**Doğal Türkçe Dil İfadelerinin SQL Sorgulama Diline Otomatik Olarak Çevrilmesi**

**Sedat Doğan1,\***

 *1Ondokuz Mayıs Üniversitesi, Mühendislik Fakültesi, Harita Mühendisliği Bölümü, 55200, Samsun.*

*Özet*

*Veritabanlarının sorgulanması için kullanılan dillerin hemen hepsi kendine özgü yapılar, anahtar sözcükler, kurallar vs. içermektedir. Bu kuralları bilmeyen ve bilgisayarların çalışma prensipleri ve programlama mantığını öğrenmemiş insanların bu dilleri kullanması mümkün değildir. İdeal olarak düşünüldüğünde, bir sorgulama dili (veya genel olarak bir yapay dil) doğal bir dile ne kadar çok benziyorsa, o dili kullanmak için doğal dili bilmenin ötesinde o kadar daha az özel ve teknik bilgiye gerek duyulacağı söylenebilir. Bu çalışmada, bir CBS veritabanının sorgulanması için yazar tarafından geliştirilmekte olan bir Türkçe-SQL çeviri sisteminin ilk sonuçları örneklerle açıklanmaktadır. Söz konusu sistemle, veritabanı sorgulama dillerini hiç bilmeyen kullanıcılar, Türkçe cümleler yazarak istedikleri sorgulamaları yapabileceklerdir. Makalede, ilk olarak konuyla ilgili literatür bilgileri ve kullanılabilecek yaklaşımlar özetle ele alınmış ve daha sonra geliştirilen Türkçe-SQL çeviri sisteminin kullandığı yaklaşımlar açıklanmıştır.*

Anahtar Sözcükler

Veritabanı, Sorgulama, Semantik, Doğal Dil İşleme, NLP

**1. Giriş**

Veritabanlarının sorgulanması için günümüzde bu amaçla geliştirilmiş sorgulama dilleri kullanılmaktadır. Hem ANSI hem de ISO standartları ile tanımlanmış en temel ve en genel sorgulama dili SQL (Structured Query Language) dilidir (Microsoft.com, 2013) . SQL, ilişkisel cebir ve ilişkisel dizi hesapları üzerine kurulmuştur. Kurgulandığı temeller doğrudan matematiksel formalizm üzerinde olduğu için, çalışma prensipleri doğrudan doğruya bu matematiksel yapılar arasındaki ilişkilerin yorumlanması esasına dayanmaktadır. Bu içkin ilişkiler, matematik fonksiyonlarla ifade edilen formel ifadeler (formüller) ve diziler arasındaki sıralanma kuramlarından oluşmaktadır. Dolayısıyla bu iki gösterim biçimlerinin sunduğu bilgiler, ancak bu ifadelerin anlamlarını bilen uzmanlar için anlaşılabilir durumdadır. SQL ile bir veritabanının güncelleştirilmesi, analiz edilmesi, analiz sonuçlarında elde edilen yeni verilerin veritabanında yönetilmesi, verilerin görüntülenmesi gibi her türlü veritabanı işlemi gerçekleştirilebilmektedir. Özünde aynı yaklaşımları kullanan LINQ (Language Integrated Query) (Grimes, 2002, Turtschi; Werry et al., 2002), XQuery (W3Schools.com, 2013) vs. gibi birçok sorgulama dili bulunmaktadır. Genelde sorgulama dilleri, dilin ifadelerinin yapısına bağlı olarak gruplandırılabilir. Eğer bir sorgulama dili doğal bir dile ne kadar yakınsa, o sorgulama dilini kullanmak o kadar kolay olur. Doğal dile daha yakın olan sorgulama dillerine “yüksek düzeyli dil”, uzak olanlara ise “düşük düzeyli dil” adı verilir. Bu tanıma göre, yukarıda sayılan sorgulama dillerinin hepsi de düşük düzeyli dillerdir. Ancak, bunların birbirlerine göre düzeyleri bağıl olarak farklıdır. Örneğin MS.NET Framework’ün sunduğu LINQ sorguları, SQL’e göre daha yüksek düzeylidir (Grimes, 2002). Ama ikisi de doğal dillerden çok uzak oldukları için düşük düzeyli diller grubundadır. Düşük düzeyli dilleri kullanarak sorgulama yapmak zorunda olanlar, bu dillerin bütün kurallarını, terimlerini, değişkenlerin nasıl tanımlanması gerektiğini, bunların nasıl kullanılacağını vs. öğrenmek zorundadır.

Yukarıda ifade edilen zorunluluğun ve onunla bağlantılı olarak ortaya çıkan zorlukların azaltılması ve hatta ortadan kaldırılması konusu; yapay zeka, dil felsefesi, matematiksel mantık, dil mantığı vb. bilim ve felsefe dallarının ve bunlarla ilişkili olan hemen her alanın temel araştırma konularından birisi olmuştur (Luger and Stubblefield, 1998; Rowe, 1988). Bu amaçla, doğal dillere yakın sorgulama dillerinin geliştirilmesi için birçok çalışma yapılmaktadır. Örneğin CQL (Contextual Query Language) adı verilen ve formel dil yaklaşımına dayanan bir yazılım sistemi, web üzerinde aramalar yapılırken, aramanın anahtar sözcükler yerine aranan içeriğe göre arama yapmayı amaçlayan bir dil olarak tasarlanmıştır (www.loc.gov, 2013). Burada asıl amaç, web arama motorlarında aranan konunun içeriğine uygun olan sonuçların geri döndürülmesidir. Bu sistem, veritabanları için tasarlanmamıştır. Yine doğrudan sorgulama amacı olmayan ve doğal diller arasında çeviri işlemlerinin gerçekleştirilmesi için kullanılan yazılım sistemleri de vardır. Proxem bu alandaki çok başarılı bir örnektir (Proxem.com, 2013). Proxem sistemi, çok geniş içerikli sözlüklere sahip veritabanları kullanmaktadır. Çeviri yapacağı dilin bütün gramer kurallarının ve bunların anlamla ilişkilerini detaylıca analiz ederek semantik içerikleri çıkarmaya çalışan ve bu konuda da çok başarılı olan bir sistemdir. Ancak Proxem’de, şu an için en iyi performansı İngilizce dilinde sağlamaktadır.

Türkçe Doğal dil işleme konusunda da çok değerli çalışmalar yapılmış ve birçok çalışma da devam etmektedir. Bu konuda özellikle bilgisayar mühendisleri hem doktora çalışması düzeyinde hem de yüksek lisans çalışması düzeyinde çok sayıda çalışmalar gerçekleştirmektedirler. Bu çalışmalar, en başta yabancı bir doğal dilden Türkçe’ye, Türkçeden de başka dillere çeviri konularında yoğunlaşmıştır. Bu alanda *zemberek* adı verilen ve Java platformunda geliştirilen bir Türkçe doğal dil işleme yazılımı da geliştirilmiştir . Ayrıca, Türkçe’yi semantik içerik ve dilin morfolojik özelliklerini ele alarak detaylıca inceleme konularında da çok önemli çalışmalar bulunmaktadır,(Akın ve Akın, 2007; Say vd., 2004; Kemal vd., 2004; Torunoglu vd., 2011; Zelal ve Kemal, 1994).

Çeviri sistemlerinin kurallarını tanımlama ve yorumlama işi zor bir iş gibi görünse de; aslında çeviri işleminin doğası, bir dildeki sözcükler ve bunlar arasındaki kuralların başka bir dildeki sözcük ve eşdeğer kurallarla ilişkilendirilmesi esasına dayandığı için bu makalede ele alınan soruna göre çözümü daha kolaydır. Çünkü, çeviri işlemlerinde çevrilen yapıların semantik içeriklerinin bilgisayar tarafından anlaşılması gerekmez. Dolayısıyla çevrilen ifadelerin ifade ettikleri anlamlarının somut olarak hayata geçirilmesi gerekmemektedir. Halbu ki, bu makalede ele alınan problemde, Türkçe cümlelerin SQL diline çevrilmesi işi yeterli değildir. Çünkü bilgisayar, kendisine iletilen ifadelerden ne yapması gerektiğini anlayıp o işleri yapmak zorundadır. Buna göre problem aslında, bir dilden diğerine çeviri problemi değil, dildeki içeriğin anlaşılıp yerine getirilmesinin sağlanması problemidir. Burada hiçbir SQL karşılığa çeviri yapmadan doğrudan Türkçe ifadelerde kastedilen içeriğin anlaşılıp gerçekleştirilmesi de mümkün olabilir. Makalenin başlığındaki SQL’e çeviriden kastedilen şey aslında, bilgisayarlar için SQL’in sanki doğal bir içsel sorgulama dili (veya bunu gerçekleştirecek doğal biyolojik yapı) gibi düşünülmesidir. Yoksa Türkçe sorguda ifade edilen bir veritabanı dosyasının bilgisayar tarafından açılması ve oradan istenen koşulları sağlayan kayıtları bulması için SQL’e gerek yoktur. Dosya açma, kaydetme, dosyaya yazma, bellek yönetimi vs. gibi işlemleri SQL değil işletim sistemi gerçekleştirmektedir. Başka bir ifadeyle, SQL ifadesini somut olarak gerçekleştirecek olan şey bilgisayarın işletim sistemidir (bilgisayarın biyolojik yetenekleriymiş gibi düşünülebilir).

Problemin çözümü için nasıl bir yöntemin kullanılması gerektiği ve yaklaşımların nasıl olacağını belirlemek gerekir. Bunun için somut bir bilimsel yöntem ortaya konmalıdır. Problemin çözümü için izlenen yaklaşımlar sonraki bölümlerde açıklanmıştır. Bu aşamada sorunun çözümü için yani ifadelerin anlamını oluşturan veya taşıyan içeriğin tanımlanması sorununun ve bu tanımların sunacağı anlamların da makineler tarafından nasıl anlaşılacağı sorusunun çözümü üzerinde yoğunlaşmak gerekir. Henüz başlangıçta, ciddi felsefe sorunlarıyla karşılaşılır. Son derece soyut olan ve ne olduğunun ifade edilmesi oldukça zor olan “anlam” kavramının tanımlanması ve bu tanımların düşünce etkinliklerinin somut bir şekilde ortaya konmasının mümkün olup olmadığı sorununa bir çıkış yolu oluşturacak cevabın verilmesi gerekir. Bu bağlamda, soyut “anlam” kavramının ve onun işaret ettiği formların somut olarak ele alınabilmesi için, gizemliymiş gibi görünen düşünsel sürecin ve onun gerçeklikle bağının nesnel bir olgu olarak ele alınmasını sağlayacak ve bunu sağlarken de ikna edecek bir teorik temele gereksinim vardır. Bu teorik temelin sağlanması için, düşünceleri dışa vurmanın şu an için en gelişmiş ve etkin yöntemi olan doğal dillerin analiz edilmesi gerektiğini öne süren semantik felsefe kullanılabilir. Zira insanların, ancak ve ancak doğal dil ile düşüncelerini paylaştığı, bu durumda hem o dilin sözcüklerinin hem de bunlar arasındaki ilişkilerin, iletilen düşüncelerin iletilecek bir forma dönüşümünü sağladığı gerçeği fark edilirse, o zaman bu dil formunun, ilk bakışta çok soyutmuş gibi görünen düşünsel kavramların somut gösterimleri olduğu açıkça görülebilir. Bu açıklık da, aslında “anlam” kavramlarının, dil analizi ile nesnel bir şekilde incelenebileceği konusunda ikna edici bir temel sağlamaktadır. Konuyla ilgili olarak okuyucular (De Rijk, 2002; Evans ve Green, 2006; Hançerlioğlu, 2010; Laycock, 2006; Macbeth, 2005) kaynaklarına başvurabilirler.

Bu makalede, felsefe sorunlarına fazlaca girmeden, dar kapsamlı da olsa somut bir Türkçe-SQL çevirisinin gerçekleştirilmesi için kullanılan yaklaşımlar ve bu yaklaşımlarla gerçekleştirilen uygulama örnekleri verilmiştir.

**2. Problemin Çözümü için Nasıl Bir Yöntem Gerekir?**

Doğal Türkçe dil ifadelerinin SQL diline çevrilmesini ve bilgisayarın Türkçe ifadelerdeki istekleri SQL’i kullanarak gerçekleştirmesini sağlama problemi ilk kez ele alınmakta ve bu nedenle henüz üzerinde uzlaşma sağlanmış bir yöntem veya teorik temel bulunmamaktadır. Bu nedenle bu bölümde, problemin çözümü için önerilen yaklaşımlar ortaya konacak ve bu yaklaşımlara dayanılarak geliştirilen ilk prototip sistemin basit sorguları nasıl gerçekleştirdiğini gösteren örnekler ise sonraki bölümlerde verilecektir.

Bilgisayar ile kullanıcı, Türkçe yazılan ifadelerle iletişim kuracakları için ve bu iletişim tek yönlü olacağı için (yani kullanıcıdan bilgisayara) bilgisayarın, yazılan Türkçe ifadelerdeki sözcüklerin anlamını bilmesi gerekir. Diğer taraftan bir ifadede, anlamlı içerik taşıyan en küçük birimlerin cümleler olduğu söylenebilir. Bu nedenle, anlam çıkarma probleminin çözümüne ilk olarak yazılan ifadelerin cümlelere ayrıştırılmasıyla başlamak gerekir. Bir cümlede anlamı oluşturan elemanlar ise sözcükler ve bunlar arasındaki bağlantı kurallarıdır. İkinci olarak sözcüklerin ayrıştırılması gerekir. Genel olarak problemin çözüm aşamaları Şekil 1’de ki gibi gösterilebilir.

Bu ilk çalışmada, Şekil 1’deki genel şemada gösterilen işlemler, şekildeki sıra ile ele alınacaktır. “Anlam Analiz Sisteminde” (AAS); sözcüklerin sınıflandırılması, kuralların belirlenmesi, sözlüğün gerekliliği ve nasıl olması gerektiği, kurallarla sözcük etkileşimlerinin “anlamı” nasıl oluşturduğu vb. birçok konuda çok önemli felsefi sorgulamaların yapılması ve ikna edici kanıtların ortaya konması şarttır. Konuyla ilgili felsefe sorgulamaları yazar tarafından uzun süredir yapılmaktadır ve çok önemli sonuçlar elde edilmiştir. Ancak burada yer azlığı nedeniyle bütün hepsinin detaylıca açıklanması mümkün değildir. Bunun için en az iki cilt kitap sayfasına ihtiyaç vardır. Bununla birlikte, olası karışıklıklara yol açmayacak biçimde ve ikna edici olacağı umulan bazı önemli sonuçlar bu makalede paylaşılacaktır.

* 1. **Sistemin Geliştirilmesi için Kullanılan Araçlar**

Bir anlam analiz sistemi somut olarak bir bilgisayar yazılımı geliştirilerek hayata geçirilir. Bu nedenle, sistemin çalışması sırasında her bir işlem adımı analiz için gereken bilgileri bellekte saklayabilmeli ve girilen bilgilerden çıkardığı ilişkileri de uygun veri modellerini kullanarak gerektiğinde programın akışı esnasında gerçekleştirilecek ileriki analizlerde de kullanılabilecek şekilde bellekte yönetilmelidir.

Türkçe

Sözlük

Sorgunun gerçekleştirilmesi

Türkçe ifadelerin girilmesi

Cümlelerin ayrıştırılması

Sözcüklerin ayrıştırılması

Sözcüklerin sınıflandırılması

Cümle öğelerinin bulunması

Anlam çıkarma işlemleri

Anlam grafiğinin oluşturulması

SQL

İfadesi

Türkçe yazım kuralları

Türkçe gramer kuralları

ANLAM ANALİZ SİSTEMİ

*Şekil 1: Anlam analiz sistemi*

Bu çalışmada, AAS sistemi C# programlama dili ile geliştirilmiştir. C#, .NET Framework’ün bütün sınıflarını da kullanabilmektedir. Bu özelliği sayesinde kullanılabilecek çok sayıda veri model ve yapıları sunmaktadır. Örneğin; diziler, koleksiyonlar, liste yapıları, tablolar, ağaçlar, yığıtlar, kuyruklar vb. bir bilgisayar sisteminde kullanılabilecek her türlü veri modelini kullanıma sunmaktadır. Burada asıl sorun hangi veri yapılarının ne amaçla ve nasıl kullanılacağına karar verme sorunudur. Bu kararı verebilmek için verilerin birbiriyle hangi düzeylerde ilişkili oldukları, bu ilişkilerin hangi yeni bilgileri ürettiği, üretilen bu yeni bilgilerin mevcut bilgilerle nasıl bir ilişki kuracağı ve bunları hangi veri yapılarıyla yapacağı sorularına cevap vermek gerekmektedir.

Bir Türkçe ifade, bir *string* dizi tipinde sisteme girilebilir. Ancak bu haliyle bu diziden hiçbir anlam çıkarmak mümkün değildir. Şekil 1’de gösterildiği gibi, bu ifade kümesinin önce cümlelere, cümlelerin de sözcüklere ayrıştırılması gerekir. Her bir ayrıştırılan cümle yeni bir *string* dizi değişkeni olur. Bu cümleler bellekte ayrı değişkenlerle saklanmalıdır. Benzer şekilde, ayrıştırılan her bir sözcük yine ayrı ayrı değişkenlerle bellekte saklanmalıdır. Bu saklama işlemi, hangi sözcüğün hangi cümlede ve hangi sırada bulunduğu bilgilerini de açık bir şekilde ve uygun başka değişkenlerle bellekte tutmalı ve bütün hepsini genelden özele ve özelden genele doğru hareket ederek birlikte analiz etme olanağını da sağlamalıdır.

Diğer taraftan, ayrıştırılan her bir Türkçe sözcüğün köklerine eklenmiş ekleri de bulunabilir. Hem bu tür eklerin analizi aşamasında hem de sözcüklerin türlerinin belirlenmesi için yapılacak ileriki analizlerde, belleğe ayrıştırılarak kaydedilen bütün bilgiler kolayca analiz edilebilir bir modele sahip olmalı ve analizler esnasında hiçbir önceki ilişkisini ve bilgilerini kaybetmeyecek şekilde tasarlanmalıdır. Bilgisayar programlama teknikleri açısından bakıldığında, yukarıda bahsedilen ifadelerin ayrıştırılması veya bileşenlerine ayrılması işlemleri arama (search) işlemleridir. Örneğin sözcükler cümlelerden ayrıştırılırken, sözcükler arasındaki boşluklar, imla işaretleri vb. karakterlerin aranıp bulunması; bunların bulundukları sıraya bağlı olarak kesilip yeni değişkenlere kopyalanması gibi birçok önişlemin gerçekleştirilmesi gerekir. Bu işlemler, üzerinde çalışılan doğal veya yapay hangi dil olursa olsun mutlaka gerçekleştirilmesi gereken önişlemlerdir. Tüm diller insan aklının ürünü olduğu için ve akıl evrensel ilkelerle çalıştığı için; bütün doğal dillere evrensel aklın evrensel formları biraz farklı biçimlerde de olsa kodlanmıştır. Kodlanan bu evrensel formlar zorunlu olarak, aynı düzenli mantık yasalarına uyarlar. Özü bu zorunluluklara dayanan ve doğal ya da yapay dil dizgeleri içerisindeki düzenli ifadeler veya örüntülerin hızlıca aranıp bulunmasını sağlayan düzenli ifade bulma araçları geliştirilmiştir. Bu çalışmada, bu amaçla “***Regex (Regular Expressions)”*** veya ***“Düzenli İfadeler”*** adı verilen API (Application Programming Interface) fonksiyonları kullanılmıştır (Stubblebine, 2007). Regex, bir dili formel olarak karakter dizileri kümesi gibi ele alan ve bu küme üzerinde küme kavramı ve mantık yasalarını kullanarak her türlü ayrıştırma amacını gerçekleştirebilecek formel yapıların tanımlanmasını sağlayan bir dil mantığı (linguistic logic) aracıdır. Tanımlanan formel yapıların hızlı bir şekilde işlenmesi için de bilgi teorisi ile ortaya atılan “sonlu durum otomatı” ***(finite state automata)*** yaklaşımı ile yorumlanmıştır. Regex konusuna bundan sonra değinilmeyecek, sadece gerektiğinde analiz için yazılan regex patern örnekleri verilecektir.

* 1. **İfadelerden Cümleler ve Sözcüklerin Ayrıştırılması**

Giriş Türkçe ifadeleri, geliştirilen AAS sisteminde sorgulama penceresinden yapılmaktadır. Sorgulama penceresi, içerisine her türlü metin ifadelerinin yazılabileceği bir zengin metin kutusu yani ***RichTextBox*** nesnesidir. Yazılan bütün ifadeler, ***string*** tipindeki bir tek ***Text*** değişkeninde tutulmaktadır. Önce cümleler, daha sonra da her bir cümle sırasıyla ayrıştırılmaktadır. Cümleleri ve cümlelerin sözcüklerini ayrıştırmak için ayırıcı karakter dizileri kullanılmıştır. Bu diziler aşağıdaki gibi tanımlanmıştır:

**char[] SentenceSplitter = { '.','?', '!', ':' };** //Cümle ayırıcı karakterler.

**char[] WordSplitter = { ' ','\n',',', ':', '!', '?',';','\'','\"','(',')','.' };** // Sözcük ayırıcı karakterler.

Her bir cümle için bir ***Hastable*** nesnesi oluşturulmakta ve ayrıştırılan sözcükler cümle içerisindeki sıraları anahtar değer ve kendileri de içerik değeri olacak şekilde ***Key-Value*** çiftleri halinde bellekte tutulmaktadır.

1. **Cümlelerin Öğelerinin Bulunması**

Cümlenin öğeleri, cümle ile ifade edilen soyut konseptlerin, somut gerçek dünya ile ilişkilendirilmesi görevlerini üstlenen sembollerdir. Doğal bir dilde bu sembollere sözcükler adı verilmektedir. Bir konsept veya bir soyut düşünce imgesi gerçek dünyada bir nesne, olay veya olgu ile eşleştiği anda (ya da bunları işaret ettiği anda) ***anlam*** kazanır. Gerçek dünyadaki bir varlık durumunu işaret etmeyen imgelere **anlamsızdır** denir. Gerçek dünyadaki gerçek bir varlık durumunu ifade etmediği halde, kendi içerisinde tutarlı ve sıkı kurallar ile tanımlanmış imgeler tasarlanabilir. Bu tür imgelere de **mantıklı** imgeler adı verilir. Mantıklı imgelerin gerçek karşılıklarının bulunması gerekmez. Yani, mantıklı bir imgeler grubu anlamsız olabilir. Bununla birlikte, anlamlı konseptlerin aynı zamanda mantıklı ilişkilere de sahip olmalıdır. Bu son tahlile göre, ***anlamlı bir ifade*** gerçek dünyada var olan varlık durumlarını işaret ederken, aynı anda kendi içerisinde de mantıksal kurallar içeriyor olması şarttır. Bu durumda, doğal bir dilin cümlelerinin anlamlı olabilmesi için cümlenin öğeleri arasında kuralların var olması ve aynı zamanda bütün olarak cümlenin, gerçek varlık durumlarını işaret ediyor olması şarttır.

 Bir konsept, konseptler oluşturabilen varlıklarla taşınır. Konsept taşıyan varlıklar akıllı varlıklardır yani insanlardır. Bir konseptin gerçek dünyayla ilişki kurabilmesi için gizli veya açık bir taşıyıcıya (özneye) ihtiyacı vardır. Konsept tek başına özne ile gerçekle ilişki kuramaz. Ayrıca ilişki kurma eyleminin bir biçimde gerçekleşiyor olması da gerekir. Bu eylemi özne gerçekleştirir. Öznenin gerçekleştirebileceği eylemlerin hepsi, sahip olduğu bedeninin uzay-zamandaki kapasitesi ile sınırlıdır. Beden kapasitesi ile gerçekleştirilen tüm eylemler, gerçek dünyadaki gerçek ve doğrudan tanık olunabilen eylemlerdir. Bu tür eylemlerin hepsi fiil adı verilen sözcük gruplarıyla ifade edilir ve her cümlenin mutlaka bir tane fiil türünde öğesi bulunmalıdır. Bu fiil özne tarafından gerçekleştirileceği için fiilin yüklenicisi (sorumlusu) öznedir. Özne tarafından yüklenilen fiillere cümlede ***“yüklem”*** adı verilir. Buna göre bir cümle analizinde ilk önce cümlenin anlamında anahtar role sahip iki öğenin; *özne ve yüklemin* belirlenmesi gerekir. Daha sonra, bu iki temel öğenin uzay-zamandaki konumunu ve ilişkili oldukları diğer varlık durumlarını tanımlayan ve bu sayede konsepti gerçek dünyada (uzay-zamanda) eksiksiz olarak somutlaştıran diğer öğelerin belirlenmesi gerekir.

Sonraki bölümlerde sırasıyla; yüklem, özne ve nesnelerin belirlenmesi için çalışmada kullanılan yaklaşımlar açıklanacaktır. Bu ilk çalışmada, varlıkların niteliklerini, görünümünü, niceliklerini ifade eden öğeler (sıfatlar) ele alınmamıştır. Zaten veritabanı sorgularında çoğunlukla bunlara gerek de yoktur. Buna rağmen Türkçe ile ilgili tüm yapıların analizi üzerine çalışmalar, AAS sisteminin geliştirilmesi için devam edecektir.

* 1. **Cümlelerin Yüklem ve Öznelerinin Bulunması**

Çalışmada amaçlanan “sorgulama” işleminin gerçekleştirilmesi için önce bir diyalog gerçekleşecektir. Diyalog sonucunda üzerinde uzlaşılan özne, gereğini yapacaktır. Bu diyalog bütün süreç boyunca iki özne arasında gerçekleşecektir. Bunlar; insan ve bilgisayardır.

Özne analizinde, özneyi belirlemek demek öznenin insan mı yoksa bilgisayar mı olduğunu belirlemek demektir. Burada iki şey söz konusu olabilir. Diyalog, insanın belirleyeceği yolda ilerleyeceği için insan, makineyle girdiği diyalogda iki şey söyleyebilir: Ya istediği eylemi “birlikte yapalım” der ya da “yap” diye emir kipiyle seslenir. Her iki durumda da makinenin itiraz etme hakkı yoktur (bu hakkın bilgisayara verilmediği varsayılmaktadır). Bu durumda, eylemi gerçekleştirecek olan özne bilgisayardır. En basit haliyle bu durumda fiilin öznesi her zaman “bilgisayar” olmalıdır. Ancak, AAS sistemini kullanacak olan kullanıcı, cümleleri yazarken ya hatayla ya da bilerek veya Türkçe ifade yeteneklerinin kısıtlı olması vb. nedenlerle fiilin özneye göre çekimlerini farklı yazabilir. Bu yüzden her durumda, geliştirilen sistem yine de özne analizini gerçekleştirmektedir.

Özneden önce, cümlenin yüklemi belirlenmelidir. Çünkü cümlenin tüm öğeleriyle ilgili en çok bilgi barındıran öğe yüklemdir. Her doğal dilde eylemleri ifade eden sözcükler vardır. Türkçede de eylemleri ifade eden çok sayıda fiil türünde sözcükler vardır. Bir doğal dille iletişim kuracak iki öznenin de semboller ve bunların kullanım kuralları üzerinde uzlaşma sağlamış olmaları gerekir. Aksi halde iletişim kuramazlar. Semboller ve kurallar öznelerin bunları içselleştirmeleri için hazırlanmış tablolar gibi düşünülebilir. Tablo, bir sütununda sembol, karşısında ise bunu açıklayan bir resim veya başka bir semboller dizisi gibi düşünülebilir. Doğal dillerde sembol tabloları sözlüklere, kurallar ise dilin gramer kurallarına karşılık gelir. Buna göre, AAS sisteminde, bir cümlenin yükleminin bulunabilmesi için mutlaka kullanıcı ve bilgisayarın üzerinde uzlaşacağı bir sözlüğün olması gerekir. Bu amaçla, veritabanı yönetim sisteminin sorgulanması için yeterli sayılabilecek fiil sözcükleri ile bir sözlük oluşturulmuştur. İleriki aşamalarda, Türk Dil Kurumu’nun hazırladığı resmi Türkçe sözlüğün tamamı bu amaçla kullanılabilir. Tablo 1’de, AAS sisteminde kullanılan fiil sözcükler verilmiştir.

Tablo 1’deki fiil sözcükleri, bir ***Hashtable*** yapısıyla bellekte tutulmaktadır. Bir cümlenin yüklemini belirleyebilmek için fiil sözlüğündeki köklerden hangisinin cümlede yer aldığını bulmak gerekir. Eğer cümlede sadece bir tane fiil varsa o fiil cümlenin yüklemi olmalıdır. Eğer cümlede birden fazla fiil köklü sözcük bulunuyorsa, bu durumda her bir fiil köklü sözcüğün eklerinin detaylıca analiz edilmesi gerekir. Eğer cümlede fiil köklü sözcük yoksa, bu durumda yüklem mutlaka bir durum tespiti yapan ifadedir ve sonuna (-dir, -dır, -dur- dür vb.) ekler gelir. Gerek yüklemlerin belirlenmesi aşamasında, gerekse cümlenin nesne öğelerinin belirlenmesinde Türkçe sözcüklerin sonuna eklenen ek türlerini çok iyi analiz etmek gerekir. Sonraki bölümde, Türkçedeki ekler konusuyla ilgili özet bilgiler, AAS sisteminin ilk tasarımında kullanıldığı kadarıyla ele alınacaktır.

*Tablo 1: AAS sisteminde fiil türündeki sözcükler.*

|  |  |  |  |
| --- | --- | --- | --- |
| **Fiilin Kökü** | **Mastar Eki** | **Fiilin Kökü** | **Mastar Eki** |
| bul | mak | seç | mek |
| kullan | mak | göster | mek |
| analiz et | mek | ara | mak |
| aç | mak | hesapla | mak |
| kapat | mak | çiz | mek |

1. **Türkçe’de Ekler**

Sözcüklerin kök veya gövdelerine eklenerek, bunların cümle içerisindeki görevlerini belirleyen, onlara değişik anlamlar katan ya da onlardan yeni sözcükler türeten dil parçalarına ***ek*** denir (Akın and Akın, 2007). Bu aşamada AAS sisteminin ilk ön-modeli için tanımda ifade edilenler içerisinde en öncelikli olanı, eklerin eklendikleri sözcüklerin cümle içerisindeki görevlerini nasıl değiştirdiği sorusudur.

Ekler, çekim ekleri ve yapım ekleri olmak üzere iki ana gruba ayrılırlar. Bu ekler, isim köklü ve fiil köklü sözcüklerin sonlarına eklenirler. Eklerin sözcüklere eklenmesi sırasında Türkçenin büyük ve küçük ses uyumu kuralları ekin hangi sesle geleceğini belirler. Ses uyumu kurallarını ***Regex*** örüntüleri şeklinde ayrıca tanımlamak mümkündür. Ancak bu tanımların yapılabilmesi için ***Regex*** dil mantığı işlemlerinin detaylıca bilinmesi gerekir. Bu nedenle bu makalede, ses uyumu konusuna girilmemiştir ancak, ses uyumuyla ilgili sorunlar, geliştirilen AAS sisteminde *Regex* örüntüleri ile çözülmüştür.

* 1. **İsim Ekleri**

İsim ekleri iki gruba ayrılabilir. Bunlar isim çekim ekleri ve isim yapım ekleridir. **İsim çekim ekleri;** isim türündeki sözcüklerin sonuna eklenerek, bu isimlerin cümlede hem bir görev hem de bir anlam kazanmasını sağlayan eklerdir. **İsim yapım ekleri** ise; sözcüklerin sonlarına eklenerek onlardan başka isim türeten (veya üreten) eklerdir. Yapım ekleri ile üretilen sözcükler yeni sözcüklerdir ve bunların üretilmiş anlamları kök anlamlarından farklı hale gelir. Bu nedenle, farklı anlamlara gelen sözcükler AAS sisteminin isim sözlüğünde ayrı ayrı tanımlanmaktadır. Bu nedenle, isim yapım eklerinin analizine gerek kalmamaktadır. İsim çekim ekleri, eklendikleri sözcüklerde ortaya çıkardıkları anlamların özelliklerine göre beş gruba ayrılırlar. Tablo 2’de, bu beş grup çekim eki özet olarak verilmiştir.

*Tablo 2: İsim çekim ekleri*

|  |  |  |
| --- | --- | --- |
| **Ekin türü** | **Ek** | **İşlevi** |
| Çokluk Eki | { -lar, -ler } | İsimlerin sayı bakımından çokluğunu bildirmektir. |
| Hal (Durum) Eki | Belirtme hali: { -ı, -i, -u, -ü }Yönelme hali: { -a, -e }Bulunma hali: { -da, -de, -ta, -te }Çıkma hali: { -den, -dan, -ten, -tan }Yalın hal: { - } | BH: yüklemin neyi/ kimi etkilediğini gösterir.YH: yer/ zaman (ismi zaman zarfı yapar) bildirir.BuH: yer/ durum/ zaman bildirir. İsimleri sıfat yapabilir.ÇH: yerden/ zamandan çıkma veya karşılaştırma veya neden bildirir.YH: isimlerin hal eki almamış şeklidir. İsimler, hal ekleri dışındaki çekim eklerini aldıklarında yalın halden çıkmaz. |
| Eşitlik Eki | { -ce, -ca, -çe, -ça } | İsimlere değişik anlamlar katan ve anlama bağlı olarak onları sıfat veya zarf yapan eklerdir.  |
| İyelik Eki | { -im, -ın,-in, -un, ün } | Eklendiği ismin kime/ neye ait olduğunu gösterir. Altı kişi zamirine göre çekimlenir. Çekim ekleri, yapım eklerinden sonra eklenir. |
| İlgi Ekleri | { (y)-im, (y)-ım, (y)-üm, (y)-um }{ -sin, -sın, - sün, -sun }{ -dir, -dır, -dür, -dur, -tir, -tır, -tür, -tur }{ (y)-iz, (y)-ız, (y)-üz, (y)-uz }{ -siniz, -sınız, -sünüz, -sunuz }{ -dirler, -dırlar, -dürler, -durlar, -tirler, - tırlar, -türler, -turlar } | İyelik ekiyle çok sıkı biçimde ilişkisi vardır. Eklendiği isme ait olan başka bir sözün varlığını gösterir. Bağlı olduğu isim, ilgi ekli isimden sonra gelir.  |

* 1. **Fiil Ekleri**

Bu bölümde sadece fiillerin zaman ekleri ve bunların öznelere göre çekim ekleri ele alınacaktır. Çünkü sıfat-fiil veya isim-fiil türlerini oluşturan eklerle üretilmiş sözcüklerin her birisi sözlükte ayrı semboller olarak tanımlanmalıdır.

Bir sorgu diyalogunda, genellikle emir kipleri, gelecek zaman kipleri ve şimdiki zaman kipleri daha sık kullanılır. Mantıksal olarak bakıldığında, geçmiş zaman kipleri, gelecek zaman kipleri ve geniş zaman kipleri sorgularda genellikle kullanılmaz. Örneğin, “X tablosunu kullanmıştın.”, “X tablosundan Y alanını alıyormuşsun.”, “Kayıtları seçerdik.” Gibi cümleler bir sorgu ifadesinde kullanılmaz. Buna rağmen, geliştirilen AAS sisteminin başka alanlarda da kullanılabileceği düşünülerek, bütün zaman kipleri ve bunların çekimleri sistemde detaylıca analiz edilmektedir. Fiil zaman ekleri Tablo 3’de verilmiştir.

*Tablo 3: Fiil Zaman Ekleri*

|  |  |  |
| --- | --- | --- |
| **Zaman Kipi** | **Zaman Ekleri** | **Özne Çekimleri** |
| Belirli geçmiş zaman (di’li geçmiş zaman) | { -di, -ti, -dı, -tı, -du, -tu, -dü, -tü } | { (y)-dim, (y)-dım, (y)-düm, (y)-dum, -tim, -tım, -tüm, -tum }{ (y)-din, (y)-dın, (y)-dün, (y)-dun, -tin, -tın, -tün, -tun }{ (y)-di, (y)-dı, (y)-dü, (y)-du, -ti, -tı, -tü, -tu }{ (y)-dik, (y)-dık, (y)-dük, (y)-duk, -tik, -tık, -tük, -tuk }{ (y)-diniz, (y)-dınız, (y)-dünüz, (y)-dunuz, -tiniz, -tınız, -tünüz, -tunuz }{ (y)-diler, (y)-dılar, (y)-düler, (y)-dular, -tiler, -tılar, -tüler, -tular } |
| Belirsiz geçmiş zaman (miş’li geçmiş zaman) | { -miş, -mış, -muş, -müş } | { -mişim, -mışım, -müşüm, -muşum }{ -mişsin, -mışsın, -müşsün, -muşsun }{ -miş, -mış, -muş, -müş }{ -mişiz, -mışız, -müşüz, -muşuz }{ -mişsiniz, -mışsınız, -müşsünüz, -muşsunuz }{ -mişler, -mışlar, -müşler, -muşlar } |
| Şimdiki zaman | { (i)-yor, (ı)-yor, (u)-yor, (ü)-yor, -yor } | { (ü)-yorum, -yorum }{ (ü)-yorsun, -yorsun } { (ü)-yor, -yor } { (ü)-yoruz, -yoruz } { (ü)-yorsunuz, -yorsunuz } { (ü)-yorlar, -yorlar } |
| Gelecek zaman | { (y)-ecek, (y)-acak } | { (y)-ecek(ğ)im, (y)-acak(ğ)ım }{ (y)-eceksin, (y)-acaksın }{ (y)-ecek, (y)-acak }{ (y)-ecek(ğ)iz, (y)-acak(ğ)ız }{ (y)-eceksiniz, (y)-acaksınız }{ (y)-ecekler, (y)-acaklar } |
| Geniş zaman | { -r, -ar, -er, -ir, -ır, -ur, -ür } | { -rim, -arım, -erim, -irim, -ırım, -urum, -ürüm }{ -rsin, -arsın, -ersin, -irsin, -ırsın, -ursun, -ürsün }{ -r, -ar, -er, -ir, -ır, -ur, -ür }{ -riz, -arız, -eriz, -iriz, -ırız, -uruz, -ürüz }{ -rsiniz, -arsınız, -ersiniz, -irsiniz, -ırsınız, -ursunuz, -ürsünüz }{ -rlar, -arlar, -erler, -irler, -ırlar, -urlar, -ürler } |
| Dilekkipi | { (y)-e, (y)-a } | { (y)-e(y)im, (y)-a(y)ım } 1.tekil özne{ (y)-elim, (y)-alım } 1.çoğul özne |
| Şartkipi | { -se, -sa } | { -sem, -sam }{ -sen, -san }{ -se, -sa }{ -sek, -sak }{ -seniz, -sanız }{-seler, -salar } |
| Zorunluluk kipi | { -malı, -meli } | {-malıyım, -meliyim }{-malısın, -melisin }{ -malı, -meli }{-malıyız, -meliyiz }{-malısınız, -melisiniz }{ -malılar, -meliler } |
| Emir kipi | { (y)-in, (y)-ın, (y)-un, (y)-ün } | {-} 2.tekil özne (fiil yalın halde){ (y)-in(iz), (y)-ın(ız), (y)-un(uz), (y)-ün(üz) }2.çoğul özne /2. Tekil özne için rica){ -sin, -sın,-sun, -sün }3.tekil özne{ -inler, -ınlar, -unlar, -ünler } 3.çoğul özne |

Tablo 3’deki zaman eklerinden başka, sözkonusu kiplerin rivayet ve hikaye bildiren formları da vardır. Ancak burada bunlar ele alınmamıştır. Ekler konusunda detaylı bilgiler için (Guise, 2013, TDK., 2013) ve diğer Türk Dili kitaplarına başvurulabilir.

1. **Geliştirilen AAS Sisteminin Yapısı ve Çalışma Prensipleri**

AAS sisteminde, analizleri gerçekleştirecek sınıflar tanımlanmıştır. Sistem ilk olarak cümleleri ve her bir cümlenin sözcüklerini ayrıştırmaktadır. Daha sonra ayrıştırılan bileşenler, *Regex* örüntüleri kullanılarak analiz edilmektedir. İlk olarak cümlelerdeki fiil köklü sözcükler belirlenmektedir. Her bir cümlenin öğelerinin detaylı bir şekilde analizinden sonra elde edilen bilgiler, cümlelerin her birisi için ayrı ayrı oluşturulan veri tablolarına kaydedilmektedir. Şekil 2’de Türkçe sorgu ifadelerinin yazılması ve Şekil 3’de ise öğelerin bilgilerini içeren tablolar ve oluşturulan anlam grafiği gösterilmiştir.

Şekil 3’de görüldüğü gibi, her bir cümle için sözcük eklerinin etkileri de dikkate alınarak sözcüklerin kökü, çokluk ekleri, cümlenin yüklemi, zaman kipi, öznesi, nesne ve yer sözcükleri ile ilgili tüm bilgiler analizler sırasında otomatik olarak elde edilmiş ve ilgili sütunlara yazılmıştır. Bu bilgiler sırayla elde edilmektedir. Örneğin, önce yüklem belirlenmekte sonra ilgi ekleri ve zaman ekleri birlikte analiz edilerek hem özne hem de yüklemin zaman kipi bulunmaktadır. Özne ve zaman analizlerini gerçekleştirmek için ***sdTenseAnalyzer*** adında bir C# sınıfı geliştirilmiştir.

**

*Şekil 2: Geliştirilen sistemde Türkçe sorgu ifadesinin girilmesi.*

Özne ve zaman kipi belirlendikten sonra, cümlenin nesnelerinin belirlenmesi gerekir. Bunun için de ***sdObjectAnalyzer*** adında bir C# sınıfı yazılmıştır. Nesnelerin analizine başlamadan önce, cümlelerin içerisindeki edatların, bağlaçların ve sıfatların elenmesi gerekir. Çünkü bu türdeki sözcükler doğrudan bir nesneyi ifade etmezler. Bu eleme için edat, bağlaç ve sıfat türündeki sözcüklerin yer aldığı sözlükler kullanılmıştır. Sistemdeki bütün sözlükleri oluşturan, arayan ve analizlere hazır hale getiren ***sdLexicon*** isimli bir C# sınıfı geliştirilmiştir. Her bir sözcük türü için ayrı bir sözlüğün kullanılması yerine, sadece bir tane sözlük de kullanılabilir. Bir tane sözlüğün kullanılması durumunda, sözlükteki her bir sözcüğün açıklama alanında, sözcüğün türünü ifade eden özel semboller bulunmalıdır. Hatta sözlükte açıklama alanlarına, açıklanan sözcüğün anlamca ilişkisi olan diğer sözcükler de verilebilir. Bu sayede semantik analizler daha da kolay bir şekilde gerçekleştirilebilir. Bu konuya bu makalede değinilmeyecektir.

Nesne olması mümkün olmayan sözcükler elendikten sonra, kalan nesne adayı sözcüklerin analizine başlanmaktadır. Geriye kalan sözcükler, isim sözlüğüyle karşılaştırılır. Sözlükte eşi bulunan sözcükler cümlenin nesne öğesi olmaya aday olan sözcüklerdir. İsim sözlüğünde karşılığı bulunmayan sözcüklere AAS sisteminde “bağımsız sözcükler” adı verilmiştir. Bu bağımsız sözcükler, özel isimler veya tablo/ kayıt vb. gibi bir varlık sınıfına ait somut bir nesnenin adıdır. Başlangıçta bu sözcüklerin cümledeki hangi nesne veya varlık sınıfına ve nasıl bir anlamla bağlı olduğu bilinmez. Analiz sonucunda SQL sorgusunu gerçekleştirmek için bu bağımsız nesnelerin semantik ilişkilerinin bulunması gerekir. Bu ilişkilerin çıkarılabilmesi için cümlelerin nesne olan öğelerinin belirlenmesi gerekir.

Şekil 4’ de, sorgulama ifadesinin sonuçları gösterilmektedir. Şekilde şartı sağlayan 22 tane köy ve şartı sağlamayan 5 köyden farklı renkte gösterilmiştir.

**

*Şekil 3: Cümlelerin öğe bilgileri ve anlam grafiği.*

**

*Şekil 4: Sorgu sonuçları.*

Nesnelerin türleri, belirli mi yoksa belirsiz mi olduklarını, bir yeri mi yoksa yerdeki bir nesneyi mi işaret ettiklerini ortaya çıkaracak şekilde analiz edilmektedir.

1. **Cümle Öğelerinin Anlam Grafiğinin Oluşturulması**

Cümle ifadelerinin temsil ettiği anlamı ortaya çıkarmak için yönlü grafikler kullanılmıştır. Analiz işlemleri gerçekleştirilirken, cümle öğelerinin bağlanma biçimleri de analiz edilmektedir. Bunun için “sıfat tamlamaları”, “iaim tamlamaları”, “yönelme biçimleri” vb. gerçek dünyada karşılığı bulunan fiziksel olgular ortaya çıkarılmaktadır. Bunu yapabilmek için hem belirlenen ekler, hem de sisteme tanımlanan gramer kuralları kullanılmıştır.

Analiz sonucunda, birbiriyle ilişkili olan nesneler, grafiğin düğüm noktalarını oluşturmaktadır. Nesneler arasındaki ilişkiler ise yönlü bağlantılarla gösterilmektedir. Her bir bağlantı, bağlantının (ilişkinin) türünü ifade etiket (açıklama) alanlarına sahiptir. Bu grafik, bilgisayarın sorguları gerçekleştirmek için ihtiyaç duyduğu tüm bilgileri taşımaktadır. Bilgisayar, grafikteki yönleri izleyerek Türkçe ifadenin karşılığı olan SQL ifadesini oluşturmaktadır. Bu grafik gereken tüm bilgileri taşıdığı için buna AAS sisteminde “Anlam Grafiği” adı verilmiştir.

Şekil 3’deki ekran görüntüsünün sağ alt köşesinde, aynı şekilde görülen cümlelerden otomatik olarak çıkarılan nesne ilişkilerinin anlam grafiği görülmektedir. Nesneler arasındaki ilişkilerin belirlenmesi için gerçekleştirilen analizlerin nasıl yapıldığı konusuna burada girilmeyecektir. Ancak bu analizlerin “algısal semantik” (cognitive semantic) yaklaşımı ile çıkarıldığını belirtmek yerindedir.

1. **Sonuç**

Dünyada geliştirilen semantik sistemlerin hepsinde de semantik dil çözümleme yaklaşımı kullanılmaktadır. Yapılan bu çalışmalarla ilgili yayınlarda, konusu “semantik” terimini içeren makaleler okunduğunda, çalışmaların bir taraftan da çıkmaz bunalımlara doğru yöneldiği veya kavram ve yaklaşım karmaşalarının çoğaldığını gözlemlemek mümkündür. Bu nedenle, örneğin semantik-web, semantik veritabanı vb. terimlerle ifade edilen yeni çalışma alanlarında bile, bir çok çalışma semantik felsefeden uzaklaşmakta, belki zorunluluktan belki de sorunların sadece pragmatik bir anlayışla ele alınmasından kaynaklandığını düşündüren çelişkiler göze çarpmaktadır. Bu bağlamda; dil analizlerini matematiksel mantığın formel yapılarıyla sarmalama eğilimleri artmış ve probleme yaklaşımlar semantik analizden çok matematiksel analiz problemleri şekline dönüştürülmüş veya dönüştürülmektedir. Formel kurallarla ifade edilen anlam ilişkilerini en etkin şekilde ifade eden sistemlerden birisi Prolog’dur (Rowe, 1988, Stobo, 2005). Diğer bütün formel analiz yöntemleri de Prolog’un daha çok Frege’ye dayanan yaklaşımları üzerine kurulmaktadır. Ancak bu ifadeler bu haliyle ele alındığında, ilk baştaki tanıma göre “düşük düzeyli dil” olmaktan öteye gidememektedir. Bu anlamda, Formel semantik yaklaşımına bir seçenek olarak Algısal Semantik veya Kognitif Semantik yaklaşımları üzerinde çalışılmalıdır (Evans and Green, 2006). Algısal semantik yaklaşımları, makinelerin fiziksel gerçek dünyadaki olgularla anlamsal ilişkiler kurmasını kolaylaştıracakmış gibi görünmektedir. Eğer semantik anlam analizleri, algısal içeriğin doğasına dayanarak gerçekleştirilebilirse, bu durum yeni birçok gelişmeyi de başlatacaktır. Algısal semantik yaklaşımlarının katı matematiksel formel yapılar yerine, doğal fizik yasalarıyla ifade edilmesi konularında ümit verici görünmektedir.

Ülkemizde, bilgi teknolojilerinin kullanım alanları artmaktadır. Ancak bu artış felsefeden çok uzak ve bilginin önemi ve anlamını yadsıyan bir kültürel anlayışla birlikte gerçekleşiyormuş gibi görünmektedir. Felsefeden uzak olarak bilgi teknolojilerini kullanmaya çalışmak, onu bu şekilde kullanan toplumları düşünme eylemlerinden ve aklın değerini anlamaktan uzaklaştırır. Ülkemizde bilgi teknolojilerinin, insanın ürettiği bilgilerin sıcaklığının hissedilerek kullanılmasını sağlamak için bütün meslek alanlarında, bilgi teknolojileri konusundaki sorunların çözümü için semantik felsefe, matematiksel mantık ve dil mantığı temellerine dayanan çözümler üretmek üzere çalışmalar başlatılmalıdır.

**Kaynaklar**

Akın, A. A. and M. D. Akın. (2007). *Zemberek an open source nlp framework for turkic languages.* http:// code.google.com/p/zemberek/ [Erişim 28 Nisan 2013].

De Rijk, L. M. (2002). Aristotle;Semantics and Ontology. VOLUME II: The Metaphysics, Semantics, in Aristotle's Strategy of Argument Netherlands: Brill Leiden.

Evans, V. and M. Green (2006). Cognitive Linguistics an Introduction. Edinburgh: Edinburgh University Press

Grimes, F. (2002). Microsoft .NET for Programmers. USA: Manning Publications Co.

Guise, J. (2013). Manisa Turkish - Learn Turkish free - online. <http://www.turkishlanguage.co.uk/> [Erişim 28 Nisan 2013].

Hançerlioğlu, O. (2010). Felsefe Sözlüğü. İstanbul: Remzi Kitabevi A.Ş.

Kemal, O., et al. (2004). A Preprocessor for Turkish Text Analysis. Lecture Notes in Computer Science 3280, Proce*edings ISCIS'04*.

Laycock, H. (2006). Words without Objects:Semantics, Ontology, and Logic for Non-Singularity. Oxford: Clarendon Press.

Luger, G. F. and W. A. Stubblefield (1998). Artificial Intelligence: Structures and Strategies for Complex problem solving. Third Edition ed. USA: Addison Wesley Logman Inc.

Macbeth, D. (2005). Frege’s Logic. Cambridge, Massachusetts, and London, England: Harvard University Press.

Microsoft.com. (2013). Microsoft SQL Server. <http://www.microsoft.com/en-us/sqlserver/default.aspx> [Erişim 24 Ocak 2013].

Proxem.com. (2013). Proxem: from text to data. https://[www.proxem.com/](http://www.proxem.com/) [Erişim 24 Ocak 2013].

Rowe, N. C. (1988). Artificial Intelligence through Prolog by Artificial Intelligence through Prolog: Prentice-Hall.

Say, A. C. C., et al. (2004). A Natural Language Processing Infrastructure for Turkish. COLING 2004, Geneva, Switzerland.

Stobo, J. (2005). Problem solving with prolog. Computer systems. Programming languages:Prolog. London, Taylor & Francis e-Library.

Stubblebine, T. (2007). Regular Expression: Pocket Reference. 2nd Edition ed. Canada: O'Reilly Media, Inc.

TDK. (2013). Yazım Kuralları. <http://www.tdk.gov.tr/index.php?option=com_content&view=category&id=50> [Erişim 28 Nisan 2013].

Torunoglu, D., et al. (2011). Analysis of preprocessing methods on classification of Turkish texts. Innovations in Intelligent Systems and Applications (INISTA), 2011 International Symposium.

Turtschi, A., et al. (2002). C# . NET Web Developer’ s Guide: Develop and Deliver Enterprise-Critical Desktop and Web Applications with C# .NET  USA: Syngress Publishing, Inc.

W3Schools.com. XQuery Tutorial. <http://www.w3schools.com/xquery/default.asp> [Erişim 24 Ocak 2013].

[www.loc.gov](http://www.loc.gov). (2013). SRU Search Retrieval via URL: SRU the Protocol, CQL Query Language, ZeeRex Service Description. <http://www.loc.gov/standards/sru/index.html> [Erişim 24. Ocak 2013].

Zelal, G. and O. Kemal (1994). Parsing Turkish using the lexical functional grammar formalism. Proceedings of the 15th conference on Computational linguistics.