Semantic Role Labeling in Conversational Chat using Deep Bi-Directional Long Short-Term Memory Networks with Attention Mechanism

Valdi Rachman[1], Rahmad Mahendra[1], Alfan Farizki Wicaksono[1], Ahmad Rizqi Meydiarso[2], and Fariz Ikhwantri[2]

[1]Faculty of Computer Science, Universitas Indonesia, Depok, Indonesia
[2]Kata.ai, Jakarta, Indonesia
valdi.rachman@gmail.com
{rahmad.mahendra, alfan}@cs.ui.ac.id
{rizqi, fariz}@kata.ai

Abstract

Semantic Role Labeling (SRL) has been extensively studied, mostly for understanding English formal language. However, only a few reports exist for informal conversational text, especially for language being used in the chatbot system. The challenges of informal texting language include a wide variety of slangs and abbreviations, short sentences, as well as disorganized grammars. In this work, we propose an attention mechanism on top of Long Short-Term Memory Networks architecture for solving SRL task on informal conversations. The task is evaluated on informal language used on an Indonesian chatting platform. Our proposed model achieves F1 score of 82.68%.

1 Introduction

Semantic Role Labeling (SRL) is a task in Natural Language Processing (NLP) which aims to automatically assign semantic roles to each argument for each predicate in a given input sentence. As for a brief definition, given an input sentence, SRL system will give an output of *"Who did what to whom"* with *what* as the predicate and *who* and *whom* being the argument of the predicate. SRL is an integral part of understanding natural language as it helps the machine to retrieve semantic information from the input. In practice, SRL has been widely used as one of the intermediate steps for many NLP tasks, some of which are information extraction (Surdeanu et al., 2003; Emanuele et al., 2013), machine translation (Liu and Gildea, 2010; Lo et al., 2013), and question-answering (Shen and Lapata, 2007; Moschitti et al., 2003).

In the chat bot industry, the bots need to understand semantic information of the user's text in order to generate more personalized response. A chat bot system typically works in three steps, starting from understanding the incoming message from a user using Natural Language Understanding (NLU) system, tracking the dialogue using Dialogue State Tracking (DST), and finally generating a response using Natural Language Generation (NLG) system. SRL is a useful module under the NLU pipeline.

To illustrate, suppose that the user sends a message to the bot for ordering the tickets.
(*"buy 4 ticket 4 US"*).

The SRL system is expected to extract the semantic information from the chat as follows.

Predicate: *buy*
Patient: *tickets*
Beneficiary: *us*

The system detects that the predicate is *"buy"*, followed by *"tickets"* as the patient, or the thing being bought. The beneficiary, or the recipient, is *"us"*, the user. With these extracted semantic roles, the bot can ask the user further detail of request using DST, such as the flight time, the airline, or user's budget.

As we can see from the example, it is worth noting that the style of language used on chatting platform is different than those in formal text. In this work, we call this as *conversational language*. While formal language has been extensively studied in terms of SRL system, conversational language is yet to explore. The language is informal and thus, it has some

unique characteristics including a wide variety of slangs and abbreviations, short sentences, and disorganized grammars. These characteristics are the challenges an SRL system should tackle in understanding conversational language.

Our main contribution in this work is solving SRL problem on conversational language used in an Indonesian chatting platform. We deep dive into the semantic role characteristics found in the language. We use deep learning as the state-of-the-art approach that has been emerging in NLP field for doing the SRL task. We study Deep Bi-Directional Long Short-Term Memories and propose an attention mechanism in order to capture context information of the sentence at a higher level. Furthermore, we incorporate word embedding with POS-Tag information as the features to train the SRL model.

This paper is organized as follows. We first explain the previous works on SRL systems in Section 2. In Section 3, the methodology of the research is described, including the features and the model architectures being used. The results and analysis of the experiments are then discussed in Section 4. Finally, we report our conclusion and potential future works in the last section.

2 Related Work

SRL can be seen as either a classification or sequence labeling problem. The earlier research on SRL was conducted with the classification approach, meaning that each argument is being predicted independently from the others. Those research focused on how to extract meaningful features out of syntactic parsers (Gildea and Jurafsky, 2002; Gildea and Palmer, 2002; Pradhan et al., 2005), such as the path to predicate and constituent type. This syntactic information plays a pivotal role in solving SRL problem (Punyakanok et al., 2008) as it addresses SLR's long distance dependency. Thus, traditional SRL system heavily depends on the quality of the parsers. Pradhan et al. (2005) analyzes that most errors of the SRL system were caused by the parser's error. In addition, those parsers are costly to build, since it needs linguistic experts to annotate the data. If we want to create an SRL system on another language, one should build a new parser all over again for it.

In order to minimize the number of hand-crafted features, Collobert et al. (2011) utilized deep learning for solving NLP tasks including Part-of-Speech (POS) Tagging, Chunking, Named Entity Recognition (NER), and Semantic Role Labeling (SRL). The research aims to prevent using any task-specific feature in order to achieve state-of-the-art performance. The word embedding is used as the main feature across tasks, combined with Convolutional Neural Networks (CNN) architecture to train the model. To achieve competitive result for the SRL, the features engineered from the parser are still needed.

Zhou and Xu (2015) and He et al. (2017) view SRL as a sequence labeling problem in which the arguments are labeled sequentially instead of independently. They proposed an end-to-end learning of SRL using Deep Bi-Directional Long Short-Term Memories (DB-LSTM), with word embedding as the main feature. Their analysis suggests that the DB-LSTM model implicitly extracts the syntactic information over the sentences and thus, syntactic parser is not needed. The research result outperforms the previous state-of-the-art traditional SLR systems. The research also shows that the performance of the sequence labeling approach using DB-LSTM is better than the classification approach using CNN, since the DB-LSTM architecture can extract syntactic information implicitly.

A few number of works on conversational language explore SRL task in the context of Spoken Language Understanding. Coppola et al. (2009) exploits machine learning of frame semantics on spoken dialogs. They design and evaluate automatic FrameNet-based parsers both for English written texts and for Italian dialog utterances. Dinarelli et al. (2009) describe and analyze the annotation process in order to train semantic statistical parsers. Spoken conversations from both a human-machine and a human-human spoken dialog corpus are semantically annotated with predicate argument structure.

3 Method

We view SRL as a sequence labeling problem. Suppose that we have an input of n words $w = (w_1, w_2, ..., w_n)$, the goal is to find the best label sequence $y = (y_1, y_2, ..., y_n)$, with y_i representing the semantic roles.

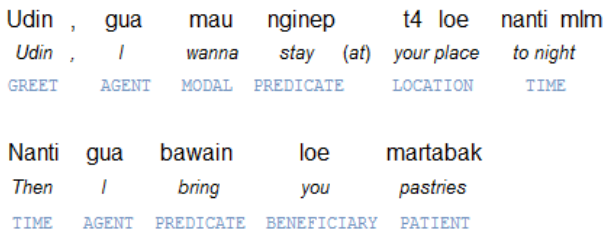

Figure 1: Semantic roles annotation example

The probability of the label in each time step i is described as follows.

$$P(y_i|w_{i-l}, ..., w_{i+l}, y_{i-l}, ..., y_{i+l})$$

3.1 Semantic Roles for Conversational Language

We observe sample of chat data and find that only limited number of semantic roles from Prop-Bank set (Palmer et al., 2005) are commonly used in the conversational language. In total, we apply 8 semantic roles[1]: AGENT, PATIENT, BENEFICIARY, PREDICATE, MODAL, LOCATION, TIME, and GREET. GREET refers to an animate object, usually a person, which is being greeted in a chat. In conversational language, one often calls the name of person it is talking to. Figure 1 depicts two examples of semantic-role-annotated conversational sentences.

3.2 Features

We utilize word embedding and POS-tag as main features with neighboring words as the secondary feature. While most of the SRL researches use predicate information as one of the main features, we omit to use it since we seek for an SRL system which finds the predicate from scratch alongside with other semantic roles.

1. Word Embedding. The embedding represents word as a vector. The interesting characteristic of word embedding is that similar words have proved to have similar vectors (Mikolov et al., 2013a). This is very important when dealing with conversational language which has a lot of slang words.

We use pre-trained word embedding as the main word representation. We trained the model with our chat corpus using Word2Vec's Skipgram architecture (Mikolov et al., 2013b). All the words were lowercased before being fed into the model.

2. POS-Tag. POS-Tag is a feature to represent the class of each word. In this research, the POS tags used are: Verb (V), Noun (NN), Adjective (ADJ), Adverb (ADV), Coordinative Conjunction (CC), Subordinative Conjunction (SC), Interjection (INTJ), Question (WH), Preposition (PREP), and Negation (NEG). Before feeding the feature to the deep learning model, we encode the POS tag as a one-hot-vector.
While most of the deep learning research aims for not using such feature, we argue that POS tag is still important for building a robust model in our case, knowing that the size of our corpus is relatively small. We argue that having Verb as one of our POS Tag will help to determine which one is the predicate, since predicate is usually in a form of verb, though some can also be adjective. As the arguments having semantic role are mostly in a form of Noun, POS Tag Noun will be a helpful information as well. Other POS tags will help to determine which word that obviously does not have any semantic role.

3. Neighboring Words. Neighboring word embeddings are the vector representations of words located before and after the word being processed. We use specifically one word before and after the word being processed. Suppose that we are processing the word w_t at time step t, the neighboring words embeddings are the vector representations of the word w_{t-1} and w_{t+1}. We argue that this feature can be useful for capturing the context of the word by looking at the surrounding words.

[1]In DeepLo publication (Ikhwantri et al., 2018), we only reported 7 semantic roles. In this work, we separate modal verb from verbal predicate

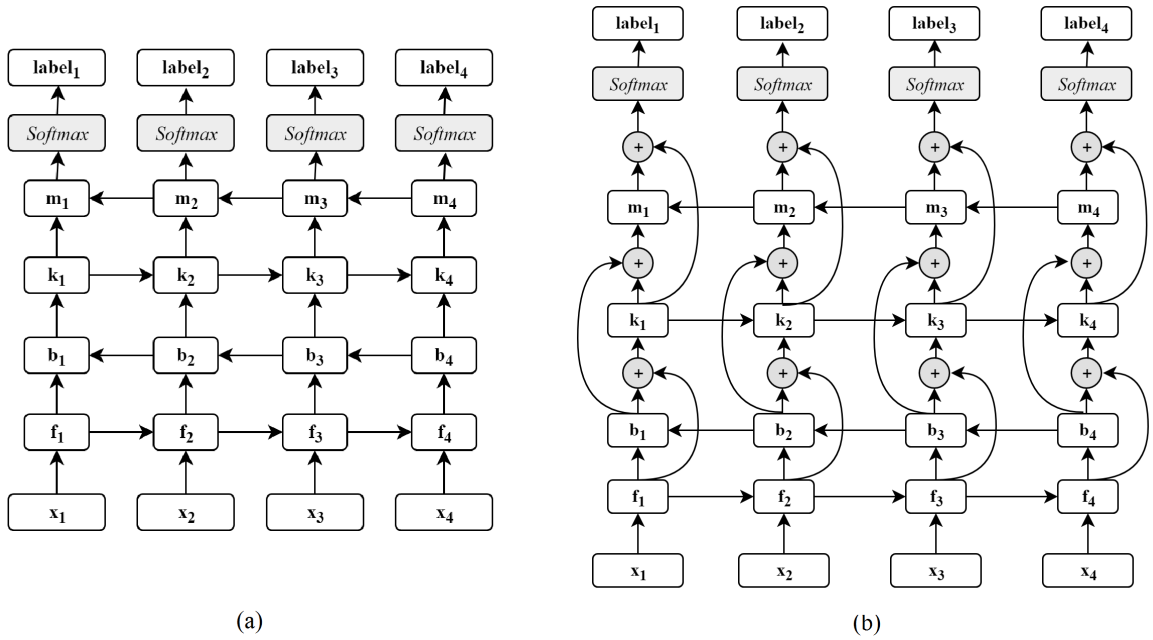

Figure 2: An architecture of (a) DBLSTM-Zhou, (b) DBLSTM-Highway with total time step of 4

3.3 Deep Learning Architecture

Recurrent Neural Networks (RNN) has a nature advantage for solving sequence labeling problem. Hochreiter and Schmidhuber (1997) proposed Long Short-Term Memories (LSTM) as the specific version of RNN designed to overcome vanishing and exploding gradient problem. LSTM architecture is used as the main layer. In addition, we propose an attention layer, which is placed on top of the main layer.

3.3.1 LSTM Layer

In this work, we experiment on various LSTM architectures, namely vanilla LSTM, Bi-Directional LSTM (BLSTM), Deep BLSTM (DBLSTM), DBLSTM-Zhou, and DBLSTM-Highway. Vanilla LSTM is the basic, one-directional LSTM networks. Bi-Directional LSTM (BLSTM) is a modification of LSTM networks. Schuster and Paliwal (1997) firstly introduced it for the original RNN. While vanilla LSTM only goes on one direction, BLSTM goes both ways in order to capture context information from the past and future. Deep BLSTM (DBLSTM) is basically formed by stacking BLSTM layers. In this work, we stack two BLSTM layers.

Zhou and Xu (2015) proposed another way of constructing the BLSTM layer. First, an LSTM layer processes the input sequence in a forward direction. The output of this layer is then fed into the next LSTM layer as input, processed in backward direction. At this point, one BLSTM-Zhou layer is built. (Zhou and Xu, 2015) then stacked the BLSTM layers and named it as the deep bi-directional LSTM. In this work, we stack two BLSTM-Zhou layers for constructing the DBLSTM-Zhou. The DBLSTM-Highway architecture is adapted from He et al. (2017). They used the same BLSTM architecture as Zhou and Xu (2015) but added the so-called highway connections (Srivastava et al., 2015) for their DBLSTM architecture. Figure 2 illustrates the DBLSTM-Zhou and DBLSTM-Highway architectures respectively.

3.3.2 Attention Mechanism

We propose an additional attention mechanism that can be used on top the main layer. The rationale is to add a dense yet useful high-level information containing a sentence context to every time step in order to help the machine to decide semantic roles better. With this in mind, we design an attention mechanism on top of the main layer which can be any of the LSTM variants.

Figure 3 illustrates the architecture in which attention mechanism is added on top of the main layer. In the illustration, the DBLSTM-Highway is used as the main layer.

The attention mechanism firstly collects the context information by multiplying trainable weights with all the vectors from every time step of the last LSTM output. We sum each element for each weighted vectors to reduce the dimension. The results are then fed into a hidden softmax layer which outputs weights with a total of 1. The original output vectors of the last LSTM output are multiplied by these distributed weights respectively. We then sum all the multiplication results as the final context information. The original LSTM outputs are concatenated with the context information before going to the last softmax layer to predict the semantic roles.

Suppose that we have $\mathbf{s_i}$ as the last LSTM output, it is then fed into a differentiable neural networks function $g(\mathbf{s_i})$ in which it is multiplied by the time-distributed matrix $\mathbf{W} \in \mathbb{R}^{\mathrm{H}\times\mathrm{K}}$ and all the elements in it are summed. H is the vector dimension of $\mathbf{s_i}$, meanwhile K is the dimension size that we want as an output when we multiply W with $\mathbf{s_i}$.

$$g(\mathbf{s_i}) = Sum(\mathbf{W}.\mathbf{s_i})$$

Once we have all the values of $[g(\mathbf{s_1}); g(\mathbf{s_2}); ...; g(\mathbf{s_n})]$, we make it as an input for the softmax layer, resulting weights $[\alpha_1; \alpha_2; ...; \alpha_n]$. All the original LSTM outputs $[\mathbf{s_1}; \mathbf{s_2}; ...; \mathbf{s_n}]$ are multiplied by these weights, with the results of $[\mathbf{r_1}; \mathbf{r_2}; ...; \mathbf{r_n}]$

$$[\alpha_1; \alpha_2; ...; \alpha_n] = Softmax([g(\mathbf{s_1}); g(\mathbf{s_2}); ..., g(\mathbf{s_n})])$$

$$\mathbf{r_i} = \alpha_i.\mathbf{s_i}$$

We then sum all these vectors element-wise to have a context vector $\mathbf{z}$. All the original LSTM outputs are thus concatenated with vector $\mathbf{z}$ as the additional information to predict the semantic roles.

$$\mathbf{z} = \mathbf{r_1} + \mathbf{r_2} + ... + \mathbf{r_n}$$

$$\mathbf{j_i} = Concatenate(\mathbf{s_i}, \mathbf{z})$$

Lastly, the time-distributed softmax layer produces the final semantic roles label.

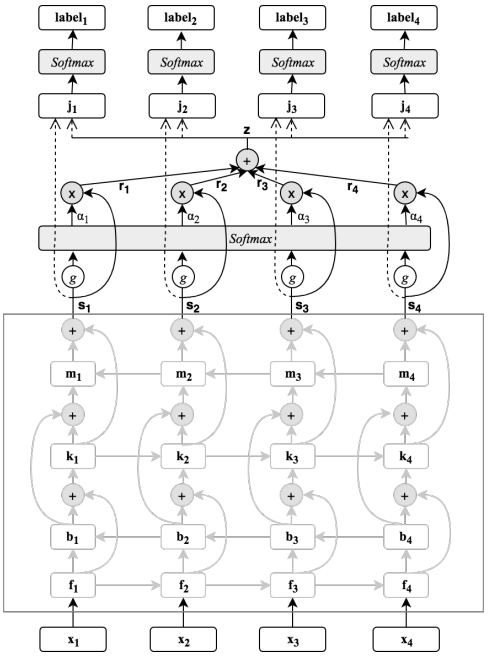

Figure 3: An architecture of adding attention mechanism on top of the main layer with total time step of 4. The main layer is illustrated by the shaded architecture inside the rectangle. The main layer can be changed into any LSTM variant.

4 Experiments

The dataset for the experiment consists of 6,057 sentences that are obtained from an Indonesian chatbot platform (Ikhwantri et al., 2018). Almost all sentences containing PREDICATE role, while BENEFICIARY and LOCATION roles are only found in less than 4% data. Figure 4 shows the portion of semantic roles in annotated dataset.

Our experiment consists of two scenario sets: feature selection and model selection. In feature selection, we experiment to find which feature combination outputs the best result. For model selection, the experiment focuses on finding the best model architecture to get the highest result.

We apply 5-fold cross validation in our experimental setting. In each experiment, we evaluate the model in order to see how good it predicts the semantic roles as expected. The metrics for our evaluation are precision, recall, and F1. These metrics are applied to all semantic role labels. We then average each metrics from all semantic role labels to get the average precision, recall, and F1 of a model.

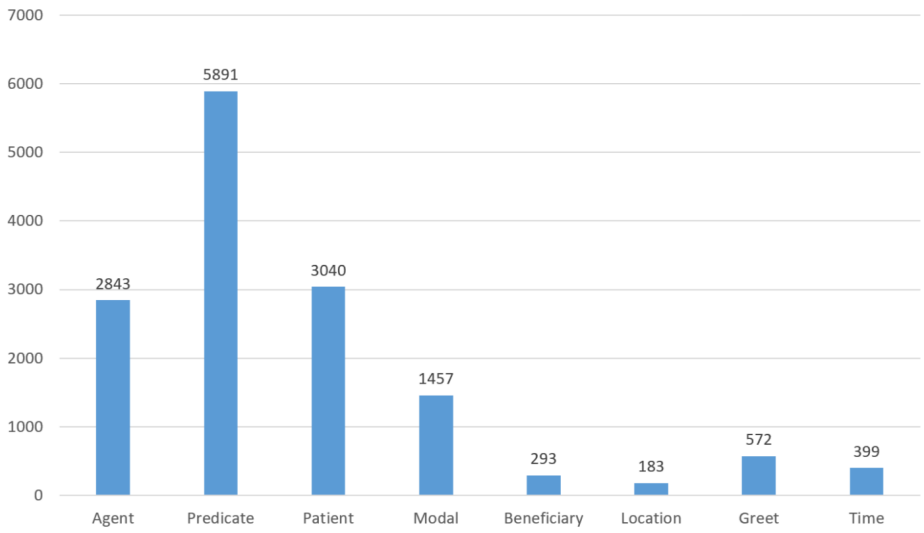

Figure 4: Semantic Role Label Distribution in Dataset

Table 1: Results (in %) of Feature Selection Scenario

Feature	Precision	Recall	F1
WE	75.20	71.16	73.12
WE + NW	76.07	71.58	73.76
WE + POS	**80.09**	**78.14**	**79.10**
WE + POS + NW	**80.96**	**78.60**	**79.76**

4.1 Feature Selection

The feature selection set consists of 4 combination scenarios, which are:

1. Word Embedding (WE)
2. Word Embedding + Neighboring Word Embeddings (WE + NW)
3. Word Embedding + POS Tag (WE + POS)
4. Word Embedding + POS Tag + Neighboring Word Embeddings (WE + POS + NW)

In this scenario set, we use vanilla LSTM for the base architecture.

Table 1 shows the result of feature selection experiments. The highest performance is achieved with the combination of all features, followed by WE + POS, with F1 scores of 79.76% and 79.10%, respectively.

4.2 Model Selection

The model selection consists of 3 scenario sets. The first set aims to find which feature combination is better when the LSTM layers are added. The next set compares the performance between DBLSTM, DBLSTM-Zhou, and DBLSTM Highway. Lastly, the third set aims to see the impact of our proposed attention layer towards the performance.

We first experiment on comparing the top two feature combinations from the previous set (WE + POS and WE + POS + NW) when using vanilla LSTM, BLSTM, and Deep BLSTM (DBLSTM) architectures. Table 2 shows the result of this experiment.

When using vanilla LSTM, the F1 scores of WE + POS and WE + POS + NW are 79.10% and 79.76%, respectively. When bi-directional LSTM (BLSTM) is used, the performances of both feature combinations slightly decrease: WE + POS decreases from 79.10% to 78.93%, and WE + POS + NW decreases from 79.76%. However, when two BLSTM layers are stacked (DBLSTM), the F1 scores of both feature combinations increase compared to the vanilla LSTM architecture. While WE + POS + NW slightly increases by 0.11% (from 79.76% to 79.87%), the feature combination WE + POS improves by 3.20% (from 79.10% to 82.30%).

According to Zhou and Xu (2015), bi-directional LSTM architecture is able to implicitly capture the context information from the past (words on the left)

Table 2: F1 scores (in %) of WE + POS and WE + POS + NW when using vanilla LSTM (LSTM), BLSTM, and Deep BLSTM (DBLSTM) architectures.

Feature	LSTM	BLSTM	DBLSTM
WE + POS	79.10	78.93	**82.30**
WE + POS + NW	79.76	79.12	79.87

Table 3: Precision, Recall, and F1 scores (in %) of Deep BLSTM (DBLSTM), DBLSTM-Zhou, and DBLSTM-Highway

Model	Precision	Recall	F1
DBLSTM	**82.97**	**81.66**	**82.30**
DBLSTM-Zhou	81.52	81.15	81.33
DBLSTM-Highway	80.63	81.66	81.14

and future (words on the right). Moreover, their research also shows that the stacked BLSTM architecture (DBLSTM) could enhance the model performance. In this case, we suggest that the BLSTM model, especially stacked BLSTM, does not need explicit context information such as neighboring word embeddings. Therefore, by only using WE + POS as the features, combined with DBLSTM, the model can achieve compelling result. Thus, we use WE + POS feature combination for the next model selection scenarios.

Then, we compare the Deep BLSTM (DBLSTM), DBLSTM-Zhou, and DBLSTM-Highway. Table 3 shows the precision, recall, and F1 scores of each architecture. The highest F1 score is achieved by the DBLSTM architecture (82.30%), followed by DBLSTM-Zhou (81.33%) and DBLSTM-Highway (81.14%).

In their work, Zhou and Xu (2015) did not elaborate on why they picked their proposed architecture (DBLSTM-Zhou) rather than the original DBLSTM. They did not compare the performances between original DBLSTM and their proposed architecture. Moreover, He et.al. (2017), who proposed DBLSTM-Highway as the modification of Zhou and Xu's architecture, also did not explain the rationale of using Zhou's BLSTM rather than the original one as the basis. In our experiment, it turns out that the original DBLSTM still outperforms the other BLSTM architectures.

Table 4 shows the result of experiments aiming to see the impact of using our proposed attention layer on top of the three DBLSTM architectures. The original DBLSTM model slightly decreases by 0.09% when using attention layer (from 82.30% to 82.21%). On the other hand, the performances of both DBLSTM-Zhou and DBLSTM-Highway models increase when attention layer is used. While DBLSTM-Zhou model increases by 0.64% (from 81.33% to 81.97%), the DBLSTM-Highway model increases by 1.54% (from 81.14% to 82.68%). From these results, we suggest that the attention layer successfully contributes to the DBLSTM-Zhou and DBLSTM-Highway models, while the layer may not suitable to be used on top of original DBLSTM architecture.

5 Conclusions and Future Works

Semantic Role Labeling (SRL) is an integral part of understanding semantic information of a text. One of its applications is to make chat bots understand user's chat better. Even though the SRL on English formal language has been widely studied, only a few reports exist for the informal conversational language, especially for language being used in the chatbot system. In Indonesian, both formal and conversational language are barely tapped for building SRL system. In this work, we focus on solving SRL for Indonesian conversational language. Our contributions are introducing a new set of semantic roles and proposing an attention mechanism on top of the Long Short-Term Memory Networks architecture.

We view SRL as a sequence labeling problem. One of the deep learning architectures that fits to solve sequence labeling problem is Long Short-Term Memories (LSTM), a modification of Recurrent Neural Networks (RNN). In this work, we experiment with a variety of LSTM networks, including vanilla LSTM, Bi-Directional LSTM (BLSTM), Deep BLSTM (DBLSTM), DBLSTM-Zhou, and DBLSTM-Highway. The features used are Word Embedding (WE), Part-of-Speech Tag (POS), and Neighboring Word Embeddings (NW).

We conducted two sets of experiment scenarios: feature selection and model selection. Our experiments on feature selection show that when the size of training data is relatively small, one still needs to use a traditional feature such as POS tag in addition to word embedding. The neighboring word embed-

Table 4: The precision, recall, and F1 scores (in %) of the DBLSTM architectures with and without attention layer.

	Without Attention			With Attention		
Model	Precision	Recall	F1	Precision	Recall	F1
DBLSTM	82.97	81.66	82.30	82.36	82.07	82.21
DBLSTM-Zhou	81.52	81.15	81.33	82.32	81.62	81.97
DBLSTM-Highway	80.63	81.66	81.14	**83.07**	**82.30**	**82.68**

dings feature can slightly improve the results. By using vanilla LSTM, the top two best feature combinations are WE + POS + NW and WE + POS with the F1 scores of 79.76% and 79.10%, respectively.

In model selection set, our experiments show that DBLSTM architecture can outperform the vanilla LSTM and BLSTM. The results also show that when using DBLSTM, the feature combination WE + POS can outperform the WE + POS + NW combination. Furthermore, our results show that the original DBLSTM can outperform its variants: DBLSTM-Zhou and DBLSTM-Highway. For the additional layer, we experiment on adding proposed attention mechanism to the three afore-mentioned DBLSTM architectures. In our experiment of adding the attention mechanism, the DBLSTM architecture's performance slightly decreases while the results of both DBLSTM-Zhou and DBLSTM-Highway improve. The highest result is achieved by using DBLSTM-Highway with the attention mechanism with an F1 score of 82.68%.

For future works, there is a lot of possibilities to design the architectures. For example, instead of putting the attention layer on top of the main layer, one can add it right after the input layer. Beside architectures, one can experiment with many features such as word shape, prefix, suffix, and neighboring POS tags to enhance the performance. Once the SRL system is established, one can focus on building the Natural Language Generation (NLG) system for chat bots based on the semantic roles of the Indonesian conversational language. This way, one can create more intelligent chat bots which understand deeper on conversational language. Another interesting work would be integrating coreference resolution on the SRL system knowing that conversational language is usually in a form of dialogue.

Acknowledgments

The publication of this work is supported by the PITTA UI Grant Contract No. 1886/UN2.R3.1/HKP.05.00/2018.
The authors thank to the members of Information Retrival Lab, Faculty of Computer Science, Universitas Indonesia, especially Dr. Mirna Adriani.

References

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Bonaventura Coppola, Alessandro Moschitti, and Giuseppe Riccardi. 2009. Shallow semantic parsing for spoken language understanding. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, NAACL-Short '09, pages 85–88, Stroudsburg, PA, USA. Association for Computational Linguistics.

Marco Dinarelli, Silvia Quarteroni, Sara Tonelli, Alessandro Moschitti, and Giuseppe Riccardi. 2009. Annotating spoken dialogs: From speech segments to dialog acts and frame semantics. In *Proceedings of the 2nd Workshop on Semantic Representation of Spoken Language*, SRSL '09, pages 34–41, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bastianelli Emanuele, Giuseppe Castellucci, Danilo Croce, and Roberto Basili. 2013. Textual inference and meaning representation in human robot interaction. In *Joint Symposium on Semantic Processing.*, page 65.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational linguistics*, 28(3):245–288.

Daniel Gildea and Martha Palmer. 2002. The necessity of parsing for predicate argument recognition. In *Proceedings of the 40th Annual Meeting on Association*

for Computational Linguistics, pages 239–246. Association for Computational Linguistics.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780. MIT Press.

Fariz Ikhwantri, Samuel Louvan, Kemal Kurniawan, Bagas Abisena, Valdi Rachman, Alfan Farizki Wicaksono, and Rahmad Mahendra. 2018. Multi-task active learning for neural semantic role labeling on low resource conversational corpus. In *Proceedings of the Workshop on Deep Learning Approaches for Low-Resource NLP*, pages 43–50. Association for Computational Linguistics.

Ding Liu and Daniel Gildea. 2010. Semantic role features for machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 716–724. Association for Computational Linguistics.

Chi-kiu Lo, Karteek Addanki, Markus Saers, and Dekai Wu. 2013. Improving machine translation by training against an automatic semantic frame based evaluation metric. In *ACL (2)*, pages 375–381.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Alessandro Moschitti, Paul Morarescu, and Sanda M Harabagiu. 2003. Open domain information extraction via automatic semantic labeling. In *FLAIRS conference*, pages 397–401.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Comput. Linguist.*, 31(1):71–106, March.

Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James H Martin, and Daniel Jurafsky. 2005. Semantic role labeling using different syntactic views. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 581–588. Association for Computational Linguistics.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2):257–287.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681. IEEE.

Dan Shen and Mirella Lapata. 2007. Using semantic roles to improve question answering. In *EMNLP-CoNLL*, pages 12–21.

Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Training very deep networks. In *Advances in neural information processing systems*, pages 2377–2385.

Mihai Surdeanu, Sanda Harabagiu, John Williams, and Paul Aarseth. 2003. Using predicate-argument structures for information extraction. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 8–15. Association for Computational Linguistics.

Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *ACL (1)*, pages 1127–1137.