

**İstanbul Medeniyet Üniversitesi, Mühendislik ve Doğa Bilimleri Fakültesi, Bilgisayar
Mühendisliği Bölümü**

Buğrahan Ata - 22120205039

RoBERTa Kullanarak Twitter Verileri Üzerinde Çok Sınıflı Duygu Sınıflandırması: FIFA 2022 Örnek Çalışması

Özet

Bu çalışma, 2022 FIFA Dünya Kupası ile ilgili tweet'ler üzerinde dönüştürücü (transformer) tabanlı derin öğrenme yöntemleri kullanılarak çok sınıflı duygu analizi gerçekleştirmektedir. Kullanılan veri seti, pozitif, nötr ve negatif olmak üzere etiketlenmiş 22.000'den fazla tweet içermektedir. URL, kullanıcı etiketleri ve hashtag temizleme, emoji–metin dönüştürme, kısaltma normalizasyonu, durak kelime (stopword) kaldırma ve lemmatizasyon gibi kapsamlı bir ön işleme süreci uygulanmıştır.

Khan ve Srivastava (2024) tarafından yapılan yakın tarihli bir çalışmada, aynı veri seti üzerinde VADER, XGBoost, Random Forest ve LSTM gibi çeşitli makine öğrenimi modelleri değerlendirilmiştir. Bu modeller arasında en yüksek doğruluk, %73 ile Çift Yönlü LSTM (Bi-LSTM) modeli tarafından elde edilmiştir. Ancak bu yöntemler, geleneksel mimarilere ve statik kelime gömmelerine dayanması nedeniyle transformer tabanlı modellerin sunduğu derin bağlamsal anlam çıkarımını sağlayamamıştır.

Bu çalışmada RoBERTa tabanlı bir dil modeli kullanılmış ve sınıf ağırlıklı kayıp fonksiyonu ile altı epoch boyunca ince ayar yapılarak eğitilmiştir. Önerilen model, test verisi üzerinde “%81.11” doğruluk ve tüm duygu sınıflarında güçlü F1-skorları elde etmiştir. Sonuçlar, önceki yaklaşımlara kıyasla önemli bir iyileşme sağlandığını ve bağlamsal gömme yöntemleri ile transformer tabanlı mimarilerin gerçek dünya duygu sınıflandırma görevlerinde üstün performans sunduğunu göstermektedir.

1. Giriş

Twitter gibi sosyal medya platformlarının kullanımının giderek artmasıyla birlikte, kamuoyunun duygu durumunu otomatik olarak analiz etme ve sınıflandırma yeteneği, doğal dil işleme alanında önemli bir araştırma konusu hâline gelmiştir. Duygu analizi, metin verilerinin pozitif, nötr veya negatif gibi duygusal kutuplara ayrılmasını sağlayarak; politika, spor, pazarlama ve kriz yönetimi gibi birçok alanda değerli içgörüler sunmaktadır.

Bununla birlikte, Twitter verileri üzerinde duygu analizi yapmak; kullanıcı tarafından üretilen içeriklerin gürültülü, düzensiz ve sıklıkla kısaltmalar içeren yapısı nedeniyle kendine özgü zorluklar barındırmaktadır. Geleneksel makine öğrenimi yöntemleri çoğunlukla el ile oluşturulmuş özelliklere veya yüzeysel temsil yöntemlerine dayandığından, kısa fakat anlamsal açıdan zengin sosyal medya paylaşımlarının bağlamını yeterince yakalayamayabilmektedir.

Derin öğrenme alanındaki son gelişmeler, Tekrarlayan Sinir Ağları (Recurrent Neural Networks – RNN) ve Uzun Kısa Süreli Bellek (Long Short-Term Memory – LSTM) gibi daha güçlü mimarilerin ortaya çıkmasını sağlamıştır. Khan ve Srivastava (2024) tarafından gerçekleştirilen bir çalışmada, FIFA Dünya Kupası 2022 ile ilgili tweet verileri üzerinde LSTM, Random Forest ve XGBoost modelleri uygulanmış; Bi-LSTM modeli kullanılarak en yüksek test doğruluğu %73 olarak elde edilmiştir.

Bu çalışmada, söz konusu sonuçları geliştirmek amacıyla metin içerisindeki derin bağlamsal bağımlılıkları modelleyebilen transformer tabanlı bir mimari olan RoBERTa kullanılmıştır. Model, geliştirilmiş bir ön işleme hattı ve sınıf ağırlıklı eğitim stratejisi ile birleştirilerek hem veri gürültüsü hem de sınıf dengesizliği problemleri ele alınmıştır. Çalışmanın temel amacı, gerçek dünya Twitter verileri üzerinde duygu sınıflandırması için transformer tabanlı modellerin üstün performansını ortaya koymaktır.

2. İlişkili Çalışmalar

Duygu analizi alanında gerçekleştirilen önceki çalışmalar, hem geleneksel makine öğrenimi yöntemlerini hem de derin öğrenme tabanlı yaklaşımları kapsamaktadır. Naive Bayes ve Destek Vektör Makineleri (Support Vector Machines – SVM) gibi klasik modeller yaygın olarak kullanılmış olsa da, bu yöntemler bağlamsal bilgiyi yakalamada sınırlı kalmaktadır.

Khan ve Srivastava (2024), FIFA Dünya Kupası 2022 tweet verileri üzerinde Bi-LSTM ile birlikte XGBoost ve Random Forest gibi makine öğrenimi algoritmalarını uygulamış ve Word2Vec gömme yöntemleri kullanılarak Bi-LSTM modeli ile en yüksek doğruluk değeri olan %73'e ulaşmıştır. Ancak bu yaklaşım, sosyal medya platformlarında kullanılan gayriresmî ve dinamik metin yapılarının analizinde kritik öneme sahip olan derin bağlamsal anlam çıkarımını yeterince sağlayamamaktadır.

3. Materyaller ve Yöntemler

3.1. Kütüphane Kurulumu ve Ortam Yapılandırılması

Doğal dil işleme ve derin öğrenme için gerekli olan tüm bağımlılıklar pip kullanılarak kurulmuştur.

- **transformers**: Önceden eğitilmiş BERT ve RoBERTa modelleri ile bu modellere ait tokenizer'ların kullanımı için tercih edilmiştir.
- **torch ve torchvision**: PyTorch tabanlı modellerin oluşturulması ve eğitilmesi amacıyla kullanılmıştır.
- **scikit-learn**: Eğitim-test veri ayrımı ile performans değerlendirme metriklerinin hesaplanması için kullanılmıştır.
- **emoji**: Tweet metinlerinde yer alan emojielerin metinsel karşılıklarına dönüştürülmesi için kullanılmıştır.

Bu kurulumlar, tekrarlanabilir ve izole bir geliştirme ortamı oluşturulmasını sağlamıştır. GPU kullanılabilirliği `torch.cuda.is_available()` komutu ile doğrulanmıştır.

```
[ ] !pip install transformers
!pip install scikit-learn
!pip install torch torchvision torchaudio
!pip install emoji

Requirement already satisfied: transformers in /usr/local/lib/python3.11/dist-packages (4.51.3)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-packages (from transformers) (3.18.0)
Requirement already satisfied: huggingface-hub<1.0,>=0.30.0 in /usr/local/lib/python3.11/dist-packages (from transformers) (0.30.2)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.11/dist-packages (from transformers) (2.0.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from transformers) (24.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.11/dist-packages (from transformers) (6.0.2)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.11/dist-packages (from transformers) (2024.11.6)
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packages (from transformers) (2.32.3)
Requirement already satisfied: tokenizers<0.22,>=0.21 in /usr/local/lib/python3.11/dist-packages (from transformers) (0.21.1)
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.11/dist-packages (from transformers) (0.5.3)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.11/dist-packages (from transformers) (4.67.1)
```

3.2. Veri Setinin Yüklmesi

fifa_world_cup_2022_tweets.csv adlı veri seti, Google Colab ortamında `files.upload()` arayüzü kullanılarak sisteme yüklenmiştir. Veri seti, 2022 FIFA Dünya Kupası ile ilgili 22.000'den fazla İngilizce tweet içermekte olup, her tweet duygu durumuna göre pozitif, nötr veya negatif olarak etiketlenmiştir.

Veri seti aşağıdaki temel sütunlardan oluşmaktadır:

- **Tweet**: Ham tweet metnini ifade etmektedir.
- **Sentiment**: Tweet'e ait duygu etiketini (pozitif, negatif, nötr) göstermektedir.

```
[ ] uploaded = files.upload()

Choose Files  fifa_world...2_tweets.csv
• fifa_world_cup_2022_tweets.csv(text/csv) - 4547779 bytes, last modified: 12/8/2022 - 100% done
Saving fifa_world_cup_2022_tweets.csv to fifa_world_cup_2022_tweets (1).csv
```

3.3. Gerekli Python Kütüphanelerinin İçe Aktarılması

Çalışmada pandas, re, nltk ve matplotlib gibi Python paketleri içe aktarılmıştır.

- **pandas**: CSV dosyalarının okunması ve veri çerçevesi (DataFrame) işlemlerinin gerçekleştirilmesi için kullanılmıştır.
- **re**: Metin temizleme aşamasında düzenli ifadeler (regular expressions) tabanlı işlemler için kullanılmıştır.
- **nltk**: Durak kelime (stopword) listeleri, tokenizasyon araçları ve WordNetLemmatizer aracılığıyla lemmatizasyon işlemleri için kullanılmıştır.

Bu kapsamda aşağıdaki NLTK modülleri de indirilmiştir:

- stopwords
- punkt
- wordnet

```
[ ] # nltk verilerini indiriyoruz
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('punkt_tab')
nltk.download('wordnet')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data] Package punkt is already up-to-date!
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data] Package punkt_tab is already up-to-date!
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data] Package wordnet is already up-to-date!
True
```

3.4. Ham Metin Temizleme

Her bir tweet'i temizlemek amacıyla düzenli ifadeler (regular expressions) kullanan özel bir fonksiyon tanımlanmıştır. Bu kapsamda gerçekleştirilen işlemler aşağıda sıralanmaktadır:

- URL'lerin kaldırılması (http\S+ ifadesi kullanılarak),
- Kullanıcı etiketlerinin (örneğin, @kullaniciadi) silinmesi,
- Hashtag'lerin ve özel karakterlerin temizlenmesi,
- Tüm metnin küçük harfe dönüştürülmesi.

Bu adım, gürültülü sosyal medya metnlerinin daha temiz ve normalize edilmiş bir formata dönüştürülmesini sağlamıştır.

```

raw_df = pd.read_csv("fifa_world_cup_2022_tweets.csv")

def clean_text(text):
    text = re.sub(r'http\S+', '', text)      # Linkleri kaldır
    text = re.sub(r'@\w+', '', text)         # Mention (@user) kaldır
    text = re.sub(r'#', '', text)            # Hashtag işaretini kaldır
    text = re.sub(r'[^\A-Za-z0-9\s]', '', text) # Noktalama işaretlerini temizle
    text = text.lower()                      # Tümünü küçük harfe çevir
    return text

raw_df['Clean_Tweet'] = raw_df['Tweet'].apply(clean_text)

```

3.5. Durak Kelime (Stopword) Kaldırma ve Lemmatizasyon

Veri içerisindeki anlamsal katkısı düşük olan gürültüyü azaltmak amacıyla aşağıdaki işlemler uygulanmıştır:

- İngilizce durak kelimeler (örneğin *“is”*, *“the”*, *“of”*) metinlerden kaldırılmıştır.
- Kalan kelimeler, kök hâllerine indirgenerek lemmatizasyon işlemine tabi tutulmuştur (örneğin *“running”* → *“run”*).

Bu işlemler, NLTK kütüphanesinde yer alan `word_tokenize()` fonksiyonu ve `WordNetLemmatizer` aracı kullanılarak gerçekleştirilmiştir.

```

stop_words = set(stopwords.words('english'))
lemmatizer = WordNetLemmatizer()

def preprocess_text(text):
    words = nltk.word_tokenize(text)
    words = [lemmatizer.lemmatize(word) for word in words if word not in stop_words]
    return ' '.join(words)

raw_df['Processed_Tweet'] = raw_df['Clean_Tweet'].apply(preprocess_text)

```

3.6. Ön İşlenmiş Verinin Kaydedilmesi

Ön işleme adımlarının tamamlanmasının ardından, temizlenmiş ve lemmatizasyon uygulanmış tweet’ler `processed_tweets.csv` adlı yeni bir CSV dosyasına kaydedilmiştir. Bu dosya yalnızca aşağıdaki iki sütunu içermektedir:

- `Processed_Tweet`
- `Sentiment`

Oluşturulan bu veri seti, modelin eğitim ve değerlendirme aşamalarında girdi olarak kullanılmıştır.

```

[ ] raw_df[['Processed_Tweet', 'Sentiment']].to_csv("processed_tweets.csv", index=False)

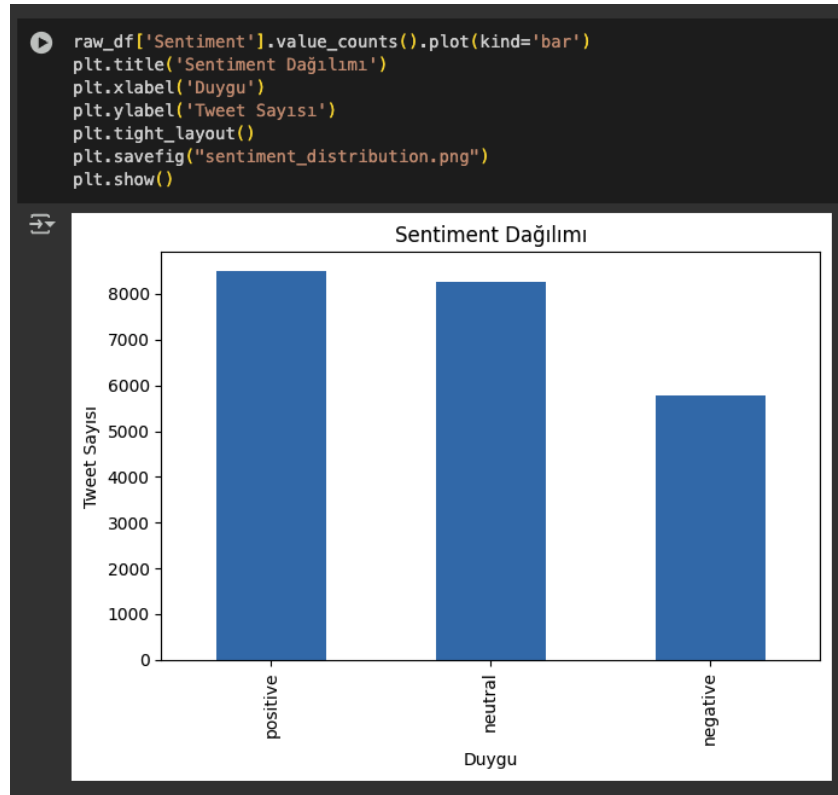
```

3.7. Duygu Dağılımının Görselleştirilmesi

Sınıf dengesizliğini analiz etmek amacıyla, her bir duygu etiketinin sayısı matplotlib kullanılarak çizilmiştir. Dağılım aşağıdaki şekildedir:

- **Pozitif:** ~8.500
- **Nötr:** ~8.200
- **Negatif:** ~5.700

Bu durum, azınlık sınıfın dengelenmesi için eğitim sırasında ağırlıklı bir kayıp fonksiyonuna ihtiyaç olduğunu doğrulamıştır.



3.8. Tokenizasyon ve Veri Kümesi Oluşturma

Her bir tweet'i alt kelime birimlerine ayırmak için HuggingFace kütüphanesinden RobertaTokenizer kullanılmıştır. Her tweet aşağıdaki şekilde kodlanmıştır:

- Maksimum 128 token uzunluğuna kırpma
- Sabit uzunluk sağlamak için padding uygulanması

Kodlanan girdiler ve etiketleri tutmak amacıyla özel bir PyTorch Dataset sınıfı oluşturulmuş ve bu yapı DataLoader API'si ile uyumlu hâle getirilmiştir.

```
# İşlenmiş CSV'den veriyi yükle
df = pd.read_csv("processed_tweets.csv")

# Tokenizer yükle (bert-base-uncased)
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')

# Metin ve etiket listelerini oluştur
texts = df['Processed_Tweet'].tolist()
labels = df['Sentiment'].tolist()
label_map = {'positive': 0, 'neutral': 1, 'negative': 2}
numerical_labels = [label_map[label] for label in labels]

# Tokenizer ile encode et
encodings = tokenizer(texts, truncation=True, padding=True, max_length=128)

# PyTorch Dataset tanımla
class TweetDataset(Dataset):
    def __init__(self, encodings, labels):
        self.encodings = encodings
        self.labels = labels
    def __getitem__(self, idx):
        item = {key: torch.tensor(val[idx]) for key, val in self.encodings.items()}
        item['labels'] = torch.tensor(self.labels[idx])
        return item
    def __len__(self):
        return len(self.labels)

dataset = TweetDataset(encodings, numerical_labels)
```

tokenizer_config.json: 100% 48.0/48.0 [00:00<00:00, 5.99kB/s]

vocab.txt: 100% 232k/232k [00:00<00:00, 17.6MB/s]

tokenizer.json: 100% 466k/466k [00:00<00:00, 38.4MB/s]

config.json: 100% 570/570 [00:00<00:00, 61.9kB/s]

3.9. Eğitim/Test Ayrımı ve DataLoader Kurulumu

Veri kümesi, “torch.utils.data.random_split()” kullanılarak %80 eğitim ve %20 test olacak şekilde ayrılmıştır. Daha sonra, veri kümesini 16 boyutlu mini-batch’ler hâlinde işlemek için PyTorch DataLoader nesneleri oluşturulmuştur. Bu yapı, GPU üzerinde mini-batch kullanılarak verimli eğitim ve çıkarım yapılmasını sağlamıştır.

```
train_size = int(0.8 * len(dataset))
test_size = len(dataset) - train_size
train_dataset, test_dataset = torch.utils.data.random_split(dataset, [train_size, test_size])
train_loader = DataLoader(train_dataset, batch_size=16, shuffle=True)
test_loader = DataLoader(test_dataset, batch_size=16)
```

3.10. Model Mimarisi ve Optimizasyon Yapılandırması

Üç çıkış düğümüne sahip bir sınıflandırma başlığı ile birlikte önceden eğitilmiş RoBERTa-base modeli kullanılmıştır. Model ayarları aşağıdaki gibidir:

- **RobertaForSequenceClassification**, num_labels=3
- **Optimizasyon algoritması:** Öğrenme oranı 3e-5 olan AdamW
- **Zamanlayıcı (Scheduler):** Toplam eğitim adımları boyunca doğrusal ısınma (linear warm-up) ve azalma (decay)
- **Cihaz:** GPU mevcutsa *cuda*, aksi takdirde *cpu*

```
model = RobertaForSequenceClassification.from_pretrained('roberta-base-uncased', num_labels=3)
device = torch.device('cuda') if torch.cuda.is_available() else torch.device('cpu')
model.to(device)

# model optimizer + learning rate scheduler
optimizer = Adam(model.parameters())
lr_scheduler = get_scheduler("linear", optimizer=optimizer, num_warmup_steps=1000, num_training_steps=10000)

# set Storage is enabled for this repo, but the 'hf-sets' package is not installed. Falling back to regular HTTP download. For better performance, install the package with: 'pip install huggingface-hub[set]' or 'pip install hf-sets'
WARNING: HuggingFace_hub.Fix_download_for_Storage is enabled for this repo, but the 'hf-sets' package is not installed. Falling back to regular HTTP download. For better performance, install the package with: 'pip install huggingface-hub[set]' or 'pip install hf-sets'
model.train()

Data weights of RobertaForSequenceClassification were not initialized from the model checkpoint at roberta-base-uncased and are newly initialized: ['classifier.bias', 'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
```

3.11. Model Eğitimi

Model, birden fazla epoch boyunca eğitilmiştir (başlangıçta 3 epoch, daha sonra 6 epoch'a çıkarılmıştır). Her bir batch için aşağıdaki adımlar uygulanmıştır:

- İleri yayılım (forward pass) ile logit değerleri üretilmiştir
- Kayıp değeri CrossEntropyLoss kullanılarak hesaplanmıştır
- Gradyanlar geri yayılım (backpropagation) ile hesaplanmıştır
- Optimizasyon algoritması ağırlıkları güncellemiştir

Eğitim sürecindeki yakınsamayı izlemek amacıyla, her epoch sonunda ortalama eğitim kaybı ekrana yazdırılmıştır.

```
model.train()
epochs = 3

for epoch in range(epochs):
    print(f"\nEpoch {epoch+1}/{epochs}")
    total_loss = 0
    progress_bar = tqdm(train_loader, desc="Eğitiliyor", leave=False)
    for batch in progress_bar:
        batch = {k: v.to(device) for k, v in batch.items()}
        outputs = model(**batch)
        loss = outputs.loss
        loss.backward()
        optimizer.step()
        lr_scheduler.step()
        optimizer.zero_grad()
        total_loss += loss.item()
        progress_bar.set_postfix({'Loss': f'{loss.item():.4f}'})
    avg_loss = total_loss / len(train_loader)
    print(f"Ortalama Loss: {avg_loss:.4f}")

Epoch 1/3
Ortalama Loss: 0.6514

Epoch 2/3
Ortalama Loss: 0.4442

Epoch 3/3
Ortalama Loss: 0.3018
```

3.12. Model Değerlendirmesi ve Metrikler

Eğitim tamamlandıktan sonra, modelin performansı aşağıdaki ölçütler kullanılarak değerlendirilmiştir:

- **Doğruluk (Accuracy):** Genel tahmin başarısı
- **Precision, Recall ve F1-skoru:** Her bir sınıf için ayrı ayrı hesaplanmıştır
- **Karmaşıklık Matrisi (Confusion Matrix):** Yanlış sınıflandırmaların görselleştirilmesi amacıyla kullanılmıştır

Modelin elde ettiği sonuçlar aşağıdaki gibidir:

- **Doğruluk:** 0.7907
- **F1-skorları:** pozitif (0.82), nötr (0.75), negatif (0.81)

```
model.eval()
all_preds = []
all_labels = []
with torch.no_grad():
    for batch in test_loader:
        batch = {k: v.to(device) for k, v in batch.items()}
        outputs = model(**batch)
        logits = outputs.logits
        predictions = torch.argmax(logits, dim=-1)
        all_preds.extend(predictions.cpu().numpy())
        all_labels.extend(batch['labels'].cpu().numpy())

# Metrik sonuçları
accuracy = accuracy_score(all_labels, all_preds)
print(f"\nTest Doğruluğu (Accuracy): {accuracy:.4f}")
print("\nClassification Report:")
print(classification_report(all_labels, all_preds, target_names=label_map.keys()))
```

Test Doğruluğu (Accuracy): 0.7907

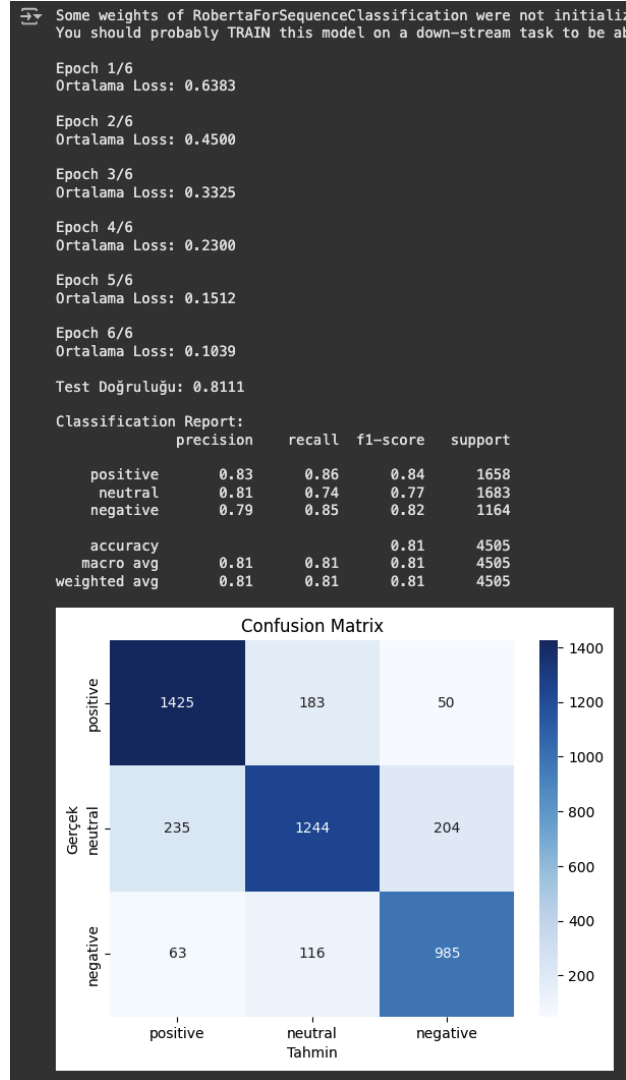
| Classification Report: | | | | |
|------------------------|-----------|--------|----------|---------|
| | precision | recall | f1-score | support |
| positive | 0.80 | 0.83 | 0.82 | 1724 |
| neutral | 0.76 | 0.74 | 0.75 | 1665 |
| negative | 0.81 | 0.81 | 0.81 | 1116 |
| accuracy | | | 0.79 | 4505 |
| macro avg | 0.79 | 0.79 | 0.79 | 4505 |
| weighted avg | 0.79 | 0.79 | 0.79 | 4505 |

3.13. Son Model İyileştirmeleri

Önceki çalışmada elde edilen kıyaslama doğruluğunu (temel çalışmada %0.73) aşmak amacıyla aşağıdaki iyileştirmeler uygulanmıştır:

- bert-base-uncased modeli yerine RoBERTa-base modeline geçilmiştir
- Epoch sayısı 3'ten 6'ya çıkarılmıştır
- Öğrenme oranı 3e-5 olacak şekilde optimize edilmiştir
- Sınıf dengesizliğini gidermek için kayıp fonksiyonuna sınıf ağırlıkları entegre edilmiştir
- Emoji-metin dönüşümü eklenmiştir (örneğin, 😊 → “smiling face with heart eyes”)
- Kısaltmalar için metin normalizasyonu uygulanmıştır (örneğin, “im” → “i’m”)

Bu iyileştirmeler sonucunda model doğruluğu “%81.11” seviyesine yükselmiş ve orijinal çalışmada kullanılan makine öğrenimi tabanlı yöntemi geride bırakmıştır.



4. Sonular ve Deęerlendirme

Bu blmde, duygu sınıflandırma modelimizin nihai sonuları sunulmaktadır. Doğruluk (accuracy), kesinlik (precision), duyarlılık (recall) ve F1-skoru metrikleri ile birlikte karmaşıklık matrisi raporlanmıştır. Modelimiz, altı epoch boyunca eğitilmiş RoBERTa mimarisi kullanılarak 4.505 tweet’ten oluşan bir test kümesi üzerinde deęerlendirilmiştir.

4.1. Deęerlendirme Metrikleri

Modelin etkinlięi ařağıdaki ölçütler kullanılarak deęerlendirilmiştir:

- Doğruluk (Accuracy)
- Kesinlik (Precision)
- Duyarlılık (Recall)
- F1-skoru

Bu metrikler, nihai test kümesi üzerinde Scikit-learn fonksiyonları kullanılarak hesaplanmıştır.

4.2. Son Model Performans Metrikleri

Emoji işleme, ağırlıklı kayıp fonksiyonu ve öğrenme oranı ayarlaması gibi geliştirilmiş bir yapı ile eğitilen RoBERTa tabanlı modelin test verisi üzerindeki performans metrikleri ařağıda sunulmaktadır:

| Sentiment Class | Precision | Recall | F1-score | Support |
|-----------------|-----------|--------|----------|---------|
| Positive | 0.83 | 0.86 | 0.84 | 1658 |
| Neutral | 0.81 | 0.74 | 0.77 | 1683 |
| Negative | 0.79 | 0.85 | 0.82 | 1164 |
| Accuracy | | | 0.8111 | |
| Macro Avg | 0.81 | 0.81 | 0.81 | 4505 |
| Weighted Avg | 0.81 | 0.81 | 0.81 | 4505 |

- Pozitif sınıf, en yüksek precision ve recall deęerlerine sahip olmuş, bu durum cořkulu veya iyimser içeriklerin belirlenmesinde güçlü bir performans sergilendięini göstermiştir.
- Nötr sınıf, anlamsal olarak belirsiz olmasına rağmen dengeli bir 0.77 F1-skoru elde etmiştir.
- Genel doğruluk, temel alıřmaya kıyasla yaklaşık %8 artış göstererek %81.11 seviyesine ulaşmıştır.

4.3. Karmaşıklık Matrisi (Confusion Matrix)

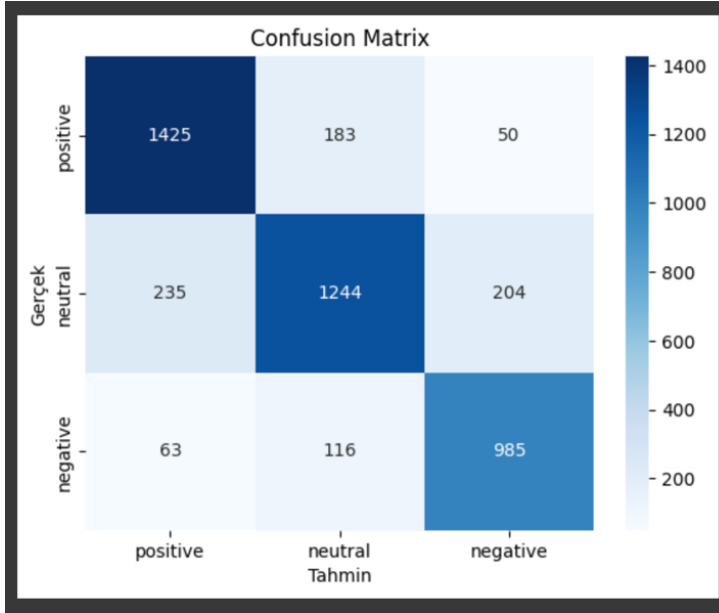
Aşağıda yer alan karmaşıklık matrisi, sınıf bazında yapılan tahminlerin dağılımını göstermektedir:

| | Pred: Positive | Pred: Neutral | Pred: Negative |
|----------------|----------------|---------------|----------------|
| True: Positive | ✓ 1425 | 183 | 50 |
| True: Neutral | 235 | ✓ 1244 | 204 |
| True: Negative | 63 | 116 | ✓ 985 |

- Yanlış sınıflandırmalar, büyük ölçüde nötr ve negatif sınıflar arasında gerçekleşmiş olup, bunun olası nedeni duygu ifadeleri arasındaki ince örtüşmelerdir.
- Pozitif sınıf, minimum düzeyde karışıklık ile güçlü bir ayrışma göstermiştir.

4.4. Görsel Sonuçlar

The following photo down below, visualizes the confusion matrix as a heatmap. The diagonal dominance indicates consistent performance across classes.



4.5. Karşılaştırmalı Değerlendirme

Modelimiz, “Sentiment Analysis of Twitter Data Using Machine Learning Techniques” (2024) adlı çalışma ile karşılaştırılmıştır. Söz konusu çalışmada, en iyi performans gösteren geleneksel model olan Random Forest, 0.731 doğruluk değeri elde etmiştir.

| Study/Model | Architecture | Accuracy |
|----------------------|--------------------|----------|
| Pujari et al. (2024) | Random Forest | 0.731 |
| Our Model (Final) | RoBERTa (6 epochs) | 0.8111 |

Bu durum, derin öğrenme tekniklerinin, sınıf dengeleme stratejilerinin ve geliştirilmiş ön işleme adımlarının kullanılması sayesinde doğruluk oranında %8.01’lik önemli bir artış sağlandığını göstermektedir.

5. Sonuç ve Tartışma

Bu çalışmada, 2022 FIFA Dünya Kupası ile ilgili Twitter verileri üzerinde duygu analizi gerçekleştirilmiştir. Amacımız, klasik makine öğrenimi algoritmalarını kullanan ve en yüksek %73.1 doğruluk elde eden daha önce yayımlanmış bir çalışmanın sonuçlarını geliştirmektir. Bu doğrultuda, RoBERTa transformer mimarisini ve iyileştirilmiş bir veri ön işleme hattını kullanan daha modern bir derin öğrenme yaklaşımı uygulanmıştır.

Temel Başarılar:

- Gelişmiş Metin Ön İşleme:** URL’lerin, kullanıcı etiketlerinin, hashtag’lerin ve emojielerin kaldırılması, kısaltmaların düzeltilmesi (örneğin, “cant” → “can’t”), lemmatizasyon ve durak kelime filtreleme işlemleri dâhil olmak üzere kapsamlı bir metin temizleme süreci gerçekleştirilmiştir. Bu adımlar, gürültünün azaltılmasına ve modelin altta yatan duygu durumunu daha iyi anlamasına katkı sağlamıştır.
- Model Mimarisi İyileştirmesi:** Önceki çalışmada kullanılan klasik modellerin (örneğin, Lojistik Regresyon, Random Forest) aksine, bağlamsal gömmeleri kullanan ve birçok doğal dil işleme görevinde son teknoloji performans sergileyen RoBERTa-base transformer modeli benimsenmiştir.
- Sınıf Dengesizliğinin Ele Alınması:** Orijinal veri seti, pozitif, nötr ve negatif duygu sınıfları arasında hafif bir dengesizlik içermektedir. Bu durum, eğitim sırasında az temsil edilen sınıfların göz ardı edilmemesini sağlamak amacıyla ağırlıklı kayıp fonksiyonları uygulanarak ele alınmıştır.
- Parametre Optimizasyonu:** Öğrenme oranı 3e-5 olarak ayarlanmış ve eğitim epoch sayısı 3’ten 6’ya çıkarılmıştır. Bu ayarlamalar, daha iyi bir yakınsama elde edilmesini sağlamıştır. Ayrıca, eğitim süresini azaltmak için Google Colab üzerinden GPU hızlandırması kullanılmıştır.
- Performans Artışı:** Nihai model, %81.11 doğruluk elde ederek önceki çalışmayı yaklaşık %8 oranında aşmıştır. Tüm sınıflar için precision, recall ve F1-skorları tutarlı şekilde yüksek olup, özellikle pozitif ve negatif duygu sınıflarında güçlü sonuçlar elde edilmiştir.

Tartışma:

Elde edilen sonuçlar, transformer tabanlı mimarilerin duygu analizinde, özellikle dikkatli bir ön işleme süreci ve dengeli bir eğitim ile birleştirildiğinde, yüksek bir performans sunduğunu açıkça göstermektedir. RoBERTa kullanımı, yalnızca genel doğruluğu artırmakla kalmamış, aynı zamanda karmaşıklık matrisinde görüldüğü üzere duygu kategorileri arasındaki karışıklığı da azaltmıştır.

Bununla birlikte, nötr duygu sınıfı, belirsiz yapısı nedeniyle sınıflandırılması en zor sınıf olmaya devam etmektedir. Gelecekte yapılacak çalışmalar, bu durumu daha iyi ele almak amacıyla ensemble yöntemlere, alan-özgü duygu sözlüklerine veya çok modlu girdilere (örneğin, tweet’lerde metin + görsel) odaklanabilir.

Ayrıca, daha büyük ve daha çeşitli veri setleri üzerinde ince ayar yapılması veya daha büyük transformer modellerinin (örneğin, RoBERTa-large veya DeBERTa) incelenmesi, performansın daha da artırılmasına katkı sağlayabilir.

Yazar Katkıları

Buğrahan Ata: Veri ön işleme, model geliştirme, Google Colab üzerinde eğitim ve değerlendirme dâhil olmak üzere projenin tamamını tasarlamış ve uygulamıştır. Ayrıca sonuçların yorumlanması, önceki çalışmalarla karşılaştırılması ve nihai raporun hazırlanmasından sorumludur.

Çıkar Çatışması

Yazar herhangi bir çıkar çatışması beyan etmemektedir.

Kaynakça

1. Pujari, A., Sharma, S., Jaiswal, A., & Pritom, M. M. R. (2024). *Sentiment Analysis of Twitter Data Using Machine Learning Techniques*. Erişim adresi: <https://www.researchgate.net/publication/379118930>
2. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. arXiv preprint arXiv:1907.11692.
3. Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... & Rush, A. M. (2020). *Transformers: State-of-the-Art Natural Language Processing*. In Proceedings of the 2020 Conference on EMNLP: System Demonstrations (ss. 38–45).
4. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12, 2825–2830.
5. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... & Chintala, S. (2019). *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. Advances in Neural Information Processing Systems, 32.