# Türkçe Cümlelerin Kural Tabanlı Bağlılık Analizi

Gülşen Eryiğit[1], Eşref Adalı[1], Kemal Oflazer[2]

[1] İstanbul Teknik Üniversitesi, Elektrik Elektronik Fakültesi
Bilgisayar Mühendisliği Bölümü, Maslak 34469 İstanbul
{gulsen, adali}@cs.itu.edu.tr
[2] Sabancı Üniversitesi, Mühendislik ve Doğa Bilimleri Fakültesi
Bilgisayar Mühendisliği Bölümü, Tuzla 34956, İstanbul
oflazer@sabanciuniv.edu

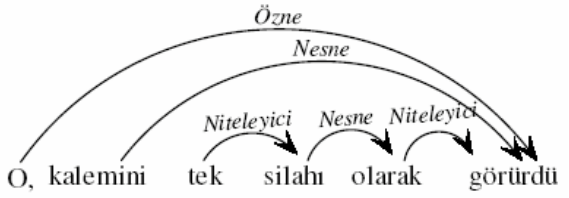
**Özet.** Bu makalede, Türkçe cümlelerin kural tabanlı bağlılık analizi yöntemi ile ayrıştırılmaları sonucunda elde edilen başarım sunulmaktadır. Çalışma, test verisi olarak kullanılan ODTÜ-Sabancı Ağaç Yapılı Derlemi'nin bütünü üzerindeki ilk kural tabanlı sonuçları içermektedir. Uygulanan ayrıştırma algoritması ve kural yapıları detaylı olarak verilmiştir. Sonuçlar Türkçe'nin Bağlılık Analizi konusunda yapılacak çalışmalara temel olma niteliğindedir.

## 1 Giriş

Cümle Analizi yöntemlerinden biri olan Bağlılık Analizi, kökleri çok eskiye dayanmasına rağmen "Doğal Dil Ayrıştırması (DDA)" alanında ancak son yıllarda popüler hale gelmiş bir yöntemdir. Özellikle farklı diller için bağlılık gramerleri kullanılarak oluşturulan derlemlerin (Arapça[1], Çekçe[2], Danimarkaca[3], İtalyanca[4], Japonca[5], Slovakça[6], Türkçe[7]) sayısının artması ile birlikte, bu konuda yapılan araştırmalar da hız kazanmıştır. Bunlara ek olarak birçok çalışmada da, diğer gramer yöntemleri kullanılarak işaretlenmiş derlemler bağlılık gramerine uygun bir yapıya dönüştürülmüş ve bağlılık analizleri yapılmıştır. Nivre ve Nilsson[8]'un Çekçe, Kudo ve Matsumoto[9][10]'nun ve Sekine ve ark.[11]'nın Japonca, Yamada ve Matsumoto[12]'nun İngilizce, Nivre[13]'nin İsveççe, Oflazer[14]'in ve Eryiğit ve Oflazer[15]'in Türkçe için yaptığı çalışmalar bu konuda yapılan araştırmalara örnek olarak gösterilebilir.

Modern Bağlılık Grameri Teorisinin Tesnière'in 1959'daki çalışmasına dayandığı söylenebilir. Tesnière'e göre "Cümle, kendisini oluşturan öğeleri sözcükler olan düzenli bir topluluktur"[16]. "Zihin, cümleyi oluşturan sözcükler ve komşuları arasında ilişkiler bulur ve bu ilişkilerin bütünü cümlenin iskeletini oluşturur. Her bir ilişki bir alt terimi bir üst terime bağlamaktadır." Günümüzde DDA alanında kullanılan Bağlılık Gramerlerinde bu ilişki bağımlı(alt terim)-sahip(üst terim)[1] ilişkisi olarak tanımlanmaktadır. Şekil 1'de Türkçe bir cümlenin bağlılık ilişkileri gösterilmektedir. Sözcükler arasında çizilen oklar bağımlı sözcükten sahip sözcüğe doğru olan bağlılığı ve bu bağlılığın tipini göstermektedirler.

[1] Literatürde, *Dependent-Head* veya *Subordinate-Governor* ilişkisi olarak da geçmektedir.




**Şekil. 1.** Bağlılık İlişkileri

Bu çalışmada, önceden oluşturulmuş kurallar ve gerekirci bir ayrıştırma algoritması kullanılarak gerçekleştirilen Türkçe'nin Bağlılık Analizinin sonuçları sunulmaktadır. İleriki bölümlerde sırasıyla Türkçe'nin özellikleri, Bağlılık Analizi yöntemleri, üzerinde çalışan derlemin özellikleri, kullanılan kural tabanlı ayrıştırıcının yapısı ve elde edilen başarım verilecektir.

## 2 Türkçe

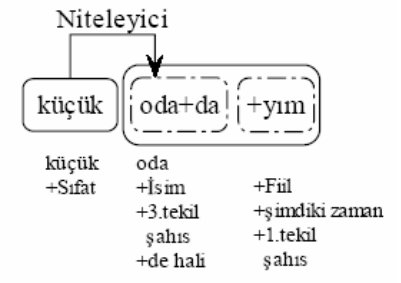
Bitişken bir dil olan Türkçe'de, sözcüklerin sonlarına ardarda ekler konularak yüzlerce farklı sözcük oluşturulabilir. Cümlelerin, sözcük dizilişleri itibari ile büyük çoğunlukla ÖNY(Özne-Nesne-Yüklem) veya NÖY kalıplarına uymasına rağmen, öğeler anlatılmak istenen içeriğe ve vurguya bağlı olarak cümle içerisinde serbestçe yer değiştirebilirler. Bozşahin[17] çalışmasında Türkçe'nin bu yapısını detaylı olarak incelemiştir.

Türkçe'de sözcükler sonlarına eklenen eklerle farklı tipte sözcüklere dönüşebilirler; fiiller isimlere, isimler fiillere vb... Bağlılıkları sadece sözcükler arasında göstermek ayrıştırma işlemi için yeterince anlamlı bilgi taşımamaktadır. Şekil 2'de *küçük odadayım* cümlesi içerisindeki bağlılık gösterilmektedir. Sözcüklerin kökleri ve biçimbirimsel çözümlemeleri sözcüklerin aşağısında verilmektedir. Verilen örnekte *küçük* olan, *odadayım* sözcüğü değil *oda*'dır. *odadayım* isimden fiile dönüşmüş bir sözcüktür. İki sözcük arasında kurulan bağlantı *odadayım* sözcüğünün fiile dönüşmeden önceki isim halinden kaynaklanmaktadır.Bu durum sıfatların genel olarak isimlere bağlanması kuralından ötürüdür. Bağımlı-Sahip ilişkisini sadece sözcükler arasında göstermek, sıfatların fiillere bağlandığı gibi yanlış bir kanı ortaya çıkarabilir. Özellikle kuralların veya istatistiklerin bir derlemden otomatik olarak çekildiği yöntemlerde bu tip bir gösterim başarımı önemli ölçüde düşürmektedir[15].

Türkçe'nin bu yapısı literatürde [7][14][15][18] sözcüklerin çekim gruplarına(*ÇG*) ayrılması ile gösterilmektedir. Bu gösterimde, tip değiştiren sözcükler türetim sınırlarından(*TS*) bölünerek *ÇG*'lere ayrılırlar. Her çekim grubu ilgili sözcük bölümüne ait biçimbirimsel bilgiyi barındırır. Şekil 2'deki *odadayım* sözcüğü iki *ÇG*'den oluşmaktadır:

$$\underbrace{oda{+}\text{İsim}{+}3.\text{t.k.}{+}\text{de\_hali}}_{ÇG_1} \,\hat{}\, TS \,\hat{}\, \underbrace{\text{Fiil}{+}\text{ş.z.}{+}1.\text{t.k.}}_{ÇG_2}{}^{2}$$

[2] 1.t.k - 1. tekil kişi, 3.t.k. - 3. tekil kişi, ş.z. - şimdiki zaman




**Şekil. 2.** Çekim Grubu Yapıları

## 3 Derlem

Bu çalışmada test verisi olarak bağlılık grameri yapısına uygun hazırlanmış ODTÜ-Sabancı Ağaç Yapılı Derlemi[7] kullanılmıştır. Derlem, biçimbirimsel çözümleyiciden geçirilmiş ve farklı biçimbirimsel çözümler arasında belirsizlik giderme işlemi yapılmış sözcüklerden oluşan 5635 adet cümle içermektedir. Derlemde bağlılıklar *ÇG*'ler arasında kurulmuştur. Bağlılıklar, bağımlı sözcüğün son *ÇG*'sinden başlayarak sahip sözcüğün herhangi bir *ÇG*'sinde sonlanabilirler.

Türkçe genelde ve özellikle yazım dilinde "sahip-sonda"[3] bir dil olarak nitelendirilebilir. Nitekim derlemdeki bağlılıkların %95'i bu tip bağlılıklardan oluşmaktadır. Türkçe'de bir sözcüğe ait biçimbirimsel özellikler büyük çoğunlukla o sözcüğün içerisinde bir çekim eki olarak yer almaktadırlar. Ancak bazı ekler ("de, mi, ki"[4]) sözcüğe ait biçimbirimsel özellik taşımalarına rağmen sözcükten sonra ve sözcükten ayrı olarak yazılırlar ve kendilerinden önce gelen sözcüğe bağlanırlar. Bu bağlılıklar bir önişlemci yazılarak rahatlıkla bulunabilir. Bu işlemin sonucunda "sahip-sonda" kuralına uymayan bağlılıkların oranı %5'den %3'e inmektedir.

Son birkaç yıla kadar, bağlılık analizi yaklaşımlarında yapılan genel varsayımlardan biri cümlelerin izdüşel[5] olmayan bağlılıklar içermeyeceği yönündedir. Buna karşın, farklı dillere ait birçok derlemde izdüşel olmayan cümlelere rastlanmaktadır. Bu nedenle son yıllarda bu tip cümleleri de içerecek yöntemler üzerinde çalışılmaktadır[8][20]. Bir sahip sözcüğe bağlanmamış noktalama işaretleri gözardı edildiğinde, Türkçe derlemdeki cümlelerin %7.2'sinin izdüşel olmayan bağlılıklardan oluştuğu saptanmaktadır.

[3] Sahip sözcüğün, Bağımlı sözcükten sonra gelmesi (birbaşka deyişle Bağımlı sözcüğün sağ tarafında yer alması) durumu

[4] de/da, mi soru ekinin kişi ve zaman ekleri almış tüm varyasyonları, ki

[5] izdüşel(*projective*): kesişen veya Bağımlı → Sahip bağlılık oku altında Sahip sözcükten bağımsız herhangi bir sözcük barındırmayan[19]

# 4 Kural Tabanlı Ayrıştırıcı

Ayrıştırcı, bir cümlenin bağlılık ağacını oluşturmak amacıyla olası bütün bağlılık kombinasyonları içerisinden optimum olanı bulmaya çalışır. Ayrıştırcıların genel olarak iki bölümden oluştukları söylenebilir: 1-Ayrıştırma Algoritması 2-Ayrıştırma Modeli. Ayrıştırma algoritmaları, en iyi ağacı oluşturmak üzere ayrıştırma modelinden faydalanırlar. Son yıllara kadar, özellikle istatistiksel ayrıştırma alanında birinci bölümde çeşitli dinamik programlama algoritmaları kullanılmıştır. Bu algoritmalar ayrıştırma modelinin kendilerine verdikleri ikili bağlılık olasılıklarını kullanarak, arama uzayında yer alan en yüksek olasılıklı ayrıştırma ağacını bulmaya çalışırlar. Bu yöntemden tamamen farklı olan bir başka yaklaşım da gerekirci ayrıştırma algoritmaları kullanmaktır. Birçok çalışmada[9][12][14][19] bu algoritmaların yüksek başarımı raporlanmıştır. Gerekirci ayrıştırma algoritmaları, ayrıştırıcının her adımında bir sonraki hareketin ne olacağına ayrıştırma modeli yardımıyla karar verirler. Bu durumda ayrıştırma modeli herhangi bir makine öğrenimi sınıflandırıcısı olabileceği gibi kural tabanlı bir sınıflandırıcı da olabilir. Aşağıdaki bölümlerde kullanılan gerekirci ayrıştırma algoritması ve modeli verilmektedir.

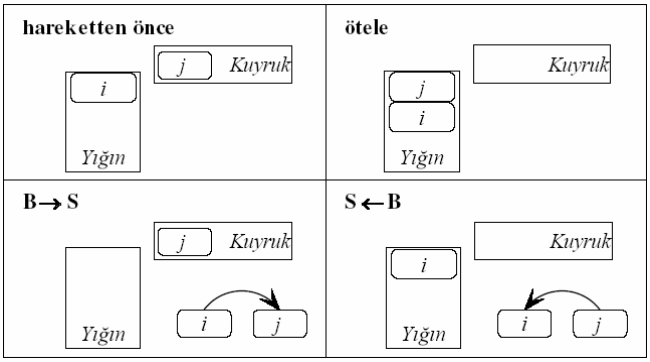
## 4.1 Ayrıştırma Algoritması

En sık kullanılan gerekirci ayrıştırma algoritmaları basit ötele-indirge algoritmasının çeşitli varyasyonlarıdır. Bu algoritmalar genelde cümleyi soldan sağa doğru, iki farklı veri yapısından faydalanarak ayrıştırırlar: 1- İşlenmekte olan sözcüklerin tutulduğu yığın yapısı 2-İşlenmek üzere bekleyen sözcüklerin tutulduğu kuyruk yapısı. Bu çalışmada Nivre[13]'nin *arc-standard* gerekirci algoritmasına benzer bir algoritma Türkçe'nin bağlılık yapısına uygun olacak şekilde oluşturulmuştur.

Ayrıştırma sırasında ayrıştırıcı *ÇG*'ler arasında ilişkiler oluşturmaya çalışır. Bu nedenle yığında ve kuyrukta tutulan elemanlar sözcükler değil *ÇG*'lerdir. Algoritmanın işleyişi şöyledir (i=yığının en üstünde duran *ÇG*'nin indisi, j=kuyrukta bekleyen sıradaki *ÇG*'nin indisi):

```
Kuyrukta bekleyen ÇG olduğu sürece tekrarla{
        eğer Yığın boş ise
            Ötele(Yığın)
        değil ise
            hareket=Ayrıştırma_Modeli(i,j)
            eğer hareket==Ötele ise
                Ötele(Yığın)
            eğer hareket==B → S ise
                Bağlılık_Kur(i → j)
                Çek(Yığın)
            eğer hareket==S ← B ise
                Bağlılık_Kur(j → i)
                Kuyrukta bekleyen sıradaki ÇG'ye ilerle
}
```

Ayrıştırıcı[6] her adımında 3 farklı hareketten (Ötele, B → S, S ← B) birini gerçekleştirir (Hareketler sonrasında yığının ve kuyruğun durumu Şekil 3'de gösterilmiştir). Ayrıştırıcının bir sonraki hareketinin ne olacağına, ayrıştırma modeli "i" ve "j" indisli elemanların özelliklerine bakarak karar verir. Bu işlem sırasında kullanılan özelliklerin neler olduğu Bölüm 4.2'de anlatılmaktadır. "Ötele"me işleminde kuyrukta bekleyen eleman yığına itilir. Bu işlem yığının boş olduğu durumlarda veya "i" ve "j" indisli elemanlar arasında herhangi bir bağlılık kurulamadığı durumlarda gerçekleşir. "B → S" işlemi "i" indisli eleman ile cümle içerisinde sağ tarafında yer alan "j" indisli eleman arasında bağımlı-sahip ilişkisi olduğu durumlarda gerçekleşir. "S ← B" işlemi ise sahip sözcüğün bağımlının sol tarafında yer alması durumlarında kurulan bağlılıklar için geçerlidir. Bu işlemlerin her birinde ilişkiler yığının en üstünde yer alan sözcük ile kuyruğun en başında yer alan sözcük arasında kurulmaktadır. Bunlara ek olarak ayrıştırma modeli, "j" yi yığının en üstünde olmayan bir "h" elemanına bağlama kararı da alabilir. Bu durumda ayrıştırıcıya basit "S ← B" hareketini döndürmeden önce kendi içerisinde "S ←* B" işlemi yürütür. Bu işlem yığında "h" nin üzerinde yer alan tüm elemanları "h" nin bağımlısı haline getirir ve bu elemanları yığından çeker. Böylece işlem sonunda "h" yığındaki en üst eleman haline gelir. Bundan sonra basit "S ← B" hareketi döndürülür. Bir başka deyişle bu hareket $h \xleftarrow{S \leftarrow B} j$ bağlılığına karar verildiğinde (h < x < j) "h" ve "j" arasında yer alan tüm "x" leri "h" ye bağlamak anlamındadır.




**Şekil. 3.** Ayrıştırma Algoritması

[6] - Bu algoritma sadece izdüşel bağlılıkları bulmaya yöneliktir.

- $a \xleftarrow{S \leftarrow B} b \xleftarrow{S \leftarrow B} c$ tipinde bağlılıkların derlemde görülme sıklığı %0.1'den daha azdır. Bu nedenle bu algoritma bu tip bağlılıkları oluşturmayacak şekilde tasarlanmıştır.

### 4.2 Ayrıştırma Modeli

Ayrıştırma modeli, yığının en üstünde(i) ve kuyruğun en başında(j) bulunan elemanların özelliklerine bakarak bir sonraki hareketin ne olacağına karar verir. Bu karar sırasında kullanılan özellikler ilgili elemanın sözcük bilgisi, tip bilgisi, o ana kadar yapılan işlemler sonucunda kendisine bağlanmış olan bağımlı elemanların özellikleri veya komşularının özellikleri olabilir.

Model ilk olarak, sözcükten ayrı yazılan ve çekim eklerinden dolayı oluşan sahip başta türünde bağlılıkları bulmaya çalışır. Bu durumda eğer "j" elemanı bir vurgulayıcı(de, da), soru eki (mi, mu, mısın vb...), ilişkilendirici (ki) veya olumsuzluk (değil) belirten bir ek [7] ise "S ← B" hareketi döndürülür.

Ayrı yazılan çekim ekleri kontrolü yapıldıktan sonra, ayrıştırıcı elle yazılmış 26 adet kuralı kullanarak "i" ve "j" sözcükleri arasında "B → S" ilişkisi kurmaya çalışır. Bu kurallar, "sıfat yanındaki isime bağlanır", "zarf yanındaki fiile bağlanır" tipinde kurallardır. Birden fazla *ÇG*'ye sahip olan sözcükler içerisinde yer alan *ÇG*'lerin, aynı sözcük içerisinde sağ taraflarında yer alan ilk *ÇG*'ye bağlandıkları varsayılır. Bu durumda ayrıştırma modeli "B → S" hareketi döndürür. Kurulan bu tip bağlılıklar başarım ölçümünde puanlanmazlar.

"i" ve "j" sözcüklerinin "B → S" kurallarından hiçbirine uymaması durumunda eğer "j" elemanı isim soylu ise ve "j"den sonra bir noktalama işareti geliyorsa, ayrıştırma modeli sahip başta tipinde bir bağlılık bulmaya çalışır. Derlemde yer alan sahip başta türündeki bağlılıkların %83'ü bu tipte bağlılıklardır. Bu durumda, yığının en üstündeki elemandan başlanarak eylem soylu bir eleman bulunmaya çalışılır. j'nin yığın içerisinde yer alan ilk eylem soylu "h" elemanına bağlanmasına karar verilir. Eğer "h" yığının en üstteki elemanı değilse bu durumda "h"nin üzerindeki birimlerin yığından çekilerek bir yere bağlanmaları gerekir. Bu durum izdüşellik prensibinin gereğidir. Ayrıştırma algoritması, "h" yığının en üstüne gelene dek, bu elemanları "h" sahip elemanına bağlar. Ve daha sonra "S ← B" hareketi döndürülür.

Yukarıda belirtilen hareketlerden hiçbirinin bulunamaması durumunda model "ötele" hareketi döndürür. Ayrıştırmanın tamamlanabilmesi için cümle içerisindeki kök sözcük (derlemde genelde en sonda yer alan noktalama işareti) hariç tüm sözcüklerin bir sahip sözcüğe bağlanmaları gerekmektedir. Ayrıştırma sonucunda kuyrukta bekleyen hiçbir eleman kalmamasına rağmen yığında birden fazla eleman bulunuyorsa, bu elemanlar yığının en üstündeki elemana bağlanırlar.

## 5 Başarım ve Karşılaştırma

Bölüm 2'de değinildiği üzere, Türkçe bir ayrıştırıcının başarısı ölçülürken, bağımlı sözcüğün doğru sahip sözcüğün doğru *ÇG*'sine bağlanma oranı(*ÇG*-*ÇG* oranı) hesaplanmalıdır. Bu bölümde literatürde yer alan diğer çalışmalarla karşılaştırma yapabilmek amacıyla hem *ÇG*-*ÇG* oranı hem de sadece doğru sahip sözcüğe bağlanma (S-S oranı) oranı verilmektedir. Başarım ölçümü yapılırken, noktalama işaretlerine ve sözcük içerisindeki *ÇG*'lere ait bağlılıklar hesaplanmamaktadır. Sonuçlar bağlılık tipi gözetmeksizin sadece doğru *ÇG*'ler arasında bağlılık tespiti oranlarını yansıtmaktadır.

**Tablo 1.** Başarım

| | İkili Olasılık | | Kural Tabanlı | | Maltparser | |
|---|---|---|---|---|---|---|
| | *ÇG-ÇG* | S-S | *ÇG-ÇG* | S-S | *ÇG-ÇG* | S-S |
| Filitrelenmiş Bölüm | 73.5 | 81.2 | 73.4 | 82.7 | 78.1 | 85.1 |
| Tüm Derlem | x | x | 70.5 | 79.3 | 76.1 | 82.7 |

Türkçe'nin bağlılık analizi konusunda yapılan ilk çalışma Oflazer[14]'in kural tabanlı gerekirci ayrıştırıcısıdır. Ancak literatürde buradaki çalışmada kullanılan derlem üzerinde test edilmiş sadece iki bağlılık ayrıştırıcısı bulunmaktadır. Bunlardan birincisi Eryiğit ve Oflazer[15]'in basit ikili bağlılık olasılıklarına dayalı istatistiksel ayrıştırıcısıdır. Bu çalışmada, Türkçe'nin Bağlılık Analizi konusunda detaylı incelemeler yapılmış ve sonuçlar derlemin filitrelenmiş bir bölümü üzerinde verilmiştir. Ayrıştırıcı sadece izdüşel ve sahip sonda tipinde bağlılıklar içeren cümleler üzerinde test edilmiştir. Sonuçları karşılaştırabilmek üzere, Tablo 1'deki başarımlar hem bu filitrelenmiş bölüm üzerinde hem de tüm derlem üzerinde verilmiştir. İkinci ayrıştırıcı ise Nivre[13]'nin Maltparser isimli ayrıştırıcısıdır. Bu ayrıştırıcı Türkçe'nin *ÇG* yapısına uygun olarak yeniden modellenmiş ve Ayrıştırma modeli'nde sınıflandırıcı olarak Karar Destek Makineleri kullanılmıştır. Tabloda yer alan her üç ayrıştırıcı da en küçük ayrıştırma birimi olarak *ÇG*'leri kullanmaktadırlar. Kural Tabanlı ayrıştırıcımızın başarımı *ÇG-ÇG* oranı olarak Eryiğit ve Oflazer[15][7]'in ayrıştırıcısı ile aynı seviyedir. S-S oranının bu ayrıştırıcıdan biraz daha iyi olduğu görülmektedir. Buna karşın, ikili bağlılık olasılıklarına dayanan bu modele nazaran daha başarılı olduğu kanıtlanmış karar destek makinelerini kullanan Maltparser'ın başarımı her iki ayrıştırıcıya göre oldukça yüksektir.

## 6 Sonuç

Bu çalışmada, ODTÜ-Sabancı Ağaç Yapılı Derlemi üzerinde test edilmiş kural tabanlı bağlılık ayrıştırıcısı sunulmuştur. Elde edilen başarımlar, Türkçe'nin Bağlılık Analizi konusunda bundan sonra yapılacak çalışmalar için önemli bir temel oluşturma niteliğindedir. Bundan sonraki aşama olarak, ayrıştırıcının kurallarının bağlılık tiplerini de belirlemek üzere geliştirilmesi hedeflenmektedir. Bağlılık analizindeki başarımları oldukça yüksek olan istatistiksel modellerin, kural tabanlı modellerle birleştirilerek kullanılmasının başarımı daha da artıracağı öngörülmektedir. Bu nedenle, Türkçe için bu tip modeller oluşturulması hedeflenmektedir.

## Teşekkür

Bu çalışma Tübitak BAYG ve İTÜ BAP birimi tarafından desteklenmektedir.

[7] Aynı çalışmada temel modellerden biri olarak, buradaki ayrıştırıcıya benzer kural tabanlı bir ayrıştırıcının sonuçları verilmiştir. Ancak başarımı Tablo 1'de birinci kolonda yer alan sonuçlardan daha da düşüktür.

## Kaynakça


1. Hajic, J., Smrz, O., Zemanek, P., Snaidauf, J., Beska, E.: Prague arabic dependency treebank: Development in data and tools. In: Proceedings of the International Conference on Arabic Language Resources and Tools, Cairo, Egypt (2004)
2. Hajic, J., Hajicova, E., Pajas, P., Panevova, J., Sgall, P., Hladka B.: Prague dependency treebank 1.0 (final production label). CDROM CAT: LDC2001T10., ISBN 1-58563-212-0 (2001)
3. Kromann, M.T.: The Danish Dependency Treebank and the DTAG treebank tool, Vaxjo, Sweden, Proceedings of the Second Workshop on Treebanks and Linguistic Theories (2003)
4. Bosco, C.: A Grammatical Relation System for Treebank Annotation. PhD thesis, University of Torino (2004)
5. Lepage, Y., Shin-Ichit, A., Susumu, A., Hitoshi, I.: An annotated corpus in japanese using Tesniere's structural syntax. In: Proceeding of the Content Visualization and Intermedia Representations COLING-ACL'98, Montreal, Quebec, Canada (1998)
6. Dzeroski, S., Erjavec, T., Ledinek, N., Pajas, P., Zabokrtsky, Z., Zele, A.: Towards a slovene dependency treebank. In: Proceedings of the Fifth International Conference on Language Resources and Evaluation, Genoa, Italy (2006)
7. Oflazer, K., Say, B., Hakkani-Tür, D.Z., Tür, G.: Building a Turkish treebank. In Abeille, A., ed.: Building and Exploiting Syntactically-annotated Corpora. Kluwer Academic Publishers (2003)
8. Nivre, J., Nilsson, J.: Pseudo-projective dependency parsing. In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05), Ann Arbor, Michigan, Association for Computational Linguistics (2005) 99--106
9. Kudo, T., Matsumoto, Y.: Japanese dependency analysis based on support vector machines. In: Joint Sigdat Conference On Empirical Methods In Natural Language Processing and Very Large Corpora, Hong Kong (2000)
10. Kudo, T., Matsumoto, Y.: Japanese dependency analysis using cascaded chunking. In: Sixth Conference on Natural Language Learning, Taipei, Taiwan (2002)
11. Sekine, S., Uchimoto, K., Isahara, H.: Backward beam search algorithm for dependency analysis of Japanese. In: 17th International Conference on Computational Linguistics, Saarbrücken, Germany (2000) 754 -- 760
12. Yamada, H., Matsumoto, Y.: Statistical dependency analysis with support vector machines. In: 8th International Workshop of Parsing Technologies, Nancy, France (2003)
13. Nivre, J.: Inductive Dependency Parsing. PhD thesis, Vaxjo University, Sweden (2006)
14. Oflazer, K.: Dependency parsing with an extended finite-state approach. Computational Linguistics **29**(4) (2003)
15. Eryiğit, G., Oflazer, K.: Statistical dependency parsing of Turkish. In: Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics, Trento, Italy (2006)
16. Tesnière, L.: Elements de syntaxe structurale. Klincksieck edn., Paris (1959)
17. Bozşahin, C.: Gapping and word order in Turkish. In: Proceeding of the 10th Int. Conf. on Turkish Linguistics, Istanbul (2000)
18. Hakkani-Tür, D.Z., Tür, G., Oflazer, K.: Statistical morphological disambiguation for agglutinative languages. Computers and the Humanities 36(4) (2002) 381--410
19. Nivre, J.: An efficient algorithm for projective dependency parsing. In: Proceedings of the 8th International Workshop on Parsing Technologies, Nancy, France (2003) 23--25
20. McDonald, R., Pereira, F., Ribarov, K., Hajic, J.: Non-projective dependency parsing using spanning tree algorithms. In: Proceedings of the HLT-EMNLP. (2005)