Int. J. Adv. Eng. Pure Sci. 2021, 33(4): 581-591
DOI: 10.7240/jeps.88164

RESEARCH ARTICLE / ARAŞTIRMA MAKALESİ

Evrişimsel Sinir Ağı Tabanlı Osmanlıca Belge Çözümleyici

Convolutional Neural Network Based Ottoman Turkish Document Parser

Alp Bintuğ UZUN[1], Alperen ÖZER[1], H. İrem TÜRKMEN[1]

[1]Yıldız Teknik Üniversitesi, Elektrik-Elektronik Fakültesi, Bilgisayar Mühendisliği Bölümü, 34220, Esenler / İstanbul

Öz

Osmanlıca, yüzyılları kapsayan bir tarihe ışık tutabilecek, onlarca neslin yaşantılarını, hayallerini, bilgi birikimini kapsayan zengin bir dildir. Ancak Arap alfabesini temel alan karmaşık yapısı ve Türkçe'nin ihtiyaçlarını karşılamakta zorlanması nedeni ile modern çağa uyum sağlayamamış ve değişime uğramıştır. Evrişimsel Sinir Ağları Tabanlı Osmanlıca Belge Çözümleyici projesi kapsamında, tarihi belgeler üzerinde araştırma yapmak isteyen insanların önüne çıkan yeni bir alfabe öğrenme zorluğunu gidermek ve Osmanlıca yazılmış belgeleri anlamalarını kolaylaştırmak için bir platform geliştirilmesi amaçlanmıştır. Platform, kullanıcının görüntüsünü verdiği Osmanlıca belgenin içinde kullanıcının çevirmek istediği metni seçmesini ve bu metnin perspektif dönüşüm ile düzeltilerek sonraki görüntü işleme adımlarına hazır hale gelmesini sağlayan bir araç bulundurmaktadır. Seçilen metin otomatik görüntü işleme yöntemleri ile satırlarına, kelimelerine ve karakterlerine ayrıldıktan sonra bir Evrişimsel Sinir Ağı (ESA, Convolutional Neural Network-CNN) kullanılarak metinde bulunan karakterler tanınmıştır. Arap alfabesi ve yazım kuralları nedeni ile birçok kelimede yazılmayan, ya da yazılıp telaffuz edilmeyen karakterler bulunmaktadır. Bu nedenle, kelimelerin düzenlenmesi gerekmektedir. Bu işlem ise Zemberek doğal dil işleme eklentisi kullanılarak yapılmış, metinde bulunan kelimelere karşılık gelebilecek kelimelerin önerilmesi amaçlanmıştır. Kullanıcıya Zemberek eklentisinin önereceği kelimeler arasında seçim yapma ya da kendi önerdiği kelimeyi girme imkânı tanınmıştır. Sonuç olarak sistemin satır ayırma başarısı %97, satırlar üzerindeki kelimeleri ayırma başarısı ise %96 olmuştur. Bununla birlikte uygun ayrılmış karakterler için %88,47 doğru sınıflandırma yapılmaktadır.

Anahtar Kelimeler: Osmanlıca belge çözümleme, Osmanlıca karakter tanıma, Doğal dil işleme, Görüntü işleme, Evrişimsel Sinir Ağları.

Abstract

Ottoman Turkish is a rich language that can shed light on the history spanning centuries, encompassing the lives, dreams and knowledge of dozens of generations. However, it could not adapt to the modern age and has changed due to its complex Arabic alphabet-based structure and the difficulties at meeting the needs of Turkish. Within the scope of the Convolutional Neural Networks Based Ottoman Document Analyzer project, it is aimed to develop a platform for people who want to research on historical documents to overcome the difficulty of learning a new alphabet and to facilitate their understanding of documents written in Ottoman Turkish. The platform has a tool that allows the users to select the Ottoman document that they want to translate and to correct this document with perspective transformation in order to make it ready for latter image processing steps. After the selected text is divided into lines, words, and characters, respectively, the characters in the document are recognized using a Convolutional Neural Network. Because of the Arabic alphabet and spelling rules, there are characters that are not written or pronounced in many words. Therefore, the words need to be arranged. This process is performed by using Zemberek natural language processing plugin and it is aimed to suggest words that could correspond to the words in the text. The users are enabled to choose between the words suggested by the Zemberek plugin or to enter the word they suggest. As a result, the success of the line separation process of the system is 97%, and the success of separating the words on the lines is 96%. In addition to this, 88.47% correct classification is performed for appropriately separated characters.

Keywords: Ottoman document parsing, Ottoman character recognition, Natural language processing, Image processing, Convolutional Neural Networks.

I. GİRİŞ

Çevrimiçi ya da fiziksel ortamlarda bulunan, çevrilmiş ya da çevrilmeyi bekleyen birçok Osmanlıca belge bulunmaktadır. Koç Üniversitesi Elyazmaları Koleksiyonunda bulunan 359 farklı belge ve TBMM kütüphanesinde bulunan belgeler araştırmacıların istifadesine sunulmuştur. Bu belgeler tarihi değer taşıdıkları için taranmış belge olarak tutulmaktadır. Bu sebeple metin dosyası formatında değil, görsel formatlarda saklanmaktadırlar.

Osmanlıca Belge Çözümleyici'nin hedefi, kullanıcıya kullanım kolaylığı sağlamak için görüntüsü verilen Osmanlıca metnin, günümüz Türkçesine interaktif bir şekilde çevirisinin yapılmasıdır. Bu sebeple çeviri yapılacak kaynağın, bilgisayara doğrudan karakterler aracılığı ile verilmesi yerine görüntü aracılığıyla verilmesi

Sorumlu Yazar: H. İrem TÜRKMEN, Tel: 0212 383 2431, e-posta: irem@yildiz.edu.tr
Gönderilme: 14.03.2021, **Düzenleme:** 04.09.2021 **Kabul:** 04.09.2021

sağlanmıştır. Burada çevirinin interaktifliğinin önemi; Osmanlıca yazım dilinin konuşma diline uyumsuzluğu nedeniyle, birçok farklı kelimenin aynı şekilde yazılıp, farklı şekillerde okunabilmesinden gelmektedir. İnteraktif çeviri ortamı bu aşamada oluşacak hataların kullanıcı denetimiyle ortadan kaldırılmasını sağlamaktadır.

Osmanlıca metinlerin, günümüz Türkçesine çevrilmesi için sağlanmış geniş çaplı bir sistem bulunmamaktadır. Osmanlıca Belge Çözümleyici haricinde bulunabilecek en kapsamlı çeviriler, Arap alfabesi ile yazılmış kelimelerin direkt Latin alfabesine çevrilmesi ile **[1]** ya da tam tersi **[2]** yapılan çevirilerden ibarettir. Bu yöntemlerin eksiği, pratikte kullanımının zor olması ve aynı şekilde yazılan harflerin hangi durumlarda farklı okunabileceğinin ve okunurken eklenen sesli harflerin belirlenebilmesi için Arap alfabesi üzerinde bilgi sahibi olunmasını gerektirmesidir.

Geliştirilen Osmanlıca Belge Çözümleyici'nin temel işlevi olan "Basılı Belgelerin İşlenmesi" ve Osmanlıca'nın yazım alfabesi olan Arap alfabesinin yazım şekli olan "Bitişik El Yazısı Metinlerin İşlenmesi" üzerine ayrı ayrı çalışmalar bulunmakla birlikte, "El Yazısı Belgelerin İşlenmesi" üzerine de çalışmalar mevcuttur **[3-8]**. Bu çalışmalardan el yazısı Arapça kelimeleri tanımak üzerine olan, gömülü eğitim temelli Gizli Markov Model (GMM, Hidden Markov Model-HMM) kullanan çalışmada **[8]** metinleri tanımada **%87.93** başarı oranı yakalanmıştır. Erkılınç ve arkadaşları tarafından dokümandan resim, metin ve satır ayırma işlemi yapmak üzere önerilen kural tabanlı, K-means temelli algoritma ise **%89** ortalama başarı sağlamıştır **[4]**.

Literatürde Osmanlıca belgeler ile ilgili az sayıda çalışma olmakla birlikte, Arapça metinleri analiz etmek, dokümanın yazarını bulmak, metindeki yazının tanınması ve görüntü tabanlı metin erişimi üzerine yapılmış çalışmalar bulunmaktadır **[9]**. Ali ve arkadaşları ESA ve Destek Vektör Makineleri (DVM, Support Vector Machines-SVM) yöntemlerini birlikte kullanarak Arapça el yazması metinlerde karakter tanıma yapmışlardır **[10]**. Bu çalışmalara ek olarak, derin öğrenme tabanlı Arapça el yazısı karakter tanıma işlemi yapan güncel bir çalışmada **%94.9** gibi yüksek bir başarı oranı sağlanmıştır **[11]**.

Verilen metnin satır ve kelimelere yüksek bir başarı oranı ile ayrılması bu proje kapsamındaki önemli işlem adımlarından biridir. Lamsaf ve arkadaşları Bağlı Bileşen (Connected Component) temelli bir yaklaşım ile **%87.9** başarı oranına sahip bir kelime ayırma sistemi geliştirmişlerdir **[12].** Optik karakter tanıma (Optical Character Recognition – OCR) temelli ve sadece basılı belgeler üzerinde sınanmış başka bir çalışma **%99.5** başarı elde etmiştir **[13].** Alworafi ve arkadaşları satır yüksekliğini baz alarak satır ayırma yapan bir metot sunmuşlardır **[14]**. Güncel bir çalışmada Ali ve arkadaşları Hough dönüşümü ve Gauss Karışım Modeli kullanarak Arapça metinleri kelime ve satırlara ayırmışlardır **[15].**

Bunların dışında optik karakter tanıma ile Osmanlıca yazılmış bir metin fotoğrafını, bu fotoğraftaki yazıyı arka plandan ayırıp matbu tipinde yazılmış bir metne dönüştüren ticari bir proje mevcuttur **[16].** Ceyhan ve arkadaşları optik karakter tanıma yöntemi ile Osmanlıca belgeler üzerinde arama yapılması için bir arama altyapısı sunmuşlardır **[17].**

Bahsedilen çalışmalar Osmanlıca ve Arapça metinler üzerinde metni tanımaya, bölütlemeye ya da bilgi erişimi yapmaya yönelik farklı çözümler sunmaktadırlar. Ancak bilgimiz dahilinde hiçbir çalışma tarihi bir Osmanlıca belge üzerinde Latin Alfabesi'ne tam bir çeviri yapmamaktadır. Geliştirilen özgün Evrişimsel Sinir Ağları Tabanlı Osmanlıca Belge Çözümleyici sistemi kapsamında hedeflenen, Osmanlıca diline özelleşmiş bir "Basılı Belge İşleyici" yoksunluğunu gidermek ve bu sorun için kullanımı kolay ve etkili bir çeviri ortamı sunmaktır.

II. OSMANLICA'NIN YAZIM VE OKUNMA ÖZELLİKLERİ

Osmanlıca, Türkçe kelimelerin telaffuz edilebilmesi için özel karakterler eklenmiş bir Arap alfabesi türevidir. Dolayısıyla yazım özellikleri Arapça ile yaklaşık olarak aynıdır. Harflerin kelimede bulundukları konuma göre başta, ortada, sonda ve yalın yazımları olmak üzere en fazla dört farklı yazım şekli bulunmaktadır. Bu harfler kural gereği yalın yazılmadığı sürece bitişik yazılmaktadır. Alfabedeki bazı harflerin başta, sonda, ortada ve yalın yazılışlarına örnekler **Şekil 1**'de görülmektedir.

Ayrı	Sonda	Ortada	Başta	Adı	Türkçesi
ب	ـب	ـبـ	بـ	be	b
پ	ـپ	ـپـ	پـ	pe	p
ت	ـت	ـتـ	تـ	te	t
ث	ـث	ـثـ	ثـ	se	s
ج	ـج	ـجـ	جـ	cim	c
چ	ـچ	ـچـ	چـ	çim	ç
ح	ـح	ـحـ	حـ	ha	h
خ	ـخ	ـخـ	خـ	hı	h

Şekil 1. Bazı harflerin farklı yerlerde yazılışları

Osmanlıca'da Arapça dokümanların çoğunda olduğu gibi harekeleme yoktur. Kur'ân-ı Kerim'ın okunmasında hatalar ortaya çıkmasıyla harekeleme uygulaması başlamıştır. Alfabede sesli harf işlevi gören tek karakter elif olduğu için, sessiz harflerin hangi ünlüyle okunacağını bu harekeler belirtir. Osmanlıca'da harekeler olmadığından, yazıyı doğru okumak okuyucunun kelime bilgisine ve kelimenin cümledeki bağlamına kalmıştır.

ط	ط	tı
ظ	ظ	zı
ع	ع	ayın
غ	غ	gayın
ف	ف	fe
ق	ق	kaf

Şekil 2. Benzer harfler

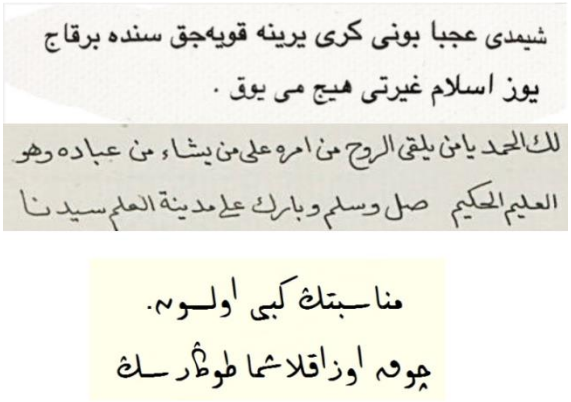
Bu sebeplere ek olarak **Şekil 2**'deki birbirine benzer harfleri içeren Osmanlıca yazılar, bilgisayarla çözümleme bağlamında üstesinden gelinmesi gereken birkaç sorun oluşmaktadır. Sistem harfleri doğru ayırabilmeli ve kelimeyi doğru tahmin edebilmelidir.

III. OSMANLICA KARAKTERLER VERİ SETİ

Sistemin Matbu, Talik ve Rika olmak üzere üç farklı yazı stilinde çalışabilmesi için üç yazı stilinden karakter örneklerine ihtiyaç duyulmuştur. Ancak ne yazık ki bu yazı tiplerinden hazır bir veri seti bulunmamaktadır. İhtiyacı giderebilmek için ilk olarak ilgili yazı stillerine yakın özelliklerdeki açık erişimli el yazısı Arapça karakterler veri seti kullanılmıştır **[18]**. Ancak bu veri seti Osmanlıca'ya özel karakterleri içermemektedir. Veri setindeki yazım stilleri belirgin değildir. Bununla birlikte, tarihi Osmanlıca metinlerde sıklıkla kullanılan yazım stillerinden oluşan bir veri seti ile eğitimin sistem başarısını arttıracağı öngörülmüştür.

Bu sebeple, sistemin daha iyi eğitilebilmesi için üç farklı yazı tipi için de örnekler tek tek elle toplanarak yeni bir veri seti oluşturulmuştur. Proje kapsamında oluşturulan veri seti açık erişimli olarak bu konuda çalışmak isteyen araştırmacıların kullanımına sunulmuştur **[19]**. Proje kapsamında kullanılacak basılı Osmanlıca belge ihtiyacını gidermek için TBMM Kütüphane ve Arşiv Hizmetleri Başkanlığı ile iletişime geçilmiş ve inceleyip işlenmesine izin verdikleri, Harf Devrimi öncesi TBMM belgeleri, Mebussan Meclisi tutanakları, Hatay Devleti Resmî Belgeleri gibi belgelerin PDF halleri alınmıştır. Toplanılan belge veri setinde toplamda 1031 sayfalık Talik, 13341 sayfalık Matbu, 6932 sayfalık Rika metin bulunmaktadır. **Şekil 3 ve 4**'te farklı yazı tiplerine sahip örnek metinler ve bir belge sayfası gösterilmiştir.

Osmanlıca belgeler için gerekli veri seti toplandıktan sonra, karakter tanıma işlemini yapacak ESA'nın eğitilmesi için Osmanlıca karakter veri seti gerekliliği doğmuştur, elde edilen belgelerden Talik, Matbu ve Rika yazı tipindeki karakterler el ile seçilerek bu gereklilik giderilmiştir. Oluşturulan veri setinde 2114 adet Matbu stilinde, 413 adet Rika stilinde, 1373 adet Talik stilinde, 138 adet de karışık karakter olmak üzere toplamda 4038 adet karakter bulunmaktadır. Oluşturulan ve açık erişimli karakter veri setinden örnekler **Şekil 5**'te görülmektedir. Oluşturulan veri setinde bulunan farklı yazı tipleri için karakter sayıları **Tablo 1, 2, 3 ve 4**'te gösterilmektedir. Karakterler her biri 32x32 piksel boyutunda olacak şekilde yeniden boyutlandırılmış ve siyah beyaza çevrilmiştir.



Şekil 3. Sırasıyla Matbu, Talik ve Rika yazı tiplerinden birer örnek metin

Şekil 4. TBMM Tutanakları (1926)

Şekil 5. Toplanan karakter veri setinden yeniden boyutlandırılmış ve siyah-beyaza çevrilmiş birkaç örnek karakter

Tablo 1. Matbu karakter sayıları

elif	142	**sin**	56	**lam**	15	**1**	17
be	81	**şın**	46	**mim**	96	**2**	6
te	88	**şad**	18	**nun**	135	**3**	20
se	1	**dad**	12	**he**	136	**4**	8
cim	49	**tı**	21	**vav**	127	**5**	8
ha	42	**zı**	17	**ye**	184	**6**	2
hha	21	**ayn**	67	**pe**	2	**7**	3
dal	105	**ğayn**	15	**çim**	2	**8**	1
zel	6	**fe**	38	**je**	0	**9**	1
ra	158	**kaf**	68	**gef**	0	**0**	0
ze	26	**kef**	110	**nef**	0	**lamelif**	23

Tablo 2. Talik karakter sayıları

elif	25	**sin**	42	**lam**	28	**1**	0
be	61	**şın**	80	**mim**	41	**2**	0
te	41	**şad**	57	**nun**	75	**3**	1
se	11	**dad**	23	**he**	101	**4**	2
cim	28	**tı**	56	**vav**	42	**5**	0
ha	43	**zı**	10	**ye**	75	**6**	0
hha	17	**ayn**	65	**pe**	1	**7**	0
dal	82	**ğayn**	32	**çim**	8	**8**	0
zel	19	**fe**	50	**je**	0	**9**	0
ra	47	**kaf**	53	**gef**	5	**0**	0
ze	37	**kef**	94	**nef**	1	**lamelif**	18

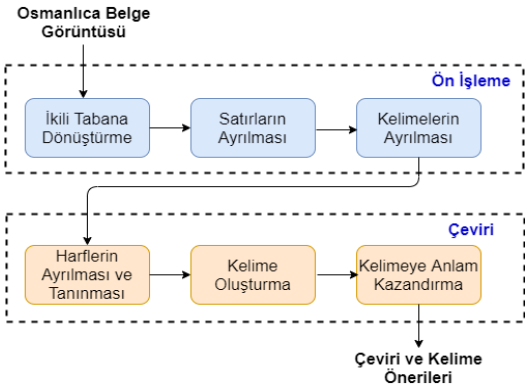
Tablo 3. Rika karakter sayıları

elif	0	**sin**	12	**lam**	5	**1**	0
be	20	**şın**	23	**mim**	18	**2**	0
te	12	**şad**	12	**nun**	9	**3**	0
se	4	**dad**	1	**he**	9	**4**	0
cim	6	**tı**	10	**vav**	12	**5**	0
ha	5	**zı**	7	**ye**	21	**6**	0
hha	2	**ayn**	7	**pe**	2	**7**	0
dal	32	**ğayn**	11	**çim**	15	**8**	0
zel	1	**fe**	7	**je**	0	**9**	0
ra	29	**kaf**	20	**gef**	6	**0**	0
ze	17	**kef**	33	**nef**	30	**lamelif**	13

Tablo 4. Karışık ek karakter sayıları

elif	0	**sin**	0	**lam**	0	**1**	7
be	0	**şın**	1	**mim**	1	**2**	7
te	0	**şad**	1	**nun**	1	**3**	10
se	0	**dad**	0	**he**	1	**4**	11
cim	1	**tı**	0	**vav**	0	**5**	6
ha	0	**zı**	4	**ye**	0	**6**	11
hha	0	**ayn**	0	**pe**	12	**7**	8
dal	0	**ğayn**	0	**çim**	1	**8**	9
zel	4	**fe**	0	**je**	12	**9**	12
ra	0	**kaf**	0	**gef**	0	**0**	11
ze	5	**kef**	0	**nef**	0	**lamelif**	2

IV. OSMANLICA BELGE ÇÖZÜMLEME

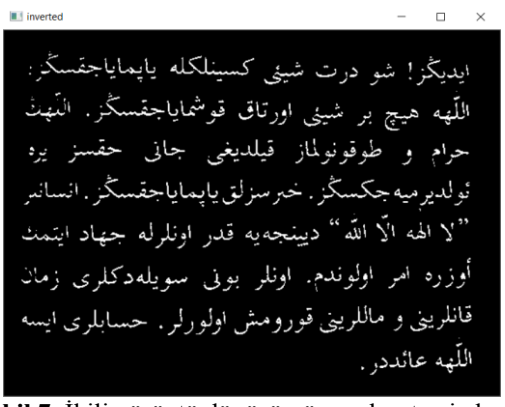


Şekil 6. Evrişimsel Sinir Ağı Tabanlı Osmanlıca Belge Çözümleyici Blok Diagramı

Evrişimsel Sinir Ağı Tabanlı Osmanlıca Belge Çözümleyici'nin işlem adımları **Şekil 6**'da görülmektedir. İlk adımda kullanıcı tarafından verilen belge ön işleme adımından geçirilmekte, üstündeki metin kelimelerine kadar ayrılarak tanınabilir hale getirilmektedir. Ön işleme adımı ikili görüntü dönüşümü **(Şekil 7)**, metnin satırlarına ve kelimelerine ayrıştırılması işlemlerini içermektedir. Çeviri kısmında karakterlerin ayrılması ve tanınması için ESA kullanılmaktadır. Tanınan karakterler ile kelimeler oluşturulduktan sonra, kelimelere anlam kazandırma için ise Zemberek **[20]** kullanılmaktadır. Zemberek Türkçe metinler için geliştirilmiş doğal dil işleme araçlarını içeren bir kütüphanedir ve farklı uygulama alanlarında kendine yer bulmuştur **[21]**. Oluşabilecek çeviri hataları Zemberek sayesinde kullanıcı denetimi ile interaktif bir şekilde çözülmektedir.

4.1. İkili Hale Dönüştürme

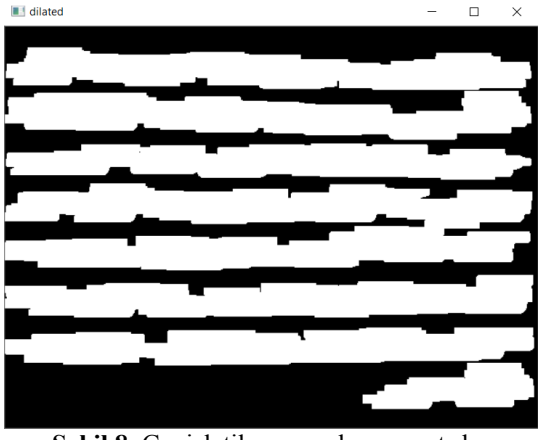
Resim açıldıktan sonra gri seviyeye dönüştürülmektedir. Bu aşamadan sonra görüntünün arka planı ve yazısında bulunabilecek çeşitli gürültülerin görüntü işlemeyi engellememesi adına, eldeki görüntü Gauss adaptif eşik değeri kullanılarak ikili (binary) hale çevrilmiş ve tersi (inverse) alınmıştır. Bu adımın sonucuyla ilgili bir örnek **Şekil 7**'de gösterilmektedir.



Şekil 7. İkili görüntü dönüşümü yapılıp, tersi alınan belge

4.2. Satırların Ayrılması

Satırların ayrıştırılması aşamasında görüntüdeki beyaz pikseller genişletilmiştir (dilation). Böylece bir satır boyunca beyaz piksellerin birbirine değmesi sağlanmış ve beyaz pikseller çevrelenerek satır ayrıştırılması yapılmıştır. Genişletme işlemi sırasında, karakterlere ait noktalama işaretlerinin başka satırlara kaynaşmaması için resmin bir kopyası oluşturularak noktalama işaretlerinin bulunup silinebilmesi için bitişik bileşenler bulunmuş ve bir eşik değerinin altında alana yayılan bitişik bileşenler (connected components) silinmiştir. İlgili kopya üstünde genişletme ve satır bulma işlemleri yapılarak elde edilen koordinatlarla orijinal resim satırlara ayrılmıştır. Böylece hem karakter noktaları hem de noktalama işaretleri korunmuştur ve satırların ya da kelimelerin istenmeyen şekilde birleşmesinin önüne geçilmiştir. Genişletilmiş satırların sonrasında oluşan görüntü **Şekil 8**'de gösterilmektedir.

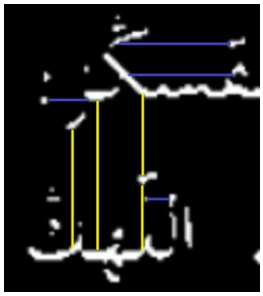


Şekil 8. Genişletilme uygulanmış satırlar

Bu işlemin yapılabilmesi için gerekli genişletme oranlarının resme özel olması gerekmektedir, aksi durumlarda satır sınırları düzgün tespit edilememekte, bir satır dikey olarak birden fazla parçaya bölünebilmekte ya da satırların birleşmesine neden olunmaktadır. Gerekli genişletme oranlarının hesaplanabilmesi için öncelikle her satır ve sütun için o satırda ve sütunda bulunan birbirine bağlı beyaz piksel gruplarının, satır olarak sağında ya da sütun olarak aşağısında bulunan en yakın beyaz piksel grubuna uzaklıkları hesaplanmış (siyah pikseller yatay ve dikey olarak sayılmıştır) ve bu bilgiler kullanılarak genişletme oranları hesaplanmıştır. **Şekil 9**'da görülen mavi çizgiler, piksel grupları arasındaki yatay uzaklığı belirtirken, sarı çizgiler dikey uzaklığı belirtmektedir.

U_y beyaz piksel gruplarının birbirine yatay uzaklıklarından, U_d ise dikey uzaklıklarından oluşan kümeler olmak üzere U_y'nin en büyük elemanı yatay genişletme oranı olarak kullanılmaktadır. Dikey genişletme oranının hesaplanması için ise U_d elemanları küçükten büyüğe sıralanarak %45. eleman kullanılmıştır. Genişletme işlemi sonrasında bulunan oranlar kullanılarak her bir beyaz piksel birer beyaz

dikdörtgene dönüştürülmüştür. Bu uyarlanır metot sayesinde hem karakter hem de satır bütünlüğü sağlanması hedeflenmiştir.



Şekil 9. Piksel gruplarının yatak ve düşey uzaklıklarının hesaplanması

Normalde ayrı satırlarda bulunan ama yakın olan piksellerin, genişletmeden sonra birbirine temas ederek tek parça gibi birleşik olarak görüntülenmesinden kaynaklı satır ayrıştırılması sorunları için bir yöntem geliştirilmesi gerekmiştir. Kaç adet birleşmiş satır olduğunu bulup bunları gerektiği şekilde ayırmak ve böylece gerçek satır sayısını bulmak için izlenen işlem adımları aşağıda verilmiştir.

1. Genişletme yapıldıktan sonraki bitişik beyaz piksel gruplarını satır olarak kabul et
2. Her bir satırın yüksekliğini hesapla ve kaydet
3. Satır yükseklikleri bilgisini kullanarak bir medyan ve çeyrekler açıklığı (Inter Quartile Range - IQR) hesabı yap
4. Bulunan medyan ve çeyrekler açıklığı değerlerini kullanarak uç (outlier) yükseklikleri bul
5. Uç yüksekliğe sahip satırların yüksekliğinin, ortalama satır yüksekliğinin kaç katı olduğunu hesapla ve bu sayıyı bitişik satır sayısı olarak kabul et
6. Birleşmiş satırı, bitişik satır sayısı kadar böl

Yukarıdaki işlem adımları gerçeklendiğinde **Şekil 8**'de görülen bitişik dördüncü ve beşinci satırlar **Şekil 10**'daki gibi ayrılabilmektedir.

Şekil 10. Bitişik satırların ayrılması

4.3. Kelimelerin Ayrılması

Kelime ayırma işlemi için izlenen adımlar, satır ayırma için izlenenler ile benzerdir. Kelime ayırmada satır ayırma işlem adımları, genişletme oranları değiştirilerek aynı şekilde uygulanmıştır. Genişletme oranlarına belirli bir aralıktaki değerler sırasıyla test edilerek deneysel olarak karar verilmiştir. Dikey olarak en düşük %50. eleman dikey genişletme oranı, yatay olarak en düşük %25. eleman ise yatay genişletme oranı olarak seçilmiştir. Yatay olarak daha az genişletilerek kelimelerin ayrılması sağlanırken dikey olarak daha çok genişletilerek noktalama işaretlerinin ayrılmaması hedeflenmiştir.

4.4. Karakterlerin Ayrılıp Tanınması

Karakterlerin ayrılması işleminde her satırın yatay olarak ufak parçalara bölünmesi işlemi yapılmaktadır. Sonrasında bu karakter bölgelerinin bitişiğindeki bölgeler ile birleştirilerek içinde bulunan karakterlerin tanınması ve en yüksek olasılıkla tahmin edilen karakterin seçilmesi işlemi yapılmaktadır.

Karakter bölgeleri belirlenmesi işlemi sırasında karakterlere ait noktaların kullanılan algoritma başarısını düşürmesi nedeniyle öncelikle bitişik bileşenler bulunmuş ve noktalar silinmiştir. Daha sonra uygulanan işlemle sütunlardaki en yüksek beyaz piksel ile en alçak beyaz piksel arasındaki fark alınmıştır. Böylece her bir sütundaki karakterlerin yükseklikleri hesaplanmış ve bu bilgi kullanılarak ani yükseklik düşüşü ya da artışının olduğu yerler "Olası Ayrım Sütunu" olarak etiketlenmiştir. Bir sütunda beyaz piksel yoksa ve bir önceki sütunda beyaz piksel varsa bu sütun ayrım noktası olarak belirlenmiş ve ayrı yazılmış karakterler kelimeden ayrılmıştır. Ayrım sütunu olabilecek sütunlar için ise sütundaki beyaz piksel yüksekliğinin belirli bir eşik değerinin altında olup olmadığı kontrol edilmiştir. Eşik değerinin altında olan sütunlar -bir önceki sütun ayrım noktası değil ise- ayrım sütunu olarak kabul edilmiştir. Böylece içinde bir ya da daha az karakter bulunma olasılığı yüksek olan bölgeler belirlenerek işlenmeye hazır hale getirilmiştir. Karakter ayrım bölgelerinin çizgi olarak gösterildiği bir örnek **Şekil 11**'de verilmiştir.

Şekil 11. Olası karakter ayrım bölgeleri

Karakterlerin tanınması için kullanılan ağ yapısı **Tablo 5**'te verilmiştir. Teoride başarılı olabileceği düşünülen 31 farklı model denenerek en iyi sonuçları veren aşağıdaki model kullanılmıştır. Resmin üzerinde hangi karakterlerin hangi konumlarda olduğunu genel olarak tespit edebilmek ve resim özelliklerini çıkarabilmek için iki Evrişim Katmanı (Convolutional Layer) kullanılmıştır. Ardından resmin işlenmesini kolaylaştırmak ve bir yapıyı ezberlemesi yerine

yaklaşık yer tahminleri yapmasını sağlamak için Maksimum Havuzlama Katmanı (Maxpooling Layer), oranların sistematik bir şekilde ezberlenmesi ve çok spesifik bir kalıp halinde kalmasının önlenmesi için 0.25 bırakma oranına sahip bir Bırakma Katmanı (Dropout Layer) eklenmiştir. Bırakma Katmanını iki adet Evrişim Katmanı, Maksimum Havuzlama Katmanı ve yeni bir Bırakma Katmanı izlemektedir. Sonuçları bulmadan önce gelen veriyi düzleştirmek için bir Düzleştirme Katmanı (Flatten Layer), karar indirgemesi için ise 128 üniteli ve Düzleştirilmiş Doğrusal Birim (Rectified Linear Unit - ReLU) aktivasyon fonksiyonlu Dense katman ve Relu aktivasyon katmanı eklenmiştir. Son çıktı için 45 birimli Yumuşak Bağlı (Softmax) Dense katmanı kullanılmıştır.

Tablo 5. Karakter tanıma ve ayırma için oluşturulan ağ yapısı

Katmanlar ve Özellikleri
Evrişim Katmanı (Filtre sayısı: 32 Çekirdek Boyutu: 5*5)
Evrişim Katmanı (Filtre sayısı: 32 Çekirdek Boyutu: 5*5)
Maksimum Havuzlama Katmanı (Havuz Boyutu: 2*2)
Bırakma Katmanı (Bırakma Oranına: 0.25)
Evrişim Katmanı (Filtre sayısı: 32 Çekirdek Boyutu: 3*3)
Evrişim Katmanı (Filtre sayısı: 32 Çekirdek Boyutu: 3*3)
Maksimum Havuzlama Katmanı (Havuz boyutu: 2*2)
Bırakma Katmanı (Bırakma Oranına: 0.25)
Düzleştirme Katmanı
Relu Dense Katmanı (128 Üniteli)
Relu Aktivasyon Katmanı
Softmax Dense Katmanı (45 Üniteli)

Veri eğitiminde 0.001 öğrenme oranlı Adam Optimize Edici kullanılmıştır. Eğitim, 10 devirde (epoch), her bir yığında (batch) 100 veri olacak şekilde yapılmıştır. Eğitim doğruluğu 0.91'dur ve eğitim kaybı %29'dır. Test doğruluğu ise %86'dır.

Bir karakterin birden fazla parçaya bölünüp bölünmediğinin anlaşılması için ESA'ya verilmiş olan bölge ile bir sonraki bölge birleştirilerek yeniden boyutlandırılmış ve ESA'ya gönderilmiştir. Bu iki durumun çıktıları karşılaştırılmış ve daha yüksek olasılıklı çıktı seçilerek diğer seçenek elenmiştir. Bu işlem belirli kez tekrarlanmış, böylece bir karakterin birden fazla karakter bölgesine yayılmasının sorun oluşturması engellenmiştir. **Şekil 12**'de bir karakter için ESA'nın tahmin süreci görülmektedir. (Soldan sağa doğru) ilk şekilde ilk gönderilen bölge görülmektedir, sonrasında bu bölge ile bir sonraki ayrım sütunu ile ayrılmış bölge birleştirilerek gönderilmiştir ve ESA'nın tahminler için verdiği olasılıklar karşılaştırılmıştır, ikinci şekil için verilen olasılık ilkinden yüksek olduğu için tahmin işlemi bir sonraki adıma geçmiştir, bu sefer tekrardan bir sonraki ayrım sütununa kadar olan alan gönderilerek tekrar tahmin yapılmış ve olasılıklar karşılaştırılmıştır. Bu işlem dördüncü karedeki şekil için tekrarlandığında tahmin olasılığının düştüğü gözlemlenmiş ve son olarak ilgili karakterin beşinci karede bulunan karakter olduğuna (Şın karakteri) karar verilmiştir.

Şekil 12. Karakter tahmin süreci

Burada tekrar sayısı 5 olarak alınmıştır. Tekrar sayısı daha büyük tutulursa son adımda alınan bölge çok geniş olacak, yeniden boyutlandırılıp ESA'ya gönderilince yatay düz bir çizgi gibi görünecektir. Bu durumda ilgili karakterin bu şekilde yazılabilen "Sin" gibi karakterlerle karıştırılması çok kolaylaşacaktır. Tekrar sayısı daha az seçildiğinde ise sütunları arasında ani piksel değişimi çok olan "Şın" gibi karakterlerde eksik kesmeye sebep olabilmektedir.

4.5. Kelime Oluşturma

Bu aşamada, karakter gruplarının karşılık gelebileceği Türkçe kelimeler belirlenerek çeviri işlemi yapılmıştır. Osmanlıca'da Arap alfabesindeki karakterlerin birden fazla karşılığı olabilmektedir (**Şekil 13**'te de görüleceği gibi "kef" karakterinde ses "k, ğ, g, n" olmak üzere farklı şekillerde çıkabilmektedir). Bu aşamada kelime gruplarında bulunan karakterlerin tüm karşılıkları kullanılarak oluşabilecek tüm kelimelerin bulunabilmesi için karakterler arasında kartezyen çarpım işlemi yapılmaktadır. Bir karakterin karşılık gelebileceği tüm karakterler, diğer karakterlerin karşılık gelebileceği tüm karakterler ile eşleştirilmektedir. Bu aşamadan sonra oluşan "aday" kelimeler, Zemberek eklentisinin yazım kontrol özelliği ile kontrol edilmiş, eğer yazılan aday kelimeler bir kelime ile eşleşmiyorsa bu kelimeler yerine bu aday kelimelere benzer kelimeler yine Zemberek tarafından önerilmiştir.

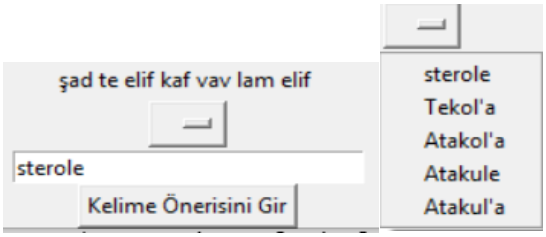
Bu adımda karşılaşılan en büyük zorluk Arap alfabesindeki karakter eksikliği sebebi ile bazı Türkçe kelimeleri desteklememesidir. Bu sebeple farklı kelimeler aynı yazıma sahip olabilmektedir. Bu sorunun giderilebilmesi için, bir kelimeye ait olabilecek en olası çeviriler kullanıcıya sunulacak ve böylece hatalar interaktif bir çeviri ortamı ile önlenecektir.

4.6. Kelime Önerileri

Görüntüden ayrıştırılan kelime gruplarının bölünmesi ile elde edilen bölgelerde bulunan karakterlere anlam kazandırılması gerekmektedir. Arap harflerinin bazen yazılmadan telaffuz edilmesi, bazen farklı şekillerde okunabilmesi de dikkat edilmesi gereken özelliklerdir. **Şekil 14**'de bu anlam kazandırma sonrası önerilen kelimelerin kullanıcıya sunulduğu bir arayüz kesimiyle ilgili bir ekran görüntüsü bulunmaktadır.

Harfin Asıl Hali	Harfin Okunuşu	Harfin Ses Karşılığı
ك	kef	k, g, ğ, n
گ	gef	g, ğ
ڭ	nef, sağır kef	ñ
ل	lâm	l
م	mim	m
ن	nun	n
و	vav	v, o, ō, u, ū, û
ه	he	h, e, a
لا	lamelif	la
ى	ye	y, ı, i, î

Şekil 13. Arap Harflerinde Farklı Okunabilen Harfler



Şekil 14. Zemberek kelime önerileri

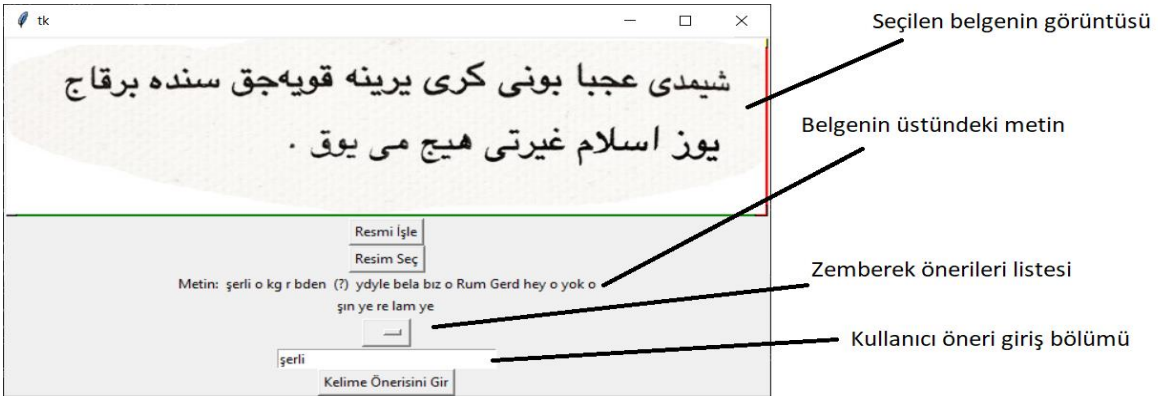
4.7. Arayüz Tasarımı

Geliştirilen arayüz; kullanıcının işlemek istediği görüntüyü açmasına, görüntüden ilgili metni seçmesine, metni işlemesine, işlemler sonrası oluşan metnin bütününü ve kelimeler için ayrı ayrı bulunan karakterleri ve önerilen çevirileri görüntülemesine olanak sağlamaktadır.

Kullanıcı arayüzünde sunulan "Metni Kırp" özelliği sayesinde kullanıcı çeviri yapmak istediği metin dışındaki şekilleri kırparak ilgilendiği metin parçasına perspektif dönüşüm ile düzeltme işlemi gerçekleştirebilmektedir. Bu işlem hem eğik olarak yazılmış metinlerin düzleştirilmesi problemini çözmekte hem de orijinal metnin taranması esnasında sayfanın görüntüsünün tam karşıdan alınamadığı durumlarda metnin doğru bir şekilde çözümlenebilmesini sağlamaktadır. Böylece geliştirilen satır ayırma algoritmasının karşılaşabileceği uç nokta hatalarının da önüne geçilmektedir. Metin kırpma aracıyla ilgili örnek bir ekran görüntüsü **Şekil 15**'de verilmiştir. Sol tarafta orijinal metin, sağ tarafta ise düzeltilmiş metin görülmektedir.

Şekil 15. Metin kırpma ve düzeltme aracı.

Şekil 16'da görülen "Resmi seç" butonuna tıklanarak bilgisayardan bir belge görüntüsü açılır, sonra bu belge görüntüsü istenilen şekilde kırpılarak üstündeki yazıya odaklanılması ve bu yazının eğiminin düzeltilmesi sağlanır. Resmi işle butonuna tıklandığında tüm ayıklama ve çeviri işlemleri kullanıcıdan soyutlanmış olarak gerçekleşir. Bu adımdan sonra kullanıcı seçilen resmin üstündeki kelimelere tıklayarak o kelimeler için bulunmuş karakterleri, o kelime için en iyi öneriyi ve öneriler listesini görebilir, isterse kendi önerisini girebilir.



Şekil 16. Kullanıcı arayüzü

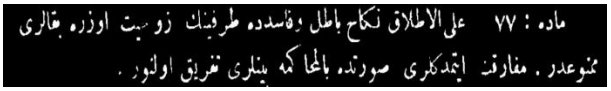
V. DENEYSEL SONUÇLAR

Bu bölümde, proje kapsamında kullanılan yöntemlerin başarıları tartışılmıştır.

5.1. Metin Ayıklama Başarısı

Metin ayıklama aşamasının üç adet temel adımı vardır; metnin satırlara ayrılması, satırların kelimelere ayrılması ve kelimelerin karakterlere ayrılması. Kelimelerin karakterlere ayrılması, karakter tanıma aşaması ile birleşik olduğu için "Karakter Tanıma Aşaması Başarıları" bölümünde tartışılacaktır.
Arayüz bölümünde belirtilen metin kırpma ve düzeltme aracının kullanılmasıyla işlenmiş 102 adet metnin 99'u başarıyla ayrılabilmektedir. Bu 99 metnin 14'ünde birleşik satırlar bulunmasına rağmen, birleşik satır ayrıştırma algoritması çalıştırılarak bu satırlar başarıyla ayrılmıştır. Satır ayırma başarısı %97'dir.

Satır ayırma işlemi, genişletmeden sonra birbirine temas eden satırların temas noktalarını baz alarak ayırma yapmaktadır. Ancak arka arkaya gelen genişletilmiş satırların çok noktada temas ettikleri örneklerde, sunulan yöntem **Şekil 17**'de görüldüğü gibi iki satırı ayıramayarak tek bir satır gibi alabilmekte, ya da **Şekil 18**'deki gibi satırları ortadan bölebilmektedir.



Şekil 17. Ayrılamayan satırlardan bir örnek

Şekil 18. Satır ayırma algoritmasının hatalı olarak böldüğü bir satır örneği

Başarılı olarak ayrılmış 204 satır üzerinde yapılan kelime ayırma işleminin başarısı %96'dır. Başarı oranının düşmesini tetikleyen en büyük etmen, Elif gibi bazı karakterlerin, bağlı olduğu kelimelerden çok uzakta yazılarak başka kelimeler olarak sayılmasıdır.

5.2. Karakter Tanıma Başarısı

Karakter tanıma aşamasında başarı ideal olarak bölünmüş ve ESA'ya gönderilmiş karakterlerin tanınma başarısı ve girilen metin üstünde karakterlerin otomatik ayrılarak tanınması başarısı olmak üzere iki şekilde ölçülmüştür. Tüm yazı tipleri için karışık tahminlerin başarı oranları **Tablo 6**'da gösterilmektedir.

İdeal olarak bölünmüş karakterlerin başarı oranları hesaplanırken test edilen veriler elle hazırlanmış olan veri setinden alınmıştır. Her bir karakter için toplamda o karakter için toplanan verinin 1/8'i rastgele olarak seçilerek test için ayrılmıştır. Eğer bir karakter için sekizden az sayıda örnek toplandıysa, rastgele biri eğitim seti ayrılmıştır. Bu ayırma işlemi ayrı bir sistemle hazırlanmış, ana sistemde verilerin halihazırda ayrılmış halleri kullanılmıştır.

Tablo 6. Karakterlerin hatırlanma ve hassasiyet oranları (tüm yazı tipleri için)

	Hatırlama	Hassasiyet		Hatırlama	Hassasiyet
elif	1	0,952381	**lam**	1	1
be	0,789474	0,789474	**mim**	0,736842	0,933333
te	1	0,772727	**nun**	0,851852	0,884615
se	0	0	**he**	1	0,857143
cim	0,9	1	**vav**	0,952381	0,869565
ha	0,909091	0,909091	**ye**	0,882353	0,909091
hha	1	1	**pe**	0	0
dal	0,814815	0,916667	**çim**	1	1
zel	0,25	0,5	**je**	1	0,5
ra	0,925926	0,925926	**gef**	1	0,666667
ze	0,888889	0,727273	**nef**	0,75	1
sin	0,923077	1	**1**	0,666667	1
şın	1	0,85	**2**	1	0,5
şad	0,8	1	**3**	0,75	0,75
dad	0,5	0,666667	**4**	1	0,75
tı	1	0,909091	**5**	1	0,5
zı	0,75	1	**6**	1	1
ayn	0,941176	0,8	**7**	1	1
ğayn	0,666667	0,8	**8**	1	1
fe	0,909091	0,833333	**9**	1	1
kaf	0,9375	0,9375	**0**	1	1
kef	0,928571	1	**lamelif**	1	1

Çok sınıflı olan bu problemle ilgili isabetli bir başarı oranı gösterilebilmesi için hatırlama (recall) ve hassasiyet (precision) **[22]** başarı hesapları kullanılmıştır. İdeal olarak bölünmüş karakterlerde hatırlama %88,47, hassasiyet %82,75 olmuştur; bu başarı oranına sayıların tanınma oranı dahildir.

Programın karakter ayırma kısmında aldığı kelimeler üzerinde bulduğu karakterlerin sayısının, kelimede bulunan karakterlerin sayıyla aynı olma başarısı Matbu metinler için %87,31, Rika metinler için %86,06, Talik metinler için%59,43 olmuştur. Bir kelimenin doğru sayıda karaktere bölünmesi, karakterlerin tanınmasındaki başarı oranını artırmaktadır. Talik metinlerde karakter ayırma başarısının bu denli düşmesinin sebebi, karakterlerin diğer stillerden daha fazla uzatılarak, daha eğik yazılması ve karakterlerin dikey olarak birbirinin üstüne gelebilmesidir.

Karakter tanıma kısmında ise kelimede bulunan karakter sayısına oranla Matbu metinler için karakterler %46,36, Rika metinler için %44,13, Talik metinler içinse %19,38 ortalamayla doğru bulunmuştur. Bu oranlar kelimelerdeki karakter sayısı doğru bulunduğunda ise doğru karakter tahmini için sırasıyla %53,38, %51,28 ve %34,6'ya çıkmıştır. Talik metinlerde, karakterlerin dikey olarak üst üste geçebilmesine karşın karakterlerin doğru bölünmesinin başarıyı artırdığı gözlemlenmiştir. Karakter sayısının doğru olduğu kelimelerde bulunan doğru ve benzer karakterlerin toplamının kelimede bulunan karakter sayısına oranlarının ortalaması ise sırasıyla; %71,63, %70,12 ve %51,13 olmuştur.

Literatürde Osmanlıca el yazması ve Matbu belgeler üzerinde çözümleme yapan bir sistem bulunmamaktadır. Mevcut çalışmaların çoğu isole edilmiş karakterleri tanımaya yöneliktir ve Osmanlıca bir belge üzerinde tam bir çeviri amaçlanmamaktadır **[23]**. Bu konuda açık bir veri seti bulunmadığı için araştırmacılar kendi verisetlerini kullanmışlarıdır. Tüm bu sebeplerle önerilen sistemin diğer sistemler karşılaştırılması mümkün değildir. Rabi ve arkadaşları Arapça kelimelerden oluşan bir veri seti üzerinde çalışmışlardır **[8]**. Çalışmaları resim olarak verilen tam bir metin analizi üzerinde değildir. Isole edilmiş kelimeleri GMM tabanlı bir sistem ile %87,93 başarı ile tanımaktadırlar. Önerilen Evrişimsel Sinir Ağları Tabanlı Osmanlıca Belge Çözümleyici sistemi metinlerdeki satırları %89 başarı oranı ile ayırabilen K-means temelli çalışmadan daha iyi bir performans sergilemiştir **[4]**. Bu çalışmalara ek olarak, derin öğrenme tabanlı Arapça el yazısı karakter tanıma işlemi yapan çalışmada **[11]** ise %94,9 gibi yüksek bir başarı oranı sağlanmıştır. Adil bir karşılaştırma yapmak mümkün olmamakla birlikte, özel çıkarılmış bir veri setinde, üç farklı yazı tipine göre ve fazladan karakterler ile (pe, je gibi Osmanlıca'da olup da Arapça'da olmayan karakterler ve rakamlar) eğitilmiş olan sistemimiz %88,47 ile bu oranla yarışabilir düzeydedir.

Bir yazı stili kuralları içinde, kişiden kişiye değişebilecek yazım şekilleri, karakter algılama başarısını değiştirebilir. Bu durum Rika ve Talik tipinde yazılmış metinlerde geçerlidir. Matbu metinler kalıp halinde yazıldıkları için kişiden kişiye el yazısı değişme durumu yoktur ve başarı farklı metinler için stabil olacaktır.

VI. SONUÇ

Evrişimsel Sinir Ağları Tabanlı Osmanlıca Belge Çözümleyici, Osmanlıca belgelerin günümüz Türkçe'sine çevrilmesi hedefiyle geliştirilmiştir. Projeyi hedefine ulaştırmak için çeşitli görüntü işleme yöntemleri, yapay sinir ağları ve Zemberek eklentisi kullanılmıştır. Geliştirilen sistemin pratik olarak kullanılmasını sağlamak amacı ile interaktif bir çeviri ortamı sağlanmıştır. Proje kapsamında başarılı bir şekilde farklı belge tipleri, yazım stilleri ve sayfa şekilleri üstünde çalışılmış, birden fazla stil için ortak bir çözüm yöntemi önerilmiştir.

Önerilen sistemin oluşturulan yapısı, girdi değişikliklerine karşı dayanıklı olduğunu göstermiştir. Belgenin üstünde yapılan oynamaların, gürültünün ve yazı stilinin başarı üstündeki etkilerinin oranı, bu etkilere karşı dirençli bir sistem üretildiğinin göstergesidir. Ön işleme aşaması olan metnin satırlara ve kelimelere ayrılması adımlarında gözlemlenen yüksek başarı oranı bu konular için önerilen istatistik temelli algoritmaların güçlü çözümler olduğunu göstermektedir.

Karakter ayırma ve tanıma algoritmasında karşılaşılabilecek hataların düzeltilmesi amacı ile Zemberek eklentisi kullanılmıştır. Kelimelerdeki sesli harflerin kontrolünü yapacak ve kelimeleri buna göre düzenleyerek Zemberek'e verecek bir sistemin geliştirilmesi ile Zemberek'ten daha yüksek verim alınabilir. Buna ek olarak algılanan karakterlere karşılık kullanıcının kendi girdiği kelimeleri hatırlayacak bir sistem geliştirilerek kelime tahmini başarısı artırılabilir ve özelleştirilebilir.

Yapılan deneylerde, önerilen sistemin güvenilirliğini arttırmak için karakterlerin ayrılması noktasındaki başarısının yükseltilmesi gerektiği sonucuna varılmıştır. Bu amaçla takip edilebilecek bir başka yol karakter ayrımı yapmak yerine obje tanıma mantığıyla çalışacak derin öğrenme tabanlı bir sistem kurmaktır. Ancak kural tabanlı kısıtlamalar olmadığı sürece bu tarz bir yaklaşımın hesaplama karmaşıklığı da yüksek olacaktır. İstatistiksel yöntemler kullanılarak oluşturulan kurallar yardımıyla karakter ayırma yaklaşımı ise işlem yükünü bariz bir miktarda düşürmekle beraber kısıtlı bir sistem başarısı vermektedir. Daha fazla veri ile eğitilen bir sistem veya bu iki yaklaşımı birleştirecek hibrit bir yöntem daha başarılı sonuçlar verebilir. Oluşturulan proje, daha önce kapsam olarak eşi bulunmamasıyla birlikte birden fazla disiplini bir araya getirerek, tarihi belgelerin işlenmesi için yapılabilecek araştırmalara önayak olmuştur.

TEŞEKKÜR

Evrişimsel Sinir Ağları Tabanlı Osmanlıca Belge Çözümleyici projesi kapsamında bize arşivlerini açan Türkiye Büyük Millet Meclisi Kütüphane ve Arşivler Başkanlığı'na teşekkür ederiz.

KAYNAKLAR

[1] T. Y. D. A.,S. (2016). Dervaze metin mütercimi, http:// dervaze.com/translate-ott/, (November 2020).

[2] www.osmanlicayaceviri.com/. (2017). Osmanlıca çeviri, www.osmanlicayaceviri.com/, (November 2020).

[3] F. Farooq, Venu Govindaraju, and M. Perrone, (2005), Pre-processing methods for handwritten arabic documents *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)*, 267–271 Vol. 1.

[4] S. Erkilinc, M. Jaber, E. Saber, P. Bauer, and D. Depalov, (2012), Text, photo, and line extraction in scanned documents *Journal of Electronic Imaging*, vol. 21, pp. 3006–, Jul. 2012. DOI: 10.1117/1.JEI.21.3.033006.

[5] S. Jin, Y. You, and Y. Huafen, (2010), A scanned document image processing model for ınformation system *2010 Asia-Pacific*

Conference on Wearable Computing Systems, 198–201.

[6] B. B. Chaudhuri and S. Bera, (2009), Handwritten text line ıdentification in ındian scripts *2009 10th International Conference on Document Analysis and Recognition*, 636–640.

[7] A. Alsaeedi, H. A. Mutawa, S. Snoussi, S. Natheer, K. Omri, and W. A. Subhi, (2018) Arabic words recognition using cnn and tnn on a smartphone *2018 IEEE 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR)*, 57–61.

[8] M. Rabi, M. Amrouch, Z. Mahani, and D. Mammass, (2016), *Recognition of cursive arabic handwritten text using embedded training based on hmms* 2016 International Conference on Engineering MIS (ICEMIS), 1–5.

[9] Khedher, M. I., Jmila, H., & El-Yacoubi, M. A., (2020), Automatic processing of Historical Arabic Documents: a comprehensive survey. Pattern Recognition, 100, 107144.

[10] Ali, A. A. A., & Mallaiah, S., (2021), Intelligent handwritten recognition using hybrid CNN architectures based-SVM classifier with dropout. Journal of King Saud University-Computer and Information Sciences.

[11] El-Sawy A., EL-Bakry H., Loey M., (2017), CNN for Handwritten Arabic Digits Recognition Based on LeNet-5. In: Hassanien A., Shaalan K., Gaber T., Azar A., Tolba M. (eds) Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2016. AISI 2016. Advances in Intelligent Systems and Computing, vol 533. Springer, Cham. https://doi.org/10.1007/978-3-319-48308-5_54

[12] Lamsaf, Asmae & Aitkerroum, Mounir & Boulaknadel, Siham & FAKHRI, Youssef. (2018). Lines segmentation and word extraction of Arabic handwritten text. SCA '18: Proceedings of the 3rd International Conference on Smart City Applications. 1-7. 10.1145/3286606.3286831.

[13] Ayesh, Muna & Mohammad, Khader & Qaroush, Aziz & Agaian, Sos & Washha, Mahdi. (2017), A Robust Line Segmentation Algorithm for Arabic Printed Text with Diacritics. Electronic Imaging. 2017. 42-47. 10.2352/ISSN.2470-1173.2017.13.IPAS-204.

[14] Alworafi, Mokhtar & Manjunath, Ravikumar & Pradeep, R., (2014), Text Line Segmentation of Arabic Handwritten Documents using Line Height Method *International Journal of Advanced Research in Computer Science and Software Engineering*. 4. 5.

[15] Ali, A. A. A., & Suresha, M., (2019), Efficient algorithms for text lines and words segmentation for recognition of Arabic handwritten script *Emerging research in computing, information, communication and applications*, 387-401, Springer, Singapore.

[16] Osmanlıca Optik Karakter Tanıma (OCR) Sistemi, http://miletos.co/tr/showcase/ottoman-ocr, (November 2021).

[17] C. Ozan Ceyhan & Melih Taşdizen & Berkin Malkoç & Atabey Kaygun & Kürşat Aker, (2017), Osmanlıca Baskı Metinler İçin Arama Altyapısı

[18] El-Sawy, A., Loey, M., & El-Bakry, H. (2017). Arabic handwritten characters recognition using convolutional neural network. WSEAS Transactions on Computer Research, 5, 11-19.

[19] Ottoman Turkish Characters, https://www.kaggle.com/alpbintuuzun/ottoman-turkish-characters, (2020).

[20] Akın, A. A., & Akın, M. D., (2007), Zemberek, an open source nlp framework for turkic languages. Structure, 10, 1-5.

[21] Bestil, H. İ., & Güvensan, M. A. (2019). Türkçe Kısa Mesajları Sınıflandıran Çok Katmanlı Süzgeçleme Mimarisi ve Akıllı SMS Kutusu. International Journal of Advances in Engineering and Pure Sciences, 31(1), 17-28.

[22] Powers, D. M., (2020), Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. arXiv preprint arXiv:2010.16061.

[23] Kurt, Z., Turkmen, H. I., & Karsligil, M. E., (2009), Linear discriminant analysis in ottoman alphabet character recognition *Proceedings of the European Computing Conference*, 601-607, Springer, Boston, MA.