An Artificial Neural Network Approach for Sentence Boundary Disambiguation in Urdu Language Text

Shazia Raj, Zobia Rehman, Sonia Rauf, Rehana Siddique, and Waqas Anwar
Department of Computer Science, COMSATS Institute of Information Technology, Pakistan

Abstract: Sentence boundary identification is an important step for text processing tasks, e.g., machine translation, POS tagging, text summarization etc., in this paper, we present an approach comprising of Feed Forward Neural Network (FFNN) along with part of speech information of the words in a corpus. Proposed adaptive system has been tested after training it with varying sizes of data and threshold values. The best results, our system produced are 93.05% precision, 99.53% recall and 96.18% f1-measure.

Keywords: Sentence boundary identification, feed forward neural network, back propagation learning algorithm.

Received April 22, 2013; accepted September 19, 2013; published online August 17, 2014

1. Introduction

Sentence boundary disambiguation is a problem in natural language processing that decides about the beginning and end of a sentence. Almost all language processing applications need their input text split into sentences for certain reasons. Sentence boundary detection is a difficult task as very often ambiguous punctuation marks are used in the text. For example, in English a period “.” appears at the end of the sentence as well as decimal in numbers, in email addresses, abbreviations and many more. Likewise question mark “?” and sign of exclamation “!” also, appear inside the sentence. Ellipses and quotations are also used in the text and they too add ambiguity to sentence terminator marks. Same ambiguities lie in Urdu text even with more complexity because of the absence of space and case discrimination. Case discrimination and smooth use of space between words are powerful clues to identify sentence boundary in many languages, for example, in English a period followed by a space character and a word starting with an upper case letter, is a strong candidate to be a sentence marker. Urdu follows the unicameral script of Arabic, with or without space between words. Use of space depends on the nature of the character a word ends with (joiner or non-joiner), space is used only after the words ending with a joiner character. Consider the given examples: [12].

- احمد بانگلہ - چہ سال شیر سے بار بار روا (Ahmad was out of the city for five to six years.)
- واو! کیا کامل کی جگہ ہے؟ (Wow! What a wonderful place.)
- "وہ جلالاؤ،" میری مدد کرو" (He Screamed, “Help me.”)
- کیوں؟ اس نے اپنی کسی غلطی کر دی؟ (Why? What did he do wrong?)

In such conditions it would be difficult for a machine to differentiate sentence terminators from ambiguous punctuations.

Urdu language processing is in infancy stage and application development for it is way slower for number of reasons. One of these reasons is lack of Urdu text corpus, either tagged or untagged. However, some work has been seen for sentence boundary disambiguation in Urdu text. The approach proposed in [11] produced good results but it was not adaptive. In this paper, we are introducing the use of Feed Forward Neural Network (FFNN) to identify sentence terminating punctuations in Urdu text files. The major advantages of this approach are; it can train itself with small sized corpus, it is adaptive from corpus to corpus, no hand crafted rules are required, its preprocessing is better and it requires less storage space [8]. We tested this approach for a corpus of 4288 sentences and achieved up to 93.05% precision, 99.53% recall and 96.18% f1-measure.

2. Related Work

There are different techniques available to detect sentence markers in the text. Some renowned techniques are: Collocation identification, rule based techniques, verb and inflection detection, machine learning techniques, regular expressions, Artificial Neural Network (ANN) models and part of speech tagging etc., [3, 4, 5, 6] used rule based technique to split long Vietnam’s sentences, based on linguistic information. Purpose of this work was to get rid of the syntactic structure discrepancy of different languages during translation. Poornima [10] formulates some rules based on noun and relative noun detection to split...
boundaries in the text but our purpose processing is in infancy stage and adequate language built grammatical rules, cops with scarce resources. We chose FFNN to mark sentence boundaries in Urdu text and it is robust, free of hand built grammatical rules. As sentence boundary disambiguation is a preprocessing phase for many language processing tasks, for this reason we need an approach which should be speediest in training and require less space for storage. Urdu language processing is in infancy stage and adequate language resources are not available for up to the mark training of language models, therefore the proposed approach which has been chosen best fits with limited language resources. We chose FFNN to mark sentence boundaries in Urdu text and it is robust, free of hand built grammatical rules, cops with scarce resources languages, speedier in training and has tolerable storage requirements.

For many years ANN is being applied in different areas of computation. ANN models are flexible and adaptive; they learn and adjust with each different internal or external stimulus. Artificial neurons are non linear, parameterized functions with restricted output range. Back propagation is a common method to train ANN. It is a supervised learning method and implements delta learning rule [14]. Most commonly used ANN models are nothing more than non-linear regression and discriminant models which can be implemented with standard statistical software [13].

In proposed approach a corpus of Urdu text with 123714 tagged words has been used. Using word-tag frequency the corpus has been converted to bipolar descriptor arrays which in turn are used to train the FFNN. Purpose of preparing bipolar arrays is to reduce the error rate and to speed up the training. After its adequate training we tested the model for different data sets and achieved pleasing results.

3. Proposed Technique

Although, there are many techniques for different languages of the world to disambiguate sentence boundaries in the text [3, 4, 5, 6, 9, 10] but our purpose is to find the one which should be robust and free of hand built grammatical rules. As sentence boundary disambiguation is a preprocessing phase for many language processing tasks, for this reason we need an approach which should be speediest in training and require less space for storage. Urdu language processing is in infancy stage and adequate language resources are not available for up to the mark training of language models, therefore the proposed approach which has been chosen best fits with limited language resources. We chose FFNN to mark sentence boundaries in Urdu text and it is robust, free of hand built grammatical rules, cops with scarce resources languages, speedier in training and has tolerable storage requirements.

For many years ANN is being applied in different areas of computation. ANN models are flexible and adaptive; they learn and adjust with each different internal or external stimulus. Artificial neurons are non linear, parameterized functions with restricted output range. Back propagation is a common method to train ANN. It is a supervised learning method and implements delta learning rule [14]. Most commonly used ANN models are nothing more than non-linear regression and discriminant models which can be implemented with standard statistical software [13].

In proposed approach a corpus of Urdu text with 123714 tagged words has been used. Using word-tag frequency the corpus has been converted to bipolar descriptor arrays which in turn are used to train the FFNN. Purpose of preparing bipolar arrays is to reduce the error rate and to speed up the training. After its adequate training we tested the model for different data sets and achieved pleasing results.

3.1. Corpus Description

The corpus [1] which has been used for training and testing this system; contains 123714 tagged words, its vocabulary size is 11924, it has 4288 sentences, and it was tagged using 48 different parts of speech. Obviously this corpus is not adequate enough but it is the only available corpus for Urdu language processing. Moreover, corpus was not standard and there were anomalies of inappropriate spaces, zero width non joiner and word tag order. We removed all these anomalies manually.

We reduced the Urdu tag set as shown in Table 1 to 28 more general tags as shown in Table 2 to reduce the number of input neurons in the model and to speed up training process. For example; Verb (VB), Light Verb (VBL), Infinitive Verb (VBI), Infinitive Light Verb (VBLI) and Verb To be (VBT), have all been grouped under a general category Verb (VB). Tag set and its generalized form both are given in appendix.

<table>
<thead>
<tr>
<th>Sr. No</th>
<th>Tag Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DM</td>
</tr>
<tr>
<td>2</td>
<td>DMRL</td>
</tr>
<tr>
<td>3</td>
<td>NN</td>
</tr>
<tr>
<td>4</td>
<td>NNCM</td>
</tr>
<tr>
<td>5</td>
<td>NNP</td>
</tr>
<tr>
<td>6</td>
<td>NNPC</td>
</tr>
<tr>
<td>7</td>
<td>NNC</td>
</tr>
<tr>
<td>8</td>
<td>NNCR</td>
</tr>
<tr>
<td>9</td>
<td>PR</td>
</tr>
<tr>
<td>10</td>
<td>PRRF</td>
</tr>
<tr>
<td>11</td>
<td>PRRL</td>
</tr>
<tr>
<td>12</td>
<td>PRPS</td>
</tr>
<tr>
<td>13</td>
<td>PRRFPS</td>
</tr>
<tr>
<td>14</td>
<td>VB</td>
</tr>
<tr>
<td>15</td>
<td>VBL</td>
</tr>
<tr>
<td>16</td>
<td>VBI</td>
</tr>
<tr>
<td>17</td>
<td>VBLI</td>
</tr>
<tr>
<td>18</td>
<td>VBT</td>
</tr>
<tr>
<td>19</td>
<td>AUXA</td>
</tr>
<tr>
<td>20</td>
<td>AUXT</td>
</tr>
<tr>
<td>21</td>
<td>JJ</td>
</tr>
<tr>
<td>22</td>
<td>RB</td>
</tr>
<tr>
<td>23</td>
<td>Q</td>
</tr>
<tr>
<td>24</td>
<td>CD</td>
</tr>
<tr>
<td>25</td>
<td>OD</td>
</tr>
<tr>
<td>26</td>
<td>FR</td>
</tr>
<tr>
<td>27</td>
<td>MUL</td>
</tr>
<tr>
<td>28</td>
<td>U</td>
</tr>
<tr>
<td>29</td>
<td>CC</td>
</tr>
<tr>
<td>30</td>
<td>SC</td>
</tr>
<tr>
<td>31</td>
<td>I</td>
</tr>
<tr>
<td>32</td>
<td>ITRP</td>
</tr>
<tr>
<td>33</td>
<td>JJRP</td>
</tr>
<tr>
<td>34</td>
<td>KER</td>
</tr>
<tr>
<td>35</td>
<td>MOPE</td>
</tr>
<tr>
<td>36</td>
<td>MOPO</td>
</tr>
<tr>
<td>37</td>
<td>CM</td>
</tr>
<tr>
<td>38</td>
<td>RBRP</td>
</tr>
<tr>
<td>39</td>
<td>WALA</td>
</tr>
<tr>
<td>40</td>
<td>INJ</td>
</tr>
<tr>
<td>41</td>
<td>QW</td>
</tr>
<tr>
<td>42</td>
<td>SM</td>
</tr>
<tr>
<td>43</td>
<td>PM</td>
</tr>
<tr>
<td>44</td>
<td>DATE</td>
</tr>
<tr>
<td>45</td>
<td>SYM</td>
</tr>
<tr>
<td>46</td>
<td>UNK</td>
</tr>
</tbody>
</table>
An Artificial Neural Network Approach for Sentence Boundary Disambiguation in Urdu Language Text

Table 2. Generalized urdu tag set.

<table>
<thead>
<tr>
<th>S. No</th>
<th>General Tag</th>
<th>Sub Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DM</td>
<td>DM, DMRL</td>
</tr>
<tr>
<td>2</td>
<td>NN</td>
<td>NN, NNCM, NNP, NNPC, NNC, NNCR, PR, PRRF, PRL, PRS, PRRFPs</td>
</tr>
<tr>
<td>3</td>
<td>VB</td>
<td>VB, VBL, VH, VBLJ, VBT</td>
</tr>
<tr>
<td>4</td>
<td>AUX</td>
<td>AUXA, AUNIT</td>
</tr>
<tr>
<td>5</td>
<td>JJ</td>
<td>JJ</td>
</tr>
<tr>
<td>6</td>
<td>RB</td>
<td>RB</td>
</tr>
<tr>
<td>7</td>
<td>Q</td>
<td>Q</td>
</tr>
<tr>
<td>8</td>
<td>CD</td>
<td>CD</td>
</tr>
<tr>
<td>9</td>
<td>OD</td>
<td>OD</td>
</tr>
<tr>
<td>10</td>
<td>FR</td>
<td>FR</td>
</tr>
<tr>
<td>11</td>
<td>MUL</td>
<td>MUL</td>
</tr>
<tr>
<td>12</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>13</td>
<td>CC, SC</td>
<td>CC, SC</td>
</tr>
<tr>
<td>14</td>
<td>I, ITRP</td>
<td>I, ITRP</td>
</tr>
<tr>
<td>15</td>
<td>JRBP</td>
<td>JRBP</td>
</tr>
<tr>
<td>16</td>
<td>KER</td>
<td>KER</td>
</tr>
<tr>
<td>17</td>
<td>MOP</td>
<td>MOPE, MOPO</td>
</tr>
<tr>
<td>18</td>
<td>RBP</td>
<td>RBP</td>
</tr>
<tr>
<td>19</td>
<td>WALA</td>
<td>WALA</td>
</tr>
<tr>
<td>20</td>
<td>INJ</td>
<td>INJ</td>
</tr>
<tr>
<td>21</td>
<td>QW</td>
<td>QW</td>
</tr>
<tr>
<td>22</td>
<td>SM</td>
<td>SM</td>
</tr>
<tr>
<td>23</td>
<td>PM</td>
<td>PM</td>
</tr>
<tr>
<td>24</td>
<td>DATE</td>
<td>DATE</td>
</tr>
<tr>
<td>25</td>
<td>SYM</td>
<td>SYM</td>
</tr>
<tr>
<td>26</td>
<td>UNK</td>
<td>UNK</td>
</tr>
</tbody>
</table>

3.2. ANN Based Sentence Splitting Algorithm for Urdu Text

- Read Urdu tagged corpus.
- For each distinct word in the corpus, calculate word-tag frequency, for each tag in the generalized tag set given in Table 2.
- Convert each frequency into probabilistic value by dividing it with the total number of occurrences of the word, the given frequency is calculated for.
- To form the final descriptor array convert each probability into bipolar form using the universal threshold 0.5, say if probability is less than 0.5, it will be converted to -1. If it is equal to 0.5, it will be transformed to 0 and if it is greater than 0.5, it will be changed to +1.
- Use the descriptor arrays to train the FFNN using back propagation algorithm and delta learning rule.
- After the training of network, pass the test data for sentence boundary disambiguation. If test data is un-tagged, tag it by using unigram statistical model before passing to the network.

3.3. Descriptor Arrays

After removing the anomalies from the corpus we created descriptor arrays for each word in the training corpus. For this purpose, we calculated the frequency of each word regardless of the part of speech tag it is used with. Moreover, the frequency of each word has also been calculated with every part of speech tag individually.

The order of part of speech tags in descriptor array is kept same as shown in the tag set. Consider the example given in Figure 1. Here the word خلاف (against) is used 11 times in whole corpus. Its frequency with all possible tags shows that this word is used 9 times as noun <NN>, 2 times as adjective <JJ> and it is not used with rest of the probable tags.

This frequency is used to find the probabilities, by dividing it with the word count (the no. of times the word appeared in the corpus regardless of its part of speech tag). So, the descriptor array for this word would be as given in Figure 2.

Later on these probabilities are converted to bipolar format (-1, 0, +1) by setting a threshold value, because bipolar inputs are easy to deal with in computation as compared to long floating point values, which requires more complex computations. Hence, it takes relatively less time for computation and convergence and is more accurate. We are using universal threshold 0.5 for this conversion. If the probability is greater than threshold, it is set to +1 and if it is less, then it is set to -1. Similarly if there is a probability equal to threshold, it is set to 0. Figure 3 shows the transformation of the descriptor array from probabilistic to bipolar values, which are used to train the neural network.

3.4. Training

We accomplished the disambiguation of punctuation marks using FFNN trained with back propagation algorithm. It works efficiently on the continuous data as compared to the other training algorithms. Moreover, it can be used in training neural models even with small sized data sets.

As it is a supervised learning algorithm therefore some targets have been set before its training. If a punctuation mark in data is sentence marker, target is set to +1, otherwise target is -1. For the output of each epoch, error value is calculated and propagated back to the hidden layer and input layer. On the basis of this error value, weights are updated and again entire process is repeated. The process continued till the stopping criteria is met.
In this work we used a context window of size 2. That’s why we are considering two neighboring words (preceding and following) of each candidate punctuation mark. We trained ANN on the basis of probabilities calculated by part of speech tags. As tag set contains 28 tags, therefore there are 28 inputs against each word. In case of context size 2 there are 28 × 2 input neurons. Similarly for hidden layer 28 × 2 neurons are selected. Input layer with 56 neurons is fully connected with hidden layer and there are 56 × 56 connections between input and hidden layer. Likewise 56 links are used to connect hidden layer to output layer. At output layer there is only one neuron. All these links carry some weights. Initially random weights have been assigned to links which are further updated during training. Smallest or constant error minimization ratio has been set as stopping criteria for training. Once this criterion is met, training is stopped and final weights of each connection of ANN are stored in a file for further use.

4. Testing

We tested our system both for tagged and untagged data sets. If a file is passed to this system for sentence boundary identification, it checks either if it is tagged or untagged. If a file is untagged then it is passed to a trained statistical tagger (unigram tagger), so that each token in the file could have an appropriate part of speech tag. Once the test file is tagged, all candidate punctuation signs say “.”, “’”, “!” and “?” are tagged as <TST> (candidate punctuation), irrespective of their current status which could be either <SM> (Sentence Marker) or <PM> (Phrase Marker). After tagging the test file, it is passed to ANN which replaces each <TST> tag with <SM> or <PM> according to its learning. The threshold of the ANN, on the basis of which a suitable tag is assigned to candidate punctuation, has been set experimentally.

4.1. Experiments and Results

Results under discussion have been calculated for three different data sets. In each data set available corpus has been divided into two halves; in first data set 50% of the corpus is used as training data for the network and remaining 50% as testing data, in other two data sets 60% and 40%, 70% and 30% are the ratios between training and testing data respectively. Details of the punctuation signs in training and testing data are given in Table 3. Precision, recall and F-measure are the parameters to evaluate the results which are defined as:

\[ \text{Precision} = \frac{\text{Correctly identified sentence markers}}{\text{Total number of sentence markers identified by the system}} \times 100 \]  
\[ \text{Recall} = \frac{\text{Correctly identified sentence markers}}{\text{Total number of sentence markers in the test data}} \times 100 \]  
\[ F\text{-Measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \times 100 \]

<table>
<thead>
<tr>
<th>Data Sets</th>
<th>Number of Tokens</th>
<th>Period</th>
<th>Question Mark</th>
<th>Exclamation Sign</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>61857</td>
<td>209</td>
<td>204</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>74228</td>
<td>2485</td>
<td>209</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>74228</td>
<td>2485</td>
<td>209</td>
<td>30</td>
</tr>
<tr>
<td>Testing</td>
<td>86599</td>
<td>2915</td>
<td>253</td>
<td>37</td>
</tr>
<tr>
<td></td>
<td>49486</td>
<td>1718</td>
<td>213</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>37115</td>
<td>1288</td>
<td>187</td>
<td>8</td>
</tr>
</tbody>
</table>

In first experiment corpus is divided into two equal halves containing 61857 tagged words in each, first half has 2131 sentences and the other half has 2157 sentences. The first half of the corpus has been used to train the ANN and after its training it has been tested for the same. The results for this seen data set are given in the graph as shown in Figure 4.

Later on trained neural network is tested for the unseen data (the remaining 50% of the data) and results are given in Figure 5. In second experiment we divided the corpus into two halves; first halve containing 74228 tagged words and 2530 sentences and the other half containing 49486 tagged words and 1758 sentences. Results obtained on training and testing the system with these 2530 sentences are shown in the Figure 6, whereas Figure 7 shows the results of this system for unseen 1758 sentences. The two halves used in third experiment have 86599 words with 2971 sentences and 37115 words with 1317 sentences respectively.
In third experiment the results of training and testing the system with same 2971 sentences and different 1317 sentences, are given in Figures 8 and 9 respectively.

It is obvious from all the results that 0.1 is the most suitable threshold value for our system and it is set experimentally. Initially we calculated the results for fixed threshold value 0.5 but as its obvious from the above graphs that on universal threshold there is fluctuation in the results that’s why we decided to choose the threshold values experimentally between 0.1 and 0.9. While comparing this approach to the one available for Urdu sentence boundary identification [11], it is found that the average recall of our experiments which is 97.29% is better than the average recall of the existing approach which is 86.77%. Although, the average precision we achieved 90.15%, is less than the best achieved precision 95.35% of the existing approach, (these are the results when training and testing data is kept different from each other), but the major advantage of this approach is that it is adaptive and independent of the hand built rules.

5. Conclusions

Sentence boundary disambiguation is an obscure task, especially for Arabic script languages, which lack the character case discrimination. We used FFNN trained with part of speech probabilities calculated from Urdu text corpus, for Urdu sentence boundary disambiguation. ANN is a better approach if it is required to attain better results from a system which is trained with a small sized lexicon. Moreover, as we reduced the input vector to generalized tags and converted the input data into bipolar values, it made it less time consuming as compared to if we would tried to keep all specialized tags and input vector values in continuous form. We attained the bipolar input vectors after calculating the part of speech probabilities for each distinct word in the corpus. These bipolar values have been used by back propagation learning algorithm to train the network. The trained network produced good results, when it is tested for different data sets, with varying threshold value.

References


**Shazia Raj** received her BS degree in computer science from COMSATS Institute of Information Technology, Pakistan, in 2012. Currently, she is an MS scholar at the same institute. Her area of research is natural language processing.

**Zobia Rehman** received her MS degree in computer science from COMSATS Institute of Information technology, Pakistan in 2009. Currently, she is a PhD scholar at Lucian Blaga University of Sibiu, Romania. Her Area of research is natural language processing.

**Sonia Rauf** received her MS degree in computer science from COMSATS Institute of Information Technology, Pakistan in 2010. Currently, she is a lecturer at the same institute. Her Areas of research are artificial intelligence, machine learning and medical image processing.

**Rehana Siddique** received her BS degree in computer science from COMSATS Institute of Information Technology, Pakistan, in 2012. Her area of research is natural language processing.

**Waqas Anwar** received his PhD degree in computer science from Harbin Institute of Technology Harbin, China in 2008. Currently, he is an Associate Professor in the department of Computer Science at COMSATS Institute of Information Technology, Pakistan. His areas of research are natural language processing and computational intelligence.