```
from sklearn.metrics import f1_score, precision_score, recall_score
import pandas as pd
from google.colab import drive
import json
```

```
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
actual = []
with open('/content/drive/MyDrive/Cleaned_Sentences_Task/tags_original.txt', 'r') as file:
    actual = json.load(file)
```

```
print(len(actual))
```

```
662728
```

```
rnn = []
with open('/content/drive/MyDrive/Cleaned_Sentences_Task/rnn_predictions.txt', 'r') as file:
    rnn = json.load(file)
```

```
print(len(rnn))
```

```
662728
```

```
nltk = []
with open('/content/drive/MyDrive/Cleaned_Sentences_Task/nltk_predictions.txt', 'r') as file:
    nltk = json.load(file)
```

```
print(len(nltk))
```

```
662728
```

```
ann = []
with open('/content/drive/MyDrive/Cleaned_Sentences_Task/ann_predictions.txt', 'r') as file:
    ann = json.load(file)
```

```
print(len(ann))
```

```
22000
```

```
svm = []
with open('/content/drive/MyDrive/Cleaned_Sentences_Task/svm_predictions.txt', 'r') as file:
    svm = json.load(file)
```

```
print(len(svm))
```

```
22000
```

```
print(set(actual))
```

```
{'prp', 'nn', 'ech', 'rp', 'rb', 'ccc', 'cc', 'vaux', 'nst', 'qf', 'jj', 'nnp', 'dem', 'unk', 'qo', 'qc', 'inj', 'rdp', 'nnpc',
```

```
pos_to_tags = {
    "jj": "Adjective",
    "rb": "Adverb",
    "nn": "Noun",
    "prp": "Preposition",
    "vb": "Verb",
    "rp": "Particle",
    "cc": "Cordinating Conjuction",
    "vaux": "Auxiliary Verb",
    "vm": "Main Verb",
}
```

```
combinations = []
keys = list(pos_to_tags.keys())
for i in range(0, len(pos_to_tags)):
    for j in range(0, len(keys)):
        if i == j:
            continue
        combinations.append([keys[i], keys[j]])
```

```
print(keys)
```

```
['jj', 'rb', 'nn', 'prp', 'vb', 'rp', 'cc', 'vaux', 'vm']
```

```
print(len(combinations))
```

```
72
```

```
precision_1 = []
precision_2 = []
precision_3 = []
precision_4 = []
recall_1 = []
recall_2 = []
recall_3 = []
recall_4 = []
accuray_1 = []
accuray_2 = []
accuray_3 = []
accuracy_4 = []
actuals = []
used_tags = []
```

```
def tag_2_tag(actual, nltk_tags, ml_tags, deep_tags, rnn_tags, tag_1, tag_2):
  count_actual = []
  count_nltk = []
  count_ml = []
  count_deep = []
  count_rnn = []

  for i in range(0, len(actual) - 1, 2):
    if tag_1 in actual[i] and tag_2 in actual[i + 1]:
      count_actual.append(1)
    else:
```

```
    count_actual.append(0)
  if tag_1 in nltk_tags[i] and tag_2 in nltk_tags[i + 1]:
    count_nltk.append(1)
  else:
    count_nltk.append(0)
  if tag_1 in ml_tags[i] and tag_2 in ml_tags[i + 1]:
    count_ml.append(1)
  else:
    count_ml.append(0)
  if tag_1 in deep_tags[i] and tag_2 in deep_tags[i + 1]:
    count_deep.append(1)
  else:
    count_deep.append(0)
  if tag_1 in rnn_tags[i] and tag_2 in rnn_tags[i + 1]:
    count_rnn.append(1)
  else:
    count_rnn.append(0)
if count_actual.count(1) < 50:
    return
used_tags.append([pos_to_tags[tag_1], pos_to_tags[tag_2]])
accuracy_nltk = f1_score(count_actual, count_nltk, zero_division=True)
accuracy_ml = f1_score(count_actual, count_ml, zero_division=True)
accuracy_deep = f1_score(count_actual, count_deep, zero_division=True)
accuracy_rnn = f1_score(count_actual, count_rnn, zero_division=True)
precision_nltk = precision_score(count_actual, count_nltk, zero_division=True)
precision_ml = precision_score(count_actual, count_ml, zero_division=True)
precision_deep = precision_score(count_actual, count_deep, zero_division=True)
precision_rnn = precision_score(count_actual, count_rnn, zero_division=True)
recall_nltk = recall_score(count_actual, count_nltk, zero_division=True)
recall_ml = recall_score(count_actual, count_ml, zero_division=True)
recall_deep = recall_score(count_actual, count_deep, zero_division=True)
recall_rnn = recall_score(count_actual, count_rnn, zero_division=True)

precision_1.append(precision_nltk)
precision_2.append(precision_ml)
precision_3.append(precision_deep)
precision_4.append(precision_rnn)
recall_1.append(recall_nltk)
recall_2.append(recall_ml)
recall_3.append(recall_deep)
recall_4.append(recall_rnn)
accuray_1.append(accuracy_nltk)
```

```
accuray_2.append(accuracy_ml)
accuray_3.append(accuracy_deep)
accuracy_4.append(accuracy_rnn)

actuals.append(count_actual.count(1))

print("--> {0} And {1}".format(pos_to_tags[tag_1], pos_to_tags[tag_2]))
print("=================== NLTK TAGGER ====================")
print("Count of Actual = {0}".format(count_actual.count(1)))
print("accuracy =", accuracy_nltk)
print("precision =", precision_nltk)
print("recall =", recall_nltk)

print()
print("--> {0} And {1}".format(pos_to_tags[tag_1], pos_to_tags[tag_2]))
print("========== MACHINE LEARNING TAGGER (SVM) ===========")
print("Count of Actual = {0}".format(count_actual.count(1)))
print("accuracy =", accuracy_ml)
print("precision =", precision_ml)
print("recall =", recall_ml)

print()
print("--> {0} And {1}".format(pos_to_tags[tag_1], pos_to_tags[tag_2]))
print("=========== DEEP LEARNING TAGGER (ANN) =============")
print("Count of Actual = {0}".format(count_actual.count(1)))
print("accuracy =", accuracy_deep)
print("precision =", precision_deep)
print("recall =", recall_deep)

print()
print("--> {0} And {1}".format(pos_to_tags[tag_1], pos_to_tags[tag_2]))
print("=========== DEEP LEARNING TAGGER (RNN) =============")
print("Count of Actual = {0}".format(count_actual.count(1)))
print("accuracy =", accuracy_rnn)
print("precision =", precision_rnn)
print("recall =", recall_rnn)
print("====================================================")
print()
print()
```

```
print("========== For First 22,000 Data: ==========")
print()
for i in combinations:
    tag_2_tag(actual[0: 22000], nltk[0: 22000], svm, ann, rnn[0: 22000], i[0], i[1])

```

```
========== For First 22,000 Data: ==========

--> Adjective And Noun
=================== NLTK TAGGER ====================
Count of Actual = 385
accuracy = 0.9230769230769231
precision = 0.9267015706806283
recall = 0.9194805194805195

--> Adjective And Noun
========== MACHINE LEARNING TAGGER (SVM) ===========
Count of Actual = 385
accuracy = 0.884318766066838
precision = 0.8753180661577609
recall = 0.8935064935064935

--> Adjective And Noun
=========== DEEP LEARNING TAGGER (ANN) =============
Count of Actual = 385
accuracy = 0.8906455862977603
precision = 0.9037433155080213
recall = 0.8779220779220779

--> Adjective And Noun
=========== DEEP LEARNING TAGGER (RNN) =============
Count of Actual = 385
accuracy = 0.976923076923077
precision = 0.9645569620253165
recall = 0.9896103896103896
====================================================


--> Adjective And Main Verb
=================== NLTK TAGGER ====================
Count of Actual = 214
accuracy = 0.9577464788732395
precision = 0.9622641509433962
```

```
recall = 0.9532710280373832

--> Adjective And Main Verb
========== MACHINE LEARNING TAGGER (SVM) ===========
Count of Actual = 214
accuracy = 0.7944444444444445
precision = 0.9794520547945206
recall = 0.6682242990654206

--> Adjective And Main Verb
=========== DEEP LEARNING TAGGER (ANN) =============
Count of Actual = 214
accuracy = 0.9211822660098521
precision = 0.9739583333333334
recall = 0.8738317757009346

--> Adjective And Main Verb
=========== DEEP LEARNING TAGGER (RNN) =============
Count of Actual = 214
accuracy = 0.9836829836829836
precision = 0.9813953488372092
recall = 0.985981308411215
```

```python
print(len(used_tags))
print(len(actuals))
```

```
24
24
```

```python
index = [i[0] + " And " + i[1] for i in used_tags]
```

```python
index[0]
```

```
'Adjective And Noun'
```

```python
# FOR NLTK TAGGER
data_frame = pd.DataFrame(actuals, index=index,
                          columns=["Count_Actual"])
data_frame["precision"] = precision_1
```

```
data_frame["recall"] = recall_1
data_frame["accuracy"] = accuray_1
print("==================== NLTK TAGGER ====================")
print(data_frame.to_string())
```

```
===================== NLTK TAGGER =====================
                                            Count_Actual  precision    recall  accuracy
Adjective And Noun                                   385   0.926702  0.919481  0.923077
Adjective And Main Verb                              214   0.962264  0.953271  0.957746
Noun And Adjective                                   111   0.904348  0.936937  0.920354
Noun And Preposition                                  52   0.911111  0.788462  0.845361
Noun And Particle                                    110   0.950000  0.863636  0.904762
Noun And Cordinating Conjuction                      112   0.911504  0.919643  0.915556
Noun And Main Verb                                   580   0.981238  0.901724  0.939802
Preposition And Adjective                             52   0.888889  0.923077  0.905660
Preposition And Noun                                 245   0.958333  0.938776  0.948454
Preposition And Particle                              89   0.915663  0.853933  0.883721
Preposition And Main Verb                             51   1.000000  0.882353  0.937500
Particle And Adjective                                78   0.900000  0.923077  0.911392
Particle And Noun                                    318   0.954984  0.933962  0.944356
Particle And Preposition                              72   0.943396  0.694444  0.800000
Particle And Main Verb                                77   0.957746  0.883117  0.918919
Cordinating Conjuction And Adjective                  54   0.818182  0.833333  0.825688
Cordinating Conjuction And Noun                      266   0.944444  0.958647  0.951493
Cordinating Conjuction And Preposition               115   0.844262  0.895652  0.869198
Cordinating Conjuction And Particle                  119   0.840000  0.882353  0.860656
Auxiliary Verb And Cordinating Conjuction            120   0.651163  0.933333  0.767123
Main Verb And Noun                                    82   0.907895  0.841463  0.873418
Main Verb And Particle                                55   0.863636  0.690909  0.767677
Main Verb And Cordinating Conjuction                 214   0.939024  0.719626  0.814815
Main Verb And Auxiliary Verb                         542   0.913696  0.898524  0.906047
```

```
# FOR ML(SVM) TAGGER
data_frame = pd.DataFrame(actuals, index=index,
                      columns=["Count_Actual"])
data_frame["precision"] = precision_2
data_frame["recall"] = recall_2
data_frame["accuracy"] = accuray_2
print("==================== ML(SVM) TAGGER ====================")
print(data_frame.to_string())
```

```
==================== ML(SVM) TAGGER ====================
                                                 Count_Actual  precision    recall  accuracy
Adjective And Noun                                        385   0.875318  0.893506  0.884319
Adjective And Main Verb                                   214   0.979452  0.668224  0.794444
Noun And Adjective                                        111   0.767442  0.891892  0.825000
Noun And Preposition                                       52   0.759259  0.788462  0.773585
Noun And Particle                                         110   0.834783  0.872727  0.853333
Noun And Cordinating Conjuction                           112   0.868852  0.946429  0.905983
Noun And Main Verb                                        580   0.907298  0.793103  0.846366
Preposition And Adjective                                  52   0.863636  0.730769  0.791667
Preposition And Noun                                      245   0.845324  0.959184  0.898662
Preposition And Particle                                   89   0.949367  0.842697  0.892857
Preposition And Main Verb                                  51   0.954545  0.823529  0.884211
Particle And Adjective                                     78   0.904762  0.730769  0.808511
Particle And Noun                                         318   0.840220  0.959119  0.895742
Particle And Preposition                                   72   0.901639  0.763889  0.827068
Particle And Main Verb                                     77   0.935484  0.753247  0.834532
Cordinating Conjuction And Adjective                       54   0.795918  0.722222  0.757282
Cordinating Conjuction And Noun                           266   0.896057  0.939850  0.917431
Cordinating Conjuction And Preposition                    115   0.948980  0.808696  0.873239
Cordinating Conjuction And Particle                       119   0.940594  0.798319  0.863636
Auxiliary Verb And Cordinating Conjuction                 120   0.660000  0.825000  0.733333
Main Verb And Noun                                         82   0.701031  0.829268  0.759777
Main Verb And Particle                                     55   0.787234  0.672727  0.725490
Main Verb And Cordinating Conjuction                      214   0.940397  0.663551  0.778082
Main Verb And Auxiliary Verb                              542   0.920771  0.793358  0.852329
```

```python
# FOR DEEP(ANN) TAGGER
data_frame = pd.DataFrame(actuals, index=index,
                          columns=["Count_Actual"])
data_frame["precision"] = precision_3
data_frame["recall"] = recall_3
data_frame["accuracy"] = accuray_3
print("==================== DEEP(ANN) TAGGER ====================")
print(data_frame.to_string())
```

```
==================== DEEP(ANN) TAGGER ====================
                                                 Count_Actual  precision    recall  accuracy
Adjective And Noun                                        385   0.903743  0.877922  0.890646
Adjective And Main Verb                                   214   0.973958  0.873832  0.921182
```

```
    Noun And Adjective                                111   0.757812  0.873874  0.811715
    Noun And Preposition                               52   0.913043  0.807692  0.857143
    Noun And Particle                                 110   0.960000  0.872727  0.914286
    Noun And Cordinating Conjuction                   112   0.954955  0.946429  0.950673
    Noun And Main Verb                                580   0.912397  0.951724  0.931646
    Preposition And Adjective                          52   0.865385  0.865385  0.865385
    Preposition And Noun                              245   0.879245  0.951020  0.913725
    Preposition And Particle                           89   0.944444  0.955056  0.949721
    Preposition And Main Verb                          51   0.897959  0.862745  0.880000
    Particle And Adjective                             78   0.891892  0.846154  0.868421
    Particle And Noun                                 318   0.878261  0.952830  0.914027
    Particle And Preposition                           72   0.931507  0.944444  0.937931
    Particle And Main Verb                             77   0.931507  0.883117  0.906667
    Cordinating Conjuction And Adjective               54   0.803571  0.833333  0.818182
    Cordinating Conjuction And Noun                   266   0.950943  0.947368  0.949153
    Cordinating Conjuction And Preposition            115   0.908333  0.947826  0.927660
    Cordinating Conjuction And Particle               119   0.904000  0.949580  0.926230
    Auxiliary Verb And Cordinating Conjuction         120   0.685535  0.908333  0.781362
    Main Verb And Noun                                 82   0.822222  0.902439  0.860465
    Main Verb And Particle                             55   0.818182  0.818182  0.818182
    Main Verb And Cordinating Conjuction              214   0.968553  0.719626  0.825737
    Main Verb And Auxiliary Verb                      542   0.975877  0.821033  0.891784
```

```python
# FOR DEEP(RNN) TAGGER
data_frame = pd.DataFrame(actuals, index=index,
                          columns=["Count_Actual"])
data_frame["precision"] = precision_4
data_frame["recall"] = recall_4
data_frame["accuracy"] = accuracy_4
print("==================== DEEP(RNN) TAGGER ====================")
print(data_frame.to_string())
```

```
    ===================== DEEP(RNN) TAGGER =====================
                                             Count_Actual  precision    recall  accuracy
    Adjective And Noun                                385   0.964557  0.989610  0.976923
    Adjective And Main Verb                           214   0.981395  0.985981  0.983683
    Noun And Adjective                                111   0.964602  0.981982  0.973214
    Noun And Preposition                               52   1.000000  1.000000  1.000000
    Noun And Particle                                 110   1.000000  0.990909  0.995434
    Noun And Cordinating Conjuction                   112   0.990991  0.982143  0.986547
    Noun And Main Verb                                580   0.991334  0.986207  0.988764
    Preposition And Adjective                          52   0.928571  1.000000  0.962963
    Preposition And Noun                              245   0.995868  0.983673  0.989733
    Preposition And Particle                           89   0.945055  0.966292  0.955556
    Preposition And Main Verb                          51   0.962264  1.000000  0.980769
    Particle And Adjective                             78   0.927711  0.987179  0.956522
    Particle And Noun                                 318   0.996795  0.977987  0.987302
    Particle And Preposition                           72   0.931507  0.944444  0.937931
    Particle And Main Verb                             77   0.974684  1.000000  0.987179
    Cordinating Conjuction And Adjective               54   0.943396  0.925926  0.934579
    Cordinating Conjuction And Noun                   266   0.984791  0.973684  0.979206
    Cordinating Conjuction And Preposition            115   0.971154  0.878261  0.922374
    Cordinating Conjuction And Particle               119   0.971963  0.873950  0.920354
    Auxiliary Verb And Cordinating Conjuction         120   0.701613  0.725000  0.713115
    Main Verb And Noun                                 82   0.975904  0.987805  0.981818
    Main Verb And Particle                             55   0.933333  0.763636  0.840000
    Main Verb And Cordinating Conjuction              214   0.844340  0.836449  0.840376
    Main Verb And Auxiliary Verb                      542   0.965649  0.933579  0.949343
```

```python
precision_5 = []
recall_5 = []
accuracy_5 = []
precision_6 = []
accuracy_6 = []
recall_6 = []
actual_big = []
used_tags_bigs = []
```

```python
def tag_2_tag_big(actual, nltk_tags, rnn_tags, tag_1, tag_2):
  count_actual = []
  count_nltk = []
  count_rnn = []

  for i in range(0, len(actual) - 1, 2):
    if tag_1 in actual[i] and tag_2 in actual[i + 1]:
      count_actual.append(1)
    else:
      count_actual.append(0)
    if tag_1 in nltk_tags[i] and tag_2 in nltk_tags[i + 1]:
      count_nltk.append(1)
    else:
      count_nltk.append(0)
    if tag_1 in rnn_tags[i] and tag_2 in rnn_tags[i + 1]:
      count_rnn.append(1)
    else:
      count_rnn.append(0)
  if count_actual.count(1) < 300:
    return
  used_tags_bigs.append([pos_to_tags[tag_1], pos_to_tags[tag_2]])
  accuracy_nltk = f1_score(count_actual, count_nltk, zero_division=True)
  accuracy_rnn = f1_score(count_actual, count_rnn, zero_division=True)
  precision_nltk = precision_score(count_actual, count_nltk, zero_division=True)
  precision_rnn = precision_score(count_actual, count_rnn, zero_division=True)
  recall_nltk = recall_score(count_actual, count_nltk, zero_division=True)
  recall_rnn = recall_score(count_actual, count_rnn, zero_division=True)

  precision_5.append(precision_nltk)
  precision_6.append(precision_rnn)
  recall_5.append(recall_nltk)
  recall_6.append(recall_rnn)
  accuracy_5.append(accuracy_nltk)
  accuracy_6.append(accuracy_rnn)

  actual_big.append(count_actual.count(1))
  print("--> {0} And {1}".format(pos_to_tags[tag_1], pos_to_tags[tag_2]))
  print("=================== NLTK TAGGER ====================")
  print("Count of Actual = {}".format(count_actual.count(1)))
  print("accuracy =", accuracy_nltk)
  print("precision =", precision_nltk)
  print("recall =", recall_nltk)

  print()
  print("--> {0} And {1}".format(pos_to_tags[tag_1], pos_to_tags[tag_2]))
  print("=========== DEEP LEARNING TAGGER (RNN) =============")
  print("Count of Actual = {}".format(count_actual.count(1)))
  print("accuracy =", accuracy_rnn)
  print("precision =", precision_rnn)
  print("recall =", recall_rnn)
  print("====================================================")
  print()
  print()
```

```
print("================== For All Data ====================")
print()
for i in combinations:
  tag_2_tag_big(actual, nltk, rnn, i[0], i[1])
```

```
    ====================================================



    --> Cordinating Conjuction And Particle
    =================== NLTK TAGGER ====================
    Count of Actual = 3071
    accuracy = 0.8886783514921839
    precision = 0.8626609442060086
    recall = 0.9163139042657115

    --> Cordinating Conjuction And Particle
    =========== DEEP LEARNING TAGGER (RNN) =============
    Count of Actual = 3071
    accuracy = 0.9044607190412783
    precision = 0.9250936329588015
    recall = 0.8847281015955715
    ====================================================



    --> Auxiliary Verb And Noun
    =================== NLTK TAGGER ====================
    Count of Actual = 1189
    accuracy = 0.8177848352520841
    precision = 0.7744360902255639
    recall = 0.8662741799831791

    --> Auxiliary Verb And Noun
    =========== DEEP LEARNING TAGGER (RNN) =============
    Count of Actual = 1189
    accuracy = 0.943744752308984
    precision = 0.942162615255658
    recall = 0.945332211942809
    ====================================================



    --> Auxiliary Verb And Preposition
    =================== NLTK TAGGER ====================
    Count of Actual = 859
    accuracy = 0.7038988408851423
    precision = 0.6429258902791145
    recall = 0.7776484284051223

    --> Auxiliary Verb And Preposition
    =========== DEEP LEARNING TAGGER (RNN) =============
    Count of Actual = 859
    accuracy = 0.7626943005181348
    precision = 0.6872082166199813
    recall = 0.8568102444703143
    ====================================================


    --> Auxiliary Verb And Particle
    ==================== NLTK TAGGER ====================
    Count of Actual = 931
    accuracy = 0.6875000000000001
    precision = 0.6387096774193548
    recall = 0.7443609022556391

    --> Auxiliary Verb And Particle
```

```
print(len(used_tags_bigs))
```

```
    33
```

```
index = [i[0] + " And " + i[1] for i in used_tags_bigs]
```

```
print(index[0])
```

```
    Adjective And Noun
```

```
# FOR NLTK TAGGER
data_frame = pd.DataFrame(actual_big, index=index,
                          columns=["Count_Actual"])
data_frame["precision"] = precision_5
data_frame["recall"] = recall_5
data_frame["accuracy"] = accuracy_5
```

```
print("===================================== NLTK TAGGER ====================================")
print(data_frame.to_string())
```

```
==================================== NLTK TAGGER ====================================
                                          Count_Actual  precision    recall  accuracy
Adjective And Noun                               11942   0.922283  0.950008  0.935940
Adjective And Main Verb                           6197   0.962151  0.935291  0.948531
Adverb And Noun                                    634   0.897600  0.884858  0.891183
Adverb And Particle                                349   0.830325  0.659026  0.734824
Noun And Adjective                                3850   0.893803  0.929091  0.911105
Noun And Adverb                                    373   0.955357  0.860590  0.905501
Noun And Preposition                              1645   0.928811  0.840729  0.882578
Noun And Particle                                 3420   0.939857  0.881871  0.909941
Noun And Cordinating Conjuction                   3804   0.957509  0.941903  0.949642
Noun And Main Verb                               17536   0.977546  0.903684  0.939165
Preposition And Adjective                         1725   0.892440  0.841739  0.866348
Preposition And Noun                              6917   0.951310  0.929305  0.940178
Preposition And Particle                          2387   0.926795  0.827398  0.874281
Preposition And Main Verb                         2013   0.963473  0.904123  0.932855
Particle And Adjective                            2467   0.895937  0.875963  0.885837
Particle And Noun                                 9061   0.943718  0.923408  0.933452
Particle And Preposition                          2052   0.911602  0.723684  0.806846
Particle And Main Verb                            3036   0.945889  0.886693  0.915335
Cordinating Conjuction And Adjective              1456   0.745599  0.901786  0.816288
Cordinating Conjuction And Noun                   7850   0.947154  0.947516  0.947335
Cordinating Conjuction And Preposition            2977   0.861908  0.928787  0.894099
Cordinating Conjuction And Particle               3071   0.862661  0.916314  0.888678
Auxiliary Verb And Noun                           1189   0.774436  0.866274  0.817785
Auxiliary Verb And Preposition                     859   0.642926  0.777648  0.703899
Auxiliary Verb And Particle                        931   0.638710  0.744361  0.687500
Auxiliary Verb And Cordinating Conjuction         3037   0.601848  0.922292  0.728384
Auxiliary Verb And Main Verb                       320   0.817518  0.700000  0.754209
Main Verb And Adjective                            452   0.849188  0.809735  0.828992
Main Verb And Noun                                2106   0.869454  0.809592  0.838456
Main Verb And Preposition                         1069   0.861111  0.521983  0.649971
Main Verb And Particle                            1418   0.878307  0.585331  0.702497
Main Verb And Cordinating Conjuction              6097   0.925667  0.694440  0.793553
Main Verb And Auxiliary Verb                     17521   0.933345  0.915073  0.924119
```

```
# FOR RNN TAGGER
data_frame = pd.DataFrame(actual_big, index=index,
                          columns=["Count_Actual"])
```

```
                        columns=["Count_Actual"])
data_frame["precision"] = precision_6
data_frame["recall"] = recall_6
data_frame["accuracy"] = accuracy_6
print("================================= DEEP TAGGER (RNN) ==================================")
print(data_frame.to_string())
```

```
================================= DEEP TAGGER (RNN) ==================================
                                           Count_Actual  precision    recall  accuracy
Adjective And Noun                                11942   0.960263  0.977391  0.968751
Adjective And Main Verb                            6197   0.982825  0.988059  0.985435
Adverb And Noun                                     634   0.944012  0.957413  0.950666
Adverb And Particle                                 349   0.901198  0.862464  0.881406
Noun And Adjective                                 3850   0.948744  0.961558  0.955108
Noun And Adverb                                     373   0.955801  0.927614  0.941497
Noun And Preposition                               1645   0.980928  0.937994  0.958981
Noun And Particle                                  3420   0.984671  0.957895  0.971098
Noun And Cordinating Conjuction                    3804   0.979576  0.970820  0.975178
Noun And Main Verb                                17536   0.985636  0.982151  0.983890
Preposition And Adjective                          1725   0.923973  0.965217  0.944145
Preposition And Noun                               6917   0.981983  0.969206  0.975553
Preposition And Particle                           2387   0.949378  0.927105  0.938109
Preposition And Main Verb                          2013   0.987988  0.980626  0.984293
Particle And Adjective                             2467   0.933099  0.966761  0.949632
Particle And Noun                                  9061   0.981479  0.965015  0.973178
Particle And Preposition                           2052   0.914850  0.879630  0.896894
Particle And Main Verb                             3036   0.976806  0.971014  0.973902
Cordinating Conjuction And Adjective               1456   0.908184  0.937500  0.922609
Cordinating Conjuction And Noun                    7850   0.973552  0.965987  0.969755
Cordinating Conjuction And Preposition             2977   0.930767  0.885119  0.907369
Cordinating Conjuction And Particle                3071   0.925094  0.884728  0.904461
Auxiliary Verb And Noun                            1189   0.942163  0.945332  0.943745
Auxiliary Verb And Preposition                      859   0.687208  0.856810  0.762694
Auxiliary Verb And Particle                         931   0.690129  0.863588  0.767176
Auxiliary Verb And Cordinating Conjuction          3037   0.691459  0.722423  0.706602
Auxiliary Verb And Main Verb                        320   0.986885  0.940625  0.963200
Main Verb And Adjective                             452   0.932462  0.946903  0.939627
Main Verb And Noun                                 2106   0.948292  0.949193  0.948742
Main Verb And Preposition                          1069   0.890013  0.658559  0.756989
Main Verb And Particle                             1418   0.910245  0.708039  0.796509
Main Verb And Cordinating Conjuction               6097   0.851956  0.846646  0.849293
Main Verb And Auxiliary Verb                      17521   0.970837  0.942412  0.956413
```

```
```