Türk Dilinin Biçimbilim Yapısından Yararlanarak Türkçe Metinlerin Farklı İmgelere Ayrılarak Kodlanması ve Sıkıştırılması

Banu DİRİ, M.Yahya KARSLIGİL
Yıldız Teknik Üniversitesi
Elektrik Elektronik Fakültesi - Bilgisayar Mühendisliği
Yıldız-İstanbul 80750
banu@ce.yildiz.edu.tr,yahya@ce.yildiz.edu.tr

Özetçe

Türkçe'nin biçimbilim yapısından yararlanarak Türkçe bir metin içerisindeki her kelime kök-ek, hece ve karakter gibi farklı imgelere ayrılarak, geliştirilen üç ayrı program yardımıyla kodlanmıştır. Böylece Türkçe metin dosyaları hem Huffman algoritması ile kodlanmış hem de sıkıştırılmıştır. Ayrıca Türk alfabesini oluşturan karakterlerin kullanım sıklık eğrisinin üstel bir fonksiyon olduğu gösterilmiştir.

1. Giriş

Biçimbilim (morphology), dilleri kelime yapıları bakımından incelemektedir. Bitişken diller (agglutinative) sınıfından bir dil olan Türkçe'de, kelimenin köklerine çeşitli ekler takılarak kelimeye yeni anlamlar yüklenebildiği gibi, yeni kelimeler de elde edilebilir. Türkçe sondan eklemeli bir dil olması sebebiyle biçimbilim yapısı dikkate alınarak bir kelime kök, ek veya gövde gibi farklı imgelere ayrılarak kodlanabilir. Yapılan bu çalışmada bir Türkçe metindeki kelimelerin kök-gövde, ek ve hecelerine ayrılarak daha çok bilginin bir seferde kodlanması amaçlanmıştır. İmgelerin kodlanmasında statik Huffman kodlama yöntemi temel alınarak [3] farklı algoritmalar geliştirilmiş ve bunların verimlerine göre karşılaştırılması yapılmıştır [2].

2. Türkçe'de Kullanılan Karakterlerin Kullanım Sıklıkları

Çalışmanın ilk bölümünde öncelikli olarak Türk alfabesindeki karakterlerin ve özel işaretlerin kullanım sıklıkları çıkarılmıştır. Kullanım sıklıklarını elde etmek için önceden oluşturulmuş farklı uzunluk ve içeriklere sahip ondört Türkçe metin dosyasından oluşan bir *test-kümesi* kullanılmıştır. Bu test kümesi içerisinde geçen her bir karakterin (sadece Türk alfabesinde yer alan 29 karakter) kullanım olasılığı Çizelge 1'de verilmiştir.

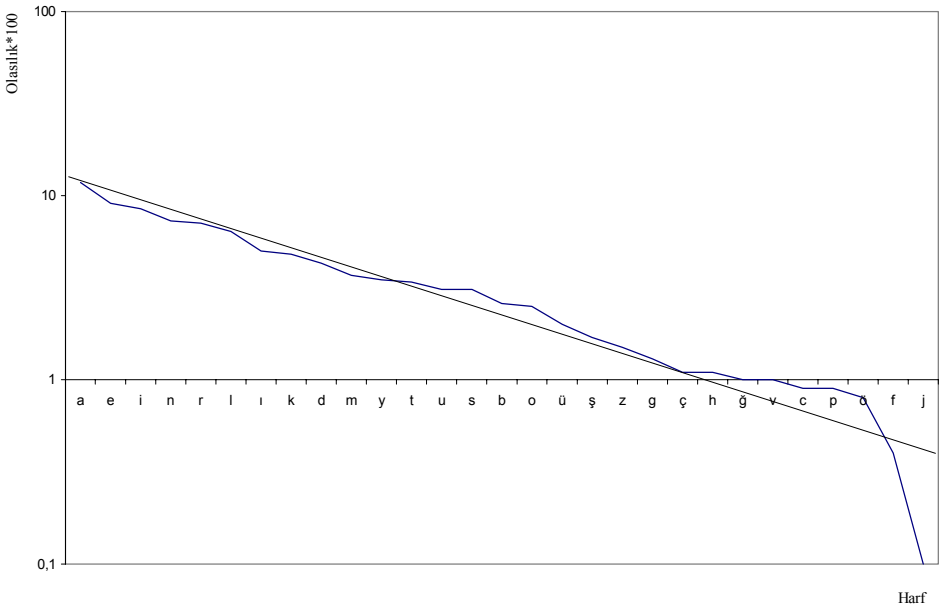
Çizelge 1 Türk alfabesindeki karakterlerin kullanım sıklık olasılıkları

Harf	Olasılık	Harf	Olasılık	Harf	Olasılık	Harf	Olasılık	Harf	Olasılık
a	0,118	ı	0,050	u	0,031	z	0,015	c	0,009
e	0,091	k	0,048	s	0,031	g	0,013	p	0,009
i	0,085	d	0,043	b	0,026	ç	0,011	ö	0,008
n	0,073	m	0,037	o	0,025	h	0,011	f	0,004
r	0,071	y	0,035	ü	0,020	ğ	0,010	j	0,001
l	0,064	t	0,034	ş	0,017	v	0,010		

Türkçe'deki harflerin kodlanması için gerekli ortalama bit uzunluğu, entropy hesabı (Denklem 1) yapılarak 4,37 bpc (bit per char) bulunmuştur.

$$H = -\sum_{i=1}^{n} P_i \log_2 P_i \tag{1}$$

Türk alfabesini oluşturan karakterlerin kullanım sıklığı eğrisi (Şekil 1) lineer doğru denklemine yakın değer göstermekte olup, $y=a*e^{-bx}$ fonksiyonu ile gösterilir.



Şekil 1 Türk alfabesindeki karakterlerin kullanım sıklık eğrisi (yarı logaritmik eksen)

Bu üstel fonksiyonun her iki tarafının da logaritması alınır ve gerekli dönüşümler yapılırsa Denklem 2'deki doğru denklemi elde edilir [2].

$$\ln(y) = \ln(a) + (-bx) \tag{2}$$
$$Y = A + B$$

Doğru denkleminin A ve B katsayılarını bulmak için En Küçük Kareler yöntemi kullanılarak Denklem 3'deki matris formu elde edilir.

$$\begin{bmatrix} n & \sum x \\ \sum x & \sum x^2 \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix} = \begin{bmatrix} \sum \ln y \\ \sum x \ln y \end{bmatrix} \tag{3}$$

Bu form lineer denklem sistemlerinin [M][X]=[N] şeklindeki genel ifadesine benzetilirse n=29 için A=13,35 ve B=-0,12 olup $a=e^{13,35}=627,814$ ve b≅B=0,12 olarak bulunup doğru denklemi $Y=627,814e^{-0,12x}$ şeklinde elde edilir.

3. Karakter-İmge Kodlama

Çalışmanın bu bölümünde, tek bir karakterden oluşan imgelere ait kullanım sıklıkları statik Huffman kodlama ağacına yerleştirilerek, herbir imge için yeni bir kod elde edilmiştir. Karaktere dayalı olarak gerçekleştirilmiş bu kodlama yönteminde (krk_kod) metin içerisindeki her karakter kendisi için oluşturulmuş Huffman kodlarıyla kodlandığında ortalama %45'lik bir sıkıştırma verimi elde edilmiştir (Çizelge 2). Ortalama sıkıştırma verimi (μ) de Denklem 4'deki ağırlıklı ortalama formülünden yararlanılarak hesaplanmıştır [2]. (l:sıkıştırılmış dosya boyu, k:dosya adedi, η:sıkıştırma verimi=100*(1- çıkış bilgisi uzunluğu/giriş bilgisi uzunluğu))

$$\mu = \frac{\sum_{i=1}^{k} l_i \eta_i}{\sum_{i=1}^{k} l_i} \qquad (4)$$

4. Hece-İmge Kodlama

Çalışmanın hece kodlama bölümünde, Türkçe'de kullanılan hecelerin kullanım sıklıkları çıkarılmış ve kullanım sıklığı en çok olan ilk 578 adet hece seçilmiştir. Bu bölümde her hece imgesi en az iki, en çok üç karakterden oluşmaktadır. Türkçe'deki hecelerin kodlanması için gerekli ortalama bit uzunluğu entropy hesabı yapılarak 7,64bpc olarak bulunmuştur. Hecelere ait olan kullanım sıklıkları statik Huffman kodlama ağacına yerleştirilerek her hece için yeni bir kod elde edilmiştir. Heceye dayalı olarak gerçekleştirilmiş bu kodlama yönteminde (hec_kod) metin içerisindeki her kelime hecelerine bölünmüş daha sonra her hece kendi için oluşturulmuş Huffman kodlarıyla kodlandığında ortalama %46' lık bir sıkıştırma verimi elde edilmiştir (Çizelge 2).

5. Kök ve Ek-İmge Kodlama

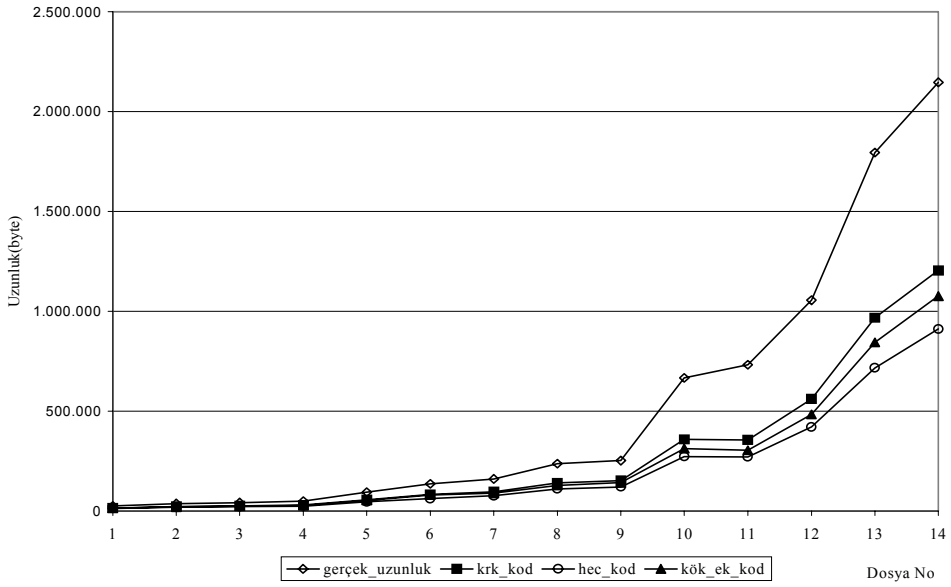
Çalışmanın Kök ve Ek-İmge kodlama bölümünde, Türkçe'de kullanılan kök ve eklerin kullanım sıklıkları çıkarılmış ve en çok sıklıkta kullanılan 478 adet kök ve ek seçilmiştir [1]. Bu bölümde her imge en az iki, en çok dört karakterden oluşmaktadır. Türkçe'deki kök ve eklerin kodlanması için gerekli ortalama bit uzunluğu entropy hesabı yapılarak 7,40bpc olarak bulunmuştur. Kelimenin kök ve eklerine ait olan kullanım sıklıkları statik Huffman kodlama ağacına yerleştirilerek her kök ve ek için yeni bir kod elde edilmiştir. Kök ve eklere dayalı olarak gerçekleştirilmiş bu kodlama yönteminde (kök_ek_kod) metin içerisindeki her kelimenin önce kökü bulunup, kelimeden ayrıldıktan sonra eklerine ayrılmış ve herbiri kendisi için oluşturulmuş Huffman kodlarıyla kodlandığında ortalama %51' lik bir sıkıştırma verimi elde edilmiştir (Çizelge 2).

Krk_kod, hec_kod ve kök_ek_kod adını verdiğimiz bu üç yöntemin sıkıştırma verimlerinin karşılaştırılması Çizelge 2'de, grafik olarak gösterilimi de Şekil 2'de verilmiştir.

Kelimeleri hece, kök ve eklerine ayırabilmek için sonlu durum makineleri (SDM) kullanılmıştır. Analiz yapılacak olan metin dosyası bloklar halinde hafızaya çekilmektedir. SDM, blok içerisinde bir kelimeyi yakaladığında gerekli algoritmaları çağırarak, bu kelimenin en uygun şekilde kök ve eklerine ayrılmasını sağlamaktadır.

Çizelge 2 Gerçek metin dosyası ile krk_kod, hec_kod ve kök_ek_kod karşılaştırılması

Sıra No	Dosya Uzunluğu(byte)	Krk_kod η %	Hec_kod η %	Kök_ek_kod η %
1	24.624	40,0	40,7	46,3
2	36.596	40,4	41,0	46,5
3	41.377	38,1	35,5	42,4
4	48.881	40,1	39,1	45,8
5	93.884	39,9	35,7	44,5
6	135.361	39,3	30,5	40,9
7	160.240	39,9	40,4	44,6
8	236.705	40,6	38,7	45,9
9	251.606	39,6	37,0	43,7
10	666.120	46,2	49,8	53,1
11	731.929	51,4	55,5	58,4
12	1.055.244	46,9	50,8	54,2
13	1.795.226	46,1	49,1	52,9
14	2.147.095	43,9	43,8	49,9



Şekil 2 Gerçek metin dosyası ile diğer yöntemlerin karşılaştırılması

6. Sonuç

Türkçe metindeki herbir kelime önce harf harf sonra da Türkçe'nin biçimbilimine dayalı olarak hecelerine, kök ve eklerine ayrılarak herbir imge için önceden oluşturulmuş olan statik kodlama şablonlarıyla kodlandığında değişik sıkıştırma verimleri alınmıştır. Ortalama sıkıştırma verimi %45 ile %51 arasında kullanılan algoritmaya bağlı olarak değişmektedir. Metin dosyalarının uzunluğu arttıkça daha iyi sıkıştırma verimi alındığı gözlenmiştir. İleri ki çalışmalarda, imge olarak kelimeler seçilirse daha iyi sıkıştırma verimi elde edilir.

Kaynakça

[1]. Çotuksöken, Y., (1991), Türkçede Ekler-Kökler-Gövdeler, Cem Yayınevi, İstanbul

[2]. Diri, B., (1999), Türkçe'nin Biçimbilim Yapısına Dayalı Bir Metin Sıkıştırma Sistemi, Doktora Tezi, Yıldız Teknik Üniversitesi, İstanbul

[3]. Salomon, D., (1998), Data Compression, Springer-Verlag Inc., NewYork

[4]. Tanaka, H., (1987), "Data Structure of Huffman Codes and its Application to Efficient Encoding and Decoding", IEEE Transactions on Information Theory, 33:154-156