Rule Based Grammar Checker For Pashto Language

Mustafa Toufiq
P157001@nu.edu.pk

MS(Computer Science)

Supervisor: Dr.Omar Usman Khan
omar.khan@nu.edu.pk

National University of Computer and Emerging Sciences

May 14, 2019

Contents

- Introduction
- Literature Review
- Rule Based Method
- Probabilistic Production Rules
- DataSet
- Viterbi Parser
- State Transition Diagram
- Problem Statement
- Methodology
- Future Work
- References

Introduction

Area of Research: Natural Language Processing (NLP)

Topic of Research: Rule Based Grammar Checker For Pashto Language.

Background

- A **Grammar Checker** program allows us to correct a mistake while the word or phrase is still fresh in our mind.
- Grammar Checkers typically make use of Natural Language Processing and Grammatical Rules to identify grammatical mistakes.

Literature Review

- **Statistical Based Approach**
  - A POS-annotated corpus is used to build a list of POS tag sequences.
  - Some sequences will be very common, others will probably not occur at all.
  - Sequences which occur often in the corpus can be considered correct while uncommon sequences could be errors.
  - **Difficult to interpret:**
    - If the system raises false errors, users will wonder why their input is considered incorrect when no specific error message is given be errors.

Asanilta Fahda Ayu Purwarianti "A Statistical and Rule-Based Spelling and Grammar Checker for Indonesian Text"
International Conference on data and software engineering Indonesia(2017)

Literature Review

- **Example**

غواړم	بخښنه	ډیره	زه
Verb	Adverb	Noun	Noun

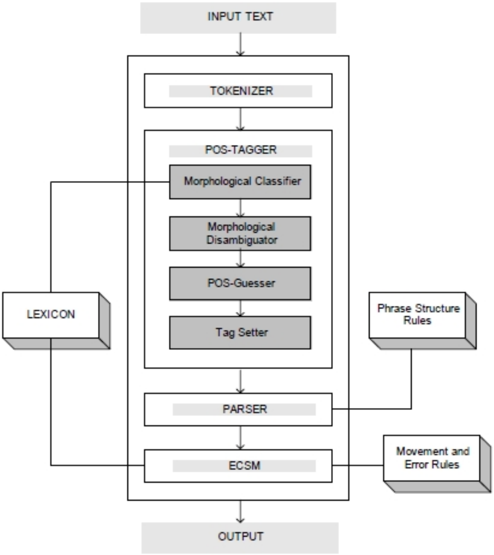
غواړم	غواړم	بخښنه	زه
Verb	Verb	Noun	Noun

Literature Review

- **Two Pass Parsing Implementation for an Urdu Grammar Checker**
  - A sentence is first parsed on basic PSG (Phrase Structure Grammar) rules.
  - Upon failure, Movement Rules are applied to convert it to a desired correct form.
    - It helps in reducing the number of PSR needed to represent the sentence.
    - It helps to repharse the structure of the sentence.
  - After that the sentence is reparsed to check for errors.

Hammad Kabir, Shanza Nayyer, Jahangir Zaman, and Dr. Sarmad Hussain "Two Pass Parsing Implementation for an Urdu Grammar Checker" Inmic(2002), Karachi

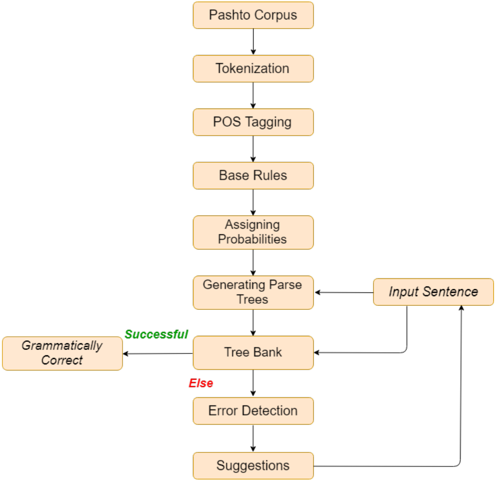
Flow Chart of Urdu Grammar Checker



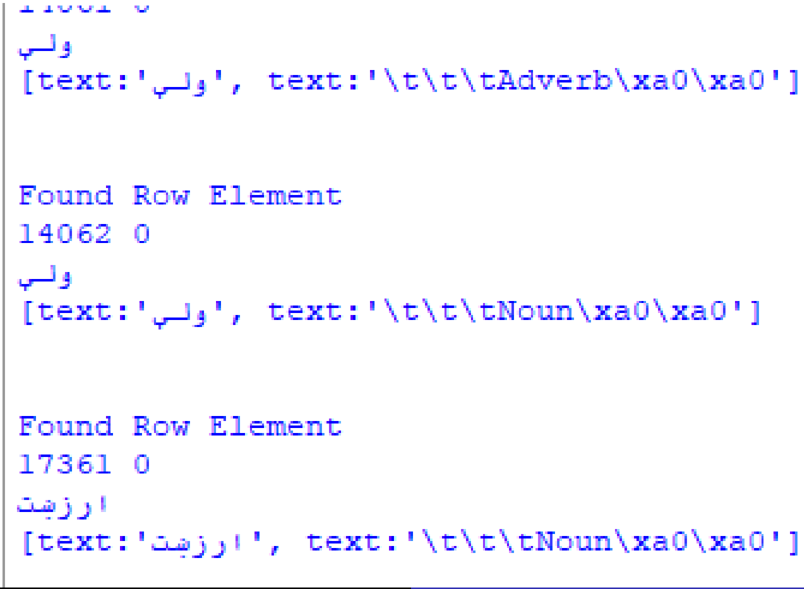
Rule Based Grammar Checker

- A POS-annotated corpus is used to build a list of POS tag sequences.
- Input is tokenized and every word is assigned with its POS tag.
- Some computational base rules are made to generate parse trees and a probability is assigned to every rule.
- As a result a tree-bank is created.
- The input sentence is then checked with the tree bank.
- If parsing is successful, the sentence will be marked as correct.
- Else it will go to the next module which is error detection and suggestion.

Rule Based Grammar Checker



Probabilistic Production Rule



```
14061 0
ولي
[text:'ولي', text:'\t\t\tAdverb\xa0\xa0']


Found Row Element
14062 0
ولي
[text:'ولي', text:'\t\t\tNoun\xa0\xa0']


Found Row Element
17361 0
ارزښت
[text:'ارزښت', text:'\t\t\tNoun\xa0\xa0']
```

DataSet (Corpus)

- Initially we had 75 sentences in our Dataset.

- For Training = 25 sentences (for creation of production rules).
- For Testing = 50 sentences (for testing)

Parsers

- **Initially we applied three parsers.**
  1. Shift Reduce Parser.
  2. Recursive Decent Parser.
  3. Chart Parser.
- **Problems with these parsers**.
  - Infinite loop.
  - Exponential time.
  - Problems in generating probability.
- **Solution**
  - Viterbi Parser.

Viterbi Parser

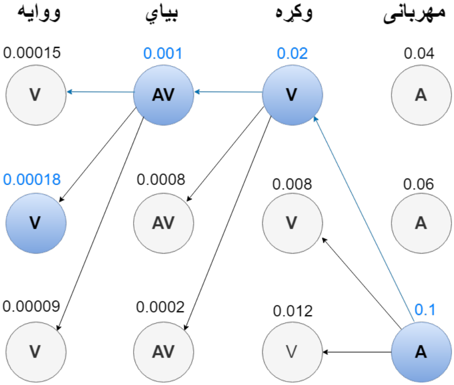
- A bottom-up parser that uses dynamic programming approach.
- Finds the highest probability sequence amoung all the state sequences.
- Creates a table which records the most probable tree representation.
- The Parser fills in this table incrementally.
- Finally backtracking is done to record the highest probability sequence.

Example

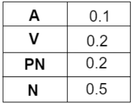
A	0.5
V	0.3
AV	0.2

	A	V	AV
A	0.5	0.4	0.1
V	0.2	0.6	0.2
AV	0.5	0.2	0.3

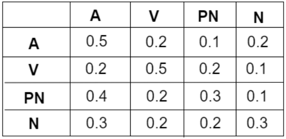
ووايه	بياي	وکړه	مهربانى
0.3	0.1	0.4	0.2



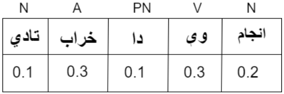
Example



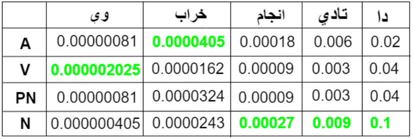
A	0.1
V	0.2
PN	0.2
N	0.5



	A	**V**	**PN**	**N**
A	0.5	0.2	0.1	0.2
V	0.2	0.5	0.2	0.1
PN	0.4	0.2	0.3	0.1
N	0.3	0.2	0.2	0.3



N	A	PN	V	N
تادي	**خراب**	**دا**	**وي**	**انجام**
0.1	0.3	0.1	0.3	0.2

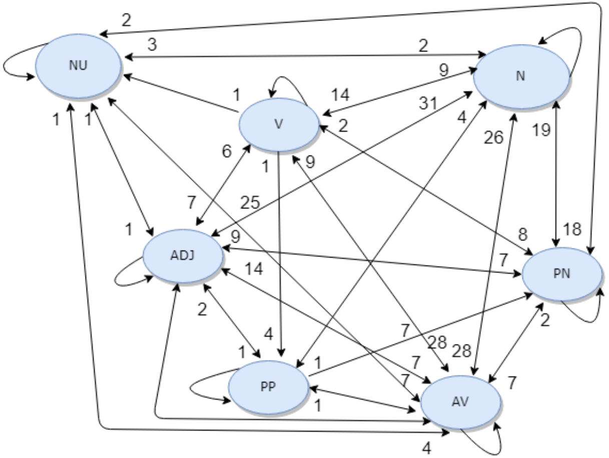


	وي	خراب	انجام	تادي	دا
A	0.00000081	**0.0000405**	0.00018	0.006	0.02
V	**0.000002025**	0.0000162	0.00009	0.003	0.04
PN	0.00000081	0.0000324	0.00009	0.003	0.04
N	0.000000405	0.0000243	**0.00027**	**0.009**	**0.1**

Assigning Probabilities

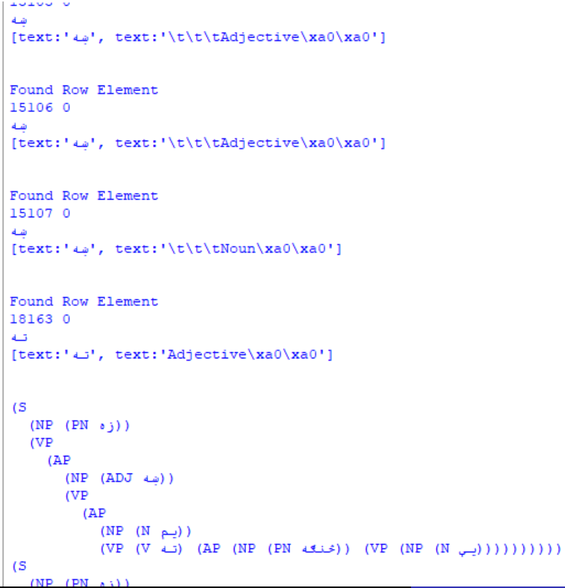
- Every production rule have some probability.
- In order to assign probability we first check the state transition frequency.
- State transition frequencies are generated from the dataset.
- It is obtained by adding all the values and then by dividing it by total number of POS tag.

State Transition Diagram



N N ->	149
NU NU ->	8
ADJ ADJ ->	77
AV AV ->	72
V V ->	42
PP PP ->	7
PN PN ->	47

Initial Results



```
15105 0
ښه
[text:'ښه', text:'\t\t\tAdjective\xa0\xa0']


Found Row Element
15106 0
ښه
[text:'ښه', text:'\t\t\tAdjective\xa0\xa0']


Found Row Element
15107 0
ښه
[text:'ښه', text:'\t\t\tNoun\xa0\xa0']


Found Row Element
18163 0
ته
[text:'ته', text:'Adjective\xa0\xa0']


(S
  (NP (PN زه))
  (VP
    (AP
      (NP (ADJ ښه))
      (VP
        (AP
          (NP (N یم))
          (VP (V ته) (AP (NP (PN څنګه)) (VP (NP (N یې)))))))))
(S
  (NP (PN زه))
```

Future Work

1. Creation of a Tree Bank.
2. After we obtain all the trees we will create a tree bank
3. Tree bank will only contain the trees having maximum probability.

Problem Statement

- No **Rule Based** Grammar Checker is available for **Pashto Language** which can identify grammatical mistakes.

Methodology

- Creating a POS tagge and some traiing probablistic rules inorder to generate parse trees and also developing a tree-bank with the help of which we can identify grammatical errors in a sentence and give suggestions,

References

1. Khaled F. Shaalan "Arabic GramCheck: A grammar checker for Arabic"
   Wiley InterScience(2005)

2. Asanilta Fahda Ayu Purwarianti "A Statistical and Rule-Based Spelling and Grammar Checker for Indonesian Text"
   International Conference on data and software engineering Indonesia(2017)

3. Hammad Kabir, Shanza Nayyer, Jahangir Zaman, and Dr. Sarmad Hussain "Two Pass Parsing Implementation for an Urdu Grammar Checker"
   Inmic(2002), Karachi

References

4. Syed Muhhamad Jafar Rizvi "Development of algorithm and computational grammar for urdu"
Pakistan Infinitude of engineering and applied sciences nilore Islamabad 45650 Pakistan. March 2007

5. Stuart M. Shieber "Sentence disambiguation by a shift-reduce parsing technique"
ACL(1983), Cambridge, Massachusetts