**OPTICAL CHARACTER RECOGNITION**

Deepthi Thalagundamatada

CSC 532- Design and Analysis of Algorithm

Dr. Gene Tagliarini

**ABSTRACT**

***The purpose of the project is to analyze the algorithms upon their implementation using Markov (HMM) and Naive Bayes. The implementation of the project gives insights into which algorithm provides more efficiency. OCR (Optical Character Recognition) has been used widely nowadays in which different techniques are also used to determine the characters in the image. The projecta .***

**BACKGROUND**

**OCR: Optical Character Recognition**

 OCR converts the document photo into machine-encoded text. OCR can be implemented using many popular tools such as Tesseract OCR and Cloud Vision. These tools specifically use AI and machine learning algorithms which train the machine with a set of inputs and the image is given as input which can determine the character in the image and convert that to the text. Text Recognition depends on various factors to have a good quality output.

In this predominant Digital Era, The documents are needed to be stored digitally. If the historical documents are stored digitally and still seeks the quality of storage and necessary access to it which image representation cannot provide. In Deed, they have to be converted to editable and some text where OCR comes to the picture. The Significance has improved though but the accuracy has not been increased even in the printed text identification and converting them to text.[3] There 2 significant problems which OCR faces with any kind of algorithm behind it.

* The quality of optical character recognition (OCR) on historical texts is often surprisingly low.
* Historical spelling variation represents a barrier for search even if texts are properly reconstructed.

Various Algorithms are implemented and are a test to have a good impression and conversion of the image to text. Predominantly Supervised and unsupervised algorithms of machine learning or AI were used. Nowadays deep neural networks which are structured and inspired by human mind process are used to process and determine the Texts in OCR too.

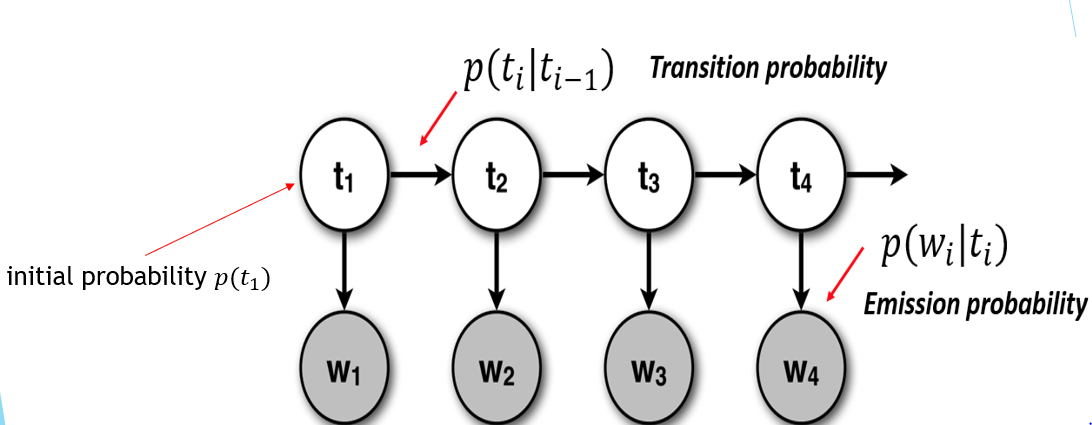
**INTRODUCTION**

There are several machine learning or AI algorithms which can be used to train and identify the image correctly. I have decided to use HMM (Vertibi) and Navie Bayes algorithms to compare and to identify the image.

**Definition of HMM and Bayes:**

**HMM(Viterbi): Hidden Markov model**

[1]” The HMM is based on augmenting the Markov chain. A Markov chain is a model that tells us something about the probabilities of sequences of random variables, states, each of which can take on values from some set”.



**Definition:** [2]A variant of a [finite state machine](https://xlinux.nist.gov/dads/HTML/finiteStateMachine.html) having a set of [states](https://xlinux.nist.gov/dads/HTML/state.html), Q, an output [alphabet](https://xlinux.nist.gov/dads/HTML/alphabet.html), O, transition probabilities, A, output probabilities, B, and initial state probabilities, Π. The current state is not observable. Instead, each state produces an output with a certain probability (B). Usually the states, Q, and outputs, O, are understood, so an HMM is said to be a triple, (A, B, Π).

**[2]Formal Definition:**

Hidden states Q = { qi }, i = 1, . . . , N.

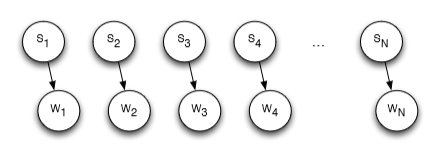
Transition probabilities A = {aij = P(qj at t +1 | qi at t)}, where P(a | b) is the conditional probability of a given b, t = 1, . . . , T is time, and qi in Q. Informally, A is the probability that the next state is qj given that the current state is qi.

Observations (symbols) O = { ok }, k = 1, . . . , M .

Emission probabilities B = { bik = bi(ok) = P(ok | qi) }, were ok in O. Informally, B is the probability that the output is ok given that the current state is qi.

Initial state probabilities Π = {pi = P(qi at t = 1)}.

**Naive Bayes:**



[4] Mathematically, Bayes' theorem gives the relationship between the probabilities of A and B, P(A) and P(B), and the conditional probabilities of A given B and B given A, P(A|B)and P(B|A). In its most common form, it is:

[5]



[4]

In the Bayesian (or epistemological) interpretation, probability measures a degree of belief. Bayes' theorem then links the degree of belief in a proposition before and after accounting for evidence

For proposition A and evidence B,

• P(A), the prior, is the initial degree of belief in A.

• P(A|B)/P(B), the posterior, is the degree of belief having accounted for B.

• P(B|A)/P(B)represents the support B provides for A.

**EXPERIMENTAL DESIGN**

As I have decided on HMM and Naive Bayes I need 3 kinds of probabilities which are

* Initial Probability P(S1)
* Transition probability P(Si+1|Si),
* Emission Probability P(Wi|Si)

Here the

* Hidden States - Actual textual letters
* Observed variables – Characters in images
* A letter transition if it is not in the trained dictionary I substitute by small probability math.log(1.7976931348623157e-308) to avoid the underflow of floating point numbers.
* **Initial State Distribution:**

P(S1)(initial state distribution) that is the prior of "letter" at position 1, I have formed the dictionary where all the letters of the first position in each sentence are collected and I have put them as a key in the dictionary (being ProbOfLetter1AsDict {} )of the logarithm of probabilities. So now I have letters as keys and their corresponding logarithm of probabilities of them appearing in the first position.

For the letters other than appearing in the first position, all Later letters after first letter position are collected and probabilities of the same are found and stored in the dictionary ProbOfLetter\_AllAsDict {}. I have made this distinction to get better accuracy

* **Transition probabilities:**

P(Si+1|Si): (For HMM)

Since P(Si+1|Si) = P(Si+1 , Si)/P(Si), I am taking probability of two 'letters' appearing in sequence, that is P(Si+1 , Si)[for example [a-r], P(a,r) is the probability of a-r pair appearing] are stored as the transition probabilities(in the form of log(P)),transitionProbs{} dictionary.

log(P(Si+1|Si)) = log(P(Si+1 , Si))-log(P(Si)),

For example,

log(P('a'/'r')) = log(P('a' , 'r'))-log(P('r')) =

ProbOfLetterSeqAsDict ["a\_r"]- ProbOfLetter\_AllAsDict ["r"]

* **Emission probabilities:**

P(Observed Character Image|letter) :

I have decided to consider F - Scores by matching the image at the pixel level, I have the following table similar to true positives, true negatives, false positives, and false negatives

|  |  |  |
| --- | --- | --- |
|  | Actually Lit | Actually Unlit |
| Test Pixel Lit | litMatched | unlitUnmatched |
| Test Pixel Unlit | litUnmacthed | unlitMatched |

It is considered as a lit pixel where there is '\*'

It is considered as an unlit pixel where there is ' '

For F Score I have to calculate precision and recall for lit pixel as matching a lit pixel is more important

precision = number of litMatched pixels / number of litMatched +number of unlitUnmatched

recall = number of it matched/ number of 'litUnmatched'+ number of 'litMatched'

**F Score is the harmonic mean of precision and recall**

i.e

emissionGivenActLetter = (((1/precision) + (1/recall))/2)^(-1)

emissionGivenActLetter = emissionGivenActLetter \*\* TOTAL\_PIXELS

FScore is the accuracy of the match of the test image with the train image or the probability of each pixel at a letter matching the test image's pixel so I have to raise it to the power of Total Number of pixels assuming a Naive Bayes assumption for the pixels to get the overall emission probability.

**Naive Bayes Approach:**

In Naïve Bayes approach, I am making a naive assumption that each letter at a give position is independent of any other positions.

That is, P(S1,S2,S3,S4,....,Sn/O1,O2,O3,...,On) = P(O1/S1)\*P(S1)\*P(O2/S2)\*P(S2)....\*P(On/Sn)\*P(Sn), so at every ith position, I find out the "si" leading to the highest probability of P(Si = si/W).and output the sequence of the letters.

**HMM, Model By Viterbi:**

Since each state is dependent on the previous state I would need to calculate the sequence "s1,s2,s3,s4,....,sn" such that the probability P(Si = si|O) leads to the maximum.

That is,

P(S1,S2,S3,S4,....,Sn/O1,O2,O3,...,On)=

P(O1/S1)\*P(S1)\*P(O2/S2)\*P(S2/S1)....\*P(On/Sn)\*P(Sn/Sn-1),

so at every ith position, I have to find the "si" which leads to the highest probability of P(Si = si/W).

For every state at position t, I must find the max of previous states' probability times the transition probability from the previous state to the current state and multiply with emission probability of the current state.

Now I have to store this probability in a dictionary dictOfPostions[position][letter] in the form of log(P), after the calculation of the probability at each position for each state,

to backtrack I append in the form of string

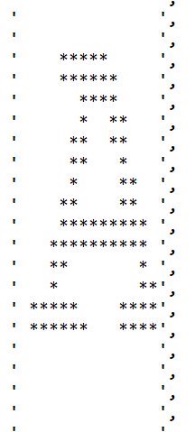
maxPaths[position][queryletter] = maxPaths[position-1][prevStateLeadingToMax] + "=>" + queryletter,

where queryletter is the current letter.( queryletter at position "t"), such that I do not have to actually traverse or backtrack through the entire trace, rather at the last position find the letter that had the maximum probability and pulls the string that maxPaths [LastPosition][maxLetter] had. Hence saving the requirement of return loop.

**Algorithm:**

STEP 1: Load the Training Image letter by letter as a list of strings where each dark pixel in the image is represented by \* and the white pixel is represented with space

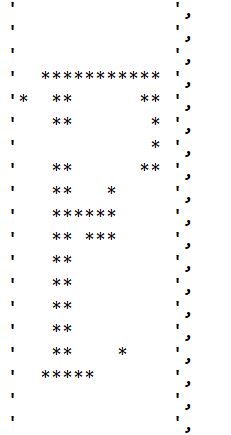
For Example Character A in String of Pixels is represented as below:

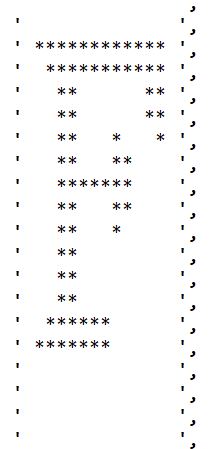


STEP2: From the brown corpus Transition probabilities and starting probabilities are calculated as discussed previously

STEP3: Load the Test image Letter by Letter and store it a string of pixels representation as in the step1

STEP4: Emission Probability that is the probability of test letter given an actual letter for Example

Test F Train F



### given

We match each test pixel to the train and form the lit and unlit table as discussed above.

STEP 5: After getting the Probabilities HMM and Naive Bayes is applied to identify the letters.

### FINDINGS

**[5]Application of the OCR:**

## Practical OCR Applications

In recent years, **OCR** (**Optical Character Recognition**) technology has been useful for the entire [spectrum of industries](http://www.cvisiontech.com/industries.html), revolutionizing the document management process. OCR has enabled scanned documents to become more than just image files, turning into fully searchable documents with text content that is recognized by computers. After using OCR, people work became hazel free and they no longer need to manually retype important documents when entering them into electronic databases.

## Banking

OCR reduced the human involvement so reducing the long queues. A check can be inserted into a machine, the writing on it is scanned instantly, and the correct amount of money is transferred. This technology has nearly been perfected for printed checks, and is fairly accurate for handwritten checks as well, though it occasionally requires manual confirmation.

## Legal

In the legal industry, OCR has also been a significant movement to digitize paper documents. OCR further simplified the process by making documents text-searchable, which made it easier to locate and work with once in the database. Legal professionals now have fast, easy access to a huge library of documents in electronic format

## Healthcare

Healthcare has also seen an increase in the use of OCR technology to process paperwork. To keep up with all of this information which are large forms of patients and medical records, it is useful to input relevant data into an electronic database that can be accessed as necessary.

## OCR in Other Industries

OCR is widely used in many other fields, including [education](http://www.cvisiontech.com/industries/education.html), [finance](http://www.cvisiontech.com/industries/legal.html), and [government agencies](http://www.cvisiontech.com/industries/government.html). OCR has made countless texts available online, saving money for students and allowing knowledge to be shared. As technology continues to develop, more and more applications are found for OCR technology, including increased use of handwriting recognition. Furthermore, other technologies related to OCR, such as barcode recognition, are used daily in retail and other industries.

**RESULTS:**

**Specifications of the letters in the image are set to:**

Width :14

Height: 25

Font: Courier New M

**TEST 1:**

**Input image:** 

**Output:** 

**TEST2:**

**Input image: **

**Output :** 

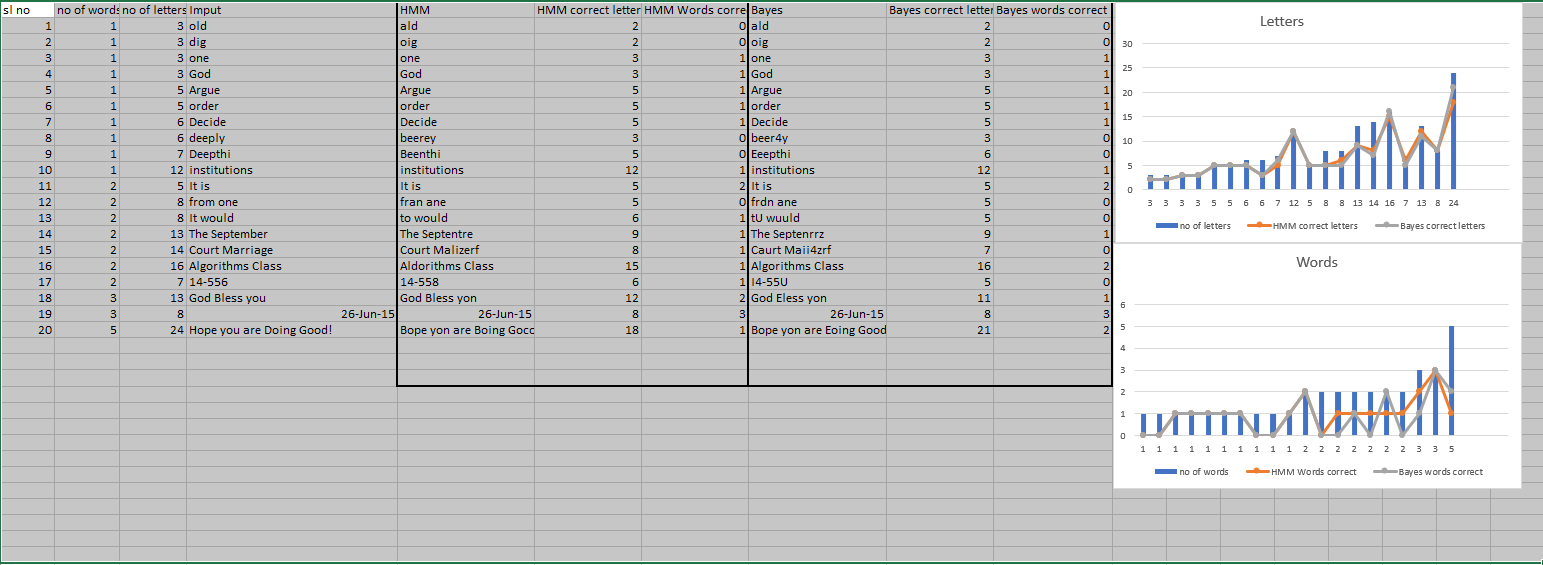
**Test 3:**

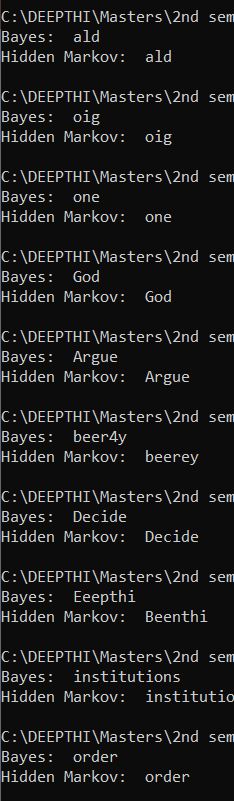
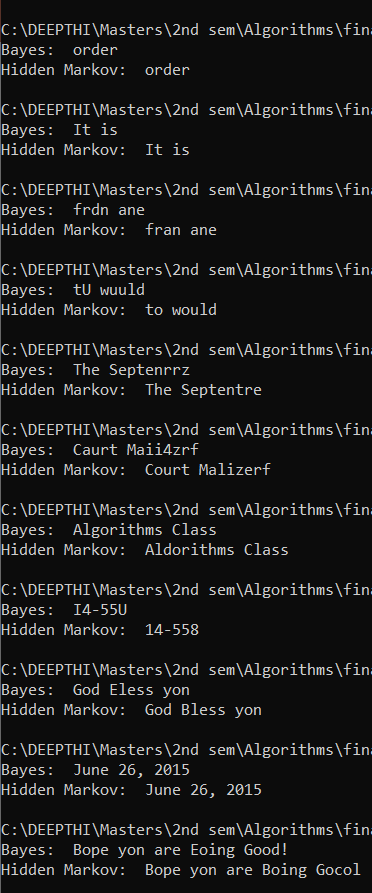
**Input Image:**

**Output:** 

**Test4:** 

**Input Image:** 

**Outputs:** 

****

### CONCLUSIONS

The Project represents the small part of OCR wherein a lot of work needs to be to identify the accuracy and sustainability. HMM(Markov) and Bayes do the fair job. But the Accuracy of the HMM is more than Bayes. If the image contains noise HMM do the fair job which gives more accuracy.

### FUTURE WORK

### My future work will be testing this model on training different languages, number of inputs making, it more precise and accurate in identifying the letters in the image.

### REFERENCES

# References

[1]<https://web.stanford.edu/~jurafsky/slp3/A.pdf>

[2] <http://www.shokhirev.com/nikolai/abc/alg/hmm/hmm.html>

And <https://xlinux.nist.gov/dads/HTML/hiddenMarkovModel.html>

[3] <https://doi.org/10.1016/j.patcog.2012.10.002>

[4] <https://www.immagic.com/eLibrary/ARCHIVES/GENERAL/WIKIPEDI/W1120615B.pdf>

[5] <http://www.cvisiontech.com/reference/general-information/ocr-applications.html>