HMM BASED POS TAGGER FOR HINDI

Taneya soni, Sonam, Abhishek Kumar Pathak
taneyas22@iitk.ac.in sonamk22@iitk.ac.in akpathak22@iitk.ac.in

Department of Computer Science, IIT Kanpur, India

Abstract

Part of Speech tagging in Indian Languages is still an open problem. We still lack a clear approach in implementing a POS tagger for Indian Languages. In this project we describe our efforts to build a Hidden Markov Model based Part of Speech Tagger. We have achieved the accuracy of 93% in our test data.

Keywords

POS tagging, Hindi, HMM, Rule-based tagging, Viterbi Algorithm

Introduction

Part of Speech (POS) Tagging is the first step in the development of any NLP Application. It is a task which assigns POS labels to words supplied in the text. This is the reason why researchers consider this as a sequence labeling task where words are considered as sequences which needs to be labeled. Each word's tag is identified within a context using the previous word/tag combination. POS tagging is used in various applications like parsing where word and their tags are transformed into chunks which can be combined to generate the complete parse of a text. Taggers are used in Machine Translation (MT) while developing a transfer based MT Engine.Here, we require the text in the source language to be POS tagged and then parsed which can then be transferred to the target side using transfer grammar. Taggers can also be used in Name Entity Recognition (NER) where a word tagged as a noun (either proper or common noun) is further classified as a name of a person, organization, location, time, date etc.

Machine translation has expanded immensely, particularly in this period. Machine translation can be broken into seven main steps namely- token generation, analyzing morphology, lexeme, tagging Part of Speech, chunking, parsing, and disambiguation in words. For development of any NLP application, POS tagging is a necessary step. English language tagging is already available so our concentration was basically more on Hindi corpus POS tagging. Although there are many approaches available for POS tagging like rule- based POS tagging, lexical analysis etc. we have considered the HMM based POS tagging for our project because of its better results in other languages.

Contribution of this Project include:

- Splitting of sentences into tokens and distributing them.
- Part of Speech of different tokens detected.
- Presenting POS tagging list for the sentence.

Our HMM-Viterbi based POS tagger gives an accuracy of 93% in our test data.

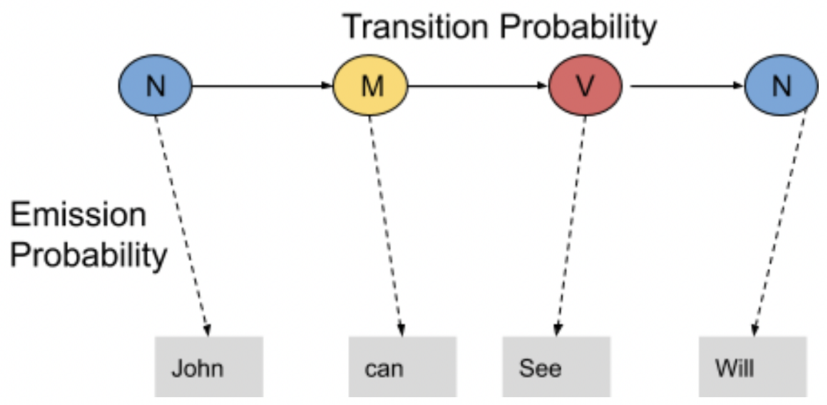
Literature Survey

Over the years, a lot of research has been done on POS tagging. Broadly, all the efforts can be categorized in three directions. They are: rule based approach where a human annotator is required to develop rules for tagging words or statistical approach where we use mathematical formulations and tag words or hybrid approach which is partially rule based and partially statistical. In the context of European languages POS taggers are generally developed using machine learning approach, but in the Indian context, we still do not have a clear good approach. In this project we discuss the development of a POS tagger for Hindi using Hidden Markov Model (HMM)

POS tagging efforts in Indian context dates back to 1990s with Bharti et. al. proposing a POS tagger for Hindi with morphological analyzer where a morphological analyzer would first provide a root word with its morphological features and a general POS category with can then be further classified using this generic pos category and morphological features. This approach was slightly modified by Singh et. al. where they used the results of morphological analysis for training using a decision tree based classifier. Their tagger gave an accuracy of 93.45%.

Hidden Markov Model

HMM (Hidden Markov Model) is a Stochastic technique for POS tagging. Hidden Markov models are known for their applications to reinforcement learning and temporal pattern recognition such as speech, handwriting, gesture recognition, musical score following, partial discharges, and bioinformatics. Let us consider an example proposed by Dr.Luis Serrano and find out how HMM selects an appropriate tag sequence for a sentence.



The **transition probability** is the likelihood of a particular sequence for example, how likely is that a noun is followed by a model and a model by a verb and a verb by a noun. This probability is known as Transition probability. It should be high for a particular sequence to be correct.

Now, what is the probability that the word Ted is a noun, will is a model, spot is a verb and Will is a noun. These sets of probabilities are **Emission probabilities** and should be high for our tagging to be likely.

Let us calculate the above two probabilities for the set of training sentences below.

- Mary Jane can see Will
- Spot will see Mary
- Will Jane spot Mary?
- Mary will pat spot.

N	**N**	**M**	**V**	**N**
Mary	*Jane*	*can*	*see*	*will*

N	**M**	**V**	**N**
Spot	*will*	*see*	*Mary*

M	**N**	**V**	**N**
Will	*Jane*	*spot*	*Mary*

N	**M**	**V**	**N**
Mary	*will*	*pat*	*spot*

In the above sentences, the word Mary appears four times as a noun. To calculate the emission probabilities, let us create a counting table in a similar manner.

Words	Noun	Model	Verb
Mary	4	0	0
Jane	2	0	0
will	1	3	0
spot	2	0	1
can	0	1	0
see	0	0	2
pat	0	0	1

Now let us divide each column by the total number of their appearances for example, ‘noun’ appears nine times in the above sentences so divide each term by 9 in the noun column. We get the following table after this operation.

Words	Noun	Model	Verb
Mary	4/9	0	0
Jane	2/9	0	0
will	1/9	3/4	0
spot	2/9	0	1/4
can	0	1/4	0
see	0	0	2/4
pat	0	0	1/4

From the above table, we infer that

The probability that Mary is Noun = 4/9

The probability that Mary is Model = 0

The probability that Will is Noun = 1/9

The probability that Will is Model = 3/4

In a similar manner, you can figure out the rest of the probabilities. These are the emission probabilities.

Next, we have to calculate the transition probabilities, so define two more tags <S> and <E>. <S> is placed at the beginning of each sentence and <E> at the end as shown in the figure below.

<S>	**N**	**N**	**M**	**V**	**N**	**<E>**
	Mary	*Jane*	*can*	*see*	*Will*	

<S>	**N**	**M**	**V**	**N**	**<E>**
	Spot	*will*	*see*	*Mary*	

<S>	**M**	**N**	**V**	**N**	**<E>**
	Will	*Jane*	*spot*	*Mary*	

<S>	**N**	**M**	**V**	**N**	**<E>**
	Mary	*will*	*pat*	*spot*	

Let us again create a table and fill it with the co-occurrence counts of the tags.In the above figure, we can see that the <S> tag is followed by the N tag three times, thus the first entry is

3.The model tag follows the <S> just once, thus the second entry is 1. In a similar manner, the rest of the table is filled.

	N	**M**	**V**	**<E>**
<S>	3	1	0	0
N	1	3	1	4
M	1	0	3	0
V	4	0	0	0

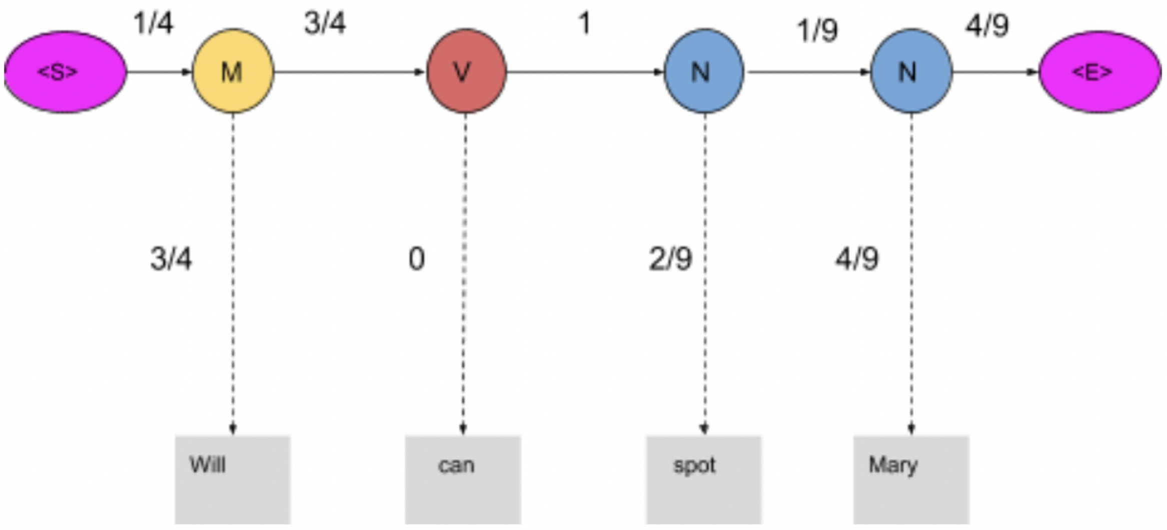
Next, we divide each term in a row of the table by the total number of co-occurrences of the tag in consideration, for example, The Model tag is followed by any other tag four times as shown below, thus we divide each element in the third row by four.

	N	**M**	**V**	**<E>**
<S>	3/4	1/4	0	0
N	1/9	3/9	1/9	4/9
M	1/4	0	3/4	0
V	4/4	0	0	0

These are the respective transition probabilities for the above four sentences. Now how does the HMM determine the appropriate sequence of tags for a particular sentence from the above tables? Let us find it out.

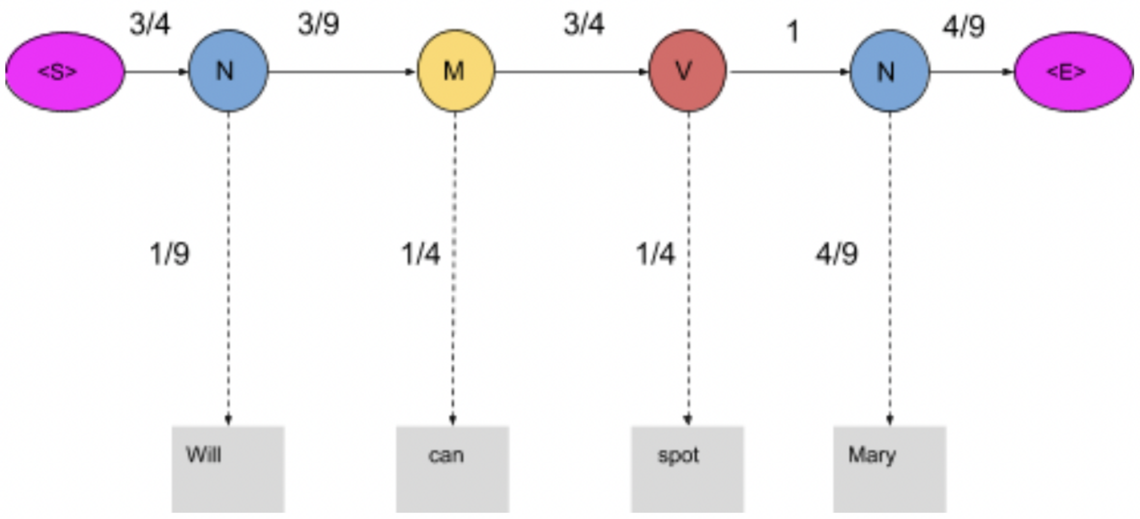
Take a new sentence and tag them with wrong tags. Let the sentence, ‘ Will can spot Mary’ be tagged as- Will{N} can{M} spot{V} Mary{N}

Now calculate the probability of this sequence being correct in the following manner.The probability of the tag Model (M) comes after the tag <S> is ¼ as seen in the table. Also, the probability that the word Will is a Model is 3/4. In the same manner, we calculate each and every probability in the graph. Now the product of these probabilities is the likelihood that this sequence is right. Since the tags are not correct, the product is zero.



1/4*3/4*3/4*0*1*2/9*1/9*4/9*4/9=0

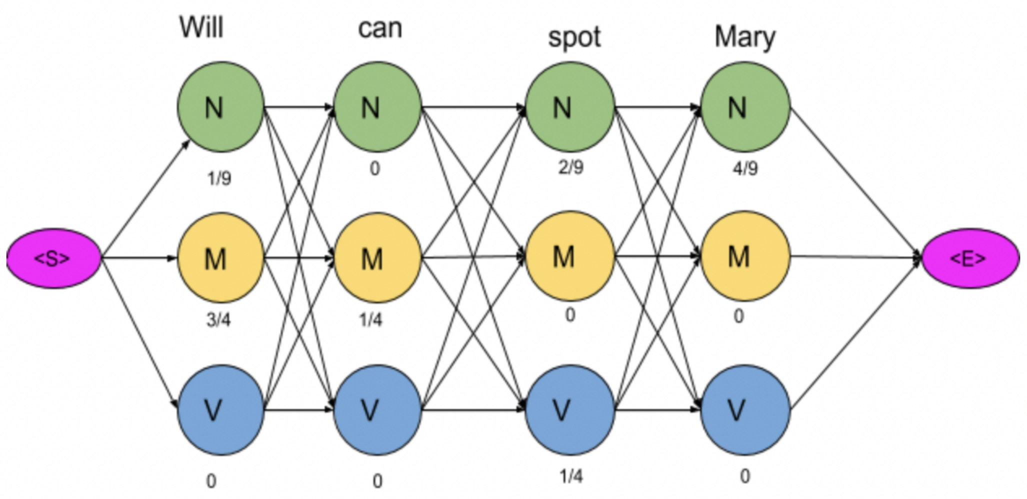
When these words are correctly tagged, we get a probability greater than zero as shown below.



Calculating the product of these terms we get,

3/4*1/9*3/9*1/4*3/4*1/4*1*4/9*4/9=0.00025720164

For our example, keeping into consideration just three POS tags we have mentioned, 81 different combinations of tags can be formed. In this case, calculating the probabilities of all 81 combinations seems achievable. But when the task is to tag a larger sentence and all the POS tags in the Penn Treebank project are taken into consideration, the number of possible combinations grows exponentially and this task seems impossible to achieve. Now let us visualize these 81 combinations as paths and using the transition and emission probability mark each vertex and edge as shown below.



Viterbi Algorithm

“The Viterbi algorithm is a dynamic programming algorithm for finding the most likely sequence of hidden states—called the Viterbi path—that results in a sequence of observed events, especially in the context of Markov information sources and hidden Markov models (HMM).”-wikipedia.

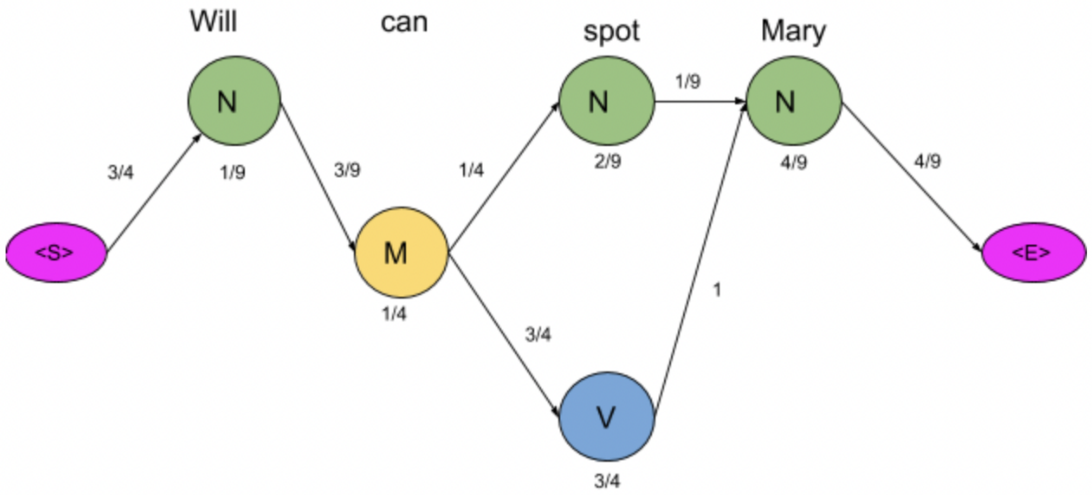
Basically Viterbi Algorithm is the optimization of HMM model.For our example, keeping into consideration just three POS tags we have mentioned, 81 different combinations of tags can be formed. In this case, calculating the probabilities of all 81 combinations seems achievable. But when the task is to tag a larger sentence and all the POS tags in the Penn Treebank project are taken into consideration, the number of possible combinations grows exponentially and this task seems impossible to achieve.

delete all the vertices and edges with probability zero, also the vertices which do not lead to the endpoint are removed. Also, we will mention-

Now there are only two paths that lead to the end, let us calculate the probability associated with each path.

<S>→N→M→N→N→<E> =**3/4*1/9*3/9*1/4*1/4*2/9*1/9*4/9*4/9=0.00000846754**

<S>→N→M→N→V→<E>**=3/4*1/9*3/9*1/4*3/4*1/4*1*4/9*4/9=0.00025720164**



Working of Tagger

Our Project includes a Hindi and part-of-speech tagger which has three fundamental steps. First, input Hindi and Marathi text is splitted into sentences. In the next step, the sentences are tokenized into words and the third step allocates part-of-speech tags to sentences.The data set used for training contains around 2000 sentences respectively.

A POS tagger based on HMM assigns the best tag to a word by calculating the forward and backward probabilities of tags along with the sequence provided as an input. The following equation explains this phenomenon.

$$P(t_i|w_i) = P(t_i|t_{i-1}).P(t_{i+1}|t_i).P(w_i|t_i)$$

$$P(t_i|t_{i-1}) = \frac{freq\,(t_{i-1},t_i)}{freq(t_{i-1})}$$

Each tag transition probability is computed by calculating the frequency count of two tags seen together in the corpus divided by the frequency count of the previous tag seen independently in the corpus. This is done because we know that it is more likely for some tags to precede the other tags.

Tags	**Meanings**
NN	Noun
NNP	Proper Noun
JJ	Adjective
DEM	Determiner / Demonstrative
INJ	Interjection
INTF	Adverb of Type Degree (Intensifier)
NEG	Negation Words
NST	Spatial Nouns
RP	Particles
PRP	Pronoun
RB	Adverb
RDP	Reduplications
AF	Quantifiers
VAUX	Verb Auxiliary
SYM	Symbol
PSP	Postposition
CC	Coordinating Conjunction
QC	Ordinals

We also computed the word likelihood probabilities using i.e. the probability of the word given a current tag.

$$P(w_i|t_i) = \frac{freq\ (t_i, w_i)}{freq(t_i)}$$

We train our hmm model using train.txt .Train.txt contain sentences that are poss tagged we get the dataset from universal.org.

Some lines from dataset is given as follows:-

यह/DEM एशिया/NNP की/PSP सबसे/INTF बड़ी/JJ मस्जिदों/NN में/PSP से/PSP एक/QC है/VM ।/SYM

इसे/PRP नवाब/NN शाहजेहन/NNP ने/PSP बनवाया/VM था/VAUX ।/SYM

इसका/PRP प्रवेश/NN द्वार/NN दो/QC मंजिला/JJ है/VM ।/SYM

जिसमें/PRP चार/QC मेहराबें/NN हैं/VM और/CC मुख्य/JJ प्रार्थना/NN हॉल/NN में/PSP जाने/VM के/PSP लिए/PSP 9/QC प्रवेश/NN द्वार/NN हैं/VM ।/SYM

पूरी/JJ इमारत/NN बेहद/INTF खूबसूरत/JJ है/VM ।/SYM

यहाँ/PRP लगने/VM वाला/PSP तीन/QC दिन/NN का/PSP इज्तिमा/NN पूरे/JJ देश/NN के/PSP लोगों/NN को/PSP आमंत्रित/JJ करता/VM है/VAUX ।/SYM

शौकत/NNP महल/NNP के/PSP सामने/NST बड़ी/JJ झील/NN के/PSP किनारे/NN स्थित/JJ वास्तुकला/NN का/PSP यह/DEM खूबसूरत/JJ नमूना/NN कुदसिया/NNP बेगम/NNP के/PSP काल/NN का/PSP है/VM जिन्हें/PRP गोहर/NNP बेगम/NNP भी/RP कहा/VM जाता/VAUX था/VAUX ।/SYM

यह/PRP हिंदू/NN और/CC मुगल/NN कला/NN का/PSP अद्भुत/JJ संगम/NN है/VM ।/SYM

यह/PRP भारत/NNP की/PSP अनूठी/JJ राष्ट्रीय/JJ संस्था/NN है/VM ।/SYM

मुख्य/JJ रूप/NN से/PSP यह/PRP प्रदर्शन/NN कला/NN और/CC दृश्य/NN कला/NN का/PSP केंद्र/NN है/VM ।/SYM

इसे/PRP चार्ल्स/NN कोरिया/NNP ने/PSP डिजाइन/NN किया/VM है/VAUX ।/SYM

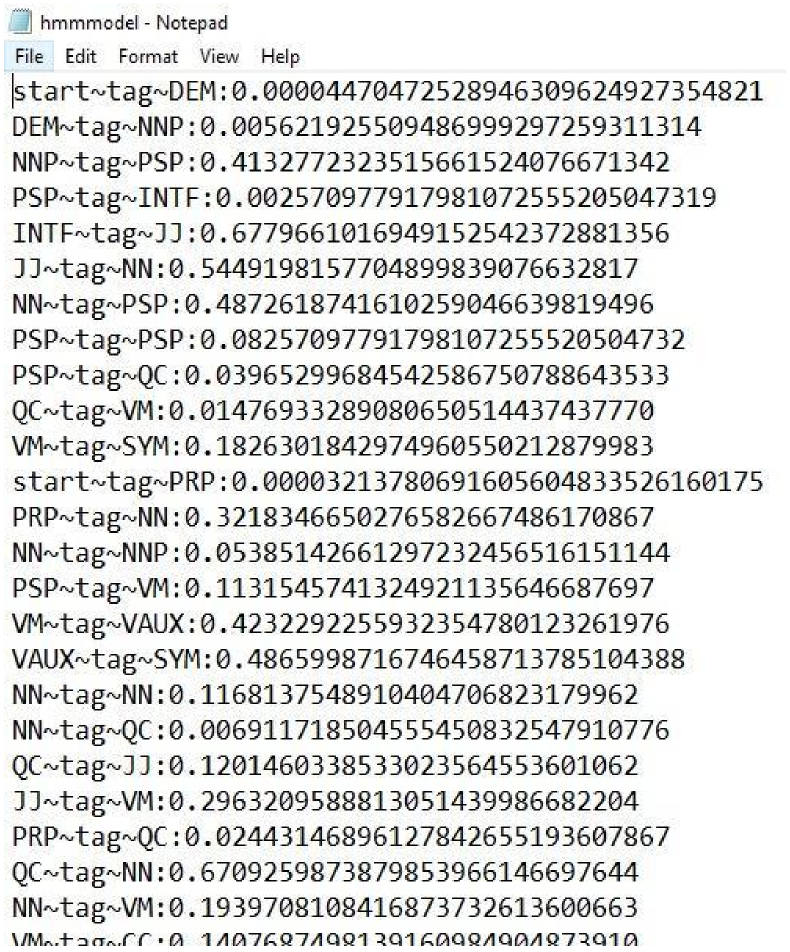
विशाल/JJ क्षेत्र/NN में/PSP फैले/VM इस/DEM भवन/NN के/PSP आस/NST -/SYM पास/NST का/PSP प्राकृतिक/JJ सौंदर्य/NN इसे/PRP और/QF भी/RP भव्य/JJ बनाता/VM है/VAUX ।/SYM

यहीं/PRP पर/PSP एक/QC कला/NN संग्रहालय/NN ,/SYM कला/NN दीर्घा/NN ,/SYM फाइन/NN आर्ट/NN के/PSP लिए/PSP कार्यशाला/NN ,/SYM एक/QC थिएटर/NN ,/SYM अंतरंग/JJ और/CC

बहिरंग**/JJ** ऑडिटोरियम**/NN ,/SYM** रिहर्सल**/NN** कक्ष**/NN ,/SYM** भारतीय**/JJ** कविताओं**/NN** का**/PSP** पुस्तकालय**/NN ,/SYM** शास्त्रीय**/JJ** और**/CC** लोक**/JJ** संगीत**/NN** संग्रहालय**/NN** भी**/RP** हैं**/VM ।/SYM**

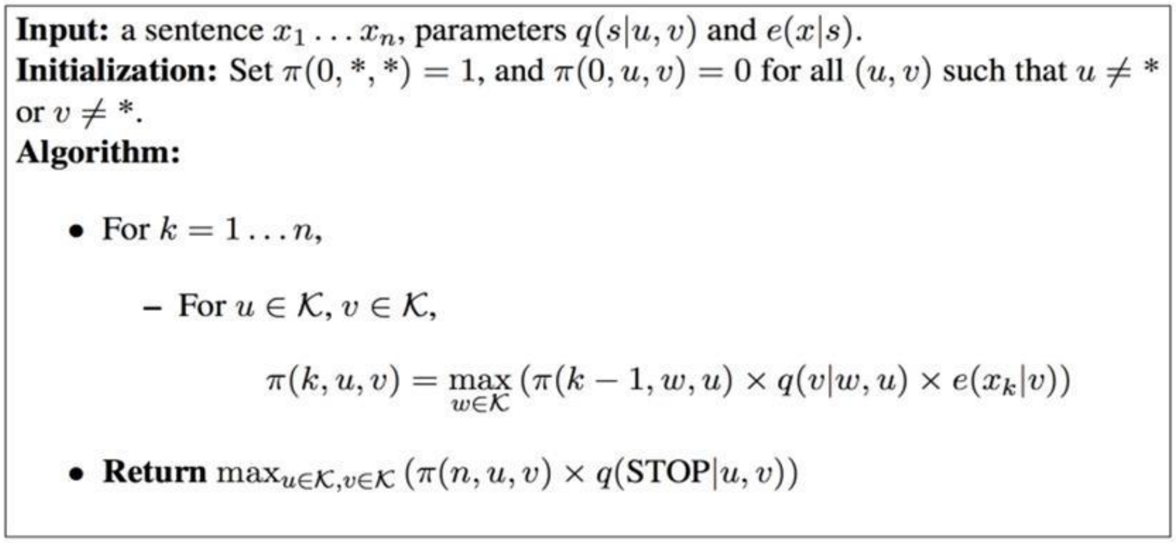
यह**/PRP** सोमवार**/NNP** के**/PSP** अलावा**/PSP** पूरे**/JJ** सप्ताह**/NN** दोपहर**/NN 2/QC** बजे**/NN** से**/PSP** रात**/NN 8/QC** बजे**/NN** तक**/PSP** खुला**/VM** रहता**/VAUX** है**/VAUX ।/SYM**

After training our model from traindata our model generate hmmmodel.txt that contain the transiton probability of tags.



```
start~tag~DEM:0.0000447047252894630962492735482l
DEM~tag~NNP:0.005621925509486999297259311314
NNP~tag~PSP:0.41327723235156615240766713 42
PSP~tag~INTF:0.0025709779179810725552050473l9
INTF~tag~JJ:0.677966101694915254237288l356
JJ~tag~NN:0.5449198157704899839076632817
NN~tag~PSP:0.4872618741610259046639819496
PSP~tag~PSP:0.0825709779179810725552050473 2
PSP~tag~QC:0.0396529968454258675078864353 3
QC~tag~VM:0.0147693328908065051443743777 0
VM~tag~SYM:0.1826301842974960550212879983
start~tag~PRP:0.0000321378069160560483352616017 5
PRP~tag~NN:0.32183466502765826674861708 67
NN~tag~NNP:0.0538514266129723245651615114 4
PSP~tag~VM:0.1131545741324921135646687697
VM~tag~VAUX:0.4232292255932354780123261976
VAUX~tag~SYM:0.4865998716746458713785104388
NN~tag~NN:0.1168137548910404706823179962
NN~tag~QC:0.0069117185045554508325479107 76
QC~tag~JJ:0.1201460338533023564553601062
JJ~tag~VM:0.2963209588813051439986682204
PRP~tag~QC:0.0244314689612784265519360786 7
QC~tag~NN:0.6709259873879853966146697644
NN~tag~VM:0.1939708108416873732613600663
VM~tag~CC:0.1407687498139160984904873910
```

We build our model using hmm and test our model using viterbi algorithm.



```
**Input:** a sentence $x_1 \dots x_n$, parameters $q(s|u,v)$ and $e(x|s)$.
**Initialization:** Set $\pi(0,*,*) = 1$, and $\pi(0,u,v) = 0$ for all $(u,v)$ such that $u \neq *$ or $v \neq *$.
**Algorithm:**
  • For $k = 1 \dots n$,
      – For $u \in \mathcal{K}, v \in \mathcal{K}$,
            $\pi(k,u,v) = \max_{w \in \mathcal{K}} (\pi(k-1,w,u) \times q(v|w,u) \times e(x_k|v))$
  • **Return** $\max_{u \in \mathcal{K}, v \in \mathcal{K}} (\pi(n,u,v) \times q(\text{STOP}|u,v))$
```

Some lines from our test data are as follows.

इसके अतिरिक्त गुग्गुल कुंड **,** भीम गुफा तथा भीमशिला भी दर्शनीय स्थल हैं ।

आधा किमी की दूरी पर भैरवनाथ मंदिर है **,** जहाँ केवल केदारनाथ के पट खुलने और बंद होने के दिन ही पूजन किया जाता है ।

भैरव का स्थान उत्तराखंड में क्षेत्रपाल अथवा भूमिदेव के रूप में महत्वपूर्ण है ।

यह सोनप्रयाग से **5** किमी आगे और केदारनाथ में **6** किमी पहले **(** पैदल **)** पड़ने वाला एक अत्यंत महत्वपूर्ण तीर्थ एवं विश्राम स्थल है ।

इसकी ऊँचाई केवल **1982** मीटर है ।

यहाँ गर्म पानी और ठंडे पानी के दो बहुत ही उत्तम कुंड हैं ।

यहाँ गौरी मंदिर है **,** जो कुछ छोटा है और प्राचीन नहीं है ।

जनश्रुति है कि इसी स्थान पर पार्वती ने महादेव को पाने के लिए तपस्या की थी ।

मंदिर में गौरी और पार्वती की धातु की मूर्तियाँ हैं ।

दूसरा मंदिर राधाकृष्ण का है **,** जिसे कृष्ण भक्तों का नूतन प्रयास समझा जाना ही उत्तम होगा ।

मंदिर के गर्भगृह में नारायण भगवान की सुंदर मूर्ति है ।

अन्य मूर्तियों में भू **-** देवी तथा लक्ष्मी की मूर्तियाँ उल्लेखनीय हैं ।

यहाँ अनेक कुंड भी हैं ।

Evaluation

Accuracy is the fraction of predictions our model got correct. Here, we can explain the accuracy as "total number of correctly handled words by the presented approach out of total input data." The range of the accuracy lies between 0 and 100 (in percentage).

Accuracy = number of correct tag/ total words

Wrong Predictions = 2409
Total Predictions = 35247
Accuracy is = 93.16537577666185 %