found at the next token, the system would leave them as separated tokens.

It is possible to have an overlapping multi-word expression which means a multi-word expression could be formed if we append a suitable word to an existing multi-word expression. For example, consider the phrase "*rumah sakit*" which means hospital. The phrase "*rumah sakit*" formed from "*rumah*" which means house and "*sakit*" which means sick. This phrase already has a part-of-speech tag as noun but if we put the word "*jiwa*", which means soul, the phrase would be "*rumah sakit jiwa*" which means asylum or mental hospital. This is still a valid multi-word expression. In this case we found that "*rumah sakit*" and "*rumah sakit jiwa*" are overlapping multi-word expression. To handle this case our multi-word expressions tokenizer employs greedy algorithm. The system would keep the latest best multi-word expression candidate. So when the system found "*rumah*" followed by "*sakit*", the system updates the current best candidate as "*rumah sakit*". If "*jiwa*" is found as the next token, then the best candidate would be updated to "*rumah sakit jiwa*". If a non-suitable word is found, the the system would give the latest best candidate, "*rumah sakit*", as a multi-word expression.

The system formats the document as a token and corresponding part-of-speech tag per line. The token and corresponding tag is separated by a tab.

B. *Name Entity Recognizer*

After the document is tokenized, the system continues to the next stage which is recognizing named entities as proper nouns. The system employs Name Entity Recognizer as a reliable information source to do this task. We put this stage first to ensure the name entity would not be mistaken to be a closed-class or open-class word.

The system simply passes the document to the name entity recognizer. The name entity recognizer would scan every token and detect any possible name entity. Every possible name entity is then tagged as proper noun. The format of the output document is also formatted one token per line with the corresponding tag separated by tab.

C. *Closed-Class Words & Open-Class Words Tagging*

After securing the multi-word expressions and named entities the system proceeds to closed-class words and open-class words tagging. The system annotates the closed-class words first, then open-class words. The system employs closed-class tagging dictionary to annotate closed-class words and MorphInd for open-class words.

For the **closed-class words tagging**, the system scans through the document and iterates every token in it. The system then consults the token with the closed-class tagging dictionary. If any suitable token is found, then the corresponding part-of-speech tag would be added to the token as the output.

The second is **open-class words tagging**. For this stage, the system applies the same methodology as Name Entity Recognizer. The system passes the document to MorphInd to analyze every token and get the corresponding part-of-speech tag for every token. As mentioned above, the system filters the outputs from MorphInd for only noun, verb and adjective part-of-speech tags. Figure 5 shows us an example of open-class words tagging result.

muncul	VB
monyet	NN
antikorupsi	JJ,NN
prasyarat	JJ,NN

Figure 5. Open-class words tagging result

From Figure 5 we can understand that MorphInd identifies "*muncul*" as verb (VB) and "*monyet*" as noun (NN). Different from those tokens above, "*antikorupsi*" and "*prasyarat*" were identified by MorphInd as noun but also adjective. This means that these tokens are ambiguous. MorphInd actually provides disambiguation features. This feature forced MorphInd to give only one part-of-speech tag for every token.

D. *Rule-Based Disambiguation*

During this stage, every token could contain one or more part-of-speech tag given by the previous stage. Although MorphInd provide disambiguation feature in it, we could still have ambiguous token from the other stage. This is why the system still needs to have a disambiguation process in its workflow.

If a token contains more than one part-of-speech tag, the system will try to disambiguate the token by employing a rule-based approach. The system would search through the disambiguation rules to find the suitable rule for the ambiguous token. After the suitable rule is found, the system would analyze the premises by performing lookup to the neighboring token's tag before issuing the tag.

As the disambiguation process could never found any suitable premise for the ambiguous token, the system would leave the token ambiguous as it is. The system confident that token has the right part-of-speech tag between those possible tag, but doesn't has any clue which one is the right tag.

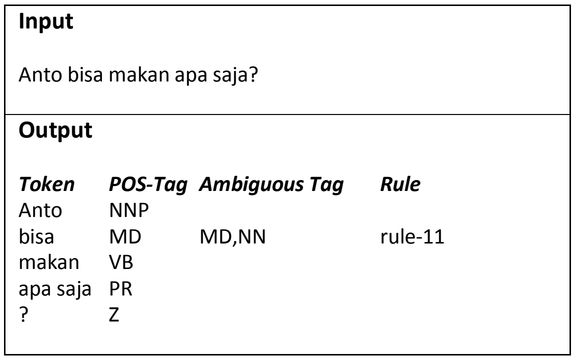
Jenis	NN	NND,NN	rule-4
bisa	MD	MD,NN	rule-11
untuk	SC,IN	SC,IN	no rule matched

Figure 6. Rule-based disambiguation result

As can be seen from Figure 6, "*Jenis*" has part-of-speech tag as noun (NN) from the ambiguous tag noun (NN) and classifier (NND) by using rule number four in the list of disambiguation rule. The same applied for "*bisa*" has part-of-speech tag as modal (MD) from the ambiguous tag modal (MD) and noun (NN) by using rule number eleven in the list of disambiguation rule. The last example shows the output of the system when there is not any satisfying rule and premise for the ambiguous token. There is a possibility that the rule match but no premises could satisfy the context. For those cases, the system would leave the ambiguous token as it is.

E. *Resolver*

This is the last stage of the tagging process. Up to this point, every token should have their own tag whether it is ambiguous or not. Shall a token still does not have a token, the system would give a special "X" tag for the respective token as a meaning of unknown token. This special "X" tag indicates that the system does not know the right part-of-speech tag for that token. We believe that better for the system to tell that it does not know rather than giving unreliable answer.



Input

Anto bisa makan apa saja?

Output

Token	*POS-Tag*	*Ambiguous Tag*	*Rule*
Anto	NNP		
bisa	MD	MD,NN	rule-11
makan	VB		
apa saja	PR		
?	Z		

Figure 7. Example of input and final output of the system

The example of input and final output of the system can be seen in Figure 7. Given the input string as the first row, "*Anto bisa makan apa saja?*". The system gave the result as the second row. The first column is the token, followed by the corresponding part-of-speech tag. The two next columns is the ambiguous tag and disambiguation rule applied to the token if the token is ambiguous.

V. Experiments and Evaluation

We conducted experiments using the system against Indonesian IDENTIC corpus. We have manually tagged the first 10.000 Indonesian sentences of this corpus and use this as gold standard later on. Different scenarios were conducted in our experiments which help us to tune up the system's performance.

The experiments conducted by using two scenarios. The first one is without Named Entity Recognizer and Multi-word Expressions, and the second is without Multi-word Expressions. Table 1 shows us the result we got from the experiment conducted by the system. We can easily concluded that the accuracy of the system constantly grow while using more feature. There were less false tags when the system uses MWE and lesser when it combines the MWE into consideration.

TABLE I. Experiments results

	Wo/ NER & MWE	**Wo/ MWE**
TRUE	182506	205715
False	78464	55255
Token Total	260970	260970

Figure 8 shows us the differences of the accuracy while using different configuration on the system. When the system depends only with the dictionary, the system obtains a 70% accuracy. When the system uses NER to identify the proper noun, the accuracy rose into 79%.

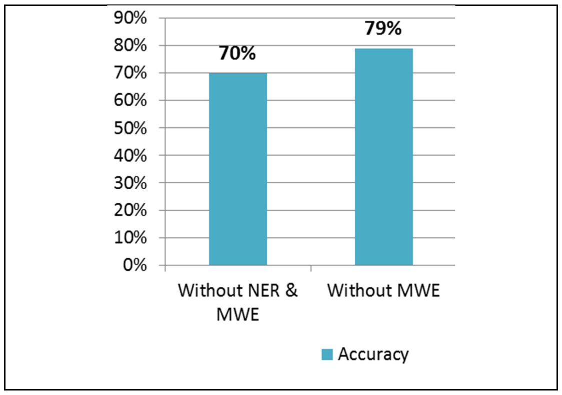


Figure 8. Experiments result using three scenarios

VI. Conclusion and Future Work

We have developed an Indonesian part-of-speech by employing a rule-based approach. The system combines several language resources such as closed-class tagging dictionary, multi-word expressions dictionary, MorphInd and disambiguation rules to provide the accurate part-of-speech tag.

We developed a web interface which provides the service for our part-of-speech tagger on http://bahasa.cs.ui.ac.id/pos-tagger/ as a demo for our system. Every user can write the input text and the system would provide the part-of-speech tag as the output. We also put on API for the web service so that everyone can easily access our part-of-speech tagger service by specific url and defined parameter.

Many aspects from the system are still rudimentary, and there are still many opportunity to improve the system, such as foreign language detector, expanding the language resources, and improving the tokenizer. A careful examination could help us, especially linguist to learn the Bahasa Indonesia structure and grammar as we build the system with rule-based approach. A better and complete disambiguation rule could be provided by doing this examination. For a far more advance improvements could be building a Treebank by using the outputs of the system.

References

[1] F. Pisceldo, M. Adriani, and R. Manurung, "Probabilistic Part-of-Speech Tagging for Bahasa Indonesia," in Proceedings of Third International Wokshop on Malay and Indonesian Language Engineering, Singapore, 2009.

[2] S. Dian Larasati, V. Kubon, and D. Zeman, "Indonesian Morphology Tool (MorphInd): Towards Indonesian Corpus," in Proceedings of the Workshop on Systems and Frameworks for Computational Morphology, Zurich, 2011.

[3] D. Cutting, J. Kupiec, J. Pederson, and P. Sibun, "A Practical Part of Speech Tagger," in Proceedings of the Third Conference on Applied Natural Language Processing, 1992.

[4] E. Brill, "A Simple Rule-Based Part of Speech Tagger," in Proceedings of the Third Conference on Applied Computational Linguistics (ACL), Trento, Italy, 1992.

[5] M. Mistica, T. Baldwin, and I W. Arka, "Word classes in Indonesian: A linguistic reality or a convenient fallacy in natural language processing?", in Proceedings of the 25th Pacific Asia Conference on Language, Information, and Computation (PACLIC 25), Singapore, December 2011.