A Hybrid Approach for Urdu Sentence Boundary Disambiguation

Zobia Rehman and Waqas Anwar
Department of Computer Science, COMSATS Institute of Information Technology, Pakistan

Abstract: *Sentence boundary identification is a preliminary step for preparing a text document for Natural Language Processing tasks, e.g., machine translation, POS tagging, text summarization and etc. We present a hybrid approach for Urdu sentence boundary disambiguation comprising of unigram statistical model and rule based algorithm. After implementing this approach, we obtained 99.48% precision, 86.35% recall and 92.45% F1-Measure while keeping training and testing data different from each other, and with same training and testing data, we obtained 99.36% precision, 96.45% recall and 97.89% F1-Measure.*

Keywords: *Sentence boundary disambiguation, unigram model.*

Received October 19, 2009; accepted May 20, 2010

1. Introduction

Sentence boundary detection is a preliminary task for numerous NLP applications, e.g., information retrieval, information extraction, part of speech tagging, machine translation, chunking, parsing, and text summarization. All these tasks require their input text to be alienated into sentences for further processing. But it is difficult to properly alienate text into sentences, as there are numerous ambiguity issues as Urdu script is derived from Persian and Arabic [7].

Sentence terminating punctuations, e.g., '.', '?', and '!' often appear inside the sentence, e.g., period can appear inside the sentence, such as, part of abbreviation, decimal between numbers and line breaker. So it is nontrivial to decide about a punctuation that either it is a sentence marker or not and it becomes more complex in the languages like Urdu where there is no discrimination between upper and lower case characters. Consider the given examples:

یو۔ ایس۔ اے۔ کی معیشت پچھلےدو۔ تین سالوں میں بہت متاثر ہوئ ۔

نہیں! بچے ایسا کیوں کرتے؟

اسنےکہا،"مجھےجانےدو۔"

مسٹرجی-بیل نے یو-کے- میں جنوری- اکتوبرقیام کیا-

آگ نے بہت سے لوگوں کو اپنی لپیٹ میں لے لیا۔ ۔ ۔ ۔

Obviously in above cases it is difficult for machine to isolate the punctuations behaving like sentence marker. In this paper a hybrid technique is presented that combines the unigram statistical model and the algorithm comprising of rules. Unigram model has been trained over tagged data. After testing, this model shows very low precision and high error rate, as all the periods are tagged as sentence markers. To improve precision, some rules are formulated. These rules are based on the tag of the word preceding the punctuation mark that is contestant of being a sentence marker. After implementing these rules we got significant improvement in our results. This work has shown 99.36% precision, 96.45% recall, and 97.89% F1-measure while keeping the same training and testing data, whereas 99.48% precision, 86.35% recall, and 92.45% F1-Measure on keeping training and testing data different from each other.

The paper is organized as follows, section 2 describes the related work, section 3 describes proposed strategy, results and discussions are given in section 4, and section 5 gives conclusion of the work.

2. Related Work

2.1. Sentence Splitting Techniques Used for Various Languages of the World Except Urdu

The task of sentence boundary disambiguation is performed for numerous languages. Although few of them are Arabic script languages, written from right to left, but still no work has been found for Urdu sentence boundary disambiguation.

Various techniques have been used for different languages, e.g., rule based techniques, collocation identification [5, 6] verb and inflection detection [9], machine learning techniques, regular expressions [16], heuristic rules, artificial neural network models [15], part of speech tagging [8] and maximum entropy [11].

In [12] Reynar and Ratnaparkhi used maximum entropy approach for sentence boundary detection in raw text. They trained their system on wall street

journal and brown corpora and achieved 98.0% and 98.5% accuracy respectively. In Bondec [17] rule based approach, maximum entropy and HMM were used. Wall Street Journal was taken as training and testing corpus. All three techniques worked independent of each other and gave 16.25%, 10.00%, and 1.99% error rate respectively. In [1] maximum entropy along with set of rules was used. It worked on Wall Street Journal, Brown, and GENIA corpora and obtained 98.8%, 96.2%, and 98.2% accuracy respectively. This system was also trained for tagged data but improvement was negligible. In [4] syllabication information and phonetic rules were used for Turkish sentence boundary disambiguation and got 96.02% accuracy. In [10] Palmer and Hearts used feed forward neural network. This network worked on tagged corpora along with probabilistic POS information and produced 98.5% accuracy for English text. In [13] for Persian, Kurdish and Arabic texts finite state model was used to disambiguate the sentence marker punctuations from the punctuations used in dates, numbers, acronyms and abbreviations. In [12] rules were formulated regarding the position of period, verb, and proper noun in the sentence. Capitalization was also considered while developing these rules. On 2435 sentences it gave about 79.5% recall. [10] Was based on rules. These rules were based on the location of period in the text and on the type of its preceding punctuation and following punctuation. As well as these rules were based on the length of period's preceding and following words also. Text was divided in to tokens and candidate punctuations were identified. These rules produced 99.4% accuracy for Greek text containing 8736 number of sentences.

2.2. Sentence Splitting Techniques for Urdu Language

Unluckily till now no efforts have been found for Urdu sentence boundary identification. This work is equally important for Urdu text processing but efforts are missing for it. Urdu is a bi-directional language and it uses Arabic orthography Unicode standards. All currently available operating systems do not provide support for such languages. Urdu language processing also needs bi-directional text supporting IDE and text editors. Moreover there are programming language limitations as many programming languages do not have any support for Unicode text processing [14]. There are limitations of training and testing data also. Till now comprehensive tagged and plain corpora for Urdu language are not seen.

3. Proposed Approach

A hybrid approach is presented for identifying sentence boundaries in Urdu text. This approach combines unigram statistical model [2, 3] and the rule based algorithm. Tagged data has been used to train unigram model. This trained model has been used to identify word boundaries in test data. Once the test data is tagged it is seen that every '-' has been tagged as sentence boundary, neglecting that either it is a boundary marker or not (It happened because our training corpus contains a very small amount of data. Urdu language processing is in early stages and till now no such corpus has been developed having size like Brown or Wall Street Journal). Thus unigram model produced very low precision. To solve the problem of low precision in test data we have formulated some rules. These rules are based on the tag assigned to the word preceding the period or any other punctuation behaving like a sentence boundary. The algorithm has been applied on the data tagged by unigram tagger for identifying all putative sentence boundaries in it. Using POS information of the words preceding the candidate sentence boundaries, some of the potential boundaries have been selected as correct sentence markers, while all the rest have been disqualified.

3.1. Algorithm

- Ascertain '-', '؟', '!' and '"' in the tagged data.
- Mark them as sentence periphery by replacing their tags with <B>.
- Make out <B> in the tagged corpus.
- Read tag t of its preceding word.
- If t ∈ {<NN>, <NNPC>, <NNC>, <JJ>, <CM>, <NNP>, <SC>, <PM>, <NNCM>, <CC>, <Q>, <FR>, <DATE>, <CD>}.
- Debar <B> as boundary and replace it with <PM>.
- Rests of the punctuations ending with <B> are actual sentence markers.

3.2. Illustration

According to Figure 1 firstly algorithm reads the data to identify the punctuations '-', '؟', '!' and '"' in it. Further it replaces their tags with tag <B>. Tag <B> shows that tagged punctuation is a putative sentence boundary. Further it reads preceding tag of each putative sentence boundary and compares it with set T. Set T contains all those tags extracted from tagged corpora that if a putative sentence boundary is preceded by any one of them, will be disqualified. Remaining are actual sentence peripheries.

4. Results and Discussions

4.1. Properties of the Training and Testing Data

Five different corpora files have been used to train and test the unigram statistical model for close test result evaluation. In open test file 5 (divided into eight different sets) has been used for training the system

and file 4 has been used as test data. Specifications of these files are given in Tables 1, 2, 3, 4, and 5.

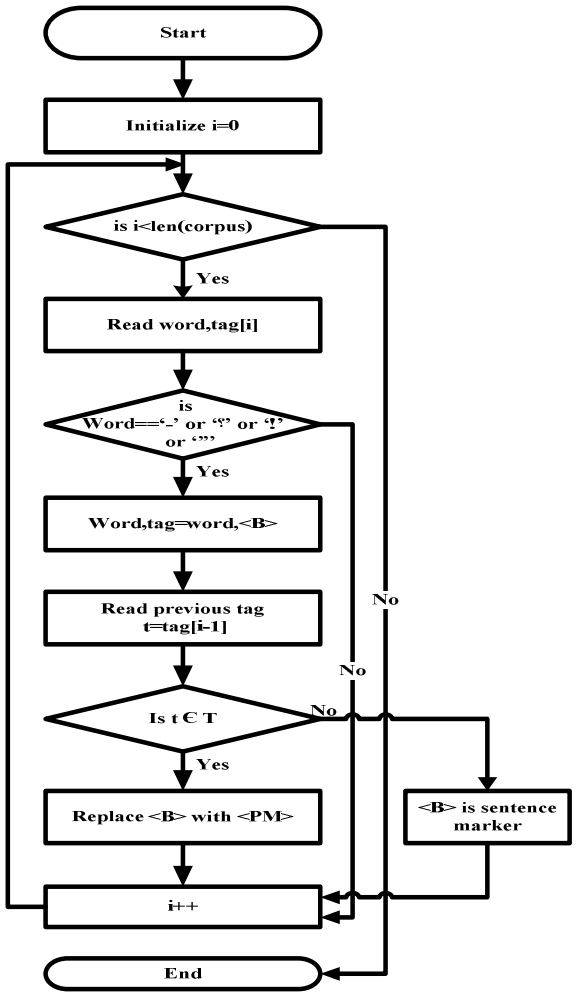

Figure 1. FD for proposed algorithm.

Table 1. Use of period in training and testing data.

No. of Tokens	Periods (Sentence Markers)	Periods (Not Sentence Markers)	Total No. of Periods
12808	453	155	608
55034	1852	649	2501
56970	1924	449	2373
12481	437	117	554
124812	3881	1253	5134

Table 1 shows that file 1 contains 12808 tokens and 608 periods in it, out of it there are 453 periods behaving as sentence marker while 155 are not sentence markers. File 2 has 55034 tokens. It has 2501 periods in it and out of 2501 periods 1852 show the behavior of sentence marker punctuation where as rest of 649 show non-terminating behavior. File 3 contains 56970 tokens out of it there are 2373 periods. These periods contain 1924 actual sentence terminators and 449 periods used in abbreviations, dates or numeric values. File 4 has about 437 terminating and 117 non-terminating periods and file 5 contains 3881 and 1253 terminating and non-terminating periods respectively. Table 2 shows that there are 6 question marks in file 1 and all of them are used as sentence terminators. File 2 has about 17 question marks and only one of them is ambiguity creating mark. File 3 has 22 question marks behaving as sentence marker. Files 4 and 5 contain 45 and 9 question marks respectively, as sentence terminators, whereas each contains only one non-terminating question mark.

Table 2. Use of question mark in training and testing data.

No. of Tokens	Question Mark (Sentence Terminator)	Question Mark (Not Sentence Terminator)	Total No. of Question Marks
12808	6	0	6
55034	17	1	18
56970	22	0	22
12481	45	1	46
124812	9	1	10

Table 3. Use of exclamation mark in training and testing data.

No. of Tokens	Exclamation Sign (Sentence Terminator)	Exclamation Sign (Not Sentence Terminator)	Total No. of Exclamation Signs
12808	0	0	0
55034	0	2	2
56970	2	0	2
12481	0	0	0
124812	2	2	4

According to Table 3 there are only 2 exclamation marks in file 2 and both of them are non-terminating marks. File3 has only 2 exclamation marks and both of these are sentence terminator. File 5 contains two terminator and two non-terminator exclamation marks.

Table 4. Sentence terminating punctuations in training and testing data.

No. of Tokens	No. of Periods	No. of Question Marks	No. of Exclamation Marks	Total No. of Sentence Markers
12808	453	6	0	459
55034	1852	17	0	1869
56970	1924	22	2	1948
12481	437	45	0	482
12482	3881	9	2	3892

Table 5. Non-terminating punctuations in training and testing data.

No. of Tokens	No. of Periods	No. of Question Marks	No. of Exclamation Marks	Total No. of Non-Terminating Punctuations
12808	155	0	0	155
55034	649	1	2	652
56970	449	0	0	449
12481	117	1	0	118
12482	1253	1	2	1256

According to Tables 4 and 5, file1 contains 459 terminating and 155 non-terminating punctuations, file2 has 1869 terminating and 652 non-terminating punctuations and file3 has 1948 terminating and 449 non-terminating punctuations. Files 4 and 5 contain 118 and 1256 ambiguity creating punctuations respectively.

4.2. Results Obtained by Keeping Training and Testing Data Similar to Each other

Results we obtained while keeping same training and testing data, by using unigram statistical model are given in Table 6.

Table 6. Results using unigram statistical model.

No. of Sentences	Precision	Recall	F1-Measure
459	75.70%	99.78%	86.09%
1869	74.61%	100%	85.45%
1948	81.54%	100%	89.83%

Performance evaluation of the model is given in Figure 2. These results show that even using same training and testing data we could not get very high values of precision. It is all because of small sized training data.

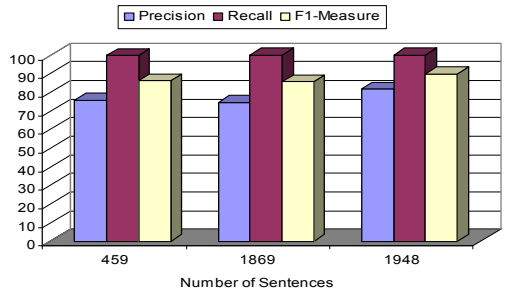

Figure 2. Performance using unigram statistical model.

Data set having 459 sentences showed 75.70% precision, 99.78% recall, and 86.09% F1-Measure. Data set of 1869 sentences could produce 74.61% precision, 100% recall, and 85.04% F1-Measure. Data set with 1948 sentences showed slightly better performance. It showed about 81.54% precision, 100% recall, and 89.83% F1-Measure. After applying our proposed algorithm we obtained following results.

Table 7. Results obtained by applying proposed approach.

No. of Sentences	Precision	Recall	F1-Measure
459	92.61%	76.47%	83.77%
1869	95.13%	85.82%	90.23%
1948	99.36%	96.45%	97.89%

Figure 3 shows evaluation of precision, recall, and F1-Measure results obtained after applying rule based algorithm to tagged data returned by unigram tagger.

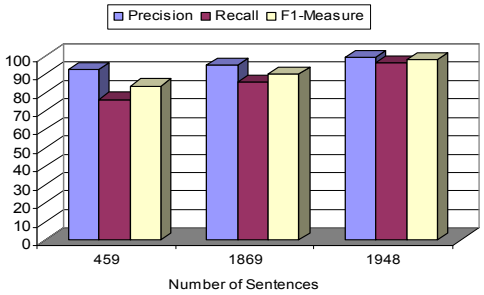

Figure 3. Performance using unigram statistical model along with rules.

Table 7 shows that we got significant improvement in results after applying our proposed algorithm to data tagged by unigram tagger. For 459 sentences, we got 92.61% precision, 76.47% recall, and 83.71% F1-Measure. For 1869 sentences, we obtained 95.13% precision, 85.82% recall, and 90.23% F1-Measure. For 1948 sentences, we got 99.36% precision, 96.45% recall, and 97.89% F1-Measure.

Table 8. Error rate of unigram model and hybrid approach in close test.

No. of Sentences	Error Rate of Unigram Model	Error Rate of Our Approach
459	32%	29%
1869	34%	18%
1948	22%	4%

Table 8 shows the comparison of error rates produced by unigram statistical model and hybrid approach. With 1948 sentences there is significant difference in error rate produced by unigram model (22%) and hybrid approach (4%). Figure 4 shows this comparison graphically.

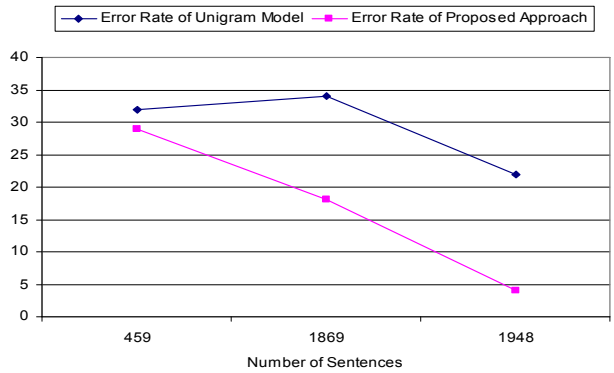

Figure 4. Comparison of unigram model and hybrid approach.

4.3. Results Obtained by Keeping Training and Testing Data Different from Each Other

While keeping training and testing data in such a way that there is no similarity in training and testing data files. The file used to train the tagger contains totally different contents from the file being used as test data file. The results we obtained are given in Tables 9 and 10 before and after applying algorithm.

Table 9. Results obtained by using unigram statistical model.

No. of Sentences in Training Data	Precision	Recall	F1-Measure
500	60.18%	99.77%	75.08%
1000	75.5%	100%	86.04%
1500	79.11%	100%	88.33%
2000	79.11%	100%	88.33%
2500	79.11%	100%	88.33%
3000	79.11%	100%	88.33%
3500	79.11%	100%	88.33%
3928	79.11%	100%	88.33%

Figure 5 shows evaluation of results contained in Table 9. Table 9 shows that keeping the test data fix and varying size of training data, as we are increasing the size of the training data set, results are improved. Therefore, significant performance of the unigram tagger requires sufficient amount of training data. Obviously, training data provided to unigram tagger is not as sufficient that it can learn where to mark a period as sentence marker and where it should leave it as an ordinary punctuation. That's why results are not as better as unigram tagger shows usually. With 1500 sentences and onwards it shows no change in results that means, its not learning something new from training data to improve its performance. Training data set, containing 1500 number of sentences and complete corpus file with 3928 number of sentences showed equivalent results. They produced about 79.11% precision, 100% recall, and 88.33% F1-Measure. Results given in Table 10 are obtained using unigram tagger along with our algorithm.

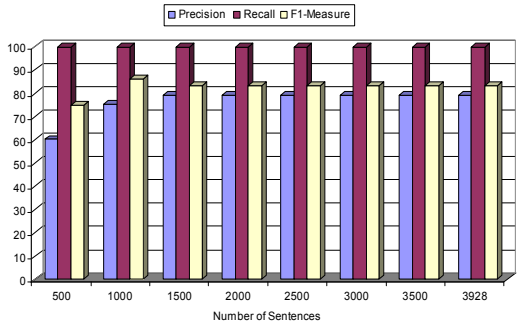

Figure 5. Performance by using unigram statistical model for test data.

Table 10. Results obtained by applying proposed approach.

No. of Sentences in Training Data	Precision	Recall	F1-Measure
500%	72.2%	89.48%	79.92%
1000%	93.7%	86.57%	90%
1500%	99.48%	86.35%	92.45%
2000%	99.48%	86.35%	92.45%
2500%	99.48%	86.35%	92.45%
3000%	99.48%	86.35%	92.45%
3500%	99.48%	86.35%	92.45%
3928%	99.48%	86.35%	92.45%

Performance evaluation of results shown in the above table is given in Figure 6.

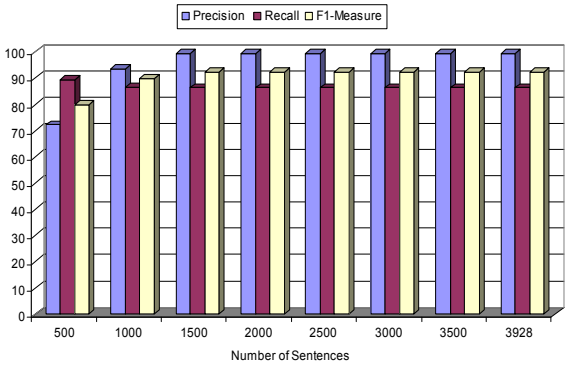

Figure 6. Performance of proposed approach for test data.

After applying algorithm to data returned by tagger, results are improved. We obtained 99.48% precision, 86.35% recall, and 92.45% F1-Measure, with larger sets of training data.

Table 11. Error rate of unigram model and hybrid approach in open test.

Number of Sentences in Training Data Set	Error Rate of Unigram Model	Error Rate of Our Approach
500	66%	44%
1000	32%	19%
1500	26%	14%
2000	26%	14%
2500	26%	14%
3000	26%	14%
3500	26%	14%
3928	26%	14%

Table 11 shows the comparison of unigram model and hybrid approach, while keeping fix sized testing data and training data with varying size. It shows that error rate is reduced, by both of these techniques, with larger training data sets. It also shows that low error rate is achieved by our proposed approach as compared to the unigram model with same training and testing data. Figure 7 shows this comparison graphically.

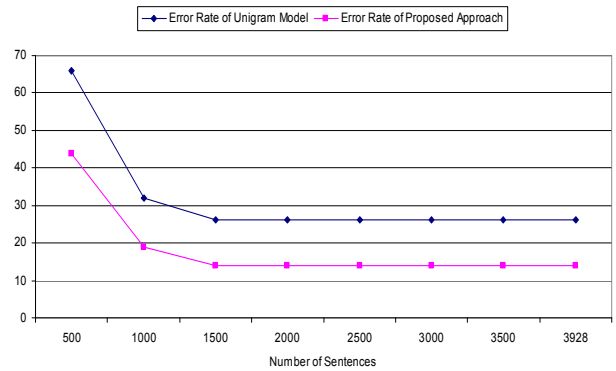

Figure 7. Comparison of unigram model and hybrid approach.

Following formulas are used for precision, recall, F1-Measure and error rate.

Precision=Total number of correct sentence markers/Total number of speculated sentence markers.
Recall=Total number of correct sentence markers/Total number of actual sentence markers.
*F1-Measure=2*Precision*Recall/Precision + Recall.*
Error Rate= (Total no. of missed sentence markers + Total no. of incorrectly detected sentence markers)/Total number of actual sentence markers.

Correct sentence markers are those which have been correctly identified after applying our technique to test data. Whereas speculated sentence markers are all periphery punctuations obtained from test data after implementing our approach. While actual sentence markers are the punctuations behaving like sentence boundary in the test data.

5. Conclusions

Sentence boundary identification is an intricate job especially for Arabic script languages. We have implemented a hybrid approach for Urdu sentence boundary disambiguation. Our approach combines unigram statistical model and rule based algorithm. Unigram model (trained over tagged data) has been used to tag the test data. Sentence boundary ambiguities have been found in the data tagged by the unigram tagger. To solve these ambiguities, rule based algorithm has been formulated. To a significant extent these ambiguities have been resolved by the algorithm. We have performed both open and close tests to compute accuracy of our proposed approach.

Open test showed 99.48% precision, 86.35% recall, and 92.45% F1-Measure, whereas, close test produced 99.36% precision, 96.45% recall, and 97.89% F1-Measure with maximum number of sentences in training data set. Error rates produced by both unigram model and our proposed approach have been compared and it is seen that proposed approach gives low error rates as compared to unigram model.

References

[1] Agarwal N., Ford K., and Shneider M., "Sentence Boundary Detection Using a MaxEnt Classifier," *in Proceedings of MISC*, CA, pp. 1-6, 2005.

[2] Anwar W., Wang X., and Li L., "A Statistical Based Part of Speech Tagger for Urdu Language," *in Proceedings of Machine Learning and Cybernatics*, Hong Kong, pp. 3418-3424, 2007.

[3] Manning C. and Schutze H., *Foundations of Statistical Natural Language processing*, Massachusetts Institute of Technology Press, UK, 1999.

[4] Dincer B. and Karaoglan B., "Sentence Boundary Detection in Turkish," *in Proceedings of Advances in Information Systems*, Berlin, pp. 255-262, 2004.

[5] Kiss T. and Strunk J., "Unsupervised Multilingual Sentence Boundary Detection," *Journal of MIT Press*, vol. 32, no. 4, pp. 485-525, 2006.

[6] Kiss T. and Strunk J., "Viewing Sentence Boundary Detection as Collocation Identification," *in Proceedings of KONVENS*, pp. 75-82, 2002.

[7] Malik A., "A Hybrid Model for Urdu Hindi Translation," *in Proceedings of the Named Entities Workshop*, Singapore, pp. 177-185, 2009.

[8] Mikheev A., "Tagging Sentence Boundaries," *in Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*, pp. 264-271, 2000.

[9] Mobarakeh I. and Bidgoli M., "Verb Detection in Persian Corpous," *International Journal of Digital Content Technology and its Applications*, vol. 3, no. 1, pp. 58-65, 2009.

[10] Palmer D. and Hearst M., "Adaptive Sentence Boundary Disambiguation," *in Proceedings of the 4th Conference on Applied Natural Language processing*, Germany, pp. 73-83, 1994.

[11] Phuong H. and Vinh T., "A Maximum Entropy Approach to Sentence Boundary Detection of Vietnamese Texts," *in Proceedings of IEEE Conference of Research*, pp. 1-5, 2008.

[12] Reynar J. and Ratnaparkhi A., "A Maximum Entropy Approach to Identifying Sentence Boundaries," *in Proceedings of the 5th Conference on Applied Natural Language Processing*, USA, pp. 16-19, 1997.

[13] Rezaei S., "Tokenizing an Arabic Script Language," *Arabic NLP Workshop at ACL/EACL*, France, 2001.

[14] Riaz K., "Challenges for Urdu Stemming a Progress Report," *in Proceedings of BCS IRSG Symposium: Future Directions in Information Access*, pp. 1-6, 2007.

[15] Romportl J., Tihelka D., and Matousek J., "Sentence Boundary Detection in Czech TTS System Using Neural Networks," *in Proceedings of IEEE*, pp. 247-250, 2003.

[16] Walker D., Clements D., Darwin M., and Amtrup J., "Sentence Boundary Detection: A Comparison of Paradigms for Improving MT Quality," *in Proceedings of Machine Translation in the Information Age*, pp. 369-372, 2001.

[17] Wang H. and Huang Y., "Bondec- A Sentence Boundary Detector," CS224N Project, Stanford, 2003.

Zobia Rehman is a lecturer at COMSATS Institute of Information Technology, Pakistan since October 2009. She did her MS in computer science from COMSATS in 2009. Her area of interest is natural language processing and artificial neural networks.

Waqas Anwar is working in COMSATS Institute of Information Technology, Pakistan as assistant professor since April 2008. He got his PhD degree in Computer application technology from Harbin Institute of Technology, PR China in 2008. He did Masters in computer science from Hamdard University, Pakistan in 2001. He is an active researcher and his areas of interest are Natural language processing and computational intelligence.