Perceptron POS-Tagger

Yanick Nedderhoff, Moritz Wittmann

15.06.2015

Features Used

Uppercase (yes/no)

Capitalized (yes/no)

Position of token in sentence

Word form of current/previous/next token

Word form length of current/previous/next token

All suffixes of length 2-5

All prefixes of length 2-5

All letter combinations within tokens of length 2-5

Model Representation

- Features:
  - Extracted features are stored into a dictionary *feat_vec* and assigned a dimension value (see example entries in table)
  - 145553 total features/dimensions in dictionary
- Tokens:
  - Each token receives a binary sparse feature vector
  - Vector is implemented as a list of corresponding dimensions (see example in next slide)
- Classifiers:
  - Each POS has a classifier with its own weight vector and binary vector (for turning features on/off)
  - Weight vector: [0.0 for ind in range(len(feat_vec))]
  - Binary vector: [0 for ind in range(len(feat_vec))]

Key	Value
uppercase	0
capitalized	1
suffix_uche	3923
prefix_payme	37108
prev_form_souza	81068
next_form_deduction	114066
current_form_32.2	112940
lettercombs_mod	47084
next_word_len_4	51135
position_in_sentence_34	51265
current_word_len_20	66060
prev_word_len_1	51201

Model Representation

Example sentence: NATO has n't **budged** from its insistence that any gun - carrying plane has offensive capability .

token *budged* sparse feature vector: [60728, 51507, 51466, 51144, 51160, 51135, 51140, 22194, 193, 25744, 1788, 27691, 10712, 45197, 16719]

Feature mapping in *feat_vec*:

Feature	Dimension	Feature	Dimension
current_form_budged	60728	prefix_bu	22194
prev_form_n't	51507	prefix_bud	25744
next_form_from	51466	prefix_budg	27691
current_word_len_6	51144	prefix_budge	45197
prev_word_len_3	51160	suffix_ed	193
next_word_len_4	51135	suffix_ged	1788
position_in_sentence_3	51140	suffix_dged	10712
		suffix_udged	16719

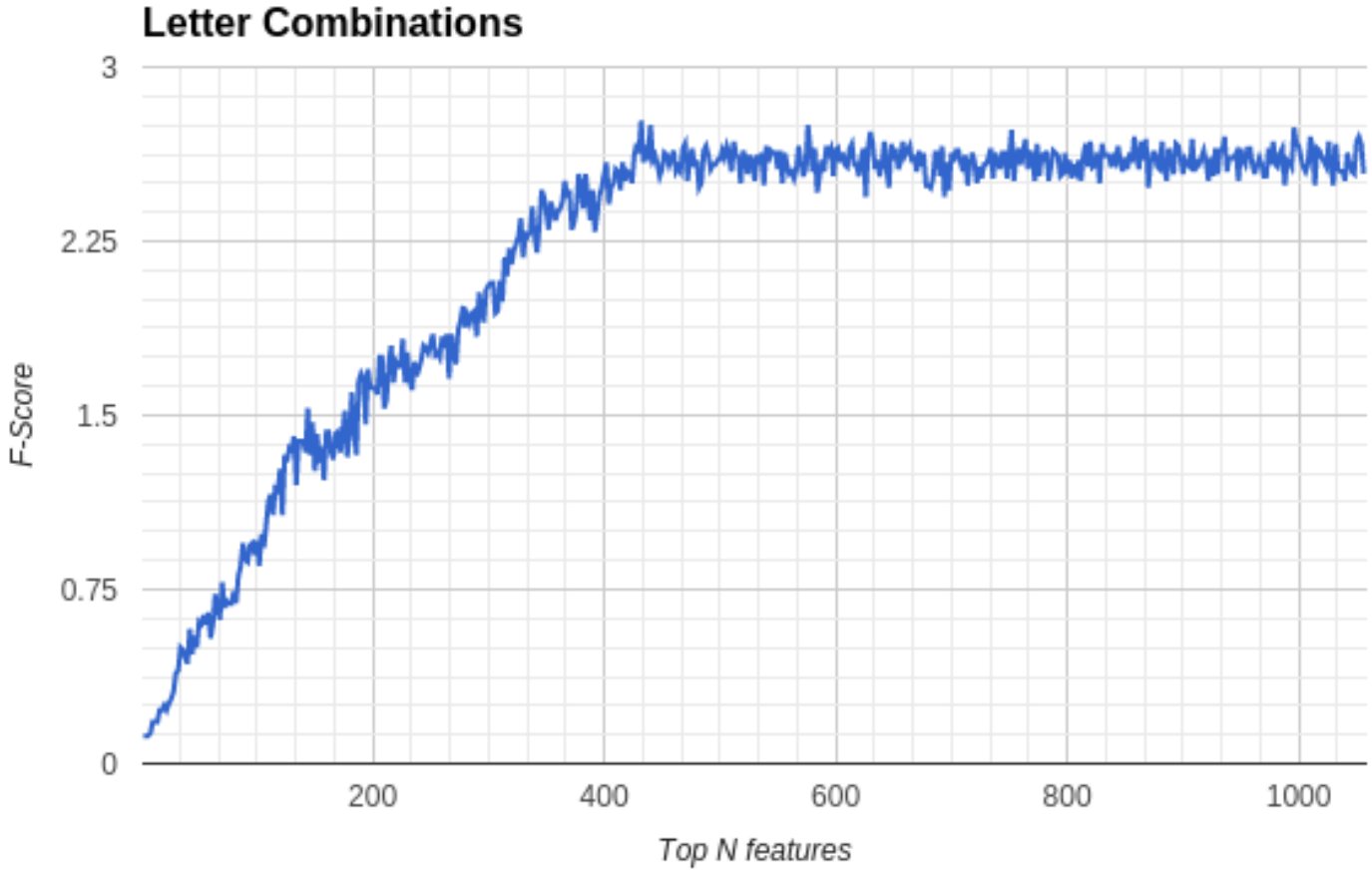
Training

- Training method: Perceptron
- Improvements:
  - Shuffle: the order of the tokens is shuffled after each training iteration (epoch)
  - Alpha decrease: decrease smoothing coefficient after each n training iterations
  - Batch training:
    - before each training iteration: create a copy of the weight vectors
    - perform a training iteration using the copies
    - overwrite the original weight vectors only at the end of each iteration

Feature Selection

- LMI value assigned to every combination of POS-Tag (classifier) and feature
- Features are ranked by LMI for each classifier
- A cut-off value is determined for each feature group:
    1. Word form
    2. Word length
    3. Position in sentence
    4. Suffixes
    5. Prefixes
    6. Letter combinations
- Cut-off value was optimized for each feature group (see graph on next slide)

Feature Selection

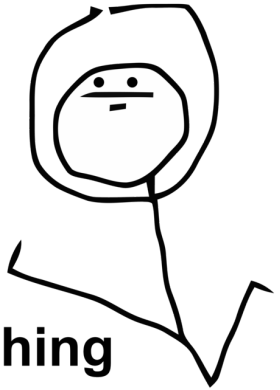


Results

Configuration	Macro-averaged F-Score	Micro-averaged F-Score	Training Time (10 Epochs)
C,U,1	79.72	91.46	1270.92193413sec.
C,U,1,2	85.47	91.17	1315.89282608sec.
C,U,1,2,3	84.46	91.03	1367.65588999sec.
C,U,1,2,3,4	85.42	90.6	1377.6500361sec.
C,U,1,2,3,4,5	86.09	92.1	1384.51872206sec.
C,U,1,2,3,4,5,6	84.94	91.68	1389.44422507sec.
C,U,1,2,3,4,5,6,7	89.42	93.96	1449.96832395sec.
C,U,1,2,3,4,5,6,7,8	91.17	94.25	1463.89461088sec
C,U,1,2,3,4,5,6,7,8,9	90.2	94.34	1542.64573598sec.
C,U,1,2,4,5,7,8	89.77	94,27	958.861662149sec.

1. Word form
2. Word length
3. Position in sentence
4. Suffixes
5. Prefixes
6. Letter combinations
7. Alpha decrease
8. Shuffle
9. Batch training

Feature selection results (averages over 10 runs):
Macro-averaged F-Score: 71.064, Training time (10 Epochs): 1094.2sec



Future Work

- Global LMI ranking for feature selection
- Different methods of feature selection
- Different configurations
- Vary the smoothing coefficient (alpha)
- Try more different LMI cut-offs
- Different/additional features
- ...

GitHub

Thank you!