T.C.

BALIKESİR ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Yapay Sinir Ağları ile Gazete Haberleri Kategorizasyonu ve Duygu Analizi

202013709019 - Ahmet Eren ÖZTÜRK

BMM4101 YAPAY ZEKA TEKNİKLERİ VİZE ÖDEVİ

Danışman: Dr. Öğr. Üyesi Kadriye ERGÜN

BALIKESİR, Aralık – 2023

İÇİNDEKİLER

PROJE TANITIMI – Duygu Analizi

Dijital iletişim çağında, halkın tepkilerini anlamak için duygu analizi çok önemlidir. Bu proje, haber sitesi yorumlarındaki duyguları analiz etmek için Kaggle'ın "e-ticaret_urun_yorumlari" dataset üzerinde eğitilmiş bir yapay sinir ağı (ANN) modelidir. Başlangıçta e-ticaret için tasarlanan dataset içerisinde; durumlar olumsuz için 0, olumlu için 1 ve nötr için 2 olarak etiketlendi.

Yapay Sinir Ağı, yorum metnindeki kalıpları ayırt etmek için çeşitli katmanları kullanıyor ve haber makalelerindeki duyguları kategorize etmek için otomatik bir çözüm sunuyor. Bu girişim, paydaşlara halkın tepkilerini ölçmek için güçlü bir araç sunmayı ve çevrimiçi söylemin dinamik bağlamında daha geniş bir duygu analizi alanına katkıda bulunmayı amaçlıyor. Aşağıdaki bölümler; model mimarisini, veri kümesi ön işlemesini ve eğitim sürecini özetliyor.

DATASET

Duyarlılık analizi modelini eğitmek ve değerlendirmek için kullanılan dataset "e-ticaret_urun_yorumlari "; Kaggle'dan alınmıştır. Bu dataset iki ana bileşenden oluşur:

Metin Verileri: Dataset, kullanıcılardan gelen yorumlar veya incelemeler biçimindeki metinsel bilgileri içerir. Bu metinler, duygu analizi modeline input görevi görür.

Durum Etiketleri: Dataset, her yorumun duyarlılık kategorisini belirten karşılık gelen etiketleri içerir. Duygular üç sınıfa ayrılmıştır: Olumsuz için 0, olumlu için 1 ve nötr için 2. Bu etiketler, eğitim aşamasında temel gerçek görevi görür ve modelin, girdi metni ile ilgili duygular arasındaki ilişkileri öğrenmesine olanak tanır.

	A	B
1	Metin	Durum
2	evet anlatıldığı gibi	1
3	Daha öncede almıştım bu cihazdan ense ve sakal tüketmek için on numara sıfıra yakın alıyor	1
4	Ürün gayet başarılı sakal kesmede başlık sayısı biraz daha fazla olabilirdi.Hem 0 a yakın aliyor. hem de kirli sakal için 3 numara başlık ideal.	1
5	Daha öncede aynısını almıştım çok güzel ve kaliteli bir ürün.	1
6	Erkek kuaförüyüm ense ve sıfır sakal traşı için uygun bir ürün	1
7	ürün gerçekten çok güzel	1
8	Ürün beklediğimden güzel çıktı gayet kullanışlı tarklarda bir iki numara saha olsa daha iyidi ama her şey harika	1
9	güzel makina tavsiye ederim	1
10	tavsiye edebileceğim çok güzel bir makina	1
11	ürün geldiğinde şarjı vardı. ilk lullanım öncesi 10 saat kadar şarjda kaldı. yaklaşık 30 dk kadar kesintisiz kullandım vebhala şarjı iyi seviyedeydi. 30 dk kullanım sürecinde ısınr	1
12	Ürün sadece aldığım gün güzel bir şekilde kesti. sonrasında saatlerce uğraşıp sadece sakalımı toplayabildim. Allah kahretsin başka bı makine alacam ama bu sefer powertec ı	0
13	denedim ürün çok güzel herkese tavsiye ederim	1
14	ürün harika tam istediğim gibi ayrıca mağazaya da teşekkürler hızlı oldukları için	1
15	Ürün hala elime geçmedi	0
16	Bir kere kullandım memnun kaldım	1
17	35 TL ye berber makinesi powertec almıştım. 300 liraya aldığım dünya markası yerine onu kullandım. powertec yine şaşırtmadı harika bir ürün. genelde sakal kesiyorum çok	1
18	Ürün çalışıyor şimdilik	2
19	tam istedigim gbi orjinal	1
20	Ürün, kullanım açısından kolay. Yanında 1mm 2mm 3mm başlıkları ile geliyor. Sakal traşım ve şekil verme konularında bir sıkıntısını görmedim.	1
21	hiç dusunmeden alın derim jilet gibi kesiyor	1

METODOLOJİ

1. Kütüphaneleri İçe Aktarma ve Veri Ön İşleme

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from keras.models import Sequential
from keras.layers import Dense, Embedding, Flatten
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.utils import to_categorical
```

```
# CSV dosyasini dogru sinirlayici ile yukleme
df = pd.read_csv('/kaggle/input/erenum/e-ticaret_urun_yorumlari.csv', delimiter=';')

# DataFrame'in ilk birkac satirini goruntuleme
print(df.head())

# Veri On Isleme
X = df['Metin'].values
y = df['Durum'].values

# Hedef degiskeni kodlama
le = LabelEncoder()
y = le.fit_transform(y)
y = to_categorical(y)

# Verileri egitim ve test setlerine ayirma
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

1. **pandas** veri manipülasyonu ve analizi için kullanılır.
2. Veri setini eğitim ve test setlerine bölmek için **sklearn**'den "*train_test_split*" kullanılır.
3. Hedef değişkeni kodlamak için **sklearn**'den "*LabelEncoder"* kullanılır.
4. Sinir ağı oluşturmak için **keras.models**'den "*Sequential, Dense, Embedding, Flatten"* ve diğer **Keras** modülleri içe aktarılır.
5. E-ticaret ürün yorumlarını ve duygu analizini içeren bir CSV dosyasını pandas DataFrame'e *('df')* yükler.
6. Verilerin yapısını incelemek için DataFrame'in ilk birkaç satırını yazdırır.
7. 'Metin' (text) sütununu özellikler (X) olarak ve 'Durum' (label) sütununu hedef değişken (y) olarak çıkarır.

8. Kategorik label'leri sayısal formata dönüştürmek için '*LabelEncoder*' kullanır.
9. *'to_categorical'*, sayısal etiketleri **one-hot** kodlanmış vektörlere dönüştürmek için kullanılır.

2. Tokenleştirme

```
# Tokenlestirme
max_words = 1000
tokenizer = Tokenizer(num_words=max_words, oov_token="<OOV>")
tokenizer.fit_on_texts(X_train)

X_train_seq = tokenizer.texts_to_sequences(X_train)
X_test_seq = tokenizer.texts_to_sequences(X_test)

X_train_padded = pad_sequences(X_train_seq)
X_test_padded = pad_sequences(X_test_seq, maxlen=X_train_padded.shape[1])
```

1. *"max_words"* 1000 olarak ayarlanmıştır ve bu, tokenleştirme sırasında dikkate alınacak benzersiz kelimelerin sayısını sınırlamaktadır.
2. *"Tokenizer"*, metin verilerini tokenize etmek için kullanılan Keras'tan bir nesnedir.
3. *"num_words"*, kelime dağarcığını en sık kullanılan 1000 kelimeyle sınırlandıracak şekilde ayarlanır ve *"oov_token",* kelime dağarcığında olmayan kelimeleri temsil etmek için "<OOV>" (out of vocabulary) olarak ayarlanır.
4. Kelime sıklığına dayalı kelime dağarcığı oluşturmak için eğitim verilerine *(X_train) fit_on_texts* metodu uygulanır.
5. *"texts_to_sequences"*, metin verilerini, gömme işlemi sırasında belirteç tarafından oluşturulan kelime dağarcığına dayalı olarak karşılık gelen kelime endekslerinin dizilerine dönüştürmek için kullanılır.
6. Bu adım esasen incelemelerdeki her kelimeyi sözlükteki indeksiyle temsil eder.
7. *"pad_sequences"* tüm dizilerin aynı uzunluğa sahip olmasını sağlamak için uygulanır. Verilerin bir sinir ağına beslenmesi için bu gereklidir.

8. *"X_train_padded* ve *X_test_padded*" sırasıyla eğitim ve test dizilerinin yastıklı versiyonlarıdır.
9. *"maxlen"* parametresi, her iki sette de tutarlı dizi uzunluğu sağlamak için eğitim verilerindeki en uzun dizinin uzunluğuna ayarlanır.

Özetle, bu kod metin verilerini tokenleştirir, metin dizilerini kelime dağarcığına dayalı sayısal dizilere dönüştürür ve dizileri tutarlı uzunluklara sahip olacak şekilde doldurur.

3. Yapay Sinir Ağı Mimarisi

```
# Yapay Sinir Agi Mimarisi Olusturma
embedding_dim = 100
model = Sequential()
model.add(Embedding(input_dim=max_words, output_dim=embedding_dim, input_length=X_train_padded.shape[1]))
model.add(Flatten())
model.add(Dense(100, activation='relu'))
model.add(Dense(3, activation='softmax'))

# Modeli Derleme
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

# Modeli Egitme
model.fit(X_train_padded, y_train, epochs=10, batch_size=32, validation_data=(X_test_padded, y_test))

# Modeli degerlendirme
loss, accuracy = model.evaluate(X_test_padded, y_test)
print(f'Accuracy: {accuracy * 100:.2f}%')
```

1. *"embedding_dim"* 100'e ayarlanmıştır; bu, sözcük dağarcığı içindeki kelimelere yönelik gömmelerin boyutluluğunu temsil eder.
2. *"Sequential*", doğrusal bir katman yığını olan **Keras** sıralı modelidir.
3. İlk katman, tam sayı endekslerini sabit boyutlu yoğun vektörlere dönüştürmekten sorumlu olan **Embedding** katmanıdır. Genellikle bir DDİ(Doğal Dil İşleme) modelinde ilk katman olarak kullanılır.
4. **Embedding** katmanından gelen 2D tensör çıktısını 1D tensöre düzleştirmek için **Flatten** katmanı eklenir.
5. Tamamen bağlı katmanları temsil eden iki Dense katman bunu takip eder. Birincisinde **ReLU** aktivasyonuna sahip 100 ünite, ikincisinde ise çoklu

sınıf sınıflandırmaya uygun **softmax** aktivasyonuna sahip 3 ünite bulunmaktadır.

6. **'compile'** Eğitim için modeli yapılandırmak için kullanılır.
7. Kayıp fonksiyonu, çok sınıflı sınıflandırmaya uygun olarak *'categorical_crossentropy'* olarak ayarlanmıştır.
8. Optimizer, popüler bir optimizasyon algoritması olan '*adam*'a ayarlıdır.
9. Eğitim sırasında izlenecek ölçüm '*accuracy*' olarak ayarlanmıştır.
10. Modeli eğitmek için "*fit*" kullanılır.
11. "*X_train_padded*" ve "*y_train*" eğitime yönelik input özellikleri ve label'lerdir.
12. Model, ***batch*** büyüklüğü 32 olan 10 ***epoch*** için eğitilmiştir.
13. *'validation_data'*, eğitim sırasında doğrulama seti üzerindeki modeli değerlendirmek için sağlanır.
14. evaluate, test setindeki modeli değerlendirmek için kullanılır.
15. Test seti ***(X_test_padded ve y_test)***, kaybı ve doğruluğu hesaplamak için kullanılır.
16. Daha sonra ***accuracy*** yazdırılır.

Özetle, bu kod bir sinir ağı mimarisini tanımlar, onu derler, sağlanan veriler üzerinde eğitir ve performansını bir test kümesinde değerlendirir. Embedding katmanı, Flatten ve Dense katmanlarla metin sınıflandırması için tasarlanmış bir yapay sinir ağıdır.

4. Modeli Kaydetme ve Yükleme

```
# Modeli Kaydetme
model.save("/kaggle/working/my_model")
# Modeli Yukleme
from keras.models import load_model
loaded_model = load_model("/kaggle/working/my_model")
```

1. ***"save"*** Keras'ta mimari, optimize edici ve öğrenilen ağırlıklar dahil tüm modeli bir dosyaya kaydetmek için kullanılan bir yöntemdir.
2. ***"load_model"*** bir dosyadan kayıtlı bir modeli yüklemek için kullanılan Keras'ın bir fonksiyonudur.

5. Otomatik Veri Çekme

```python
import pandas as pd
import requests
from bs4 import BeautifulSoup
from keras.preprocessing.sequence import pad_sequences
from unidecode import unidecode
import re

# Metni temizleme ve on isleme fonksiyonu
def preprocess_text(text):
    # Remove HTML tags
    soup = BeautifulSoup(text, 'html.parser')
    text = soup.get_text(separator=' ')
    # Remove URLs
    text = re.sub(r'http\S+', '', text)
    # Convert to lowercase
    text = text.lower()
    # Convert Turkish characters to ASCII
    text = unidecode(text)
    # Remove non-alphabetic characters
    text = re.sub(r'[^a-zA-Z\s]', '', text)
    # Remove extra whitespaces
    text = ' '.join(text.split())
    return text
```

1. Metin verilerinin temizlenmesi ve ön işlenmesi için "*preprocess_text*" fonksiyonunu tanımlar.
2. HTML etiketlerini kaldırmak için **BeautifulSoup**'u, URL'leri kaldırmak için **regular expressions**'ı kullanır, metni küçük harfe dönüştürür, Türkçe karakterleri ASCII'ye dönüştürür, alfabetik olmayan karakterleri kaldırır ve fazladan boşlukları kaldırır.

```
# URL'yi belirleyin
url = "https://www.haber7.com/yorum/oku/3373867"
# Web sayfasini cekme
response = requests.get(url)
soup = BeautifulSoup(response.text, 'html.parser')
# Yorumlari iceren HTML etiketlerini bulup yerlestirme
comments = soup.find_all('div', class_='content')
# Yorumlari bir liste icinde saklayip on islem uygulama
comments_list = [preprocess_text(comment.text.strip()) for comment in comments]
```

3. Bir URL belirtir ve "*requests*" kullanarak web sayfası içeriğini alır.
4. HTML içeriğini ayrıştırmak ve belirtilen HTML etiketlerinden yorumları çıkarmak için **BeautifulSoup**'u kullanır.
5. Her yorumu temizlemek ve ön işlemek için "*preprocess_text"* fonksiyonunu uygular.

```
# Tokenizer'i kullanarak yorumlari sayilara donusturme ve padding uygulama
X_new_seq = tokenizer.texts_to_sequences(comments_list)
X_new_padded = pad_sequences(X_new_seq, maxlen=X_train_padded.shape[1])
# Modeli kullanarak tahmin yapar
predictions = loaded_model.predict(X_new_padded)
# Tahminleri label'lere cevirme
predicted_labels = le.inverse_transform(predictions.argmax(axis=1))
# Tahmin sonuclarini iceren DataFrame'i olusturma
result_df = pd.DataFrame({'Metin': comments_list, 'Durum': predicted_labels})
# DataFrame'i .csv dosyasina kaydetme
result_df.to_csv('predicted_comments_15.csv', index=False, sep=';')
```

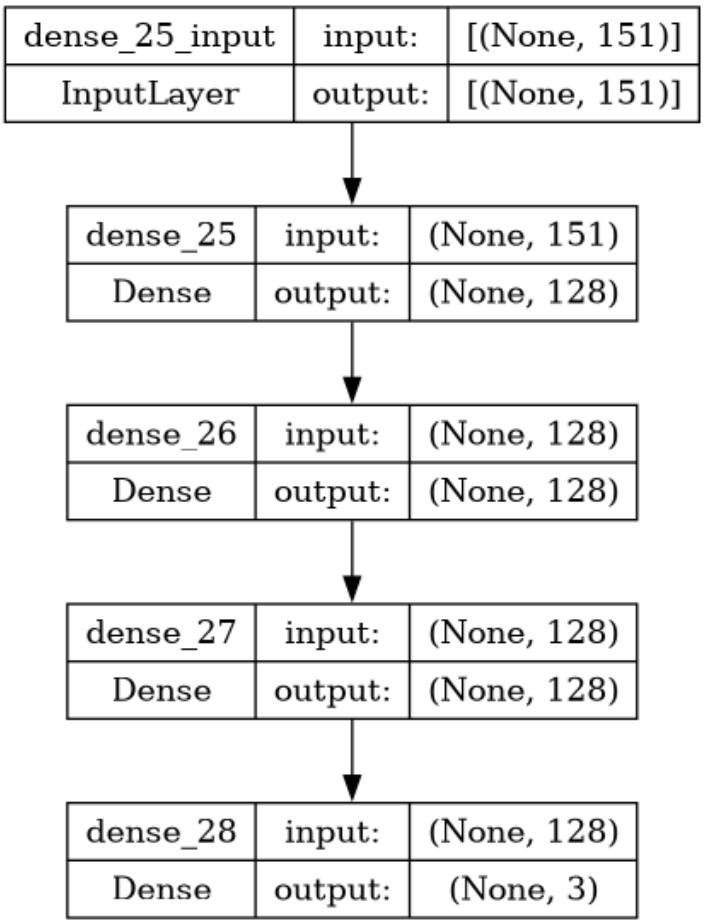
6. Önceden işlenmiş yorumları indeks dizilerine dönüştürmek için önceden tanımlanmış **tokenleştiriciyi** kullanır.
7. *"pad_sequences"* kullanarak tutarlı dizi uzunluğu sağlamak için **padding** uygular.
8. Önceden işlenmiş ve doldurulmuş yorumlara ilişkin tahminlerde bulunmak için yüklü sinir ağı modelini ***(loaded_model)*** kullanır.
9. Tahmin edilen sayısal label'leri orijinal kategorik label'lara geri dönüştürmek için ***inverse transformation*** kullanır.
10. Önceden işlenmiş yorumları ve tahmin edilen label'leri içeren bir ***DataFrame*** *'(result_df)'* oluşturur.

11. DataFrame'i belirtilen CSV dosyasına kaydeder. Verileri ayırmak için ";" kullanır.(farklı sütunlarda gösterebilmek için.)

12. Sonuç:

	A	B
1	Metin	Durum
2	tenceredibinkaraseninkibendenkara	0
3	dil yarasi	0
4	kocaeli mhp istemiyordu bu kadar acik isabet olmus	1
5	devlet bey istifasini istedi sancakli da emri yerine getirdikendi ayrilmadi yanibence yanlis yaptibu kadar cabuk adam harcanm	0
6	samsun ilkadim da mhp ye beledeye baskanligi adayi kesinlikle verilmelidir yoksa buyuksehir bile kaybedelibilr	0
7	radyo programindan istek yapmiyorsun bu tur istekler mhp de gecersiz ibret al	0
8	ser odaklarin cumhur ittifaki na yonelik asli astari olmayan haberleri yipratma amaclidirherkes kendine duseni yapmali cumh	0
9	turk siyaseti derebeylik gibi parti ile meclise giren partiden istifa ettiginde vekilligi de dusmelidir siyasete devam edecek ise bi	0
10	gule gule	0
11	saffet bey cumhur ittifakina yanlis yapacak birisi degildir	0
12	sizin acilen millet vekilliginden istifa etmeniz gerekiyor diye dusunuyorum	0
13	madem devlet baskana bu kadar baglisin saygin var neden elini opup ozur dilemedin o kadar milletvekilligi verdi sana devlet	0
14	futbolcuysan git yorumcu ol antrenor ol ne isin var siyasetteturkiyede niye kimse isini yapmiyor	0
15	yerin cumhur ittifaki olmali eger meral e gidersen adam degilsin derim	1
16	mhp sadece bahceli fikir beyan edernedense gazzeye gitmeyi erteledimillet desdek verdi devlet izin vermisti hele ferdi tayfur	0
17	sayin sancakli hem ulke hem parti ve hem de birey olarak ince bir cizgi uzerinde yurudugumuz cok hassas donemlerden geciy	0

Yapay Sinir Ağlarında Oluşturulan Ağın Şekli



AĞ TOPOLOJİSİ VE VARSAYIMLAR

1. Input Katmanı:
   - Tür: Embedding Katman
   - Parametreler:
     - ***input_dim***: Maksimum kelime sayısı 1000 olarak ayarlandı.
     - ***output_dim***: Kelime gömmelerinin boyutsallığı 100 olarak ayarlandı.
     - ***input_length***: Giriş dizilerinin uzunluğu, dolgulu dizilerin maksimum uzunluğuna ayarlandı.
2. Gizli Katmanlar:
   - Katman 1:
     - Tür: Flatten Katman
     - Amaç: Embedding katmanındaki 2D tensör çıktısını 1D tensöre düzleştirir.
   - Katman 2:
     - Tür: Dense Katman (Tam Bağlantılı)
     - Birimler: 100
     - Aktivasyon Fonksiyonu: Doğrultulmuş Doğrusal Birim (ReLU)
3. Çıkış Katmanı:
   - Tür: Dense Katman (Tam Bağlantılı)
   - Birimler: 3 (Duygu analizi için 3 sınıf varsayılmaktadır)
   - Aktivasyon Fonksiyonu: Softmax (çok sınıflı sınıflandırma için)

Veri Yükleme

- E-ticaret ürün incelemelerini içeren bir CSV dosyası pandas *(pd.read_csv)* kullanılarak sağlanır ve yüklenir.
- Varsayım: CSV dosyası, 'Metin' sütunu altındaki metin verileri ve 'Durum' sütunu altındaki etiketlerle doğru biçimlendirilmiştir.

Veri Ön İşleme

- Giriş metni verileri ('Metin') çıkarılır ve özellikler (X) olarak kullanılır ve etiketler ('Durum') ***LabelEncoder*** kullanılarak ve ***to_categorical*** ile one-hot kodlama kullanılarak kodlanmıştır.
- Varsayım: Giriş verileri model eğitimi için uygun şekilde önceden işlenmiş ve kodlanmıştır.

Train-Test Split

- Veriler, %20 test boyutuyla ***train_test_split*** kullanılarak eğitim ve test kümelerine bölünür.
- Varsayım: Veri kümesi temsilidir ve rastgele tohum ***(random_state=42)*** bölünmenin tekrarlanabilirliğini sağlar.

Tokenizasyon ve Padding

- Tokenizasyon, maksimum kelime büyüklüğü 1000 olan Tokenizer ve kelime dışı token <OOV> kullanılarak gerçekleştirilir. Tutarlı uzunluk sağlamak için diziler ***pad_sequences*** kullanılarak doldurulur.
- Varsayım: Tokenizer, eğitim verileri üzerinde eğitilir ve diziler, model girişi için uygun şekilde doldurulur.

Yapay Sinir Ağı Mimarisi

- Bir yapay sinir ağı, gömme katmanı, flatten katmanı ve iki dense katmandan oluşur. Gömme katmanı, sözlükteki kelimeler için 100 boyutlu gömmelere sahiptir.
- Varsayım: Ağ mimarisi duyarlılık analizine uygundur ve çıktı katmanındaki birim sayısı sınıf sayısına karşılık gelir.

Model Derlemesi

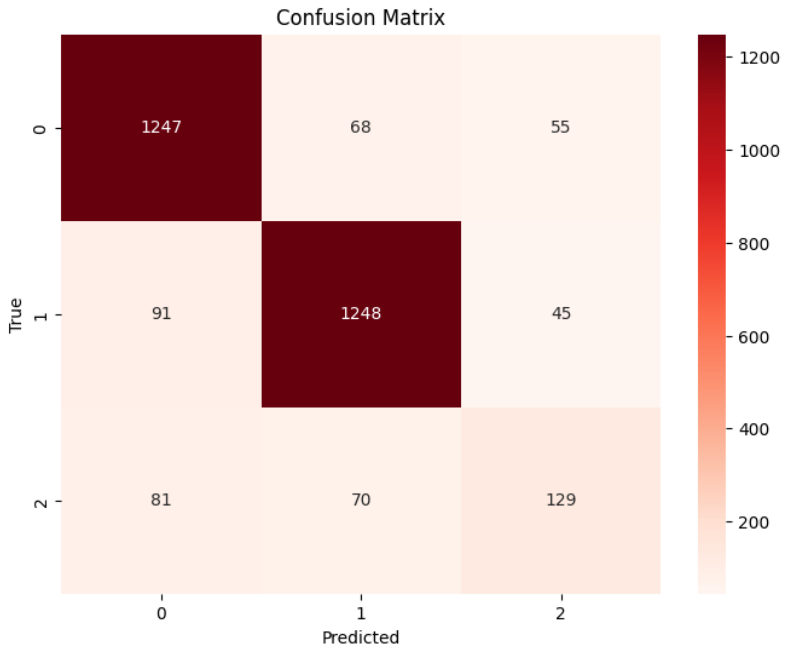
- Model, değerlendirme ölçüsü olarak *categorical crossentropy loss*, Adam optimizer ve accuracy derlenmiştir.
- Varsayım: Seçilen kayıp ve optimize edici göreve uygundur.

Model Eğitimi

- Model, eğitim seti kullanılarak 32 batch büyüklüğünde 10 epoch boyunca eğitilir ve test seti üzerinde doğrulama gerçekleştirilir.
- Varsayım: Model yeterli sayıda epoch'ta eğitilmiştir ve batch büyüklüğü mevcut kaynaklar için makuldür.

Optimum Ağ Tasarımı

Modelin ***evaluate*** kısmı Yapay sinir ağının 10 epoch boyunca eğitim ve doğrulama performansını gösterir. Model %86,45'lik en yüksek doğrulama doğruluğuna ulaşır. Eğitim doğruluğu istikrarlı bir şekilde gelişerek son epoch'tax %98,55'e ulaştı; bu da modelin eğitim verilerini iyi öğrenmiş olabileceğini gösteriyor. Bununla birlikte, validation accuracy düzleşir ve hatta birkaç dönem sonra hafifçe azalır, bu da potansiyel overfitting olduğunu gösterir. Ağı geliştirmek için dropout, model karmaşıklığını ayarlama veya eğitim verilerini artırma gibi düzenleme teknikleri denendi. Ek olarak, performansı artırmak için kesinlik, geri çağırma gibi diğer ölçümlerin izlenmesi veya sıralı veriler için tekrarlayan sinir ağları (RNN'ler) gibi daha gelişmiş mimarilerin kullanılması da araştırılabilir (Geleceğe yönelik model geliştirmeleri için).



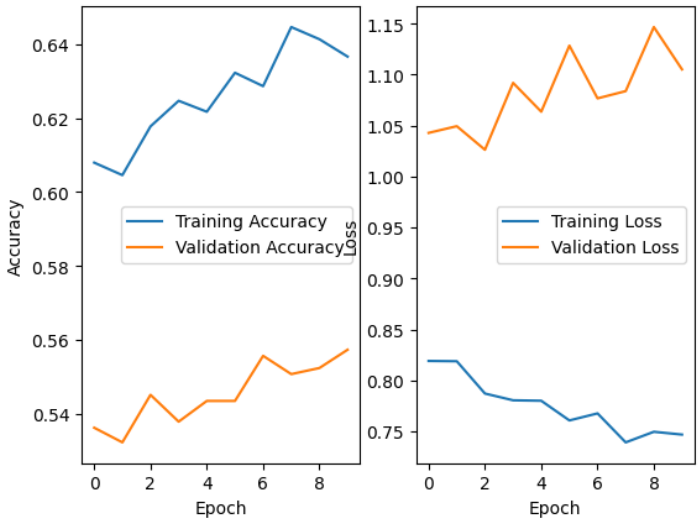
Karşılaştırma

Temel Farklılıklar ve Optimizasyonlar

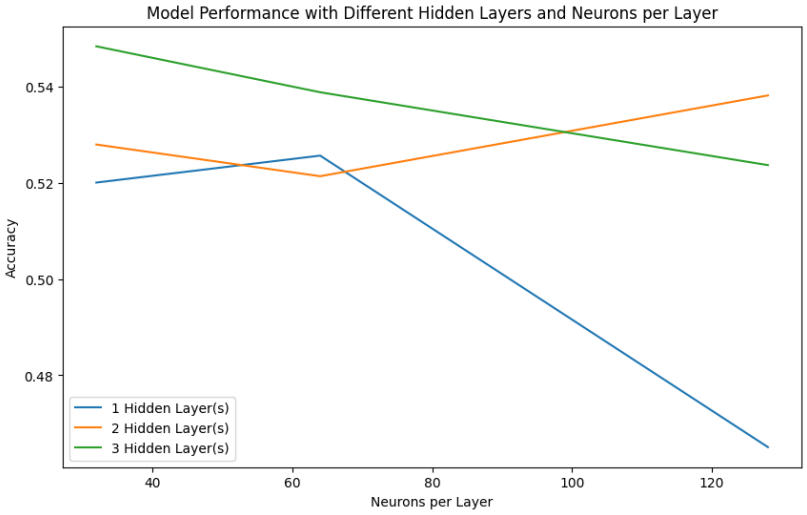
- Embedding Dimension:
  - İlk Model: embedding_dim = 200
  - Optimal Model: embedding_dim = 100
  - En uygun model, modelin karmaşıklığını azaltabilen ve potansiyel olarak eğitim hızını artırabilen daha düşük bir gömme boyutu kullanır.
- Dense Layers:
  - Başlangıç Modeli: Sırasıyla 1024, 300 ve 3 birimden oluşan üç dense katman.
  - Optimal Model: Sırasıyla 100 birim ve 3 birimden oluşan iki dense katman.
  - Optimum model, daha az parametreye sahip daha basit bir mimariye sahiptir ve bu, özellikle sınırlı verilerle uğraşırken overfitting önlenmesine yardımcı olabilir.
- Giriş Uzunluğu ve Padding:
  - Başlangıç Modeli: input_length=X_train_padded.shape[3]
  - Optimal Model: input_length=X_train_padded.shape[1]
  - Optimal model, gömme katmanının girdi uzunluğu olarak dolgulu dizilerin uzunluğunu doğru bir şekilde kullanır.

Özetle, optimal model, daha düşük gömme boyutuna ve daha az yoğun katmana sahip daha basit bir mimariye sahiptir; bu, özellikle sınırlı verilerle uğraşırken eğitim verimliliği ve genelleme açısından faydalı olabilir.

Model Eğitimi İlerlemesini ve Değerlendirme Sonuçlarını Görselleştirme



Yapay sinir ağları modelinin eğitim sürecinin anlaşılır görselleştirmelerini oluşturarak doğruluğun evrimini ve epoch'lar boyunca kayıpları sergiler. Çift panelli grafik, hem eğitim hem de doğrulama performansının kapsamlı bir görünümünü sağlar. Ayrıca komut dosyası, test veri kümesindeki nihai doğruluğu bildiren bir model değerlendirmesiyle sona erer. Bu görselleştirmeler ve ölçümler, modelin eğitim yolculuğunun ve nihai etkililiğinin kısa bir özetini sunar.



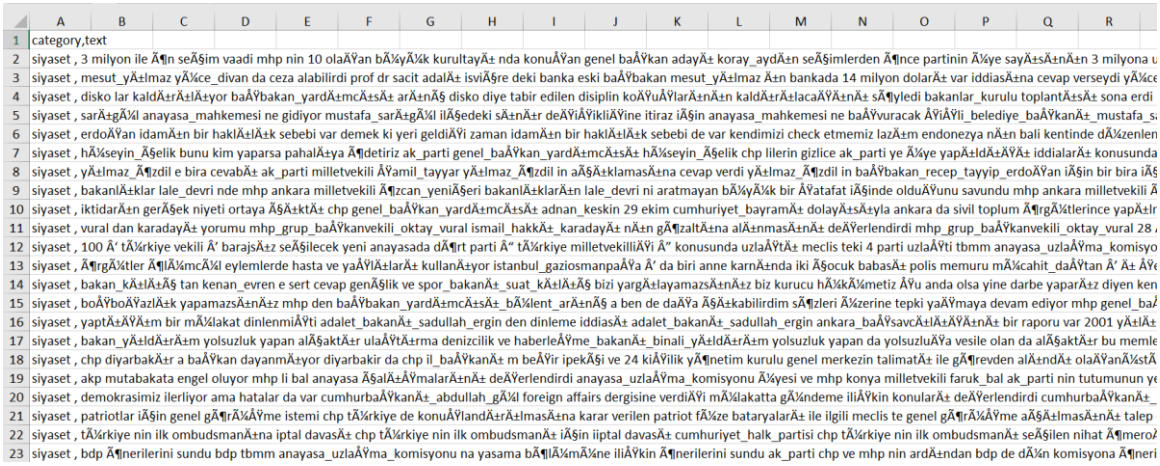
"Grafik, değişen gizli katmanlara ve katman başına nöronlara sahip sınıflandırma modellerinin doğruluğunu göstermektedir. Her satır, farklı sayıda gizli katmana karşılık gelir ve katman başına nöronlar değiştikçe doğruluk üzerindeki etkiyi gösterir."

PROJE TANITIMI – Haber Kategorizasyonu

Bu çalışmanın amacı Yapay Sinir Ağlarını (YSA) kullanarak Haber Kategorizasyonu modeli geliştirmektir. Proje, ilgili kategorilere sahip haberleri içeren bir dataset kullanıyor. Amaç, haber içeriklerini önceden tanımlanmış kategorilere doğru bir şekilde sınıflandırabilen bir yapay sinir ağı modeli geliştirmektir.

DATASET

Bu çalışmada kullanılan dataset, metin içeriğine ve ilgili kategorilere ilişkin sütunları içeren bir CSV dosyasından ('7allV03.csv') yüklenmiştir. Dataset önceden işlenir ve metin verileri, model değerlendirmesi için eğitim ve test kümelerine bölünür.



	A
1	category,text
2	siyaset , 3 milyon ile Ã¶n seÃ§im vaadi mhp nin 10 olaÄŸan bÃ¼yÃ¼k kurultayÄ± nda konuÅŸan genel baÅŸkan adayÄ± koray_aydÄ±n seÃ§imlerden Ã¶nce partinin Ã¼ye sayÄ±sÄ±nÄ±n 3 milyona u
3	siyaset , mesut_yÄ±lmaz yÃ¼ce_divan da ceza alabilirdi prof dr sacit adalÄ± isviÃ§re deki banka eski baÅŸbakan mesut_yÄ±lmaz Ä±n bankada 14 milyon dolarÄ± var iddiasÄ±na cevap verseydi yÃ¼ce
4	siyaset , disko lar kaldÄ±rÄ±lÄ±yor baÅŸbakan_yardÄ±mcÄ±sÄ± arÄ±nÃ§ disko diye tabir edilen disiplin koÅŸuÅŸlarÄ±nÄ±n kaldÄ±rÄ±lacaÄŸÄ±nÄ± sÃ¶yledi bakanlar_kurulu toplantÄ±sÄ± sona erdi
5	siyaset , sarÄ±gÃ¼l anayasa_mahkemesi ne gidiyor mustafa_sarÄ±gÃ¼l iÃ§edeki sÄ±nÄ±r deÄŸiÅŸikliÄŸine itiraz iÃ§in anayasa_mahkemesi ne baÅŸvuracak ÅŸiÅŸli_belediye_baÅŸkanÄ±_mustafa_s
6	siyaset , erdoÄŸan idamÄ±n bir haklÄ±lÄ±k sebebi var demek ki yeri geldiÄŸi zaman idamÄ±n bir haklÄ±lÄ±k sebebi de var kendimizi check etmemiz lazÄ±m endonezya nÄ±n bali kentinde dÃ¼zenlen
7	siyaset , hÃ¼seyin_Ã§elik bunu kim yaparsa pahalÄ±ya Ã¶detiriz ak_parti genel_baÅŸkan_yardÄ±mcÄ±sÄ± hÃ¼seyin_Ã§elik chp lilerin gizlice ak_parti ye Ã¼ye yapÄ±ldÄ±ÄŸÄ± iddialarÄ± konusunda
8	siyaset , yÄ±lmaz_Ã¶zdil e bira cevabÄ± ak_parti milletvekili ÅŸamil_tayyar yÄ±lmaz_Ã¶zdil in aÃ§Ä±klamasÄ±na cevap verdi yÄ±lmaz_Ã¶zdil in baÅŸbakan_recep_tayyip_erdoÄŸan iÃ§in bir bira iÃ§
9	siyaset , bakanlÄ±klar lale_devri nde mhp ankara milletvekili Ã¶zcan_yeniÃ§eri bakanlÄ±klarÄ±n lale_devri ni aratmayan bÃ¼yÃ¼k bir ÅŸatafat iÃ§inde olduÄŸunu savundu mhp ankara milletvekili Ã
10	siyaset , iktidarÄ±n gerÃ§ek niyeti ortaya Ã§Ä±ktÄ± chp genel_baÅŸkan_yardÄ±mcÄ±sÄ± adnan_keskin 29 ekim cumhuriyet_bayramÄ± dolayÄ±sÄ±yla ankara da sivil toplum Ã¶rgÃ¼tlerince yapÄ±lr
11	siyaset , vural dan karadayÄ± yorumu mhp_grup_baÅŸkanvekili_oktay_vural ismail_hakkÄ±_karadayÄ± nÄ±n gÃ¶zaltÄ±na alÄ±nmasÄ±nÄ± deÄŸerlendirdi mhp_grup_baÅŸkanvekili_oktay_vural 28
12	siyaset , 100 Â' tÃ¼rkiye vekili Â' barajsÄ±z seÃ§ilecek yeni anayasada dÃ¶rt parti Â" tÃ¼rkiye milletvekilliÄŸi Â" konusunda uzlaÅŸtÄ± meclis teki 4 parti uzlaÅŸti tbmm anayasa_uzlaÅŸma_komisyo
13	siyaset , Ã¶rgÃ¼tler Ã¶lÃ¼mcÃ¼l eylemlerde hasta ve yaÅŸlÄ±larÄ± kullanÄ±yor istanbul_gaziosmanpaÅŸa Â' da biri anne karnÄ±nda iki Ã§ocuk babasÄ± polis memuru mÃ¼cahit_daÅŸtan Â' Ä± ÅŸ
14	siyaset , bakan_kÄ±lÄ±Ã§ tan kenan_evren e sert cevap genÃ§lik ve spor_bakanÄ±_suat_kÄ±lÄ±Ã§ bizi yargÄ±layamazsÄ±nÄ±z biz kurucu hÃ¼kÃ¼metiz ÅŸu anda olsa yine darbe yaparÄ±z diyen ken
15	siyaset , boÅŸboÄŸazlÄ±k yapamazsÄ±nÄ±z mhp den baÅŸbakan_yardÄ±mcÄ±sÄ±_bÃ¼lent_arÄ±nÃ§ a ben de daÄŸa Ã§Ä±kabilirdim sÃ¶zleri Ã½zerine tepki yaÄŸmaya devam ediyor mhp genel_baÅ
16	siyaset , yaptÄ±ÄŸÄ±m bir mÃ¼lakat dinlenmiÅŸti adalet_bakanÄ±_sadullah_ergin den dinleme iddiasÄ± adalet_bakanÄ±_sadullah_ergin ankara_baÅŸsavcÄ±lÄ±ÄŸÄ±nÄ± bir raporu var 2001 yÄ±lÄ±
17	siyaset , bakan_yÄ±ldÄ±rÄ±m yolsuzluk yapan alÃ§aktÄ±r ulaÅŸtÄ±rma denizcilik ve haberleÅŸme_bakanÄ±_binali_yÄ±ldÄ±rÄ±m yolsuzluk yapan da yolsuzluÄŸa vesile olan da alÃ§aktÄ±r bu memle
18	siyaset , chp diyarbakÄ±r a baÅŸkan dayanmÄ±yor diyarbakir da chp il_baÅŸkanÄ± m beÅŸir ipekÃ§i ve 24 kiÅŸilik yÃ¶netim kurulu genel merkezin talimatÄ± ile gÃ¶revden alÄ±ndÄ± olaÄŸanÃ¼stÃ
19	siyaset , akp mutabakata engel oluyor mhp li bal anayasa Ã§alÄ±ÅŸmalarÄ±nÄ± deÄŸerlendirdi anayasa_uzlaÅŸma_komisyonu Ã¼yesi ve mhp konya milletvekili faruk_bal ak_parti nin tutumunun ye
20	siyaset , demokrasimiz ilerliyor ama hatalar da var cumhurbaÅŸkanÄ±_abdullah_gÃ¼l foreign affairs dergisine verdiÄŸi mÃ¼lakatta gÃ¼ndeme iliÅŸkin konularÄ± deÄŸerlendirdi cumhurbaÅŸkanÄ±_
21	siyaset , patriotlar iÃ§in genel gÃ¶rÃ¼ÅŸme istemi chp tÃ¼rkiye de konuÅŸlandÄ±rÄ±lmasÄ±na karar verilen patriot fÃ¼ze bataryalarÄ± ile ilgili meclis te genel gÃ¶rÃ¼ÅŸme aÃ§Ä±lmasÄ±nÄ± talep
22	siyaset , tÃ¼rkiye nin ilk ombudsmanÄ±na iptal davasÄ± chp tÃ¼rkiye nin ilk ombudsmanÄ± iÃ§in iiptal davasÄ± cumhuriyet_halk_partisi chp tÃ¼rkiye nin ilk ombudsmanÄ± seÃ§ilen nihat Ã¶mero
23	siyaset , bdp Ã¶nerilerini sundu bdp tbmm anayasa_uzlaÅŸma_komisyonu na yasama bÃ¶lÃ¼mÃ¼ne iliÅŸkin Ã¶nerilerini sundu ak_parti chp ve mhp nin ardÄ±ndan bdp de dÃ¼n komisyona Ã¶neri

METODOLOJİ

1. Kütüphaneleri İçe Aktarma ve Veri Ön İşleme

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from keras.models import Sequential
from keras.layers import Embedding, Bidirectional, LSTM, Dropout, Dense
from keras.preprocessing.text import text_to_word_sequence
from keras.preprocessing.sequence import pad_sequences
from gensim.models import Word2Vec
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Embedding, Dropout
from sklearn.preprocessing import StandardScaler
from keras.layers import Flatten
```

```
# Step 1: Dataset Yukleme
file_path = '/kaggle/input/ttc4900/7allV03.csv'
df = pd.read_csv(file_path)

# Step 2: Veri On Isleme
train_data, test_data, train_labels, test_labels = train_test_split(
    df['text'], df['category'], test_size=0.2, random_state=42
)
label_encoder = LabelEncoder()
train_labels_encoded = label_encoder.fit_transform(train_labels)
test_labels_encoded = label_encoder.transform(test_labels)
```

1. Kod, veri işleme için '***pandas***', veri kümesini bölmek için **sklearn.model_selection**'dan *train_test_split*, kategorik label'leri kodlamak için *LabelEncoder* ve bir sinir ağı oluşturmak için **Keras**'tan çeşitli katmanlar ve modeller gibi gerekli kitaplıkların içe aktarılmasıyla başlar.
2. Veri kümesi, pandas kitaplığından pd.read_csv kullanılarak belirtilen CSV dosyasından ('/kaggle/input/ttc4900/7allV03.csv') yüklenir.
3. Veri seti, **scikit-learn**'den *train_test_split* kullanılarak eğitim ve test setlerine bölünmüştür.
4. Label'lar (kategoriler), kategorik label'leri sayısal formata dönüştürmek için *LabelEncoder* kullanılarak kodlanır.

Yukarıdaki kod, özellikle metin sınıflandırması için bir doğal dil işleme (DDİ) projesinin ilk aşamalarını oluşturur. Bir veri kümesini yükler, veri ön işlemesini gerçekleştirir ve verileri tokenizasyon ve model oluşturma gibi sonraki adımlara

hazırlar. Spesifik veri kümesi ve yapısı, sağlanan kod parçacığında yer almayan sonraki adımları etkileyecektir.

2. Tokenleştirme ve Word2Vec ile Kelime Gömme

```
# Step 3: Word2Vec ile Tokenizasyon ve Kelime Gomme
preprocessed_data = [text_to_word_sequence(text) for text in train_data]
word2vec_model = Word2Vec(sentences=preprocessed_data, vector_size=128, window=5, min_count=1, workers=4)
word_index = {word: index for index, word in enumerate(word2vec_model.wv.index_to_key)}


# Sozlukteki her kelime icin Word2Vec yerlestirmelerini yazdirma
max_words_to_display = 1
for word, index in word_index.items():
    if index < max_words_to_display:
        try:
            embedding_vector = word2vec_model.wv[word]
            print(f"Word: {word}, Embedding: {embedding_vector}")
        except KeyError:
            print(f"Word: {word} not found in Word2Vec embeddings.")


train_sequences_word2vec = [[word_index[word] for word in text_to_word_sequence(text) if word in word_index] for text in train_data]
test_sequences_word2vec = [[word_index[word] for word in text_to_word_sequence(text) if word in word_index] for text in test_data]

max_length = 100
train_padded_word2vec = pad_sequences(train_sequences_word2vec, maxlen=max_length, truncating='post')
test_padded_word2vec = pad_sequences(test_sequences_word2vec, maxlen=max_length, truncating='post')
```

1. *preprocessed_data*, **Keras**'ın *text_to_word_sequence* özelliğini kullanarak eğitim verilerindeki her metni tokenize eden bir liste anlayışıdır. Bu, her bir iç listenin önceden işlenmiş bir metnin sözcüklerini içerdiği bir liste listesi oluşturur.
2. Daha sonra **Word2Vec**, önceden işlenmiş verilere dayalı olarak sözcük gömmeleri oluşturmak için kullanılır. Model, **vector*_size*** (kelime vektörlerinin boyutluluğu), ***window*** (bir cümle içindeki ve tahmin edilen kelime arasındaki maksimum mesafe), ***min*_count** (toplam frekansı bundan daha düşük olan tüm kelimeleri yok sayar) ve ***workers*** (modeli eğitirken kullanılacak CPU çekirdeği sayısı) gibi parametrelerle eğitilir.
3. ***word_index***, her kelimeyi **Word2Vec** modelinin sözlüğündeki indeksiyle eşlemek için yaratılmıştır.
4. Kod daha sonra sözcük dağarcığından birkaç sözcük için sözcük gömmelerini yazdırır ***(max_words_to_display)***. Kelimenin indeksinin belirtilen limitten küçük olup olmadığını kontrol eder ve gömme vektörünü almaya çalışır. Eğer kelime gömmelerde bulunamazsa, bir mesaj yazdırarak ***KeyError***'ı işler.
5. ***train_sequences_word2vec*** ve ***test_sequences_word2vec***, eğitim ve test verilerindeki her kelimenin Word2Vec modelinin sözlüğündeki ilgili indeksle eşleştirilmesiyle oluşturulur.

6. Daha sonra diziler, hepsinin aynı uzunluğa ***(max_length)*** sahip olmasını sağlamak için ***pad_sequences*** kullanılarak doldurulur. Padding sonradan yapılır; bu, dizilerin belirtilen uzunluktan daha kısa olması durumunda sonuna sıfırların ekleneceği anlamına gelir.

Özetle, kodun bu bölümü, metin verilerini **tokenleştirme**, ***Word2Vec*** ile gömme ve tekdüze uzunluk sağlamak için dizileri doldurma yoluyla bir sinir ağına giriş için hazırlar. Word2Vec gömmeleri, veri kümesindeki kelimeler arasındaki anlamsal ilişkileri yakalar.

3. Yapay Sinir Ağı Mimarisi

```
# Step 4: Word2Vec ve Model Mimarinizle Sinir Agini Olusturma ve Egitme
embedding_matrix = word2vec_model.wv.vectors
embedding_layer = Embedding(input_dim=embedding_matrix.shape[0], output_dim=embedding_matrix.shape[1],
input_length=max_length, weights=[embedding_matrix], trainable=False)

# Flatten the embedded sequences
flatten_layer = Flatten()

# Dense layers
dense_layer1 = Dense(128, activation='relu')
dropout_layer1 = Dropout(0.5)
dense_layer2 = Dense(64, activation='relu')
dropout_layer2 = Dropout(0.5)

# Output layer
output_layer = Dense(7, activation='softmax')

# Modeli olusturma
model_ann = Sequential([
    embedding_layer,
    flatten_layer,
    dense_layer1,
    dropout_layer1,
    dense_layer2,
    dropout_layer2,
    output_layer
])
```

1. ***embedding_matrix***, önceden eğitilmiş modelden elde edilen **Word2Vec** yerleştirmelerinin matrisidir.
2. ***embedding_layer***, **Keras**'taki Gömme katmanının bir örneğidir. Word2Vec yerleştirme matrisi ile başlatılır.
3. ***input_dim***, Word2Vec modelindeki benzersiz kelimelerin sayısına ayarlanır, ***output_dim***, kelime vektörlerinin boyutluluğudur ve ***input_length***, giriş dizilerinin uzunluğudur (***max_length'e*** ayarlıdır).
4. ***weights*** parametresi önceden eğitilmiş Word2Vec gömmelerine ayarlanmıştır ve eğitim sırasında gömmeleri sabit tutmak için ***trainable=false*** olarak ayarlanmıştır.

5. ***flatten_layer*** Keras'taki **Flatten** katmanının bir örneğidir. Gömme katmanından çıkan çıktıyı düzleştirmek için kullanılır. Dense katmanlar için 3 boyutlu girişten (embedding layer output) 1 boyutlu girişe geçiş yaparken bu gereklidir.
6. ***dense_layer1***, 128 nörona ve **ReLU** aktivasyonuna sahip ilk dense katmandır.
7. ***dropout_layer1***, ilk dense katmandan sonra dropout oranı 0,5 olan bir dropout katmanıdır. Dropout, eğitim sırasında her güncellemede giriş birimlerinin bir kısmını rastgele 0'a ayarlayarak overfitting'i önlenmesine yardımcı olur.
8. ***dense_layer2*** , 64 nörona ve **ReLU** aktivasyonuna sahip ikinci dense katmandır.
9. ***dropout_layer2***, ikinci dense katmandan sonra dropout oranı 0,5 olan başka bir dropout katmanıdır.
10. ***Output_layer***, 7 nöronlu (7 kategoriniz olduğundan mütevellit) ve bir **softmax** aktivasyon fonksiyonundan oluşan çıktı katmanıdır. Softmax, her sınıf için olasılıkların çıktısını almak amacıyla çoklu sınıf sınıflandırması için kullanılır.

4. Modeli Derleme, Eğitme, Değerlendirme, Kaydetme ve Yükleme

```
# Modeli Derleme
model_ann.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])

# Modeli Egitme
model_ann.fit(train_padded_word2vec, train_labels_encoded, epochs=10, validation_data=(test_padded_word2vec, test_labels_encoded))

# Modeli degerlendirme
test_loss, test_acc = model_ann.evaluate(test_padded_word2vec, test_labels_encoded)
print(f'Test Accuracy: {test_acc}')

# Modeli Kaydetme
model_ann.save("/kaggle/working/my_word2vec_model")
# Modeli Yukleme
from keras.models import load_model
loaded_model_word2vec = load_model("/kaggle/working/my_word2vec_model")
```

1. ***model_ann*** yukarıda tanımlanan katmanlardan oluşan sıralı bir modeldir. Sinir ağı mimarisini temsil eder.
2. Model, ***Adam optimizer***, kayıp fonksiyonu olarak ***sparse categorical crossentropy*** ve değerlendirme ölçütü olarak ***accuracy*** kullanılarak derlenir.
3. ***model_ann.fit()*** sinir ağını eğitmek için kullanılır. Girdi olarak yastıklı eğitim dizilerini ***(train_padded_word2vec)*** ve karşılık gelen kodlanmış etiketleri ***(train_labels_encoded)*** alır.
4. Eğitim 10 defa (epochs=10) gerçekleştirilir ve eğitim sırasında modelin görünmeyen veriler üzerindeki performansını izlemek için doğrulama verileri ***(test_padded_word2vec ve test_labels_encoded)*** sağlanır.

5. ***model_ann.evaluate()***, modelin test verileri üzerindeki performansını değerlendirmek için kullanılır. Padded test dizilerini ***(test_padded_word2vec)*** ve karşılık gelen kodlanmış etiketleri ***(test_labels_encoded)*** alır.
6. ***model_ann.save()*** eğitilen modeli bir dosyaya kaydetmek için kullanılır.
7. Keras'ın ***load_model***'i kayıtlı bir modeli tekrar belleğe yüklemek için kullanılır.

5. Otomatik Veri Çekme

```
import feedparser
import re
import numpy as np
from bs4 import BeautifulSoup
from unidecode import unidecode
import pandas as pd

# Metni temizleme ve on isleme fonksiyonu
def preprocess_text(text):
    # Remove HTML tags
    soup = BeautifulSoup(text, 'html.parser')
    text = soup.get_text(separator=' ')

    # Remove URLs
    text = re.sub(r'http\S+', '', text)
    # Convert to lowercase
    text = text.lower()
    # Convert Turkish characters to ASCII
    text = unidecode(text)
    # Remove non-alphabetic characters
    text = re.sub(r'[^a-zA-Z\s]', '', text)
    # Remove extra whitespaces
    text = ' '.join(text.split())
    return text
```

1. ***BeautifulSoup***, HTML içeriğini ayrıştırmak ve giriş metninden tüm HTML etiketlerini kaldırmak için kullanılır.
2. ***re* (regular expression)** modülü, URL'leri metinden kaldırmak için kullanılır. Metindeki URL'leri eşleştirmek ve kaldırmak için *r'http\S+'* normal ifadesi kullanılır.
3. Tutarlılığı sağlamak ve verilerin boyutunu azaltmak için metnin tamamı küçük harfe dönüştürülür.
4. ***Unidecode*** kütüphanesi, herhangi bir Türkçe karakteri en yakın ASCII eşdeğerine dönüştürmek için kullanılır.
5. *r'[^a-zA-Z\s]'* normal ifadesi, alfabetik (harfler) veya boşluk olmayan karakterleri kaldırmak için kullanılır.
6. Fazladan boşluklar, metni kelimelere bölerek ve ardından tekrar birleştirerek kaldırılır.

```python
def tokenize_and_pad(text, word_index, max_length):
    sequence = [word_index[word] for word in text_to_word_sequence(text) if word in word_index]
    padded_sequence = pad_sequences([sequence], maxlen=max_length, truncating='post')
    return padded_sequence

# URL'yi girme
rss_feed_url = 'https://www.ntv.com.tr/son-dakika.rss'

# Parse the RSS feed
feed = feedparser.parse(rss_feed_url)

# Icerigi ayiklama ve on isleme
contents = [preprocess_text(entry.content[0].value) for entry in feed.entries]

# Tokenlestirme ve pad sequences
sequences = [tokenize_and_pad(content, word_index, max_length) for content in contents if content]

# Herhangi bir dizinin mevcut olup olmadigini kontrol etme
if not sequences:
    print("No valid sequences found. Please check your data.")

# Dizilerin uzunluklarini yazdirma
for i, seq in enumerate(sequences, 1):
    print(f"Sequence {i} Length: {len(seq[0])}")

# Yuklenen modeli kullanarak tahminler yapma
predictions = loaded_model_word2vec.predict(np.vstack(sequences))

# Decode predictions
decoded_predictions = label_encoder.inverse_transform(np.argmax(predictions, axis=1))
```

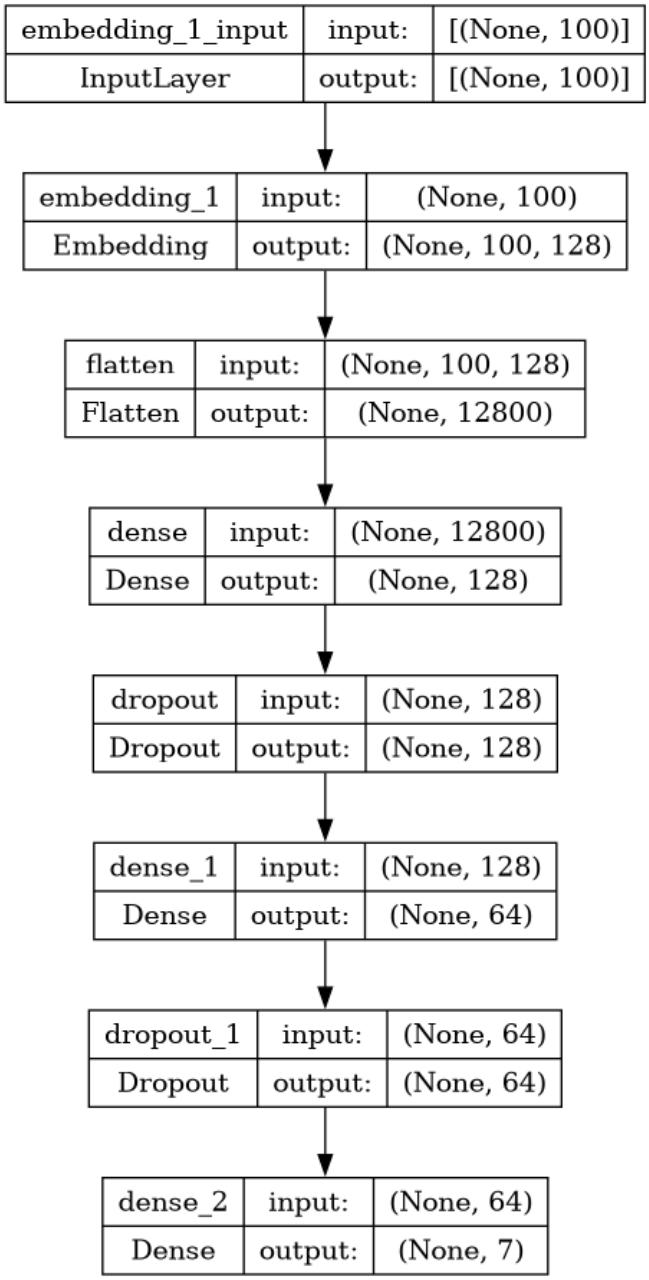
7. ***tokenize_and_pad*** fonksiyonu girdi olarak bir metni, bir ***word_index*** sözlüğünü ve ***max_length***'i alır.
8. ***Text_to_word_sequence'ı*** kullanarak giriş metnini tokenleştirir ve ardından sağlanan ***word_index'i*** kullanarak kelimeleri karşılık gelen indekslere dönüştürür.
9. Ortaya çıkan dizi daha sonra ***pad_sequences*** kullanılarak sabit bir uzunluk sağlamak için doldurulur. Padding sonradan yapılır (sıranın sonuna sıfırlar eklenir).
10. Bir RSS beslemesi URL'si ***(rss_feed_url)*** belirtilir.
11. RSS beslemesindeki her girişin içeriği, daha önce tanımlanan ***preprocess_text*** fonksi kullanılarak önceden işlenir.
12. Önceden işlenmiş içerikler daha sonra ***tokenize_and_pad*** fonksiyonu kullanılarak tokenleştirilir ve doldurulur. Diziler dizi listesinde saklanır.
13. Yüklenen Word2Vec modeli ***(loaded_model_word2vec),*** tokenize edilmiş ve dolgulu diziler üzerinde tahminler yapmak için kullanılır.
14. Kodlanmış tahminlerin kodu, kategori label'lerini elde etmek için label_encoder kullanılarak çözülür.

```
# Icerik ve tahminlerle bir DataFrame olusturma
output_data = {'Haber': contents, 'Kategorizasyon': decoded_predictions}
output_df = pd.DataFrame(output_data)

# DataFrame'i bir CSV dosyasina kaydetme
output_csv_path = '/kaggle/working/output_predictions01.csv'  # Specify the desired path
output_df.to_csv(output_csv_path, index=False,sep=';')

# CSV dosya yolunu belirten bir mesaj yazdirma
print(f'Output saved to CSV: {output_csv_path}')
```

15. 'Haber' (içerik) ve 'Kategorizasyon' (şifresi çözülmüş tahminler) tuşlarıyla ***output_data*** adında bir sözlük oluşturulur.
16. Bu sözlüğü kullanarak bir *pandas* DataFrame ***(output_df)*** oluşturulur.
17. ***DataFrame***, ***to_csv*** yöntemi kullanılarak bir CSV dosyasına kaydedilir.
18. Sonuç:

	A	B	C
1	Haber	Kategorizasyon	
2	anayasa mahkemesi aym gezi parki davasi hukumlusu turkiye isci partisi tip hatay milletvekili can atalay hakkindaki ikinci hak ihlali ba	teknoloji	
3	istanbul avcilarda kentsel donusum kapsaminda yikilmasi planlanarak kismen bosaltilan binadaki bir daireye yerlesen hirsiz apartmar	teknoloji	
4	klasik otomobillere tutkusuyla taninan mhp genel baskani devlet bahceli partisinin antalya milletvekili abdurrahman baskana model	teknoloji	
5	osmaniyede polis ekiplerinin durdurdugu hafif ticari aracta suriyeli kacak gocmen yakalandi arac surucusu me ile ona baska aracla reh	teknoloji	
6	turkiyede oyuncak sektorunun ihracati yilda katina cikarken ithalat yuzde geriledi turkiye istatistik kurumu tuik verilerine gore turkiye	saglik	
7	burdur golu kenarina kimligi belirsiz kisi veya kisilerce insaat molozu ve mermer atiklari dokuldu moloz ve atik dokenlerin tespiti icin c	teknoloji	
8	iskur verilerine gore isverenler en cok ozel guvenlik gorevlisi satis danismani turizm ve otelcilik elemani garson ve konfeksiyon iscisi tal	teknoloji	
9	olasi istanbul depremi yillardir gundemde yuz binlerce binanin yenilenmesi gerekiyor bazi binalarin yikilmasi icin deprem olmasina da	teknoloji	
10	yasinda manisali bir kiz cocugu ciftligin afgan calisani tarafindan kacirilarak irana goturulmustu o kiz interpolun de devreye girmesiyle	teknoloji	
11	takasbank tarafindan subat de vatandaslarin hizmetine sunulan tasit takas sistemine artik edevlet uzerinden de ulasilabilecek tasit ali	teknoloji	
12	kagithane seyrantepede evinin onunde scooter ile yokus asagiya kayan cocuk otomobilin altinda kalarak agir sekilde yaralandi kucuk	teknoloji	
13	canakkalede kaz daglarindan beslenen bayramic barajinin yuzde a dusen su seviyesi yagislar sonrasi ayda yuzde a ulasti turkiyede oze	teknoloji	
14	ankaragucucaykur rizespor maci sonrasinda faruk kocanin saldirisina ugrayan hakem halil umut meler taburcu oldu taburcu islemlerir	teknoloji	
15	yili ekim ayinda olarak elde edilmis olan dis ticaret haddi yili ekim ayinda puan artarak oldu turkiye istatistik kurumu tuik yili ekim ayi	teknoloji	
16	insaat maliyet endeksi yillik yuzde artti aylik yuzde artti turkiye istatistik kurumu tuik ekim ayi insaat maliyet endeksini acikladi buna g	teknoloji	
17	sabit fiyatlarla perakende satis hacmi ekim ayinda bir onceki aya gore yuzde artti turkiye istatistik kurumu tuik ekim ayi perakende sat	teknoloji	
18	borsa istanbulda bist endeksi gune yuzde dususle puandan basladi acilista bist endeksi onceki kapanisa gore puan azalisla ve yuzde d	teknoloji	
19	istanbulda yasa disi bahis oynattigi belirlenen zanlilara yonelik operasyon duzenlendi operasyonda supheli gozaltina alindi istanbuld	teknoloji	
20	turkiyenin uzay yolcusu alper gezeravci son hazirliklarini tamamladi cumhuriyet tarihinin ilk insanli uzay gorevini gerceklestirmek uzer	teknoloji	
21	toki ilde konut ve ilde is yerini acik artirma ile satisa sunuyor toki istanbul hizmet binasinda gerceklestirilecek acik artirma surecinin a	teknoloji	

Yapay Sinir Ağlarında Oluşturulan Ağın Şekli



AĞ TOPOLOJİSİ VE VARSAYIMLAR

Embedding Katman:

İlk katman, kelime indekslerini yoğun vektörlere dönüştürmek için kullanılan Gömme katmanıdır. Önceden eğitilmiş Word2Vec yerleştirmelerini kullanır.

Flatten Katman:

Bu katman, gömme katmanından gelen çıktıyı 1 boyutlu bir diziye düzleştirmek için kullanılır. Verileri sonraki yoğun katmanlar için hazırlar.

Dense Katman 1:

128 nöronlu ve ReLU aktivasyon fonksiyonuna sahip ilk Dense katman.

Dropout Katmanı 1:

İlk dense katmandan sonra dropout oranı 0,5 olan bir dropout katmanı uygulanır. Dropout, overfitting önlenmesine yardımcı olur.

Dense Katman 2:

64 nöronlu ve ReLU aktivasyon fonksiyonuna sahip ikinci dense katman.

Dropout Katmanı 2:

İkinci yoğun katmandan sonra dropout oranı 0,5 olan başka bir dropout katmanı uygulanır.

Çıkış Katmanı:

7 nöronlu (7 sınıflı bir sınıflandırma görevi olduğu için) ve softmax aktivasyon fonksiyonundan oluşan çıktı katmanı.

Model Derlemesi:

Model, çok sınıflı sınıflandırma problemlerine uygun olan Adam optimizer ve sparse categorical cross-entropy kaybı kullanılarak derlenmiştir.

Model Eğitimi:

Model, 10 epoch boyunca sağlanan eğitim verileri ***(train_padded_word2vec)*** üzerinde eğitilir.

Varsayımlar

Word2Vec gömmeleri:

Kod, Word2Vec gömmelerinin eğitim verileri üzerinde önceden eğitildiğini varsayar.

Veri Ön İşleme:

Giriş verilerinin ***(df['text'])*** eğitimden önce, etiketlerin tokenleştirilmesi ve kodlanması da dahil olmak üzere önceden işlendiğini varsayar.

Model Mimarisi:

Mimari, sağlanan dense katmanların ve çıktı katmanının verilen NLP görevi için uygun olduğunu varsayar.

Eğitim Parametreleri:

Komut dosyası, eğitim için sabit sayıda epoch (10) kullanır. Modelin yakınsamasına bağlı olarak ayarlamalar yapılması gerekebilir.

Değerlendirme Metriği:

Komut dosyası, değerlendirme ölçütü olarak **accuracy** kullanır. Göreve bağlı olarak hassasiyet, geri çağırma veya F1 puanı gibi diğer ölçümler de alakalı olabilir.

Katıştırma Katmanı Eğitilebilirliği:

Gömme katmanı eğitilemez olacak şekilde ayarlandı **(trainable=False).** Bu, önceden eğitilmiş yerleştirmelerde eğitim sırasında fine-tuned yapılmaması gerektiğini varsayar.

Optimum Ağ Tasarımı

Model mimarisi, giriş dizilerini temsil etmek üzere önceden eğitilmiş Word2Vec gömmeleriyle başlatılan bir Gömme katmanından oluşur. Gömme katmanını, gömülü dizileri yeniden şekillendirmek için bir Flatten katmanı izler. Daha sonra, her ikisi de ReLU aktivasyon fonksiyonlarını kullanan, sırasıyla 128 ve 64 nöron içeren iki Dense katman birleştirilir. Overfitting'i azaltmak için her dense katmanı 0,5 bırakma oranına sahip dropout katmanları takip eder. Son katman, varsayılan çıktı sınıfı sayısıyla eşleşecek 7 nörona sahip, softmax aktivasyonuna sahip Yoğun bir katmandır. Bu ağı optimize etmek için potansiyel iyileştirmeler, farklı aktivasyon fonksiyonlarının denenmesini, yoğun katmanlardaki nöron sayısının ayarlanmasını, ayrılma oranlarının ayarlanmasını ve ek gizli katmanların veya yinelenen katmanların dahil edilmesi gibi alternatif mimarilerin araştırılması işlemleri yapıldı. Sistematik bir hiperparametre araması ve çapraz doğrulama süreci yoluyla bu yönlere ince ayar yapmak, eldeki belirli NLP görevi için en uygun konfigürasyonun belirlenmesine yardımcı olabilir.(Gelecekteki Model geliştirşmeleri için)

Karşılaştırma

Embedding Layer:

Her iki tasarım da önceden eğitilmiş Word2Vec gömmeleriyle başlatılan, eğitilemeyen bir Embedding katmanı kullanıyor.

Dense Layers:

İlk tasarım daha büyük yoğun katmanlara (300 ve 40 nöron) sahipken, optimal tasarım daha küçük katmanlara (128 ve 64 nöron) sahiptir. Bu ayarlama model karmaşıklığının azaltılmasına yardımcı olabilir.

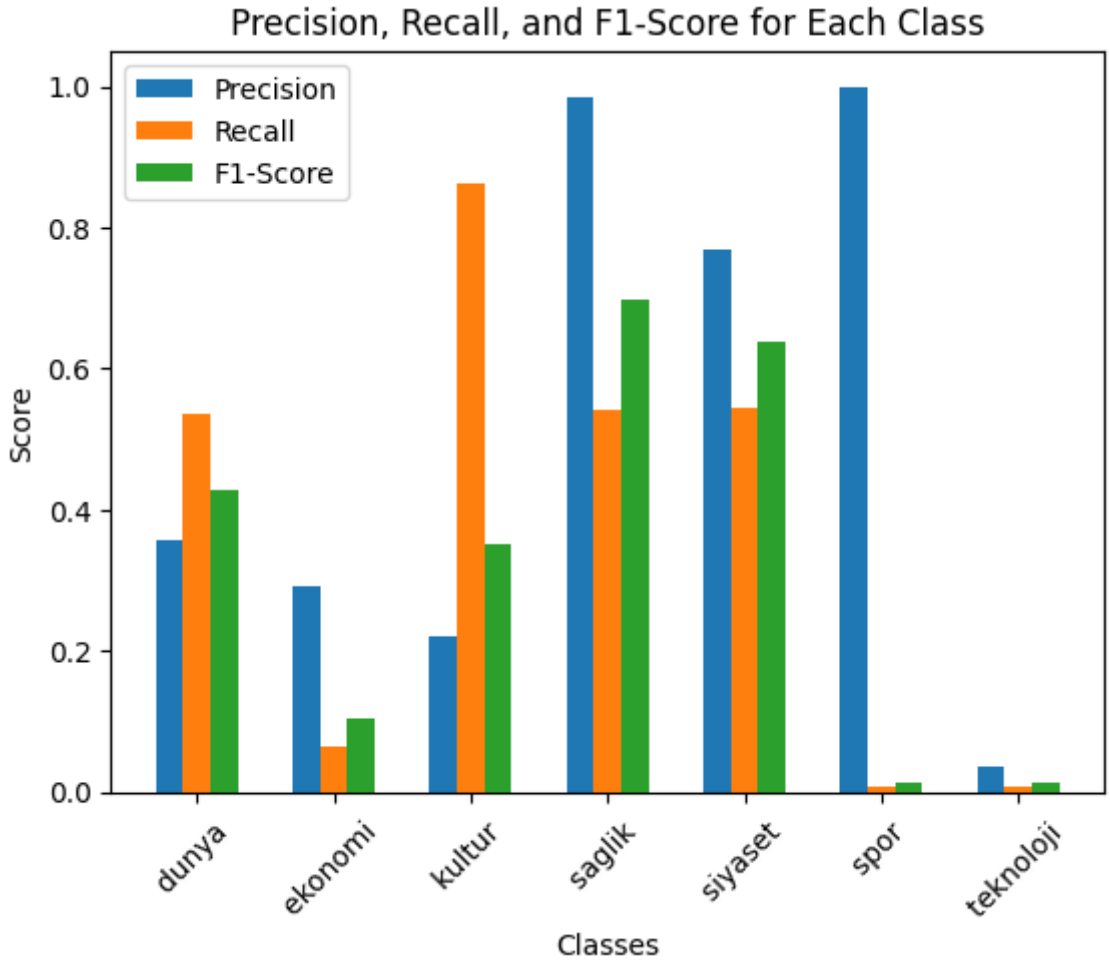
Dropout Layers:

Dropout oranları optimal tasarımda değiştirildi (ilk katman için 0,7 ila 0,5 ve ikinci katman için 0,2 ila 0,5). Bu ince ayar aşırı uyumu dengelemeye yardımcı olur.

Optimum tasarım, yoğun katmanların boyutunu azaltarak ve bırakma oranlarını ayarlayarak potansiyel aşırı uyum sorunlarını gideriyor gibi görünüyor. Bu, görünmeyen veriler üzerinde daha iyi genelleme yapılmasına yol açabilir.

"Optimal" tasarımın göreve bağlı olduğunu ve veri kümesinin belirli özelliklerine ve model karmaşıklığı ile genelleme arasında istenen dengeye göre ayarlamalar yapılması gerektiğini unutmamak önemlidir. Belirli bir NLP görevi için en etkili mimariyi belirlemek amacıyla daha fazla ince ayar ve deneme yapılması gerekebilir.(Gelecekte modelin tekrar geliştirilebilmesim için)

Model Eğitimi İlerlemesini ve Değerlendirme Sonuçlarını Görselleştirme



KAYNAKLAR

Çabuk, M. (2021). E-Ticaret Ürün Yorumları. Manisa Celal Bayar Üniversitesi. https://www.kaggle.com/datasets/mujdatcabuk/eticaret-urun-yorumlari/data

Yıldırım, S. (2020). A Benchmark Data for Turkish Text Categorization. İstanbul Bilgi Üniversitesi. https://www.kaggle.com/datasets/savasy/ttc4900/

Wang, Y. D., Blunt, M. J., Armstrong, R. T., & Mostaghimi, P. (2021). Deep Learning in Pore Scale Imaging and Modeling. Earth-Science Reviews, 215(2), 103555. doi:10.1016/j.earscirev.2021.103555