Avrupa Bilim ve Teknoloji Dergisi
Özel Sayı 32, S. 177-182, Aralık 2021

Araştırma Makalesi

www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
Special Issue 32, pp. 177-182, December 2021

Research Article

Kısa Metinlerin Sıkıştırılması için BERT Tabanlı bir Yöntem

Emir Öztürk[1*], Altan Mesut[2]

[1*] Trakya Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Edirne, Türkiye, (ORCID: 0000-0002-3734-5171), emirozturk@trakya.edu.tr
[2] Trakya Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Edirne, Türkiye (ORCID: 0000-0002-1477-3093), altanmesut@trakya.edu.tr

(International Conference on Design, Research and Development (RDCONF) 2021 – 15-18 December 2021)

(**DOI:** 10.31590/ejosat.1039450)

ATIF/REFERENCE: Öztürk, E., Mesut, A. (2021). Kısa Metinlerin Sıkıştırılması için BERT Tabanlı bir Yöntem. *Avrupa Bilim ve Teknoloji Dergisi*, (32), 177-182.

Öz

Veri aktarımı ve saklanmasında veri sıkıştırma algoritmalarının kullanılması, aktarım süresi ve saklama maliyeti açısından avantaj sağlamaktadır. En çok üretilen veri türlerinden biri olan doğal dildeki metinlerin sıkıştırılması için farklı yöntemler bulunmaktadır. Geleneksel birçok yöntem kısa metinlerin sıkıştırılmasında başarı gösterememektedir. Kısa metinlerin sıkıştırılması için genel amaçlı sıkıştırma yöntemlerinden daha farklı yöntemlere ihtiyaç duyulmaktadır. Bu çalışmada BERT'in tahmin mekanizmasını kullanan bir kısa metin sıkıştırma algoritması önerilmiş ve geleneksel yöntemler ile karşılaştırılmıştır. Ayrıca önerilen yöntemin başarısı farklı parametreler ve modeller için incelenmiş ve karşılaştırılmıştır. Önerilen yöntem Gzip, Bzip2 ve Zstd gibi bilinen algoritmalara göre %39'a kadar daha başarılı sıkıştırma oranları elde etmiştir.

Anahtar Kelimeler: Veri Sıkıştırma, Kısa Metin Sıkıştırma, BERT.

A BERT-Based Method for Compressing Short Texts

Abstract

Using data compression algorithms in data transmission and storage provides advantages in terms of time and storage cost. There are several methods for compressing texts created in natural language which is one of the most produced data types. Many traditional methods are not successful in compressing short texts. Compressing short texts requires different methods than general-purpose compression methods. In this study, a short text compression algorithm which uses the prediction mechanism of BERT is proposed and compared with traditional methods. In addition, the results of the proposed method were examined and compared for different parameters and models. The proposed method has achieved compression ratios up to 39% better than traditional algorithms such as Gzip, Bzip2 and Zstd.

Keywords: Data Compression, Short Text Compression, BERT.

[*] Sorumlu Yazar: emirozturk@trakya.edu.tr

1. Giriş

Veri sıkıştırma, mevcut veri saklama ortamlarının tümünde ihtiyaç duyulan bir işlemdir. Veri sıkıştırma sayesinde disk alanından ve özellikle akış tabanlı çalışan sistemlerde bant genişliğinden fayda sağlanabilmektedir (Mathews, 1995; Ziviani vd., 2000). Veri sıkıştırma yöntemleri görüntü, hareketli görüntü, metin verisi gibi veriler üzerinde uygulanabileceği gibi, hiperspektral ve medikal görüntüler ile tıp ve sağlık uygulamalarında da kullanılmaktadır.

Veri sıkıştırma yöntemleri kayıplı ve kayıpsız olarak kategorize edilir. Genellikle ses, görüntü ve video türündeki verilerin sıkıştırılması için kullanılan kayıplı yöntemlerde, silinmesi halinde insan gözünün ve kullağının bu eksilmeyi fazla hissedemeyeceği türdeki veriler atılır. Kalan veri uygun kayıpsz yöntemler ile sıkıştırılır. Metinler ve diğer ikili dosyalar ise sıkıştırılan veri geri açıldığında birebir orijinal veri elde edilmesi gerektiğinden sadece kayıpsız yöntemler ile sıkıştırılır.

Metin verisi üzerinde kullanılan kayıpsız yöntemler harfleri, n-gram'ları veya kelimeleri sembol (sıkıştırılacak en küçük birim) olarak kullanabilmektedir (Öztürk vd., 2017). Bu yöntemlerden olasılık tabanlı olanlar çok sık geçen sembollere kısa kodlar tanımlarken, sözlük tabanlı olanlar ise sık geçen sembollerin belirli bir kısmından sözlük oluşturarak bu sözlükteki sembollere sabit uzunlukta bir kod atama işlemi gerçekleştirmektedir. Sözlük tabanlı yöntemlerde kod uzunluğu sabit olduğundan dolayı sembol olarak harften ziyade n-gram (Nguyen vd., 2016) ya da kelimeleri seçerler (Öztürk vd., 2018).

Sıkıştırma yöntemlerinde veri boyutu azaldıkça sıkıştırma için gerekli olan tekrar eden veri de azalmakta, bu sebeple kısa metinlerin sıkıştırma oranı büyük metinlere göre daha düşük olmaktadır. Hatta bazı durumlarda sıkıştırma yerine genişletme gerçekleşmektedir (Platoš vd., 2008). Kısa metinlerin iyi oranda sıkıştırılabilmesi için bazı özel yöntemler de geliştirilmiştir (Aslanyürek ve Mesut, 2021; Gardner-Stephen vd., 2013; Sanfilippo, 2009; Schramm, 2013).

Transformer mimarileri metin özetleme, sınıflandırma, soru cevaplama ve metin üretme gibi görevler için sıklıkla kullanılmaktadır. Transformer mimarilerinden BERT (Bidirectional Encoder Representations from Transformers) 2018 yılında geliştirilmiştir (Devlin vd., 2018). RNN (Recurrent neural network) veya CNN'in (Convolutional neural network) aksine BERT, çift yönlü işleme gerçekleştirebilmektedir. Ayrıca BERT kullanılarak verilen bir metin için kelime tabanlı tahmin de gerçekleştirebilmektedir.

Bu çalışmada Türkçe BERT modelini kullanan kelime tabanlı bir kısa metin sıkıştırma yöntemi önerilmiştir. Önerilen yöntem, BERT modelinin maskeler ile metin içerisinde istediği konumdaki kelimeyi tahmin etme mekanizmasını kullanarak, kelimenin verilen tahmin listesindeki sırasını kodlamaktadır.

Çalışmanın ikinci bölümünde önerilen yöntem ile karşılaştırılan diğer sıkıştırma yöntemlerinden bahsedilmiş, üçüncü bölümde önerilen yöntem açıklanmış, dördüncü bölümde ise deneysel sonuçlar verilmiştir. Son bölümde ise sonuçlar irdelenmiştir.

2. Karşılaştırmada Kullanılan Yöntemler

Bu bölümde, deneysel sonuçların elde edilmesinde önerilen yöntem ile karşılaştırılan diğer bilinen yöntemler açıklanmıştır.

2.1. Gzip

Gzip, Jean-Ioup Gailly ve Mark Adler tarafından Unix Compress'e alternatif olarak geliştirilmiş kayıpsız bir sıkıştırma yöntemidir (Deutsch, 1996-1). Yöntemin temelinde Phil Katz tarafından geliştirilen DEFLATE (Deutsch, 1996-2) yer almaktadır. DEFLATE, verinin içindeki tekrar eden yapıları LZSS (Storer & Szymanski, 1982) algoritması ile azaltır ve ardından Huffman kodlamasını (Huffman, 1952) kullanarak sıkıştırma oranını daha da arttırır.

2.2. Bzip2

Bzip2 girdi verisini bloklara ayırıp bu blokları BWT (Burrows Wheeler Transform) ile dönüştüren bir yöntemdir (Seward, 1996). BWT (Manzini, 2001) üç ana işlemden oluşmaktadır. İlk olarak girdi verisi bloklara ayrılır ve sıralanır. Sıralama aşamasından sonra MTF (Move-To-Front) dönüşümü ile benzerliklerin arttırılması amaçlanır. Bu işlemlerden sonra son aşamada Huffman kodlama gerçekleştirilir.

2.3. Zstd (Zstandard)

Zstd, Yann Collet tarafından 2016 yılında geliştirilmiş kayıpsız bir sıkıştırma metodudur (Collet ve Kucherawy, 2018). DEFLATE'in sıkıştırma oranlarına çok daha hızlı bir şekilde ulaşmayı amaçlar. En yüksek sıkıştırma parametreleri ile LZMA oranlarına yakın sonuçlar verebilmektedir. Zstd LZ77 tabanlıdır ve klasik LZ77'ye göre çok daha büyük bir arama penceresi kullanır. Entropi kodlama aşamasında Finite State Entropy (FSE) ve ANS (Asymmetric numeral system) yönteminin tablolar kullanılarak hızlandırılmış versiyonu olan tANS'ı (Duda vd., 2015) ve Huffman kodlamayı birlikte kullanır.

2.4. Smaz

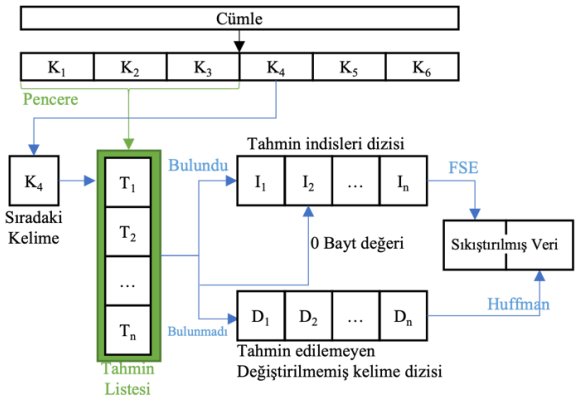
Smaz, kısa metinleri iyi oranlarda ve hızlı sıkıştırabilen Salvatore Sanfilippo tarafından geliştirilmiş basit bir yöntemdir (Sanfilippo, 2009). Küçük harflerle yazılmış İngilizce metinlere özgü statik bir sözlük içerdiği için diğer dillere göre İngilizce metinleri daha iyi oranda sıkıştırır. Sözlükte "http://" ve "</" gibi girdiler de olduğu için HTML ve URL'ler için de biraz sıkıştırma yapabilir.

2.5. Shoco

Shoco, kısa metinleri sıkıştırmak için Christian Schramm tarafından C dilinde yazılmış bir yöntemdir (Schramm, 2013). Varsayılan sıkıştırma modeli İngilizce kelimeler için optimize edilmiştir, ancak farklı girdi verileriyle başka bir sıkıştırma modeli de oluşturulabilir. İngilizce metinlerin sıkıştırılmasında entropi kodlayıcı olan Shoco, sözlük tabanlı sıkıştırma yapan Smaz'a göre genellikle daha kötü sıkıştırma oranları verir. Bununla birlikte, metinde sayılar gibi sözlükte yer almayan kelimeler çoksa Smaz metni genişletebilir. Shoco ise ASCII türü metinlerde hiçbir zaman genişletme yapmaz (ASCII olmayan karakterleri ise 2 kat büyütebilir).

3. Önerilen Yöntem

Önerilen yöntem, BERT'in kelime tahmin işlemine dayanmaktadır. BERT kullanılarak bir cümlede istenen bir kelime <MASK> etiketi ile gizlenebilmektedir. Daha sonra maskelenmiş bu cümle kodlama ve açma aşamalarından geçtikten sonra BERT, <MASK> etiketinin yerine bir kelime listesi sunar. Örneğin "Bugün hava çok <MASK>" gibi bir cümle için BERT "güzel", "güneşli", "yağışlı" gibi kelime tahminleri gerçekleştirirken, "Bu araba çok <MASK>" gibi bir cümle için "güzel", "hızlı", "rahat" gibi kelime tahminleri yapabilmektedir. Önerilen yöntemde amaç BERT'in tahmin ettiği bu kelime listesini kullanarak sıkıştırma gerçekleştirmektir. Maskelenen kelime sonda değil arada da olabilir. BERT, "Bugün <MASK> çok güzel" için "hava" ve "deniz" gibi kelimeler önerebilir. Fakat sıkıştırma amacıyla kullanıldığında son kelimenin tahmin edilmesi gereklidir. Bunun sebebi maske arada olursa açma aşamasında sonraki kelimelerin bilinememesi ve tahmin işleminin kullanılamayacak olmasıdır. Yöntemin sıkıştırma aşamaları Şekil 1'de verilmiştir.



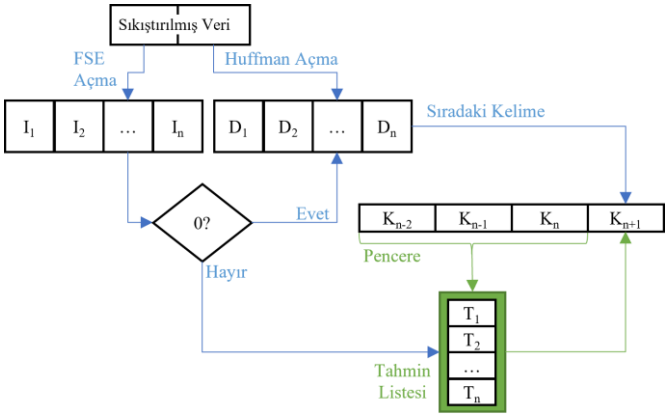
Şekil 1. Sıkıştırma Aşamaları

Sıkıştırma işleminin ilk aşamasında cümle kelimelere (K_1, K_2, …, K_n) ayrılmaktadır. Bu aşamada ayrıca noktalama işaretleri boşluklar ile kelimelerden ayrılmaktadır. Bu sayede kelime tahmininin uygun yapılabilme olasılığını arttırmak amaçlanmıştır. Kelime elde etme aşamasından sonra verilen pencere boyutu kadar kelime birleştirilip sonuna <MASK> etiketi eklenir. Hazırlanmış cümle BERT'e verilerek maske yerine kelime tahmini yapması sağlanır. BERT'in tahmin aşamasında tahmin edeceği kelime sayısı da verilebilmektedir. Bu sayede BERT maksimum verilen tahmin sayısı kadar uygun kelimeyi bir liste olarak döndürmektedir. Bunun anlamı, BERT'ten 100 kelime tahmin edilmesi istendiğinde en fazla 100 kelime tahmin etmesidir. Bazı durumlarda 100 kelimeden az tahmin etme ihtimali de bulunmaktadır.

Sıkıştırma işleminde iki adet kaçış değeri kullanılmaktadır. Birinci aşamada kelimelerin bölünmesi ile oluşan boş stringlerin kodlanması için 1, bulunamayan kelimelerin I dizisinde belirtilmesi için ise 0 indisi kullanılmaktadır.

Tahmin aşamasından sonra kodlanacak kelime alınır (örnek için K_4) ve BERT'in önerdiği liste içerisinde olup olmadığı kontrol edilir. Eğer sıradaki kelime BERT'in öneri listesinde bulunuyorsa, kelimenin tahmin listesindeki indisi kodların saklandığı bir diziye eklenir (I). Eğer kelime BERT'in öneri listesinde bulunmuyorsa, bu durumda I dizisine bir 0 değeri eklendikten sonra kelime açık bir şekilde, tahmin edilemeyen kelimelerin bulunduğu bir diziye (D) eklenir. Kodlama esnasında pencere boyutuna ulaşılmadığı sürece pencere büyütülmekte ve pencere boyutuna ulaşıldıktan sonra da kaydırma işlemi gerçekleştirilmektedir. Ayrıca açma aşamasında ilk tahmin olarak kullanılmak amacıyla ilk kelime sıkıştırılmadan D dizisine eklenmektedir. Sıkıştırma işlemi tamamlandıktan sonra I dizisi FSE ile, D dizisi ise Huffman kodlama ile sıkıştırılarak boyut bilgileri ile birlikte tek bir dizi haline getirilip kaydedilmektedir.

Açma aşamaları Şekil 2'de verilmiştir. Açma aşamasında öncelikle boyut bilgileri kullanılarak sıkıştırılmış verideki I ve D dizilerinin kodlanmış halleri ayrılır. Daha sonra kodlanmış olan I dizisi FSE, D dizisi ise Huffman açma yöntemleri ile açılarak elde edilir. D dizisindeki ilk kelime çıktıya yazıldıktan sonra indis değerleri sırasıyla okunur. Eğer indis değeri 0'sa D dizisinden sıradaki kelime okunup çıktı akışına yazılır. Her aşamada ayrıca pencere de güncellenmektedir. Eğer I dizisinde 0'dan farklı bir indis (i) bulunursa mevcut pencere sonuna bir <MASK> etiketi eklenerek BERT'e verilir ve tahmin listesi (T) yine tahmin sayısı kullanılarak elde edilir. Daha sonra bu tahmin listesinin elde edilen indis değerindeki kelime (T_i) çıktı dizisine yazılır. Bu sayede tüm indisler bitirildiğinde açılmış metin elde edilmiş olur.



Şekil 2. Açma Aşamaları

4. Deneysel Sonuçlar

Deneysel sonuçların elde edilmesinde SuDer Korpustaki (Sabancı Üniv. VERİM, 2018) Sabah ve Cumhuriyet dosyaları ile birlikte ODTÜ Türkçe derleminden (Say vd., 2002) metinler kullanılmıştır. Her dosya içerisinden 0-100, 100-200, …, 800-900 ve 900 bayttan büyük olmak üzere 10 farklı boyut değerinde 100'er satır rastgele olarak elde edilmiştir. Her veri seti için seçilen satırların uzunluk bilgileri Tablo 1'de verilmiştir.

Tablo 1. Veri setlerindeki satırların minimum, maksimum ve ortalama uzunlukları (bayt)

	Sabah	**Cumhuriyet**	**ODTÜ**
Minimum	20	20	21
Maksimum	2552	3411	1991
Ortalama	534,91	529,40	511,96

Sıkıştırma işleminde farklı pencere boyutlarının ve farklı sayıda kelime tahmininin yapılmasının da sıkıştırma oranına etkisinin incelenebilmesi amacı ile 5-10-15 uzunluğunda pencereler ve 64-128-254 tahmin sayıları kullanılarak sonuçlar elde edilmiştir. Sonuçlar Loodos'un "bert-base-turkish-cased" (Türkçe büyük-küçük harf duyarlı) modeli kullanılarak alınmıştır. Elde edilen sonuçlar Bölüm 2'de verilen diğer

sıkıştırma yöntemleri ile de karşılaştırılmıştır. Shoco TR, Türkçe korpus ile oluşturulan modeli kullanan versiyonu temsil etmektedir.

Sabah korpusu için elde edilen sıkıştırma oranları Tablo 2'de verilmiştir. W değeri penceredeki kelime sayısını veya diğer bir deyişle pencere boyutunu ifade ederken, P değeri ise tahmin edilen kelime sayısını ifade etmektedir. Tabloda görüldüğü gibi Smaz ve Shoco dışındaki algoritmalar metin boyutu arttıkça sıkıştırma oranlarını arttırmışlardır. İngilizce odaklı Smaz ve Shoco tüm metinlerde, diğer yöntemler ise küçük boyutlu metinlerde %100'ün üzerinde değerler elde etmiş, yani veriyi genişletmişlerdir. Türkçe test verisi ile eğitilen modeli kullanan Shoco TR ise 500 bayttan küçük metinlerde BERT modelinden sonra en iyi sonuçları vermiştir. Önerilen yöntem sadece 100-199 boyutundaki metinler için W=5 ve P=254 konfigürasyonunda Shoco TR'nin %4,3 oranında gerisinde kalmış, diğer tüm sonuçlarda en iyi oranları elde etmiştir. Genel amaçlı sıkıştırma yöntemlerinden Gzip ve Bzip2 200 bayt boyutundan büyük olan metinlerde sıkıştırma yapabiliyorken, Zstd ise 100-199 bayt boyutunda da sıkıştırma yapabilmiş ve 300 bayttan küçük metinlerde Gzip'e üstünlük sağlamıştır.

Tablo 2. Sabah veri seti için metin uzunluklarına göre sıkıştırma oranları

Metin Boyutu	**Gzip**	**Bzip2**	**Zstd**	**Smaz**	**Shoco**	**Shoco TR**	**BERT**								
							W=5			**W=10**			**W=15**		
							P=64	**P=128**	**P=254**	**P=64**	**P=128**	**P=254**	**P=64**	**P=128**	**P=254**
0-99	165,4	228,2	134,8	131,0	128,9	122,2	103,4	105,1	108,2	**102,8**	104,9	106,7	**102,8**	104,8	107,0
100-199	104,1	114,3	95,0	104,8	108,6	72,5	55,9	65,1	76,8	55,3	62,7	69,7	**54,4**	61,3	71,5
200-299	87,4	97,0	83,3	104,3	109,2	71,5	51,8	52,5	64,0	49,8	50,5	60,6	**49,5**	49,9	61,2
300-399	79,7	86,2	79,8	104,6	109,4	71,7	50,7	50,1	55,8	48,8	48,4	53,0	48,4	**48,0**	52,6
400-499	73,8	78,7	74,6	102,4	107,5	70,8	49,2	48,4	52,0	47,1	46,5	50,9	46,9	**46,0**	50,9
500-599	71,0	74,7	72,1	103,1	108,3	71,3	48,5	47,6	48,5	46,4	45,6	46,6	46,0	**45,2**	45,7
600-699	69,3	73,2	70,5	103,2	108,6	71,6	48,8	47,8	47,7	46,8	46,0	45,9	46,4	**45,6**	**45,6**
700-799	66,6	70,5	68,0	103,2	108,1	71,4	47,4	46,8	46,4	45,6	44,9	44,6	45,1	44,3	**44,0**
800-899	64,5	67,6	65,9	103,3	108,4	70,9	46,6	46,1	45,8	44,5	44,0	43,6	43,9	43,4	**43,1**
900-	60,4	62,0	61,9	104,0	108,9	71,4	45,6	44,6	44,2	43,6	42,6	42,2	43,3	42,2	**41,7**

Tablo 3'te Cumhuriyet veri seti için sıkıştırma oranı sonuçları gösterilmektedir. Bu veri setinde 0-99 bayt arasında en iyi sonucu Shoco TR vermiş, diğer boyutlarda yine önerilen yöntem daha başarılı olmuştur (yine sadece 100-199 boyutundaki metinler için W=5 ve P=254 konfigürasyonunda Shoco TR'nin gerisinde kalmıştır).

Tablo 3. Cumhuriyet veri seti için metin uzunluklarına göre sıkıştırma oranları

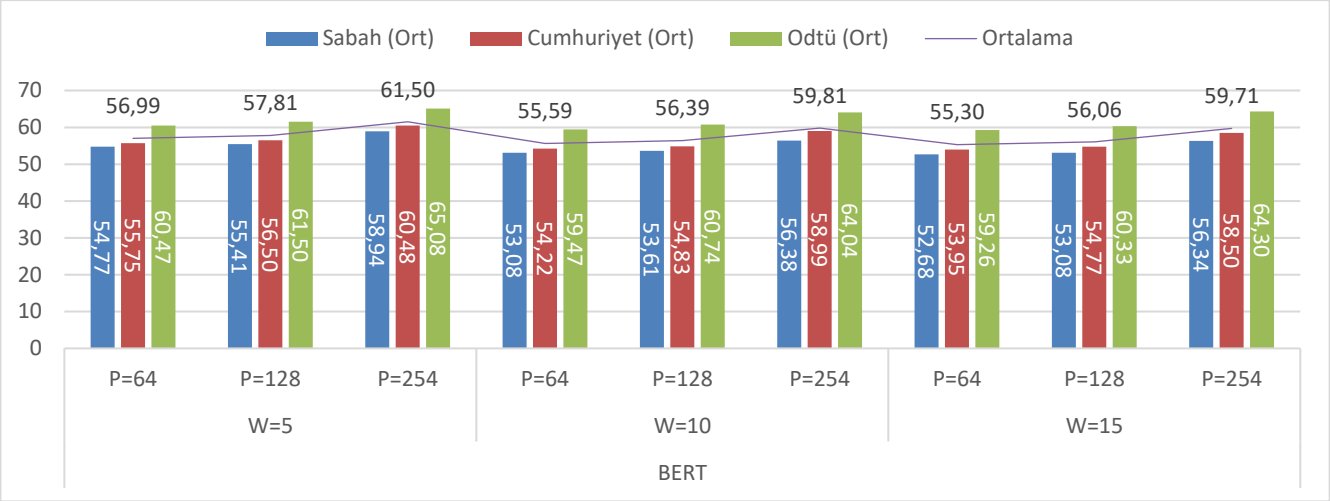
Metin Boyutu	**Gzip**	**Bzip2**	**Zstd**	**Smaz**	**Shoco**	**Shoco TR**	**BERT**								
							W=5			**W=10**			**W=15**		
							P=64	**P=128**	**P=254**	**P=64**	**P=128**	**P=254**	**P=64**	**P=128**	**P=254**
0-99	160,4	217,6	133,8	120,2	123,1	**97,1**	111,4	116,2	118,7	111,1	115,7	117,8	111,0	115,7	117,8
100-199	103,5	114,0	95,1	106,2	111,0	73,8	56,9	62,9	74,8	55,0	59,4	70,7	**54,4**	60,1	72,1
200-299	88,3	98,2	84,6	104,0	109,3	73,4	51,4	52,7	67,1	**50,0**	52,0	65,2	50,1	52,6	63,6
300-399	79,8	86,8	79,9	104,9	110,2	73,7	49,8	49,6	56,5	48,1	47,7	54,2	47,8	**47,6**	54,7
400-499	75,5	80,4	76,5	105,0	111,2	73,5	48,8	48,2	51,9	47,3	46,6	51,7	46,8	**46,0**	49,4
500-599	72,1	75,7	73,1	104,3	110,9	72,7	48,4	47,7	48,9	46,9	46,1	49,5	46,8	**46,0**	48,6
600-699	69,3	73,7	70,6	104,6	111,0	73,1	48,2	47,2	47,2	46,5	45,4	45,7	46,0	45,1	**45,0**
700-799	66,9	70,9	68,6	105,0	111,4	73,3	47,7	47,1	46,8	45,9	45,4	45,3	45,7	45,1	**45,0**
800-899	65,8	68,7	67,7	105,4	112,2	73,6	47,5	47,0	46,8	46,0	45,3	45,5	45,6	45,0	**44,8**
900-	62,4	64,1	64,3	105,2	111,8	73,9	47,4	46,4	46,0	45,6	44,8	44,4	45,3	44,4	**44,0**

ODTÜ veri seti için sıkıştırma oranı sonuçları Tablo 4'te verilmiştir. Bu veri setinde de 0-99 bayt aralığında Shoco TR, diğer boyutlarda önerilen yöntem en iyi sonuçları elde etmiştir. Diğer veri setlerinden farklı olarak bu veri setinde önerilen BERT yöntemi %50'nin, Shoco TR ise %80'nin altına inememiş, diğer yöntemler ise bu veri setinde daha iyi oranlar vermiştir. 900 bayttan büyük metinlerde Gzip ve Zstd %60'ın altına inerek BERT yöntemine daha çok yaklaşabilmiştir.

Tablo 4. ODTÜ veri seti için metin uzunluklarına göre sıkıştırma oranları

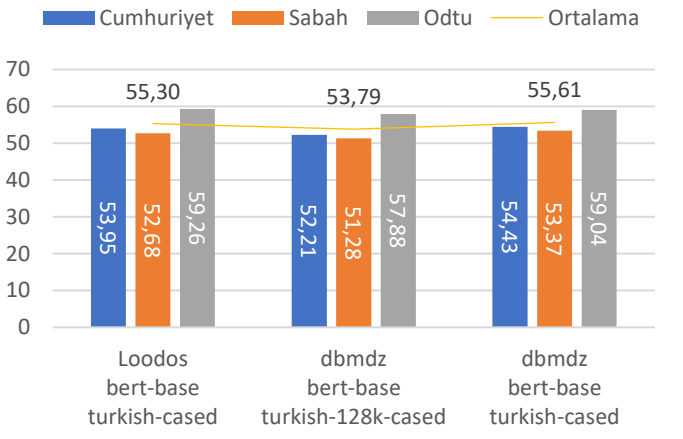
Metin Boyutu	**Gzip**	**Bzip2**	**Zstd**	**Smaz**	**Shoco**	**Shoco TR**	**BERT**								
							W=5			**W=10**			**W=15**		
							P=64	**P=128**	**P=254**	**P=64**	**P=128**	**P=254**	**P=64**	**P=128**	**P=254**
0-99	141,6	186,9	118,5	88,5	90,2	**80,7**	93,9	102,7	108,7	91,5	101,2	105,8	91,4	101,2	105,9
100-199	97,4	108,5	90,3	88,9	92,9	81,6	63,5	66,7	83,0	**63,2**	68,3	81,2	63,7	67,0	84,4
200-299	79,1	88,9	76,1	87,5	91,8	80,5	58,9	59,2	68,3	58,1	58,3	69,9	**57,9**	58,0	69,0
300-399	72,6	80,7	72,6	86,8	91,0	80,4	57,4	57,3	59,3	56,2	56,1	58,7	55,9	**55,7**	60,2
400-499	68,1	74,2	69,2	87,8	92,0	80,7	56,3	56,1	57,4	55,7	55,4	56,5	55,4	**55,0**	56,5
500-599	65,4	69,9	66,9	87,8	92,5	81,0	55,8	55,4	55,8	55,1	54,7	54,8	54,7	**54,2**	54,6
600-699	63,6	67,7	65,0	88,0	92,2	81,0	55,6	55,2	55,6	54,6	54,3	54,5	54,2	**53,9**	54,2
700-799	62,9	67,5	64,1	88,1	91,7	80,9	54,9	54,8	55,0	53,8	53,6	53,6	53,5	**53,3**	**53,3**
800-899	61,6	65,5	62,9	87,3	91,6	80,4	54,9	54,7	54,8	54,0	53,8	53,8	53,8	53,6	**53,5**
900-	58,5	61,1	59,8	87,2	91,1	80,1	53,5	52,9	52,9	52,5	51,8	51,6	52,2	51,4	**51,3**

Tablo 2, 3 ve 4'te her boyut grubu için en iyi değerler kalın yazı ile gösterilmiştir. Tüm tablolarda BERT yönteminin W=15 olduğunda en iyi sonuçları verdiği görülmektedir. 300 bayttan küçük metinlerde P=64 daha başarılı iken, 300-700 bayt aralığında P=128 ve 700 bayttan büyük metinlerde P=254 daha başarılı oranlar vermiştir. Her zaman P=254'ün en başarılı sonuçları verememesinin nedeni, kısa metinlerde kullanılan FSE algoritmasının katkı yapamadığı durumlar olmasıdır. Önerilen yöntemin bütün sonuçları incelendikten sonra farklı konfigürasyonların etkisinin incelenmesi amacı ile tüm sonuçların sıkıştırma oranı değerlerinin ortalaması alınmış ve Şekil 3'te verilmiştir.



Şekil 3. Farklı pencere boyutu ve kelime tahminleri için sıkıştırma oranları

Şekil 3'te görüldüğü üzere P=64 ve W=15 seçilmesi en iyi sonuçları vermiştir. Bu veriler kullanılarak W=15 ve P=64 için *Loodos* dışında iki farklı *dbmdz* BERT modeli ile de aynı veri setleri üzerinde testler yapılmış ve sonuçlar Şekil 7'de verilmiştir. Bu BERT modellerinden biri Loodos gibi 32000 elemanlı sözlük kullanırken, diğeri 128000 elemanlı sözlük kullanmaktadır. Bu karşılaştırma ile model seçiminin sıkıştırma oranına ve aynı zamanda tahmin başarısına etkisinin ölçülmesi amaçlanmıştır.



Şekil 4. Farklı modeller için W=15 ve P=64 parametrelerinde sıkıştırma sonuçları

Şekil 4'te görüldüğü üzere en iyi sonuçlar her veri setinde 128k sözlüğe sahip model ile elde edilmiştir. 32k sözlük kullanan modellerde Loodos'un modeli Sabah ve Cumhuriyet için dbmdz modelinden daha iyi sonuçlar verirken, ODTÜ için çok küçük bir farkla (%0,22) daha kötü sonuç üretmiştir.

5. Sonuç

Kısa metinlerde yeteri kadar tekrar eden veri bulunmadığı için, genel sıkıştırma algoritmaları ile sıkıştırılmaları her zaman mümkün olmamaktadır. Gzip, Bzip2 ve Zstd gibi yöntemler kısa metinlerde sıkıştırma yapmak yerine boyutu büyütebilmektedir. Smaz ve Shoco gibi kısa metinleri sıkıştırmaya yönelik yöntemler ise metin boyutu büyüdükçe sıkıştırma oranlarını arttıramadıkları için, belirli bir boyuttan sonra genel sıkıştırma algoritmalarının gerisinde kalmaktadır.

Metinlerin kısa olmasının yanında Türkçe dilinde olması da sıkıştırma algoritmalarının başarılı sonuçlar verememesine neden olmaktadır. Türkçe gibi ekli dillerde, kökü aynı olan kelimeler farklı ekler ile birlikte değişmekte ve sıkıştırmada aranan benzerlik yakalanamaktadır.

BERT transformer mimarisinin kullanılması ile oldukça başarılı bir şekilde kelime tahmini yapılabilmektedir. Ayrıca BERT birden fazla tahminde bulunarak istenilen bir tahminin seçilebilmesine de olanak sunabilmektedir. Bu avantajlar sayesinde BERT'in tahmin ettiği kelimelerin listesi kullanılarak sıkıştırma işlemi gerçekleştirilebilmektedir. Özellikle tahmin edilen kelimeler ekler ile birlikte içeriğe göre tahmin edildiği için tahmin başarısı ve buna bağlı olarak sıkıştırma başarısı artmaktadır.

BERT'in tahmin aşamasında kullanılan pencerenin büyüklüğü de sonuçları etkilemektedir. Beklendiği gibi pencere boyutunun büyümesi ile BERT'e tahmin için verilen bilgi artmakta, bu sayede tahmin başarısı daha yüksek olabilmektedir.

Pencere boyutunun yanında BERT'in tahmin sonucu olarak vereceği kelime sayısının arttırılması da sıkıştırma oranına olumlu etki yapmaktadır. BERT'in daha fazla kelime üretmesi, metindeki kodlanacak olan kelimenin üretilen tahmin listesinde yer alması olasılığını arttırdığı için bu doğal bir sonuçtur.

İlerleyen çalışmalarda tahmin başarısını arttırmak amacı ile önişlemler gerçekleştirilmesi ve aynı zamanda model başarısının iyileştirilmesi amaçlanmaktadır.

Kaynakça

Aslanyürek, M., & Mesut, A. (2021). A Static Dictionary-Based Approach To Compressing Short Texts. *2021 6th International Conference on Computer Science and Engineering (UBMK)*, 342–347.

Collet, Y., & Kucherawy, M. (2018). Zstandard Compression and the application/zstd Media Type. *RFC 8478*.

Deutsch, P. (1996). DEFLATE compressed data format specification version 1.3. https://www.rfc-editor.org/info/rfc1951.

Deutsch, P. (1996). GZIP file format specication version 4.3. https://www.rfc-editor.org/info/rfc1952.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). {BERT:} Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR, abs/1810.0*. http://arxiv.org/abs/1810.04805

Duda, J., Tahboub, K., Gadgil, N. J., & Delp, E. J. (2015). The use of asymmetric numeral systems as an accurate replacement for Huffman coding. *2015 Picture Coding Symposium, PCS 2015 - with 2015 Packet Video Workshop, PV 2015 - Proceedings*. https://doi.org/10.1109/PCS.2015.7170048

Gardner-Stephen, P., Bettison, A., Challans, R., Hampton, J., Lakeman, J., & Wallis, C. (2013). Improving Compression of Short Messages. *International Journal of Communications, Network and System Sciences*, *06*(12), 497–504. https://doi.org/10.4236/ijcns.2013.612053

Huffman, D. A. (1952). A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, *40*(9), 1098–1101.

Manzini, G. (2001). An analysis of the Burrows—Wheeler transform. *Journal of the ACM (JACM)*, *48*(3), 407–430.

Mathews, G. J. (1995). Selecting a general-purpose data compression algorithm. *Proceedings of the Science Information Management and Data Compression Workshop*, 55–64.

Nguyen, V. H., Nguyen, H. T., Duong, H. N., & Snasel, V. (2016). *n -Gram-Based Text Compression*. *2016*.

Öztürk, E., Mesut, A., & Diri, B. (2017). Multi-stream word-based compression algorithm. *2nd International Conference on Computer Science and Engineering, UBMK 2017*. https://doi.org/10.1109/UBMK.2017.8093552

Öztürk, E., Mesut, A., & Diri, B. (2018). Multi-Stream Word-Based Compression Algorithm for Compressed Text Search. *Arabian Journal for Science and Engineering*, *43*(12), 8209–8221. https://doi.org/10.1007/s13369-018-3378-9

Platoš, J., Snášel, V., & El-Qawasmeh, E. (2008). Compression of small text files. *Advanced Engineering Informatics*, *22*(3), 410–417. https://doi.org/10.1016/j.aei.2008.05.001

Sabancı Üniversitesi Veri Analitiği Araştırma ve Uygulama Merkezi. (2018). *SuDer Corpus - Turkish News Collections for Text Categorization*. https://github.com/suverim/suder

Sanfilippo, S. (2009). SMAZ—Compression for Very Small Strings. https://github.com/antirez/smaz

Say, B., Zeyrek, D., Oflazer, K., & Özge, U. (2002). Development of a corpus and a treebank for present-day written Turkish. *In Proceedings of the eleventh international conference of Turkish linguistics* (pp. 183-192).

Schramm, C. (2013). Shoco: a fast compressor for short strings. https://ed-von-schleck.github.io/shoco/.

Seward, J. (1996). bzip2 and libbzip2, version 1.0.8. https://www.sourceware.org/bzip2/manual/manual.pdf.

Storer, J. A., & Szymanski, T. G. (1982). Data Compression via Textual Substitution. *J. ACM*, *29*(4), 928–951. https://doi.org/10.1145/322344.322346

Ziviani, N., De Moura, E. S., Navarro, G., & Baeza-Yates, R. (2000). Compression: A key for next-generation text retrieval systems. *Computer*, *33*(11), 37–44.