DOI: 10.15849/IJASCA.211128.01

Int. J. Advance Soft Compu. Appl, Vol. 13, No. 3, November 2021
Print ISSN: 2710-1274, Online ISSN: 2074-8523

Embedding from Language Models (ELMos)-based Dependency Parser for Indonesian Language

Anthony, Alethea Suryadibrata, and Julio Christian Young

Dept. of Informatics, Faculty of Informatics and Engineering,
Universitas Multimedia Nusantara, Indonesia
e-mail: anthony4@student.umn.ac.id, alethea@umn.ac.id,
julio.christian@umn.ac.id

Abstract

The goal of dependency parsing is to seek a functional relationship among words. For instance, it tells the subject-object relation in a sentence. Parsing the Indonesian language requires information about the morphology of a word. Indonesian grammar relies heavily on affixation to combine root words with affixes to form another word. Thus, morphology information should be incorporated. Fortunately, it can be encoded implicitly by word representation. Embeddings from Language Models (ELMo) is a word representation which be able to capture morphology information. Unlike most widely used word representations such as word2vec or Global Vectors (GloVe), ELMo utilizes a Convolutional Neural Network (CNN) over characters. With it, the affixation process could ideally encoded in a word representation. We did an analysis using nearest neighbor words and T-distributed Stochastic Neighbor Embedding (t-SNE) word visualization to compare word2vec and ELMo. Our result showed that ELMo representation is richer in encoding the morphology information than it's counterpart. We trained our parser using word2vec and ELMo. To no surprise, the parser which uses ELMo gets a higher accuracy than word2vec. We obtain Unlabeled Attachment Score (UAS) at 83.08 for ELMo and 81.35 for word2vec. Hence, we confirmed that morphology information is necessary, especially in a morphologically rich language like Indonesian.

Keywords: *ELMo, Dependency Parser, Natural Language Processing, word2vec*

1 Introduction

Indonesian language is morphologically rich. Word often undergoes many morphology processes such as affixation, reduplication, and compounding [1]. We need to understand word morphology so Indonesian sentences can be better analyzed [2,3,4,5]. That is also true in case of dependency parsing. The goal of dependency parsing is to tell the syntactic relation between words (e.g., subject and object relationship). With a rapid advancement of machine learning, many researchers began to adapt it to solve various Natural Language Processing (NLP) problems such as text classification [6] and word semantics visualization [7].

The parser developed in this study uses a neural network. This kind of parser needs a word vector as its input. We can encode a semantic and syntactic meaning of a word by using word vector. Furthermore, the parser developed in this study uses Embedding from Language Model (ELMo) [8] to generate word vector. ELMo's word vector can encode morphological knowledge (i.e., word shape). That is, words with the same morphemes could be related. For example, consider word *pelajaran* and word *belajar*. They both share the same morpheme that is word *ajar*. ELMo will produce similar vectors for those two words. Moreover, ELMo uses a Convolutional Neural Network (CNN) to model the morphology of a word. CNN works by taking a sequence of characters as inputs, allowing it to compose word representation from local features produced around each character of the word. Hence, ELMo is a character-level word representation [9]. Parser's performance is measured by Unlabeled Attachment Score (UAS) and Labeled Attachment Score (LAS). The result from this study shows that parser got a higher UAS and LAS when using ELMo, indicating that morphological information has a positive impact on the parser performance.

The rest of this paper is structured as follows: In the “Related Work” section, we discussed related research about dependency parser. “Literature Study” section contains theoretical background of the research. In the “Parsing Model” section, we introduce the architecture of our parser. “Experiments” section presents the result of our research. The “Discussions” section discusses the results from previous section. The final section presents the conclusions of our research.

2 Related Work

The previous study has used a neural network to do dependency parsing. Parser developed by Chen and Manning uses word2vec as its word representations [10]. Their parser got high accuracy and speed when parsing English and Chinese sentences. Yet, their parser lacks of information about the word morphology. Ballesteros, et al. improved the parser by replacing lookup-based word representations with representations constructed from the orthographic representations of the words using Long Short-Term Memory (LSTM) [11]. Their study has shown substantial improvement in parsing morphologically rich language like Indonesian. Kim, et al. also used morphologically aware word representation for their language model [9]. In contrast with Ballesteros, et al., Kim, et al. used CNN instead of LSTM to compute word representation and has shown that CNN can model word morphology as well.

In this work, we try to develop a parser similar to that in Chen and Manning's architecture. What sets apart our approach to theirs is we use character CNN from ELMo, making our parser aware of the word morphology. We use a pre-trained ELMo word vector so that we can shorten the training time thanks to the transfer-learning technique.

3 Literature Study

3.1 Transition-based Dependency Parsing

We use a transition-based parsing method. The parsing begins by processing one word at a time. Then the parser decides to either join this word to a word encountered previously or to store this word until it can attach with another word at a later point in the sentence. The transition system has a configuration. A configuration consists of a buffer to hold all the input words, a stack to hold the partial parse tree, and an arcs to keep the list of all dependencies produced throughout the parse [12]. The transition system must reach a final configuration to parse the sentence fully by running a series of transitions. Three transitions are possible to run. These are: Shift, Left-arc, and Right-arc. The Shift transition moves the first word of the buffer to the stack. The Left-arc transition adds an arc between the top two items on the stack, with the first top item being the head and the second item being the dependent. The Right-arc adds an arc between the top two items on the stack, with the first top item being the dependent and the second item being the head.

3.2 ELMo

Pre-trained word vectors are a necessary feature of many natural language processing tasks, especially those that use neural networks. A model generates word vectors by studying words from the corpus. So when the model finishes the training, word vectors will keep the meaning of the word — both syntactically and semantically. ELMo is a language model used as word embedding [8], and it has components that can model the internal structure of a word within its word vector. To understand the word morphology, ELMo uses CNN to learn subword information of a word by composting word representation from character sequences. Figure 1 depicted a process of how CNN produces character-level embedding. Suppose a word k is made up of character $[c_1, c_2, \ldots, c_M]$ where M is the length of work k . Each character is transformed into a character embedding

$\left[r_1^{chr}, r_2^{chr}, \dots, r_M^{chr}\right]$. Then the character-level representation of k is given by the matrix $C^k \in \mathbb{R}^{d \times M}$ where d is the dimensionality of character embeddings. We apply a convolution between C^k and a filter $H \in \mathbb{R}^{d \times w}$ of width w to search for a linguistic pattern (e.g., affix). After which we obtain a feature map $f^k \in \mathbb{R}^{M-w+1}$. Finally, we use max over all character windows of the word to capture the most important feature. The result is $r^{wch} = \max_i f^k[i]$ a character-level embedding of the word which stores not only syntactic and semantic meaning of a word but also its word shape information.

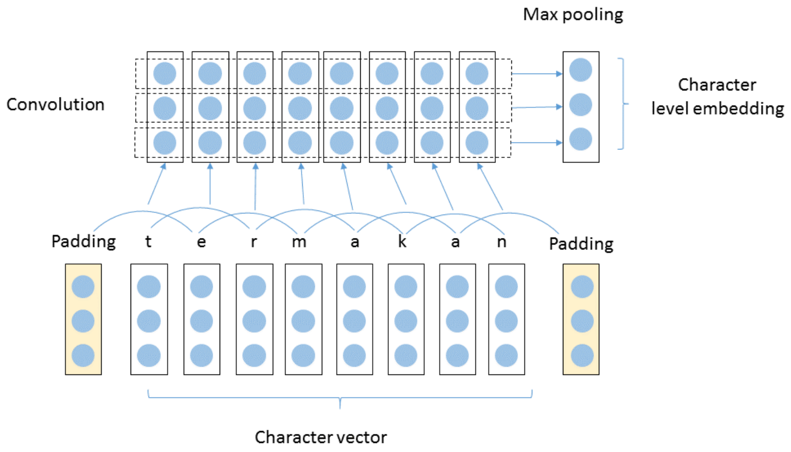

Fig. 1 The convolutional neural network to extract character-level features

4 Parsing Model

The parser developed in this research follows the parser that was developed by Chen and Manning [10]. We first define a feature function to extract features from the parser configuration. The features used in our implementation are shown in Table 1.

Table 1: List of Features Extracted from a Parser Configuration

Source	Features
Stack	s1, s2, s3
Buffer	b1, b2, b3
Arc	lc1(s1), rc1(s1), lc2(s1), rc2(s1), lc1(s2), rc1(s2), lc2(s2), rc2(s2), lc1(lc1(s1)), rc1(rc1(s1)), lc1(lc1(s2)),

rc1(rc1(s2))

As shown in Table 1. There are top 3 words on the stack and buffer, the first and second leftmost/rightmost children of the top two words on the stack, the leftmost of leftmost/rightmost of rightmost children of the top two words on the stack. Those features then convert into a vector and concatenate together, forming a long vector $x = [e^w; e^p; e^l]$. The concatenation of those feature vector is used as an input layer. The input layer then maps to a hidden layer, applying a linear transformation and a non-linear activation function as follows: $h = ReLu(W_1 x + b)$. A softmax layer is finally added on the top of the hidden layer for modeling multi-class probabilities $p = softmax(W_2 h)$. Figure 2 describes our neural network architecture.

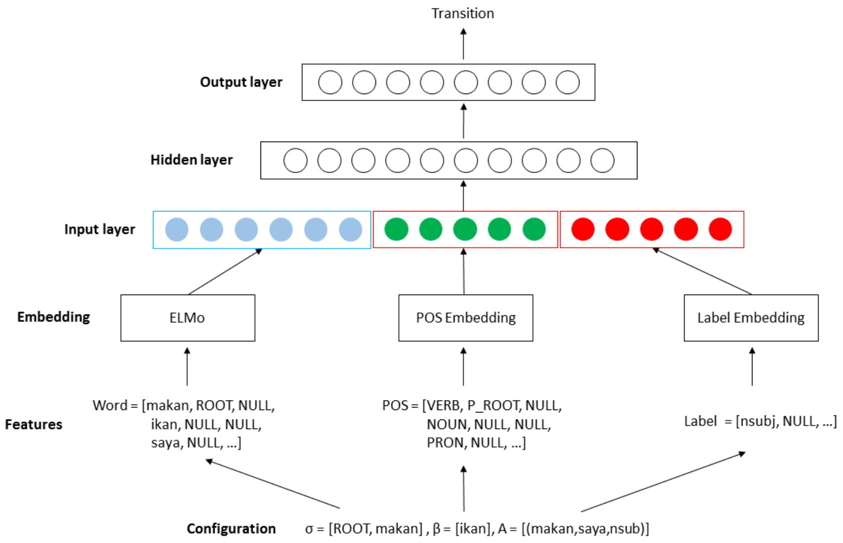
Transition
Output layer
Hidden layer
Input layer
Embedding
ELMo
POS Embedding
Label Embedding
Features
Word = [makan, ROOT, NULL, ikan, NULL, NULL, saya, NULL, ...]
POS = [VERB, P_ROOT, NULL, NOUN, NULL, NULL, PRON, NULL, ...]
Label = [nsubj, NULL, ...]
Configuration
σ = [ROOT, makan] , β = [ikan], A = [(makan,saya,nsubj)]

Fig. 2 Architecture of our parser

5 Experiments

5.1 Data

We used the Indonesian Google Stanford Dependency (GSD) treebank [13] to train our parser. This treebank comes with sentences that already annotated with their dependency relation. We follow the standard train/dev/test splits of Indonesian GSD.

5.2 Configuration

We conduct our experiment on two configurations. These are: the parsing model that uses ELMo word embedding and one that uses word2vec [14]. Word2vec is a standard word embedding that does not incorporate the character information, so it acts as a comparison of how morphological information affects parser's performance. The hyperparameter used for both parsers are explained in Table 2. We performed a hyper-parameter search using the Bayesian optimization to select those hyperparameters.

Table 2: Hyper-parameter Search Space and Final Values Used For All Experiments

Hyper-parameter	word2vec		ELMo	
	Range	Final	Range	Final
Epoch	-	100	-	100
Hidden size	-	4096	-	4096
Dropout	[0.6, 0.8]	0.69	[0.6, 0.8]	0.61
Batch size	[64, 1024]	242	[64, 1024]	244
Learning rate	[1e-5, 2e-3]	2.90e-4	[1e-5, 2e-3]	2.20e-4

5.3 Results

Table 3 shows the result of the ELMo parsing model and the word2vec parsing model. We observe that ELMo outperforms word2vec in both Unlabeled Attachment Score (UAS) and Labeled Attachment Score (LAS). We suspect that the morphological information of ELMo representation contributes to this result. Based on the score achieved by the ELMo-based dependency parser model on the experimentation part, it can be concluded that indeed this model can achieve a better performance for morphological rich language dependency parsing tasks.

Table 3: Accuracy of All Models

Source	UAS	LAS
word2vec	81.35	73.67
ELMo (CNN)	83.08	76.24

6 Discussions

Firstly, we investigate the morphological information within a word vector by computing the nearest neighbor words based on cosine similarity. Table 4 shows the nearest neighbors of word representation learned from both ELMo and word2vec.

Table 4: Nearest Neighbor Words for Words In the Vocabulary

	In Vocabulary					
	berpikir	tunanetra	bagaimanapun	tepercaya	saudari	cendekiawan
ELMo	berfikir	tunagrahita	Bagaimanapun	terstandar	saudara-saudari	cendikiawan
	Berpikir	tunarungu	menurutnya	teraplikasi	Saudari	dramawan
	berpikiran	autistik	Betapapun	terhangat	saudara/i	sejarawan
	Berfikir	tunalaras	betapapun	terpecaya	sodara	sastrawan
	berfikiran	autis	kenyataannya	berotoritas	saudara	pujangga
word2vec	berfikir	lansia	faktanya	terpercaya	sepupu	cendikiawan
	mengeluh	peretas	namun	referensinya	keponakan	akademisi
	membayangkan	penyandang	tetapi	akurat	tiri	pemikir
	menyadarkan	disabilitas	tampaknya	referensi	adik	pembaharu
	menyadari	pemula	tapi	informatif	saudara	teolog

We can see from the result that ELMo word representation seems to encode word shape similarity. For example, the nearest neighbors of *cendekiawan* are *dramawan*, *sejarahwan*, *sastrawan*, which have the same prefix *-an*. On the other hand, word2vec has words that are semantically related, some of them being synonyms. So, for example, the nearest neighbors of *cendekiawan* are *akedemisi*, *pemikir*, and *pembaharu*. In Table 5, we present out-of-vocabulary words and their respective nearest neighbors.

Table 5: Nearest Neighbor Words for Words Out of the Vocabulary

	Out of Vocabulary		
	S.Adm	ketidakbertanggung-jawabannya	sebisa-bisanya
ELMo	S.Sn	ketidakberdayaannya	sedapat-dapatnya
	S.IP	ketidaksetujuannya	sebernarnya
	S.Psi	ketidakpuasannya	seberat-beratnya
	S.PD	ketidakberuntungan	sedikit-dikitnya
	S.TP	ketidaknyamanannya	bekas-bekasnya

We only have a result for ELMo since word2vec does not have representation for words that do not appear in the vocabulary. For instance, the nearest neighbor of *ketidakbertanggungjawabannya* is *ketidakberdayaannya*, which shares the same affixes. Those affixes are prefix *tidak-*, confix *ke--an*, and suffix *-nya*. We can further investigate the morphological information by visualizing the word vector learned by ELMo and word2vec. Before visualizing the vectors, we must project the vector to a 2-dimensional space using t-SNE [15] so they can be plot to a scatter plot. We sample a total of 4.282 affixed words, consisting of suffixes: *meN-*, *ter-*, *ber-*; confixes: *se--nya*, *peN--an*, *ke--an*, *ber--an*; and suffixes: *-an*, *-pun*, *-i*. The visualization from Figure 3 shows that ELMo word vectors are grouped according to their affixes. Meanwhile, there are no clear clusters that exist in word2vec word visualization.

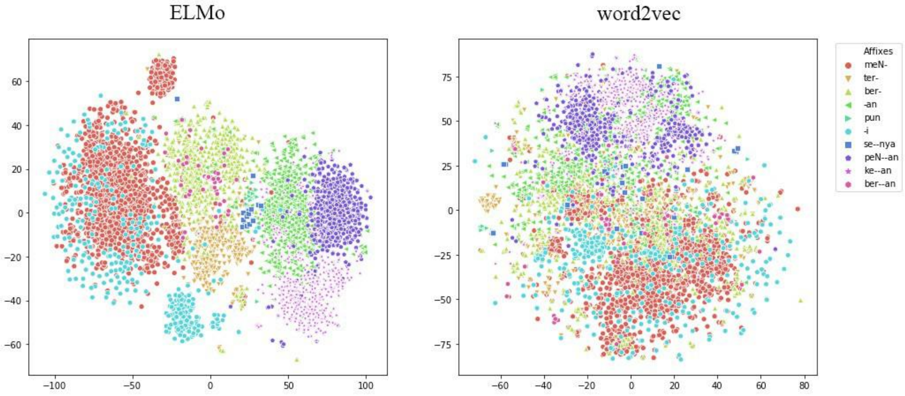

Fig. 3 Visualization of Affixed Words

7 Conclusion

The use of ELMo word representation gives an improvement in parsing the Indonesian language. The improvement is correlated with the morphological information of word representation. ELMo model performs much better than the word2vec model (83.08 vs. 81.35), with 1.73% gain in UAS. An analysis of word similarity and word visualization further indicates that ELMo can encode morphological information, which word2vec seems to lack such information. As future work, we would like to use newer word representations such as Bidirectional Encoder Representations from Transformers (BERT) or Generative Pretrained Transformer 2 (GPT-2). Another possibility for future work is to use a morphological analyzer so that information about word morphology can be obtained more accurately.

ACKNOWLEDGEMENTS

We would like to acknowledge the help provided by Research and Community Outreach at Universitas Multimedia Nusantara.

References

[1] Chaer, A. (2015). Morfologi Bahasa Indonesia. Jakarta: PT Rineka Cipta.

[2] Tsarfaty, R., Seddah, D., Goldberg, Y., et al. (2010). Statistical Parsing of Morphologically Rich Languages (SPMRL) What, How and Whither. In Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages, pages. 1-12.

[3] Tsarfaty, R., Seddah, D., Kübler, S., et al. (2013). Parsing morphologically rich languages: Introduction to the special issue. Computational Linguistics, vol.39 n.1, pages. 15-22.

[4] Smith, A., de Lhoneux, M., Stymne, S., et al. (2018). An Investigation of the Interactions Between Pre-Trained Word Embeddings, Character Models and POS Tags in Dependency Parsing. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages. 2711–2720.

[5] Vania, C., Grivas, A. and Lopez, A. (2018). What do character-level models learn about morphology? the case of dependency parsing. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages. 2573–2583.

[6] Rusli, A., Suryadibrata, A., Nusantara, S. B., and Young, J. C. (2020). A Comparison of Traditional Machine Learning Approaches for Supervised Feedback Classification in Bahasa Indonesia in International Journal of New Media Technology. In Press.

[7] Rusli, A. and Young, J.C. (2019). Review and Visualization of Facebook's FastText Pretrained Word Vector Model. In Proceedings of the International Conference on Engineering, Science, and Industrial Application 2019, pp. 1-6.

[8] Peters, M.E., Neumann, M., Iyyer, M., et al. (2018). Deep contextualized word representations. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Vol.1, pages. 2227–2237.

[9] Kim, Y., Jernite, Y., Sontag, D., et al. (2016). Character-aware neural language models. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16), pages. 2741-2749.

[10] Chen, D. and Manning, C.D. (2014). A fast and accurate dependency parser using neural networks. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, pages. 740-750.

[11] Ballesteros, M., Dyer, C. and Smith, N.A. (2015). Improved Transition-based Parsing by Modeling Characters instead of Words with LSTMs. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing,

pages. 349-359

[12] Kübler, S., McDonald, R. and Nivre, J. (2009). Dependency Parsing: Synthesis Lectures on Human Language Technologies. New York: Morgan and Claypool Publishers.

[13] McDonald, R., Nivre, J., Zeman, D., et al. (2018). UD Indonesian GSD. [online] Available from: <https://universaldependencies.org/treebanks/id_gsd/index.html> [Accessed 27 October. 2019]

[14] Mikolov, T., Sutskever, I., Chen, K., et al. (2013). Distributed representations of words and phrases and their compositionality. In Proceedings of the 26th International Conference on Neural Information Processing Systems, pages. 3111-3119.

[15] van der Maaten, L. and Hinton, G. (2008). Visualizing High-Dimensional Data Using t-SNE. In Journal of Machine Learning Research, vol. 9, pages. 2579-2605.

Notes on contributors

Anthony is a student at the Department of Informatics, Universitas Multimedia Nusantara, Indonesia. He puts interests in Artificial Intelligence, especially on Natural Languange Processing.

Alethea Suryadibrata is a lecturer at the Department of Informatics, Universitas Multimedia Nusantara, Indonesia. Her main teaching and research interests include digital image processing, computer graphics and animation, reliable software engineering, and applied artificial intelligence and machine learning. She has published several research articles in International Journal of Advances in Soft Computing and its Applications; International Journal of Interactive Mobile Technologies; International Journal of Advanced Trends in Computer Science and Engineering; Journal of Information and Communication Convergence Engineering; and International Journal of New Media Technology.

Julio Christian Young is a lecturer at the Department of Informatics, Universitas Multimedia Nusantara, Indonesia. His main teaching and research interests include digital image processing, declarative programming, bioinformatics and applied artificial intelligence and machine learning. He has published several research articles in International Journal of Advances in Soft Computing and its Applications; International Journal of Scientific Technology and Research; International Journal on Telecommunication, Computing, Electronics and Control; and International Journal of Emerging Trends in Engineering Research.