Assignment 2 - Implementation of Recurrent Perceptron

Jimut Bahan Pal, 22D1594
Shambhavi Pandey, 23D1145
Saikat Dutta, 23D2031

IIT Bombay

CS-772

31 March, 2024

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

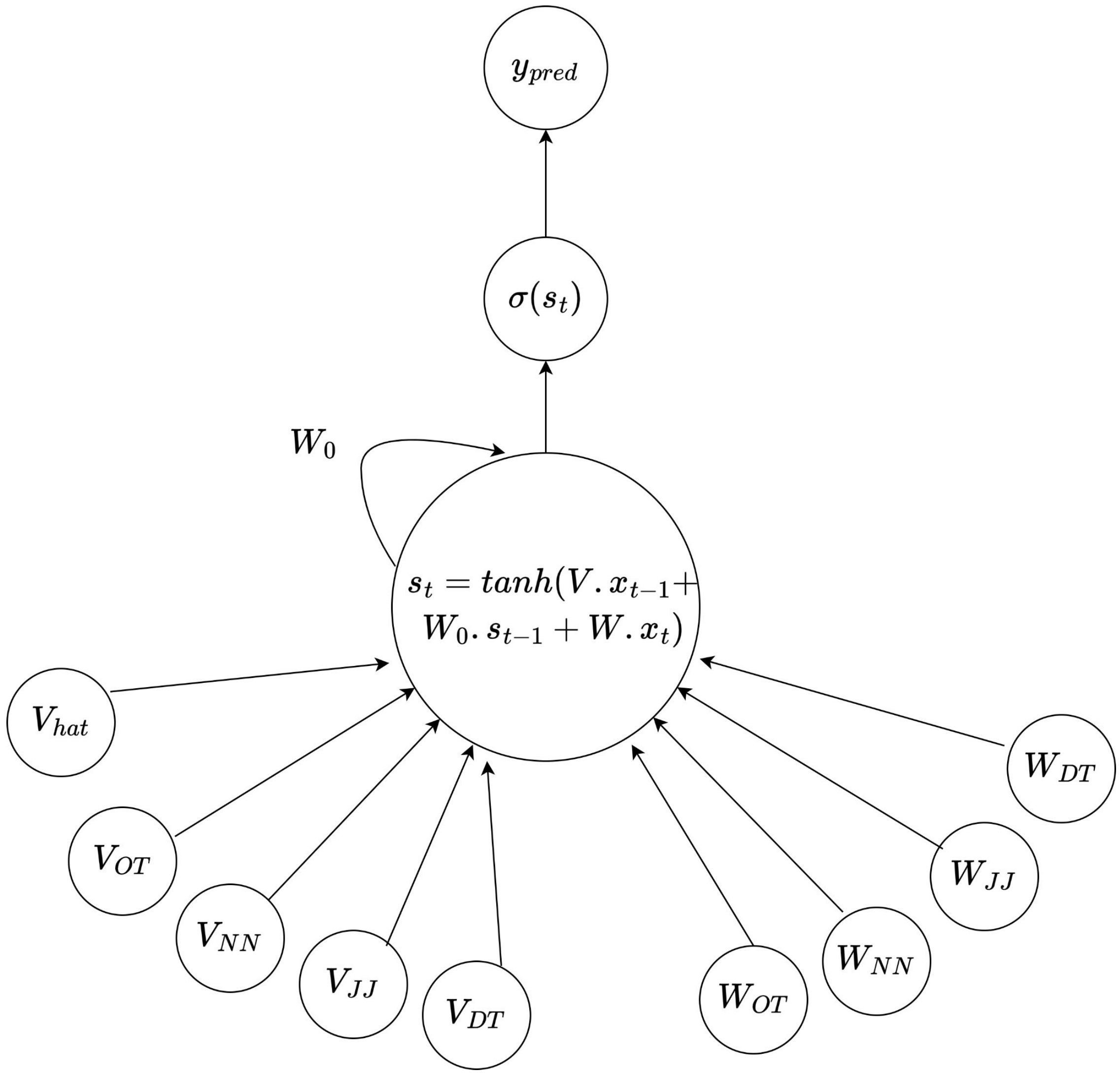
$$s_t = \tanh(Wx_t + W_0 s_{t-1} + Vp_{t-1})$$
$$o_t = \text{sigmoid}(s_t)$$

Here $\mathrm{p}_{t-1} = x_{t-1}$ *and* $x_t = x_t[1:l]$

$$x_t \in \mathbb{R}^5 \iff x \in \mathbb{R}^{l \times 5}$$
$$o_t \in \mathbb{R}^5 \iff x \in \mathbb{R}^{l \times 5}$$
$$s_t \in \mathbb{R}^1 \iff x \in \mathbb{R}^{l \times 1}$$
$$\mathrm{W} \in \mathbb{R}^{4 \times 1}$$
$$\mathrm{V} \in \mathbb{R}^{5 \times 1}$$
$$W_0 \in \mathbb{R}^{1 \times 1}$$

Here l is the length of the sentence.

Implementation Details



Implementation Details

- The BPTT equations with respect to weights W,W_0,V are given as follows at 3rd time step-

$$\frac{\partial E_3}{\partial W} = \frac{\partial E_3}{\partial \hat{y}^3} \frac{\partial \hat{y}^3}{\partial s_3} \frac{\partial s_3}{\partial W}$$

$$\frac{\partial E_3}{\partial W_O} = \frac{\partial E_3}{\partial \hat{y}^3} \frac{\partial \hat{y}^3}{\partial s_3} \frac{\partial s_3}{\partial W_O}$$

$$\frac{\partial E_3}{\partial V} = \frac{\partial E_3}{\partial \hat{y}^3} \frac{\partial \hat{y}^3}{\partial s_3} \frac{\partial s_3}{\partial V}$$

- The last term of each equation which will incorporate propagation towards back upto initial time.

Implementation Details

- We have used cross-entropy loss
- Lr = 0.001
- Experimented with uniform initialization between [-5, 5] and [$-1/\sqrt{4}, 1/\sqrt{4}$].
- Experimented with adaptive learning rate
- Experimented with linear activation function versus non-linear activation function.

Overall performance

- **Quantitative Results:**
  - We have used 4 metrics for evaluation using 5-fold cross validation and full dataset training.
  - Upper row reports metrics on the held out fold and the below row on the full test dataset.
  - We see that initialization plays an important role in performance.

	Dataset	Accuracy	Precision	Recall	F1
LeCun Uniform Initialization	Fold-1	37.1 38.9	8.2 8.0	88.3 81.8	0.15 0.15
	Fold-2	37.8 38.9	8.3 8.0	87.7 81.8	0.15 0.15
	Fold-3	37.6 38.9	8.2 8.0	86.8 81.8	0.15 0.15
	Fold-4	50.7 50.2	45.4 40.9	71.4 70.3	0.55 0.52
	Fold-5	61.1 59.7	88.7 88.3	65.5 63.8	0.75 0.74
	Full-train	50.2	40.9	70.3	0.52
Uniform Initialization [-5, 5]	Fold-1	65.0 62.9	78.4 78.7	72.3 68.8	0.75 0.73
	Fold-2	64.4 62.9	78.4 78.7	71.3 68.8	0.75 0.73
	Fold-3	67.0 **65.2**	100.0 **100.0**	67.0 **65.2**	0.80 **0.79**
	Fold-4	64.7 62.9	78.5 78.7	71.9 69.8	0.75 0.73
	Fold-5	64.3 62.9	78.0 78.7	71.4 69.8	0.74 0.73
	Full-train	**62.9**	**78.7**	**68.8**	**0.73**

Language constraint table

- Learnt weight values:

V_hat	16627.57	W_0	-10772.25
V_NN	16626.59	W_NN	-10796.90
V_DT	16634.67	W_DT	-10798.23
V_JJ	16632.02	W_JJ	2.31
V_OT	16633.09	W_OT	0.98
theta	4885.66		

Language constraint table

- Constraints table (1/2):

Current(W) / Prev (V)	DT	JJ	NN	OT
hat	f(V_hat + W_DT + theta) > 0.5 **YES**	f(V_hat + W_JJ + theta) > 0.5 **YES**	f(V_hat + W_NN + theta) > 0.5 **YES**	f(V_hat + W_OT + theta) > 0.5 **YES**
DT	x	f(W_0 +V_DT + W_JJ + theta) < 0.5 **NO**	f(W_0 + V_DT + W_NN + theta) < 0.5 **YES**	x
JJ	x	f(V_JJ + W_JJ + theta) < 0.5 **NO**	f(V_JJ + W_NN + theta) < 0.5 **NO**	x
	x	f(W_0 + V_JJ + W_JJ + theta) < 0.5 **NO**	f(W_0 + V_JJ + W_NN +theta) < 0.5 **YES**	x

Language constraint table

- Constraints table (2/2):

Current(W) / Prev (V)	DT	JJ	NN	OT
NN	x	x	x	f(V_NN + W_OT + theta) > 0.5 **YES**
	x	x	x	f(W_0 + V_NN + W_OT + theta) > 0.5 **YES**
OT	f(W_0 + V_OT + W_DT + theta) > 0.5 **NO**	f(W_0 + V_OT + W_JJ + theta) > 0.5 **YES**	f(W_0 + V_OT + W_NN + theta) > 0.5 **NO**	f(W_0 + V_OT + W_OT + theta) > 0.5 **YES**

- In total, 10 out of 16 constraints are satisfied.

Error Analysis - Qualitative

- Sentences from test set which received **over 85% accuracy**:
  - **Example: 1**
    - Tokens: ['RKC', 'Waalwijk', '1', 'Willem', 'II', 'Tilburg', '2']
    - POS Tags: [1, 1, 4, 1, 1, 1, 4]
    - Chunk Tags: [1, 0, 0, 0, 0, 0, 0]
    - Predicted Chunk Tags: [0 0 0 0 0 0 0]
    - Accuracy: **85.71%**
  - **Example: 2**
    - Tokens: ['Ironi', 'Rishon', 'Lezion', '1', 'Maccabi', 'Herzliya', '0']
    - POS Tags: [1, 1, 1, 4, 1, 1, 4]
    - Chunk Tags: [1, 0, 0, 0, 0, 0, 0]
    - Predicted Chunk Tags: [0 0 0 0 0 0 0]
    - Accuracy: **85.71%**

Error Analysis - Qualitative

- Sentences from test set which received **between 60%-70% accuracy**:
- **Example: 1**
  - Tokens: ['UAE', '-', 'Hassan', 'Ahmed', '53', ',', 'Adnan', 'Al', 'Talyani', '55', ',', 'Bakhit', 'Saad', '80']
  - POS Tags: [1, 4, 1, 1, 4, 4, 1, 1, 1, 4, 4, 1, 1, 4]
  - Chunk Tags: [1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0]
  - Predicted Chunk Tags: [0 0 0 0 0 1 0 0 0 0 1 0 0 0]
  - Accuracy: **66.67%**
- **Example: 2**
  - Tokens: ['Tasmania', '481', 'for', 'eight', 'declared', '(', 'Michael', 'DiVenuto', '119', ',', 'David', 'Boon', '118', ',', 'Shaun', 'Young', '113', ')', ';', 'Victoria', '220', 'for', 'three', '(', 'Dean', 'Jones', '130', 'not', 'out', ')', '.']
  - POS Tags: [1, 4, 4, 4, 1, 4, 1, 1, 4, 4, 1, 1, 4, 4, 1, 1, 4, 4, 4, 1, 4, 4, 4, 4, 1, 1, 4, 4, 4, 4, 4]
  - Chunk Tags: [1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1]
  - Predicted Chunk Tags: [0 0 1 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 1 0 0 0 1 0 1 0]
  - Accuracy: **66.67%**

Error Analysis - Qualitative

- Sentences from test set which received between **40%-60%** accuracy:
- **Example: 1**
  - Tokens: ['The', 'lanky', 'former', 'Leeds', 'United', 'defender', 'did', 'not', 'make', 'his', 'England', 'debut', 'until', 'the', 'age', 'of', '30', 'but', 'eventually', 'won', '35', 'caps', 'and', 'was', 'a', 'key', 'member', 'of', 'the', '1966', 'World', 'Cup', 'winning', 'team', 'with', 'his', 'younger', 'brother', ',', 'Bobby', '.']
  - POS Tags: [2, 3, 3, 1, 1, 1, 4, 4, 4, 4, 1, 1, 4, 2, 1, 4, 4, 4, 4, 4, 4, 1, 4, 4, 2, 3, 1, 4, 2, 4, 1, 1, 3, 1, 4, 4, 3, 1, 4, 1, 4]
  - Chunk Tags: [1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1]
  - Predicted Chunk Tags: [0 1 0 0 0 0 1 0 1 0 0 0 1 0 0 1 0 1 0 1 0 0 1 0 1 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0]
  - Accuracy: **56.09%**

Error Analysis - Qualitative

- Sentences from test set which received **below 40% accuracy**:
- **Example: 1**
  - Tokens: ['It', 'all', 'culminated', 'in', 'the', 'fact', 'that', 'I', 'now', 'have', 'lots', 'of', 'great', ',', 'great', 'friends', 'in', 'Ireland', '.']
  - POS Tags: [4, 2, 4, 4, 2, 1, 4, 4, 4, 4, 1, 4, 3, 4, 3, 1, 4, 1, 4]
  - Chunk Tags: [1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1]
  - Predicted Chunk Tags: [0 1 0 1 0 1 0 1 0 1 0 0 1 0 1 0 0 0 0]
  - Accuracy: **36.84%**
- **Example: 2**
  - Tokens: ['(', 'tabulate', 'under', 'won', ',', 'lost', ',', 'percentage', ',', 'games', 'behind', ')', ':']
  - POS Tags: [4, 1, 4, 3, 4, 4, 4, 1, 4, 1, 4, 4, 4]
  - Chunk Tags: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
  - Predicted Chunk Tags: [0 0 0 1 0 1 0 0 0 0 0 1 0]
  - Accuracy: **23.07%**

Learnings

- Weight initialization plays an important role in convergence of models.
- Even a very small model like recurrent perceptron does pretty good job in classifying noun chunk.
- The simpler the model, the more explainable it is.
- As per our finding large values of weights could be controlled using some regularisation technique or gradient clipping.

Demo

Thank You