Journal of the Faculty of Engineering and Architecture of Gazi University 39:3 (2024) 1999-2009

Mühendislik Mimarlık Fakültesi Dergisi
Journal of The Faculty of Engineering and Architecture of Gazi University
Elektronik / Online ISSN : 1304 - 4915
Basılı / Printed ISSN : 1300 - 1884

Encoder character based using decoder and attention algorithms word production

İsa Ergin[1]*, Timur İnan[2]
[1]Department of Information Technologies, Faculty of Engineering, Altınbaş University, 34217, İstanbul, Türkiye
[2]Department of Software Engineering, Faculty of Engineering and Architecture, Altınbaş University, 06680, Ankara, Türkiye

Highlights:

- Using artificial neural networks for Turkish meaningful word generation
- Language model development using encoder decoder and attention architecture
- Production of meaningful words with the character-based trained language model

Keywords:

- Artificial intelligence, processing
- Machine learning
- Natural language
- Decoder encoder attention
- Text generation

Article Info:

Research Article
Received: 17.11.2022
Accepted: 13.10.2023

DOI:

10.17341/gazimmfd.1206277

Correspondence:

Author: İsa Ergin
e-mail: isaergin@gmail.com
phone: +90 545 898 7749

Graphical/Tabular Abstract

This study aims to produce meaningful words in accordance with character-based Turkish grammar rules by using encoder decoder and attention architecture from deep learning algorithms. In encoder-decoder models, information about the input data to the decoder is transmitted only by the fixed-length content vector obtained by the encoder. That is, the output data has no direct relationship with the input value. As a result, when the array size of the text data gets longer, the relationships between the first value and the last output values are forgotten. In order to overcome this problem, attention architecture is used in the structure of Encoder decoder models. Thanks to this method, when the decoder makes a guess, it reviews all the hidden state information of the LSTM cells in the encoder (Figure A). In this case, even if the length of the text data increases, word production results in more successful results as the input values are constantly controlled by the attention architecture in every value that is produced continuously.

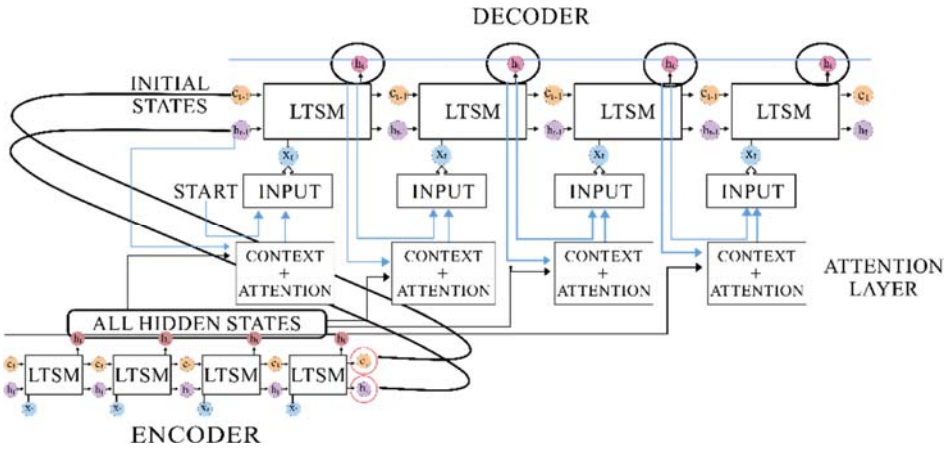


Figure A. Encoder decoder and attention model

Purpose: It is to produce meaningful words in accordance with Turkish grammar rules by developing a character-based language model working on Turkish data set by using encoder decoder and attention architecture from deep learning algorithms.

Theory and Methods: In the creation of the language model, encoder-decoder and attention architecture, which are deep learning algorithms, are used. LSTM networks are used in the structure of the encoder and decoder model. These networks cannot show sufficient success in long text strings because their memory is not enough. Therefore, this problem is tried to be overcome by using the attention mechanism together with LSTM networks in the structure of the encoder and decoder model. The attention architecture tries to produce output data by focusing on different parts of the input data at each processing step. Thanks to this structure, the model can learn on its own what it should pay attention to in the input data, depending on what it produces.

Results: The model is operated at different threshold values of the temperature sampling method at epoch values of 100 and 200. Model; The 100 epoch and temperature sampling method gives the best result with a 90.6% success rate at the 0.3 threshold, while the 200 epoch and temperature sampling method gives the best result with a 91.9% success rate at the 0.5 threshold. It is seen that the model is successful in creating short and long words and can create some meaningful word groups.

Conclusion: It is thought that to improve the results of the language model used, using a larger and higher quality training data set, increasing the number of LSTM units, increasing the number of training iterations (epoch) and representing with high-dimensional word vectors can increase the success of the model.

Journal of the Faculty of Engineering and Architecture of Gazi University 39:3 (2024) 1999-2009

Mühendislik Mimarlık Fakültesi Dergisi
Journal of The Faculty of Engineering and Architecture of Gazi University
Elektronik / Online ISSN : 1304 - 4915
Basılı / Printed ISSN : 1300 - 1884

Kodlayıcı-kod çözücü ve dikkat algoritmaları kullanılarak karakter tabanlı kelime üretimi

İsa Ergin[1]*, Timur İnan[2]
[1]Altınbaş Üniversitesi, Fen Bilimleri Enstitüsü, Bilişim Teknolojileri Bölümü, 34217, Bağcılar, İstanbul, Türkiye
[2]Altınbaş Üniversitesi, Mühendislik ve Mimarlık Fakültesi, Yazılım Mühendisliği Bölümü, 34217, Bağcılar, İstanbul, Türkiye

ÖNEÇIKANLAR

- Türkçe kelime üretimi için yapay sinir ağlarının kullanılması
- Kodlayıcı-kod çözücü ve dikkat mimarisi kullanılarak dil modeli geliştirme
- Karakter tabanlı eğitilen dil modeli ile anlamlı kelime üretimi

Makale Bilgileri
Araştırma Makalesi
Geliş: 17.11.2022
Kabul: 13.10.2023

DOI:
10.17341/gazimmfd.1206277

Anahtar Kelimeler:
Yapay zekâ,
makina öğrenmesi,
doğal dil işleme,
dil modeli,
metin üretimi,
derin öğrenme

ÖZ

Bu çalışmada, derin öğrenme algoritmalarından kodlayıcı-kod çözücü ve dikkat mimarisi kullanılarak karakter tabanlı Türkçe dil bilgisi kurallarına uygun anlamlı kelime üretimi amaçlanmıştır. Geliştirilen modelin sonuçları diğer derin öğrenme algoritmaları olan LSTM ve GRU modellerinin sonuçları ile karşılaştırılmaktadır. LSTM ve GRU modelleri ile oluşturulan dil modelleri 100 ve 200 epoch değerlerinde ve sıcaklık örnek alma yönteminin farklı eşik değerlerinde birbirine yakın sonuçlar verdiği görülmektedir. Bu modellerden en yüksek başarı değerini 200 epoch ve 0,5 sıcaklık eşik değerinde %88,40 ile GRU modeli vermektedir. Bu çalışma için geliştirilen kodlayıcı-kod çözücü ve dikkat dil modeli ise 100 ve 200 epoch değerlerinde ve sıcaklık örnek alma yönteminin farklı eşik değerlerinde en yüksek başarı değerini 200 epoch ve 0,5 sıcaklık eşik değerinde %91,90 ile vermektedir. Yapılan denemeler sonunda, kodlayıcı-kod çözücü ve dikkat mimarisi modeli LSTM modeline göre ortalama olarak %2,83 ve GRU modeline göre ortalama olarak %0,19 oranında daha fazla başarı göstermiştir.

Encoder character based using decoder and attention algorithms word production

HIGHLIGHTS

- Using artificial neural networks for Turkish meaningful word generation
- Language model development using encoder decoder and attention architecture
- Production of meaningful words with the character-based trained language model

Article Info
Research Article
Received: 17.11.2022
Accepted: 13.10.2023

DOI:
10.17341/gazimmfd.1206277

Keywords:
Artificial intelligence,
machine learning,
natural language processing,
language model,
text generation,
deep learning

ABSTRACT

In this study, it is aimed to produce meaningful words in accordance with character-based Turkish grammar rules by using encoder-decoder and attention architecture, which are deep learning algorithms. The results of the developed model are compared with the results of LSTM and GRU models, which are other deep learning algorithms. It is seen that the language models created with LSTM and GRU models give similar results at 100 and 200 epoch values and at different threshold values of the temperature sampling method. Among these models, the GRU model gives the highest success value with 88.40% at 200 epochs and 0.5 temperature threshold value. The encoder-decoder and attention language model developed for this study gives the highest success value of 91.90% at 100 and 200 epoch values and at different threshold values of the temperature sampling method at 200 epoch and 0.5 temperature threshold value. At the end of the experiments, the encoder-decoder and attention architecture model showed an average of 2.83% more success than the LSTM model and an average of 0.19% more success than the GRU model.

*Sorumlu Yazar/Yazarlar / Corresponding Author/Authors : *isaergin@gmail.com, timur.inan@altinbas.edu.tr / Tel: +90 545 898 7749

1. Giriş (Introduction)

Doğal dil işleme (DDİ) insanların yazılı ve sözlü olarak kullandığı doğal dillerin bilgisayarlar tarafından analiz edilerek kurallarının belirlendiği yapay zekanın bir alt uygulama alanıdır [1]. DDİ, insan bilgisayar etkileşimi alanına odaklanmaktadır. Bundan dolayı doğal dilde üretilen ve bilgisayarlar aracılığıyla erişilebilen bilgilerin geniş boyutlara ulaşmasından dolayı doğal dil işleme ihtiyacı günümüzde artarak devam etmektedir. DDİ alanının en büyük sorunu metin üretimi esnasında dil kurallarına uygun anlamlı ve tutarlı metinler oluşturmada yaşanılan zorluklardır. Metin üretimi esnasında kullanılan karakterlerin ve kelimelerin bir bütünlük içerisinde doğru bir şekilde yan yana sıralanması çok önemlidir. Metin verileri arka arkaya gelen ve aralarında zaman sırasına göre sıralanmış sözcüklerden, sözcük öbeklerinden ve cümlelerden oluşan kendi aralarında bütünlük özelliği gösteren bir düzen halinde bulunmaktadır. Bu nedenle DDİ alanında başarılı metin uygulamaları geliştirebilmek için doğal dillerin yapısını oluşturan karakterlerin, kelimelerin ve kelime gruplarının birbirileriyle nasıl bir bağlantı ve sıra içerisinde olduklarını analiz edebilmek çok önemlidir [2].

DDİ uygulamaları geliştirirken bilgisayar bilimi, istatistik, dil bilimi, makine öğrenmesi, yapay sinir ağları gibi farklı alanlardan yararlanılmaktadır. Makinalar tarafından kullanılacak doğal dillerin, dilbilim ve bilgisayar bilimi tarafından incelenip analiz edilmesi sonucunda kurallarının belirlenmesi ve derin öğrenme algoritmaları tarafından kullanılabilecek bir formata dönüştürülmesi gerekmektedir [3]. DDİ uygulamaları günümüze kadar farklı zamanlarda, farklı yöntemler kullanılarak geliştirilmiştir. DDİ'nin ilk yıllarında bilgisayar programlarının daha çok kullanıldığı, metin içerisinde geçen kelime ve kelime gruplarının sayısının hesaplandığı ve buna bağlı olarak belirli kararların verildiği kural tabanlı makine öğrenmesi yöntemleri kullanılmıştır [4]. Artan veri miktarı ve bilgisayar teknolojilerindeki gelişmelerin hızla artması, makina öğrenmesi yöntemlerinin geliştirilmesi ve buna paralel olarak geçmiş bilgileri hatırlayarak öğrenen özyinelemeli derin yapay sinir ağlarının kullanılması DDİ alanında başarılı sonuçlar alınmasını sağlamıştır [5]. Özyinelemeli yapay sinir ağları doğal dillerin yapısını çözümleyerek ve geçmiş bilgileri hatırlayarak öğrenen yöntemler olmasından dolayı birbiriyle tutarlı ve anlamlı metinler oluşturmada daha başarılı sonuçlar vermektedir. Bu yöntemler metin verisi içerindeki ilişkilerden çıkarımlar elde edecek istatistiksel hesaplamalar yapmakta ve modelin kendi kendine öğrenebilmesini sağlamaktadır [6].

Günümüzde metin üretimi uygulamalarında genellikle özyinelemeli yapay sinir ağları olan tekrarlayan sinir ağları (recurrent neural network-RNN), uzun kısa süreli bellek (long short-term memory-LSTM), geçitli tekrarlayan birim (gated recurrent unit-GRU), üretken düşman ağları (generative adversarial network- GAN) kodlayıcı-kod çözücü (encoder-decoder) modelleri ile birlikte dikkat mekanizması (attention mechanism) ve transformer modelleri kullanılmaktadır [7]. Bu modellerden RNN'ler DDİ uygulamalarında çok uzun zamandır kullanılan, uzun metinlerin üretilmesinde kötü sonuçlar veren daha çok küçük veri setleri üzerinde çalışan derin öğrenme algoritmalarıdır. LSTM ve GRU modelleri ise RNN ağlarının geliştirilmiş bir modelidir Kodlayıcı-kod çözücü modeller ise yapılarında genellikle LSTM ve GRU ağlarının farklı sayıda ve farklı parametrelerde kullanılması ile oluşturulan çok daha uzun metin verilerinin işlenmesinde kullanılan daha başarılı yöntemlerdir [8]. Dikkat algoritması ise kodlayıcı-kod çözücü modellerinin başarısını artıran çok daha uzun metin dizelerindeki ilişkileri hatırlamakta daha başarılı olan ve metin üretiminde daha iyi sonuçlar veren derin öğrenme algoritmalarıdır Transformer modeller son zamanlarda DDİ uygulamalarında yoğun bir şekilde kullanılan, yapısında sadece dikkat mekanizmasının kullanıldığı yeni bir modeldir. Bu modellerin eğitilmeleri için çok büyük veri setlerine ve güçlü donanımlara ihtiyaç duymaktadır. GAN modeller ise genellikle görüntü üretmede kullanılan algoritmalardır. Günümüzde metin üretimi alanında da kullanılmaktadır. Ancak bu ağların eğitilmeleri çok zahmetli ve zordur [9]. Yapılan çalışmada özellikle derin öğrenme algoritmalarından kodlayıcı-kod çözücü ve dikkat mekanizmasının kullanılma nedeni, dikkat mekanizmasının daha önce DDİ alanında kullanılan derin öğrenme algoritmalarının (RNN, LSTM, GRU) çalışma mantığını değiştirerek daha iyi sonuçlar vermesidir. Eski modellerde (RNN, LSTM, GRU) tüm giriş verileri tek seferde işlenerek giriş verisinin tüm bilgilerinin özetini tutan bir çıkış özet vektörü oluşturulmaktadır. Bu özet vektör giriş verisiyle ilgili yeterli bilgiye sahip olamadığı için kod çözücüye giriş verisiyle ilgili yetersiz bilgi aktarmakta ve bunun sonucunda kod çözücü metin ile ilgili doğru tahminler yapamamaktadır. Bu modellerin giriş ve çıkış verilerinin uzunluğu arttıkça performansı daha da yetersiz kalmaktadır. Bundan dolayı bu modellerin çok fazla eksikliği bulunmaktadır. Bu sorunların üstesinden gelebilmek için kodlayıcı ve kod çözücü modellerinin yapısında dikkat mimarisi kullanılmaya başlanmıştır. Kodlayıcı-kod çözücü modellere dikkat mimarisinin eklemlenmesiyle giriş verisinin tüm bilgisi artık tek bir özet vektör olarak tutulmamaktadır. Kod çözücü artık sadece kodlayıcı vektöre bağlı kalarak çıktı üretmek yerine, her adımda çıktıyı üretirken sürekli olarak giriş verilerinin farklı bölümlerine ve şu ana kadar ne ürettiğine bağlı olarak neye dikkat etmesi gerektiğine bakarak çıktı değerlerini üretmektedir [10].

2. Literatür Araştırması (Literature Review)

Karakter tabanlı dikkat merkezli kodlayıcı-kod çözücü modeller kullanılarak kelime üretimiyle ilgili literatür incelendiğinde, karakter düzeyinde temsil gücü üzerine uzun zamandır çalışmalar yapıldığı gözlemlenmiştir [29,30]. Bu çalışmalardan biri olan Poulos vd. [11] çalışmalarında karakter dizinlerini işleyerek eğitilen dikkat merkezli kodlayıcı-kod çözücü modelde metin transkripsiyonu gerçekleştirilmiştir. Modelde kullanılan dikkat mekanizmasının kaynak ve hedef karakterler arasındaki sıralama işlemlerinde başarılı olduğu sonucuna varmışlardır. Eriguchi vd. [12] ise dikkate dayalı karakter ve kelime tabanlı makine çevirisi modeli geliştirerek İngilizce ve Japonca dilinde çeviri denemeleri yapmışlardır. Bu çalışmada hedef dillerdeki kelimeleri kullanarak öğrenen karakter tabanlı sistemlerin kelime tabanlı modellerden daha hızlı ve daha doğru çeviriler ürettiği sonucuna ulaşmışlardır. Feng vd. [13] yaptıkları çalışmada Skip–Gram mimarisi ile karakter tabanlı kodlayıcı-kod çözücü modelini birleştirerek yeni bir model tasarlamışlardır. Çalışmanın sonucunda karakter düzeyindeki bilgilerden faydalanarak öğrenen modelin, zengin morfolojiye sahip dillerdeki kelimeler arasındaki ilişki düzeylerini daha iyi tespit ettiği sonucuna varmışlardır. Renduchintal vd. [14] ise karakter tabanlı dikkat merkezli kodlayıcı-kod çözücü kullandıkları modelde, dillerin morfolojik kalıplarını göz ardı ederek sadece kelimeler üzerinde çalışmışlardır. Bu çalışma sonucunda karakter tabanlı dil modellerinin dil kurallarını kelime tabanlı modellere göre daha iyi yakaladığı sonucunu elde etmişlerdir. Modelin aynı zamanda tahmin aşamasında basit bir softmax katmanı kullanarak karakterler arasındaki ilişkileri yakalamada başarı elde etmesine rağmen modelin çok fazla gizli katman kullandığını ve çok fazla bellek tükettiğini tespit etmişlerdir. Makine çevirisi üzerinde çalışma yapan Yang vd. [15] benzer bir model geliştirmişlerdir. Bu model ile hedef dildeki tüm kelimeleri kullanarak oluşturdukları modelin kelime tabanlı modelden daha hızlı ve daha doğru çeviriler yaptığını tespit etmişlerdir. Bahdanau vd. [16] konuşma tanıma çalışmalarında karakter tabanlı dikkat mekanizmasına dayalı bir model geliştirmişlerdir. Bu modelin büyük bir veri seti kullanıldığında giriş ve çıkış dizeleri arasındaki ilişkileri hizalamada başarılı olduğunu görmüşlerdir. Meng vd. [17] yaptıkları

çalışmada karakterlere dayalı dikkat tabanlı uçtan uca çalışan konuşma tanıma modeli geliştirmişlerdir. Bu modelin sözcükleri ve alt sözcükleri başarılı bir şekilde ürettiği ve modelde kullanılan parametre sayısını azaltarak %11,9 seviyesinde iyileştirme sağladığını görmüşlerdir. Kwon vd. [18] yaptıkları çalışmada karakterleri hem girdi hem de çıktı olarak alan hiyerarşik bir char2word kodlayıcıya sahip bir model önermişlerdir. Günümüzde mevcut makine çevirisi modellerinin çoğu, girdi ve çıktı birimleri olarak karakter gruplarını veya tam kelimeyi kullanmaktadır. Bu çalışmanın sonucunda karakter kodlayıcının hiyerarşik temsilinin hesaplama karmaşıklığını azalttığını ve çeviri performansını iyileştirdiğini tespit etmişlerdir. Ayrıca kod çözücüden alınan dikkat değerlerini incelediklerinde modelin yaygın kelimeleri tek bir gömme içine sıkıştırmayı öğrendiğini, oysa isimler ve yerler gibi az geçen özel isimlerin karakter olarak temsil edildiğini gözlemlemişlerdir. Noord vd. [19] tarafından yapılan çalışmada dikkat tabanlı kodlayıcı-kod çözücü modellerde karakter düzeyinde temsiller oluşturmanın performansı genel olarak iyileştirdiği düşüncesine varmışlardır.

3. Yöntem (Method)

3.1. Veri Kümesi (Dataset)

Bu çalışmada yazar Sabahattin Ali'nin İçimizdeki Şeytan, Kuyucaklı Yusuf, Kürk Mantolu Madonna, Merkopaşa Yazıları, Yeni Dünya ve Kamyon eserlerinden oluşturulan Türkçe veri seti üzerinde çalışma yapılmıştır. Yazarın eserleri birleştirilerek tek bir "txt" uzantılı metin dosyası haline getirilmiştir. DDİ alanında yapılan çalışmalarda metin verilerinin istenen seviyede işlenebilir düzeyde olmamasından dolayı veri seti veri ön işleme adımlarından geçirilerek çalışmaya uygun bir düzeye getirilmektedir. Bu şekilde düzenlenmiş ve temizlenmiş bir veri seti üzerinde yapılan çalışmalar daha iyi sonuçlar vermektedir. Veri temizleme işlemleri Python programlama dili ve NLTK kütüphanesi kullanılarak gerçekleştirilmiştir. Kullanılan veri setinden tüm noktalama işaretleri, html etiketleri, url, @ işareti ve istenmeyen diğer özel işaretler kaldırılmış ve tüm harfler küçük harfe çevrilmiştir.

3.2. Kullanılan Modeller (Used Models)

3.2.1. Kodlayıcı ve kod çözücüler (Encoders and decoders)

DDİ uygulamalarında 2014 yılından itibaren kullanılmaya başlanılan kodlayıcı-kod çözücü modeller günümüze kadar etkili bir şekilde kullanılmaktadır. Bu yöntem sayesinde metin üretiminde çok daha uzun metin dizelerinin işlenmesinde başarılı sonuçlar alınmaktadır. Metin verilerinde kullanılan kelimelerin zaman ve sıra ilişkisinin büyük önemi bulunduğundan dolayı bu yöntemler sıra ve zaman ilişkilerinin yakalanmasında etkili bir yöntem olarak kullanılmaktadır.

Şekil 1'de kodlayıcı-kod çözücü modellerinin genel yapısı görülmektedir. Modelin yapısında kodlayıcı-kod çözücü olmak üzere iki ana birim bulunmaktadır. Bu birimlerin yapısında genellikle LSTM ağları farklı sayıda ve farklı parametre değerlerinde kullanılarak model tasarımı yapılmaktadır. Modelde gösterilen x değerleri girdi değerlerini y değerleri ise çıktı değerlerini ifade etmektedir. Modelin yapısında bulunana kodlayıcı birimi girdi verisini işleyerek girdi verisinin özetinin elde edildiği bir içerik vektörü oluşturmaktadır. Elde edilen bu içerik vektörü kod çözücüye aktarılarak kod çözücünün girdi verisi olarak kullanılmaktadır. Buradaki en kritik nokta bu içerik vektörünün girdi verisiyle ilgili tüm bilgileri içermesindeki zorlukta yatmaktadır. Çünkü bu bilgi kod çözücünün doğru üretim yapabilmesi için kodlayıcıdan aldığı tek bilgidir

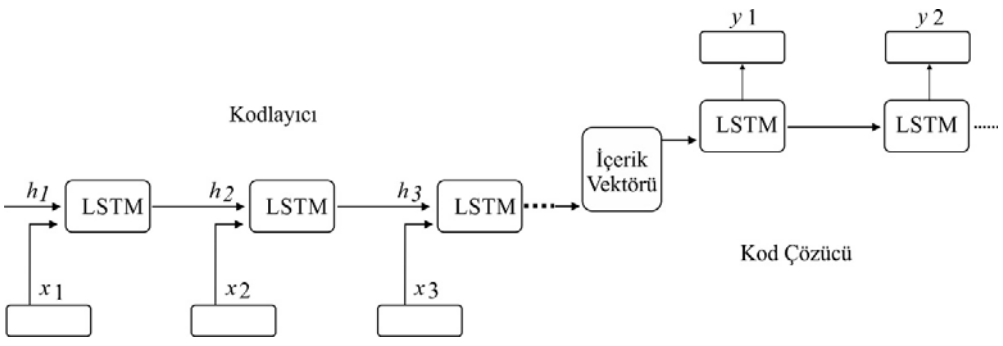
3.2.2. Dikkat mekanizması (Attention mechanism)

Kodlayıcı-kod çözücü modellerinde kod çözücüye sadece girdi verisiyle ilgili bilgi kodlayıcının elde ettiği sabit uzunluktaki içerik vektörü tarafından aktarılmaktadır. Yani çıktı verisinin girdi değeriyle doğrudan bir ilişkisi bulunmamaktadır. Dolayısıyla uzun metin verilerinde çalışırken bu modeller kötü sonuçlar vermektedir. Kodlayıcı-kod çözücü modellerde çıktı değeri en son LSTM hücresi ile bir önceki LSTM hücresinin gizli durum ve hücre durumu bilgileri üzerinden tahmin edilmektedir. Bu durumda geriye doğru gidilerek girdi verisiyle ilgili bilgiler sadece içerik vektörü üzerinden alınabilmektedir [20]. Bunun sonucunda metin verilerinin dizi boyutu uzadığında ilk değer ile son çıktı değerleri arasındaki ilişkiler unutulmaktadır. Kodlayıcı-kod çözücü modellerde daha uzun dizelerde başarılı sonuçlar alabilmek için bu modellerin yapısında dikkat mimarisi kullanılmaktadır. Dikkat mimarisi sayesinde kod çözücü bir tahminde bulunduğunda kodlayıcıdaki LSTM hücrelerinin bütün gizli durum bilgileri gözden geçirilmektedir. Metin verilerinin uzunluğu artsa bile dikkat mimarisi sürekli olarak üretilen her değerde girdi değerlerini sürekli olarak kontrol ettiği için üretim aşaması daha başarılı sonuçlanmaktadır [21]. Bu yöntem sayesinde uzun metin verilerinin işlenmesi aşamasında girdi ile çıktı değerleri arasındaki ilişkiler daha iyi hatırlanmaktadır.

Şekil 2'de kodlayıcı-kod çözücü modelde dikkat mekanizmasının nasıl kullanıldığı gösterilmektedir. Dikkat mimarisi sayesinde kodlayıcı artık sadece son gizli ve hücre durumlarını değil, aynı zamanda tüm zaman adımlarında üretilen kodlayıcının tüm gizli durumlarını da kod çözücüye aktarmaktadır. Kod çözücü ise çıktı oluşturmak için ilk giriş ve kodlayıcının tüm gizli durumlarını kullanarak dikkat vektörünü hesaplamaktadır. Daha sonra dikkati kodlayıcının tüm gizli durumlarına uygulayarak bağlam vektörünü hesaplar. Kod çözücü ise girdisini oluşturmak için bağlam vektörü ve ilk girişi birleştirir. Sonra model bu şekilde bir döngüde halinde çalışmaktadır [22].

3.3. Modelin Oluşturulması (Creating the Model)

DDİ uygulamalarında metin veri seti içerisinde kullanılan anlamlı her bir parçaya token adı verilmektedir. Metinde parçasında bulunan



Şekil 1. Kodlayıcı ve kod çözücü model (Encoder and decoder model)

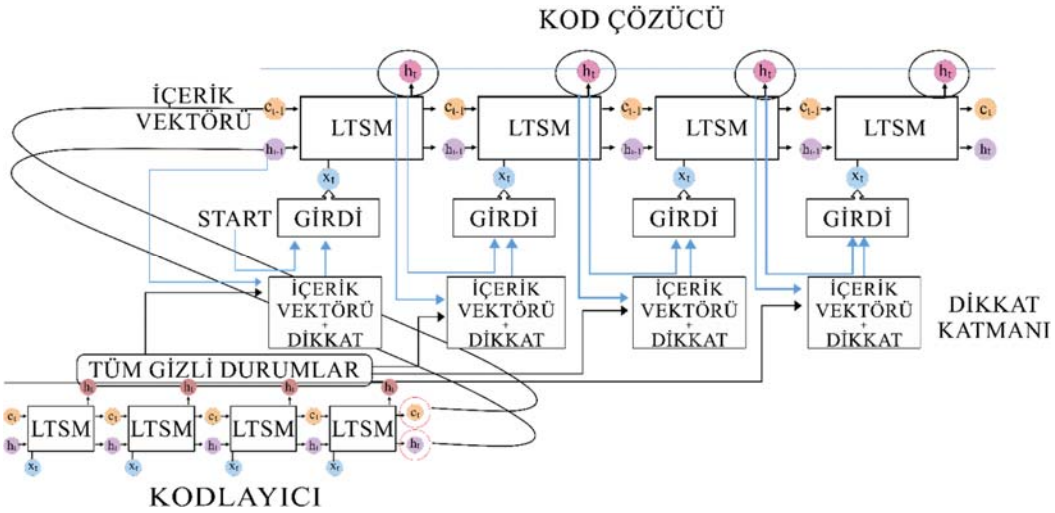
harfler, heceler, kelimeler, kelime grupları ve cümleler token olarak ifade edilmektedir. Metin tokonizasyonu ise metin verisinin istenen token birimlerine ayrılmasını ifade etmektedir. Dil modelleri eğitilirken metin verisinin dil modellerinin işleyebileceği tokenlere ayrılması gerekmektedir. Metin üretiminde hangi token yapılarının kullanılacağı kullanılan veri setine ve kullanılan uygulamalara göre değişiklik göstermektedir. Metin üretiminde genel olarak karakterler, n-gramlar ve kelimeler kullanılarak metinler üretilmektedir [23-27]. Dil modeline verilecek token birimlerinin oluşturulmadan önce veri setinin istenmeyen verilerden temizlenmesi gerekmektedir. Daha sonra dil modelinin eğitimi için gerekli olan token boyutuna karar verilmelidir. Metinde bulunan çok farklı kelime veya karakterlerden oluşacak bir sözlük elde etmek gerekmektedir. Oluşturulacak sözlük boyutu veri setinin büyüklüğüne modelin yapısına ve kullanılan bilgisayar donanımının gücüne bağlı olarak değişmektedir. Sözcük dağarcığının boyutu belirlendikten sonra bu tokenlerin sayısal olarak nasıl temsil edilmesi gerektiğinin belirlenmesi gerekmektedir. Bu aşamada farklı embeding yöntemleri kullanılarak metin tokenlerinin sayısal vektörlerle temsil edilmesi gerekmektedir. Yukarıda belirtilen adımlardan sonra dil modelinin kullanabileceği bir formata dönüşen veri seti derin sinir ağlarına verilerek metin üretimi gerçekleştirecek bir dil modeli haline gelmektedir

3.3.1. Dil modeli (Language model)

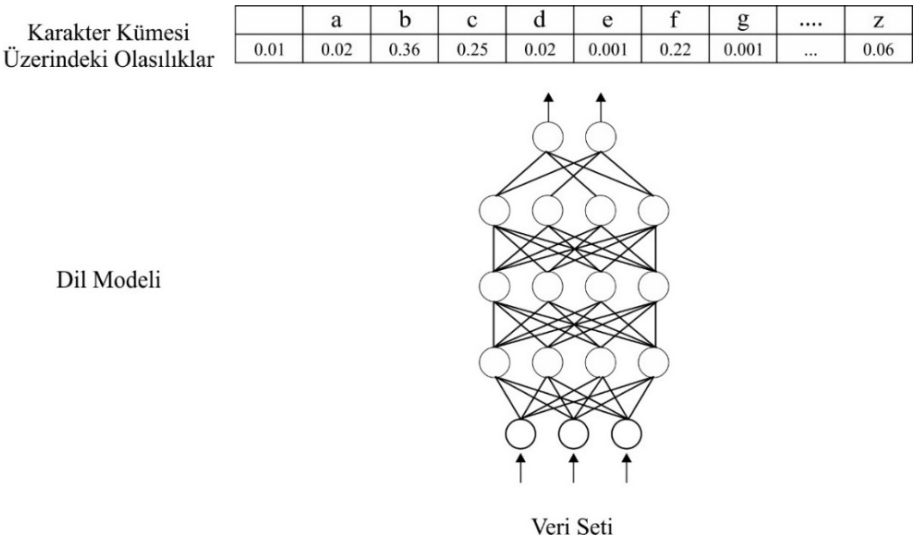
Dil modelleri bir DDİ uygulamasının merkezinde bulunan ana bileşenlerdir. Metin üretimi gibi DDİ uygulamaları geliştirmek için derin öğrenme algoritmaları yardımıyla bir dil modelleri eğitmek gerekmektedir. Dil modeli token haline getirilmiş veri setinin derin öğrenme algoritmaları yardımıyla istatistiksel yöntemler kullanarak oluşturulan ve metin üretimi gerçekleştiren eğitilmiş hazır bir modeldir. Metin tabanlı bir dil modeli eğitilirken veri setinde bulunan cümleler, kelimeler, harfler ya da n-gram denilen kelime ya da harf grupları kullanılmaktadır.

Şekil 3'de bir dil modelinin sözlüğünde bulunan karakterlerin şartlı olasılık değerleri gösterilmektedir. Bir dil modeli eğitilirken metin içerisinde bulunan tüm karakterlerin derin öğrenme algoritmalarına verilerek tüm karakterlerin anlamsal, sırasal ve birliktelik ilişkileri istatistiksel olarak hesaplanmakta ve öğrenilmektedir. Bu şekilde öğrenen dil modeli verilen metin parçasına göre kendi sözlüğünde bulunan karakterlerin şartlı olasılık değerlerini hesaplayarak metin sonuna gelecek olası karakteri tahmin etmektedir [24-28].

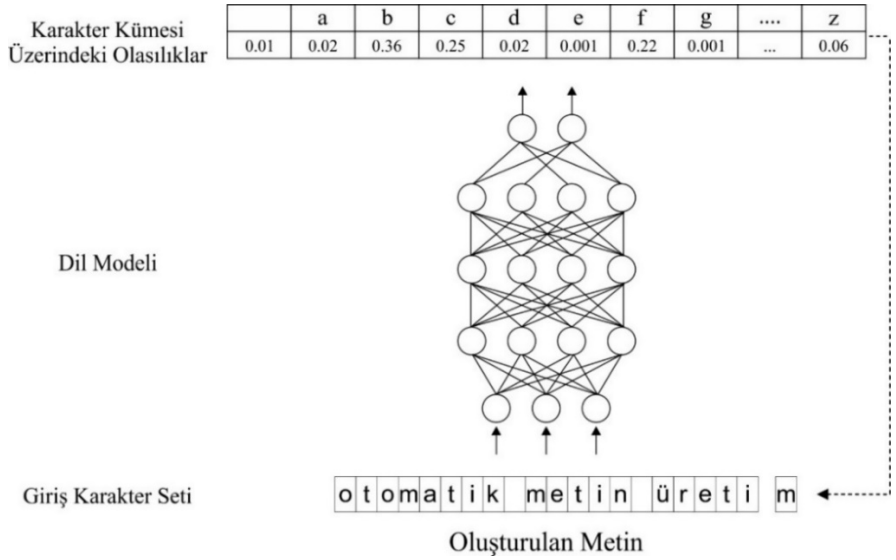
Şekil 4'te dil modelinin eğitimi bittikten sonra model kendisine verilen metin parçasına göre kendi sözlüğünde bulunan karakterlerin



Şekil 2. Kodlayıcı-kod çözücü ve dikkat modeli (Encoder decoder and attention model)



Şekil 3. Karakter tabanlı dil modeli (Character-based language model)



Şekil 4. Karakter tabanlı dil modeli çalışma örneği (Character based language model working example)

şartlı olasılık değerlerine bakarak ve örnek alma yöntemlerinden sıcaklık örnek alma yöntemini kullanarak metin sonuna gelebilecek en olasılıklı karakteri getirmektedir. Elde edilen metin dizisi tekrar dil modeline verilerek benzer adımlar diğer karakterler içinde döngüsel olarak yapılmakta ve metin üretimi ardışık olarak gerçekleşmektedir.

3.3.2. Örnek alma yöntemleri (Sampling methods)

Metin üretimi adımlarından biri de örnek alma yönteminin belirlenmesidir. Örnek alma yöntemleri kullanılarak dil modeli kendisine verilen metin parçasına göre sözlüğünde bulunan karakterlerin şartlı olasılık değerlerini hesaplayarak en olasılıklı karakteri verilen metnin bir sonraki karakteri olarak belirlemektedir. Metin üretimi esnasında kullanılan farklı örnek seçme yöntemleri bulunmaktadır. Çalışmada örnek alma yöntemi olarak sıcaklık örnek alma yöntemi (algoritma) kullanılmaktadır.

3.3.2.1. Sıcaklık örneklemesi (Temperature sampling)

Sıcaklık örneklemesi yöntemine göre yüksek sıcaklık değerleri düşük enerji seviyelerini ifade etmektedir. Bu yöntemde dil modelinin sözlüğünde bulunan karakterlerin, belirlenen sıcaklık değerlerine göre normalizasyon işlemlerine tabi tutularak olasılığı düşük olan karakterler dil modelinin sözlüğünden çıkarılarak yüksek seviyede olasılığa sahip olan karakterlerin belirlenmesi sağlanmaktadır. Bu yüksek olasılığa sahip karakterler daha sonra softmax aktivasyon fonksiyonundan geçirilerek değerler 0-1 olasılık aralığına getirilmekte ve bu değerler içinden örnek seçme işlemleri yapılmaktadır [25]. Bu yöntemde kullanılan sıcaklık eşik değerlerinin büyük önemi bulunmaktadır. Sıcaklık eşik değerleri küçük seçildiğinde daha az aday içerisinden seçim yapılmakta iken, sıcaklık değerleri yüksek seçildiğinde ise daha fazla aday içerisinden seçimler yapılmaktadır. Sıcaklık eşik değerlerinin çok düşük tutulması yöntemin sürekli olarak olasılık değeri en yüksek karakteri seçmesine neden olurken, sıcaklık eşik değerlerinin çok yüksek seçilmesi ise rastgeleliği arttırdığı için çok anlamsız metinler üretilmesine sebep olmaktadır. Dolayısıyla iyi sonuçlar alabilmek için sıcaklık eşik değerinin orta değerlerde tutulması gerekmektedir. En doğru değeri bulabilmek için ise birçok sıcaklık değeri ile uygulama üzerinde deneme yapmak gerekmektedir [25, 26].

3.3.3. Kodlama mimarisi (Coding architecture)

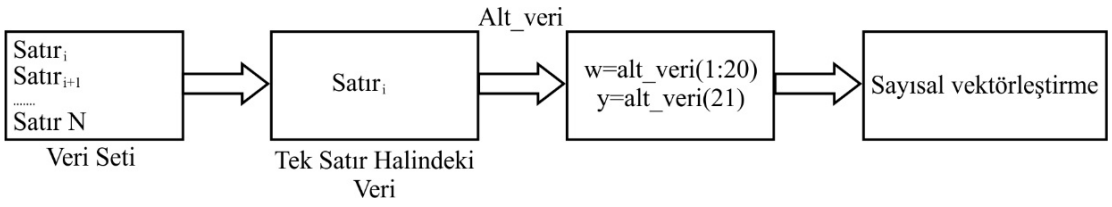
Karakter tabanlı kelime üretimi uygulaması google colap ide'si, phyton programlama dili, phyton üzerinde kullanılan derin öğrenme kütüphaneleri (tensorflow, keras) kullanılarak oluşturulmaktadır. Keras'ın ön işleme katmanı olan textvectorization katmanı metin önişleme adımlarını yapabilmek için kullanılırken, tensorflow kütüphanesinin veri hattı otomatikleştirilmiş olarak veriyi işlememizi sağlayan bir yaklaşım olarak kullanılmaktadır. Tanımlanan veri hattı sayesinde verinin çıkarılması, dönüştürülmesi ve doğruluğunun kontrol edilmesi gibi uçtan uca bir bağlantı sağlanarak işlemlerin otomatik olarak gerçekleştirmesi sağlanmaktadır.

Bir veri hattında şu işlemler yapılmaktadır:

- İlk olarak veri hattına veri seti eklenmektedir.
- Veri seti karakterlere dönüştürülmektedir.
- Metin içindeki noktalama işaretleri, html etiketleri, beyaz boşluklar vb. gibi istenmeyen karakterler kaldırılarak veri seti temizlenmektedir.
- Karakter dizileri giriş (x) ve çıkış (y) çiftleri haline dönüştürülmektedir.
- Girdi (x) ve çıktı (y) birleştirilerek tek bir veri seti haline getirilmektedir.
- Karakterler bilgisayarın anlayacağı sayısal değerlere (embeding) dönüştürülmektedir.
- Modelin eğitimi için veri, dil modeline verilmektedir.

Şekil 5'te veri hattı yapısının işleyişi görülmektedir. İlk olarak veri seti okunmakta daha sonra metin hatasız karakterlere ayrılması için tek satır haline getirilmektedir. Metin önişleme adımları yapılarak istenmeyen karakterler kaldırılmaktadır. Veri seti girdi ve çıktı verisi olarak iki parçaya ayrılmaktadır. İlk yirmi karakteri girdi (x) olarak almakta, yirmi birinci karakteri ise çıktı (y) değeri olarak almaktadır. Daha sonra her bir karakter bir tamsayı ile ifade edilerek modelin işleyeceği bir formata dönüştürülmektedir.

Şekil 6'da kodlayıcı-kod çözücü ve dikkat modelinin çalışma parametreleri görülmektedir. Modelin 20'lik bir girdi değeri aldığı, embedding katmanında her bir girdinin 200'lük karakter temsil vektörü haline dönüştürüldüğü görülmektedir. Kodlayıcı ve kod çözücüde bulunacak LSTM hücre sayısı 512 olarak belirlenmektedir. Dikkat mekanizmasının 512 olan değeri LSTM hücrelerinin tüm gizli durum bilgilerinin işlemlere katıldığını göstermektedir. Çıktıya bakıldığında batch size değerinin 512 olduğu ve her iterasyonda 1 karakterin tahmin edildiği görülmektedir. Modelin yaklaşık 4,5 milyon parametresi bulunmaktadır. Model bu toplam parametreleri işleyerek öğrenmeyi gerçekleştirmektedir.



Şekil 5. Kodlama veri hattı (Coding data line)

```
Model: "model_encoder_decoder"
_________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
=================================================================================================
 encoder_inputs (InputLayer)    [(None, 20)]         0           []

 embedding_1 (Embedding)        (None, 20, 200)      40000       ['encoder_inputs[0][0]']

 encoder_lstm (LSTM)            [(None, 20, 512),    1460224     ['embedding_1[0][0]']
                                 (None, 512),
                                 (None, 512)]

 bahdanau_attention_1 (Bahdanau  ((None, 512),       525825      ['encoder_lstm[0][1]',
 Attention)                      (None, 20, 1))                   'encoder_lstm[0][0]']

 tf.expand_dims_2 (TFOpLambda)  (None, 1, 512)       0           ['bahdanau_attention_1[0][0]']

 tf.concat_1 (TFOpLambda)       (512, 1, 712)        0           ['tf.expand_dims_2[0][0]']

 decoder_lstm (LSTM)            [(512, 512),         2508800     ['tf.concat_1[0][0]',
                                 (None, 512),                     'encoder_lstm[0][1]',
                                 (None, 512)]                     'encoder_lstm[0][2]']

 decoder_dense (Dense)          (512, 200)           102600      ['decoder_lstm[0][0]']

 tf.expand_dims_3 (TFOpLambda)  (512, 1, 200)        0           ['decoder_dense[0][0]']

 lambda_1 (Lambda)              (512, 1, 200)        0           ['tf.expand_dims_3[0][0]']

=================================================================================================
Total params: 4,637,449
Trainable params: 4,637,449
Non-trainable params: 0
```

Şekil 6: Kodlayıcı-kod çözücü ve dikkat modeli parametreleri (Encoder decoder and attention model parameters)

4. Bulgular (Results)

Çalışmada kullanılan kodlayıcı-kod çözücü ve dikkat algoritması kendisine verilen Türkçe veri setini kullanarak kendi sözlüğünü oluşturmaktadır. Model kendisine verilen bir metnin ilk 20 karakterini kullanarak bir sonraki karakterin hangi karakter olacağını tahmin etmeye çalışmaktadır. Model bunu oluşturduğu kendi sözlüğündeki karakterler içerisinden, verilen metin parçasını dikkate alarak yapmaktadır. Bunu yaparken sözlüğündeki en olasılıklı karakteri girilen metin parçasına göre bir sonraki karakter olarak tahmin etmekte ve karakterlerin birbiriyle ilişkilerini öğrenerek anlamlı kelimeler üretmeye çalışmaktadır. Aşağıda modele verilen örnek metin parçası gösterilmektedir.
"Merak etmeye başladım."

Yapılan literatür çalışmalarına paralel olarak bu çalışmada elde edilen sonuçların başarı seviyelerini ölçebilmek için "Accuracy" başarı ölçütü baz alınarak hesaplama yöntemi elde edilmeye çalışılmıştır. Accuracy değeri modelde doğru tahmin edilen kelime sayısının üretilen toplam kelime sayısına oranı ile hesaplanmaktır. Buna göre; doğru tahmin edilen kelime sayısı (DTEKS) ve yanlış tahmin edilen kelime sayısı (YTEKS) ölçütleri modelin ürettiği sonuçların güvenirlik düzeylerinin (G.D) belirlenmesi için kullanılmaktadır. Elde edilen güvenirlik düzeyi formülü Eş. 1'de gösterildiği gibidir.

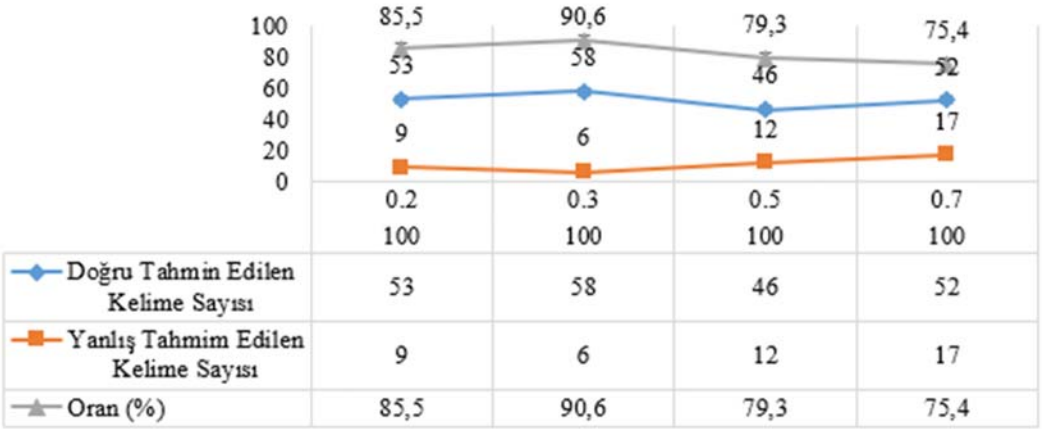
$$G.\mathrm{D} = \frac{DTEKS}{DTEKS+YTEKS} \quad (1)$$

Tablo 1'de karakter tabanlı kodlayıcı-kod çözücü ve dikkat modelinin 100 epoch ve sıcaklık örnek alma yönteminin farklı eşik değerlerinde elde edilen sonuçları görülmektedir. Tablo 1 incelendiğinde modelin kısa ve uzun anlamlı kelimeler oluşturmada başarılı sonuçlar verdiği görülmektedir. Ayrıca yanlış tahmin edilen kelimeler incelendiğinde bazı kelimelerin bir iki harfle ("sarışıp", "döğün") yanlış olarak tahmin edildiği görülmektedir. Modelin aynı zamanda bazı anlamlı kelime gruplarını başarılı bir şekilde ürettiği görülmektedir. Bunlar: "yok mu dedim", "iki insan gibi", "arkadan atların sesi", "başını çekip baktı ve ayağına", "birden iki garson", "hesabı gördün mü", "mazur görünmez", "kestirmeye başladı", "yok mu deli gibi köylerde", "hiç durmadan", "kımıldamadan gördükleri", "güzel bakmaya başladılar", şeklindeki kelime gruplarıdır.

Şekil 7'de kodlayıcı-kod çözücü ve dikkat algoritmaları kullanılarak oluşturulan dil modelinin 100 epoch çalıştırıldığında sıcaklık örnek alma yönteminin farklı eşik değerlerinde üretilen kelime sayıları ve başarı yüzdeleri gösterilmektedir. Grafik incelendiğinde 100 epoch sonucunda anlamlı kelime üretiminde sıcaklık örnek alma yönteminin 0,2 eşik değerinde başarının %85,5 0,3 eşik değerinde %90,6, 0,5 eşik değerinde %79,3 ve 0,7 eşik değerinde ise %75,4 oranında başarı elde edildiği görülmektedir. Bu sonuçlar değerlendirildiğinde kullanılan Türkçe veri setinin büyüklüğünün sınırlı olmasına rağmen sonuçların

Tablo 1. Karakter tabanlı kodlayıcı-kod çözücü ve dikkat algoritması 100 epoch sonuçları
(Character based encoder decoder and attention detection 100 epoch results)

Model	Epoch	Sıcaklık	Anlamlı Kelimeler	Anlamsız Kelimeler
Dikkat Alg.	100	0,2	"merak", "etmeye", "başladım" ,"ama", "iş", "günde", "böylece","arkadaşı","olacaktı","yolcularının","büyük","bir","odada" ,"bir", "kere", "koşan", "bir", "toplayıp", "bir", "derecede", "para", "ile", "sarılmadı", "gibiydi", "diğer", "bir", "delikanlı", "parayı", "aldığı", "birkaç", "göründü", "ve", "kendisi", "de", "bir", "taş", "bulgur", "tesislerden", "kadar", "yanında" ,"veya", "kaçmaktan", "kurtulmaya", "başladı", "ve", "gözlerin", "yusuf ", "dalmaz", "memnun", "olmadı", "bizim", "ortalığın", "altındaki", "günlerde", "yıkamaya"	"söoyan", "loşanı", "tirk", "kalkıyordudiyecek" , "yutuvak", "hardiçe", "kandıgını", "ba", "kağnının"
Dikkat Alg.	100	0,3	"merak", "etmeye", "başladım", "ama", "işlerin", "yanında", "çok", "sürme", "ile" "ve", "onun", "gözünü", "çözdü", "kaldırımlardan", "bahsettiler", "ayık", "bir", "bütün", "anası", "vermek", "istiyordu", "yok", "mu", "dedim", "ya", "erimez", "lira", "veren", "arkasındaki", "iki", "insan", "biz", "yürümemek", "için", "birbirlerine", "boğuldular", "şeyler", "var", "mı", "parçalarının", "geçip", "dışarı" "çıkarak", "kalkıp", "gelen", "gözleri", "hapsine", "bakar" "ve", "önünde", "dikerek", "elinden", "arkadan", "atların", "sesi" ,"indi", "adım", "adım", "atılmış", "ve"	"Doda", "old", "atılayırdıbüstün", "hayısla", "kukların", "yeşerek"
Dikkat Alg.	100	0,5	"merak", "etmeye", "başladınız", "salladı", "yüzünden", "kaçardım", "birden", "iki", "garson", "köyde", "zorlayan", "üzerini", "satmadan", "da", "yiyeceklerinden", "beri", "hesabı", "gördün", "mü", "başladı", "ve", "onları", "arasında", "çalışan", "sözümü", "bizi", "oynattı", "ve", "gözlerini", "eline", "aldı", "ve", "ancak", "konuşmalarından", "ayır", "görmeye", "başladı", "ve", "hiç", "bir", "kimseye", "olması", "sevdi", "bile", "mazur", "görünmez", "kestirmeye", "başladı"	"yüzümdü", "kurunç", "sevvesın", "sarışıp", "döğün", "sayıltıyordu", "eyrisini", "hoplayacaktınbeklemeye" ,"buldularak", "eğşere", "idi", "selecek"
Dikkat Alg.	100	0,7	"merak", "etmeye", "başladım", "ama", "işlerin", "yanında", "çalıların", "suyu", "yerinden", "kalkarak", "sallanmadı", "aksini", "ve", "kokuyordu", "ve", "bu", "sırada", "yalnız", "kendimizi", "aldık", "kadar", "başka", "aman", "biz", "bekliyordu", "yok", "mu", "deli", "gibi", "köylerde", "en", "şeyi", "susturmuştur", "birini" ,"burda", "gecelerini", "kalkıyorlardı", "emine" ,"İlk", "almış", "ve", "ellerinin", "bir", "kenarına", "büsbütün", "arkasına", "atılıp", "bu", "sıralarda", "olacak", "kadar", "küçük"	"fırlatırlardıfil", "yarabilirdi", "aiy", "girmedildiğim", "haril", "suyalık", "büseterek", "müracaaat", "oynanın"



Şekil 7. Modelin 100 epoch sonucundaki değerleri (Values of the model at 100 epochs)

farklı eşik değerlerinde %75,4 ve %90,6 aralığında olduğu görülmektedir. Modelin en iyi sonucu %90,6 ile sıcaklık örnek alma yönteminin 0,3 eşik değeri için verdiği görülmektedir.

Tablo 2'de karakter tabanlı kodlayıcı-kod çözücü ve dikkat modelinin 200 epoch ve sıcaklık örnek alma yönteminin farklı eşik değerlerinde elde edilen sonuçlar görülmektedir. Modelin anlamlı kelimeler oluşturmada başarısının artarak daha iyi sonuçlar verdiği görülmektedir. Modelin 200 epoch değerinde anlamlı ürettiği kelime gruplarının uzunluğunun arttığı ve ilişkileri daha iyi yakaladığı görülmektedir. Bu kelime grupları ise şöyledir: "düşkün ve dalgın haline gelmişler ve", "bu gün de bunun", "delikanlı paralarını alıp", "evin başında oturan adamlara", "yazı yazmaya kalkarak çocukları", "muavinin bu sözlerinin altında", " kolay geldiği için", "sol elinden

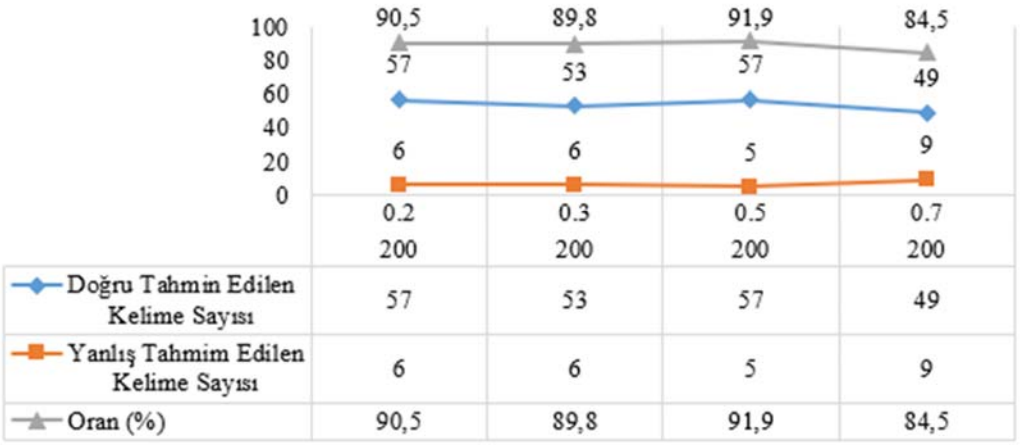
Tablo 2. Karakter tabanlı kodlayıcı-kod çözücü ve dikkat modeli 200 epoch sonuçları
(Character based encoder decoder and attention detection 200 epoch results)

Model	Epoch	Sıcaklık	Anlamlı Kelimeler	Manasız Kelimeler
Dikkat Alg.	200	0,2	"merak", "etmeye", "başladım", "ve", "bu", "ve", "fırıldayan", "diyor", "ve", "koca", "münevverler", "verdiğine", "inan", "eser", "de", "düşkün", "ve", "dalgın", "haline", "gelmişler", "ve", "derhal", "görünüyordu", "diye", "düşünerek", "arkasından", "ayıralım", "kadın", "sürükleyen", "bu", "gün", "de", "bunun", "dede", "üç", "müdürlü", "taraftaki", "karanlık", "alının", "sinirlendi", "ve", "delikanlı", "paralarını", "alıp", "yakaladı", "öğretmesi", "her", "şeye", "haline", "getirmek", "seçimden", "gelen", "bu", "evin", "başında", "oturan", "adamlara", "titremeye", "benziyor"	"sesiriydi", "çannesini", "boşaları", "bırakırızca", "silan", "tepisine"
Dikkat Alg.	200	0,3	"merak", "etmeye", "başladım", "ve", "bu", "sesi", "sualimin", "alsında", "bulunduğunu", "gören", "yazı", "yazmaya", "kalkarak", "çocukları", "karanlığı", "son", "zamanlardan", "biri", "için", "rastgele", "alimler", "almak", "istiyordu", "dedim", "birini", "yapraklarını", "bağladı", "üstüne", "bıraktı", "ve", "orada", "uzaklaştırmaktan", "sesi", "adım", "ediyordu", "muavinin", "bu", "sözlerinin", "altında", "kolay", "geldiği", "için", "karlar", "sallanan", "bu", "çocuklar", "bir", "sesle", "beklemişti", "taşıyan", "karımı", "şiddetle", "terlet", "çiftlere", "tahammül",	"yahalarından", "senirli", "kapalışarak", "al", "ayakka", "sarışsız"
Dikkat Alg.	200	0,5	"merak", "etmeye", "başladım", "ve", "bu", "sesi", "yazının", "önüne", "baktılar", "sanki", "bu", "kitapları", "var", "diye", "tozlar", "vuruyordu", "berker", "kendini", "anlayamadı", "ve", "onlardan", "biri", "gibi", "çıkıp", "gitti", "ve", "bir", "kadından", "beri", "de", "istasyona", "girmiş", "olan", "bir", dalga", "vardı", "işlerinde", "bir", "tutan", "olduğuna", "benziyordu", "ve", "sabahın", "aklı", "kadınlarının", "arasında", "ve", "ekmeği", "kadar", "yakından", "ayrılmıştım", "gönderdiği", "asla", "tarafından", "gösterdi", "ki", "çocuklar", "halinde", "yusuf"	"banya" "eziliyet" "sövektini" "münevası" "sesbiribirbir"
Dikkat Alg.	200	0,7	"merak", "etmeye", "başladım", "ve", "sessizce", "yüreği", "başını", "gösterdi", "ve", "genç", "kadın", "meydandadır", "tanıdığı", "için", "bu", "çocuklar", "bekliyordu", "saçlarında", "garip", "ağa", "tutuldu", "ve", "köyün", "bütün", "kolay", "koparak", "değildir", "ardından", "başka", "çarpan", "kış", "genç", "köşeye", "kadar", "telaş", "söylüyorlar", "diye", "noktası", "da", "bilmediği", "bir", "vaziyete", "sürüklemeye", "başladığı", "sırada", "ve", "hafifler", "rengi", "değil", "miydi", "ama"	"memkeleliği", "muhteh" ,"serbastan", "elte", "lakaların", "dörtkeye", "dokundan", "öylendirtiyorduona", "tavanfık"

ağzından bir şey olmadığı" "altından geçerken gezinip duruyor", " birbirlerinin oğlu gibi masum bir sürü", "ve onlardan biri gibi çıkıp gitti", "ve genç kadın meydandadır", "bilmediği bir vaziyete sürüklemeye başladığı sırada", "hamlelerden öğrendi", "bir kelime bile"

Şekil 8'de modelin 200 epoch çalıştırıldığında sıcaklık örnek alma yönteminin farklı eşik değerlerinde üretilen kelime sayıları ve modelin başarı yüzdelerini gösteren grafik görülmektedir. Grafik incelendiğinde 200 epoch sonucunda anlamlı kelime üretiminde sıcaklık örnek alma yönteminin 0,2 eşik değerinde başarının %90,5, 0,3 eşik değerinde %89,8 0,5 eşik değerinde %91,9 ve 0,7 eşik değerinde ise %84,5 oranında başarı elde edildiği görülmektedir. Sonuçlar incelendiğinde epoch sayısının arttırılıp 200 seviyesine çıkarılması sonucunda modelin anlamlı kelime üretim başarısının arttığı görülmektedir. Modelin en iyi sonucu sıcaklık örnek alma yönteminin 0,5 eşik değeri için %91,9 olarak vermektedir. Diğer sonuçların da bu değerlere yakın olduğu görülmektedir.

Tablo 3'de LSTM, GRU, kodlayıcı-kod çözücü ve dikkat algoritması kullanılarak karakter tabanlı olarak geliştirilen modellerin 100, 200 epoch değerlerinde ve sıcaklık örnek alma yönteminin farklı eşik değerleri için üretilen anlamlı ve anlamsız kelime sayıları ve modellerin başarıları yüzdeleri verilmektedir. LSTM ve GRU modelleri kullanılarak oluşturulan karakter tabanlı dil modellerinin sonuçları incelendiğinde; modeller 100, 200 epoch değerlerinde ve sıcaklık örnek alma yönteminin farklı eşik değerlerinde çalıştırılmaktadır. Alınan sonuçlar incelendiğinde anlamlı kelime



Şekil 8. Modelin 200 epoch sonucundaki değerleri (Values of the model at the result of 200 epochs)

Tablo 3. LSTM, GRU, kodlayıcı-kod çözücü ve dikkat modeli sonuçları (LSTM, GRU, encoder decoder and attention model results)

Yöntem-1	LSTM							
Epoch	100				200			
Sıcaklık	0,2	0,3	0,5	0,7	0,2	0,3	0,5	0,7
Anlamlı Kelime Sayısı	56	57	60	57	59	60	59	59
Anlamsız Kelime Sayısı	13	10	10	14	12	10	10	11
Başarı Oranı (%)	81,15	85,07	85,71	80,28	83,09	58,71	85,50	84,28
Yöntem-2	GRU							
Epoch	100				200			
Sıcaklık	0,2	0,3	0,5	0,7	0,2	0,3	0,5	0,7
Anlamlı Kelime Sayısı	57	59	60	55	59	61	60	58
Anlamsız Kelime Sayısı	10	8	8	12	10	8	9	13
Başarı Oranı (%)	85,07	88,05	88,23	82,08	85,50	88,40	86,95	81,69
Yöntem-3	Dikkat Mekanizması							
Epoch	100				200			
Sıcaklık	0,2	0,3	0,5	0,7	0,2	0,3	0,5	0,7
Anlamlı Kelime Sayısı	53	58	46	52	57	53	57	49
Anlamsız Kelime Sayısı	9	6	12	17	6	6	5	9
Başarı Oranı (%)	85,48	90,62	79,31	75,36	90,47	89,83	91,93	84,48

üretiminde LSTM ve GRU ile oluşturulan dil modelinin 100 ve 200 epoch değerlerinde ve örnek alma yönteminin farklı eşik değerlerinde yakın sonuçlar verdiği görülmektedir. Bu modellerden en yüksek başarı değerini 200 epoch ve 0,5 sıcaklık değerinde %88,40 ile GRU modelinin verdiği görülmektedir. Kodlayıcı-kod çözücü ve dikkat mimarisi kullanılarak oluşturulan karakter tabanlı dil modellerinin sonuçları incelendiğinde ise modelin 100, 200 epoch değerlerinde ve sıcaklık örnek alma yönteminin farklı eşik değerlerinde alınan sonuçlara göre anlamlı kelime üretiminde başarının LSTM yöntemine göre ortalamada %2,83 ve GRU yöntemine göre %0,19 oranında arttığı görülmektedir. Model en başarılı sonucu 200 epoch değerinde ve örnek alma yönteminin 0,5 eşik değerinde %91,93 ile verdiği görülmektedir.

5. Sonuçlar (Conclusions)

Bu çalışmada karakter tabanlı kelime üretimi uygulaması için yapay sinir ağları kullanılarak bir dil modeli geliştirilmekte ve verilen metin parçasına bağlı olarak kelimeler üretilmektedir. Dil modelinin oluşturulmasında özyinelemeli yapay sinir ağ modellerinden olan kodlayıcı-kod çözücü ve dikkat mimarisi kullanılmaktadır. Kodlayıcı-kod çözücü modellerinin yapısında LSTM ağları kullanılmaktadır. Bu ağlar uzun metin dizelerinde hafızaları yetmediği için metin üretiminde yeterli başarıyı gösterememektedir. Bundan dolayı Kodlayıcı-kod çözücü modellerinin yapısında LSTM ağları ile birlikte dikkat mekanizması kullanılarak bu sorunun üstesinden gelinmeye çalışılmaktadır. Dikkat mimarisi her işlem adımında girdi verilerinin farklı bölümlerine odaklanarak çıktı verisini üretmeye çalışmaktadır. Bu yapı sayesinde model ne ürettiğine bağlı olarak giriş verisinde nelere dikkat etmesi gerektiğini kendi kendine öğrenebilmektedir. Model bunu yaparken çıkış verisiyle giriş verisi arasında bir uygunluk skor değeri hesaplamaktadır. Bu skor giriş değerleri ile çıktı değerleri arasındaki ilişki düzeyini ortaya koyan bir puan olarak düşünülmektedir.

Kullanılan dil modelinin eğitim sürecinin tamamlanmasından sonra model 100 epoch ve 200 epoch değerlerinde çalıştırılmaktadır. Üretilen sonuçlar sıcaklık örnek alma yönteminin farlı eşik değerlerinden elde edilerek oluşturulmaktadır. Elde edilen sonuçlar incelendiğinde 100 epoch değerinde %79,31 ile %90,62 aralığında başarı elde edilmektedir. En iyi sonucun sıcaklık örnek alma yönteminin 0,3 eşik değerinde %90,62 olarak alındığı görülmektedir. Modelin doğru tahmin ettiği kelimelere bakıldığında kısa ve uzun kelimeleri oluşturmada başarılı olduğu ve bazı anlamlı kelime gruplarını oluşturabildiği görülmektedir. Model 200 epoch değerinde ise sıcaklık örnek alma yönteminin 0,5 eşik değeri için en yüksek

%91,93 değerini vererek modelin başarısını arttırdığı görülmektedir. Aynı zamanda model 200 epoch değerinde oluşturduğu anlamlı kelime ve kelime gruplarının uzunluğunun arttığı ve ilişkilileri daha iyi yakaladığı görülmektedir. Üretilen kelimelerin anlamlılığı ve modelin başarı yüzdeleri ele alındığında sonuçların oldukça tatmin edici olduğu düşünülmektedir.

Yapay zekâ uygulamalarının performansının veri setinin kalitesi ve büyüklüğü ile doğru orantılı olduğu bilinmektedir. Bu kapsamda Türkçe 'ye ait kaliteli ve yeterli büyüklükte veri setlerinin kısıtlı olması, çalışmalar incelendiğinde daha çok İngilizceye ait veri setlerinin bulunması bu çalışmanın en büyük kısıtlarından birini oluşturmaktadır. Türkçe dilinin kök, gövde ve ek yapısının diğer dillere oranla daha karmaşık olması sonuçların belirli başarı seviyesinde kalmasına neden olmaktadır. Literatür incelendiğinde metin üretimi ile ilgili daha çok kelime tabanlı yöntemler kullanılması ve karakter tabanlı kelime üretme çalışması yöntemlerinin daha az yer alması bu çalışmanın özgünlük ve öncüllük değerini yükseltmektedir.

Kullanılan dil modelinin sonuçlarını iyileştirmek için daha geniş ve kaliteli bir eğitim veri seti kullanmak, veri setini daha yüksek boyutlu kelime vektörleri ile temsil etmek, LSTM birim sayısını arttırmak, eğitim iterasyon (epoch) sayısını arttırmak, farklı örnek alma yöntemleri kullanmak, sıcaklık örnek alma yönteminin eşik değerinin hesaplanmasında üst sezgisel yaklaşımların kullanılması (genetik algoritmalar, parçacık sürü optimizasyonu, karınca kolonisi vb.) gibi iyileştirmelerin modelin başarısını attırabileceği düşünülmektedir.

Kaynaklar (References)

1. Kibble, R., Introduction to natural language processing, Undergraduate study in Computing and related programmes, University of London International Programmes, Department of Computing, 1 (2), 1-52, 2013.
2. Agarwal, M., An overview of natural language processing, International Journal for Research in Applied Science and Engineering Technology (IJRASET), 7, 2811-2813, 2019.
3. Özkan, İ., Ülker, E., 2017, Derin öğrenme ve görüntü analizinde kullanılan derin öğrenme modelleri, Gaziosmanpaşa Bilimsel Araştırma Dergisi, 6 (3), 85-104.
4. Khurana, D., Koli, A., Khatter, K., Natural language processing: state of the art, current trends and challenges, Multimed Tools Appl, 82 (3), 3713–3744, 2023.
5. Nadeau, D., Sekine, S., A Survey of Named Entity Recognition and Classification, Linguisticae Investigationes, John Benjamins Publisher Company, Holland, 30 (1), 3-26, 2007.
6. Dahl, D. A., Natural language processing: past, present and future, In Mobile speech and advanced natural language solutions, Springer, New York, 49-73. 2013.
7. Kostadinov S., Understanding Encoder-Decoder Sequence to Sequence Model. towardsdatascience.com. https://towardsdatascience.com/understanding-encoder-decoder-sequence-to-sequence-model-679e04af4346. Yayın tarihi Şubat 5, 2019. Erişim tarihi Mayıs 20, 2022.
8. Wang Z., Su X., Ding Z., Long-Term Traffic Prediction Based on LSTM Encoder-Decoder Architecture, in IEEE Transactions on Intelligent Transportation Systems, 22 (10), 6561-6571, 2021.
9. Alqahtani, H., Kavakli-Thorne, M., & Kumar, G., Applications of generative adversarial networks (gans): An updated review, Archives of Computational Methods in Engineering, 28 (2), 525-552, 2021.
10. Poulos, J., & Valle, R., Character-based handwritten text transcription with attention networks, Neural Computing and Applications, 33 (16), 10563-10573, 2021.
11. Eriguchi, A., Hashimoto, K., & Tsuruoka, Y, Character-based decoding in tree-to-sequence attention-based neural machine translation, In Proceedings of the 3rd Workshop on Asian Translation, 175-183, 2016.
12. Feng, Y., Hu, C., Kamigaito, H., Takamura, H., & Okumura, M., Improving Character-Aware Neural Language Model by Warming Up Character Encoder under Skip-gram Architecture, In Proceedings of the International Conference on Recent Advances in Natural Language Processing, 421-427, 2021.
13. Renduchintala, A., Shapiro, P., Duh, K., and Koehn, P., Character-aware decoder for translation into morphologically rich languages, In Proceedings of Machine Translation Summit XVII, 1, 244-255, 2019.
14. Yang, Z., Chen, W., Wang, F., & Xu, B., A character-aware encoder for neural machine translation, In Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics, Osaka-Japan, 3063-3070, 11-17 Aralık, 2016.
15. Bahdanau, D., Chorowski, J., Serdyuk, D., Brakel, P., & Bengio, Y., End-to-end attention-based large vocabulary speech recognition, In 2016 IEEE international conference on acoustics, speech and signal processing (ICASSP), Shanghai-China, 4945-4949, 20-25 Mart, 2016.
16. Meng, Z., Gaur, Y., Li, J., & Gong, Y., Character-Aware Attention-Based End-to-End Speech Recognition. 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), Singapore, 949-955, 14-18 Aralık, 2019.
17. Kwon, D., Kim, H., Kim, J., Suh, S. C., Kim, I. ve Kim, K. J., A survey of deep learning-based network anomaly detection. Cluster Computing, 22 (1), 949- 961, 2019.
18. Noord, R.V., Toral, A., & Bos, J., Character-level representations improve DRS-based semantic parsing Even in the age of BERT, In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online. Association for Computational Linguistics, 4587–4603, 2020.
19. Otter, D. W., Medina, J. R., & Kalita, J. K., A survey of the usages of deep learning for natural language processing, IEEE transactions on neural networks and learning systems, 32 (2), 604-624, 2020.
20. Chen Y., Li F., Wang J., Tang B., Zhou X., Quantum recurrent encoder–decoder neural network for performance trend prediction of rotating machinery, Knowledge-Based Systems, 197, 2020.
21. Khandelwal, Renu, Attention: Sequence 2 Sequence model with Attention Mechanism. https://towardsdatascience.com/sequence-2-sequence-model-with-attention-mechanism-9e9ca2a613a. Yayın tarihi Ocak 20, 2020. Erişim tarihi Ekim 2, 2022.
22. Niu, Z., Zhong, G., & Yu, H., A review on the attention mechanism of deep learning, Neurocomputing, 452, 48-62, 2021.
23. Karakaya, M., Sampling in Text Generation. https://medium.com/deep-learning-with-keras/sampling-in-text-generation-b2f4825e1dad. Yayın tarihi Mart 7, 2021. Erişim tarihi Eylül 25, 2022.
24. Stokes, J., A guide to language model sampling in AllenNLP. https://blog.allenai.org/a-guide-to-language-model-sampling-in-allennlp-3b1239274bc3 Yayın tarihi kasım 18, 2020, Erişim tarihi Ekim 5, 2022.
25. Mann, B. How to sample from language models. https://towardsdatascience.com/how-to-sample-from-language-models-682bceb97277. Yayın tarihi Mayıs 25, 2019. Erişim tarihi Ekim 10, 2022.
26. Renggli, C., Rimanic, L., Gürel, N. M., Karlaš, B., Wu, W., & Zhang, C, A data qualitydriven view of mlops, IEEE Data Eng, Bull, 44 (1) 11–23, 2021.
27. Karaca A., Aydın Ö., Generating headlines for Turkish news texts with transformer architecture-based deep learning method. Journal of Gazi University Faculty of Engineering and Architecture, 39 (1), 485-496, 2024.
28. Noyan T. , Kuncan F., Tekin R., Kaya Y., A new content-independent approach for document language recognition: Angle Patterns, Journal of Gazi University Faculty of Engineering and Architecture, 37 (3), 1277-1292, 2022.
29. Somuncu E., Aydın Atasoy N., Implementing character recognition on text images with a convolutional recurrent neural network. Gazi University Faculty of Engineering and Architecture Journal, 37 (1), 17-28, 2022.
30. Çakın, Ö., Post-Semiyotik Okurun İktidarı: Göstergelerin Bağlamsal Yolculuğu, Postlar Çağında İletişim, Editör M.N. Erdem-N.K. Şener, Nüve Kültür Yayınevi, Literatürk Academia, İstanbul, 105-123, 2019.