Assignment-Discussion
Vector Based POS Tagging

Suvvari Jaswanth : 200050140
T.Krishna Kamal : 200050142
Ashish Patel : 200260011

…

Date : 9th October 2022

Problem Statement: **part 1**

- Given a sequence of words, produce the POS tag sequence
- Technique to be used: HMM-Viterbi-vector (vector based; the whole corpus is corpus of word vectors which replace words)
- Use Universal Tag Set (12 in number);
  'noun', 'verb', '.', 'adp', 'det', 'adj', 'adv', 'pron', 'conj', 'prt', 'num', 'x'
- 5-fold cross validation
- Compare with HMM-Viterbi-symbolic

Problem Statement: part 2

- Given a sequence of words, produce the POS tag sequence
- Technique to be used: word2vec vectors, FFNN and BP (a slide on FFNN-BP architecture is a must)
- Use Universal Tag Set (12 in number);
  'noun', 'verb', '.', 'adp', 'det', 'adj', 'adv', 'pron', 'conj', 'prt', 'num', 'x'
- 5-fold cross validation
- Compare with HMM-Viterbi-symbolic

Overall performance Word2Vec,FFNN,BP

overall_precision= 0.9135520428977748

overall_recall= 0.8866414117822909

overall_F0.5score= 0.8465395938988288

overall_F1score= 0.8227032387045317

overall_F2score= 0.7615568277798146

Overall performance Viterbi Symbolic

overall_precision= 0.802104524776551

overall_recall= 0.9348077739120372

overall_F0.5score= 0.8255309162440694

overall_F1score= 0.8633651864977725

overall_F2score= 0.9048509207605036

Overall performance
Viterbi Vector

overall_precision= 0.9535525678688
overall_recall= 0.9466576798776
overall_F0.5score= 0.9565395938988288
overall_F1score= 0.9427032387045317
overall_F2score= 0.9459268277798146

Per POS performance

Word2Vec,FFNN,BP

Error Analysis

	Precision	Recall	F0.5_Score	F1_Score	F2_Score
noun	0.9672416957239861	0.9347344151813426	0.9605606079935848	0.9507102593010148	0.9410598864407386
verb	0.9718363266572443	0.9475347023829529	0.9668767876863136	0.9595316701269527	0.9522973091655912
.	0.9998777605737104	1.0	0.9999022060681134	0.9999388765510075	0.9999755497237118
adp	0.9359933133908147	0.9075282576242268	0.9301583415666285	0.9215410269181625	0.9130819143092557
det	0.8814683053040103	0.9825604974989861	0.8999876171213934	0.9292731263451744	0.9605286343612336
adj	0.9279279279279279	0.8961716937354989	0.9213979007633589	0.9117733844791975	0.9023478565588134
adv	0.8912166704010756	0.8814404432132964	0.8892441145565516	0.8863015987967244	0.8833784924265978
pron	0.998938710533298	0.9345910388482065	0.9853699390196027	0.9656941327348509	0.9467887139767641
conj	0.9906837201957998	0.995714965878432	0.9916858976385421	0.9931929713471586	0.9947046326537083
prt	0.6797365599632409	0.9243907519266819	0.7177281107481319	0.7834068843777582	0.8623168693894997
num	0.9609214315096668	0.955810147299509	0.9598948060486523	0.9583589743589743	0.9568280494798066
x	1.0	0.08661417322834646	0.3216374269005848	0.15942028985507248	0.10597302504816956

Viterbi symbolic

Error Analysis

	Precision	Recall	F1_Score
VERB	0.9121842105263158	0.9141327566655238	0.9131574441180732
NOUN	0.9452070596377898	0.8934963782226031	0.9186245775877256
PRON	0.9311670160726765	0.9458180527623329	0.9384353541874523
ADJ	0.8312334891282259	0.9199887545684566	0.8733619792361685
ADV	0.8212652755840053	0.9352456688440784	0.8745573654390936
ADP	0.9898683191324554	0.8602509558942323	0.9205192110987854
CONJ	0.9876171739382016	0.994059405940594	0.9908278184140252
DET	0.9931872037914692	0.9902539870053159	0.991718426501035
NUM	0.6758664955070603	0.995274102079395	0.8050458715596331
PRT	0.3210202286719437	0.8820686321894635	0.470724787206603
X	0.13257575757575757	0.7	0.2229299363057325
.	1.0	1.0	1.0

Per POS performance

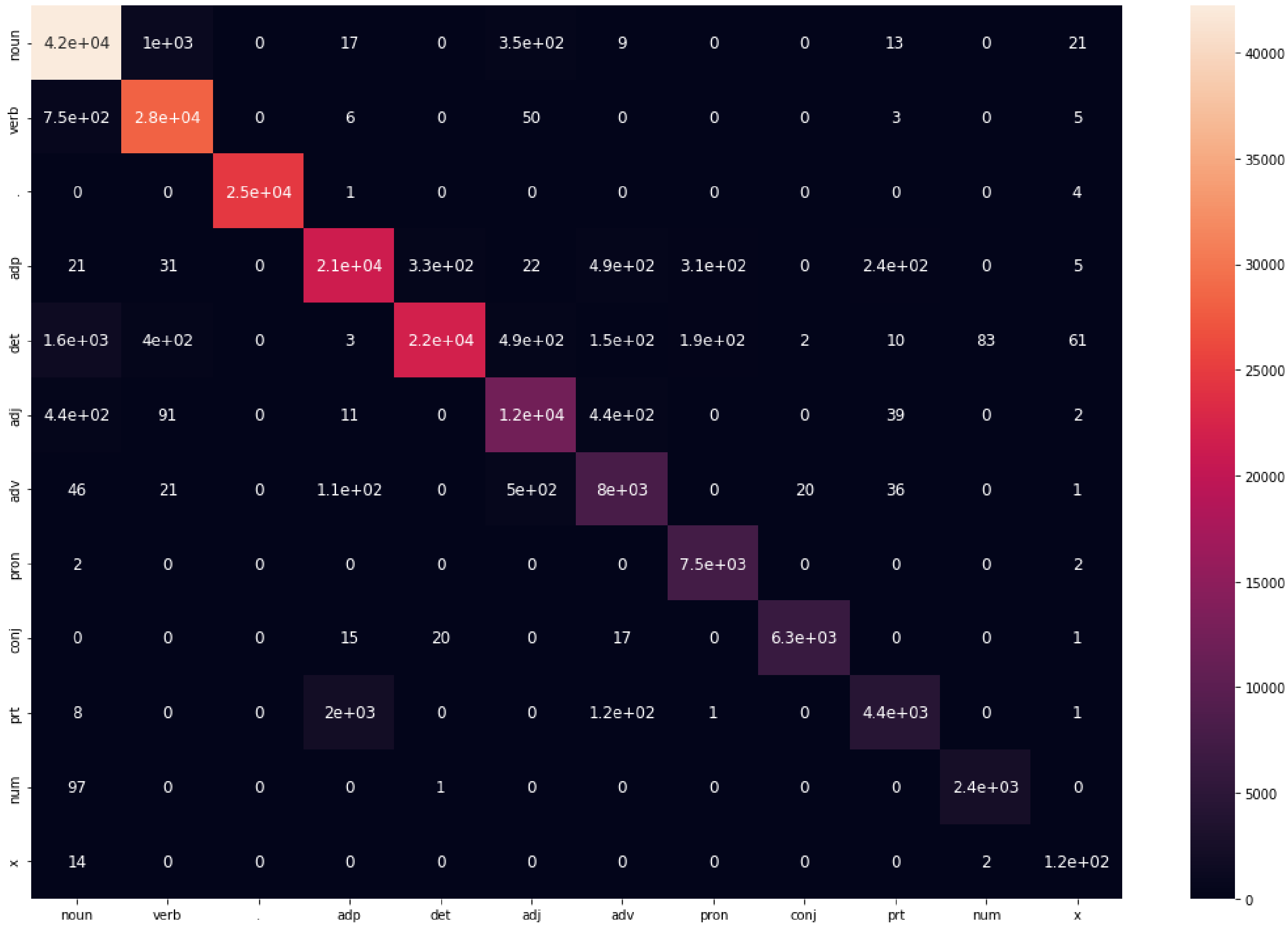
Viterbi vector

Error Analysis

	Precision	Recall	F0.5_Score	F1_Score	F2_Score
noun	0.965557461406518	0.9359537110933759	0.9594878368059343	0.9505251426834622	0.9417283421887338
verb	0.9725436021412537	0.9464591805868316	0.9672123264616377	0.959324112557062	0.9515635242993371
.	0.9998777605737104	1.0	0.9999022060681134	0.9999388765510075	0.9999755497237118
adp	0.9355334856741079	0.9080400938366389	0.9299024189955359	0.9215817839440704	0.913408732065627
det	0.8814683053040103	0.9825604974989861	0.8999876171213934	0.9292731263451744	0.9605286343612336
adj	0.9130623777439687	0.9137906032482599	0.9132079299750739	0.913426345352419	0.913644865233214
adv	0.9260465677403789	0.8505263157894738	0.9098883383513905	0.8866812983712603	0.8646286242087003
pron	0.9994687209456767	0.9339704604691572	0.9856443035653472	0.9656101629667652	0.9463742234071987
conj	0.9906837201957998	0.995714965878432	0.9916858976385421	0.9931929713471586	0.9947046326537083
prt	0.6797365599632409	0.9243907519266819	0.7177281107481319	0.7834068843777582	0.8623168693894997
num	0.9404186795491143	0.9558101472995091	0.9434571890145396	0.948051948051948	0.9526916802610114
x	1.0	0.11417322834645668	0.3918918918918919	0.20494699646643108	0.13875598086124402

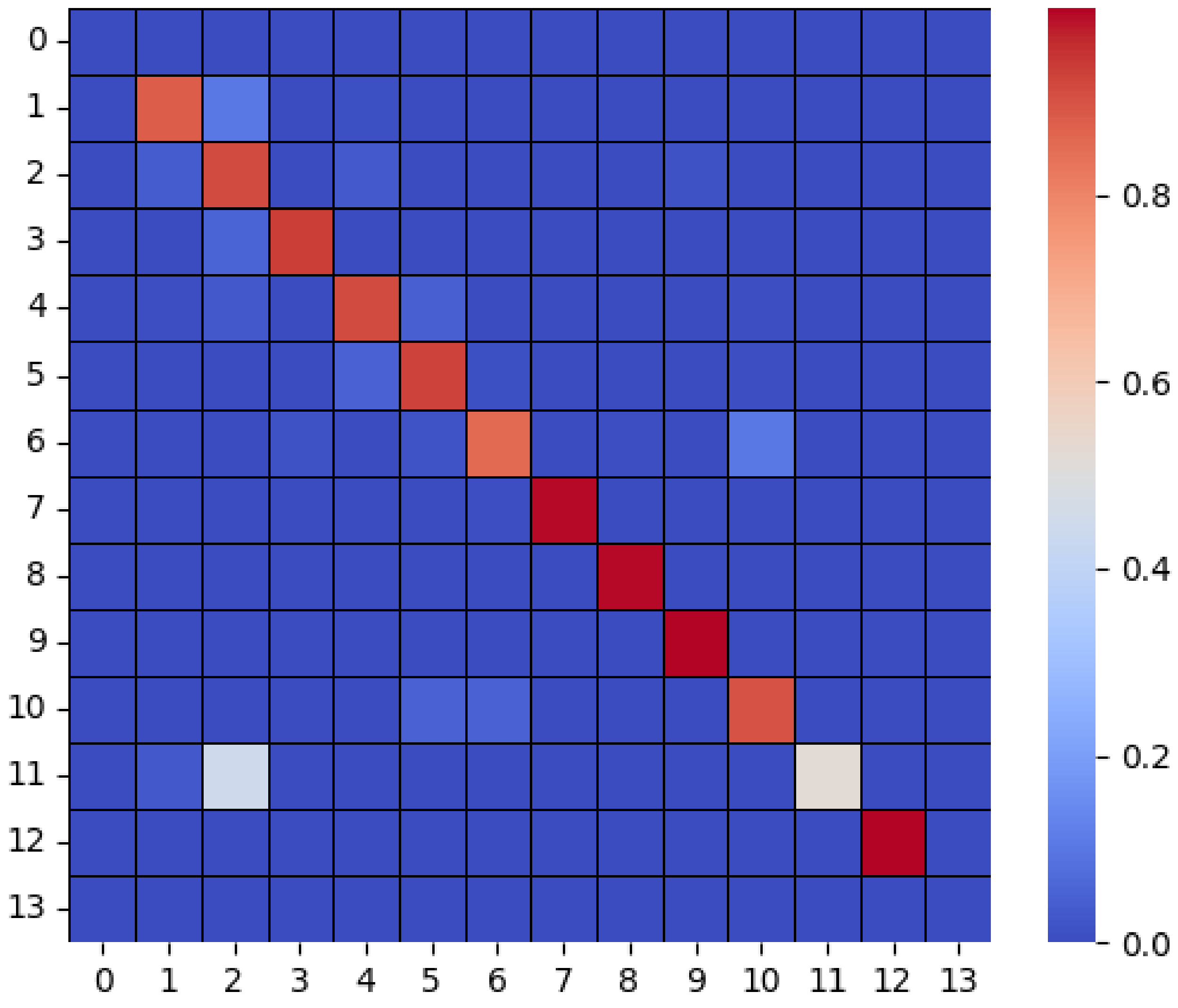
Confusion Matrix (12 X 12) (can give heat map) (compare all 3 models)

Word2Vec,FFNN,BP



Confusion Matrix (12 X 12) (can give heat map) (compare all 3 models)

Viterbi Symbolic



Interpretation of confusion (error analysis)

Actual Tag	Tag it is most confused with	Reason
.	None	. is rarely confused with other tags
ADJ	ADV, NOUN	It is pretty clear why adjectives are confused with adverbs and nouns.
ADP	ADV	Confused with adverb because both frequently precede a verb.
ADV	NOUN	Nouns precede verbs frequently and so do adverbs.
CONJ	ADP	
DET	None	Determiners (articles) are rarely confused with any other tags.
NOUN	ADJ, PRON	Many nouns can also be used as adjectives.
NUM	NOUN	Numerals can also be used as nouns e.g., "three is a small number"
PRON	ADP	Prepositional pronouns
PRT	ADP	Many adpositions cannot be declined further.
VERB	NOUN	Many verbs can also be used as nouns.
X	NOUN	Because of the transition probabilities e.g., "great", "gr8"

Data Processing and Data Sparsity

-> For the first part I used the genisma lib to the assignment-1 part to get the word vectors.

-> I found the word vectors using the word embedding method.In which I used the Neural Networks to train the data , I initially tokenized the dataset by assigning a unique id to the word. Similarly to the tags and during training it will get mapped. When we use a Test Set we will tokenize as above and search for the tag mapped to the id according the nearby tags which are used by the neural layers while tracing the tag. For the unknown words as they occur rarely we will handle them by using the neighbouring word tags.

-> For solving the problem of unseen words use cosine similarity of vectors.