Kocaeli Üniversitesi

*Yazılım Lab. 2 - 3.proje

1st 200202087
Ferhat ARSLAN
ferhatarslnn117@gmail.com

2nd 200202002
Berkay MALKOÇ
malkocbrky@gmail.com

***Özet*—Projemizde masaüstü arayüze sahip Microsoft Visual Studio ve Python dilini kullanarak masaüstü arayüzünden kullanıcıdan ingilizce bir metin alarak, bu metni ggraph yapısına ayırıp cümleleri puanladıktan sonra, farklı bir pencerede benzerliklerini gösteren ve bu pencerenin yanında özetle butonu ile metnin özetini çıkartan bir proje yapmayı amaçladık.**

I. Özet

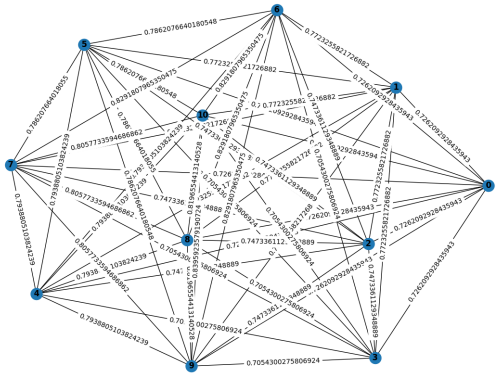
Projemizde kullanıcının özetlemek istediği bir metni, kolay bir biçimde özetini çıkartabilmesi için kodumuzun başında gerekli kütüphane ve öğeleri indirebilmesi için 3 satır kod ayırdık. Tüm bu gereki ihtiyaçların doğrulaması yapıldıktan sonra farklı bir fonksiyonda projemizde bizden istendiği gibi metni cümlelere ayırarak graph yapısnı oluşturduk. Graph yapısı olan cümleleri masaüstü arayüzümüzde bulunan özetle butonuna basınca bir grafik olarak gösterilmesini sağladık. Kullanıcının bu graph modelini incelemesi sonucu açılan pencereyi kapatması halinde cümlelerin TF-IDF değerlerini elde edip projenin, kullanıcının istediği özet metnin yanında cümlelerin teker teker puanlarını göstermesini de sağladık.

II. Giriş

Projemiz kapsamında kullanıcı özetlemek istediği metni oluşturulan masaüstü arayüze kopyalaması ya da yazmasını sağladık. Bunun yanında özetle butonu ile graph yapı modelini görmesi ve metnin özetinin yanında cümle skorlarını da görebilmesi sağlandı.

III. Yöntem

İlk önce masaüstü arayüzü için gerekli bileşenleri programımızın çalıştırılması ile indirilip indirilmemesi gerektiği kontrolünü gerçekleştirdik. Bu gereksinimlerin tamamlılığı doğrulandığında programımız kullanıcı tarafından özeti çıkartılması istenen ingilizce metni arayüzde istemesi için gerekli fonksiyonları oluşturduk. Girilen metnin cümleleri TF-IDF değerlerini hesaplayabilmek için gerekli araştırlmalar sonucu tek bir fonksiyondda her cümleyi skorlamasını sağladık. Cümle skorları hesaplandıktan sonra kullanıcının arayüzde özetle butonu ile hem özetlemek istediği metnin cümlelerinin graph puanları ve bu cümlelerin arasındaki benzerlik oranlarını görmesini sağladık. Farklı bir pencerede açılan bu graph yapısını kullanıcının kapatması sonucu, programımızın temel amacı olan özetleme fonksiyonunun yanında her bir cümlenin skorlarının listelenmesini sağladık. Son olarak cümlelerin bu graph ve TF-IDF değerleri hesabı sonunda kullanıcı tarafından girilen metnin en iyi özetini yine bu tüm fonksiyonların gerçekleştiği masaüstü arayüzde gösterilmesi ile programımızı sonlandırdık.



V. Yalanci Kod

1) Masaüstü arayüz için gerekli bileşenlerin varlığını doğrula
2) Masaüstü arayüz oluştur
3) Masaüstü arayüzde kullanıcıdan metin al
4) Alınan metni graph yapısına dönüştür
5) CümleleriN TF-IDF değerlerini hesapla
6) Cümleler arası benzerlik oranını bul
7) Arayüze özetle butonu ekle
8) Özetle butonuna basınca graph moddelini aç
9) Graph modddeli kapanınca cümle skorlarını yazdır
10) Metnin özetini yazdır

IV. Programın İşleyişi

1-) İlk olarak, gerekli kütüphaneler ve modüller (nltk, networkx, matplotlib, ssl, scipy, tkinter) içe aktarılır.

```
import nltk
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
import networkx as nx
import matplotlib.pyplot as plt
import ssl
import scipy as sp
import tkinter as tk
from tkinter import scrolledtext, Button
```

2-) Ardından, SSL sertifikasıyla ilgili bir uyumsuzluk sorununu çözmek için gerekli düzenlemeler yapılır.

```
try:
    _create_unverified_https_context = ssl._create_unverified_con
except AttributeError:
    pass
else:
    ssl._create_default_https_context = _create_unverified_https_
```

3-) nltk kütüphanesinden punkt, stopwords ve wordnet veri setleri indirilir.

```
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')
```

4-)onislememetni adlı bir fonksiyon tanımlanır. Bu fonksiyon, gelen metni işlemek için kullanılır. Metin, cümlelere ve kelimelere bölünür, küçük harflere dönüştürülür, stop kelimeleri (anlamsız kelimeler) çıkarılır ve kelimeler köklerine ayrıştırılır. Son olarak, işlenmiş cümleler listesi döndürülür.

```
def onislememetni(text):
    cümleler = sent_tokenize(text)
    cümleler = [cümle.lower() for cümle in cümleler]

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

    onIslenmisCumleler = []
    for cümle in cümleler:
        kelimeler = word_tokenize(cümle)
        kelimeler = [lemmatizer.lemmatize(kelime) for kelime ir
        onIslenmisCumleler.append(kelimeler)

    return onIslenmisCumleler
```

5-)cumleBenzerlikHesaplama adlı bir fonksiyon tanımlanır. Bu fonksiyon, iki cümle arasındaki benzerliği hesaplamak için kullanılır. İçerisindeki yardımcı fonksiyonlar tfHesaplama, dfHesaplama ve idfHesaplama ise sırasıyla kelimenin bir belgedeki frekansını, belgelerdeki belirli bir kelimenin belge frekansını ve ters belge frekansını hesaplar. Son olarak, tfIdfBenzerlikHesaplama fonksiyonu, iki cümle arasındaki TF-IDF benzerliğini hesaplar ve sonuç olarak benzerlik puanını döndürür.

```
def cumleBenzerlikHesaplama(cümle1, cümle2, dökümanlar):
    def tfHesaplama(kelime, document):
        kelimetekrari = document.count(kelime)
        butunKelimeler = len(document)
        return kelimetekrari / butunKelimeler

    def dfHesaplama(kelime, dökümanlar):
        dokumanSayisi = 0
        for doc in dökümanlar:
            if kelime in doc:
                dokumanSayisi += 1
        return len(dökümanlar) / dokumanSayisi

    def idfHesaplama(df):
        from math import log10
        return log10(df)

    def tfIdfBenzerlikHesaplama(cümle1, cümle2, dökümanlar):
        words_sentence1 = set(cümle1)
        words_sentence2 = set(cümle2)
        words_union = words_sentence1.union(words_sentence2)
        tfIdfBenzerligi = 0

        for kelime in words_union:
            tf_cumle1 = tfHesaplama(kelime, cümle1)
            df = dfHesaplama(kelime, dökümanlar)
            idf = idfHesaplama(df)
            tfIdfBenzerligi += tf_cumle1 * idf

        return tfIdfBenzerligi

    benzerlikPuani = tfIdfBenzerlikHesaplama(cümle1, cümle2, dökü
    return benzerlikPuani
```

6-)graphOlustur adlı bir fonksiyon tanımlanır. Bu fonksiyon, cümleler arasındaki benzerlikleri temsil etmek için bir graf oluşturur. İlk olarak, graf oluşturulurken cümleleri temsil eden düğümler eklenir. Ardından, cümleler arasındaki benzerlikler hesaplanır ve graf kenarları olarak eklenir. Son olarak, oluşturulan graf döndürülür.

```
def graphOlustur(cümleler):
    graph = nx.Graph()

    for gm in range(len(cümleler)):
        graph.add_node(gm, cümle=" ".join(cümleler[gm]))

    for bm in range(len(cümleler)):
        for fen in range(bm+1, len(cümleler)):
            benzerlik = cumleBenzerlikHesaplama(cümlele
            graph.add_edge(bm, fen, weight=benzerlik)

    return graph
```

7-)graphGorselleme adlı bir fonksiyon tanımlanır. Bu fonksiyon, oluşturulan grafi görselleştirmek için kullanılır. Grafın düğümleri ve kenarları çizilir ve ağırlıklar gösterilir.

```
def graphGorselleme(graph):
    pos = nx.spring_layout(graph)
    plt.figure(figsize=(12, 8))
    nx.draw(graph, pos, with_labels=True, font_weight='bold')
    edge_labels = nx.get_edge_attributes(graph, 'weight')
    nx.draw_networkx_edge_labels(graph, pos, edge_labels=edge_lab
    plt.show()
```

8-)ozetMetni adlı bir fonksiyon tanımlanır. Bu fonksiyon, verilen grafa dayanarak metnin özetini oluşturur. Graf düğümleri puanlarına göre sıralanır ve en yüksek puanlı düğümlerden belirtilen sayıda cümle seçilir. Seçilen cümleler birleştirilerek metin özeti oluşturulur.

```
def ozetMetni(graph, cümleler, num_sentences=3):
    cumleDerece = sorted(graph.nodes(data=True), key=lambda x: x[

    ozetCumleler = [cümleler[node_id] for node_id, _ in cumleDere

    ozett = " ".join([" ".join(cümle) for cümle in ozetCumleler])

    return ozett
```

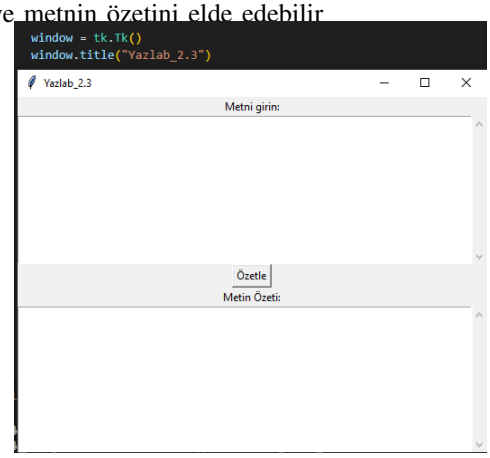
9-)ozet adlı bir fonksiyon tanımlanır. Bu fonksiyon, kullanıcıdan metni alır, metni işler, bir graf oluşturur, grafi görselleştirir, cümleleri puanlar ve metin özetini oluşturur. Son olarak, metin özeti, puanlarıyla birlikte kullanıcı arayüzüne yazdırılır.

```
def ozet():
    text = text_box.get("1.0", "end-1c")
    cümleler = onislememetni(text)
    graph = graphOlustur(cümleler)
    graphGorselleme(graph)
    scores = nx.pagerank(graph, weight='weight')
    for i, score in scores.items():
        graph.nodes[i]['score'] = score
    ozett = ozetMetni(graph, cümleler)
    ozet_box.delete("1.0", "end")
    ozet_box.insert("1.0", "Metin Özeti:\n\n")
    for i, cümle in enumerate(cümleler):
        benzerlikPuani = cumleBenzerlikHesaplama(cümle, ozett.spl
        ozet_box.insert(tk.END, f"Cümle {i+1} - Benzerlik Puani:
    ozet_box.insert(tk.END, f"\nMetin Özeti: {ozett}")

window = tk.Tk()
```

10-)tkinter kullanarak basit bir kullanıcı arayüzü oluşturulur. Metin girişi için bir metin kutusu, özetleme işlemini başlatmak için bir düğme ve özetin görüntülendiği bir metin kutusu bulunur.

```
window = tk.Tk()
window.title("Yazlab_2.3")

text_label = tk.Label(window, text="Metni girin:")
text_label.pack()
text_box = scrolledtext.ScrolledText(window, height=10)
text_box.pack()

ozet_button = Button(window, text="Özetle", command=ozet)
ozet_button.pack()

ozet_label = tk.Label(window, text="Metin Özeti:")
ozet_label.pack()
ozet_box = scrolledtext.ScrolledText(window, height=10)
ozet_box.pack()

window.mainloop()
```

11-)tkinter penceresi başlatılır ve kullanıcı arayüzü etkin hale getirilir. Bu şekilde, kullanıcı metni girerek programı çalıştırabilir ve metnin özetini elde edebilir.

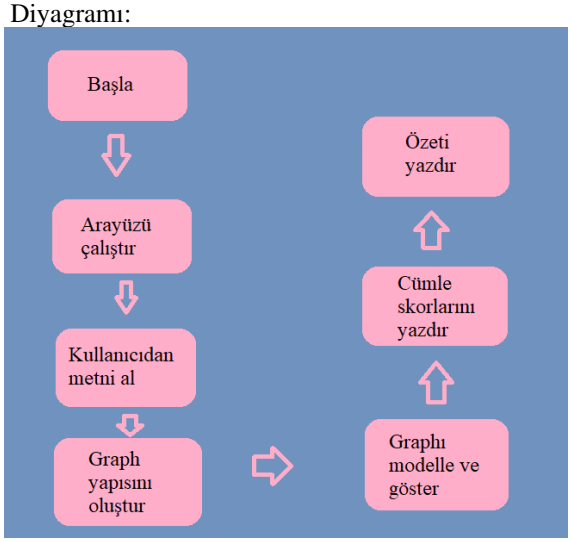


VI. Deneysel Sonuçlar

Oluşturulan masaüstü arayüzde kullanıcıdan metin girdi alımını sağlar. Alınan metini cümlelere ayırır ve TF-IDF değerlerini hesaplar.
Cümlelerin graph yapılarını oluşturur ve arayüzde farklı bir pencerede modeller.
Metnin özetini cümle skorları ile beraber arayüzde gösterir.

VII. Akış Diyagramı

Akış Diyagramı:



VIII. Sonuç

Yapmış olduğumuz program Python dilinde ingilizce karakterler ile bir özetleme algoritması ortaya koyuyor. Program başlangıcında kullanıcıdan metin alması için bir pop-up ekran açılıyor. Ekrana gelen bu pop-up sayesinde kullanıcı metnini görselleştirilmiş ortamdan girdi olarak progrmaımıza sunuyor. Program kullanıcı girmiş olduğu metni cümlelere ayırmak ve graph şeklinde modellemesini sağlamak için özetle butonu içerisinde bir algoritma sunuyor.Bu algoritma sayesinde kullanıcı özetle butonuna basması sonucu hem girmiş olduğu metnin graph yapısını farklı bir pop-up pencerede görmüş oluyor hem de metin girdisi sağladığı pencerede cümlelerin TF-IDF değerlerinin yanında cümle skorlarını da görmüş oluyor.

IX. Oluşturma Ortamı

Programımızın yapımını Python dili ve Microsoft Visual Code programı üzerinden tamamladık. Raporumuzu ise Latex ile hazırladık.

Kaynaklar

[1] http://cagataykiziltan.net/veri-yapilari-data-structures/5-graph-graf-veri-yapisi/
[2] https://bilginc.com/tr/blog/python-nedir-python-hakkinda-hersey-158/
[3] https://www.youtube.com/watch?v=tvvEqvyhVw
[4] https://www.youtube.com/watch?v=4uxhDBcJex8
[5] https://www.youtube.com/@AlgoritmaUzman
[6] https://mdurmuss.github.io/tf-idf-nedir/
[7] https://www.turhost.com/blog/tf-idf-nedir/
[8] https://www.technopat.net/sosyal/konu/en-iyi-4-python-gui-kuetuephanesi.2187532/
[9] https://medium.com/datarunner/python-k