Building a Parts of Speech tagger using Artificial Neural Network

Bharathi Raja A
Indian Institute of Information Technology-Sricity
E-mail:bharathiraja@iiits.in

Saumitra Yadav
International Institute of Information Technology, Hyderabad
saumitra.yadav@research.iiit.ac.in

Sriram Sachin Gudimella
International Institute of Information Technology, Hyderabad
sriramsachin.gudimella@research.iiit.ac.in

***Abstract*—Part of Speech Tagger is tool or system which automatically assigns the POS for a word of a given sentence. There have been many approaches for creating automatic POS Tagger. Rule Based approach was used at first but did not give expected result then researches moved to the Statistical approach like Hidden Markov Model based POS tagger. Statistical approach is the state of art for POS tagger. Recently Artificial Neural Networks proven to give better results in many related fields. In this report we present POS tagger build while considering word2vec representation of words as input rather than tagging probability as input for Neural Networks.**

***Keywords*—POS tagger, Neural Network**

I. Introduction

Part-of-Speech Tagging is well known problem of annotating the word with Part of Speech markers. Most of words in majority of language are ambiguous, so finding the Part of Speech is essential. This is Lowest level of syntactic analysis and major pre-processing step for many applications in Natural Language Processing. In Machine Translation, POS disambiguation is irreplaceable role. Our goal is to build the system which will take the sentence for example The Goose saw a Fox and produce tag sequence D N V D N automatically. With dawn of computational linguistic people have used the rule based[1], statistical[2] or hybrid approach to come up with the POS tagger but that requires either more complex encoded rules or more amount of hand annotated data.

This paper present a way to build a POS tagger using Neural Network. Following section describes about the dataset which was used in the project. Then in the third section reports the implementation details. Experiment setup and Results are presented in fourth section . Finally fifth section concludes our report.

II. Data

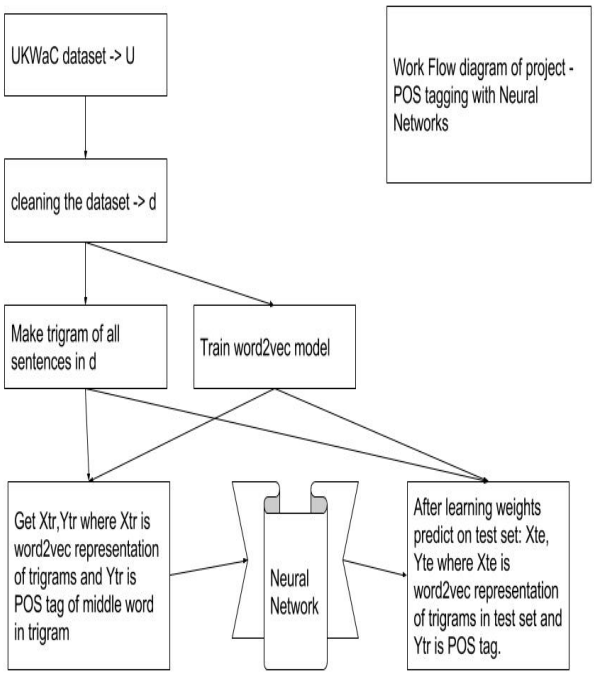
In this project UKWaC[1] dataset is used, which is a British English web corpus. There are two formats of the corpus is available, one is plain text file without morphological annotation and second format is XML format. UKWaC is prepared by Adriano Ferraresi[3] and it uses Penn Treebank Tagset.

The data is preprocessed and UKWaC(xml format) raw file is converted to required format (Two parallel files- sentences and pos_file seperated by linefeed). Then word2vec model (tool based on work of[4]) is build based on this data. One hot vector representation of POS tags.

[1] see https://www.sketchengine.co.uk/xdocumentation/wiki/Corpora/UKWaC

III. Experiment and Result

The method used for building tagger is explained in the figure given below. Word2vec representation of trigram of a sentence is sent as input to neural net which is a 300 points vector. Output layer is a 57 units one hot vector which represents POS tag of middle word in trigram. Function of hidden layer is sigmoid and output layer uses softmax function for classification.



The flow of our work in given in Figures 1. The POS tagset follows the PenTree bank tagset[5]. We got 84% accuracy on our first test set. From our results we found out that the we

got pretty good result in confusion matrix. Most of the tags were properly aligned. Some minor cases like (NN- NP and VV-VVP) were confused.

pos_tags	TP	FP	FN	TN
,	52	0	8	940
JJ	42	13	13	932
NN	104	19	46	831
:	9	1	1	989
CC	31	2	0	967
DT	87	8	7	898
NNS	31	11	6	952
SENT	55	0	2	943
VBP	5	0	0	995
IN	74	2	19	905
WRB	0	8	0	992
VVD	14	6	3	977
TO	33	0	0	967
VB	1	0	0	999
VBZ	19	1	2	978
VVP	2	14	0	984
NP	107	5	41	847
CD	9	10	2	979
PP$	12	0	0	988
WP	0	3	0	997
PP	29	4	5	962
RB	35	13	15	937
VBD	1	0	0	999
VVN	7	0	4	989
RP	3	5	2	990
VVZ	4	17	0	979
MD	3	0	3	994
VH	0	0	0	1000
POS	0	7	0	993
JJR	0	3	0	997
WDT	1	3	0	996
EX	0	4	0	996
”	0	0	0	1000
VHD	0	0	0	1000
PDT	0	0	0	1000
VVG	4	11	6	979
RBR	0	1	0	999
VHZ	0	2	0	998
VV	28	7	11	954
(	0	1	0	999
)	1	0	0	999
VHP	1	2	0	997
“	0	0	0	1000
VHN	0	0	0	1000
VBG	0	1	0	999
$	0	6	0	994
RBS	0	0	0	1000
JJS	0	0	0	1000
WP$	0	1	0	999
VBN	0	0	0	1000
SYM	0	0	0	1000
NPS	0	1	0	999
VHG	0	1	0	999
LS	0	0	0	1000
FW	0	1	0	999
UH	0	2	0	998
#	0	0	0	1000

pos_tags	precision	recall	specifity	accuracy
,	1	0.0553191	1	0.992
JJ	0.763636	0.0444444	0.986243	0.974
NN	0.845528	0.122353	0.977647	0.935
:	0.9	0.00909091	0.99899	0.998
CC	0.939394	0.0319917	0.997936	0.998
DT	0.915789	0.0960265	0.99117	0.985
NNS	0.738095	0.0321911	0.988577	0.983
SENT	1	0.0583245	1	0.998
VBP	1	0.00502513	1	1
IN	0.973684	0.0815877	0.997795	0.979
WRB	0	0	0.992	0.992
VVD	0.7	0.0142421	0.993896	0.991
TO	1	0.0341262	1	1
VB	1	0.001001	1	1
VBZ	0.95	0.0194076	0.998979	0.997
VVP	0.125	0.00200401	0.985972	0.986
NP	0.955357	0.125587	0.994131	0.954
CD	0.473684	0.0091001	0.989889	0.988
PP$	1	0.0121457	1	1
WP	0	0	0.997	0.997
PP	0.878788	0.0300207	0.995859	0.991
RB	0.729167	0.0368421	0.986316	0.972
VBD	1	0.001001	1	1
VVN	1	0.00707786	1	0.996
RP	0.375	0.00301508	0.994975	0.993
VVZ	0.190476	0.00401606	0.982932	0.983
MD	1	0.00301811	1	0.997
VH	nan	0	1	1
POS	0	0	0.993	0.993
JJR	0	0	0.997	0.997
WDT	0.25	0.001001	0.996997	0.997
EX	0	0	0.996	0.996
”	nan	0	1	1
VHD	nan	0	1	1
PDT	nan	0	1	1
VVG	0.266667	0.0040404	0.988889	0.983
RBR	0	0	0.999	0.999
VHZ	0	0	0.998	0.998
VV	0.8	0.0291363	0.992716	0.982
(	0	0	0.999	0.999
)	1	0.001001	1	1
VHP	0.333333	0.001001	0.997998	0.998
“	nan	0	1	1
VHN	nan	0	1	1
VBG	0	0	0.999	0.999
$	0	0	0.994	0.994
RBS	nan	0	1	1
JJS	nan	0	1	1
WP$	0	0	0.999	0.999
VBN	nan	0	1	1
SYM	nan	0	1	1
NPS	0	0	0.999	0.999
VHG	0	0	0.999	0.999
LS	nan	0	1	1
FW	0	0	0.999	0.999
UH	0	0	0.998	0.998
#	nan	0	1	1

IV. Conclusion

We have showed that with simple Neural network we could get good results. From this we conclude that if we optimize Neural Network then it would give great results. In future, we plan to increase the number of hidden layer in the neural network, use higher n-gram, more complex network architecture (like RNN - LSTM) and try the same for India languages.

References

[1] E. Brill, "A simple rule-based part of speech tagger," in *Proceedings of the workshop on Speech and Natural Language*, pp. 112–116, Association for Computational Linguistics, 1992.

[2] T. Brants, "Tnt: a statistical part-of-speech tagger," in *Proceedings of the sixth conference on Applied natural language processing*, pp. 224–231, Association for Computational Linguistics, 2000.

[3] A. Ferraresi, E. Zanchetta, M. Baroni, and S. Bernardini, "Introducing and evaluating ukwac, a very large web-derived corpus of english," in *Proceedings of the 4th Web as Corpus Workshop (WAC-4) Can we beat Google*, pp. 47–54, 2008.

[4] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, pp. 3111–3119, 2013.

[5] A. Taylor, M. Marcus, and B. Santorini, "The penn treebank: an overview," in *Treebanks*, pp. 5–22, Springer, 2003.