NLP202 assignment 2

Cheng-Tse Liu

February 21, 2024

1 Introduction

In this assignment, we will implement CKY algorithm in different ways.

1.1 Part 1

For part 1, we'll implement CKY algorithm by hands. The following figure 1 is my result.

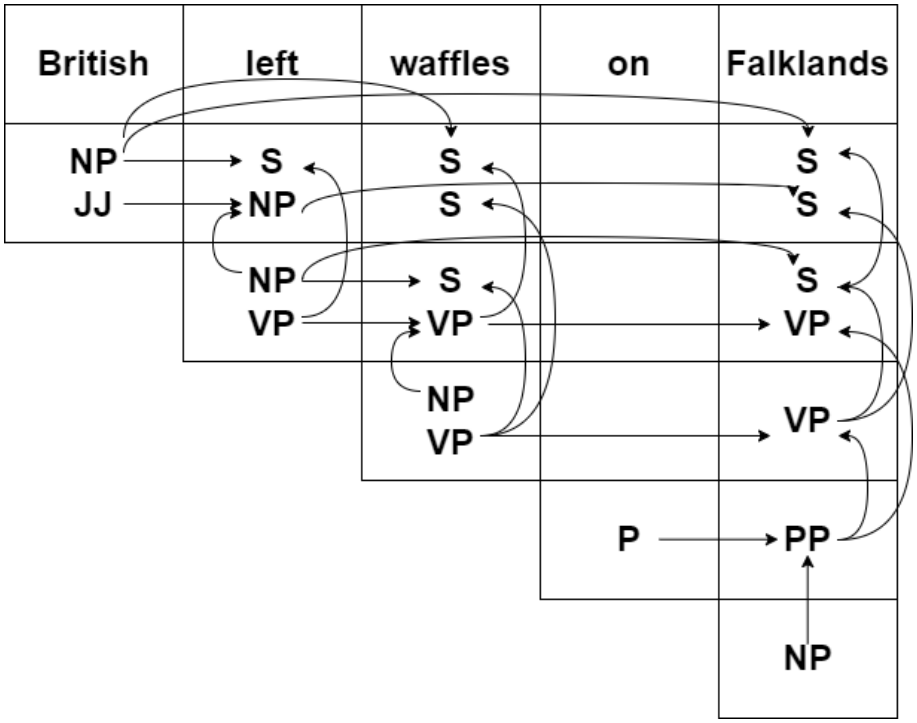

Figure 1: Chart Assuming the Use of the CKY Algorithm

1.2 Part 2

For part 2, we'll implement CKY algorithm by code. The following table is my result. I used 5 for loops to implement the algorithm. Also, since we're using set to implement this algorithm for storing the non-terminals in each cell, so we only have single tag for those blocks that have multiple same tags.

British	waffles	left	on	Falklands
$['NP','JJ']$	$['S','NP']$	$['S']$		$['S']$
	$['NP','VP']$	$['S','VP']$		$['S','VP']$
		$['NP','VP']$		$['VP']$
			$['P']$	$['PP']$
				$['NP']$

1.3 Part 3

For part 3, we'll implement weighted CKY algorithm. The difference is that each relationship has a probability and we need to calculate every relationship probability that formed the tree and get the most probable one. The calculation way is to multiply the origin probability of the children nodes and the relation probability. The following table is my result. Figure 2 is the most prabable parse tree we got. Figure 3 is the output result of my code.

astronomers	saw	stars	with	ears
$['NP': 0.4]$		$['S': 0.0504]$		$['S': 0.0036287999999999997]$
	$['NP': 0.04, 'V': 1.0]$	$['VP': 0.126]$		$['VP': 0.009071999999999998]$
		$['NP': 0.18]$		$['NP': 0.01296]$
			$['P': 1.0]$	$['PP': 0.18]$
				$['NP': 0.18]$

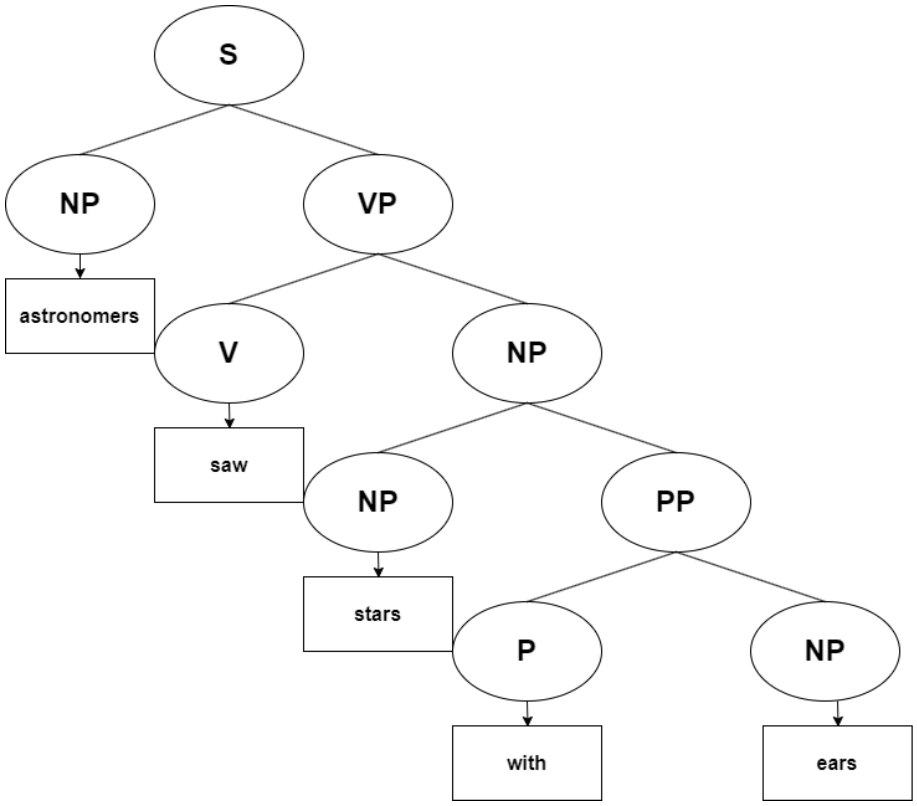

Figure 2: The Most Probable Parse Tree

```
Most probable parse tree: ('S', ('NP', 'astronomers'), ('VP', ('V', 'saw'), ('NP', ('NP', 'stars'), ('PP', ('P', 'with'), ('NP', 'ears')))))
Probability of the most probable parse tree: 0.0036287999999999997
```

Figure 3: The Most Probable Parse Tree and its Probability

1.4 Part 4

For part 4, we'll implement weighted CKY algorithm. The difference is that we're going to marginalize over the trees which means instead of taking argmax of the probability of the duplicate tags, we'll sum the probability of the duplicate tags inside one block. The following table is my result. Figure 4 is the output result of my

code.

astronomers	saw	stars	with	ears
$['NP' : 0.4]$		$['S' : 0.0504]$		$['S' : 0.006350399999999999]$
	$['NP' : 0.04, 'V' : 1.0]$	$['VP' : 0.126]$		$['VP' : 0.015875999999999998]$
		$['NP' : 0.18]$		$['NP' : 0.01296]$
			$['P' : 1.0]$	$['PP' : 0.18]$
				$['NP' : 0.18]$

```
Parse tree: ('S', ('NP', 'astronomers'), ('VP', ('VP', ('V', 'saw'), ('NP', 'stars')), ('PP', ('P', 'with'), ('NP', 'ears'))))
Probability of the tree: 0.006350399999999999
```

Figure 4: The Marginalized Parse Tree and its Probability