A word segmentation system for handling space omission problem in urdu script

Nadir Durrani
Institute for NLP
Universität Stuttgart
durrani@ims.uni-stuttgart.de

Abstract

Word Segmentation is the foremost obligatory task in almost all the NLP applications where the initial phase requires tokenization of input into words. Urdu is amongst the Asian languages that face word segmentation challenge. However, unlike other Asian languages, word segmentation in Urdu not only has space omission errors but also space insertion errors. This paper discusses how orthographic and linguistic features in Urdu trigger these two problems. It also discusses the work that has been done to tokenize input text. We employ a hybrid solution that performs an n-gram ranking on top of rule based maximum matching heuristic. Our best technique gives an error detection of 85.8% and overall accuracy of 95.8%. Further issues and possible future directions are also discussed.

1 Introduction

All language processing applications require input text to be tokenized into words for further processing. Languages like English normally use white spaces or punctuation marks to identify word boundaries, though with some complications, e.g. the word "e.g." uses a period in between and thus the period does not indicate a word boundary. However, many Asian languages like Thai, Khmer, Lao, and Dzongkha do not have word boundaries and thus do not use white space to consistently mark word endings. This makes the process of tokenization of input into words for such languages very challenging.

Urdu is spoken by more than 100 million people, mostly in Pakistan and India. It is an Indo-Aryan language, written using Arabic script from right to left, and Nastalique writing style [5]. Nastalique is a cursive writing system, which also does not have a concept of space. Thus, though space is used in typing the language, it serves other purposes, as discussed later in this paper. This entails that space cannot be used as a reliable delimiter for words. Therefore, Urdu shares the word segmentation challenge for language processing, like other Asian languages.

This paper explains the problem of word segmentation in Urdu. It gives details of work done to investigate linguistic typology of words and motivation of using space in Urdu. The paper then presents an algorithm developed to automatically process the input to produce consistent word segmentation, and finally discusses the results and future directions.

2 Space Omission Problem

Space omission problem arise in cases where words are written continuously without any space or other separating characters like ZWNJ. Languages like Chinese, Japanese, and Thai address space omission problems. Space omission problems are challenging because there are multiple ways in which a space can be omitted. A classic example from English is famously quoted. There are multiple ways of segmenting the following sentence: GODISNOWHERE.

Segmentation
GOD IS NO WHERE
GOD IS NOWHERE
GOD IS NOW HERE

Table 1: Segmentation Ambiguity

2.1 Non-Joiner Word Ending

We mentioned before that the concept of space is uncommon in hand-written Urdu orthography. Then we said that because of the limitation in technology, space (or ZWNJ) has become part of the language, and to make the word visually appropriate, the user must insert something between two words. However, when a word ends with a non-joiner, the next word can be written without inserting a space. Because non-joiners cannot acquire medial and initial shapes, they do not combine with the starting character of the next word. This allows the user to start the next word without putting a space. Consider the same example below:

بادشاہی مسجد کا دروازہ بند ہے۔

A native speaker may or may not put a space between کا and دروازہ because کا ends with a non-joiner ا and will not connect with د (the first character of the following word). So without a space, کادروازہ is as acceptable as کا دروازہ. Therefore, a sentence with all words ending with non-joiners might not have a space character at all. One such example is shown below.

As can be seen (a) and (b) look visually identical although (a) doesn't have any space while (b) has space after each word. Ambiguity arises when a word is composed of smaller words and is required to be segmented differently based on context it is occurring. Example is shown below.

Urdu Sentence	Translation
a) قافلے کے لیڈر احمد شیر ڈوگر نے کہا	The leader of the caravan, Ahmad Sher Dogar, said
b) قافلے کے لیڈر احمد شیر ڈوگر نے کہا	The leader of the caravan, Ahmad Sher Dogar, said

Urdu Text	English Translation
نوجوان ادھر آو	Young lad come here
پنجاب برگیڈ کے نوجوان شہید ہیں	Nine soldiers of Punjab brigade have been martyred
وہ نوجوان ساتھ کے ہر جگہ جاتے ہیں	Those nine that go with them every place

Table 2: Segmentation Ambiguity

In the first sentence, نوجوان ("nojawan") is a single word meaning "young lad" or "youngster." In the second example, نوجوان it comprises two words, نو "no" and جوان "jawan", meaning "nine" and "soldiers," respectively. There's an alternative translation where the second example could also mean "The soldiers of Punjab brigade have been martyred" in which case نوجوان "nojawan" again represents a single word meaning "soldiers." In the last scenario, نوجوان it consists of three words, نو , ان جو, meaning "nine," "that," and "them." The last word ان is usually pronounced as اِن "them"). Pronunciations in Urdu are marked by diacritics, such as پیش stacks above character) or زیر; connects to the bottom of a character). However, diacritics have become rare in Urdu text because a native speaker can guess the pronunciation through tacit knowledge or by looking at the context of the word. The third scenario would have been ruled out if the text were diacritized.

Non-joiner word ending and space insertion problems are initiated by Urdu orthography. They can occur in all kinds of words that end with non-joiners. Not putting a space has almost the same visual impact. This makes its use optional and the user might only put it for the sake of tidiness/readability.

3 Segmentation System for Urdu

Although many other languages share the same problem of word boundary identification for language processing, Urdu problem is unique due to its cursive script and its irregular use of space to create proper shaping. Though other languages only have space omission challenge, Urdu has both omission and insertion problems further confounding the issue. We employ a combination of techniques to investigate an effective algorithm to achieve Urdu segmentation.

These techniques are incorporated based on knowledge of Urdu linguistic and writing system specific information for effective segmentation. For space omission problem a rule based maximum matching technique is used to generate all the possible segmentations. The resulting possibilities are ranked using three different heuristics, namely min-word, unigram and bigram techniques.

The segmentation process starts with preprocessing, which involves removing diacritics (as they are optionally used in Urdu and not considered in the current algorithm because they are frequently incorrectly marked by users) and normalizing the input text to remove encoding ambiguities. Input is then tokenized based on space and punctuation characters in the input stream. As has been discussed, space does not necessarily indicate word boundary. However presence of space does imply word or morpheme boundary in many cases, which can still be useful. The tokenization process gives what we call an Orthographic Word (OW). OW is used instead of "word" because one OW may eventually give multiple words and multiple OWs may combine to give a single word. Keeping space related information also keeps the extent of problem to be solved within a reasonable computational complexity. For example input string نادر خان درانی (the name of the first author) with spaces giving three OWs, creates 2 x 1 x 7 = 14 possible segmentations when sent separately to the maximum matching module (space omission error removal - see Figure 2). However, if we remove the spaces from the input and send input as a single OW نادرخاندرانی to maximum matching process, we get 77 possible segmentations. This number grows exponentially with the length of input sentence. Throwing away space character means we are losing important information so we keep that intact to our use. After pre-processing a series of modules further process the input string and convert the OWs into a sequence of words. Each OW is sent to a module which deals with space omission errors. This module extracts all possible morpheme segmentations out of an OW. Ten best segmentations of these are selected based on minimum-word heuristic. This heuristic prefers segmentations with minimum number of morphemes. Such a heuristic is important to prevent the search space to explode. We observed that using 10-best segmentations proved to be sufficient in most cases as OW normally encapsulates two or three Urdu words but as a heuristic we also added a feature which increases this number of 10-best segmentations to 15, 20, 25-best and so on depending upon number of characters in an OW. Ten best segmentations for each OW are merged with the extracted segmentations of other OWs. Up till here we have successfully resolved all space omission errors and the input sentence has been segmented into morphemes

4 Results

The algorithm was tested on a very small, manually segmented corpus of 2367 words. The corpus we selected contained 404 segmentation errors with 221 cases of space omissions.

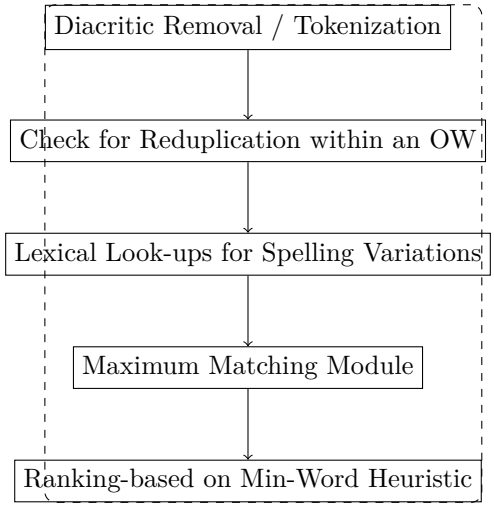

Figure 1: Urdu Word Segmentation Process

Categories	Errors	%ages
Maximum Matching	186/221	84.16
Unigram	214/221	96.83
Bigram	209/221	94.5

Table 3: %age of No. of Errors Detected in Space Omission with Different Ranking Techniques

There were 221 cases of space omission errors where multiple words were written in a continuum. Given below is a table that shows how many of these were correctly identified by each of the used techniques. Clearly, statistical techniques outperform a simple minimum number of words heuristic. Bigrams are likely to produce better results if the training corpus is improved. Our training corpus contained manually segmented 70K words. The bigram probabilities are obtained using SRILM-Toolkit [15]. Following table gives cumulative results for correctly identified space omission and insertion errors.

Categories	Errors	%ages
Maximum Matching	323/404	79.95
Unigram	347/404	85.8
Bigram	339/404	83.9

Table 4: %age of No. of Errors Detected Cumulatively

Final table counts total number of words (reduplication, compounds and abbreviations cases are inclusive) in test corpus and total number of correctly identified words after running the entire segmentation process.

Categories	Detected	%ages
Maximum Matching	2209/2367	93.3
Unigram	2269/2367	95.8
Bigram	2266/2367	95.7

Table 5: Percentage of Correctly Detected Words

5 Summary of Existing Techniques

Rule-based techniques have been extensively used for word segmentation. Techniques including longest matching [12, 13] try to match the longest possible dictionary look-up. If a match is found at n-th letter, the next lookup is performed starting from the $n + 1$ index. Longest matching with word binding force is used for Chinese word segmentation [16]. However, the problem with this technique is that it consistently segments a letter sequence the same way, and does not take the context into account. Thus, shorter word sequences are never generated, even where they are intended. Maximum matching is another rule based technique that was proposed to solve the shortcomings of longest matching. It generates all possible segmentations out of a given sequence of characters using dynamic programming. It then selects the best segmentation based on some heuristics. Most popularly used heuristic selects the segmentation with minimum number of words. This heuristic fails when alternatives have same number of words. Some additional heuristics are then often applied,

including longest match [14]. Many variants of maximum matching have been applied [4, 7, 8, 10].

There is a third category of rule based techniques, which also use additional linguistic information for generating intermediate solutions which are then eventually mapped onto words. For example, rule based techniques have also been applied to languages like Thai and Lao to determine syllables, before syllables are eventually mapped onto words [11].

There has been an increasing application of statistical methods, including n-grams, to solve word segmentation. These techniques are based at letters, syllables and words, and use contextual information to resolve segmentation ambiguities, e.g [1, 6]. The limitation of statistical methods is that they only use immediate context and long distance dependencies cannot be directly handled. Also the performance is based on training corpus. Nevertheless, statistical methods are considered to be very effective to solve segmentation ambiguities.

Finally, another class of segmentation techniques applies several types of features, e.g. Winnow and RIPPER algorithms [9, 2]. The idea is to learn several sources of features that characterize the context in which each word tends to occur. Then these features are combined to remove the segmentation ambiguities [3].

6 Future Work

This work presents a preliminary effort on word segmentation problem in Urdu. It is a multidimensional problem. Each dimension requires a deeper study and analysis. We have developed a system for space omission problems in Urdu. In the future we will work on the space insertion problem where user has inserted spaces between a word that should be deemed as a single unit.

References

[1] W. Aroonmanakul. Collocation and thai word segmentation. In *Proceedings of SNLP-Oriental COCOSDA*, 2002.

[2] A. Blum. Empirical support for winnow and weighted-majority algorithm: Results on a calendar scheduling domain. *Machine Learning*, 26:5–23, 1997.

[3] P. Charoenpornsawat and B. Kijsirikul. Feature-based thai unknown word boundary identification using winnow. In *Proceedings of the 1998 IEEE Asia-Pacific Conference on Circuits and Systems (APCCAS'98)*, 1998.

[4] P. Gu and Y. Mao. The adjacent matching algorithm of chinese automatic word segmentation and its implementation in the qhfy chinese-english system. In *International Conference on Chinese Computing*, Singapore, 1994.

[5] S. Hussain. www.lict4d.asia/fonts/nafees_nastalique. In *Proceedings of 12th AMIC Annual Conference on E-Worlds: Governments, Business and Civil Society*, Singapore, 2003. Asian Media Information Center.

[6] A. Krawtrakul, C. Thumkanon, Y. Poovorawan, and M. Suktarachan. Automatic thai unknown word recognition. In *Proceedings of the Natural Language Processing Pacific Rim Symposium*, 1997.

[7] B.Y. Li, S. Lin, C.F. Sun, and M.S. Sun. A maximum-matching word segmentation algorithm using corpus tags for disambiguation. In *ROCLING IV*, pages 135–146, Taipei, 1991.

[8] N. Liang. A written chinese automatic segmentation system-cdws. *Journal of Chinese Information Processing*, 1(1):44–52, 1986.

[9] S. Meknavin, P. Charenpornsawat, and B. Kijsirikul. Feature-based thai words segmentation. In *NLPRS, Incorporating SNLP*, 1997.

[10] J. Nie, W. Jin, and M. Hannan. A hybrid approach to unknown word detection and segmentation of chinese. In *International Conference on Chinese Computing*, Singapore, 1994.

[11] P. Phissamay, V. Dalolay, C. Chanhsililath, O. Silimasak, S. Hussain, and N. Durrani. Syllabification of lao script for line breaking. In *PAN Localization Working Papers 2004-2007*, 2007.

[12] Y. Poowarawan. Dictionary-based thai syllable separation. In *Proceedings of the Ninth Electronics Engineering Conference*, 1986.

[13] S. Rarunrom. Dictionary-based thai word separation. Senior Project Report, 1991.

[14] V. Sornlertlamvanich. Word segmentation for thai in a machine translation system (in thai). In *Papers on Natural Language Processing*, Thailand, 1995. NECTEC.

[15] Andreas Stolcke. Srilm – an extensible language modeling toolkit. In *Proc. Int. Conf. Spoken Language Processing (ICSLP 2002)*, 2002.

[16] P. Wong and C. Chan. Chinese word segmentation based on maximum matching and word binding force. In *Proceedings of COLING 96*, pages 200–203, 1996.