AINA

AI and Analytics

Volume 2 • Edition 2020-21

COVER STORY

ART IS MATH

Annual Analytics Magazine from the Students of PGDBA

Post Graduate Diploma in Business Analytics : Jointly offered by IIM Calcutta, ISI Kolkata & IIT Kharagpur

Parts of Speech Tagging using Hidden Markov Model

Prasun Kumar

In this article, we are going to learn one of the most important parts of a natural language processing pipeline, Parts of Speech tagging. Due to the complexity of the English language, it's very important that computers learn the context of each word. Parts of speech tagging is used to tag the parts of speech of the words in a sentence based on the context. For example, consider two sentences:

1. The computer is not able to understand languages because **it** is too dumb.
2. The computer is not able to understand languages because **it** is too complex.

What does **it** refer to in the above two sentences? In the first sentence, **it** refers to the computer, while in the other, **it** refers to the language English. In this example, the part of speech of **it** is the same, but due to a different context, the meaning changes. Take, for instance, the following examples:

1. Snorlax is sleeping
2. Sleeping is a boon

In these examples, the same word **sleeping** has been used as a verb (in the first sentence) as well as a noun (in the second sentence). So, it is important for a computer to understand the parts of speech (PoS) of a word. There are various methods of PoS tagging such as lookup tables, n-grams, Hidden Markov Model, and Viterbi algorithms.

Before understanding these methods, let's look at some of the terminologies. To start with the PoS tagging problem, we need to have a training set of many sentences in which we know the parts of speech of each word priorly. The collection of these sentences is known as text corpus.

Lookup tables and N-grams have some serious limitations. In lookup tables, each word will get tagged to one and only one part of speech each time, irrespective of the context. In the case of n-gram, it is possible that we will get some new combination of words in our test set, and thus n-gram will not be able to tag the PoS for these cases. These limitations make lookup tables and n-grams relatively less popular. Hidden Markov Model and Viterbi Algorithm take care of this problem, and we will understand how they work in this article.

Hidden Markov Model

We will take an example to understand the working principle of these methods. Let us consider a sentence: 'John may see Rob'. In this sentence, let us say that a way of tagging PoS is as follows: **John** is a Noun (N), **may** is a modal verb (M), **see** is a verb (V), and **Rob** is a noun (N).

<s>	John	may	see	Rob	<e>
	N	M	V	N	

Figure 1: Example of a sentence with parts of speech tags

Our aim is to calculate the probability associated with the above tagging. To find this out, we need two sets of probabilities: transition probability and emission probability. Transition probability tells us about the chance of occurrence of a part of speech after another part of speech, while emission probabilities tell us about the chance of occurrence of a particular word corresponding to a part of speech.

In the above example, transition probabilities include what is the probability that a Modal is coming after a Noun, a Verb is coming after a Modal and a Noun is coming after a Verb. Emission probabilities include the chance that a Noun will be the word **John**, and a Verb will be the word **see**, etc. For the correct tagging, we want overall probabilities (multiplication of all prob.) to be higher.

Emission Probabilities

<s>	Mary	Jane	can	see	Will	<e>
	N	N	M	V	N	
<s>	Spot	will	see	Mary	<e>	
	N	M	V	N		
<s>	Will	Jane	spot	Mary	<e>	
	M	N	V	N		
<s>	Mary	will	pat	Spot	<e>	
	N	M	V	N		

We need our training corpus to have all the PoS tags so that we can find emission and transition probabilities. Let us consider four sentences, as well as their parts of speech associated with each word. The example has been taken from NLP course at Udacity. <s> and <e> denote starting and ending tags.

We will find the probability of each word being a particular part of speech using the above information. For example, **Mary** is occurring 4 times as Noun, in the above corpus, and there are 9 occurrences of words which are Noun, so the probability that a Noun will be the word **Mary** is equal to 4/9.

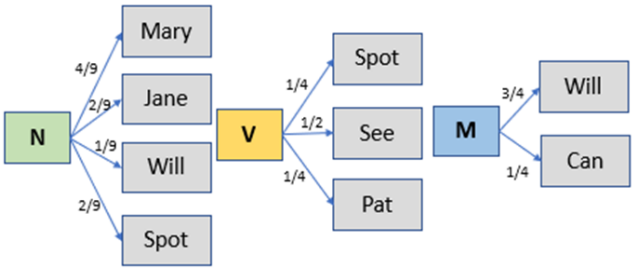

Figure 2: Emission Probabilities

In the same way, we find all the probabilities and represent them as follows, which is called emission probabilities.

Transition Probabilities

This is the set of probabilities of one part of speech following another. For example, in the above corpus, 'Noun followed by Modal' occurs three times (in first, second, and fourth sentences). In total, Noun is followed by Noun once, by Modal thrice, by Verb once and by end-of-sentence four times. Thus, the probability that Modal occurs after Noun is 3/9 = 1/3. In the same way, we calculate probabilities for all combinations, and summarize them in the following diagram, which represents transition probabilities:

	N	M	V	<e>
<s>	3/4	1/4	0	0
N	1/9	1/3	1/9	4/9
M	1/4	0	3/4	0
V	1	0	0	0

Figure 3: Transition Probability

Now we have both emission and transition probabilities. We will proceed to see them in action. The words which are there in the corpus are called observations because these are the things that we can observe. But, the parts of speech of each word are hidden to us and not directly observable, so we call them hidden states. There are 9 observations and 3 hidden states in our corpus. Each hidden state (PoS) is connected to every other hidden state, with the transition probability, and each hidden state is also connected to every observation (words) by emission probabilities. The following diagram describes this relation:

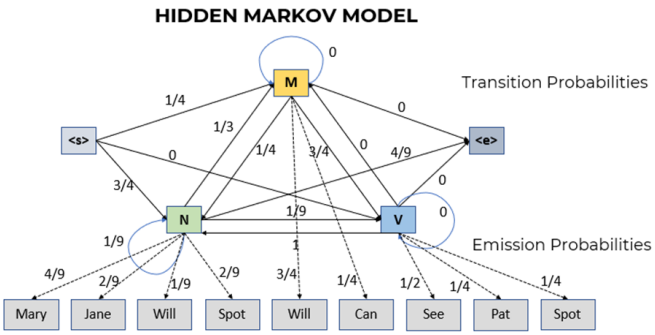

Figure 4: Hidden Markov Model

In the above diagram, values on solid arrows show the probability of a part of speech coming after another part of speech (transition prob.), while the numbers on the dashed arrow show the probability that a noun is the given word (emission probability).

The Hidden Markov Model can generate all sentences based on the sequence in which we travel from one state to another. Here state refers to the parts of speech N (noun), M (modal verb) and V (verb), start-of-sentence (<s>) and end-of-sentence (<e>). Let's consider an example where we want to generate the sentence 'Jane will spot Will.' Let us see in how many ways we can generate this sentence using the above model.

We will start from <s>. One of the ways is to reach Noun (N) with probability 3/4. We can pick the word **Jane** with probability 2/9, and can move to Modal (M) with probability 1/3, and can pick **will** with probability 3/4. We

can move to Verb (V) with probability 3/4, and can pick **spot** with probability 1/4, and can move to Noun (N) with probability 1. Then, we can choose **Will** with probability 1/9. Finally, we can reach the end-of-sentence with a probability 4/9. The above few sentences might be complex to understand, so let's have a look at the following diagram to understand the flow. This is one of the many ways to generate the sentence:

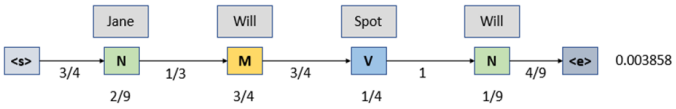

Figure 5: First possibility of occurrence of the above sentence

Moving from one state to another is independent of other states, so we can multiply all these probabilities to calculate the probability of the above combination of words and parts of speech. For the above case, we obtain 0.0003858 after multiplying all the probabilities. There are other ways in which we can generate the same sentences. Let's have a look at one of them:

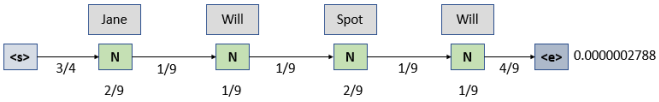

Figure 6: Second possibility of occurrence of the above sentence

The above possibility occurs when all of the words are Noun. This sentence makes no sense in the real world, and thus we have also obtained very low probability. We can check all possibilities in which the above sentence can be generated from our Hidden Markov Model and calculate the likelihood for each one of them. We will ignore those paths in which there is at least one 0 probability edge because these paths will not be possible. Apart from the above two already discussed paths, there are two more paths, whose likelihood is also shown below:

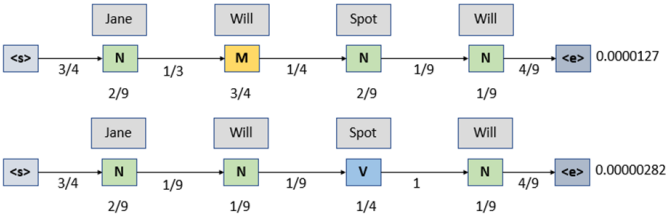

Figure 7: Third and fourth possibility of occurrence of the above sentence

Out of all 4 possibilities, we find that the likelihood of the first possibility is highest. Thus the PoS tags in that sentence will be reported as the correct PoS tags. Choosing that combination that has the highest likelihood is called the maximum likelihood principle, which is widely used in many machine learning algorithms. Based on the above values, we will report that, in the given sentence, **Jane** is Noun, **will** is Modal Verb, **Spot** is a Verb, and **Will** is a Noun. Thus, we are able to find the correct parts of speech of each word in a sentence.

In this article, we learned the application of Hidden Markov Models in Parts of Speech tagging in simple words. To further improve the Hidden Markov Model, we use the Viterbi algorithm, which uses dynamic programming to reduce the calculations required in the above method.