Model Evaluation Espa?ol con polaridades

July 18, 2019

[137]:

```
import numpy as np
import pandas as pd

import os
print(os.listdir("."))
```

```
['.ipynb_checkpoints', 'dev_NLI_M.tsv', 'Model Evaluation Español con
polaridades.ipynb', 'Model Evaluation Español con polaridades.pdf',
'test_ep_1.txt', 'test_ep_2.txt', 'test_ep_3.txt', 'test_ep_4.txt',
'test_ep_5.txt', 'test_ep_6.txt', 'test_ep_7.txt']
```

[138]:

```
test_orig = pd.read_csv('dev_NLI_M.tsv', sep='\t')
test_orig.head()
```

[138]:

```
     id                              sentence1                  context  \
0  1262   Tienda de Autoservicio. Siempre bien  Tienda de Autoservicio
1  1262  Tienda de Autoservicio. Siempre bien.  Tienda de Autoservicio
2  1262   Tienda de Autoservicio. Siempre bien  Tienda de Autoservicio
3  1262  Tienda de Autoservicio. Siempre bien.  Tienda de Autoservicio
4  1262   Tienda de Autoservicio. Siempre bien  Tienda de Autoservicio

     aspect   target     label
0   general  general  Positive
1   general  general  Positive
2  servicio  general      None
3  servicio  general      None
4  ambiente  general      None
```

[139]:

```
from glob import glob

test_models = [pd.read_csv(f, sep=' ', header=None) for f in glob('test_ep_*.
 ↪txt')]
for i, t in enumerate(test_models):
    t['label_pred_{0}'.format(i)] = np.select([(t[1] > t[2]) & (t[1] > t[3]),
                                               (t[2] > t[1]) & (t[2] > t[3]),
                                               (t[3] > t[1]) & (t[3] > t[2])],
                                              ['None', 'Positive', 'Negative'])
    del t[0], t[1], t[2], t[3]

    # - P N
test_model = pd.concat(test_models, axis = 1)
test_model.head()
```

[139]:

```
  label_pred_0 label_pred_1 label_pred_2 label_pred_3 label_pred_4  \
0     Positive     Positive     Positive     Positive     Positive   
1     Positive     Positive     Positive     Positive     Positive   
2         None         None         None         None         None   
3         None         None         None         None         None   
4         None         None         None         None         None   

  label_pred_5 label_pred_6  
0     Positive     Positive  
1     Positive     Positive  
2         None         None  
3         None         None  
4         None         None  
```

[140]:

```
test = pd.concat([test_model, test_orig], axis = 1)
test.head()
```

[140]:

```
  label_pred_0 label_pred_1 label_pred_2 label_pred_3 label_pred_4  \
0     Positive     Positive     Positive     Positive     Positive   
1     Positive     Positive     Positive     Positive     Positive   
2         None         None         None         None         None   
3         None         None         None         None         None   
4         None         None         None         None         None   

  label_pred_5 label_pred_6    id                              sentence1  \
0     Positive     Positive  1262   Tienda de Autoservicio. Siempre bien   
1     Positive     Positive  1262  Tienda de Autoservicio. Siempre bien.   
2         None         None  1262   Tienda de Autoservicio. Siempre bien   
3         None         None  1262  Tienda de Autoservicio. Siempre bien.   
4         None         None  1262   Tienda de Autoservicio. Siempre bien   

                  context    aspect   target     label  
0  Tienda de Autoservicio   general  general  Positive  
1  Tienda de Autoservicio   general  general  Positive  
2  Tienda de Autoservicio  servicio  general      None  
3  Tienda de Autoservicio  servicio  general      None  
4  Tienda de Autoservicio  ambiente  general      None  
```

[141]:

```
test['y_real'] = np.select([(test['aspect'] == 'general') & (test['label'] ==␣
 ↪'Positive'),
                          (test['aspect'] == 'general') & (test['label'] ==␣
 ↪'Negative'),
                          (test['aspect'] == 'general') & (test['label'] ==␣
 ↪'None'),
                          (test['aspect'] == 'servicio') & (test['label'] ==␣
 ↪'Positive'),
                          (test['aspect'] == 'servicio') & (test['label'] ==␣
 ↪'Negative'),
                          (test['aspect'] == 'servicio') & (test['label'] ==␣
 ↪'None'),
                          (test['aspect'] == 'ambiente') & (test['label'] ==␣
 ↪'Positive'),
                          (test['aspect'] == 'ambiente') & (test['label'] ==␣
 ↪'Negative'),
                          (test['aspect'] == 'ambiente') & (test['label'] ==␣
 ↪'None'),
                          (test['aspect'] == 'precio') & (test['label'] ==␣
 ↪'Positive'),
                          (test['aspect'] == 'precio') & (test['label'] ==␣
 ↪'Negative'),
                          (test['aspect'] == 'precio') & (test['label'] ==␣
 ↪'None'),
                          (test['aspect'] == 'comida') & (test['label'] ==␣
 ↪'Positive'),
                          (test['aspect'] == 'comida') & (test['label'] ==␣
 ↪'Negative'),
                          (test['aspect'] == 'comida') & (test['label'] ==␣
 ↪'None'),
                          (test['aspect'] == 'ubicación') & (test['label'] ==␣
 ↪'Positive'),
                          (test['aspect'] == 'ubicación') & (test['label'] ==␣
 ↪'Negative'),
                          (test['aspect'] == 'ubicación') & (test['label'] ==␣
 ↪'None'),

                           ],
                            ['GP', 'GN', 'G-',
                             'SP', 'SN', 'S-',
                             'AP', 'AN', 'A-',
                             '$P', '$N', '$-',
                             'CP', 'CN', 'C-',
                             'UP', 'UN', 'U-',
                            ])
```

[142]:

```
for k in test.keys():
    if 'label_pred_' in k:
        test['y_' + k] = np.select([(test['aspect'] == 'general') & (test[k] ==␣
 ↪'Positive'),
                                   (test['aspect'] == 'general') & (test[k] ==␣
 ↪'Negative'),
                                   (test['aspect'] == 'general') & (test[k] ==␣
 ↪'None'),
                                   (test['aspect'] == 'servicio') & (test[k] ==␣
 ↪'Positive'),
                                   (test['aspect'] == 'servicio') & (test[k] ==␣
 ↪'Negative'),
                                   (test['aspect'] == 'servicio') & (test[k] ==␣
 ↪'None'),
                                   (test['aspect'] == 'ambiente') & (test[k] ==␣
 ↪'Positive'),
                                   (test['aspect'] == 'ambiente') & (test[k] ==␣
 ↪'Negative'),
                                   (test['aspect'] == 'ambiente') & (test[k] ==␣
 ↪'None'),
                                   (test['aspect'] == 'precio') & (test[k] ==␣
 ↪'Positive'),
                                   (test['aspect'] == 'precio') & (test[k] ==␣
 ↪'Negative'),
                                   (test['aspect'] == 'precio') & (test[k] ==␣
 ↪'None'),
                                   (test['aspect'] == 'comida') & (test[k] ==␣
 ↪'Positive'),
                                   (test['aspect'] == 'comida') & (test[k] ==␣
 ↪'Negative'),
                                   (test['aspect'] == 'comida') & (test[k] ==␣
 ↪'None'),
                                   (test['aspect'] == 'ubicación') & (test[k] ==␣
 ↪'Positive'),
                                   (test['aspect'] == 'ubicación') & (test[k] ==␣
 ↪'Negative'),
                                   (test['aspect'] == 'ubicación') & (test[k] ==␣
 ↪'None'),
                                  ],
                                         ['GP', 'GN', 'G-',
                                          'SP', 'SN', 'S-',
                                          'AP', 'AN', 'A-',
                                          '$P', '$N', '$-',
                                          'CP', 'CN', 'C-',
                                          'UP', 'UN', 'U-',
                                  ])
```

[143]:

```
from sklearn.metrics import confusion_matrix
from sklearn.utils.multiclass import unique_labels
import matplotlib.pyplot as plt
```

```python
from matplotlib.pyplot import figure
import math
from matplotlib.pyplot import figure
import seaborn as sns
sns.set(style='darkgrid')

def plot_confusion_matrix(y_true, y_pred, classes, title="", cmap=plt.cm.Blues, clean=False, figsize=(20, 16), dpi=300, showLabels=True):

    cm = confusion_matrix(y_true, y_pred)
    cm_norm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis] * 100

    if clean:
        # indexes of No's '0'  and  None's '-'
        indexes = [i for i, c in enumerate(classes) if c.endswith('0') or  '-' in c]

        cm = np.delete(cm, indexes, axis=0)
        cm = np.delete(cm, indexes, axis=1)

        cm_norm = np.delete(cm_norm, indexes, axis=0)
        cm_norm = np.delete(cm_norm, indexes, axis=1)

        classes = np.delete(classes, indexes, axis=0)

    fig, ax = plt.subplots(figsize=figsize, dpi=dpi)

    im = ax.imshow(cm_norm, interpolation='nearest', cmap=cmap)
    ax.figure.colorbar(im, ax=ax)
    ax.grid(False)

    ax.set(xticks=np.arange(cm.shape[1]),
           yticks=np.arange(cm.shape[0]),
           xticklabels=classes,
           yticklabels=classes,
           ylabel='True label',
           xlabel='Predicted label',
           title="Precisión promedio = {0:.2f} %".format(np.mean(cm.diagonal())) if clean else title)

    plt.setp(ax.get_xticklabels(), rotation=45, ha="right", rotation_mode="anchor")

    fmt = 'd'
    fmt_norm = '.2f'

    thresh = 50
    if showLabels:
        for i in range(cm.shape[0]):
            for j in range(cm.shape[1]):
                if cm[i, j] == 0:
                    continue
                ax.text(j, i, '\n' + format(cm[i, j], fmt), fontsize=8,
                        ha="center",  va="top",
                        color="white" if cm_norm[i, j] > thresh else "black")

                if not math.isnan(cm_norm[i, j]):
                    ax.text(j, i, format(cm_norm[i, j], fmt_norm) + '%',␣
↪fontsize=8,
                            ha="center",  va="bottom",
                            color="white" if cm_norm[i, j] > thresh else␣
↪"black")


    fig.tight_layout()
    return ax
```

[144]:

```
y_real = test['y_real'].values
y_preds = {}

for k in test.keys():
    if 'y_label_pred_' in k:
        y_preds[k] = test[k].values
```

[145]:

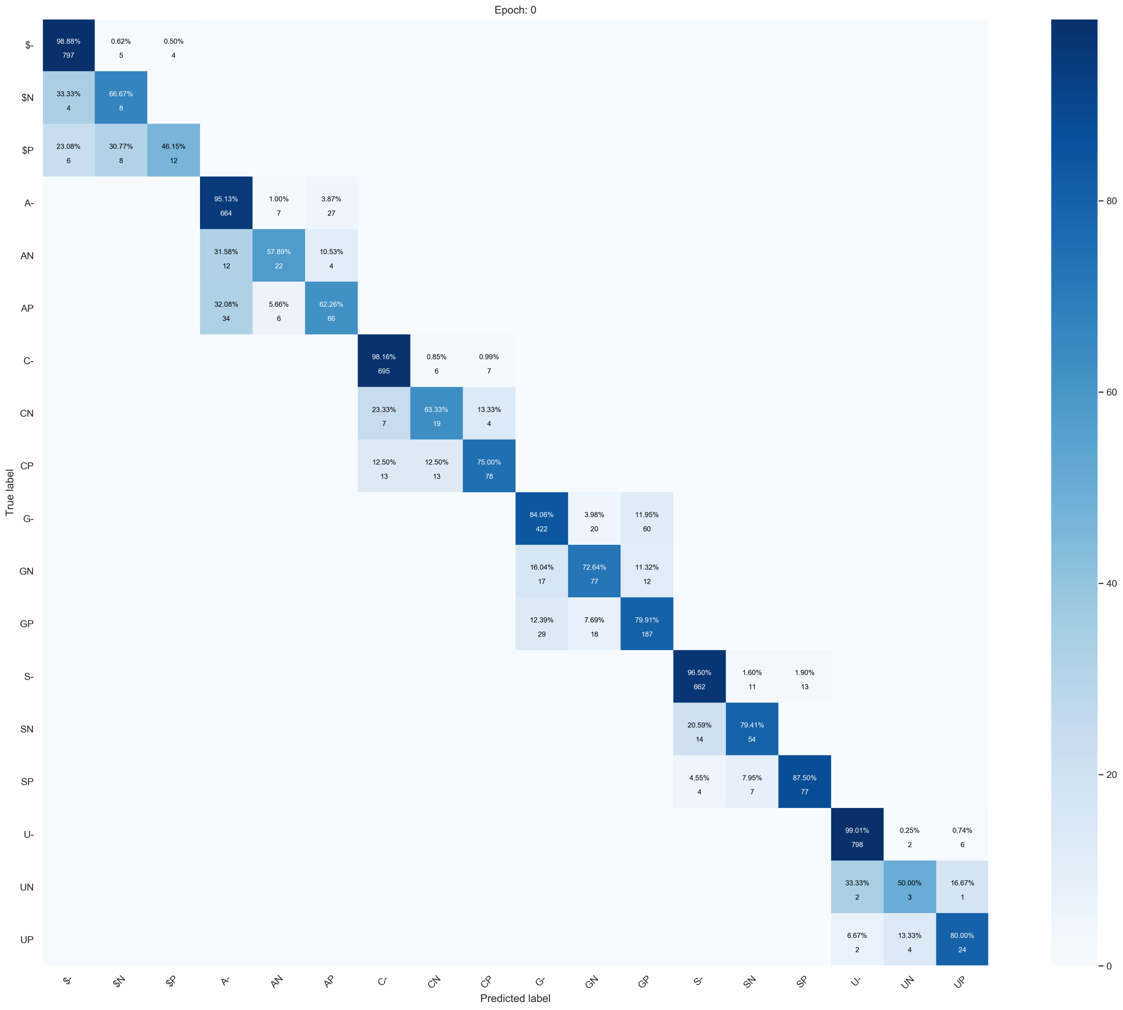
```
i = test.index[test["y_label_pred_6"].apply(lambda x: x == '0')]

test.loc[i, ['aspect', 'label_pred_6', 'y_label_pred_6']]
```

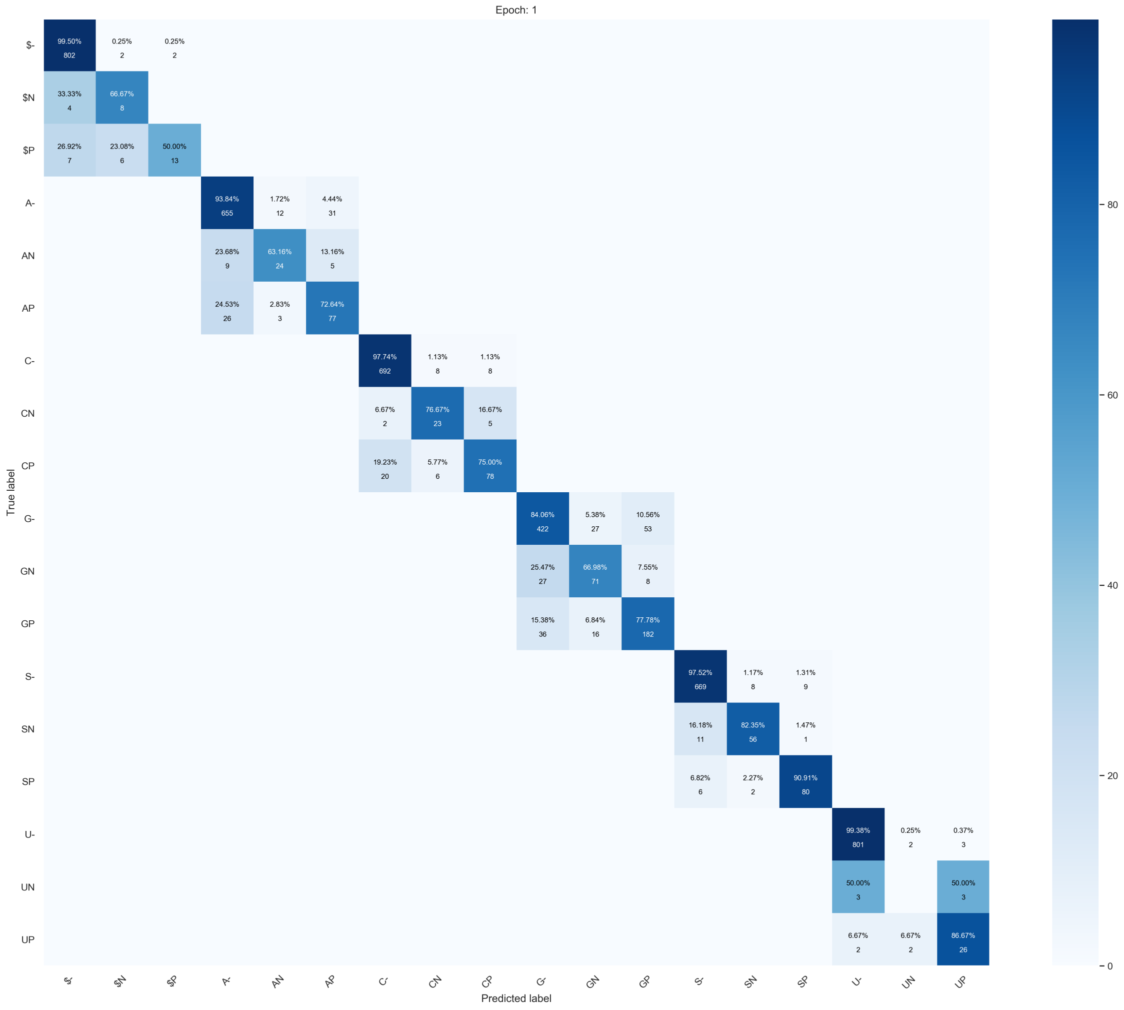
[145]:

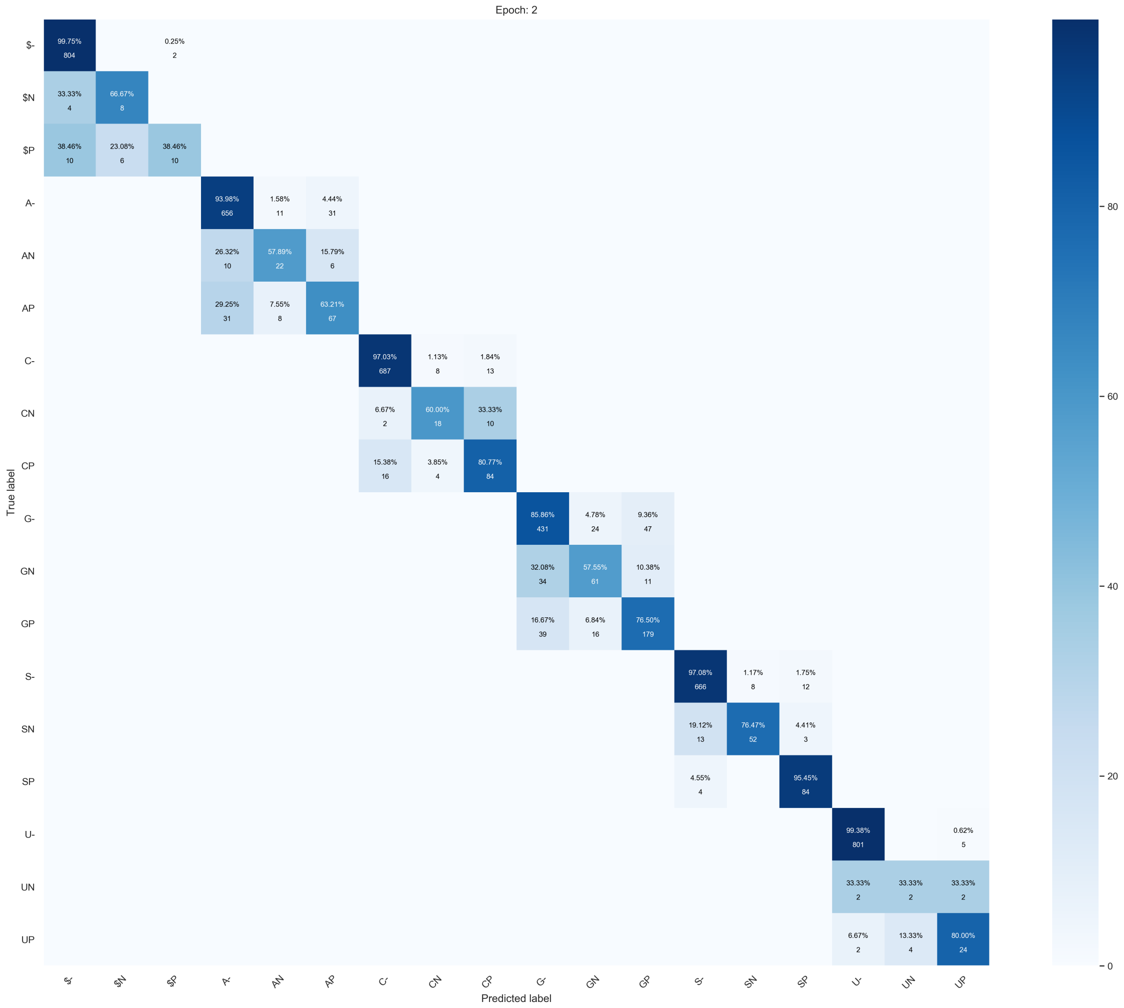
```
Empty DataFrame
Columns: [aspect, label_pred_6, y_label_pred_6]
Index: []
```

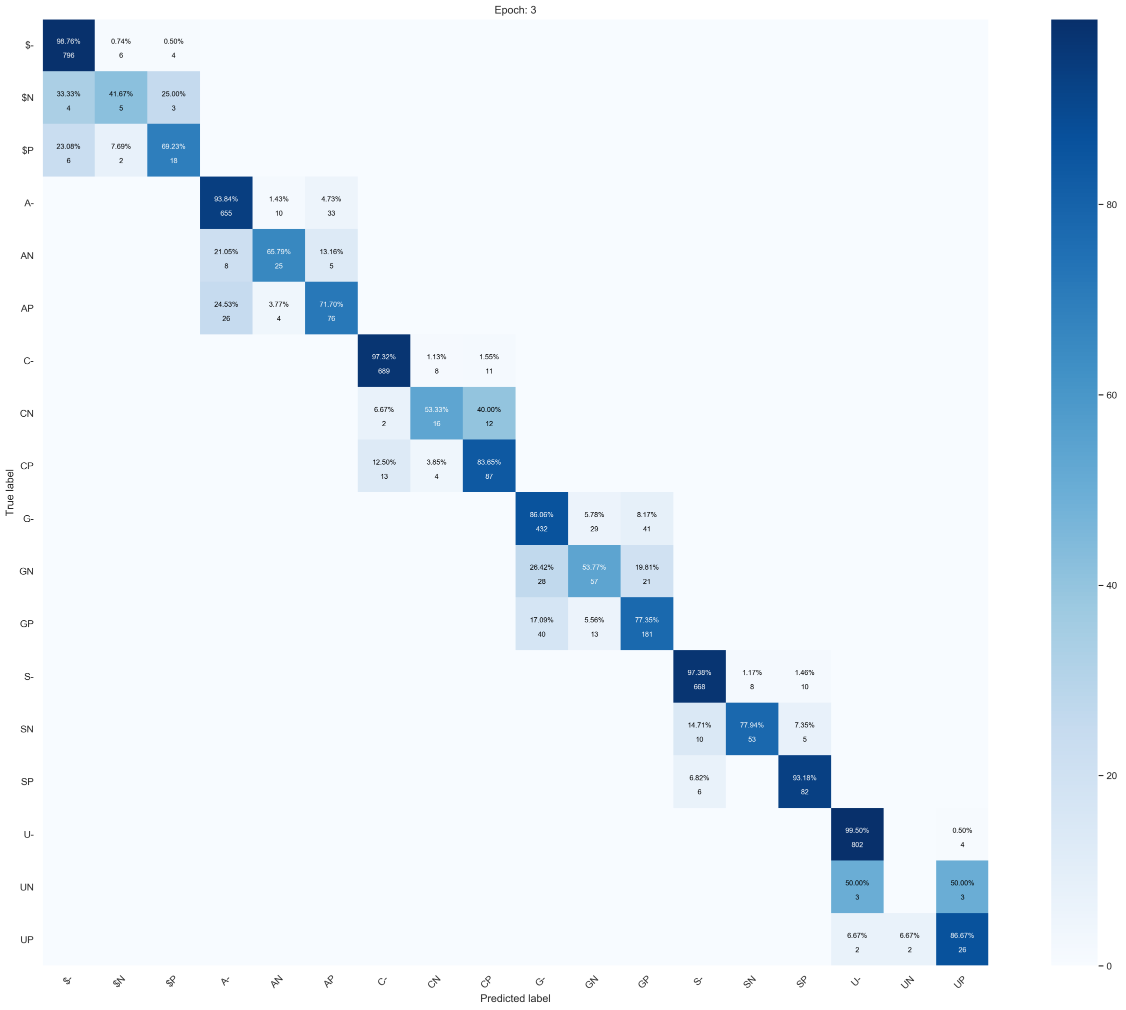
[149]:

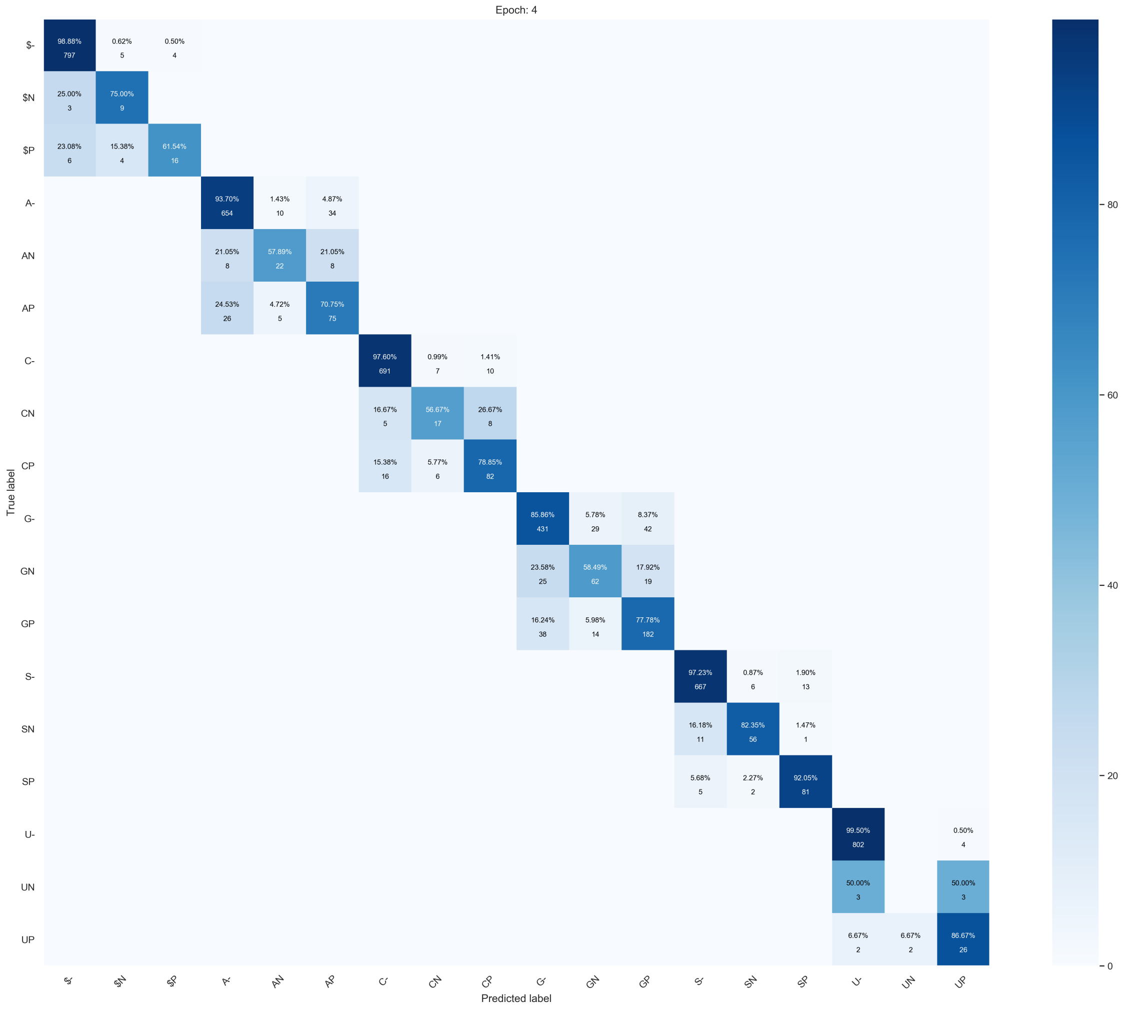
```
for k in test.keys():
    if 'y_label_pred_' in k:
        y_pred = y_preds[k]
        k = k.replace('y_label_pred_', '')
        plot_confusion_matrix(y_real, y_pred, classes=unique_labels(y_real),␣
↪title="Epoch: {0}".format(k))
```

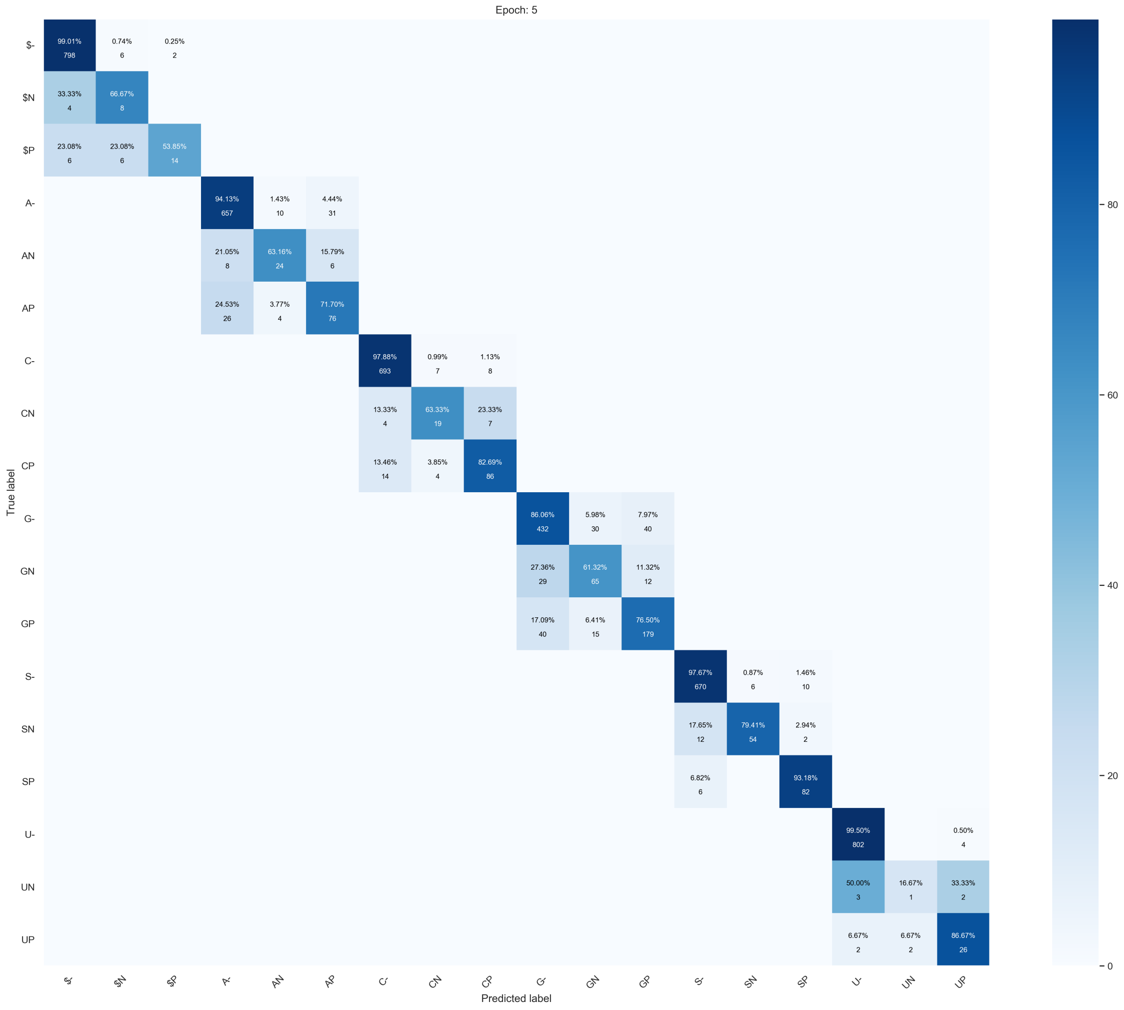

Epoch: 0
$-
98.88% 797
0.62% 5
0.50% 4
$N
33.33% 4
66.67% 8
$P
23.08% 6
30.77% 8
46.15% 12
A-
95.13% 664
1.00% 7
3.87% 27
AN
31.58% 12
57.89% 22
10.53% 4
AP
32.08% 34
5.66% 6
62.26% 66
C-
98.16% 695
0.85% 6
0.99% 7
CN
23.33% 7
63.33% 19
13.33% 4
CP
12.50% 13
12.50% 13
75.00% 78
G-
84.06% 422
3.98% 20
11.95% 60
GN
16.04% 17
72.64% 77
11.32% 12
GP
12.39% 29
7.69% 18
79.91% 187
S-
96.50% 662
1.60% 11
1.90% 13
SN
20.59% 14
79.41% 54
SP
4.55% 4
7.95% 7
87.50% 77
U-
99.01% 798
0.25% 2
0.74% 6
UN
33.33% 2
50.00% 3
16.67% 1
UP
6.67% 2
13.33% 4
80.00% 24
True label
$-
$N
$P
A-
AN
AP
C-
CN
CP
G-
GN
GP
S-
SN
SP
U-
UN
UP
Predicted label
80
60
40
20
0

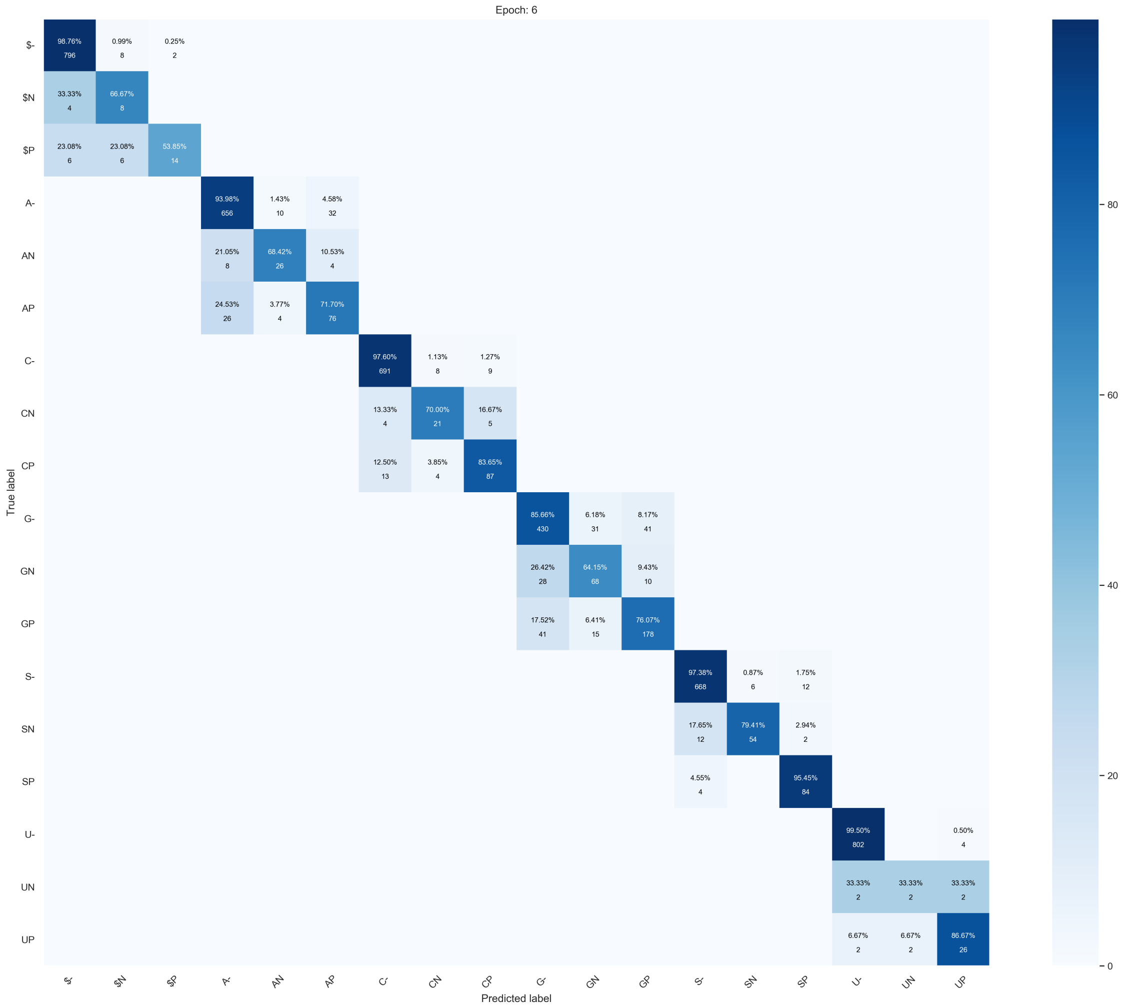

Epoch: 1
$-
99.50%
802
0.25%
2
0.25%
2
$N
33.33%
4
66.67%
8
$P
26.92%
7
23.08%
6
50.00%
13
A-
93.84%
655
1.72%
12
4.44%
31
AN
23.68%
9
63.16%
24
13.16%
5
AP
24.53%
26
2.83%
3
72.64%
77
C-
97.74%
692
1.13%
8
1.13%
8
CN
6.67%
2
76.67%
23
16.67%
5
CP
19.23%
20
5.77%
6
75.00%
78
G-
84.06%
422
5.38%
27
10.56%
53
GN
25.47%
27
66.98%
71
7.55%
8
GP
15.38%
36
6.84%
16
77.78%
182
S-
97.52%
669
1.17%
8
1.31%
9
SN
16.18%
11
82.35%
56
1.47%
1
SP
6.82%
6
2.27%
2
90.91%
80
U-
99.38%
801
0.25%
2
0.37%
3
UN
50.00%
3
50.00%
3
UP
6.67%
2
6.67%
2
86.67%
26
True label
$-
$N
$P
A-
AN
AP
C-
CN
CP
G-
GN
GP
S-
SN
SP
U-
UN
UP
Predicted label
80
60
40
20
0


Epoch: 2
True label
$-
$N
$P
A-
AN
AP
C-
CN
CP
G-
GN
GP
S-
SN
SP
U-
UN
UP
99.75%
804
0.25%
2
33.33%
4
66.67%
8
38.46%
10
23.08%
6
38.46%
10
93.98%
656
1.58%
11
4.44%
31
26.32%
10
57.89%
22
15.79%
6
29.25%
31
7.55%
8
63.21%
67
97.03%
687
1.13%
8
1.84%
13
6.67%
2
60.00%
18
33.33%
10
15.38%
16
3.85%
4
80.77%
84
85.86%
431
4.78%
24
9.36%
47
32.08%
34
57.55%
61
10.38%
11
16.67%
39
6.84%
16
76.50%
179
97.08%
666
1.17%
8
1.75%
12
19.12%
13
76.47%
52
4.41%
3
4.55%
4
95.45%
84
99.38%
801
0.62%
5
33.33%
2
33.33%
2
33.33%
2
6.67%
2
13.33%
4
80.00%
24
$-
$N
$P
A-
AN
AP
C-
CN
CP
G-
GN
GP
S-
SN
SP
U-
UN
UP
Predicted label
80
60
40
20
0


Epoch: 3
$-
98.76% 796
0.74% 6
0.50% 4
$N
33.33% 4
41.67% 5
25.00% 3
$P
23.08% 6
7.69% 2
69.23% 18
A-
93.84% 655
1.43% 10
4.73% 33
AN
21.05% 8
65.79% 25
13.16% 5
AP
24.53% 26
3.77% 4
71.70% 76
C-
97.32% 689
1.13% 8
1.55% 11
CN
6.67% 2
53.33% 16
40.00% 12
CP
12.50% 13
3.85% 4
83.65% 87
G-
86.06% 432
5.78% 29
8.17% 41
GN
26.42% 28
53.77% 57
19.81% 21
GP
17.09% 40
5.56% 13
77.35% 181
S-
97.38% 668
1.17% 8
1.46% 10
SN
14.71% 10
77.94% 53
7.35% 5
SP
6.82% 6
93.18% 82
U-
99.50% 802
0.50% 4
UN
50.00% 3
50.00% 3
UP
6.67% 2
6.67% 2
86.67% 26
True label
$-
$N
$P
A-
AN
AP
C-
CN
CP
G-
GN
GP
S-
SN
SP
U-
UN
UP
Predicted label
80
60
40
20
0

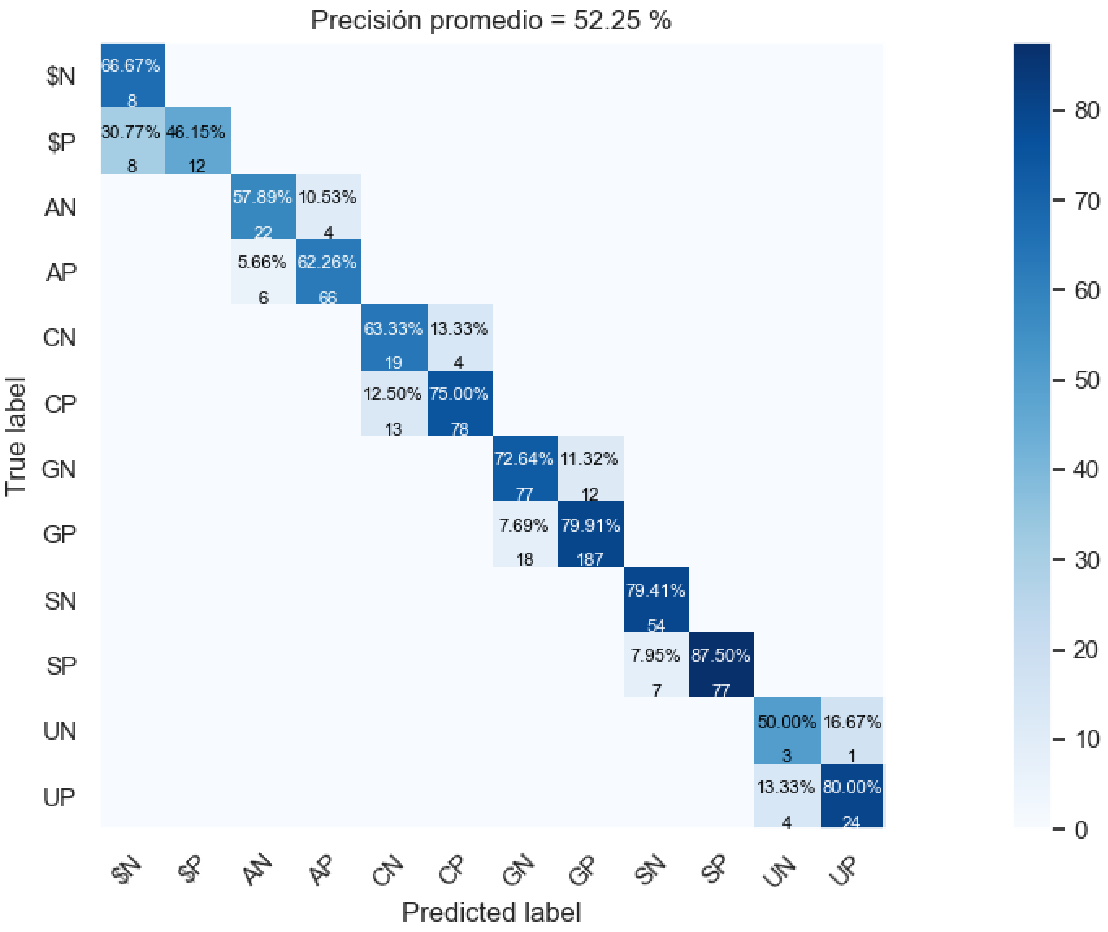

Epoch: 4
True label
$-
$N
$P
A-
AN
AP
C-
CN
CP
G-
GN
GP
S-
SN
SP
U-
UN
UP
98.88% 797
0.62% 5
0.50% 4
25.00% 3
75.00% 9
23.08% 6
15.38% 4
61.54% 16
93.70% 654
1.43% 10
4.87% 34
21.05% 8
57.89% 22
21.05% 8
24.53% 26
4.72% 5
70.75% 75
97.60% 691
0.99% 7
1.41% 10
16.67% 5
56.67% 17
26.67% 8
15.38% 16
5.77% 6
78.85% 82
85.86% 431
5.78% 29
8.37% 42
23.58% 25
58.49% 62
17.92% 19
16.24% 38
5.98% 14
77.78% 182
97.23% 667
0.87% 6
1.90% 13
16.18% 11
82.35% 56
1.47% 1
5.68% 5
2.27% 2
92.05% 81
99.50% 802
0.50% 4
50.00% 3
50.00% 3
6.67% 2
6.67% 2
86.67% 26
$-
$N
$P
A-
AN
AP
C-
CN
CP
G-
GN
GP
S-
SN
SP
U-
UN
UP
Predicted label
80
60
40
20
0

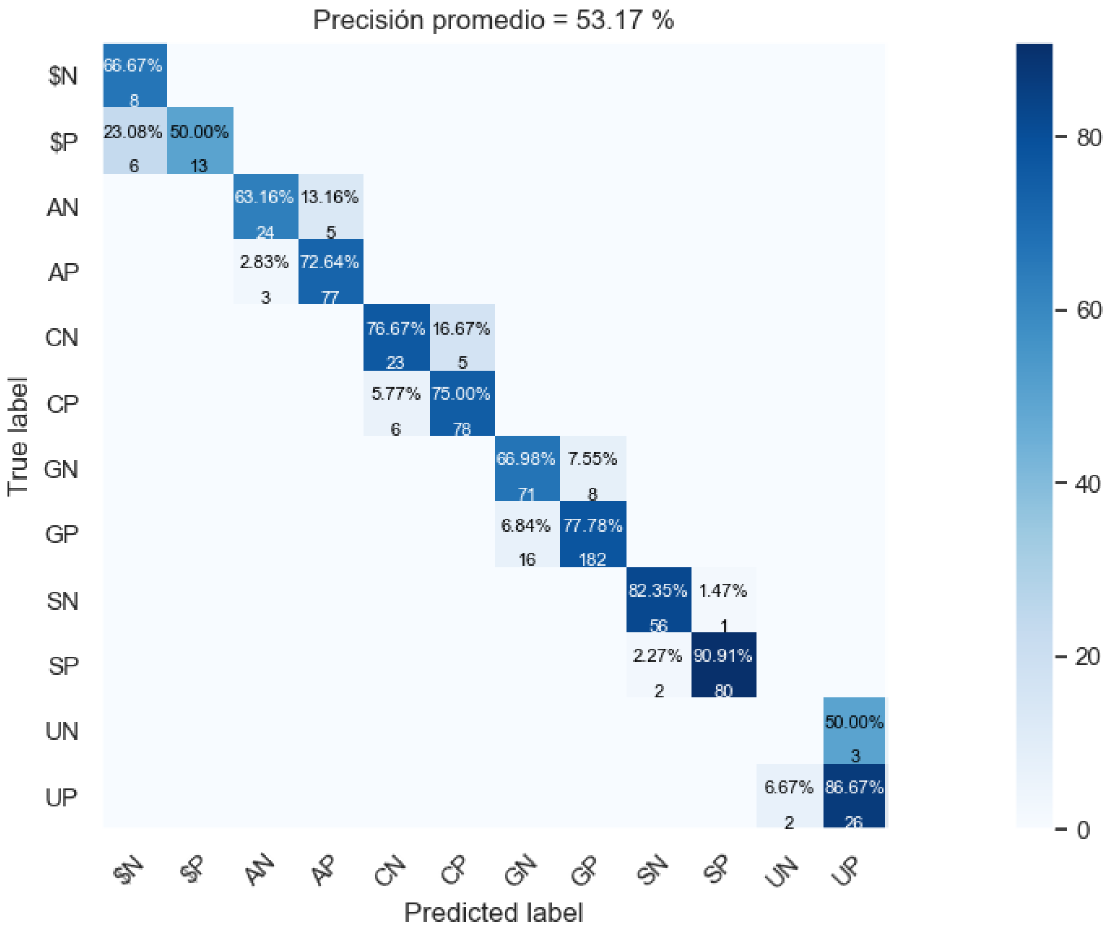

Epoch: 5
$-
99.01%
798
0.74%
6
0.25%
2
$N
33.33%
4
66.67%
8
$P
23.08%
6
23.08%
6
53.85%
14
A-
94.13%
657
1.43%
10
4.44%
31
AN
21.05%
8
63.16%
24
15.79%
6
AP
24.53%
26
3.77%
4
71.70%
76
C-
97.88%
693
0.99%
7
1.13%
8
CN
13.33%
4
63.33%
19
23.33%
7
CP
13.46%
14
3.85%
4
82.69%
86
G-
86.06%
432
5.98%
30
7.97%
40
GN
27.36%
29
61.32%
65
11.32%
12
GP
17.09%
40
6.41%
15
76.50%
179
S-
97.67%
670
0.87%
6
1.46%
10
SN
17.65%
12
79.41%
54
2.94%
2
SP
6.82%
6
93.18%
82
U-
99.50%
802
0.50%
4
UN
50.00%
3
16.67%
1
33.33%
2
UP
6.67%
2
6.67%
2
86.67%
26
True label
$-
$N
$P
A-
AN
AP
C-
CN
CP
G-
GN
GP
S-
SN
SP
U-
UN
UP
Predicted label
80
60
40
20
0

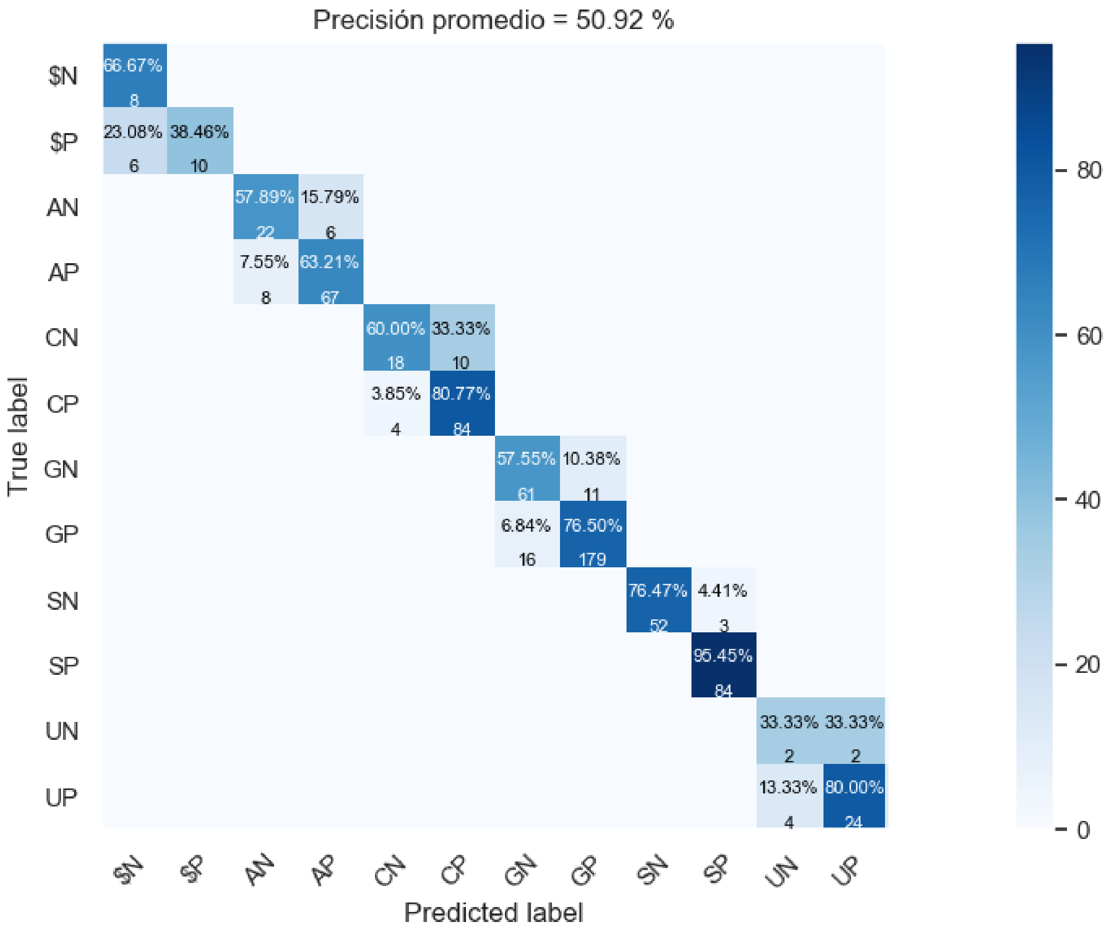


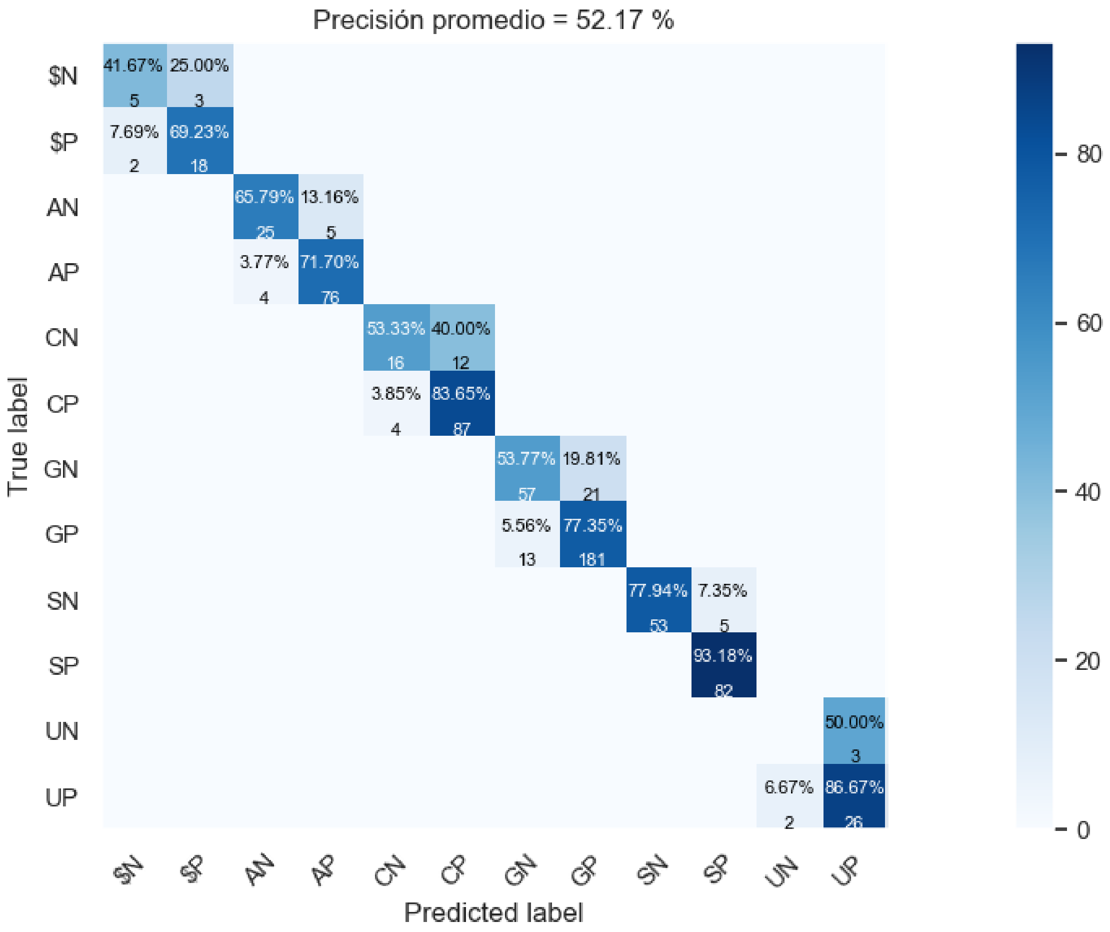
[150]:

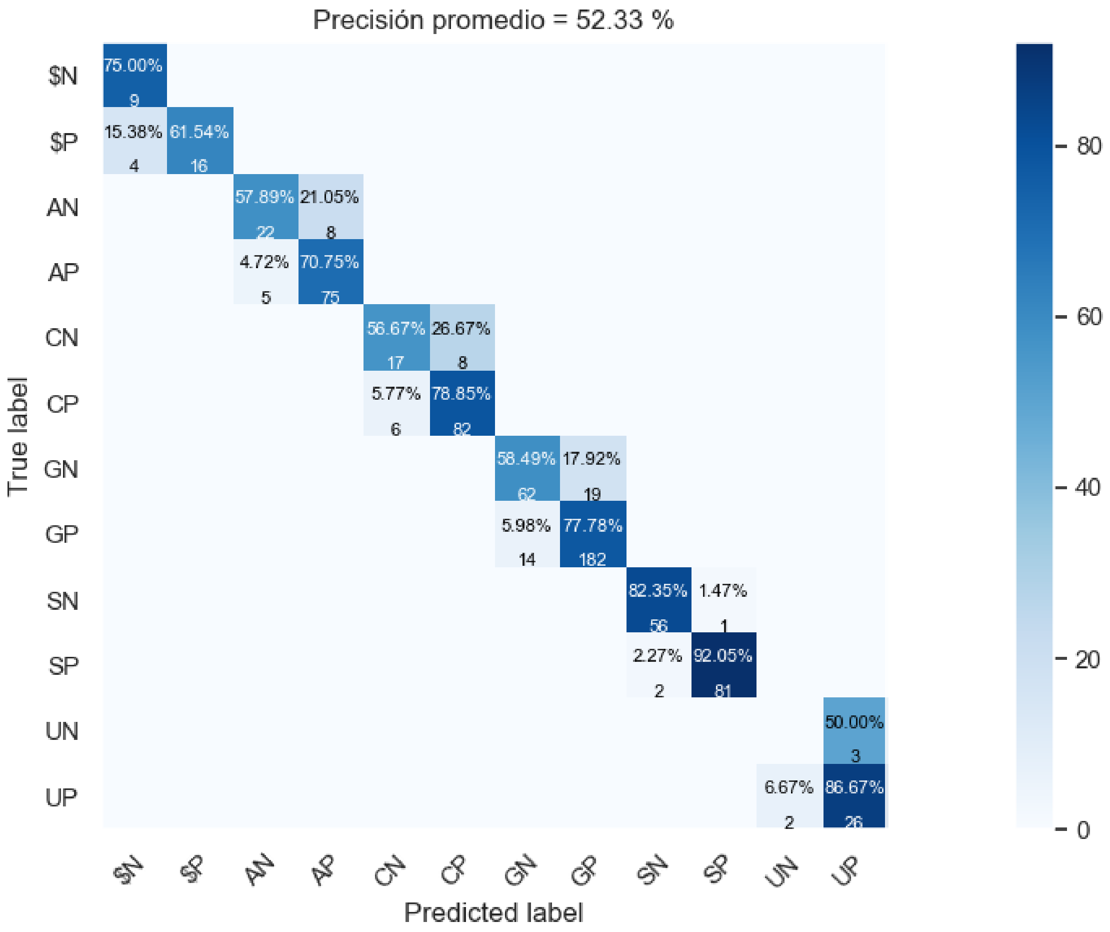
```
for k in test.keys():
    if 'y_label_pred_' in k:
        y_pred = y_preds[k]
        k = k.replace('y_label_pred_', '')
        plot_confusion_matrix(y_real, y_pred, classes=unique_labels(y_real), clean=True, figsize=(16, 6), dpi=100)
```


Precisión promedio = 52.25 %
True label
$N
$P
AN
AP
CN
CP
GN
GP
SN
SP
UN
UP
66.67%
8
30.77%
8
46.15%
12
57.89%
22
10.53%
4
5.66%
6
62.26%
66
63.33%
19
13.33%
4
12.50%
13
75.00%
78
72.64%
77
11.32%
12
7.69%
18
79.91%
187
79.41%
54
7.95%
7
87.50%
77
50.00%
3
16.67%
1
13.33%
4
80.00%
24
$N
$P
AN
AP
CN
CP
GN
GP
SN
SP
UN
UP
Predicted label
80
70
60
50
40
30
20
10
0

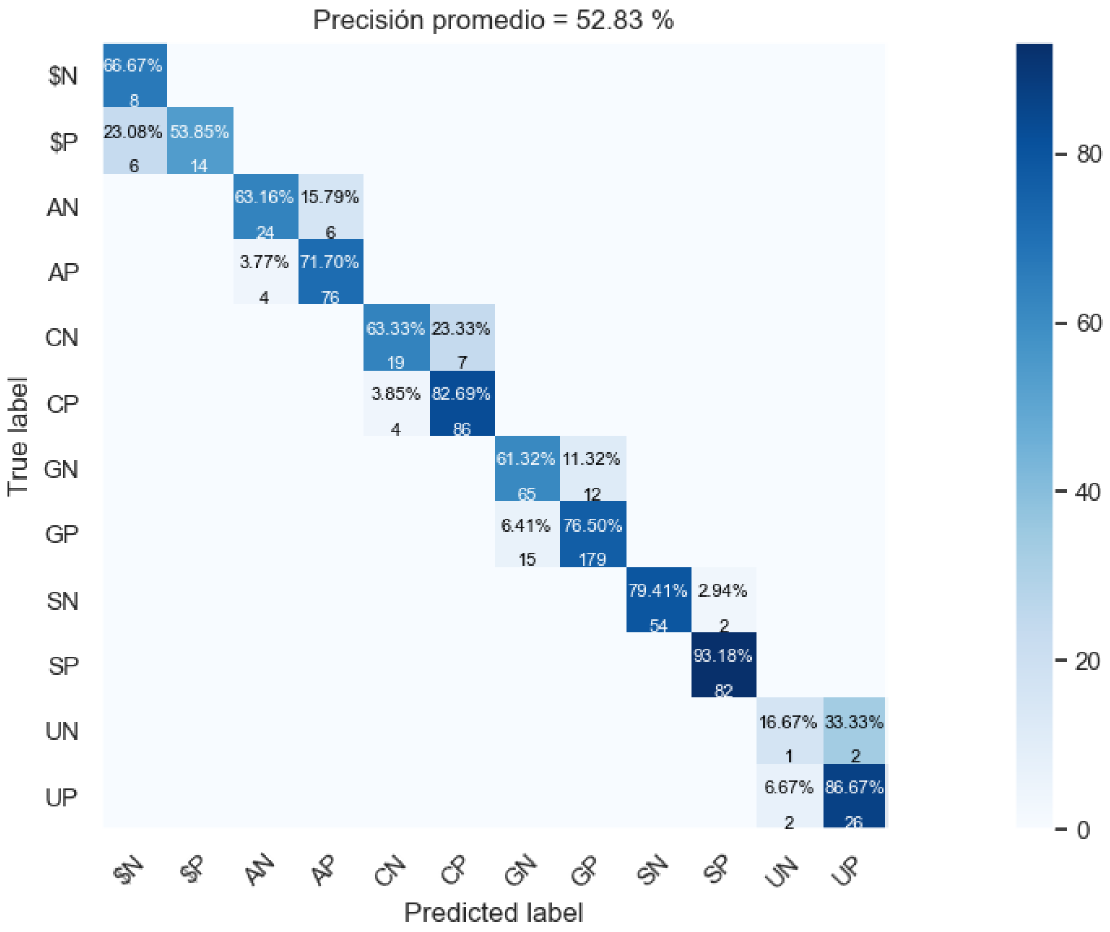

Precisión promedio = 53.17 %
True label
$N
66.67%
8
$P
23.08%
6
50.00%
13
AN
63.16%
24
13.16%
5
AP
2.83%
3
72.64%
77
CN
76.67%
23
16.67%
5
CP
5.77%
6
75.00%
78
GN
66.98%
71
7.55%
8
GP
6.84%
16
77.78%
182
SN
82.35%
56
1.47%
1
SP
2.27%
2
90.91%
80
UN
50.00%
3
UP
6.67%
2
86.67%
26
$N
$P
AN
AP
CN
CP
GN
GP
SN
SP
UN
UP
Predicted label
80
60
40
20
0

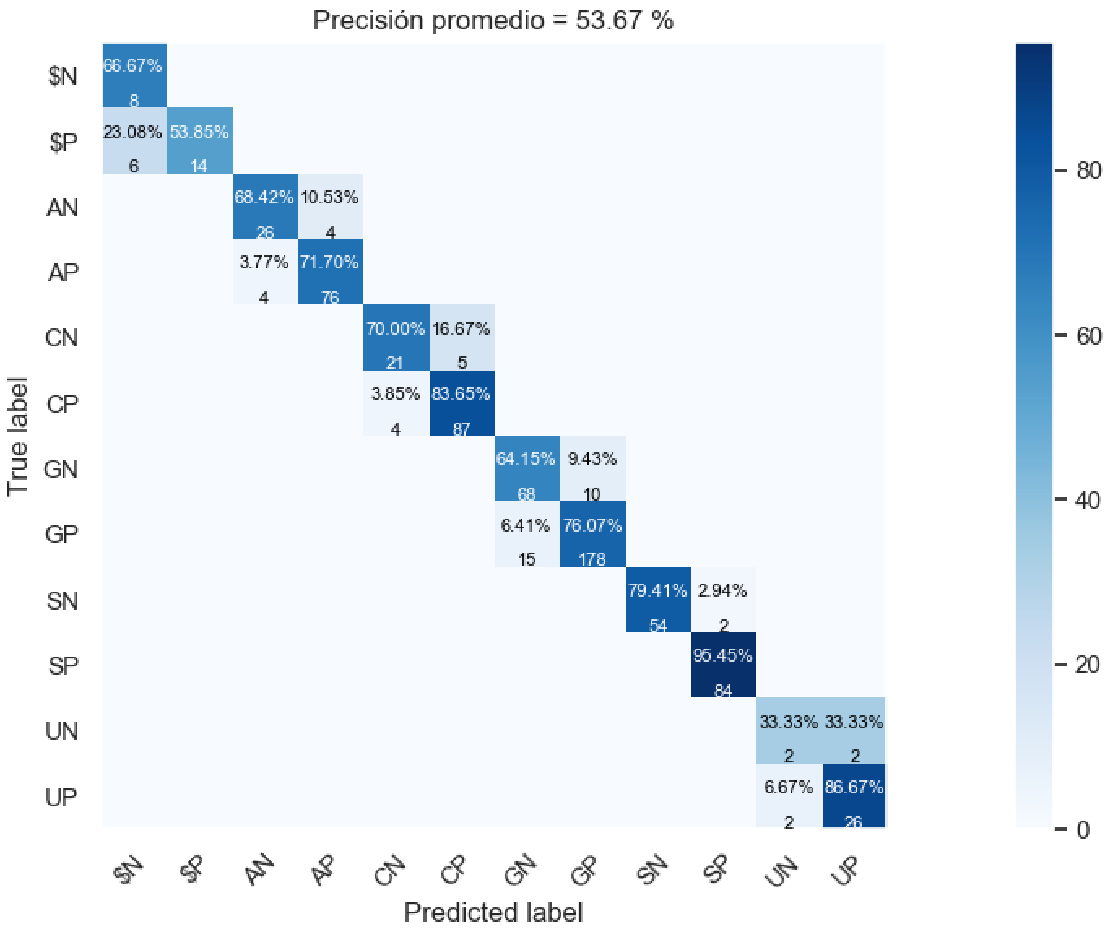

Precisión promedio = 50.92 %
True label
Predicted label
$N
$P
AN
AP
CN
CP
GN
GP
SN
SP
UN
UP
66.67%
8
23.08%
6
38.46%
10
57.89%
22
15.79%
6
7.55%
8
63.21%
67
60.00%
18
33.33%
10
3.85%
4
80.77%
84
57.55%
61
10.38%
11
6.84%
16
76.50%
179
76.47%
52
4.41%
3
95.45%
84
33.33%
2
33.33%
2
13.33%
4
80.00%
24
80
60
40
20
0


Precisión promedio = 52.17 %
$N
41.67%
5
25.00%
3
$P
7.69%
2
69.23%
18
AN
65.79%
25
13.16%
5
AP
3.77%
4
71.70%
76
CN
53.33%
16
40.00%
12
CP
3.85%
4
83.65%
87
GN
53.77%
57
19.81%
21
GP
5.56%
13
77.35%
181
SN
77.94%
53
7.35%
5
SP
93.18%
82
UN
50.00%
3
UP
6.67%
2
86.67%
26
True label
$N
$P
AN
AP
CN
CP
GN
GP
SN
SP
UN
UP
Predicted label
80
60
40
20
0



Precisión promedio = 52.33 %


Precisión promedio = 52.83 %
True label
Predicted label
$N
$P
AN
AP
CN
CP
GN
GP
SN
SP
UN
UP
66.67% 8
23.08% 6
53.85% 14
63.16% 24
15.79% 6
3.77% 4
71.70% 76
63.33% 19
23.33% 7
3.85% 4
82.69% 86
61.32% 65
11.32% 12
6.41% 15
76.50% 179
79.41% 54
2.94% 2
93.18% 82
16.67% 1
33.33% 2
6.67% 2
86.67% 26
80
60
40
20
0


Precisión promedio = 53.67 %
True label
Predicted label
$N
$P
AN
AP
CN
CP
GN
GP
SN
SP
UN
UP
66.67%
8
23.08%
6
53.85%
14
68.42%
26
10.53%
4
3.77%
4
71.70%
76
70.00%
21
16.67%
5
3.85%
4
83.65%
87
64.15%
68
9.43%
10
6.41%
15
76.07%
178
79.41%
54
2.94%
2
95.45%
84
33.33%
2
33.33%
2
6.67%
2
86.67%
26
80
60
40
20
0