Assignment 2 - Implementation of Recurrent Perceptron

31th March, 2024

…

Arijeet De 23M0742
Tirthesh Jain 23M0758
Vivek R Pawar 23M0769

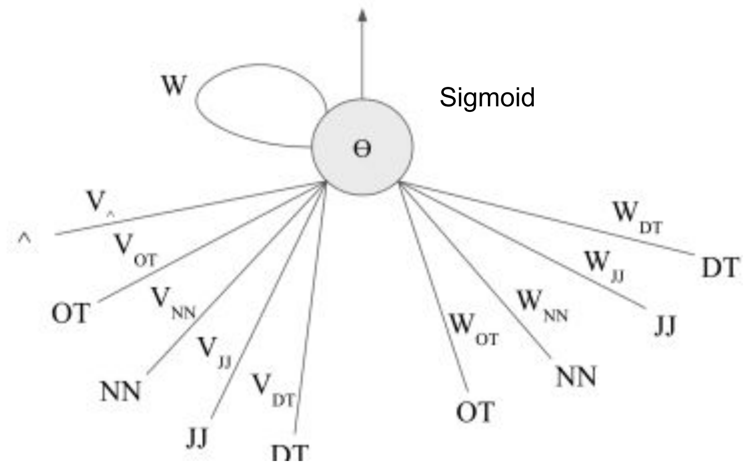
Problem Statement

- **Input**: POS-tagged input tokens

- **Output**: Noun chunk labels on tokens .The beginning of the chunk will be labeled 1 and the rest of the words in the chunk will be labeled 0. All other words are labeled 1.

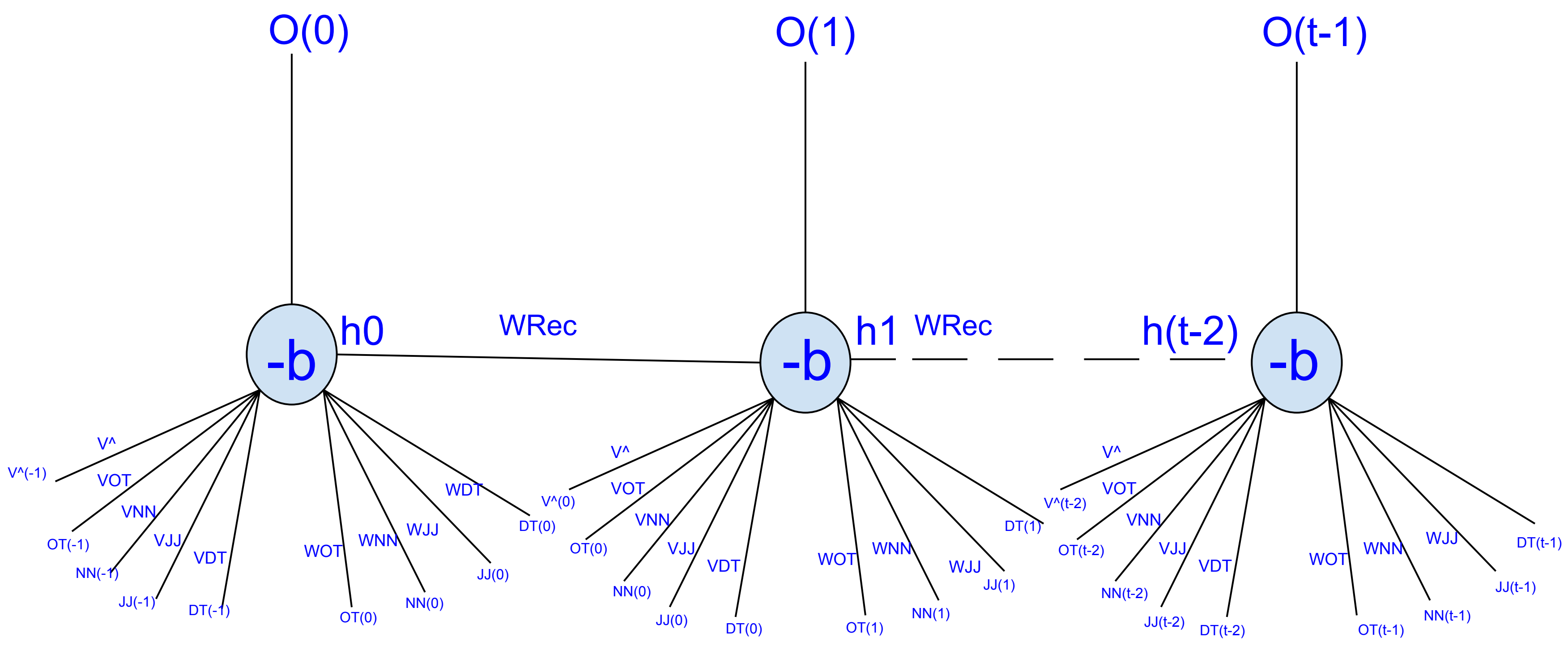
Implementation Details

- Model Architecture:
  - A sigmoid recurrent perceptron
  - **Input size : 9x1.** 5 bit 1-hot encoded representations of previous & 4 bit current POS tags concatenated together
  - Sigmoid output serves as feedback and thresholding at 0.5 yields binary output (0 or 1)



- Best model weights
  - V^: 0.84893
  - VDT: -1.33371
  - VOT: -0.04825
  - WDT: 0.95409
  - WOT: 0.90009
  - b: 0.01739229
  - VNN: -0.62494
  - VJJ: -0.57746
  - WNN: 0.00992
  - WJJ: 0.79556
  - W_rec: 0.4305

Architecture



Overall performance

- Test Precision: 0.86161
- Test Accuracy: 0.83776
- Test Recall: 0.83776
- Test F1 Score: 0.82405
- Test Class-wise Accuracy: [0.75991, 0.89768]
- Test Class-wise Precision: [0.96614 0.80573]
- Test Class-wise Recall: [0.55369 0.98963]
- Test Class-wise F1 Score: [0.70395 0.88826]

- **5-fold cross validation**
  - Mean Accuracy: 0.82531
  - Mean Precision: 0.83001
  - Mean Recall: 0.8253095
  - Mean F1 Score: 0.81731
  - Accuracies: [0.8120929, 0.867276, 0.86728, 0.80846, 0.7715]

Language constraint table

Expression	Boolean Value	Difference
V_cap + WDT > b	1	1.82043
V_cap + WJJ > b	1	1.6619
V_cap + WNN > b	1	0.876262
V_cap + WOT > b	1	1.76643
W_rec + VDT + WJJ < b	1	-0.0902281
W_rec + VDT + WNN < b	1	-0.875866
VJJ + WJJ < b	0	0.235494
VJJ + WNN < b	1	-0.550144
W_rec + VJJ + WJJ < b	0	0.666025
W_rec + VJJ + WNN < b	1	-0.119613
VNN + WOT > b	1	0.292546
W_rec + VNN + WOT > b	1	0.723077
W_rec + VOT + WDT > b	1	1.35376
W_rec + VOT + WJJ > b	1	1.19524
W_rec + VOT + WNN > b	1	0.409597
W_rec + VOT + WOT > b	1	1.29976

Error Analysis

- The model's performance is hindered due to errors or **noise** in both the training and testing data
  - Tokens ['Overall', 'women', "'s", 'World', 'Cup', 'standings', 'leaders', 'after']
  - Pos tags [3, 1, 4, 1, 1, 1, 1, 4]
  - Actual [1 0 1 0 0 0 0 1]
  - Predicted [1, 0, **1, 1, 0, 0, 0,** 1]
- The training data consist of example which tags consecutive nouns as a single noun chunk.

Error Analysis

- Error in dataset
  - Tokens ['Advertising', 'revenues', 'at', 'The', 'Times', 'grew', '20', 'percent', '.']
  - Pos tags [1, 1, 4, 2, 1, 4, 4, 1, 4]
  - Actual [1 0 1 1 0 1 1 0 1]
  - Predicted [1, 0, 1, 1, 0, 1, **1, 1**, 1]

    W_rec + VOT + WNN > b
  - The weight assigned to recurrent connections (W_rec) negative, so the context from the previous word must be very high to overcome the threshold. In this case, the context provided by "grew" might not be strong enough to overcome the threshold.

Error Analysis

- Tokens ['"', 'Its', 'not', 'an', 'accident', '.']
  Pos tags [4, 4, 4, 2, 1, 4]
  Actual [1 1 0 1 0 1]
  Predicted [1, 1, 1, 1, 0, 1]
  Here according to condition OT is Followed by OT pos-tag is classified as noun chunk in test data.

- Our model fails to satisfy the consecutive adjectives language constraint inequalities. This discrepancy may stem from the limited exposure to training examples containing consecutive adjectives in sentences. Consequently, on few test data with multiple consecutive adjectives, the model struggles to accurately tag noun chunks.

  W_rec + VJJ + WJJ < b

Error Analysis

- For the case of invalid input such a **noun followed by adjective** our model also outputs wrong value.

- The model architecture might be too simplistic to capture the complexity of the language constraints.

- There were numerous noisy examples within the dataset, but refining the model through the exclusion of such noisy instances during training could potentially enhance our accuracy.

Learnings

- Employing **BPTT to compute gradients** over sequences by unrolling the network through time
- Validating model conformity to predefined **language constraints inequalities** improves interpretation
- **One-hot encoding** represents categorical data like POS tags, aiding in understanding noun chunk constituents and their relationship
- **Hyperparameter tuning** optimizes learning rate, epochs, and **cross-validation** robustly evaluates limited data

Backpropagation Through Time

$$\frac{\partial L}{\partial w_\mathrm{h}} = \frac{1}{T}\sum_{t=1}^{T}\frac{\partial l(y_t, o_t)}{\partial w_\mathrm{h}}$$

$$\frac{\partial l_t}{\partial W_h} = \sum_{i=1}^{t}\frac{\partial l_t}{\partial O_t}\cdot\frac{\partial O_t}{\partial S_i}\cdot\frac{\partial S_i}{\partial W_h}$$

$$\frac{\delta E_3}{\delta W_s} = \frac{\delta E_3}{\delta Y_3}.\frac{\delta Y_3}{\delta S_3}.\frac{\delta S_3}{\delta W_s} + \frac{\delta E_3}{\delta Y_3}.\frac{\delta Y_3}{\delta S_3}.\frac{\delta S_3}{\delta S_2}.\frac{\delta S_2}{\delta W_s} + \frac{\delta E_3}{\delta Y_3}.\frac{\delta Y_3}{\delta S_3}.\frac{\delta S_3}{\delta S_2}.\frac{\delta S_2}{\delta S_1}.\frac{\delta S_1}{\delta W_s}$$

$$\frac{dL_3}{dw_h} = (t_3 - o_3) \times h_3 + (t_3 - o_3) \times w_h \times O_2 \times (1 - O_2) \times h_2 + (t_3 - o_3) \times w_h \times O_2 \times (1 - O_2) \times w_h \times O_1 \times (1 - O_1) \times h_1$$