567 Computational Linguistics
Project2 Report

Jayaram Kuchibhotla

ID 50208766

Ubit : Jayaram K

Answers:

1. The entropy of a conditional probability distribution over trees, given a sentence reflects the ambiguity of a sentence.

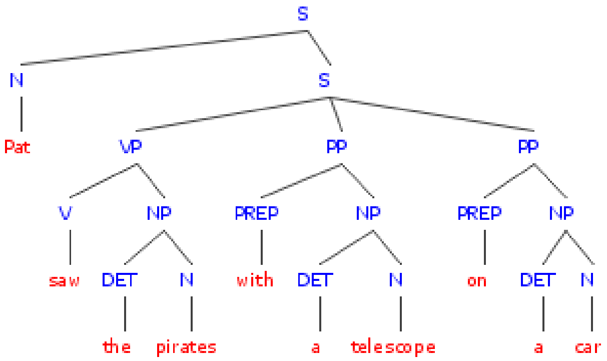
   A sentence is ambiguous if more than one tree is producible using the rules of the grammar .The more the ambiguity in a sentence, the higher would be the conditional entropy of the sentence. If only a single tree can be produced from a sentence, there is no ambiguity in such a sentence and the entropy value calculated for such a sentence is zero.

   The conditional Entropy can be calculated as:

   For non-terminal P spanning i to j, use entropy scores $\gamma_{i,j}(P)$ that store the conditional entropy of the probability distribution over trees that have P as their top label and the words from i to j as their yield $\gamma_{i,j}(P) = H(t|\ w_{i\_}, top(t) = P)$. The entropy of the probability distribution over trees for the whole sentence should be stored in $\gamma_{0,n}(S)$, where S is the start symbol and n is the length of the sentence.

   Entropy is calculated using recursion in a bottom up approach.

2. For the sentence ‘Pat saw the pirates with a telescope on a car’ ,the tree generated is as follows



Base case:

When the terminals are reached as seen in the above tree , the recursion stops. This is the base case of the recursion . The conditional entropy of the terminal nodes is zero.

Recursion:

The recursion starts from root until the base case i.e. a terminal is reached. The return of recursion (backtracking) happens from terminal to the root node.

From the above tree it can be seen that, every non terminal (other than root) has a left child , right child and a parent node, the current node can be a left or right child of other node which is above it in the tree structure . Recursion over such a tree structure helps to take inputs needed from other nodes when calculating output for a node

For each node the split score and the conditional entropy increment are calculated using the equations

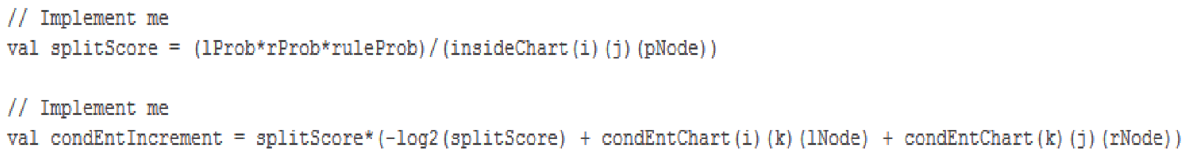
Step1: splitScore = (leftChildNodeProbability*RightChildNodeProbabiity*ruleProbabiltiy)/(Parent Node Probability calculated from the inside chart)

Step2: conditionaEntropyIncrement = splitScore*(-log$_2$(splitScore) + Entropy taken from Conditional Entropy chart for left Child Node + Entropy taken from Conditional Entropy chart for right child Node)

Step 3: Entropy of current node in Conditional Entropy Chart is the sum of Entropy of current node in Conditional Entropy chart, Entropy of the parent node taken from Conditional Entropy chart and conditionalEntropyIncrement

Above three steps are calculated for every current node till the recursion is backtracked to the root node. By the time we reach root node, we shall have the conditional entropy of the whole sentence tree

The split score and Conditional Entropy Increment are calculated as follows:



```
// Implement me
val splitScore = (lProb*rProb*ruleProb)/(insideChart(i)(j)(pNode))

// Implement me
val condEntIncrement = splitScore*(-log2(splitScore) + condEntChart(i)(k)(lNode) + condEntChart(k)(j)(rNode))
```

3.

Considering the string ‘Pat saw the pirates with a telescope on a car’

Output of Project2 code:

```
P( Pat saw the pirates with a telescope on a car )      = 4.706841743964917E-10

H( t | Pat saw the pirates with a telescope on a car )  = 2.1284618666768096
Entropy = 2.13
```

Manual Entropy Calculation (from Hw5)

Probability of a tree = (Number of samples of a tree /Total Number of Samples)

Entropy = ∑ across all trees (Probability of a tree * $\log_2$(Probability of a tree))

For n = 100 samples and seedvalue = 17,we get 5 trees

The output generated is as follows :

```
  26 (S (NP Pat) (VP (TVerb saw) (NP (NP (Det the) (Noun pirates) ) (PP (Prep with) (NP (NP (Det a) (Noun telescope) ) (PP (Prep on) (NP (Det a) (Noun car) ) ) ) ) ) ) )

    32 (S (NP Pat) (VP (TVerb saw) (NP (NP (NP (Det the) (Noun pirates) ) (PP (Prep with) (NP (Det a) (Noun telescope) ) ) ) (PP (Prep on) (NP (Det a) (Noun car) ) ) ) ) )

    21 (S (NP Pat) (VP (VP (TVerb saw) (NP (Det the) (Noun pirates) ) ) (PP (Prep with) (NP (NP (Det a) (Noun telescope) ) (PP (Prep on) (NP (Det a) (Noun car) ) ) ) ) ) )

    14 (S (NP Pat) (VP (VP (TVerb saw) (NP (NP (Det the) (Noun pirates) ) (PP (Prep with) (NP (Det a) (Noun telescope) ) ) ) ) (PP (Prep on) (NP (Det a) (Noun car) ) ) ) )

     7 (S (NP Pat) (VP (VP (VP (TVerb saw) (NP (Det the) (Noun pirates) ) ) (PP (Prep with) (NP (Det a) (Noun telescope) ) ) ) (PP (Prep on) (NP (Det a) (Noun car) ) ) ) )
```

The entropy calculation is as follows

$$0.26*\log_2(1/0.26)+0.32*\log_2(1/0.32)+0.21*\log_2(1/0.21)+0.14*\log_2(1/0.14)+0.07*\log_2(1/0.07) = 2.1697$$

For n = 1000 samples and seed value =17,we get 5 trees

```
301 (S (NP Pat) (VP (TVerb saw) (NP (NP (Det the) (Noun pirates) ) (PP (Prep with) (NP (NP (Det a) (Noun telescope) ) (PP (Prep on) (NP (Det a) (Noun car) ) ) ) ) ) ) )

   342 (S (NP Pat) (VP (TVerb saw) (NP (NP (NP (Det the) (Noun pirates) ) (PP (Prep with) (NP (Det a) (Noun telescope) ) ) ) (PP (Prep on) (NP (Det a) (Noun car) ) ) ) ) )
```

146 (S (NP Pat) (VP (VP (TVerb saw) (NP (Det the) (Noun pirates))) (PP (Prep with) (NP (NP (Det a) (Noun telescope)) (PP (Prep on) (NP (Det a) (Noun car))))))))

150 (S (NP Pat) (VP (VP (TVerb saw) (NP (NP (Det the) (Noun pirates)) (PP (Prep with) (NP (Det a) (Noun telescope))))) (PP (Prep on) (NP (Det a) (Noun car))))))

61 (S (NP Pat) (VP (VP (VP (TVerb saw) (NP (Det the) (Noun pirates))) (PP (Prep with) (NP (Det a) (Noun telescope)))) (PP (Prep on) (NP (Det a) (Noun car))))))

The entropy calculation is as follows

$0.301*\log_2(1/0.301) +0.342*\log_2(1/0.342)+0.146*\log_2(1/0.146)+0.150*\log_2(1/0.150)+0.061*\log_2(0.061) = 2.12051$

With 100 samples, the entropy value is 2.1697

With 1000 samples, the entropy value is 2.12051

With 1000 samples and seed 17, the entropy calculated manually from sampling is 2.12 close to the Entropy output 2.13

References:

1. Sentence Tree is made using the software in the below link http://www.webforditas.hu/parser