Urdu Named Entity Recognition System using Hidden Markov Model

Muhammad Kamran Malik, Syed Mansoor Sarwar

1. Punjab College of Information Technology, University of the Punjab Lahore, Pakistan

*Corresponding Author: E-mail: kamran.malik@pucit.edu.pk

Abstract

Named Entity Recognition (NER) is the process of identifying names of Persons, Organizations, Locations and other miscellaneous information like number, date, and measure in a given text. In this paper, we describe the development of a NER system for Urdu Language using Hidden Markov Model (HMM). First, we show a comparison of IOB2 and IOE2 tagging schemes. Second, we show the preprocessing of Urdu before feeding data to the HMM model for training using the IOE2 tagging scheme. Finally, we use the Part of Speech (POS) information, gazetteers, and rules to improve the accuracy of the system. Our system yields 66.71%, 71.70%, and 69.12% as the values for precision, recall, and f-measure, respectively. This system will help us improve the results of Urdu Information Retrieval, Machine Translation, and Questioning and Answering systems.

Key Words: Statistical NER, Indian language NER, Resource poor language, Language independent NER, Urdu NER, Urdu HMM

1. Introduction

Urdu is the national language of Pakistan and written from right to left using the Arabic script, and 69.1 million people worldwide speak Urdu [19]. The task of a Named Entity Recognition and Classification (NERC) system is to identify proper names from the given text and then classify them into person, organization, and location names. Sometimes the process of identifying time, date, money, and percent expression can also be considered as part of the NERC system. A NERC system has many applications, including intelligent Information Retrieval (IR), Machine Translation (MT), and Questioning and Answering (Q&A) systems.

We face the following challenges in Urdu for the development of Urdu NER system [3]:

1. There is no concept of capitalization in Urdu, which is a major clue for NEs in English.
2. Urdu is free word-order language.
3. Urdu is agglutinative in nature.
4. In Urdu, sometimes words are written with diacritic and sometimes without diacritic, which causes multiple variations of a single word.
5. Urdu contains words of different languages, including Arabic, Persian, Sanskrit, and English.
6. In Urdu, very less work has been done from computational perspective.
7. In Urdu, there is an issue of word segmentation.

8. There is also the problem of lack of standardization and spelling variations in Urdu.
9. In Urdu, many words, depending on their context, can be considered as common nouns as well as proper nouns (i.e., candidate for NE). For example, Shan, Kamran, Fazal, Kiran, Aftab, Manzoor, etc can be NEs, i.e., Person can be considered as common noun. The context may help in identifying proper nouns against common nouns but due to no concept of capitalization in Urdu, disambiguation becomes harder than English.
10. There is serious lack of labeled data in Urdu for machine learning.
11. There is a huge variation in writing numbers in Urdu.

In our Urdu NERC system, we first perform experiments using two different types of tagging schemes, i.e., Inside-Outside-Begin (IOB2) and Inside-Outside-End (IOE2). The main purpose of using these two tagging schemes for experimentation is that IOB2 is considered suitable for prepositional languages like English and IOE2 is considered suitable for postpositional languages like Japanese. Urdu is also a postpositional language; that’s why we compare the results of the two tagging schemes and results show that IOE2 produces better results than IOB2. We use character level, word level normalization, Part of Speech, and Regular expressions to improve accuracies.
2. Literature Review

The NERC system has been developed for different languages including Chinese, Dutch, English, French, German, Hindi, and Italian using supervised learning algorithms, semi-supervised learning algorithms, unsupervised algorithms, and hand crafted rules as discussed in [5]. Little work has been done on Urdu NER compared to other languages. Focused work on Urdu NER started after International Joint Conference on Natural Language Processing (IJCNLP)-08 [20] where five Indian languages were targeted for study: Bengali, Hindi, Oriya, Telugu, and Urdu.

[10] describes development of NER systems for the Urdu, Hindi, Bengali, Telugu, and Oriya languages using IJCNLP workshop NER data. Language specific rules and Maximum Entropy (ME) approach along with gazetteers are used to develop NER systems for these languages. Experiment was conducted on 12 types of NEs and the overall accuracy for the Hindi, Bengali, Oriya, Telugu, and Urdu NER systems in terms of f-measure were 65.13%, 65.96%, 44.65%, 18.74%, and 35.47% respectively.

[15] used Beaker-Riaz corpus for the development of a rule based NER system for Urdu. Hand crafted rules were developed due to the availability of a limited amount of annotated corpus for training and testing. They used 200 documents for the construction of rules for the identification of NEs like Person, Designation, Location, Date, Number, and Organization. These rules were tested on 2262 documents that contained 206 unique NEs. By using this approach, 187 NEs were extracted out of which 171 were true NEs with 90.7% recall, 91.5% precision and 91.1% f-measure. The same approach was also used in the IJCNLP 2008 NER workshop to achieve 72.5% f-measure without tuning and 81.6% f-measure with tuning.

[16] also used a rule for the extraction of numbers, non-numeral numbers, date, and time. For the identification of Person, Location, and terms, it used suffix matching along with gazetteer. Two datasets were used with 12032 and 150243 tokens, 12 NEs were used and accuracy in terms of f-measure for dataset 1 and dataset 2 were 60.09% and 88.1%, respectively.

[6] proposes a bootstrapped model for Urdu. This model has four levels of text processing. Conditional Random Field (CRF) is used for POS tagging and NE tagging. It uses three NEs for experimentation, i.e., Person, Organization, and Location. The f-measure of the two-stage model and four-stage model were 55.3% and 68.9%, respectively. The paper also describes these two-stage and four-stage models.

[9] used the n-gram models, i.e., Unigram and Bigram models, with different smoothing techniques for the development of an Urdu NERC system. Five NEs were used for experimentation, i.e., Person, Organization, Location, Date, and Time. The highest accuracy was achieved using the bigram model along with Backoff smoothing technique with 66.2% precision, 88.18% recall, and 75.83% f-measure.

[13] highlights the challenges in the development of an Urdu NER system. Urdu and other South Asian languages are discussed in detail in this regard.

Several experiments have been conducted on the IJCNLP workshop Urdu NE data. [7] used the ME approach to build an Urdu NERC system with 37.58% precision, 33.58% recall, and 25.47% f-measure. [12] applied CRF for the development of an Urdu NERC system with 48.96% precision, 39.07% recall, and 43.46% f-measure. [8] used HMM and rules to build an Urdu NERC system with 56.21% precision, 37.15% recall, and 44.73% f-measure. [14] used CRF for the training and testing of an Urdu NERC system. The system yields 54.45% precision, 26.36% recall, and 35.52% f-measure. [17] used CRF on an Urdu language on only three NEs, i.e., person, organization, and location. The reported results are 64.11% precision, 66.98% recall, and 65.51% f-measure.

3. Tagging Problem

NER can be considered as a sequence labeling problems where we want to determine a vector $Z = \{z_0, z_1, ..., z_T\}$ of random variables given an observed vector $\bar{X} = \{x_0, x_1, ..., x_T\}$. Each variable $z_s \in Z$ can be NE of the word at position $s$, and $x_s \in X$ is the word at positions.

Now we define the tagging problem for finding the most probable NE sequence $Z_{1:n}$ for the word sequence $X_{1:n}$. More formally,

$$\arg\max_{Z_{1:n}} P(Z_{1:n} | X_{1:n})$$

(1)

3.1 Hidden Markov Model (HMM)

In HMM [4] we have two set of states and a triple $(\pi, A, B)$.

First is a set of observable states that is the input sentence or word sequence $X = \{x_1, x_2, ..., x_n\}$ such that $X \in X_z$ with $x_i$ be the $i$th word in $X$.

Second is the set of hidden states that is
represented by $\text{NE} \ z_{1:n}$ for the word sequence $x_{1:n}$ with $z_i$ be the $i^{th}$ NE in the sequence. Each NE represents one of the hidden states in HMM. The observable states (the word sequence) are probabilistically related (emission probabilities) to the hidden states (NE sequence) such that the sum of probabilities of all links outgoing from a single hidden state to all observable states is 1. In triple $(\pi, A, B)$, we define $\pi$ as the initialization vector containing the initial probabilities of all NEs $z_i$ starting an NE sequence. We define $A$ as a matrix of probabilities (transition or Prior Probabilities) when the underlying Markov Process transitions from one state (NE) to another. We define $B$ as a matrix of probabilities (emission or Likelihood probabilities) of generating the word sequence $x_{1:n}$ from the underlying NE sequence $z_{1:n}$, i.e., the probability of generating or emitting a word $x_i$ once the underlying Markov Process enters a state $z_i$. The triple $(\pi, A, B)$ is learnt from our Urdu NE training data.

HMM defines the joint probability distribution over a word sequence paired with an NE sequence as

$$P(x_{1:n}, z_{1:n})$$  \(\text{(2)}\)

The output of HMM is a tag sequence that maximizes this joint probability distribution

$$\arg\max_{z_{1:n} \in z} P(x_{1:n}, z_{1:n})$$  \(\text{(3)}\)

To model this joint probability we consider our basic NE problem from Equation (1) as

$$\arg\max_{z_{1:n} \in z} P(z_{1:n}|x_{1:n})$$  \(\text{(4)}\)

Bayes Rule of probability dictates us that we can calculate the probability of $(z_{1:n}|x_{1:n})$ if we know the probability of $(x_{1:n}|z_{1:n})$ and it says

$$P(z_{1:n}|x_{1:n}) = \frac{P(x_{1:n}|z_{1:n})P(z_{1:n})}{P(x_{1:n})}$$  \(\text{(5)}\)

By applying Bayes Rule to Equation 3 we get

$$\arg\max_{z_{1:n} \in z} \frac{P(z_{1:n})P(x_{1:n}|z_{1:n})}{P(x_{1:n})}$$  \(\text{(6)}\)

We drop the denominator for being a constant for all NEs and hence Equation 6 becomes

$$\arg\max_{z_{1:n} \in z} P(z_{1:n})P(x_{1:n}|z_{1:n})$$  \(\text{(7)}\)

This means that for each NE sequence we need to calculate the product of likelihood $P(x_{1:n}|z_{1:n})$ and prior probability $P(z_{1:n})$. We make two simplifying assumptions to estimate the probability of the NE sequence. First assumption says that the probability of a word is dependent only on its own underlying NE.

$$P(x_{1:n}|z_{1:n}) \approx \prod_{i=1}^{n} P(x_i|z_i)$$  \(\text{(8)}\)

Since we have used both the Bigram and Trigram HMM to formulate our results, therefore, for Bigram HMM we assume that the probability of NE is dependent only on the previous NE (First Order Markov Assumption). Thus, $P(z_{1:n})$ is expressed as shown below.

$$P(z_{1:n}) \approx \prod_{i=1}^{n} P(z_i|z_{i-1})$$  \(\text{(9)}\)

For Trigram HMM we assume that the probability of NE is dependent only on previous two NEs (Second Order Markov Assumption). Thus, Equation (9) may be expressed as given below.

$$P(z_{1:n}) \approx \prod_{i=1}^{n} P(z_i|z_{i-2}, z_{i-1})$$  \(\text{(10)}\)

After these two assumptions, we can rewrite Equation (2) as

$$P(x_{1:n}, z_{1:n}) \approx \prod_{i=1}^{n} P(z_i|z_{i-1}) \prod_{i=1}^{n} P(x_i|z_i)$$  \(\text{(11)}\)

Where $P(z_{i-1}|z_{i-2})$ and $P(z_i|z_{i-2}, z_{i-1})$ are called the Bigram and Trigram parameters, respectively, and $P(x_i|z_i)$ is called the emission parameter of HMM. Some of the details are also mentioned in [3].

4. Hybrid Approach For Urdu NER System

For our experiments, we have used training and testing data from IJCNLP workshop. The training data consisted of 2584 NEs, 35447 tokens, and 1508 sentences, and the testing data consisted of 1027 NEs, 12805 tokens, and 498 sentences. Details of the training and testing data are given in Table 1.
We used two tagging schemes, i.e., IOB2 and IOE2, on 12 NEs for experimentation. First, we converted the Urdu NE data from the SSF format to the IOB2 and IOE2 formats. We used HMM for training and testing of Urdu NER. The precision, recall, and f-measure values using IOB2 are 38.38%, 54.04%, and 44.89%, respectively. Using IOB2 no entry for NEB, NETE, NETO, and NETP is found, whereas the highest f-measure value of NEM is found. Similarly, the precision, recall, and f-measure using IOE2 are 39.00%, 54.65%, and 45.52%, respectively. No word is correctly classified against NEA, NEB, NETE, NETO, and NETP. Detailed results of IOB2, IOE2, and comparison of each NE against IOB2 and IOE2 is shown in Table 2, Table 3, and Figure 1, respectively.

Table 1: Details of Urdu Corpus

<table>
<thead>
<tr>
<th>Named Entity</th>
<th>Training Data (tokens)</th>
<th>Testing Data (tokens)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEP (Person)</td>
<td>365</td>
<td>145</td>
</tr>
<tr>
<td>NED (Designation)</td>
<td>98</td>
<td>41</td>
</tr>
<tr>
<td>NEO (Organization)</td>
<td>155</td>
<td>40</td>
</tr>
<tr>
<td>NEA (Abbreviation)</td>
<td>39</td>
<td>3</td>
</tr>
<tr>
<td>NEB (Brand)</td>
<td>9</td>
<td>18</td>
</tr>
<tr>
<td>NETP (Title-Person)</td>
<td>36</td>
<td>15</td>
</tr>
<tr>
<td>NETO (Title-Object)</td>
<td>4</td>
<td>147</td>
</tr>
<tr>
<td>NEL (Location)</td>
<td>1118</td>
<td>468</td>
</tr>
<tr>
<td>NETI (Time)</td>
<td>279</td>
<td>59</td>
</tr>
<tr>
<td>NEN (Number)</td>
<td>310</td>
<td>47</td>
</tr>
<tr>
<td>NEM (Measure)</td>
<td>140</td>
<td>40</td>
</tr>
<tr>
<td>NETE (Terms)</td>
<td>30</td>
<td>4</td>
</tr>
<tr>
<td>Total NEs</td>
<td>2584</td>
<td>1027</td>
</tr>
<tr>
<td>Total Words</td>
<td>35447</td>
<td>12805</td>
</tr>
<tr>
<td>Total Sentences</td>
<td>1508</td>
<td>498</td>
</tr>
</tbody>
</table>

We used IOE2 for further experimentation. We normalized the whole input and then trained it using HMM. We performed two types of normalization:

1. Character level normalization
2. Word level normalization

In the character level normalization we convert different equivalent forms of a character to a standard form of character. In Urdu some of the characters that are visually the same can be written in different ways. For example، 

\[ \text{آگرہ} (Agrah), \text{آگرہ} (Agrah) \]

may be assigned different NEs. For the purpose of Normalization, we use the Center for Language Engineering (CLE) [20] utility for Urdu character level normalization using the NFC implementation.

Table 2: Results of using IOB2 tagging scheme

<table>
<thead>
<tr>
<th>NE</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEA</td>
<td>50</td>
<td>28.57</td>
<td>36.36</td>
</tr>
<tr>
<td>NED</td>
<td>11.76</td>
<td>8</td>
<td>9.5</td>
</tr>
<tr>
<td>NEL</td>
<td>54.1</td>
<td>67.57</td>
<td>60.01</td>
</tr>
<tr>
<td>NEM</td>
<td>85.26</td>
<td>80.20</td>
<td>82.65</td>
</tr>
<tr>
<td>NEN</td>
<td>8.57</td>
<td>8.1</td>
<td>8.33</td>
</tr>
<tr>
<td>NER</td>
<td>32.01</td>
<td>23.78</td>
<td>27.31</td>
</tr>
<tr>
<td>NETE</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>NETO</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>NETP</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Overall</td>
<td>38.38</td>
<td>54.04</td>
<td>44.89</td>
</tr>
</tbody>
</table>

Table 3: Results of Using IOE2 tagging scheme

<table>
<thead>
<tr>
<th>NE</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEA</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>NED</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>NEL</td>
<td>54.77</td>
<td>68.84</td>
<td>61.01</td>
</tr>
<tr>
<td>NEM</td>
<td>87.37</td>
<td>78.30</td>
<td>82.59</td>
</tr>
<tr>
<td>NEN</td>
<td>14.29</td>
<td>14.71</td>
<td>14.49</td>
</tr>
<tr>
<td>NER</td>
<td>42.45</td>
<td>28.13</td>
<td>33.83</td>
</tr>
<tr>
<td>NETE</td>
<td>24.64</td>
<td>48.15</td>
<td>32.60</td>
</tr>
<tr>
<td>NETO</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>NETP</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Total</td>
<td>39.00</td>
<td>54.65</td>
<td>45.52</td>
</tr>
</tbody>
</table>

In Urdu users may write a single word with different sequence of characters like Lahore can be
written as لاهور or لاہور. This is an example of word normalization where we need to standardize such words into a single form for correct learning of HMM. We perform word level normalization by using four steps, as described below:

4.1 Assignment of Part of Speech (POS) Tags

The POS information for each word is useful to improve the results of a NER system. For this purpose, we assign the POS tags to the training and testing data by using an online POS tool available at http://cle.org.pk/index.htm, which uses the POS tagset of [11]. The output of this tool is not up to the mark with respect to identifying Proper Noun (NNP) because the location name Lahore can be written using two ways لاهور and لاہور, as stated above, but this tool assigns the NN tag to the first word and NNP tag to second word.

4.2 Transliteration of Each Word

The transliteration task is carried out using the tool mentioned in [18]. This tool converts Urdu in Arabic script into Roman Urdu. For example, Roman Urdu of لاهور and لاہور is Lahore and, similarly, Roman Urdu of شملہ and شملا are Shimlah and Shimla, respectively. We take all Nouns and Proper Nouns as input and generate their transliteration.

4.3 SOUNDEX Code Generation

This module takes Roman Urdu form of all Nouns and Proper Nouns as input and generates their SOUNDEX codes. For example, the SOUNDEX code generated for Shimlah and Shamila is S540.

4.4 Conversion into a Standard Word

In this module words with different spelling variations are converted into one form on the basis of highest frequency.

<table>
<thead>
<tr>
<th>NE</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEA</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>NEB</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>NED</td>
<td>11.76</td>
<td>9.09</td>
<td>10.26</td>
</tr>
<tr>
<td>NEL</td>
<td>63.00</td>
<td>72.28</td>
<td>67.32</td>
</tr>
<tr>
<td>NEM</td>
<td>87.37</td>
<td>79.05</td>
<td>83.00</td>
</tr>
<tr>
<td>NEN</td>
<td>14.29</td>
<td>14.71</td>
<td>14.49</td>
</tr>
<tr>
<td>NEO</td>
<td>45.28</td>
<td>30.00</td>
<td>36.09</td>
</tr>
<tr>
<td>NEP</td>
<td>32.70</td>
<td>55.20</td>
<td>41.07</td>
</tr>
<tr>
<td>NETI</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>NETO</td>
<td>74.63</td>
<td>45.05</td>
<td>56.18</td>
</tr>
<tr>
<td>NETP</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Total</td>
<td>43.91</td>
<td>57.94</td>
<td>49.96</td>
</tr>
</tbody>
</table>

The last experiment is performed using some rules, i.e., regular expressions of NEN, NETI, NEM and NEA. Also, a small Gazetteer of person name, location name, organization name, Terms, Title Persons, and Title object is prepared using the Internet. The partial details of these rules are given below.

For NEN (number) we define regular expressions like Urdu_Digit = ایک (aik) (one) | تو (to) (two) | تین (theen) (three) | جار (chari) (four) | پانچ (paunch) (five) | چھ (chhey) (six) | سات (sataat) (seven) | آٹھ (aath) (eight) | نو (nayo) (nine) | گیارہ (gaiarah) (eleven) | تیرہ (barah) (twelve) | نیم (nanteh) (thirteen) | چودہ (chodah) (fourteen) | پندرہ (pandrah) (fifteen) | سولہ (solah) (sixteen) | سترہ (satrak) (seventeen) | آٹارہ (aturah) (eighteen) | اکسیس (akees) (twenty) | ہفتین (hafteen) (twenty one) | بیس (baees) (twenty two) | تیس (tayees) (twenty three) | چوبیس (choobees) (twenty four) | چوتیس (chotees) (twenty five) | سونتے (santayees) (santaees) (twenty six) | ساتائیس (satiaaees) (twenty seven) | تئیس (teees) (thirty) | تئینے (taynees) (thirty one) | بیس (bates) (thirty two) | چوئینے (choonees) (thirty three) | چوئینے (choonees) (thirty four) | پندرہ (pandrah) (thirty five) | سئینے (santees) (thirty six) | نینے (nantees) (thirty seven) | نینے (nantees) (thirty eight) |
For NETI (Time), we define regular expressions like $U_{\text{Digit}} = (\text{ayik})$, $M_{\text{Digit}} = [1-31]$

For NEM (measure), we define regular expressions like $\text{Units} = \text{کلومیٹر}$ (kilometer), $\text{سیلت}$ (ton), $\text{میگاوات}$ (mega watt), $\text{سافار}$ (safar), $\text{مارچ}$ (March)
Some of examples are 4 (4 say 9 saal) (4 to 9 years), 50 (50 say 50 kilometer fi ghunta) (40 to 50 kilometer per hour), aik meel (one mile), aik ye do meel (one or two), safar (one one inch) (60 to 100 foot) (60 to 100 feet), aik aik inch (one inch)

For NEA (abbreviation) we define regular expression like Abb = [A] | [B] | [C] | [D] | [E] | [F] | [G] | [H] | [I] | [J] | [K] | [L] | [M] | [N] | [O] | [P] | [Q] | [R] | [S] | [T] | [U] | [V] | [W] | [X] | [Y] | [Z]

NEA = Abb Abb+

Table 5: Results of Using IOE2 tagging scheme after executing normalization step.

<table>
<thead>
<tr>
<th>NE</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEA</td>
<td>100.00</td>
<td>58.33</td>
<td>73.68</td>
</tr>
<tr>
<td>NEB</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>NED</td>
<td>89.47</td>
<td>44.74</td>
<td>59.65</td>
</tr>
<tr>
<td>NEL</td>
<td>88.27</td>
<td>77.84</td>
<td>82.73</td>
</tr>
<tr>
<td>NEM</td>
<td>100.00</td>
<td>89.91</td>
<td>94.69</td>
</tr>
<tr>
<td>NEN</td>
<td>86.00</td>
<td>81.13</td>
<td>83.50</td>
</tr>
<tr>
<td>NEO</td>
<td>63.21</td>
<td>37.64</td>
<td>47.18</td>
</tr>
<tr>
<td>NEP</td>
<td>63.55</td>
<td>73.12</td>
<td>68.00</td>
</tr>
<tr>
<td>NETE</td>
<td>14.29</td>
<td>14.29</td>
<td>14.29</td>
</tr>
<tr>
<td>NETI</td>
<td>95.24</td>
<td>76.92</td>
<td>85.11</td>
</tr>
<tr>
<td>NETO</td>
<td>2.34</td>
<td>100.00</td>
<td>4.58</td>
</tr>
<tr>
<td>NETP</td>
<td>54.55</td>
<td>75.00</td>
<td>63.16</td>
</tr>
<tr>
<td>Total</td>
<td>66.71</td>
<td>71.70</td>
<td>69.12</td>
</tr>
</tbody>
</table>

By using the above rules, names list, and the POS information, the improved precision, recall, and f-measure are achieved as 66.71%, 71.70%, and 69.12%, respectively. The detailed results are given in Table 5. For example, aik meel (one mile) can represent digit 2 or it can represent word “give” that is a verb. We use the POS information to correctly assign an NE to the word aik meel (do). Similarly, aik aik inch (one inch) can represent number 100 or it can represent word “sleep” that is a verb. Thus, we use the POS information to correctly assign the NE to the word aik aik inch (do). Likewise, safar (one inch) can represent digit 0 or it can represent a month in the Islamic calendar. Hence, we use the POS information to correctly assign an NE to the word safar (safar). The same case with the word budh (budh) that could be a day of the week, i.e., Wednesday or it could be the name of person, i.e., Mahatma Budh (Buddhah). Again, we distinguish between the two with the help of the POS information.

5. Conclusion and Future Work

We are not aware of any research in which experimentation on Urdu NER has been performed using the character and word level normalizations, POS information, gazetteers, and rules. Literature review shows that our system has produced the best-known results using 12 NEs of IJCNLP workshop data. To the best of our knowledge, no one has previously carried out experimentation on Urdu data with IOE2 tagging scheme. In future, we can use Urdu NP chunker by [1, 2], another tagging scheme mentioned in [3], CRF, and deep learning approaches to show how these techniques affect results.

References


Maximum Entropy Hindi Named Entity Recognition. In IJCNLP (pp. 343-349).


