NERC

Named Entity Recognition and Classification

Outline

- Overall architecture
- Tools
- Feature extraction
- Classifiers and evaluation
- Preliminary results
- QA

Classifier and evaluation

- Naive Bayes classifier
- Maximum Entropy Classifier
    - Logistic Regression → scikit-learn
- Neural Network Classifier
    - Word2Vec(Python) + NN(Torch)

Maximum Entropy Classifier

- Previous results
  - **F1-Score**: 16 % **Precision**: 16.8% **Recall**: 16.8%
- Current Results
  - **F1-Score**: 40% **Precision**: 40.8% **Recall**: 40.5%
- Features used:
  - ['Word', 'PosTag', 'PreviousPosTag', 'Previous2PosTag', 'PreviousWord', 'Previous2Word', 'NextPosTag', 'Next2PosTag', 'NextWord', 'Next2Word', 'PhraseStart', 'PhraseEnd', 'NamedEntity']

Maximum Entropy Classifier - Results by set size

- Train set size 10000/ Test set size 5000
  - **F1-Score**: 40% **Precision**: 40% **Recall**: 40%
- Train set size 11000/ Test set size 4000
  - **F1-Score**: 41.3% **Precision**: 41.3% **Recall**: 41.3%
- Train set size 9000/ Test set size 6000
  - **F1-Score**: 39.7% **Precision**: 39.7% **Recall**: 39.4%
- Train set size 8000/ Test set size 7000
  - **F1-Score**: 39% **Precision**: 39% **Recall**: 39.7%
- Train set size 7000/ Test set size 8000
  - **F1-Score**: 33% **Precision**: 33% **Recall**: 33%

Naive Bayes

Precision Person: 0.64632 | Recall Person: 0.81137 | Accuracy Person: 0.97018 | **F-score Person: 0.71950**

Precision ORG: 0.49823 | Recall ORG: 0.55752 | Accuracy ORG: 0.95602 | **F-score ORG: 0.52621**

Precision LOC: 0.54844 | Recall LOC: 0.64562 | Accuracy LOC: 0.96427 | **F-score LOC: 0.59308**

New results :

Person : 0.87

Org : 0.65

Loc : 0.78

Word2Vec + NN (1) - Reminder

- The classifier was capable to detect only one class, the class with most number of samples

input -> (1) -> (2) -> (3) -> (4) -> (5) -> output]

(1): nn.Linear(80 -> 400)

(2): nn.ReLU

(3): nn.Linear(400 -> 800)

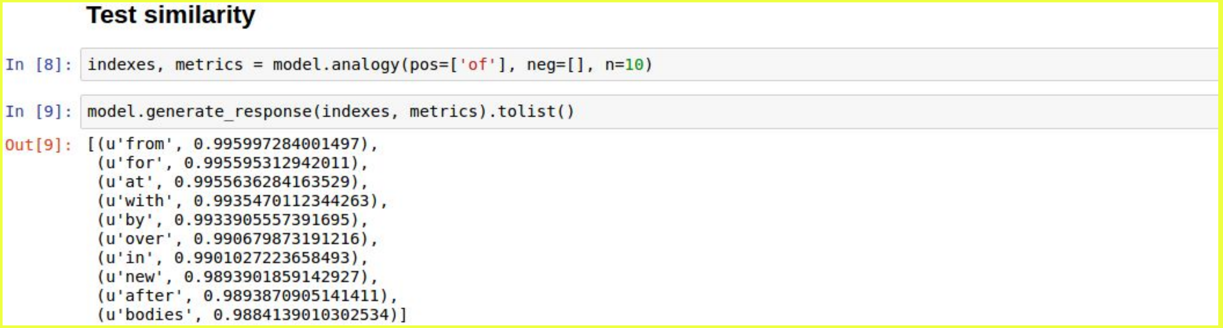
(4): nn.ReLU

(5): nn.Linear(800 -> 8)

Results:

- Accuracy: 63%
- Why 63%? Predicting all values as others
- Why predicting all values as others?
- Dataset is unbalanced

1	1218
2	4
3	13959
4	3772
5	617
6	5
7	2192
8	3



Test similarity

```
In [8]: indexes, metrics = model.analogy(pos=['of'], neg=[], n=10)

In [9]: model.generate_response(indexes, metrics).tolist()

Out[9]: [(u'from', 0.995997284001497),
         (u'for', 0.995595312942011),
         (u'at', 0.9955636284163529),
         (u'with', 0.9935470112344263),
         (u'by', 0.9933905557391695),
         (u'over', 0.990679873191216),
         (u'in', 0.9901027223658493),
         (u'new', 0.9893901859142927),
         (u'after', 0.9893870905141411),
         (u'bodies', 0.9884139010302534)]
```

Word2Vec + NN (2) - Attack Methods

- Join classes (e.g B-class + I-class = class)

  1 : 1221 (LOCATION)

  2 : 2196 (ORGANIZATION)

  3 : 622 (MISC)

  4 : 3772 (PERSON)

  5 : 13959 (OTHERS)

- Downsampling

  1 : 622 (LOCATION)

  2 : 622 (ORGANIZATION)

  3 : 622 (MISC)

  4 : 622 (PERSON)

  5 : 622 (OTHERS)

80% (5*497 samples) - training
20% (5*125 samples) - testing

Word2Vec + NN (3) - Model

- ❏ Small no of neurons in hidden layers

```
[input -> (1) -> (2) -> (3) -> (4) -> (5) -> (6) -> (7) -> output]
  (1): nn.Linear(80 -> 40)
  (2): nn.ReLU
  (3): nn.Linear(40 -> 20)
  (4): nn.ReLU
  (5): nn.Linear(20 -> 10)
  (6): nn.ReLU
  (7): nn.Linear(10 -> 5)
```

```
optimState = {
        learningRate = 1e-1,
        weightDecay = 0,
        momentum = 0.1,
        learningRateDecay = 1e-4
}

batchSize = 10
alg = sgd
```

Word2Vec + NN (4) - Results

- Training: acc = 35.21%

```
[ 11    75    75   229   107  ]        2.213%    [class: 1]
[  2   237    68   110    80  ]       47.686%    [class: 2]
[  6    58   100   172   161  ]       20.121%    [class: 3]
[  8    71    63   287    68  ]       57.746%    [class: 4]
[  4    47    84   122   240  ]       48.290%    [class: 5]
```

- Testing: acc = 29.60%

```
[     0    11    16    27    71  ]      0.000%    [class: 1]
[     1    47    14    29    34  ]     37.600%    [class: 2]
[     0     9    17    27    72  ]     13.600%    [class: 3]
[     0    12    24    42    47  ]     33.600%    [class: 4]
[     0    11    13    22    79  ]     63.200%    [class: 5]
```

References

[1] Bender, Oliver, Franz Josef Och, and Hermann Ney. "Maximum entropy models for named entity recognition." *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*. Association for Computational Linguistics, 2003.
[2] Curran, James R., and Stephen Clark. "Language independent NER using a maximum entropy tagger." *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*. Association for Computational Linguistics, 2003.
[3] Chieu, Hai Leong, and Hwee Tou Ng. "Named entity recognition: a maximum entropy approach using global information." *Proceedings of the 19th international conference on Computational linguistics-Volume 1*. Association for Computational Linguistics, 2002.
[4] Liu, 2011 “*Web Data Mining*” p.109 - p.120 Ch 3.8 : Support Vector Machines
[5] Bishop,2006 “*Pattern recognition and machine learning*” p.325 - p.357, Ch. 7 : Sparse Kernel Machines
[6] Asif Ekbal and Sivaji Bandyopadhyay - “*Named Entity Recognition using Support Vector Machine: A Language Independent Approach*”
[7] Fredrick Edward Kitoogo and Venansius Baryamureeba - “*A Methodology for Feature Selection in Named Entity Recognition*”
[8] Joel Mickelin - “*Named Entity Recognition with Support Vector Machines*”, Master of Science Thesis Stockholm, Sweden 2013
[9] Tomas Mikolov et al. - “Distributed Representations of Words and Phrases and their Compositionality”
[10] Xiang Zhang, Yann LeCun - “Text Understanding from Scratch”

Questions ?

Thank you

for your attention!