# Doğal Dil İşleme - Ödev 1 Rapor

Mümtaz Cem Eriş
*Bilgisayar Mühendisliği*
*İstanbul Teknik Üniversitesi*
504222506
erismu@itu.edu.tr

### *A. Kod Colab ve Github Linki*

Colab'ta çalışma zamanı silindiğinde yüklenen dosyalar da silinebileceğinden dolayı proje Github'a da yüklenmiştir. Colab'a github'taki "/data" klasörü yüklenmelidir.

Proje ipynb Colab linkidir. Sonuçlarla verilmiştir yavaş çalışabilir!

Proje ipynb Colab linkidir. Sonuçsuz verilmiştir, hızlı çalışır.

Proje Github linkidir.

### *B. Açık bir NER etiketli veri kümesi: Reuters Derlemi*

2003 yılında Conference on Natural Language Learning (CoNLL) adlı konferansta yayınlanmış olan ve referans sayısı oldukça yüksek olan [1] makalesinde bahsedilen Reuters haber hikayeleri bu ödevde kullanılmıştır. Bu hikayeler 1996 Ağustos ve 1997 Ağustos arası dönemden elde edilmiştir. Eğitim ve geliştirme veri kümesi 1996 Ağustos ayının 10 günlük dosyalarından toplanmıştır. Sınama verisi ise 1996 Aralık ayından alınmıştır [1].

Eğitim veri kümesinde 203621 token bulunmaktadır. Bu tokenlara denk gelen bazı etiketler ise Şekil 1'de verilmiştir.

| | Tokens | Labels |
|---|---|---|
| **0** | EU | B-ORG |
| **1** | rejects | O |
| **2** | German | B-MISC |
| **3** | call | O |
| **4** | to | O |
| **...** | ... | ... |
| **203616** | three | O |
| **203617** | Swansea | B-ORG |
| **203618** | 1 | O |
| **203619** | Lincoln | B-ORG |
| **203620** | 2 | O |

203621 rows × 2 columns

Fig. 1. Eğitim veri kümesi tokenlar ve bunlara karşılık gelen etiketler verilmiştir.

Sınama veri kümesinde ise 46435 token bulunmaktadır. Şekil 2'de Reuters'ten elde edilen bazı tokenlar ve ona karşılık gelen etiketler görülebilir.

| | Tokens | Labels |
|---|---|---|
| **0** | SOCCER | O |
| **1** | - | O |
| **2** | JAPAN | B-LOC |
| **3** | GET | O |
| **4** | LUCKY | O |
| **...** | ... | ... |
| **46430** | younger | O |
| **46431** | brother | O |
| **46432** | , | O |
| **46433** | Bobby | B-PER |
| **46434** | . | O |

46435 rows × 2 columns

Fig. 2. Sınama veri kümesi tokenlar ve bunlara karşılık gelen etiketler verilmiştir.

Şekil 1 ve 2'de verilen etiketler literatürde IOB etiketleme şeması olarak bilinmektedir. Eğer kelimeler özel bir dizin oluşturuyorsa, bu dizinin ilk kelimesi başlangıç anlamına gelen "B-..." ile etiketlenir. Bu dizilimdeki "B"'yi takip diğer kelimeler ise "I-..." ile etiketlenir. Böylelikle grup halinde bulunan özel isimler de yakalanmış olur. Kelime eğer verilen özel etiketlerin dışında kalıyorsa "O" ile etiketlenir. Özel etiketler ise 4 kategoriden oluştmaktadır: insan (PER), kurum veya kuruluş (ORG), yer (LOC) ve özel isimler (MISC). Bunların BIO kombinasyonuyla elimizde 9 adet etiket oluyor: 'B-ORG', 'O', 'B-MISC', 'B-PER', 'I-PER', 'B-LOC', 'I-ORG', 'I-MISC', 'I-LOC'.

Eğitim veri kümesinde her bir etikete denk gelen toplam kelime sayıları Şekil 3'te verilmiştir.

Aynı şekilde Sınama veri kümesinde her bir etikete denk gelen toplam kelime sayıları Şekil 4'te verilmiştir.

```
O          169578
B-LOC        7140
B-PER        6600
B-ORG        6321
I-PER        4528
I-ORG        3704
B-MISC       3438
I-LOC        1157
I-MISC       1155
```

Fig. 3. Eğitim veri kümesi etiket ve kelime sayıları verilmiştir.

```
O          38177
B-ORG       1715
B-LOC       1646
B-PER       1618
I-PER       1161
I-ORG        882
B-MISC       723
I-LOC        259
I-MISC       254
```

Fig. 4. Sınama veri kümesi etiket ve kelime sayıları verilmiştir.

### C. NER Değerlendirme Metrikleri

Aynı makalede [1] ve SemEval 2013'te NER değerlendirme metrikleri olarak verilen metrikler kesinlik (precision), duyarlılık (recall), F1 skoru, mikro ortalama (micro average), makro ortalama (macro average) ve ağırlıklı F1 (weighted F1) olarak belirlenmiştir. Bu değerlendirme metriğinde öncelikle True Positive, True Negative, False Positive ve False Negative değerleri hesaplanır. True positive etiketin gerçekleşenle tahminlenenin aynı olduğu durumların toplamıdır. True Negative elimizdeki etiketin dışındaki etiketlerin doğru bilinme sayısına denk gelir. False Positive elimizdeki etiketin yanlış tahminlendiği durumdur. False Negative ise elimizdeki etiketin dışındaki etiketlerin yanlış tahminlenme sayısının toplamıdır. Bu durumda kesinlik TP (True Positive) değerinin TP ve FP (False Positive) değerlerinin toplamına bölündüğü zaman elde edilir. Duyarlılık ise TP'nin TP ve FN (False Negative) değerlerinin toplamına bölündüğünde elde edilir. F1 skoru ise TP'nin iki katının TP'nin iki katının FP ve FN ile toplamına bölünmesiyle hesaplanır. Ödevde geliştirilen HMM denetimli öğrenme değerlendirme metrikleri Şekil 5'te verilmiştir.

Bunlara ek olarak mikro ortalama (micro average), TP, FP ve FN'yi etiket sınıfı gözetmeksizin tüm etiketlerin toplamlarını alarak hesaplar. Makro ortalama (macro average) metrikleri her bir etiket için ayrı alır ve bunların ağırlıksız ortalamasını alır. Ağırlıklı F1 (weighted F1) ise makro ortalamaya ek olarak ortalamyı ağırlıklı alır. Bu metriklerin sonuçları ise Şekil 6'da verilmiştir.

```
Aşağıdaki matris için verilen etiketler (yukarıdan aşağıya):
['B-ORG' 'O' 'B-MISC' 'B-PER' 'I-PER' 'B-LOC' 'I-ORG' 'I-MISC' 'I-LOC']
  kesinlik   duyarlılık       f1
[[0.63991976 0.37201166 0.47050147]
 [0.87167652 0.96953663 0.91800595]
 [0.71498771 0.40248963 0.51504425]
 [0.86666667 0.19283066 0.31547017]
 [0.09243697 0.00947459 0.0171875 ]
 [0.69581749 0.33353584 0.45092402]
 [0.25790139 0.23129252 0.24387328]
 [0.2247191  0.23622047 0.2303263 ]
 [0.38016529 0.35521236 0.36726547]]
```

Fig. 5. Geliştirilen HMM denetimli öğrenmenin sınama veri kümesi üzerindeki temel değerlendirme metrikleri ve sonuçları verilmiştir.

```
          kesinlik                duyarlılık                  f1
Mikro ort: 0.8435662754387854       0.8435662754387854        0.8435662754387854
Makro ort: 0.5271434339579648       0.34473381727760416       0.392066490867083
Ağırlıklı F1: 0.8168473138628336       0.8435662754387854        0.8154910628782119
```

Fig. 6. Geliştirilen HMM denetimli öğrenmenin sınama veri kümesi üzerindeki ortalama alınarak elde edilen değerlendirme metrikleri ve sonuçları verilmiştir.

### D. HMM Denetimli Öğrenme: A ve B matrisleri

Bu ödevde yukarıda verilen eğitim ve sınama veri kümeleri kullanılarak HMM denetimli öğrenme uygulanmıştır. HMM için A ve B matrisleri hazırlanmış ve sınama verisindeki cümleler Viterbi algoritması kullanılarak etiketlenmiştir. A matrisi Şekil 7'de verilmiştir.

```
------------------------------------------------------
A matrix: [[3.88387913e-01 4.80840675e-02 1.46015125e-01 2.04393939e-01
  0.00000000e+00 2.21428571e-01 0.00000000e+00 0.00000000e+00
  0.00000000e+00]
 [2.84765069e-03 2.23613912e-02 9.30773706e-03 0.00000000e+00
  0.00000000e+00 1.40056022e-03 1.61987041e-03 6.06060606e-03
  8.64304235e-04]
 [5.86141433e-01 8.19009541e-01 7.05933682e-01 3.40454545e-01
  9.07022968e-01 8.32352941e-01 6.47948164e-01 6.27705628e-01
  8.50475367e-01]
 [9.49216896e-04 1.68712923e-02 9.01687027e-03 1.51515152e-04
  0.00000000e+00 3.78151261e-03 8.09935205e-04 3.46320346e-03
  2.59291271e-03]
 [4.74608448e-04 3.05641062e-02 1.68702734e-02 0.00000000e+00
  0.00000000e+00 1.40056022e-04 8.09935205e-04 2.59740260e-03
  0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 6.49090909e-01
  5.38869258e-02 0.00000000e+00 0.00000000e+00 0.00000000e+00
  0.00000000e+00]
 [1.58202816e-04 3.26280532e-02 2.32693426e-03 4.54545455e-04
  0.00000000e+00 1.12044818e-03 2.69978402e-04 1.73160173e-03
  2.59291271e-03]
 [3.93133998e-01 0.00000000e+00 0.00000000e+00 0.00000000e+00
  0.00000000e+00 0.00000000e+00 3.29103672e-01 0.00000000e+00
  0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 2.49563700e-01 0.00000000e+00
  0.00000000e+00 0.00000000e+00 0.00000000e+00 2.57142857e-01
  0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00
  0.00000000e+00 1.45798319e-01 0.00000000e+00 0.00000000e+00
  1.00259291e-01]]
```

Fig. 7. 10x9 boyutlu A matrisi verilmiştir. İlk satır ¡s¿ yani cümledeki ilk kelimeyi temsil etmektedir.

B matrisi Şekil 8'de verilmiştir.

Viterbi algoritması Şekil 9'da verilmiştir.

Sınama veri kümesindeki bir cümleye uygulanmış ve bu örnekteki gerçekleşen ve tahminlenen etiketler kodda verilmiştir. Bu örnek Şekil 10'da incelenebilir.

```
(9, 23623)
```

```
B
```

```
array([[3.79686758e-03, 0.00000000e+00, 0.00000000e+00, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [0.00000000e+00, 5.89699136e-06, 0.00000000e+00, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [0.00000000e+00, 0.00000000e+00, 2.44328098e-02, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       ...,
       [0.00000000e+00, 0.00000000e+00, 2.69978402e-04, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00]])
```

Fig. 8. 9x23623 boyutlu B matrisi verilmiştir. Satırlar etiketleri, sütunlar ise kelimeleri temsil etmektedir.

```
def viterbi(N, T):
    viterbi_matrix = np.zeros((len(N), len(T))) # B matrix
    backpointer = np.zeros((len(N), len(T)))
    for s_index, s in enumerate(N):
        try:
            item_index = unique_token_list_lower.index(T[0].lower())
        except Exception as exc:
            #raise Exception(f"Item {T[0].lower()} is not found in the token list. Exception: {exc}")
            item_index = -1
        # if the word is found in B matrix
        if item_index > -1:
            viterbi_matrix[s_index][0] = a[0][s_index] * B[s_index][item_index]
        else:
            viterbi_matrix[s_index][0] = 0
        backpointer[s_index][0] = 0
    # initialization ends.
    # recursion step
    for t_index, t in enumerate(T[1:]):
        for s_index, s in enumerate(N):
            # maximum probability viterbi step
            max_prob = 0.0
            for s_prime_index, s_prime in enumerate(N):
                try:
                    item_index = unique_token_list_lower.index(T[t_index + 1].lower())
                except Exception as exc:
                    #raise Exception(f"Item {T[t_index].lower()} is not found in the token list. Exception: {exc}")
                    item_index = -1
                # if the word is found in B matrix
                if item_index > -1:
                    b_s_o_t = B[s_index][item_index]
                else:
                    b_s_o_t = 0
                calculated_probability = viterbi_matrix[s_prime_index][t_index] * a_w_out_start[s_prime_index][s_index]
                if calculated_probability > max_prob:
                    max_prob = calculated_probability
            viterbi_matrix[s_index][t_index + 1] = max_prob # # it is t_index + 1 since enumerate make index start with
            backpointer[s_index][t_index + 1] = max_prob # it is t_index + 1 since enumerate make index start with 0
            # maximum probability viterbi step ends
    # find bestpathprob & best_state_dict
    best_state_list = []
    best_prob_list = []
    for t_index, t in enumerate(T):
        best_state = 0.0
        best_state_list.append("Not_Assigned")
        best_prob_list.append(best_state)
        for s_index, s in enumerate(N):
            if viterbi_matrix[s_index][t_index] > best_state:
                best_state = viterbi_matrix[s_index][t_index]
                best_prob_list[t_index] = viterbi_matrix[s_index][t_index]
                best_state_list[t_index] = s
        if best_state == 0.0:
            best_state_list[t_index] = "O" # token is outside of any span of interest, thus mark it as O
    return best_prob_list, best_state_list
```

Fig. 9. Sınama veri kümesinden bir cümle ve etiket listesi verilen bu algoritma en olası etiket dizilimini bulur.

```
#SOCCER NN I-NP O
#- : O O
#JAPAN NNP I-NP B-LOC
#GET VB I-VP O
#LUCKY NNP I-NP O
#WIN NNP I-NP O
#, , O O
#CHINA NNP I-NP B-LOC
#IN IN I-PP O
#SURPRISE DT I-NP O
#DEFEAT NN I-NP O
#. . O O
np.set_printoptions(suppress=False)
T = ['SOCCER', '-', 'JAPAN', 'GET', 'LUCKY', 'WIN', ',', 'CHINA', 'IN', 'SURPRISE', 'DEFEAT', '.']
best_prob_dict, best_state_dict = viterbi(unique_label_list, T)

print(f"best_prob_dict is: {best_prob_dict}")
print(f"best_state_dict is: {best_state_dict}")
```

```
best_prob_dict is: [3.7428775614501 46e-05, 2.24696445089422e-07, 2.2788973366891825e-09, 2.3239259359147158e-14, 2.24
4769386197595e-19, 1.1383622419506817e-22, 1.9997507851263924e-24, 4.243103557808534 5e-26, 2.7676105788596764e-29, 1.
336670718358212 1e-34, 1.4202557868944555e-38, 5.049889009710358e-40]
best_state_dict is: ['O', 'O', 'B-LOC', 'O', 'O', 'O', 'O', 'B-LOC', 'O', 'O', 'O', 'O']
```

Fig. 10. Sınama veri kümesinden bir cümle ve Viterbi ile elde eliden sonuç verilmiştir..

*E. 1.dereceden ve 2. dereceden iki farklı HMM modeli geliştirerek skorlarını karşılaştırınız, raporunuzda açıklayıp yorumlayınız.*

2. dereceden HMM modeli A matrisinin 3 boyutlu hale getirilerek ve Viterbi algoritmasının "recursion step"'inde bu 3 boyutlu A matrisinin kullanılmasıyla elde edilmiştir. A matrisi iki önceki kelimeye bakılarak hesaplanmıştır. Viterbi algoritmasında ise viterbi matrisinin 2. sütunundan sonra viterbi matrisinin iki sütun öncesine bakılmıştır. İki sütun öncesindeki en yüksek olasılıklı etiket baz alınmış ve bu etiket A matrisindeki iki kelime öncesine referans verilmiştir. Böylelikle viterbi matrisinin yeni yolları hesaplanırken iki kelime öncesi de hesaba dahil edilmiştir. Fakat sadece en yüksek olasılıklı etiket hesaba dahil olduğundan dolayı bazı etiketlerde başarı oranları oldukça düşmüş, genel olarak da 1. dereceden HMM modeline göre kötü sonuçlar üretmiştir. Yapılacak en doğru 2. dereceden HMM modelinde tüm etiketler dahil edilmeli ve iki kelime öncesinden gelecek şekilde en yüksek olan yol çizilmelidir.

1. dereceden HMM modeli sınama verisi üzerindeki sonuçları Şekil 5 ve Şekil 6'da verilmiştir. 2. dereceden HMM modeli sınama verisi üzerindeki sonuçları ise Şekil 11'de verilmiştir.

```
Aşağıdaki matris için verilen etiketler (yukarıdan aşağıya):
['O' 'B-LOC' 'B-PER' 'I-PER' 'I-LOC' 'B-MISC' 'I-MISC' 'B-ORG' 'I-ORG']
  kesinlik   duyarlılık       f1
[[0.02547771 0.00233236 0.0042735 ]
 [0.82886013 0.88739293 0.8571284 ]
 [0.02373418 0.02074689 0.02214022]
 [0.04705882 0.00494438 0.00894855]
 [0.         0.         0.        ]
 [0.0079001  0.01883354 0.01113106]
 [0.2183908  0.02154195 0.03921569]
 [0.02281369 0.04724409 0.03076923]
 [0.06896552 0.01544402 0.02523659]]
                kesinlik                duyarlılık                      f1
Mikro ort: 0.7315817809841714       0.7315817809841714       0.7315817809841714
Makro ort: 0.1381334386058277       0.11316446156257538      0.11098258229262771
Ağırlıklı F1: 0.05580143174349599       0.036669188405204944       0.03689526229360453
```

Fig. 11. 2. derece HMM modelinin sınama verisi üzerindeki sonuçları verilmiştir.

## REFERENCES


[1] Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, pages 142–147.