Time Series Neural Network Model for Part-of-Speech Tagging Indonesian Language

Theo Tanadi
School of Electrical Engineering and Informatics, Institut Teknologi Bandung
18214041@std.stei.itb.ac.id

Abstract. Part-of-speech tagging (POS tagging) is an important part in natural language processing. Many methods have been used to do this task, including neural network. This paper models a neural network that attempts to do POS tagging. A time series neural network is modelled to solve the problems that a basic neural network faces when attempting to do POS tagging. In order to enable the neural network to have text data input, the text data will get clustered first using Brown Clustering, resulting a binary dictionary that the neural network can use. To further the accuracy of the neural network, other features such as the POS tag, suffix, and affix of previous words would also be fed to the neural network.

Keyword: Neural network, Part-of-speech tagging, Brown clustering

1. Introduction
Part-of-speech (POS) tagging is one of the most basic natural language processing tasks. The main objective of POS tagging is to give a part-of-speech tag to each token in a given text. There are many approaches to solve this problem, such as rule-based tagger, Hidden Markov Model [1], and artificial neural network. The usage of artificial neural network in POS tagging task has been proven to perform better than other approaches, with many variations.

Feedforward neural network that has been modified has been used for POS tagging task since 1990s [2]. Other more sophisticated neural network architecture started to emerge along with many other advancements in natural language processing field. The direction of POS tagging task approach started to lean towards deep learning and recurrent neural network [3].

In this paper we will try to create a simple time series neural network model to solve POS tagging task. Time series neural network is a feedforward neural network which have previous output(s) as its input. Time series neural network is slightly different from recurrent neural network. Recurrent neural network has recurrent connection, making its history length unlimited, whereas time series neural network only can have its history length specified by the number of previous outputs that is going to be fed to the next input.

In the next section, we will discuss about the basic of artificial neural network and Brown clustering. After that, the model and training method of the neural network will be explained. Analysis and testing plan will follow. This paper will be finished with description of future work and conclusion.
2. Artificial Neural Network
Artificial neural network is a type of computing system that is inspired from brain neural network. Artificial neural network consists of multiple layer of neurons [4]. A basic artificial neural network has 2 layer, its input and its output. A more complex artificial neural network has another layer between its input layer and output layer. Each neuron that is connected to each other will have a weight that determines the strength of the connection. Each neuron also has an activation function that will transform the value of the input.

Artificial neural network works by feeding input to its neuron, multiply it with corresponding weight, transform it with its activation function, and transfers its output all the way through from its input layer to its output layer. With output as the output of the neuron, f as the activation function, input as the input of the neuron, and weight as the weight from the previous neuron, the output of a neuron can be represented by this formula:

\[
\text{output} = f(\text{input} \cdot \text{weight})
\]

3. Brown Cluster
One of artificial neural network drawback is that it cannot process raw text. Some adjustment need to be made before feeding text into neural network. There are a few methods that can be used to enable neural network to accept text input, such as word embedding [5] and Brown clustering. The method that will be used in this paper is Brown clustering.

Brown clustering works by clustering words based on their occurrence and the occurrence of nearby words. The idea is that similar words will have similar distributions of their nearby words. Brown clustering inputs a large text corpus and outputs a binary dictionary of words that exist in the input corpus. Similar words will have similar most significant bit (MSB). To use the dictionary from Brown clustering, we take fixed length MSB from the dictionary to represent the words.

4. Model
In this paper, the neural network model that is designed to do POS tagging task will have 3 layer, which is one input layer, one hidden layer, and one output layer. Each neuron in the input layer will be connected to every neuron in the hidden layer and each neuron in the hidden layer will be connected to every neuron in the output layer. It is possible to use any other activation function, but in this paper each neuron will use sigmoid function as its activation function.

![Proposed neural network model diagram (simplified). The bottom layer is the input layer, the middle layer is the hidden layer, and the top layer is the output layer. The numbers at the bottom represent the number of iteration the neural network have done.](image)

There are several inputs that is going to be fed to the model. The first one is the word that is going to be tagged. The word will get transformed first according to the binary dictionary from the Brown
cluster. The size of the binary word hasn’t been decided yet. The model also can get the POS tag from two previous words as its input. Another input that the model can have is the suffix and prefix from the word that is going to be tagged. The suffix and prefix of the word are received from CS stemmer [6]. The input layer is expected to have seventy to ninety nodes.

As for the hidden layer, the size of the node will be determined by trial and error [7]. But as a baseline, one hidden layer and the product of the number of nodes in input layer and output layer as the size of the hidden layer will be used.

For the output layer, the only output is the POS tag of the word. The POS output will be represented by N neuron where N is the number of POS tagset [8]. Currently, there is no standard Indonesian language corpus and tagset, but there are several previous works that attempt to create a standard Indonesian language tagset. This model will use proposed Indonesian tagset by Dinakaramani et al [9].

5. Model Training

The model use gradient descent method to find the optimal weight for each neuron. These are the steps for the step to train:

1. First the method will try to calculate the output using the initial weight. Error for the output layer is calculated by using this formula:

   \[
   output\_error = desired\_output - calculated\_output
   \]

2. After the error is found, we calculate the delta for the output layer. With \( f(x) \) as the activation function and \( x \) as the calculated output, the delta for the output layer is calculated by using this formula:

   \[
   output\_delta = output\_error \ast \frac{d f(x)}{dx}
   \]

   By using sigmoid function as our activation function, the delta formula can also be written as:

   \[
   output\_delta = output\_error \ast (calculated\_output \ast (1 - calculated\_output))
   \]

3. We calculate the error for the hidden layer by back propagating the error from the output layer to the hidden layer. The error from the output layer is called confidence weighted error since it is calculated only from the desired output and the error from the hidden layer is called contributed weighted error since it is calculated from the error from the output layer. The error for the hidden layer is calculated by using this formula:

   \[
   hidden\_error = output\_delta \ast hidden\_to\_output\_weight
   \]

4. With \( calculated\_hidden \) as the value of the hidden neuron when the output is calculated, the delta from the hidden layer with this formula:

   \[
   hidden\_delta = hidden\_error \ast (calculated\_hidden \ast (1 - calculated\_hidden))
   \]

5. The new weight of the hidden layer to the output layer is calculated with this formula:

   \[
   hidden\_to\_output\_weight = hidden\_to\_output\_weight + (calculated\_hidden \ast output\_delta)
   \]

6. With \( input \) as the input value that is fed to the input neuron, the new weight of the input layer to the hidden layer is calculated with this formula:

   \[
   input\_to\_hidden\_weight = input\_to\_hidden\_weight + (input \ast hidden\_delta)
   \]

6. Analysis and Testing Plan

There are some factors that can manipulate the accuracy of the neural network, such as the input features, the size and number of hidden layer, and the neural network optimization algorithm. These factors can also affect the accuracy of the neural network that we made.
Previous works that uses feedforward neural network usually use the POS tag of some of previous words and the POS tag of some next words. The usage of the prefix and suffix of the word hasn’t been done before. Words with certain combinations of prefix and suffix tend to have certain POS tag. For example, *permainan* (game), *pelarian* (runaway), and *perlawanan* (resistance) have “pe-“ prefix, “-an” suffix and NN (noun) as its POS tag. Another example is *bermain* (playing), *berlari* (running), and *berdiri* (standing) have “be-“ prefix, no suffix, and VB (verb) as its POS tag. There are some exceptions for these pattern, but it still might lead to better accuracy for the neural network. Testing need to be done to ensure if it will really increase the accuracy of the neural network.

There are many discussions about the number of hidden layer and the size of the hidden layer that are needed in a neural network architecture to achieve optimal results. Many of them agree that more hidden layer will lead to better accuracy especially for difficult object, but the universal approximation theory claims that a feedforward neural network with one hidden layer with finite number of hidden unit and arbitrary activation function is capable to approximate a finite set [10,11]. Trial and error is needed to determine the best hidden layer number and hidden layer nodes number for a neural network to do certain task.

Neural network optimization algorithm is the algorithm that is used by neural network to find the most optimal weights of each connection in the neural network. The weights are considered optimal when its loss function is as minimal as possible. There are many optimization algorithms that can be used in neural network such as gradient descent, genetic algorithm [12], Adam [13], and Adagrad [14]. This model will use gradient descent optimization algorithm since it is the simplest. It does not rule out the possibility of using another optimization algorithm since these optimization algorithm is architecture independent.

The biggest obstacle to implement this model is the lack of training data. There are several researches in natural language processing that uses Indonesian language, which the training data can be used for this model. With these existing training data, we can create binary dictionary of words using Brown clustering and train the model itself. Unfortunately, we have not get our hand into any of these data.

There are several matrices that can be used to compare the model that we created with other model from previous works, such as accuracy and speed. This model is expected to have same or better accuracy than the HMM method as the HMM method is one of the most popular method to do POS tagging task.

7. Future Works

One of the most important thing that needs to be done is to get training data for this model. The training data can be obtained from other pervious researches or by creating one our self. The more training data that can be obtained, the better. Training data that consists at least ten thousand words and their corresponding POS tag is expected. Some preprocessing might be needed beforehand so the data can be fed to the neural network.

The prototype of the neural network has been made. This prototype can only process predefined binary data. This prototype achieved 100% accuracy for a really small training data and test data. Some adjustment is needed for this prototype so this prototype become able to do its intended purpose with sufficient accuracy. This paper is created in relevance to knowledge management research that has been conducted previously [15,16].

8. Conclusion

We have designed a neural network model to do part-of-speech tagging task. The model has not implemented yet because the lack of training data. The model will be implemented in the future after acquiring sufficient training data.
Acknowledgement
This study is financially supported by Institut Teknologi Bandung (ITB) under Program Penelitian, Pengabdian kepada Masyarakat, dan Inovasi (P3MI). We would like to thank Krisdanto Surendro (Information System Laboratory – SEEI ITB) for giving guidance throughout the writing of this paper.

References
[1] Wicaksono A F and Purwarianti A 2010 HMM Based Part-of-Speech Tagger for Bahasa Indonesia 4th International MALINDO (Malay and Indonesian Language) Workshop.
[2] Schmid H 1994 Part-of-Speech Tagging with Neural Networks COLING 94 Proc. 15th Int. Conf. Comput. Linguist. 1 p 5.
[3] Plank B, Søgaard A and Goldberg Y 2016 Multilingual Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Models and Auxiliary Loss pp 412–8.
[4] Wang S-C 2003 Artificial Neural Network Interdiscip. Comput. Java Program 81–100.
[5] Mikolov T, Chen K, Corrado G and Dean J 2000 Distributed Representations of Words and Phrases and their Compositionality CrossRef List. Deleted DOIs 1 pp 1–9.
[6] Asian J 2007 Effective Techniques for Indonesian Text Retrieval Thesis.
[7] Stathakis D 2009 How many hidden layers and nodes? Int. J. Remote Sens. 30 pp 2133–47.
[8] Aly M 2005 Survey on Multiclass Classification Methods Extensible algorithms Neural Networks pp 1–9.
[9] Dinakaramani A, Rashel F, Luthfi A and Manurung R 2014 Designing an Indonesian part of speech tagset and manually tagged Indonesian corpus Proc. of the Int. Conf. on Asian Language Processing 2014 (IALP 2014) pp 66–9.
[10] Csáji B 2001 Approximation with artificial neural networks.
[11] Hornik K 1991 Approximation capabilities of multilayer feedforward networks Neural Networks 4 pp 251–7.
[12] McCall J 2005 Genetic algorithms for modelling and optimisation J. Comput. Appl. Math. 184 pp 205–22.
[13] Kingma D P and Ba J L 2015 Adam: a Method for Stochastic Optimization Int. Conf. Learn. Represent. 2015 pp 1–15.
[14] Duchi J, Hazan E and Singer Y 2011 Adaptive Subgradient Methods for Online Learning and Stochastic Optimization J. Mach. Learn. Res. 12 pp 2121–59.
[15] Aradea, Suwardi I S and Surendro K 2015 An overview of multi agent system approach in knowledge management model 2014 Int. Conf. Inf. Technol. Syst. Innov. (ICITSI 2014) - Proc. pp 62–9.
[16] Atrinawati L H and Surendro K 2009 Assessment for knowledge management readiness Proc. 2009 Int. Conf. Electr. Eng. Informatics, (ICEEI 2009) 2 pp 399–404