International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 06 Issue: 03 | March - 2022 ISSN: 2582-3930

PARTS OF SPEECH TAGGING USING VITERBI ALGORITHM

G Venkata Vaibhav[1], N Abhishek Vardhan[2], Ch Surendhar Reddy[3], B Yashwanth Kumar[4], M Kranthi Kiran[5].

[1,2,3,4]Student, Department of Computer Science Engineering, Gitam University ,Visakhapatnam.
[5]Assistant Professor, Department of Computer Science Engineering, Gitam University ,Visakhapatnam.

Abstract

Preprocessing words and phrases are required for each NLP activity since it facilitates subsequent work.

POS tagging is becoming increasingly popular these days. Text to speech, syntactic analysis, and machine translation all benefit from POS tagging, which is an effective form of preprocessing tagging. When it comes to POS taggers, they must be well-versed on the term. It is accessible to humans because describing it is simple. However, if the number of words is increased to a million, users will be unable to complete the POS tag. We present the Viterbi method in this study to assist computers in tagging lexical categories more effectively. The Viterbi algorithm use dynamic programming to solve problems. As we all know, the word is quite sensitive to its placement. The word's POS is connected to the words around it. We run simulations to see how Viterbi Algorithms operate in POS taggers and calculate accuracy.

Introduction

We studied the distinctions between numerous parts of speech tags such as nouns, verbs, adjectives, and adverbs in primary school. POS tagging or POS annotation is the process of associating each word in a phrase with the appropriate POS (part of speech). Word classes, morphological classes, and lexical tags are all examples of POS tags. POS tags provide a wealth of information about a word and its surroundings. Information retrieval, parsing, Text to Speech (TTS) applications, information extraction, and linguistic research for corpora are only a few of the jobs where they're used. They're also employed as a step between higher-level NLP activities like parsing, semantics analysis, translation, and more, making POS tagging an essential feature for sophisticated NLP systems.

Methodology

The technique of marking up a word in a text (corpus) as relating to a part of speech, depending on both its meaning and its context, is known as POS tagging. The statement "Where are you" is an example of an input. The output should be " Where/WRB are/VBP you/PRP," where "WRB" stands for "wh-abverb" and "VBP" stands for " verb present " and "PRP" stands for " personal pronoun."

The POS tagging issue, on the other hand, is not only a manual operation. In the 1980s, researchers began using hidden Markov models (HMMs) to decipher bits of speech.

Let us assume a finite set of words V and a finite sequence of tags K.

Then the set S will be the set of all sequence, tags pairs <x1, x2, x3 ... xn, y1, y2, y3, ..., yn> such that $n > 0$ $\forall x \in V$ and $\forall y \in K$.

$$\text{For any } \langle x_1 \dots x_n, y_1 \dots y_n \rangle \in \mathcal{S},$$

$$p(x_1 \dots x_n, y_1 \dots y_n) \geq 0$$

$$\sum_{\langle x_1 \dots x_n, y_1 \dots y_n \rangle \in \mathcal{S}} p(x_1 \dots x_n, y_1 \dots y_n) = 1$$

Given a generative tagging model, the function that we talked about earlier from input to output becomes

$$f(x_1 \dots x_n) = \arg \max_{y_1 \dots y_n} p(x_1 \dots x_n, y_1 \dots y_n)$$

Thus, for any given input sequence of words, the output is the highest probability tag sequence from the model.

The part-of-speech determinant is the location of the word and the two words preceding the word, and the tagging issue is turned to an HMMs problem. The Viterbi algorithm is an effective way to address problems like these.

The Viterbi method is a dynamic programming-based approach for solving sentence POS. As we all know, the word is quite sensitive to its placement. The word's POS is connected to the words around it. For example, the word "can" can be a verb, and it always has a verb after it. At the same time, it might be the noun, which is always preceded by an adjective.

$$p(x_1 \dots x_n, y_1 \dots y_{n+1}) = \prod_{i=1}^{n} q(y_i | y_{i-2}, y_{i-1}) \prod_{i=1}^{n} e(x_i | y_i)$$

The Nave Bayes formula is used in this formula. The letter x indicates the word, while the letter y signifies the portion of speech. Then p(y|x) denotes the likelihood that this word belongs to a specific category, which is known. However, it is difficult to locate. The Bayes formula is then used to calculate p(y|x) from p(x|y), which refers to the likelihood of a specific word knowing the part-of-speech of the word, and p(y) to derive p(y|x) [6].

Simulations

We can utilise them even more now that we've prepared the word and POS fundamental possibilities. For instance, suppose we obtain a phrase and want to identify the POS tagger it belongs to.

Table 1 is a fairly basic statement in English, yet it has a lot of meaning.

In addition, it's an excellent model. Where is "wh-abverb" ,are is " verb present " and you is " personal pronoun."

The Viterbi Algorithm determined the POS of this phrase.

Table 1

PROCESSING SENTENCE: where are you

For given sentence Viterbi algorithm is applied and the probabilities are shown in table 2. log is applied to probabilities to get accurate value. For “where “WRB has highest probability. And then for “are” VBP has highest probability among other part of speech tags. for “you” PRP which is personal pronoun has highest probability.

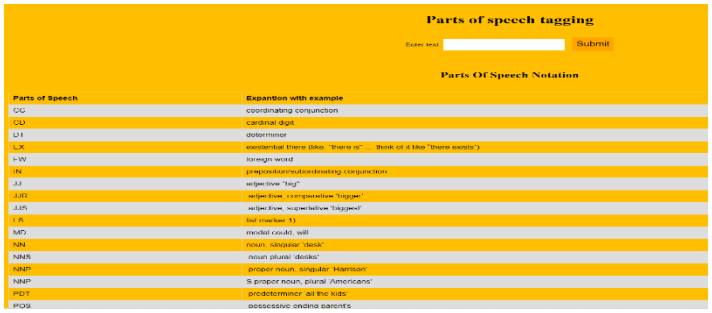
	where	are	you
--init--	-27.692103	-44.091005	-51.349272
CC	-33.984561	-38.880579	-45.040240
CD	-34.830583	-41.025856	-47.916410
DT	-36.441145	-45.931293	-48.213819
EX	-27.386550	-43.785453	-38.793625
FW	-24.900048	-38.866970	-41.648463
IN	-36.812681	-41.228649	-50.303961
JJ	-35.860597	-41.590521	-48.658748
JJR	-29.994563	-38.792063	-45.134339
JJS	-28.986167	-37.090770	-42.403341
LS	-21.787184	-38.186087	-45.444354
MD	-32.200921	-48.599823	-47.851390
NN	-37.411076	-41.137029	-47.910513
NNP	-36.663441	-35.353994	-47.483264
NNPS	-29.613911	-46.012814	-41.372886
NNS	-35.815647	-40.462599	-44.032782
PDT	-25.755943	-42.154846	-49.413113
POS	-31.962991	-41.453139	-47.102768
PRP	-33.350640	-39.797218	-28.770672
PRP$	-31.894422	-48.293325	-54.329391
RB	-34.498493	-39.307500	-44.563294
RBR	-28.795542	-38.285690	-45.543957
RBS	-26.132423	-42.531326	-42.880838
RP	-29.605731	-36.531853	-45.661499
SYM	-22.505526	-38.904429	-46.162695
TO	-33.847286	-41.546507	-50.595701
UH	-23.342211	-38.677720	-46.999380
VB	-34.182305	-38.724686	-48.629035
VBD	-34.427488	-40.047413	-49.790358
VBG	-33.029471	-39.224744	-47.581549
VBN	-33.627101	-39.404652	-47.157600
VBP	-32.684589	-23.657170	-46.868977
VBZ	-33.785112	-38.857407	-48.924888
WDT	-30.555754	-46.954656	-40.898286
WP	-29.369515	-45.768418	-40.173507
WP$	-24.298168	-40.697071	-47.955338
WRB	-16.398903	-38.666106	-43.527386

Using dynamic programming tags which resulted in lowest probability in last column is stored. This process is called backward chaining.

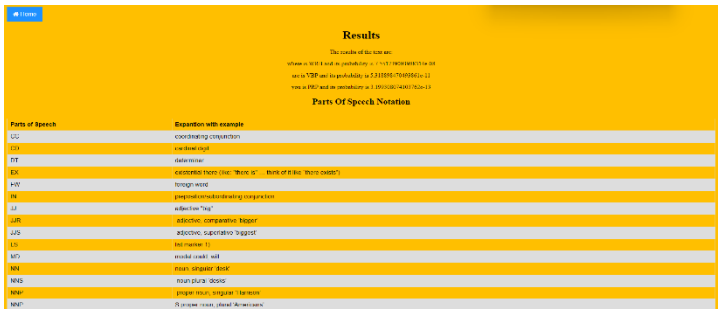
Building application:

Application is build using Python, Flask, HTML, CSS. Transition matrix and emission matrix are generated from the training data from Kaggle. When input is given to search box, input is directed to flask using GET method. Parts of speech of sentence is predicted using Viterbi algorithm which uses forword and backword propogation. Parts of speech along with its probability are directed to results page using POST method.

Pos tagging home page



Pos tagging results page



Conclusion

POS tagging is an effective preprocessing approach that is also useful in text to speech, syntactic analysis, and machine translation. It is accessible to humans since it is simple to describe. However, when the number of words is increased to a million, users are unable to complete the POS tag. As a result, we require automated machine complements. The Viterbi method is a dynamic programming-based approach for solving sentence POS. The location of the word is important to the term. The word's POS is connected to the words around it.

The Viterbi method use dynamic programming to compute the likelihood of a word in every conceivable POS and choose the best one as the final POS tagger. We can simply determine that the noun is clearly identified based on simulation data. Figures depict the word's performance in four distinct POS taggers. We'll look at each of them separately. These diagrams depict the POS Viterbi Algorithm's sentence detection details.

REFERENCES

[1] Gallier, “Evaluating Natural Language System,” Springer, vol. A24, pp. 529–551, April 1996.

[2] GD Forney, “The Viterbi algorithm,” Proceedings of the IEEE, 1973.

[3] J. Hagenauer, P. Hoeher, “A Viterbi algorithm with soft-decision outputs and its applications,” Proc. of the IEEE, vol. 61, no. 3, pp. 268-278, Mar. 73.

[4] G. Unger Boeck, "Channel Coding with Multilevel/Phase Signals", IEEE Trans. on Inf. Theory, vol. IT-28, no. 1, pp. 55-67, Jan. 82.

[5] J. B. Anderson, T. Aulin, C.-E. Sundberg, Digital Phase Modulation, New York: Plenum Publishing Co., 1986.