Assignment 2 - Demo
Vector based POS Tagging

Khyati Patel, 200050102
Koustubh Rao, 200100176
Gowri Sriya Mannepalli, 200050043

08-10-2022

Problem Statement: Part 1

- Given a sequence of words, produced the POS tag sequence
- Technique to be used: HMM-Viterbi-vector (vector based; the whole corpus is corpus of word vectors which replace words)
- Used Universal Tag Set (12 in number): tagset = ['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, produced the POS tag sequence
- Technique to be used: word2vec vectors, FFNN and BP
- Used Universal Tag Set (12 in number): tagset = ['NOUN', 'VERB', '.', 'ADP', 'DET', 'ADJ', 'ADV', 'PRON', 'CONJ', 'PRT', 'NUM', 'X']
- 5-fold cross validation
- Compare with HMM-Viterbi-symbolic

Overall performance

	HMM Viterbi symbolic	HMM Viterbi vector	FFNN and BP
Precision	0.9305	0.9385	0.9481
Recall	0.9261	0.9376	0.9382
F1-Score	0.9273	0.9378	0.9422
F0.5-Score	0.9290	0.9381	0.9471
F2-Score	0.9290	0.9381	0.9347

Per POS performance: HMM Viterbi symbolic (P, R, F1, F0.5, F2)

- NOUN: 0.95,0.89,0.92,0.94,0.90
- VERB: 0.97.0.94,0.95,0.96,0.94
- . : 0.97,1.00,0.98,0.97,0.99
- ADP : 0.92,0.93,0.93,0.92,0.93
- DET : 0.95,0.99,0.97,0.96,0.98
- ADJ : 0.84,0.88,0.86,0.85,0.87

- ADV : 0.88,0.85,0.86,0.87,0.86
- PRON: 1.00,0.94,0.97,0.98,0.95
- CONJ : 0.88,0.99,0.93,0.90,0.97
- PRT : 0.65,0.81,0.72,0.68,0.77
- NUM : 0.96,0.90,0.93,0.95,0.91
- X : 0.17,0.43,0.24,0.19,0.32

Per POS performance: HMM Viterbi vector (P, R, F1, F0.5, F2)

NOUN	0.93,0.93,0.93,0.93,0.93
VERB	0.95.0.94,0.95,0.95,0.94
.	0.99,0.99,0.99,0.99,0.99
ADP	0.92,0.93,0.93,0.92,0.93
DET	0.99,0.99,0.99,0.99,0.99
ADJ	0.83,0.87,0.85,0.84,0.86
ADV	0.90,0.85,0.88,0.89,0.86
PRON	0.99,0.94,0.97,0.98,0.95
CONJ	0.99,0.99,0.99,0.99,0.99
PRT	0.71,0.80,0.75,0.73,0.78
NUM	0.93,0.90,0.92,0.93,0.91
X	0.52,0.20,0.29,0.40,0.23

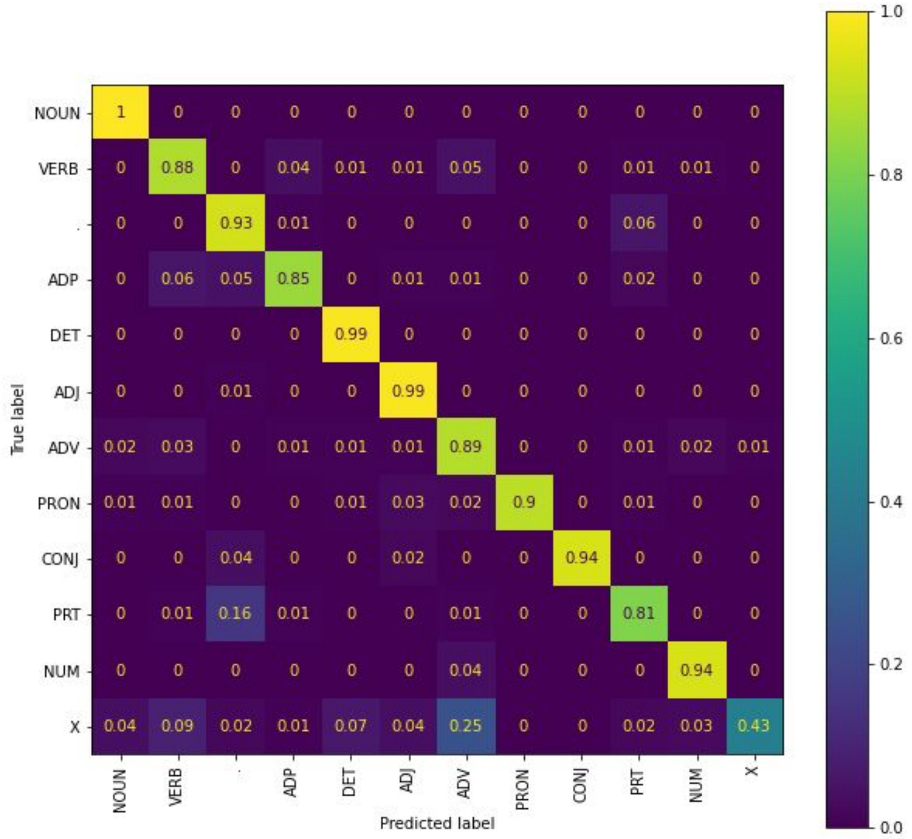
Performance(X) has increased here, as compared to viterbi-symbolic(due to better unknown word handling)

Per POS performance: FFNN and BP (P, R, F1)

NOUN: 0.98,0.93,0.95	ADV : 0.99,0.85,0.91
VERB: 0.92.0.94,0.93	PRON: 0.78,0.94,0.85
. : 0.95,0.94,0.94	CONJ : 0.89,0.99,0.94
ADP : 0.91,0.93,0.92	PRT : 0.82,0.80,0.81
DET : 0.89,0.99,0.94	NUM : 0.94,0.90,0.92
ADJ : 0.95,0.87,0.91	X : 0.69,0.20,0.31

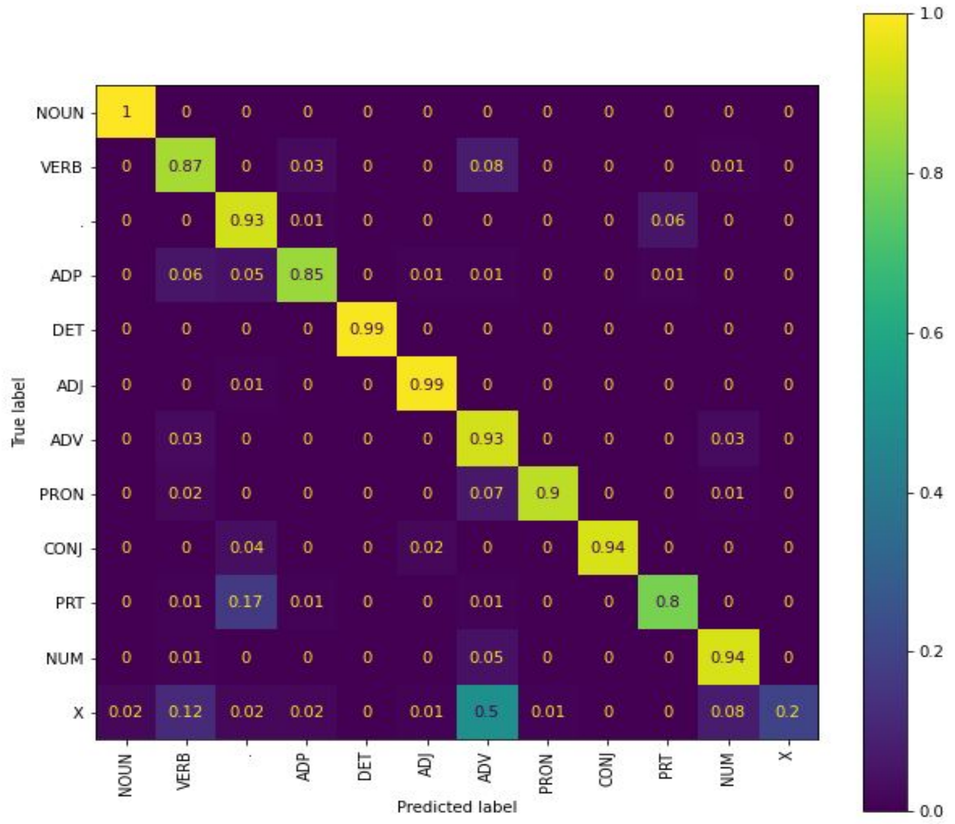
Comparison: Performance(FFNN) > Performance(Viterbi-vec)> Performance(Viterbi-symbolic)

Confusion Matrix: HMM Viterbi symbolic



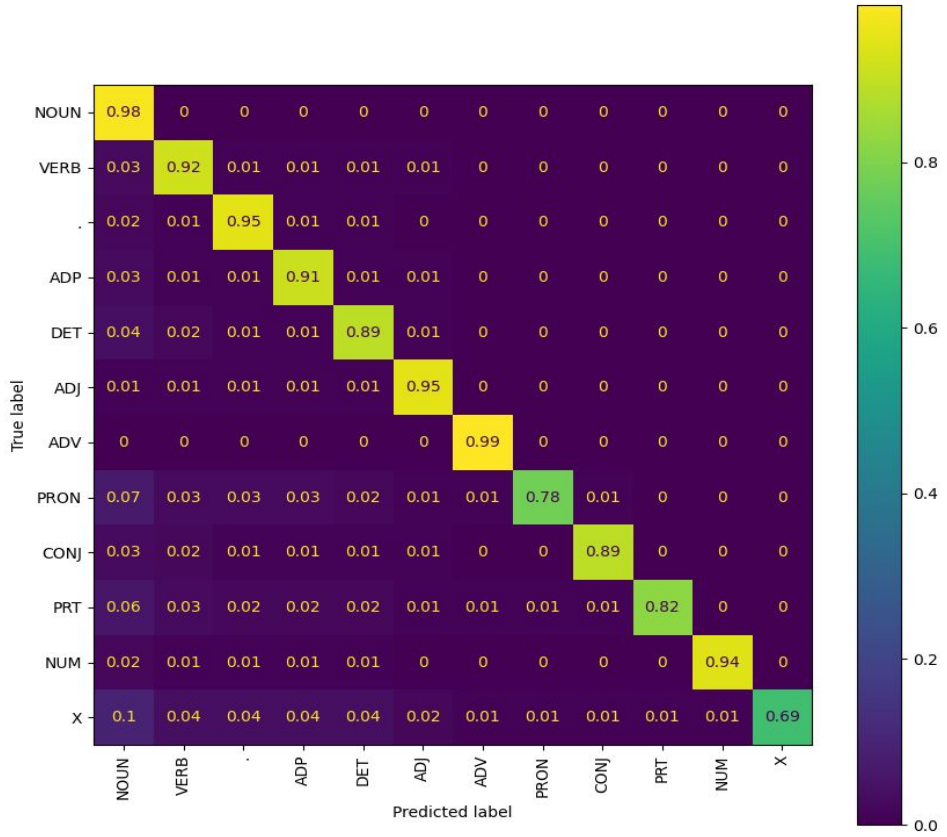
	NOUN	VERB	.	ADP	DET	ADJ	ADV	PRON	CONJ	PRT	NUM	X
NOUN	1	0	0	0	0	0	0	0	0	0	0	0
VERB	0	0.88	0	0.04	0.01	0.01	0.05	0	0	0.01	0.01	0
.	0	0	0.93	0.01	0	0	0	0	0	0.06	0	0
ADP	0	0.06	0.05	0.85	0	0.01	0.01	0	0	0.02	0	0
DET	0	0	0	0	0.99	0	0	0	0	0	0	0
ADJ	0	0	0.01	0	0	0.99	0	0	0	0	0	0
ADV	0.02	0.03	0	0.01	0.01	0.01	0.89	0	0	0.01	0.02	0.01
PRON	0.01	0.01	0	0	0.01	0.03	0.02	0.9	0	0.01	0	0
CONJ	0	0	0.04	0	0	0.02	0	0	0.94	0	0	0
PRT	0	0.01	0.16	0.01	0	0	0.01	0	0	0.81	0	0
NUM	0	0	0	0	0	0	0.04	0	0	0	0.94	0
X	0.04	0.09	0.02	0.01	0.07	0.04	0.25	0	0	0.02	0.03	0.43

Confusion Matrix: HMM Viterbi vector



	NOUN	VERB	.	ADP	DET	ADJ	ADV	PRON	CONJ	PRT	NUM	X
NOUN	1	0	0	0	0	0	0	0	0	0	0	0
VERB	0	0.87	0	0.03	0	0	0.08	0	0	0	0.01	0
.	0	0	0.93	0.01	0	0	0	0	0	0.06	0	0
ADP	0	0.06	0.05	0.85	0	0.01	0.01	0	0	0.01	0	0
DET	0	0	0	0	0.99	0	0	0	0	0	0	0
ADJ	0	0	0.01	0	0	0.99	0	0	0	0	0	0
ADV	0	0.03	0	0	0	0	0.93	0	0	0	0.03	0
PRON	0	0.02	0	0	0	0	0.07	0.9	0	0	0.01	0
CONJ	0	0	0.04	0	0	0.02	0	0	0.94	0	0	0
PRT	0	0.01	0.17	0.01	0	0	0.01	0	0	0.8	0	0
NUM	0	0.01	0	0	0	0	0.05	0	0	0	0.94	0
X	0.02	0.12	0.02	0.02	0	0.01	0.5	0.01	0	0	0.08	0.2

Confusion Matrix: FFNN and BP



True label \ Predicted label	NOUN	VERB	.	ADP	DET	ADJ	ADV	PRON	CONJ	PRT	NUM	X
NOUN	0.98	0	0	0	0	0	0	0	0	0	0	0
VERB	0.03	0.92	0.01	0.01	0.01	0.01	0	0	0	0	0	0
.	0.02	0.01	0.95	0.01	0.01	0	0	0	0	0	0	0
ADP	0.03	0.01	0.01	0.91	0.01	0.01	0	0	0	0	0	0
DET	0.04	0.02	0.01	0.01	0.89	0.01	0	0	0	0	0	0
ADJ	0.01	0.01	0.01	0.01	0.01	0.95	0	0	0	0	0	0
ADV	0	0	0	0	0	0	0.99	0	0	0	0	0
PRON	0.07	0.03	0.03	0.03	0.02	0.01	0.01	0.78	0.01	0	0	0
CONJ	0.03	0.02	0.01	0.01	0.01	0.01	0	0	0.89	0	0	0
PRT	0.06	0.03	0.02	0.02	0.02	0.01	0.01	0.01	0.01	0.82	0	0
NUM	0.02	0.01	0.01	0.01	0.01	0	0	0	0	0	0.94	0
X	0.1	0.04	0.04	0.04	0.04	0.02	0.01	0.01	0.01	0.01	0.01	0.69

Interpretation of confusion matrix (error analysis)

- **HMM-Viterbi-Symbolic**: VERB->ADV, ADV->VERB, PRON->ADJ, NUM->ADV, X->ADV
- **HMM-Viterbi-Vector**: VERB->ADV, ADP->VERB, ADV->VERB, PRON->ADV, X->ADV
- **FFNN-BP**: VERB->NOUN, ADP->VERB, DET->NOUN
- **- ADV/ADP->VERB**
  Example: words like ‘right’ and ‘like’ can be both adverb and verb
  I like sweets.(verb)
  Like as a father pitieth his own children.(adverb)
  That is a fault that will right itself.(verb)
  He stood right in my way.(adverb)
  - PRON->ADJ
  Example: words like ‘some’ can be both pronoun and adjective
  Some men are wise and some are foolish.
  - Reasoning:
  In both these cases as emission and transition probabilities can be considerably equivalent for some words there is a possibility of confusion in POS tagging for these words
- **Comparing 3 models:** The magnitude of confusion has decreased in Viterbi-vec and FFNN as compared to Viterbi-symbolic. Confusion between NOUN and VERB is more in FFNN model, as compared to the Viterbi models.

Data Processing and Data Sparsity

- We used the gensim module to obtain the word vectors for each word in the corpus. Parameters to this function:

  - min_count = 1 (Ignores all words with total absolute frequency lower than this)

  - size/vector_size = 128 (Dimensionality of the feature vectors)

  - window = 5 (Maximum distance between current and predicted word within a sentence)

  - workers = 4 (Ideally, set to “cores-1”. Can use these many worker threads to train the model. So, can train faster with multicore machines)
- For solving the problem of unseen words, we train model_train using words from the training corpus and model_test using words from the test corpus. The word in the training corpus whose vector is most similar to the vector of the unseen data replaces it in the test data. (“similarity” between words is determined using cosine similarity metric)

FFNN and BP Architecture

- The word vectors are of dimension 128. The word vectors were generated using Word2Vec function belonging to gensim library.
- The architecture was developed using PyTorch with a Cross Entropy Loss and an Adam's optimizer with learning rate 0.001. The data was divided to 1000 batches and was trained over 200 epochs.
- The layers in the neural network were as follows:

  - L1: Linear, in:128 out:256, with a ReLu activation (input layer)

  - L2: Linear, in:256 out:512, with a ReLu activation

  - L3: Linear, in:512 out:512, with a ReLu activation

  - L4: Linear, in:512 out:12(output layer)
- Two hidden layers were used. The softmax layer was present within nn.CrossEntropyLoss().
- Back propagation was implemented using the inbuilt PyTorch function 'backward()'