NLP – 097215 – HW1 – POST – MEMM

Hadar Sugarman 206567067, Tomer Grinberg 318155843

Training:

Features:

In all our models, we opted to retain all features from f100 to f107. We observed that certain additional features appeared to enhance our model's accuracy. These features seem to capture unique characteristics of words and their typical part-of-speech tags. During the process of hyperparameter tuning, we experimented with various thresholds for each feature. Ultimately, we found that a default threshold of 1 for each feature yielded the best results.

$$f_{capital}(h,t) = \begin{cases} 1 \; if \; curr \; word \; has \; an \; uppercase \; letter \; and \; t = t_i \\ 0 \; else \end{cases}$$

$$f_{digit}(h,t) = \begin{cases} 1 \; if \; curr \; word \; has \; a \; digit \; and \; t = t_i \\ 0 \; else \end{cases}$$

$$f_{alpha}(h,t) = \begin{cases} 1 \; if \; all \; chars \; in \; curr \; word \in \{a-z\} \; and \; t = t_i \\ 0 \; else \end{cases}$$

$$f_{alnum}(h,t) = \begin{cases} 1 \; if \; all \; chars \; in \; curr \; word \in \{a-z\} \cup \{1-9\} \; and \; t = t_i \\ 0 \; else \end{cases}$$

$$f_{title}(h,t) = \begin{cases} 1 \; if \; first \; char \; in \; curr \; word \; is \; capital \; and \; t = t_i \\ 0 \; else \end{cases}$$

$$f_{upper}(h,t) = \begin{cases} 1 \; if \; all \; chars \; in \; curr \; word \; are \; capital \; and \; t = t_i \\ 0 \; else \end{cases}$$

$$f_{length}(h,t) = \begin{cases} 1 \; if \; length(curr \; word) = n \; and \; t = t_i \\ 0 \; else \end{cases}$$

In total, we trained two models, one model trained solely on Train1, and tested on test1. The second model used for comp1 and comp2. From what we gathered from the forum, namely, this thread:
https://moodle2324.technion.ac.il/mod/forum/discuss.php?d=7679

We figured it is admissible to train the model with data other than train2. Using this information, under our assumption that more data = better generalization, we searched for more data for our model to be trained on. Thus, instead of training

and evaluating the model with only a small number of examples, we expanded our database, allowing for more robust testing, including k-fold cross validation.

Training the first model took 483.2 seconds, and training the second model took 761.6 seconds. Both took place on the VM's supplied by the course staff, which has 4 vCPUs and 18 Gib RAM.

Inference:

At first, we implemented a naïve version of Viterbi, without Beam-Search and without any optimization. This took about 5 hours to tag the test file. So, we implemented q in a more optimized manner, and set the beam to 2. The choice to set the beam size to 2 came after our hyperparameter tuning script yielded the same results for beam size >2, yet significantly slower run time. Final improvements helped us reach a runtime of ~60 sec to tag the test file.

Testing:

As seen in the image attached, model1, trained on train1, score 95.77% accuracy. There are a few ways we can improve our model given the confusion matrix:

1. Hyperparameter tuning, specifically the thresholds, may help decrease the mistakes here. As higher thresholds may yield better generalization.
2. Perhaps there are certain words which the model has trouble with and may need "brute force" fixing.
3. Perhaps more training data, which better capture the domain of the test would yield less miss-tagged words.

```
Training time: 483.2145082950592 seconds
100%|| 1000/1000 [00:57<00:00, 17.42it/s]
Top Ten Misclassified Tags:
True Tag: NN, Predicted Tag: JJ, Count: 86
True Tag: JJ, Predicted Tag: NN, Count: 71
True Tag: '', Predicted Tag: ', Count: 49
True Tag: IN, Predicted Tag: RB, Count: 44
True Tag: NN, Predicted Tag: NNS, Count: 36
True Tag: VBD, Predicted Tag: VBN, Count: 34
True Tag: VBN, Predicted Tag: JJ, Count: 32
True Tag: VBN, Predicted Tag: VBD, Count: 31
True Tag: NN, Predicted Tag: VBG, Count: 29
True Tag: NNPS, Predicted Tag: NNP, Count: 28

Confusion Matrix:
      ''  IN  JJ  NN  NNP  NNPS  NNS  RB  VBD  VBG  VBN
''     0   0   0   0    0     0    0   0    0    0    0
IN     0   0   1   2    5     0    0  44    0    0    0
JJ     0   0   0  71   18     0    2  14    6    8   28
NN     0   1  86   0   27     0   36   7    0   29    3
NNP    0   1  18  10    0    13    7   0    1    1    0
NNPS   0   0   4   0   28     0   23   0    0    0    0
NNS    0   0   0   2    2     1    0   0    0    0    0
RB     0  15  15   8    0     0    1   0    0    0    0
VBD    0   0   0   3    0     0    0   0    0    0   34
VBG    0   0   0  24    1     0    0   0    0    0    0
VBN    0   0  32   2    0     0    0   0   31    0    0
model1 test score: 0.9577
```

As stated in the training section, which relates to point number 3 seen above, we chose to emphasize data to better understand the performance of the smaller model and achieve better results.

Competition:

For the competition, we used model2, which is trained on a larger dataset to increase generalization. We performed k fold cross validation with our chosen params and reached an average validation score of 95.9% accuracy. We think our true scores on the comp files will be in the range of this score.

Workload:

Both team members worked on the Viterbi implementation, and its optimization. Tomer implemented and searched for features, and Hadar worked on hyperparameter tuning and forming the final submission template.