Named Entity Recognition for Hindi Text

Gurjit Singh	K Kavitha	Pooja Singh	Satyam Ankur
184101017	184101018	184101028	184101030

1 Abstract

Hindi is the most used language in India so doing Machine Learning related tasks(such as chatbot, text summarization,question and answering, text prediction, etc) in Hindi dataset shall prove to have a vast outreach. Sequence Tagging is one of the steps performed in doing Natural Language Processing(NLP) related tasks. Hence we decided to do **NER for Hindi text.**
India is known for its traditional medicinal systems—Ayurveda and Siddha. There also exist many home remedies as well to cure small diseases. We have collected health dataset which includes diseases, its symptoms ,some home remedies or the traditional medicines for the cure.

Named Entity Recognition(NER) is a sub-task of information extraction that seeks to locate and classify named entities in text into some predefined tags such as location, organization,unit of time, etc.. NER plays a very important role in other tasks such as Identification of relationships, scenario template Production, Question, and answering, etc..

NER is basically a classification problem and Machine learning algorithms are widely used to classify words into different classes. **Neural Network model** is the first choice that comes to one's mind when it comes to classification problem. That's why we decided to use Probabilistic neural network model. And since NER is also a Sequence Tagging problem we have used **Conditional Random Field**(CRF) model. This work can further be extended to develop a system that gives remedies for the asked disease or can suggest disease based on the symptoms (question and answering).

2 Introduction

Natural Language Processing(NLP) is a field of computer science and linguistics concerned with the interactions between computers and human (natural) languages.Various sub problems in NLP include speech segmentation, text segmentation, part of speech tagging, NER tagging, word sense disambiguation,syntactic ambiguity etc..NLP used in various ways such as:

- Information Extraction
- Automatic text Summarization
- Machine Translation
- Natural Language Generation
- Natural Language Understanding

Named Entity Recognition(NER) is a subtask of information extraction that seeks to locate and classify named entities in text into some predefined tags such as location, organization,unit of time, etc..NER has many applications such as information retrieval,Machine

Translation, Question answer generating system, Spell Checker, Information Extraction etc.

Our plan for this project was to do NER for Hindi dataset and we chose Hindi Health dataset. Our work can furthe be extended to develop systems that answers the disease based on symptoms or remedies for the asked disease. Hindi data set is difficult to work with. Some of the difficulties while working with hindi text arises due to:

- Lack of Captilasation of words to indicate the Named Entity or start of sentence.
- Hindi is morphologically rich language hence determining the root word becomes difficult.
- Hindi text has a lot of ambiguity between common and proper noun.
- There is lack of resources, as web mostly has list of Named entities in English.

NER is a Sequence Tagging Problem.The task of Sequence Tagging is to map a **sentence** x_1 ... x_n to a **tag sequence** y_1 ... y_n.

INPUT: Profits soared at Boeing Co., easily topping forecasts on Wall Street, astheir CEO Alan Mulally announced first quarter results. OUTPUT: Profits/NA soared/NA at/NA Boeing/SC Co./CC ,/NA easily/NA topping/NA forecasts/NA on/NA Wall/SL Street/CL ,/NA as/NA their/NA CEO/NAAlan/SP Mulally/CP announced/NA first/NA quarter/NA results/NA ./NA

Example: A Named-Entity Recognition Example. The input to the problem is a sentence. The output is a sentence annotated with named-entities corresponding to companies, location, and people.

Once this mapping has been performed on training examples, we can train a tagging model on these training examples. Given a new test sentence we can then recover the sequence of tags from the model, and it is straightforward to identify the entities identified by the model.

We have worked on NER for hindi text as a supervised learning Problem.The set-up in supervised learning problems is as follows.

We assume training examples($x^{(1)}$,$y^{(1)}$) ... ($x^{(m)}$,$y^{(m)}$), where each example consists of an input $x^{(i)}$paired with a label $y^{(i)}$. We use X to refer to the set of possible inputs, and Y to refer to the set of possible labels. Our task is to learn a function $f : X \rightarrow Y$ that maps any input x to a label f(x). There can be two ways for determining f(x):

1. **Conditional Mode**l:
   In this approach model is defined as conditional probability P(y/x) for any pair (x,y). **Maximum Entropy Markov Model(MEMM) and Conditional Random Field(CRF) are conditional model.**

2. **Generative Model**: In Generative Model the model is estimated as joint probability P(x,y) over (x,y) pairs. Parameter estimation is done from training example. **Hidden Markov Model is a Generative Model.**

2.1 Conditional Random Field(CRF)

Notation: For convenience, $\bar{x}$ refer to an input sequence $x_1...x_m$, and $\bar{s}$ to refer to a sequence of states $s_1...s_m$.The set of all possible states is again S;the set of all possible state sequences is

S^m. In CRF we build a model of

$$P(s_1 ... s_m | x_1 ... x_m) = P(\bar{s}/\bar{x})$$

First key idea in CRFs will be to define a feature vector

$$\Phi(\bar{s},\bar{x}) \in \mathbb{R}$$

that maps an entire input sequence $\bar{x}$ paired with an entire state sequence $\bar{s}$ to some d-dimensional feature vector and is referred as "Global" feature vector and is defined by:

$$\Phi(\bar{s},\bar{x}) = \sum_{j=1}^{m} \phi(\bar{x}, j, s_{j-1}, s_j)$$

i.e. Φ_k is calculated by summing the "local" feature vector ϕ_k over the m different state transitions in s_1 ... s_m.
The model is defined by:

$$p(\underline{s}|\underline{x};\underline{w}) = \frac{\exp(\underline{w} \cdot \underline{\Phi}(\underline{x},\underline{s}))}{\sum_{\underline{s}' \in \mathcal{S}^m} \exp(\underline{w} \cdot \underline{\Phi}(\underline{x},\underline{s}'))}$$

where $\bar{w}$ is a parameter vector.
Estimation of $\bar{w}$:The regularized log-likelihood function is

$$L(\underline{w}) = \sum_{i=1}^{n} \log p(\underline{s}^i|\underline{x}^i;\underline{w}) - \frac{\lambda}{2}||\underline{w}||^2$$

parameter are then estimates as:

$$\underline{w}^* = \arg\max_{\underline{w} \in \mathbb{R}^d} \sum_{i=1}^{n} \log p(\underline{s}^i|\underline{x}^i;\underline{w}) - \frac{\lambda}{2}||\underline{w}||^2$$

Many sophisticated optimization methods are used for finding optimal $\bar{w}*$: one common to choice is to use L-BFGS, a quasi-newton method. Now given input sequence $\bar{x}$ = x_1,x_2,...x_m, the task is to find the most likely underlying state sequence under the model, i.e.

$$\arg\max_{\underline{s} \in \mathcal{S}^m} p(\underline{s}|\underline{x};\underline{w})$$

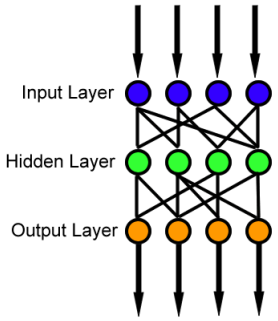
This expression can be simplified to:

$$\arg\max_{\underline{s} \in \mathcal{S}^m} \sum_{j=1}^{m} \underline{w} \cdot \underline{\phi}(\underline{x}, j, s_{j-1}, s_j)$$

Hence finding the most likely sequence under the model is equivalent to finding the sequence that maximizes the expression above.

2.2 Neural Network Model

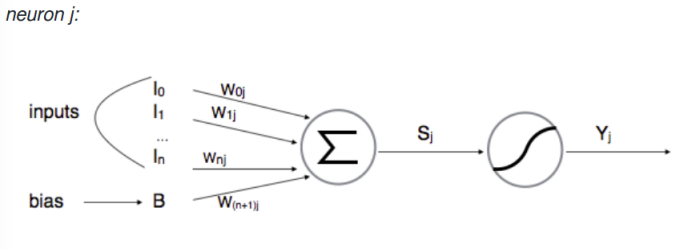
Neural Network is a computation model patterned after the operation of neurons in the human brain. It is life a artificial Human nervous system for the receiving processing, and transmitting information in computer science.



Neural Network has 3 different layer: 1.Input Layer(All the inputs to the model are fed through this layer)
2.Hidden Layer(more than one layer can be there , used for processing the input received from input layer)
3.Output Layer(After processing output made

available in output layer and it is the last layer of the network)

Neurons are the basic unit of the neural network. Each layer consist of neurons. Every neuron in i_{th} layer connected with every neuron in the $(i+1)_{th}$ layer.When a neuron activates , it accumulates all its incoming inputs. The main thing about neurons is that they can learn. All the connection between the neurons are known as weights(parameters).



As you can see they have several inputs, for each input there's a weight (the weight of that specific connection). When the artificial neuron activates, it computes its state, by adding all the incoming inputs multiplied by its corresponding connection weight. But neurons always have one extra input, the bias, which is always 1, and has its own connection weight. This makes sure that even when all the inputs are none (all 0s) there's gonna be an activation in the neuron.

$$s_j = \sum_i w_{ij}.y_i$$

where y_i is all the inputs (bias included)

$$y_j = f_j(S_j)$$

After computing its state, the neuron passes it through its activation function, which normalizes the result (normally between 0-1). Component of Neural Network:

1.)Activation function: In artificial neural networks, the activation function of a node defines the output of that node, or "neuron," given an input or set of inputs. This output is then used as input for the next node and so on until a desired solution to the original problem is found.

There are different activation function available. The one we are using is Relu.The formula for the the Relu is given below:

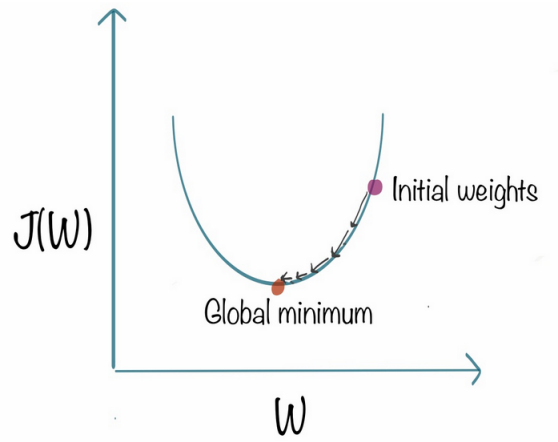
$$A(x) = \max(0,x)$$

The ReLu function is as shown above. It gives an output x if x is positive and 0 otherwise

2.Loss function:It's a method of evaluating how well your algorithm models your dataset. If your predictions are totally off, your loss function will output a higher number. If they're pretty good, it'll output a lower number. Cross_entropy_loss is a frequently used loss function in classification problem It's just a straightforward modification of the likelihood function with logarithms.

$$H(p,q) = -\sum_x p(x)\,\log q(x).$$

3.Gradient Descent:Gradient Descent is one of the most popular and widely used algorithms for training machine learning models. Gradient descent is an iterative method. In Gradient descent we start with some random set of values(weights and biases), and improve them slowly such that cost is minimized.



When we chose this topic for our Project we found that not much of work is done in NER for Hindi text and even if some work is already done nothing is done with a complicated and useful dataset.

Earlier the existing system just gave the model for the NER giving good accuracy but never shared the internal. The dataset used for training or testing contains mostly words that have the "other" tag or the count of words that belong to one of the "Tag" class is low compared to the "Other Tag" class. So even if the words that actually needs to be predicted correctly are predicted incorrectly could lead to good accuracy of the model.We have done further analysis of the obtained results and have done critical review of the CRF model.

Since NER is a prior step in NLP task like question-answering, text summarization etc.. This work can further we extended to develop a system that on asking remedies for a disease can recommend some or when given the symptoms can suggest the diseases possible etc..

3 Methods

The tags we have considered for NER of the dataset are:
Consumable as C
Disease as D
Person as P
Symptom as S
Others as O

Preparing Training Dataset:

In order to prepare the dataset for training, we need to label every word (or token) in the sentences to be either irrelevant or part of a named entity. We prepared the training dataset with a piece of code which removes punctuations, split text into sentences based on poorna viram. The code then splits each sentences in words and search the words in list of proposed tags and if found write in a file. So our training file contains data in the format: First it contains a sentence in one line and then in the next line it contains the words of the sentence which belong to any one of four tags.

लैवेंडर तेल की खुशबू माइग्रेन में बेहद प्रभावी होती है
C:तेल D:माइग्रेन
पुदीना सिर के तनाव को कम करने में माहिर औषधि है
S:तनाव C:औषधि
ज्यादा जलन भी महसूस हो सकती है
S:जलन

Example of Input

Generating Features: The features that will be useful in the training process depends on the task at hand. The features we used for a word contain:
1.)bias as 1
2.)whether the word is digit or not
3.)category(POS tag) of word
4.) Frequency of word
5.)whether it is BOS(1) or not(0)
6.)whether word is EOS(1) or not(0)

7.) Features words (except BOS word and EOS)contain features(word,whether digit or not, frequency and category) of word before and after it.

Training the Model Using CRF In pycrfsuite, a CRF model in can be trained by first creating a trainer, and then submit the training data and corresponding labels to the trainer. After that, set the parameters and call train() to start the training process.

```
trainer.set_params({
    'c1': 1.0,  # coefficient for L1 penalty
    'c2': 1e-3,  # coefficient for L2 penalty
    'max_iterations': 50,  # stop earlier
    # include transitions that are possible, but not observed
    'feature.possible_transitions': True
    })
```

Example for setting the Parameters for CRF before begining training.

Features Set for Neural Network Model CRF model is sentence based while Neural model is word based.As already mentioned that each word corresponds to 14 features in CRF model(except BOS and EOS). In neural model to extract features for a single word we have considered 3 words i.e the target word, its previous word and its next word.Hence in Neural model a word is represented as a 42 dimensional feature vector(these features contain target word and its context).Furthe these features are converted into numerical values which are then given to input layer. As we have 5 classes the output layer consists of 5 neurons, input layer consists of 42 neurons and also network has 3 hidden layers with 500 neurons each.

4 Experiment Design

Dataset: Hindi Health Dataset
Our dataset consists of Hindi Text related to diseases, their symptoms, prevention and remedies. We preprocess the dataset by first dividing it into training and testing set. In each set we split the sentences based on the delimiter - Poorna Viram - used in Hindi to indicate sentence end ("—"). For each word in a sentence we identify whether it belongs to either of the Named tags i.e; Consumable, Disease, Person or Symptom. If the word does not belong to any of the labels, we do not add a tag to it (assuming others). After preprocessing, each line along with their tags are sent to the model.
Model parameters for CRF:
minfreq: minimum occurence of each word :0.0
c1 : coefficient of L1 penalty :1.0
c2 : coefficient of L2 penalty : 1e-3
max-iterations : Number of times the model is trained

5 Result and Discussions

We calculated the precision, recall and f1 score for each of the named entities

$Precision = \frac{Truepositive}{TruePositive+FalsePositive}$

$Recall = \frac{TruePositive}{TruePositive+FalseNegative}$

$F1Score = \frac{2*Precision*Recall}{Precision+Recall}$

Tags	Precision	Recall	F1 Score
Consumable	1.0	0.770833	0.8706
Person	1.0	1.0	1.0
Symptoms	1.0	0.14285	0.25
Disease	1.0	0.5	0.66666

Average F1 score of our Conditional Random Field model : **0.6968**
For our Neural Network Model we obtained two accuracies :
Without Class Rebalancing:
Model Accuracy = **55.21**
After Class Rebalancing :
Model Accuracy = **78.85**

6 Conclusion

In this project we worked on a challenging task of performing Named Entity Recognition task on Hindi Text. We used Hindi Health Dataset available on kaggle[4]. The dataset was annotated with four tags namely Consumable, Person, Symptoms and Diseases. The selection of the dataset was based on the assumption of wide range of applications in hindi medical studies. We used two models for comparative study, Conditional Random Fields with an f1 score of 0.6968, and Probabilistic NNM with an accuracy of 78.85. However the Probabilistic model takes the 'others' tag into consideration.
Future Work: We plan to use our model to implement a question answering system. This system can be used by people for automated medical advice in Hindi. Thus can be used at local medical shops for correct medical advice, rather than depending on the pharmacist. Individual Contribution : We have implemented two models so Gurjit Singh and K Kavitha worked on Probabilistic Neural Network Model while Pooja Singh and Satyam Ankur worked on implementing CRF Model.

The Dataset preprocessing was done by Kavitha.The report is a collective effort of all team members with most of it being done by Pooja Singh.

References

[1] http://www.albertauyeung.com/post/python-sequence-labelling-with-crf/

[2] https://python-crfsuite.readthedocs.io/en/latest/pycrfsuite.html

[3] http://www.cs.columbia.edu/ mcollins/crf.pdf

[4] https://www.kaggle.com/aijain/hindi-health-datasetConsumable%20Gazetteer.txt