T.C.
TRAKYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİR KELİME ANLAMI BELİRGİNLEŞTİRME MODÜLÜ GELİŞTİRİLMESİ

Özlem AYDIN

Doktora Tezi

Bilgisayar Mühendisliği Anabilim Dalı
Danışman: Doç. Dr. Yılmaz KILIÇASLAN

2011
EDİRNE

T.C.
TRAKYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Bir Kelime Anlamı Belirginleştirme Modülü Geliştirilmesi

Özlem AYDIN

Doktora Tezi

Bilgisayar Mühendisliği Anabilim Dalı

Bu tez 28 / 01 / 2011 tarihinde aşağıdaki jüri tarafından kabul edilmiştir.

Doç.Dr. Yılmaz KILIÇASLAN

Danışman
Jüri Başkanı

Doç. Dr. Hasan Hüseyin BALIK

Üye

Doç. Dr. Tahir ALTINBALIK

Üye

Yrd. Doç. Dr. Erdem UÇAR

Üye

Yrd. Doç. Dr. Aydın CARUS

Üye

Doktora Tezi

Trakya Üniversitesi Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Bölümü

ÖZET

Kelime Anlamı Belirginleştirme (KAB), doğal dil işleme uygulamalarında gereksinim duyulan önemli bir işlemdir ve birden fazla anlamı olan bir kelimenin bulunduğu bağlamdaki anlamının belirlenmesi olarak tanımlanır. Bu tezin amacı, bazı KAB yaklaşımlarının Türkçe metinler üzerinde uygulanmasıyla elde edilen başarım sonuçlarını raporlamak ve bu sonuçlar üzerinden yapılan değerlendirmeleri sunmaktır.

Çalışmada öncelikle eşdizimlilik bilgisini kullanarak gerçekleştirilen denetimsiz derlem tabanlı bir KAB uygulaması anlatılmış ve sonuçları değerlendirilmiştir. Ardından, bu uygulamadan elde edilen başarım sonuçlarının yetersizliğini göz önüne alarak geliştirilen alternatif bir KAB uygulaması ayrıntılarıyla anlatılmıştır. Bu uygulama klasik makine öğrenme yaklaşımlarının artalan bilgisini kullanmadaki ve tümdengelim çıkarım yapabilmedeki yetersizliğini gideren bir yöntem olan Tümevarımlı Mantık Programlamaya (TMP) dayanmaktadır. Bu doğrultuda, TMP konusu ayrıntılı olarak incelenmiş ve KAB'a uygulanabilirliği Türkçe veriler üzerinden elde edilen deneysel sonuçlarla gösterilmiştir.

Anahtar Kelimeler: Kelime Anlamı Belirginleştirme, Tümevarımlı Mantık Programlama, Makine Öğrenmesi, Mantık Programlama

Doctorate Thesis

Trakya University Graduate School of
Natural and Applied Sciences
Department of Computer Engineering

ABSTRACT

Word Sense Disambiguation (WSD) is one of the important processes needed for natural language processing applications and is defined as determining the sense of a multi-sense word in a given context. The aim of this thesis is to report on the performance results achieved by applying some WSD approaches to Turkish texts and present the evaluations made using results.

In the study, firstly an unsupervised corpus based WSD application developed with collocation knowledge is presented and then its results are evaluated. Afterwards, an alternative WSD application developed considering the insufficiency of in the performance results achieved in that application is accounted for in detail. This latter application rests on Inductive Logic Programming (ILP), which is a method that circumvents the incapability of traditional machine learning approaches in employing background knowledge and making deductive inferences. To this effect, the topic of ILP is given a detailed explanation and its applicability to WSD is demonstrated with empirical results obtained using Turkish data.

Key Words: Word Sense Disambiguation, Inductive Logic Programming, Machine Learning, Logic Programming.

TEŞEKKÜR

Tez çalışmamın gerçekleşmesi sürecinde yardımlarından dolayı tez danışmanı hocam Sayın Doç. Dr. Yılmaz KILIÇASLAN'a teşekkür ederim.

Doktora tezi savunma jürimde yer alan, bilgi ve tecrübelerinden yararlandığım sayın hocalarım Doç. Dr. Hasan Hüseyin BALIK'a, Doç. Dr. Tahir ALTINBALIK'a, Yrd. Doç. Dr. Erdem UÇAR'a ve Yrd. Doç. Dr. Aydın CARUS'a teşekkür ederim.

Çalışmam sırasındaki yardımları için Hüseyin BAŞARICI'ya, Arş. Gör. Emir ÖZTÜRK'e ve ortak çalışma yaptığımız arkadaşım Mehmet Ali Aksoy TÜYSÜZ'e teşekkür ederim.

Trakya Üniversitesi Bilgisayar Mühendisliği Bölümü tüm çalışma arkadaşlarıma tez boyunca verdikleri moral desteği ve yardımlarından ötürü teşekkür ederim.

Tez çalışmam boyunca güvenleriyle her zaman yanımda olan aileme çok teşekkür ederim.

İÇİNDEKİLER

ŞEKİL LİSTESİ

TABLO LİSTESİ

KISALTMALAR LİSTESİ

ALEPH	A Learning Engine for Proposing Hypotheses (Hipotezlerin Tahmini için bir Öğrenme Aracı)
AODE	Aggregating One Dependence Estimators (Tek Bağımlı Tahminleyici Toplama)
ARPA	Advanced Research Projects Agency (İleri Proje Araştırma Ajansı)
AW	All Words (Bütün Kelimeler)
BNC	British National Corpus (İngiliz Ulusal Derlemi)
CART	Classification and Regression Trees (Sınıflandırma ve Regrasyon Ağaçları)
CES	Corpus Encoding Standard (Derlem Kodlama Standardı)
DARPA	Defense Advanced Research Projects Agency (Gelişmiş Savunma Projeleri Araştırma Ajansı)
DN	Doğru Negatif
DP	Doğru Pozitif
FOIL	First-Order Inductive Learner (Birinci Dereceden Tümevarımlı Öğrenici)
IC	Information Content (Bilgi İçeriği)
ID3	Induction Decision Tree (Tümevarım Karar Ağacı)
ILP	Inductive Logic Programming (Tümevarımlı Mantık Programlama)
ISA	is-a
KAB	Kelime Anlamı Belirginleştirme
LDOCE	Longman Dictionary of Contemporary English (Longman Çağdaş İngilizce Sözlük)
LEXAS	LEXical Ambiguity-resolving System (Sözlüksel Belirsizlik Çözücü Sistem)
LOB	Lancester-Oslo-Bergen
LS	Lexical Sample (Sözlüksel Örnek)
MIS	Model Inference System (Model Çıkarım Sistemi)
MPD	Merry Webster's Pocket Dictionary (Merry Webster'in Cep Sözlüğü)
MRD	Machine Readable Dictionary **(**Makinece Okunabilir Sözlük**)**

MUC	Message Understanding Conferences (Mesaj Anlama Konferansları)
OALD	Oxford Advanced Learner's Dictionary
OC1	Oblique Classifier 1
ODTÜ-BAP	Orta Doğu Teknik Üniversitesi – Bilimsel Araştırma Projeleri
P	Precision (Duyarlılık)
POS	Part-of-speech
R	Recall (Geriçağırım)
TDK	Türk Dil Kurumu
TL	Translation Memory (Çeviri Hafıza)
TLST	Turkish Lexical Sample Task (Türkçe Sözlüksel Örnek Görevi)
TMP	Tümevarımlı Mantık Programlama
TREC	The Text Retrieval Conference (Metin Erişimi Konferansı)
W7	Webster's Seventh New Collegiate Dictionary
WSD	Word Sense Disambiguation (Kelime Anlamı Belirginleştirme)
XML	Extensible Markup Language (Genişletilebilir İşaretleme Dili)
YN	Yanlış Negatif
YP	Yanlış Pozitif

1. GİRİŞ

Bu tezde amacımız, anlamı belirsiz olan kelimelerin anlamını belirlemek amacıyla Türkçe için yapılan bir Kelime Anlamı Beliginleştirme (KAB) işleminde kullanılabilecek yaklaşımlar içinde iyi sonuç veren yaklaşımı belirlemektir. Bu amaç doğrultusunda öncelikle denetimsiz derlem tabanlı bir uygulama geliştirilmiştir. Denetimsiz yaklaşımlarda kelime anlamlarının etiketlenmediği derlemler kullanılmaktadır. Bu uygulamada bu tür bir derlemden çıkarılan cümleler üzerinde hiçbir sözdizim, anlambilim vb. özelliğe bakılmaksızın eşdizimlilik aranmıştır. Bu uygulamada elde edilen başarımın yetersizliği nedeni ile çalışmanın devamında farklı bir yaklaşım olarak denetimli derlem tabanlı bir KAB uygulaması geliştirilmiştir. Denetimli yaklaşımlarda kelime anlamlarının etiketlendiği derlemler kullanılmaktadır. Uygulamamızda Türkçe için hazırlanmış bir derlem olan Türkçe Sözlüksel Örnek Görevi'ni kullandık. Denetimli makine öğrenmesi tekniği olarak da Tümevarımlı Mantık Programlama (TMP) kullanımı tercih edilmiştir.

KAB işlemi, bir kelimenin taşıdığı anlamın verilen bir bağlamda belirlenmesi olarak tanımlanmaktadır. KAB, makine öğrenmesinin konusu olan sınıflandırma problemi olarak ele alınabildiğinden, makine öğrenmesi teknikleri bu alanda sıklıkla kullanılmaktadır. Günümüze kadar yapılan çalışmalarda makine öğrenmesi teknikleri ile diğer yöntemlere göre daha başarılı sonuçlar elde edildiği görülmüştür. Denetimli derlem tabanlı KAB uygulamamızda, makine öğrenmesi ve mantık programlamanın kesişimi olarak tanımlanan ve bu iki alandaki tüm teknikleri kullanan bir yöntem olan TMP'yi tercih etmemizin bazı nedenleri vardır. TMP'nin makine öğrenmesini içeriyor olması bu yöntemi tercih etmemizdeki nedenlerden biridir. TMP yöntemini tercih etmemizdeki diğer bir neden TMP'nin artalan bilgisini etkin kullanabilmesidir. Klasik makine öğrenme yaklaşımları tek başına artalan bilgisini kullanmada ve tümdengelimli çıkarım yapabilmede yetersiz kalmaktadır. Makine öğrenmesindeki tümevarımlı çıkarım ve mantık programlamadaki tümdengelimli çıkarım mekanizmaları TMP'de bir araya gelmektedir. TMP'de bu mekanizmaların birlikte kullanılması, artalan bilgisinde üstü kapalı olarak bulunan önemli miktardaki bilginin çıkarımı sağlayarak başarımı olumlu yönde etkilemektedir.

KAB işlemine kelime anlamının belirsiz olduğu durumlarda ihtiyaç duyulmaktadır. Doğal dil işleme çalışmalarında doğal dilin esnekliği nedeniyle farklı belirsizlik durumları ile karşılaşılır. Belirsizlik doğal dil işlemenin çoğu uygulamasında (makine çevirisi, bilgi erişimi, dilbilgisi çözümlemesi, vb.) çözülmesi gereken önemli problemlerden biridir. Farklı belirsizlik durumları bulunmaktadır. Bu durumlar genel olarak sözdizimsel (syntactic) ve sözlüksel (lexical) olarak ortaya çıkmaktadır. Sözdizimsel belirsizlik, bir cümle sözdizimi nedeniyle farklı şekillerde yorumlanabildiği durumda ortaya çıkar. Örneğin (1.1) cümlesinde çok anlamlı bir kelime bulunmadığı halde, cümlenin ifade etmek istediği durum farklı şekillerde yorumlanabilir. Şöyle ki;

Babası Ahmet'ten kitabını getirmesini istedi. (1.1)

cümlesinde istenen kitabın Ahmet'e mi yoksa babasına mı ait olduğu belli değildir. Kelime anlamı belirsizliği, diğer bir deyişle sözlüksel belirsizlik (lexical ambiguity) bir kelimenin birden fazla anlamı olması durumunda ortaya çıkan bir durumdur. Sözlüksel belirsizliğin genel olarak iki türü vardır: eş seslilik (homonymy) ve çok anlamlılık (polysemy). Eşseslilik, sözlüksel bir birimin rastlantısal olarak iki ya da daha fazla ayrı ve birbirinden bağımsız anlam taşıması olarak tanımlanır. Örneğin, http://guzelturkcemiz.org internet sitesinden alınan (1.2)'deki dörtlükte "yüz" kelimesinin üç farklı anlamı ile kullanımı bir arada görülmektedir. İlk satırda "rakam" olan anlamıyla, ikinci satırda "surat" anlamıyla ve son satırda ise "yüzmek fiili" olan anlamıyla kullanılmıştır.

Bahçede var *yüz* güzel
Endam güzel *yüz* güzel
Uzaklara açılma
Kıyılarda *yüz* güzel (1.2)

Çok anlamlılık ise, tek bir sözlüksel birimin pek çok farklı ancak birbirleriyle bağlantılı anlamları olmasıdır. Çok anlamlılıkta kelimenin asıl anlamından kopmadan yeni bir anlamı karşılaması söz konusudur ve sözlüklerde bir madde başı altında kelimelerin birden fazla anlamı sıralanır. "Ay" kelimesi ünlem olarak kullanılan "ay!" kelimesiyle eş sesli olmakla birlikte, kendi içinde çok anlamlıdır ve dünyanın uydusu olan gök cismini ve yılın aralıklara bölünmüş zaman dilimlerini karşılamaktadır. "Ay"

kelimesinin bu iki anlamına bakıldığında birincisinin somut olan bir kavrama karşılık geldiği, ikincisinde ise soyut olan zaman kavramını karşıladığı görülmektedir. “Ay” kelimesinin bu iki anlamı arasındaki ilişki, ayın dünya etrafında bir kez dönmesinin bu zaman dilimlerini oluşturmasından kaynaklanmaktadır (Kurudayıoğlu ve Karadağ, 2005).

İnsanlar için bir kelimenin verilen bir bağlamda hangi anlamı ile kullanıldığını belirlemek kolaydır. Çünkü bu anlama işlemini gerçekleştiren bilişsel bir sisteme sahiptir. Ama bu belirleme işlemi bilgisayarlar için kolay olmamaktadır. Doğal dil işlemede sözdizimsel belirsizlik, kelime türü etiketleyicilerle yüksek doğrulukta çözülebilmektedir. Kelime anlamı belirsizliğinin çözümü ise KAB işlemiyle gerçekleştirilir ve bu işlemin sözdizimsel belirsizliğin çözümünden daha güç olduğu kanıtlanmıştır.

KAB problemi, *AI-complete* olarak tanımlanmaktadır. Yani ilk olarak yapay zekadaki bütün zor problemler çözüldükten sonra çözülebilecek bir problemdir (Ide ve Veronis, 1998). Dolayısıyla, zor bir problemdir. Bu sebeple başarımda etkili olacak yöntemin kullanımı, KAB probleminin etkin çözümünde çok önemli olmaktadır.

Bu tez çalışmasında, ilk bölümde, KAB konusu ayrıntılı olarak anlatılmıştır. KAB için işlem adımları, uygulandığı alanlar, faydalı bilgi türleri, kullanılan kaynaklar ve KAB’da karşılaşılan problemler anlatılmıştır. Ayrıca, KAB yaklaşımlarına ve bu yaklaşımlardan bilgi tabanlı KAB’a ayrıntılı olarak değinilmiştir.

İkinci bölümde, denetimsiz derlem tabanlı KAB yaklaşımı anlatılmış ve eşdizimlilik bilgisini kullanarak Türkçe için hazırlanmış denetimsiz derlem tabanlı bir KAB uygulamasından bahsedilmiştir.

Üçüncü bölümde, öncelikle denetimli derlem tabanlı KAB yaklaşımları anlatılmıştır. Devamında, tezde amacımız olan denetimli derlem tabanlı KAB uygulamasını geliştirmek için bir makine öğrenmesi yöntemi olan TMP incelenmiştir. Bu inceleme, tümevarımlı mantık programlamanın tarihsel gelişimini, TMP’deki temel kavramları ve bu alanda kullanılan çeşitli teknikleri içermektedir. Bu bölümün sonunda

ise tezin amacı doğrultusunda geliştirilen denetimli derlem tabanlı KAB uygulaması anlatılmıştır.

Dördüncü bölümde, öncelikle KAB için performans değerlendirme ölçütlerine değinilmiş ve KAB sistemlerinin değerlendirilmesi hakkında ayrıntılı bilgi verilmiştir. Bölüm sonunda elde ettiğimiz sonuçlar verilmiş ve bu sonuçların değerlendirmesi yapılmıştır.

Son bölüm olan beşinci bölümde ise yapılan çalışma ile ilgili son değerlendirmelerde bulunulmuştur.

2. KELİME ANLAMI BELİRGİNLEŞTİRME

2.1 Kelime Anlamı Belirginleştirme İşleminin Adımları

KAB, birden fazla anlamı olan bir kelimenin bulunduğu bağlamda hangi anlamıyla kullanıldığının belirlenmesi işlemidir. (Ide ve Veronis, 1998)'e göre bu işlem iki temel adıma ayrıştırılabilir:

1. İlgili kelimenin bulunduğu metin veya söylem göz önünde bulundurularak tüm farklı anlamlarının belirlenmesi.

2. Kelimeye uygun anlamın atanması.

Bu adımları, aşağıdaki örnek bağlam ve bu bağlam içinden seçtiğimiz çok anlamlı bir kelime üzerinden açıklayalım:

> "Derviş Beyin kalın kara kaşlarının altındaki gözleri bir yangın gibiydi. Elmacik kemikleri çıkık, gözleri biraz çekikti. Çenesinin çukuru derin, gölgeli ve çenesi güçlüydü." (Yaşar Kemal; Demirciler Çarşısı Cinayeti; s.1) (2.1)

Verilen bu bağlam içinde birden fazla çok anlamlı kelime bulunmaktadır. Bu kelimelerden bazıları; "kara", "göz", "yüz", "ak" kelimeleridir. Bunlar içinden KAB işlemini uygulayacağımız kelime olarak "göz"'ü seçelim. KAB işleminin ilk adımı olan istenen kelimenin anlamlarını belirleme için yapılması gerekenler şunlardır:

1. Herhangi bir sözlükten ilgili kelimenin bir anlam listesi alınmalıdır. "Göz" kelimesinin Türk Dil Kurumu (TDK) Türkçe Sözlük'ünden alınan anlamları Tablo 2.1'de verilmiştir. Bu kelimenin birçok anlamı olduğu için burada en sık kullanılan üç anlamının verilmesi tercih edilmiştir.

Anlam no	**Anlamı**
1. anlam	Görme organı.
2. anlam	Bazı deyimlerde görme, bakma.
3. anlam	Oda.

Tablo 2.1 "Göz" kelimesinin anlamları

2. Kelimenin kelime türü bilgisi ile eş ve/veya yakın anlamlı olduğu kelimelerin bir listesine gerek duyulur. Kelime türü bilgisinin alınabileceği birçok teknik sözlük vardır. "Göz" kelimesinin Tablo 2.1'de verilen anlamları için (Kılıçaslan vd., 2010) çalışmasında kulanılan sözlükten aldığımız kelime türü bilgisi *isim* olmuştur. TDK ve Dokuz Eylül Üniversitesi Dil Bilimi Bölümü'nün iş birliğiyle hazırlanmış olan eş ve/veya yakın anlamlı kelimeler sözlüğünden "göz" kelimesi için elde edilen bilgiler Tablo 2.2'de verilmiştir.

göz kelimesinin eş ve/veya yakın anlamları
hane
bölüm
nazar
çekmece
delik
kaynak
görüş
bakış
oda

Tablo 2.2 "Göz" kelimesi için eş ve/veya yakın anlamlı kelimeler

3. Son olarak da herhangi bir uygulamada bir ara işlem olarak KAB'a ihtiyaç duyulduğu durumlarda gerekli olabilecek bilgiler vardır. Örneğin, bir makine çevirisi uygulamasında KAB işlemi yapılacaksa diğer dillere çevirileri içeren bir çeviri

sözlüğündeki bir girişe gerek duyulmaktadır. Bir Türkçe–İngilizce çeviri yapan sistemde KAB uygulamak istediğimizi varsayalım. Bu durumda "göz" için çeviri karşılığı olabilecek kelimeler bulunmalıdır. Bunun için Zargan İngilizce Sözlük'ten aldığımız İngilizce karşılıklar Tablo 2.3'de verilmiştir.

Anlam no	**Anlamı**	**İngilizce karşılığı**
1. anlam	Görme organı.	eye
2. anlam	Bazı deyimlerde görme, bakma.	sight
3. anlam	Oda.	room

Tablo 2.3 "Göz" kelimesinin anlamlarına karşılık gelen İngilizce kelimeler

İkinci adımda aşağıda verilen iki büyük bilgi kaynağına dayanılarak kelimelere anlamlarının atanması gerçekleşir.

- Belirginleştirilecek kelimenin bulunduğu bağlam. Örnek olarak verilen (2.1)'deki metin, "göz" kelimesinin içinde bulunduğu bağlamdır. Bu bağlamda, "göz" kelimesi 1. anlamı ile kullanılmıştır. Farklı bir bağlam örneğine bakalım:

  "Köyün hali belli, bunu sen de biliyorsun, ben de! Bu yıl olmazsa gelecek yıl alıp başımızı gidelim, Ankara'nın bir köşesine sokulalım, bir göz ev uydursak bize yeter." (Fakir Baykurt; Kaplumbağalar; s. 87) (2.2)

  Bu bağlamda "göz" kelimesi "oda" anlamına gelen 3. anlamı ile kullanılmıştır.

- Harici bilgi kaynakları. Elle oluşturulmuş bilgi kaynaklarının yanısıra sözlüksel ve ansiklopedik bilgilerin de bulunduğu kaynakları içerir.

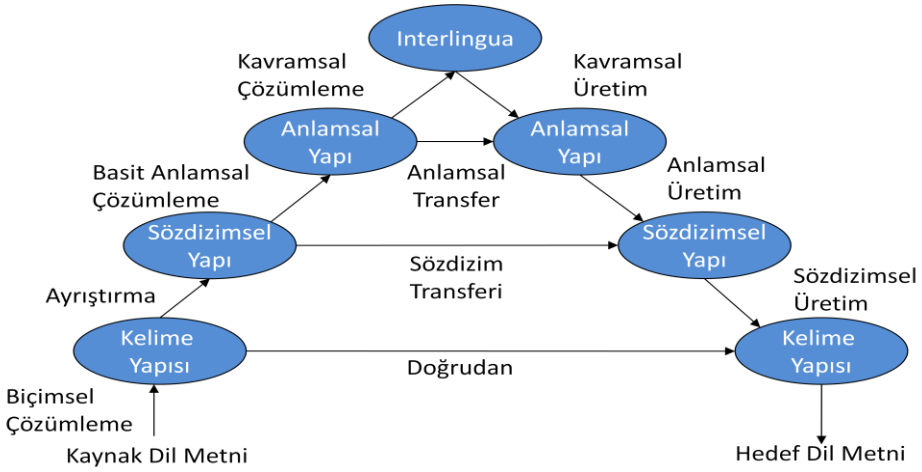
2.2 Kelime Anlamı Belirginleştirmenin Uygulandığı Alanlar

KAB birçok alana uygulanmaktadır. KAB işlemi mesaj anlama, insan-makine iletişimi gibi amacın anlama olduğu uygulamalarda kesinlikle gereklidir. Ayrıca amacın anlama olmadığı uygulamalarda da bir ara işlem olarak ihtiyaç duyulmaktadır (Ide ve Veronis, 1998). Bu uygulama alanlarından bazıları şunlardır: Makine Çevirisi (Machine Translation), Bilgi Erişimi (Information Retrieval), Ses İşleme (Speech Processing), Metin İşleme (Text Processing), İçerik ve Tematik Analizi (Content and Thematic Analysis) ve Dilbilgisi Çözümlemesi (Gramatical Analysis). KAB'ın son yıllarda Anlamsal Ağ (Semantic Web) alanındaki önemi de artmıştır. Şimdi, bu alanların her birini ayrıntılı olarak ele alalım.

2.2.1 Makine çevirisi

Çeviri, anlamın yanında biçimin de korunarak, kaynak dildeki bir ifadenin hedef dildeki en yakın doğal karşılığının üretilmesi işidir (Nida, 1975). Çevirinin başını ve sonunu oluşturan iki temel kavram, kaynak dil ve hedef dildir. Kaynak dil, hedef dile çevirisi yapılmak istenen ifadenin kodlandığı dildir. Çeviri, kaynak dildeki bir ifadenin kodladığı anlamın bozulmadan hedef dilde yeniden kodlanabilmesini gerektirir. Çeviri işinin bir kısmının veya tamamının bilgisayar kullanımı yoluyla otomatikleştirilerek gerçeklenmesi işine Makine Çevirisi denmektedir (Jurafsky ve Martin, 2009).

Kaynak dilden hedef dile çeviri yapılması sürecinin şematik olarak gösterimi Şekil 2.1'de verilen Vauquois Üçgeni ile yapılır (Vauquois, 1968). Üçgen üzerinde, kaynak dilden hedef dile geçiş yapılabilecek aşamalar gösterilir.



Şekil 2.1 Vauquois üçgeni.

Çeviri için en alt düzey olan *doğrudan yaklaşımda*, kaynak dil ifadesi içerisinde kelime kelime ilerlenir. Her kelimeyi çevirmek için ikidilli (bilingual) bir sözlüğe başvurulur. Bu sözlükteki her kelime girişi, kaynak dildeki kelimeyi hedef dile dönüştüren küçük birer program gibi düşünülebilir. Bu yaklaşımda herhangi bir analiz yapılmaz. Doğrudan çevirinin bir üst aşaması olan *transfer yaklaşımında*, girdi metni ayrıştırılırak (parse) bir yapı elde edilir, ardından bu yapıdan hedef dil cümlesi üretilir. Analizde en üst aşama olan *Interlingua yaklaşımında*, kaynak dil analizle soyut bir anlamsal gösterime dönüştürülür ve hedef dil ifadesi bu gösterimden sentezlenir.

Kelime düzeyindeki farklar, çeviride önemli sorunlara yol açabilmektedir. Bu sebeple, makine çevirisinde KAB işlemine gerek duyulur. Özellikle, eşsesli ve çok anlamlı kelimelerin çevirisi sorunlar yaratmaktadır. Eşsesli kelimelerin sorun olmasının sebebi, kaynak dildeki bir kelimenin, hedef dilde aynı türde birden fazla kelimeye karşılık gelebilmesidir.

İngilizce → bank

Türkçe → banka (isim), nehir kıyısı (isim), bank (isim)

Çok anlamlılık ise, kaynak dildeki bir kelime birden çok anlamda kullanılırken hedef dilde her bir anlam için farklı kelimenin olması durumunda zorluk yaratır. Örneğin, İngilizce “know” fiili bir olayı ya da durumu bilmek anlamında

kullanılabileceği gibi, bir insanı tanımak anlamında da kullanılabilir. Oysa Fransızca'da bu iki farklı durum için, iki farklı fiil kullanılır.

İngilizce → I know he just bought a book. (2.3)

Fransızca → Je sais qu'il vient d'acheter un livre. (2.4)

İngilizce → I know John. (2.5)

Fransızca → Je connais Jean. (2.6)

Fransızca'da bir olay ya da durumun bilinmesine karşılık "savoir" (şimdiki zaman formu: sais) fiili kullanılırken, birinin tanınmasına ilişkin durumlar için "connaitre" (şimdiki zaman formu: connais) fiili kullanılır.

2.2.2 Bilgi erişimi

Belirsizlik, bilgi erişiminde kullanılan sorgularda da çözülmesi gereken önemli bir problemdir. Çok anlamlı bir kelime bir arama motoruna sorgu olarak girildiğinde bu kelimenin sadece istenen anlamı için değil, bütün anlamları için birçok doküman dönecektir. Mevcut bilgi erişim sistemleri genellikle KAB işlemi yapmadan istenen anlamdaki dokümanları elde etmek için sorguda bağlamı genişletme yoluna gitmektedir. Örnek olarak "yüz" kelimesini ele alalım. "Yüz" kelimesi için TDK Türkçe Sözlük'te bulunan iki anlamı Tablo 2.4'deki gibidir:

Anlam no	**Anlamı**
1. anlam	Doksan dokuzdan sonra gelen sayının adı.
2. anlam	Başta, alın, göz, burun, ağız, yanak ve çenenin bulunduğu ön bölüm, sima, çehre, surat.

Tablo 2.4 "Yüz" kelimesinin anlamları

“Yüz” kelimesinin sayı olan anlamını içeren dokümanları elde etmek istediğimizi varsayalım. Bu durumda sorguyu genişletme yoluna gidip arama motoruna “yüz tane” kelime çiftini girebiliriz. Tablo 2.5’de google arama motorunda “yüz” ve “yüz tane” sorgularının girilmesi sonucu elde edilen duyarlılık sonuçları verilmiştir.

Sorgu	**P10**	
	1. anlam	**2. anlam**
yüz	0.2	0.8
yüz tane	1	0

Tablo 2.5 “Yüz” ve “yüz tane” kelimelerinin sorgu sonucu duyarlılık değerleri

Bilgi erişimi sistemlerinin değerlendirmesinde kullanılan iki önemli ölçüt bulunmaktadır. Duyarlılık (precision) değeri, getirilen bilgideki doğru sonuçların, getirilen bilginin tamamına oranı olarak hesaplanır. Geriçağırım (recall) değeri de getirilen doğru sonuçların, getirilmesi gereken doğru sonuçlara oranı ile hesaplanır. Bilgi erişimi çalışmalarında geriçağırım değerinin hesaplanması zor olmaktadır. Tablo 2.5’de verdiğimiz P10 değeri arama motorundan dönen 10 sonuç için elde edilen duyarlılık değeridir. Tablodan görüldüğü üzere “yüz” kelimesinin sayı anlamı olan 1. anlamını elde etmek amacıyla “yüz” kelimesi yerine “yüz tane” sorgusunun arama motoruna girilmesi duyarlılık değerini arttırmıştır. Bilgi erişimi sistemlerinde KAB işleminin yapılması ile çok anlamlı kelimelerde o kelimenin istenmeyen anlamının elenmesi sağlanarak arama sonucunun kalitesi arttırılabilir.

2.2.3 Ses işleme

Ses işlemede doğru seslendirme için KAB işlemine gerek duyulabilir. Aşağıda verilen cümlelerdeki “kar” kelimesinin anlamları ve okunuşları birbirinden farklıdır.

İki gündür sürekli yağan *kar* ulaşımı etkiledi. (2.7)

Bunlar kısa sürede *kâr* sağlayan yatırımlardır. (2.8)

Ayrıca kelime vurgusu da doğru seslendirmede önemlidir. Bir kelimede yanlış hecede vurgu yapılırsa anlam karışıklığı ortaya çıkmaktadır. Örneğin, (2.9) ve (2.10) cümlelerinde "kesin" kelimesinin kullanıldığı anlamına göre vurgusu da değişmektedir. (2.9) cümlesinde "kesin" kelimesinin türü isimdir ve vurgu ikinci hecede yapılmaktadır. Kelime türünün fiil olduğu (2.10) cümlesinde ise vurgu ilk hecede bulunmaktadır.

Henüz *kesin* kararını vermedi. (2.9)

Bu ekmeği ortadan ikiye *kesin*. (2.10)

2.2.4 Metin işleme

Hatalı yazılan kelimelerin düzeltilmesinde ve büyük küçük harf değiştirmede KAB gerekli olabilmektedir. Örneğin, bazı kelimeler hem özel isim hem de cins isim olabilmektedir. Aşağıda verilen cümlelerde "deniz" kelimesi cins isim olarak kullanıldığında yazımı küçük harfle başlarken, özel isim olduğunda yazımı büyük harfle başlamaktadır. (2.12)'deki cümlede "Deniz" kelimesinin hatalı olarak küçük harfle başladığını düşünelim. Bu durumda KAB işlemi yapıldığı takdirde bu kelimenin özel isim olduğu tespiti yapılarak küçük harf büyük harfe dönüştürülmesi ile "deniz" kelimesi "Deniz" olarak düzeltilir.

Tatil deyince çoğumuzun aklına; *deniz*, güneş, kum, yüzmek gelir. (2.11)

Bugün *Deniz* mutsuz görünüyor. (2.12)

2.2.5 İçerik ve tematik analizi

İçerik analizi, toplanan verilerin önce kavramsallaştırılması daha sonra da ortaya çıkan kavramlara göre mantıklı bir biçimde düzenlenmesi ve buna göre veriyi açıklayan

temanın saptanması işidir. Yaygın bir yöntem olarak önceden belirlenen kelime kategorilerinin ve tema hakkında belirleyici özelliğe sahip kelimelerin metindeki dağılımının analizi kullanılmaktadır. Dağılımı incelenecek kelimenin doğru anlamının kullanılması sonuçları etkileyecektir. Bu aşamada KAB önemlidir.

2.2.6 Dilbilgisi çözümlemesi

Biçimbilimsel ve sözdizimsel çözümlemede bazı durumlarda kelimelerin doğru anlamlarının bilinmesi fayda sağlayabilir. Örneğin,

Hemen her gün işe yürüyerek *gelirdi*. (2.13)

cümlesindeki "gelirdi" kelimesinin kelime türünün doğru işaretlenmesi için, aşağıda verilen anlamlardan hangisine ait olduğu bilinmelidir.

1. anlam: gel (fiil kök) + Geniş zaman + Geçmiş zaman
2. anlam: gelir (isim kök) + Geçmiş zaman

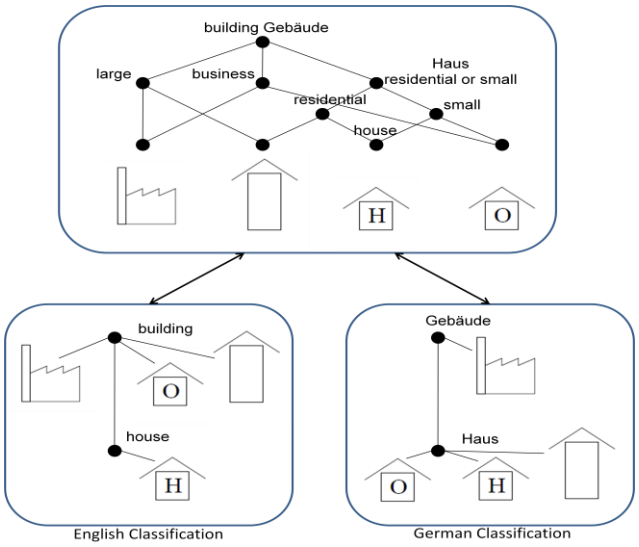
2.2.7 Anlamsal ağ

Anlamsal ağ, günümüzde kullanılmakta olan web mimarisini, verinin uygulamalar, kurumlar ve topluluklar arasında paylaşılarak ve tekrar kullanımını artırarak, geliştirmeyi amaçlayan bir çalışmadır (Herman, 2007b). Anlamsal web ile daha karmaşık ve daha iyi sonuç veren sorgular yapılabilir ve insanlar kendi ihtiyaçlarına en uygun olan veriye ulaşabilir. Günümüzde, sıradan yöntemler uygulanarak geliştirilen uygulamalarda, yapılan bir sorgu sonucunda kullanıcı bağlamıyla ilgisi olmayan sonuçların fazlalığı dikkat çeker. Anlamsal web altyapısı kullanılarak bu tür gereksiz sonuçlarla uğraşma zahmetinden kurtulmaya çalışılır.

Kullanılan ontolojiler sayesinde verilen bağlama en uygun verilere ulaşılmaya istenir. Örneğin; “ağaç” kelimesi sorgu olarak verildiğinde, bilgisayar bilimleri konusu olan ikili ağaç yapıları ile ilgili sonuçlarla, biyoloji konusu olan ağaçlarla ilgili sonuçlar birlikte gelir. Böylesi bir durumda biyoloji konusu ile ilgili sonuçları elde etmek için “ağaç, biyoloji” şeklinde bir sorgu yapılarak daha öncelikli olarak biyoloji alanı ile ilgili sonuçlar elde edilebilir.

Anlamsal ağın temelini ontolojiler oluşturur. Ontoloji öğrenme, var olan verilerden kavram oluşturma ve bu kavramlar arasındaki ilişkileri kurma işlemidir. KAB işlemi kelimenin kullanıldığı bağlamdaki anlamını belirlediğinde aynı zamanda bu kelimenin kavram düzeyi de belirlenmiş olur. Bunun da ontoloji öğrenmede faydası olabilir.

İlk olarak (Kipke ve Wille, 1987) tarafından dile getirilen ve (Priss, 2005) tarafından tekrar üzerinde durulan şekliyle, dilbilimsel veritabanları kavramsal latisler olarak formelleştirildiklerinde, latisler interlingua olarak kullanılabilmektedir. (Old ve Priss, 2001), bir kavram latisinin iki dil arasında köprü görevini yerine getirebileceğini Şekil 2.2 ile göstermişlerdir:



Şekil 2.2 Interlingua olarak kavramsal bir latis**.**

Şekil 2.2, İngilizce ve Almanca dilleri için "building" kelimesine ait kavramsal latisleri göstermektedir. Bu örnek için İngilizce ve Almanca arasındaki en önemli fark, İngilizce'de "house" sadece küçük konutlar için kullanılırken, Almanca'da küçük ofis binaları veya daha büyük konutlar için de "Haus" kullanılabilmektedir. Almanca'da sadece fabrika yapıları için "Haus" kullanılamamaktadır. Üstteki latis, Almanca ve İngilizce kavram latislerini birleştirerek bir bilgi kanalı oluşturmaktadır ve bu yolla sözkonusu anlam belirsizlikleri giderilebilir. Açıktır ki, bu belirginleştirme iki dil arasında yapılacak makine çevirilerinde kullanılabilecektir.

2.3 Kelime Anlamı Belirginleştirme İçin Faydalı Bilgi Türleri

KAB işlemi sürecinde faydalı bazı bilgi türlerinden faydalanmak gerekir. Bunlar kelime türü bilgisi (part of speech-POS), anlamların sıklıkları (frequency of senses), eşdizimlilikler (collocations), seçimsel öncelikler (selectional preferences) ve alt-ulamlama bilgisi (subcategorization information) vb. olabilir.

Kelime türü bilgisi: Herhangi bir kelimenin sözdizimsel sınıf bilgisidir. POS etiketleme belirginleştirme işleminin ilk adımı olarak kabul edilir. Bir kelimeye ait olası anlamların sayısını azaltan faydalı bir işlemdir. Örneğin, "at" kelimesini ele alalım. Tablo 2.6'da "at" kelimesi için TDK Türkçe Sözlük'ten alınan anlamlar görülmektedir. Bu tabloya göre "at" kelimesinin isim kelime türünde bir anlamı varken, fiil olarak birden fazla anlamı bulunmaktadır. Tablo 2.6'da "at" kelimesi için fiil türünde en sık kullanılan üç anlamı verilmiştir.

Kelime türü	Anlam no	Anlamı
İsim	1. anlam	Atgillerden, binme, yük çekme, taşıma vb. hizmetlerde kullanılan, tek tırnaklı hayvan, beygir.
Fiil	1. anlam	Bir cismi bir yöne doğru fırlatmak.
	2. anlam	Bir şeyi yere doğru bırakmak.
	3. anlam	Bir kimsenin ilişiğini kesmek.

Tablo 2.6 "At" kelimesinin anlamları

Eğer “at” kelimesinin verilen bir bağlamda isim türünde kullanıldığı biliniyorsa, bu kelime türünde tek bir anlamı bulunduğu için anlamı kolaylıkla belirlenebilir.

Anlamların sıklıkları: Kelimelere ait anlamların kullanım sıklığı anlam belirginleştirme işleminde önemli bir bilgidir. Genellikle istatistiksel yaklaşımlarda kullanılmaktadır. Bu bilgi elle etiketlenmiş derlemlerden çıkarılabilmektedir.

Eşdizimlilikler: KAB işleminde kullanılan önemli bilgi türlerinden biri de, eşdizimlilik bilgisidir. Herhangi bir kelime grubu içindeki kelimelerin arasındaki ilişkiyi verir. Şöyle ki; çok anlamlı bir kelime yanına başka bir kelimeyi aldığında, oluşacak kelime grubu içinde anlam belirsizliğinden kurtulabilir. Örneğin, “kahve” kelimesi tek başına birkaç anlama geliyorken, “kahve falı” kelime grubunda tek bir anlamı vardır.

Seçimsel öncelikler: Bir kelimenin sözdizimsel ve anlamsal özelliklerini diğer bir kelime belirleyebilir. Buna seçimsel öncelikler denir. Örneğin, “yemek” fiili yanına canlı ya da cansız bir nesne alabilirken, “öldürmek” fiili sadece canlı nesne alabilir. Bu durum aşağıda verilen cümle örneklerinde görülmektedir.

Kedi fareyi *yedi*. (2.14)

Akşam künefe *yedim*. (2.15)

Kayıp kadını ailesi *öldürmüş*. (2.16)

Alt-ulamlama bilgisi: Bu bilgi kelimeler ve öbekler arasındaki belirli ilişkileri gösterir. Örnek olarak “çalışmak” fiiline bakalım. Türkçede bir fiil yanına belli sayıda kelime ve bu kelimelere bağlı durum ekleri almaktadır. Türkçe’de belirtme (accusative), yönelme (dative), çıkma (ablative), bulunma (locative) ve araç (instrumental) durum ekleri bulunmaktadır.

Türkçe'de bulunan durum ekleri Tablo 2.7'de[1] verilmiştir (Kılıçaslan, 1998):

Yalın durum	Belirtme durumu	Yönelme durumu	Çıkma durumu	Bulunma durumu	Araç durumu
-	-(y)I	-(y)E	-dEn	-dE	-(y)lE

Tablo 2.7 Türkçe'de durum ekleri

Tablo 2.8'de, "çalışmak" fiili için TDK Büyük Türkçe Sözlük'ten alınan alt-ulamlama bilgisi ve anlam karşılıkları bulunmaktadır. Tabloya bakacak olursak "çalışmak" fiilinin alt-ulamlama bilgisinin sadece bulunma durumunda (-de durum eki) olduğunda 3. anlamı ile kullanıldığı görülmektedir.

Anlam no	Alt-ulamlama bilgisi[2]	Anlamı	Örnek cümle
1.anlam	nsz	Bir şeyi oluşturmak veya ortaya çıkarmak için emek harcamak.	Çalışan ilerler, yerinde kalmaz.
2.anlam	nsz	Herhangi bir iş üzerinde olmak.	Konu üzerinde çok çalıştı.
3.anlam	-de	İşi veya görevi olmak, bulunmak.	İnşaatlarda çalışan işçiler birer ikişer inşaatların kapılarından geri dönüp geldiler.
4.anlam	nsz	Makine veya aletler işe yarar durumda olmak veya işlemekte bulunmak.	Çamaşır makinası çalışmıyor.
5.anlam	-e	Bir şeyi yapmak için gereken çarelere başvurmak, o şeyi gerçekleştirmek için kendini zorlamak, çaba harcamak.	Olduğundan fazla yaşlı görünmeye çalıştığını sezdim.
6.anlam	-e	Bir şeyi öğrenmek veya yapmak için emek vermek.	Dar ve sapa yollardan hızla yürümeye çalışıyorduk.

Tablo 2.8 "Çalışmak" kelimesinin anlamları ve alt-ulamlama bilgileri

[1] Türkçe ekler, çeşitli işlemler sonucunda ses ve şekil değişikliğine uğrar. Bu işlemlerden biri sesli uyumudur. Burada kullanılan büyük harfler biçimbirimlerin (morpheme) değişen halleri için kısaltma olarak kullanılmıştır. *I, E* ve *D* harfleri sırasıyla *i/ı/ü/u, e/a* ve *d/t* ile gerçeklenirler. Bir başka değişiklik de, kelime sonunda bir sesli veya sessiz harf bulunmasına göre eklerden önce tampon görevi gören bir sessizin getirilmesidir. Bunun için kullanılan sessiz harf de *y* harfidir.

[2] Sözlükte *nsz* kısaltması, fiilin nesne almadığını, geçişsiz bir fiil olduğunu gösterir. *-e* kısaltması kelimenin yönelme durum ekiyle kullanıldığını gösterir. *–de* kısaltması kelimenin bulunma durum eki ile kullanıldığını gösterir.

2.4 Kelime Anlamı Belirginleştirme İçin Kullanılan Kaynaklar

Bir önceki bölümde bahsedilen KAB için faydalı bilgi türlerini elde etmek için, çeşitli bilgi kaynaklarından yararlanılmaktadır. Bunlar arasında, makinece okunabilir sözlük (machine readable dictionary – MRD), eş anlamlılar sözlüğü (thesauri), teknik sözlük (computational lexicon) ve derlemler (corpus) bulunur. KAB sistemleri verilen hedef kelimenin bağlamsal özelliklerini bu bilgi kaynaklarından elde edip kelimenin farklı anlam karşılıklarını kıyaslamada kullanmaktadır. Şimdi bu kaynakları ayrıntılı olarak inceleyelim.

2.4.1 Makinece okunabilir sözlük

Gerçeğe uygun doğal dil işleme uygulamalarına yardımcı olması için sağlam sözlüksel ve anlamsal bilgiye ihtiyaç olduğu bilinmektedir. Makinece okunabilir sözlükler doğal dil işlemede kullanmak için uygun bir kaynak olarak görülür. Çünkü bu sözlükler sözlük yazarlarının ortak çabalarıyla yıllar süren çalışmalar sonucu oluşan büyük miktardaki sözlüksel ve anlamsal bilgiyi içermektedir. Makinece okunabilir sözlükler bir veritabanında depolanır ve bazı arayüzeylerle sorgulanabilirler. Bu formattaki ilk sözlükler Merry Webster's Pocket Dictionary (MPD) ve Webster's Seventh New Collegiate Dictionary (W7)'dir (Olney vd., 1967). 1960 yılından önce elle oluşturulmuş ve manyetik teypler ile dağıtılmışlardır. 1960 yılından sonra bu sözlüklerin elektronik sürümleri hazırlanmış ve doğal dil işleme çalışmalarında kullanılmaya başlamışlardır (Gonçalo Oliveira, 2009). 1980'lerden sonra, Longman Dictionary of Contemporary English (LDOCE)'in elektronik sürümü doğal dil işleme çalışmalarında kullanılmıştır (Michiels vd., 1980). Bu sözlük 55.00 girişten (kelime tanımı) oluşan orta ölçekli bir sözlüktür. Sözlük her biri temel olarak anlam tanımları koleksiyonu olan girişlerden (entry) oluşan bir liste şeklinde organize edilmiştir. Her girişin kelimeyi tanımlayan bir temel (head) kısmı vardır. Her anlam tanımı tanımlardan, örneklerden ve temel kelimenin anlamları için ilgili kelimelerden oluşur.

En belirgin özelliği tüm kelime tanımlarında 2000 kelimeden oluşan bir ana sözlük kullanılıyor olmasıdır. Bu özelliği kelime tanımlarındaki kelimelerin örtüşme sayısı ile ölçüm yapan algoritmaların doğruluğunu arttırmıştır ve bu sebeple popüler bir kaynak olmuştur. Her kelime için ilişkili olan anlamlar kümesi olacak şekilde anlamlar eşyazımlı (homograph) olarak gruplanmıştır. Şekil 2.3'de "bank" kelimesinin LDOCE'deki girişi görülmektedir. Şekil 2.3 (Sanderson, 1996)'dan alınmıştır.

I
bank (n)
1 Land along the side of a river, lake.
2 Earth which is heaped up in a field or garden, often making a border or division.
3 A mass of snow, clouds, mud.
4 A slope made at bends in a road or race-track, so that they are safer for cars to go round.
5 *sandbank.*

II bank (v)
6 Of a car or aircraft to move with one side higher than the other, when making a turn *bank up*.

III bank (n)
7 A row, of oar s in an ancient boat or *keys* on a *typewriter*.

IV bank (n)
8 A place in which money is kept and paid out on demand, and where related activities go on *street*.
9 In a place where something is held ready for use, *organic* products of human origin for medical use.
10 A person who keeps a supply of money or pieces for payment or use in a game of chance.
11 Break the bank to win all the money that the *bank* {4}3 has in a game of chance.

V bank (v)
12 To put or keep money in a bank.
13 To keep one's money in the stated bank.

Şekil 2.3 LDOCE'deki "bank" kelimesi tanımı

2.4.2 Eş anlamlılar sözlüğü

Eş anlamlılar sözlüğü kelimeler arasındaki ilişki bilgisini içeren sözlük benzeri bir kaynaktır. Çoğunlukla eş anlamlılık ve zıt anlamlılık ilişkisi bulunur. Bu kaynaktaki girdiler benzer kelimeler arasındaki ayrımları ortaya koymak ve tam olarak doğru kelimeyi seçebilmek için tasarlanmıştır. Sözlüğün aksine bu kaynakta kelime tanımları bulunmaz.

KAB'da en çok kullanılan eş anlamlılar sözlüğü Roget'in eş anlamlılar sözlüğüdür (Chapman, 1977). Bu sözlük 1950'li yıllarda oluşturulmuştur ve şimdiye kadar birçok doğal dil işleme alanında kullanılmıştır. Bu alanlardan bazıları makine çevirisi (Masterman, 1957), bilgi erişimi (Sparck Jones, 1964, 1986) ve içerik analizidir (Sedelow ve Sedelow, 1969). Roget'in eş anlamlılar sözlüğü kullanıldığı ilk KAB çalışması Masterman'ın makine çevirisi konusunda yaptığı uygulamadır (Masterman, 1957). Sonraki yıllarda (Patrick, 1985) fiil anlamı belirginleştirmede bu kaynağı kullanmıştır. (Yarowsky, 1992), Roget'in eş anlamlılar sözlüğündeki genel kategorilerini kullanarak kelimeleri sınıflarına ayırmıştır. Kategorideki her kelime için bir derlemden 100 kelimelik bağlam oluşturmuştur. Derlem olarak Grolier'nin ansiklopedisini kullanmıştır. Sınıflar içinde en fazla ilgi çeken sınıf Bayes kuralında kullanılmak üzere seçilir. Bu metot ile %92 başarı elde edilmiştir. Yarowsky'e göre bu metot, özellikle isimlerin belirginleştirilmesinde başarı sağlayan konuya ait (topical) bilginin çıkarımında en iyisidir.

Roget'in eş anlamlılar sözlüğü 1805'te Peter Roget tarafından oluşturulmuş ve 1852'de ilk basımı yapılmıştır. Roget bu eserine, Yunanca "hazine" anlamına gelen "thesaurus" adını vermiştir. Sözlükteki kelime girişleri alfabetik olarak değil, kavramsal olarak sıralanmıştır. Bu kavramsal dizin yaklaşımı, kelimelerden anlamlara doğru gitmeyen, aksine anlam grupları altında kelimeleri sınıflandıran bir yaklaşımdır. Sözlükte 250.000 kelime sınıflandırılmıştır ve yapı olarak 6 sınıftan oluşmaktadır. Bunlar; Soyut ilişkiler (*Abstract relations)*, Alan (*Space)*, Madde dünyası (*Material World)*, Zihin (*Intellect)*, İrade (*Volition)*, Bilinçli olma (*Sentient)* ve Manevi güçtür *(Moral Powers).* Roget'in ontolojisinde bir yol (path) bu sınıflardan biri ile başlar ve 39 bölümden birine dallanır. Sonra 79 alt bölümden birine gider ve sonrasında 596 temel

gruptan birine ulaşır. Son olarak da 990 temel yapıdan birine ulaşır. Her temel, sözdizimsel kategorilere gore gruplanarak paragraflara bölünmüştür. Sözdizimsel kategoride sıralama isim, fiil, sıfat, zarf, edat, bağlaç, ünlem biçimindedir. Bazı kategorilerde paragraf yokken bazılarında birden fazla paragraf bulunabilmektedir. Her paragraf ise anlamsal olarak ilişkili kelimelerden oluşan ve birbirilerinden noktalı virgüllerle ayrılan gruplar şeklindedir. Şekil 2.4'de Roget'in eş anlamlılar sözlüğünden alınan "wonder" kelimesi için bir temel yapı görülmektedir. Bu örnekte her sözdizimsel kategorideki ilk paragraflar verilmiştir.

Class six: Emotion, religion and morality
Section two: Personal emotion
Sub-section: Contemplative
Head Group: 864 Wonder – 865 Lack of wonder
Head: 864 Wonder

N. *wonder*, state of wonder, wonderment, raptness; admiration, hero worship, 887 *love*; awe, fascination; cry of wonder, gasp of admiration, whistle, wolf wolf, exclamation, exclamation mark; shocked silence, 399 *silence*; open mouth, popping eyes, eyes on stalks; shock, surprise, surprisal, 508 *lack of expectation*; astonishment, astoundment, amazement; stupor, stupefaction; bewilderment, bafflement, 474 *uncertainty*; consternation, 854 *fear*.
…
Adj. *wondering*, marvelling, admiring, etc. vb.; awed, awestruck, fascinated, spellbound, 818 *impressed*; surprised, 508 *inexpectant*; astonished, amazed, astounded; in wonderment, rapt, lost in wonder, lost in amazement, unable to believe one's eyes *or* senses; wide-eyed, round-eyed, pop-eyed, with one's eyes starting out of one's head, with eyes on stalks; open-mouthed, agape, gaping; dazzled, blinded; dumbfounded, dumb, struck dumb, inarticulate, speechless, breathless, wordless, left without words, silenced, 399 *silent*; bowled over, struck all of a heap, thunderstruck; transfixed, rooted to the spot; dazed, stupefied, bewildered, 517 *puzzled*; aghast, flabbergasted; shocked, scandalized, 924 *disapproving*.
…
Vb. *wonder*, marvel, admire, whistle; hold one's breath, gasp, gasp with admiration; hero-worship, 887 *love*; stare, gaze and gaze, goggle at, gawk, open one's eyes wide, rub one's eyes, not believe one's eyes; gape, gawp, open one's mouth, stand in amazement, look aghast, 508 *not expect*; be awestruck, be overwhelmed, 854 *fear*; have no words to express, not know what to say, be reduced to silence, be struck dumb, 399 *be silent*.

Adv. *wonderfully*, marvellously, remarkably, splendidly, fearfully; wondrous strange, strange to say, wonderful to relate, mirabile dictu, to the wonder of all.

Int. amazing! incredible! I don't believe it! go on! well Inever! blow me down! did you ever! gosh! wow! how about that! bless my soul! 'pon my word! goodness gracious! whatever next! never!

Şekil 2.4 Roget'in eş anlamlılar sözlüğündeki "wonder" kelimesi girişi

2.4.3 Teknik sözlük

Teknik sözlükler doğal dil işlemede en önemli kaynaklar arasındadır. Genel olarak aşağıdaki bilgileri içerirler.

1. Kelimelerin ve öbeklerin formları ve anlamları
2. Sözlüksel kategorizasyon
3. Kelimeler ve öbeklerin uygun kullanımı
4. Kelimeler ve öbekler arasındaki ilişkiler
5. Kelimeler ve öbeklerin kategorileri

Yüksek oranda bilgi içeren bu kaynaklar, 1980'li yılların ortasından itibaren el ile oluşturulmaya başlanmıştır. Teknik sözlüklere örnek olarak WordNet (Miller, 1985), CyC (Lenat ve Guha, 1990), ACQUILEX (Briscoe, 1991), COMLEX (Grishman vd.,1994) verilebilir.

Anlamsal teknik sözlüklerin oluşturulmasında iki temel yaklaşım vardır: sıralamalı yinelemeli (enumerative) yaklaşım ve üretken (generative) yaklaşım (Ide ve Veronis, 1998). Sıralamalı yinelemeli yaklaşımda, anlamlar açıkça verilmektedir. Üretken yaklaşımda ise verilen kelimeler ile ilgili anlamsal bilgi eksik verilmiştir ve oluşum kuralları kesin anlam bilgisinin türetilmesinde kullanılır. Sıralamalı yinelemeli ve üretken yaklaşımla oluşturulan sözlüklere örnekler aşağıda verilmiştir.

I. Sıralamalı yinelemeli teknik sözlükler

Sıralamalı yinelemeli teknik sözlükler arasında en fazla bilinen ve KAB uygulamalarında çoğunlukla kullanılan kaynak WordNet'tir (Miller, 1985). Princeton Üniversitesi Bilişsel Bilimler Laboratuarı'nda, insanın zihinsel sözlüğü üzerine araştırmalar yapan bilişsel psikolog Profesör George A. Miller tarafından 1985 yılında geliştirilmiştir. Miller deneyimlerini, zihinsel sözlüğün yapısını mümkün olduğunca yakın bir biçimde yansıtan bir kaynak oluşturmak için kullanmıştır. WordNet İngilizcenin tüm kavramları arasındaki ilişkileri gösteren bir veritabanıdır. İngilizce kelimeleri eş anlamlılar kümesinde sınıflandırır ve kelimelerin kısa, genel

tanımlamalarını yaparak bu eş anlamlılar kümeleri arasındaki çeşitli anlamsal ilişkileri oluşturur (Fellbaum, 1998).

WordNet sürümlerinden biri olan WordNet 2.1 veritabanında bulunan toplam giriş ve anlam sayısı Tablo 2.9'daki gibidir:

Kelime türü	**Eşi olmayan formlar**	**Anlam sayısı**
İsim	117097	145104
Fiil	11488	24890
Sıfat	22141	31302
Zarf	4601	5720
Toplam	155327	207016

Tablo 2.9 WordNet'teki toplam giriş ve anlam sayısı

WordNet'teki kelimelerin sözdizimsel kategorilere göre bulunma sayıları ve toplam anlam sayıları ile bu kategorilerdeki kelimelerin aralarındaki ilişki türleri aşağıdaki Tablo 2.10, Tablo 2.11, Tablo 2.12 ve Tablo 2.13'de verilmiştir.

İlişki	**Tanımı**	**Örnek**
Hypernym	Kavramlardan üst sınıfına	kahvaltı-öğün
Hyponym	Kavramlardan alt sınıfına	öğün-öğle yemeği
Has-Member	Gruptan üyelerine	fakülte-profesör
Member-Of	Üyelerden gruplarına	hostes-uçuş personeli
Has-Part	Bütünden parçaya	saat-yelkovan
Part-Of	Parçadan bütüne	tabak-yemek
Antonym	Zıt anlamlılık	gece-gündüz

Tablo 2.10 WordNet'teki isim ilişkileri

İlişki	**Tanım**	**Örnek**
Hypernym	Olaylardan üst sınıflarına	uçuş-seyahat
Troponym	Olaylardan alt sınıflarına	yürümek-dolaşmak
Entails	Olaydan oluşma sebebine	horlamak-uyumak
Antonym	Zıt anlamlılık	artmak-azalmak

Tablo 2.11 WordNet'teki fiil ilişkileri

İlişki	Tanım	Örnek
Antonym	Zıt anlamlılık	ağır-hafif

Tablo 2.12 WordNet'teki sıfat ilişkisi

İlişki	Tanım	Örnek
Antonym	Zıt anlamlılık	hızlı-yavaş

Tablo 2.13 WordNet'teki zarf ilişkisi

WordNet çalışmaları çeşitli dillerde projeler halinde yürütülmektedir. Türkçe için Sabancı Üniversitesi'nde BalkaNet Projesi'nin bir parçası olarak bir kavramsal sözlük olan Türkçe WordNet hazırlanmıştır (Bilgin vd., 2004). Bulgarca, Çekçe, Yunanca, Romence, Türkçe ve Sırpça olarak 6 farklı Balkan dilinde uygulanan BalkaNet projesi temel olarak Princeton WordNet modelini kullanmıştır. Orjinal WordNet'te olduğu gibi aynı anlamı ifade eden kelimelerin oluşturduğu eşkümelerden ve bu kelimeler arasındaki ilişkilerden meydana gelmektedir. Türkçe WordNet'te Mart 2004 itibari ile 11.628 eşküme ve 17.550 ilişki vardır.

II. Üretken teknik sözlükler

Üretken teknik sözlük dilbilimsel anlambilimin doğal dilin yapısında bulunan bir araya getirilebilirlik ilkesine odaklanan bir teoridir. Bu konudaki ilk büyük çalışma James Pustejovsky'nin üretken teknik sözlüğüdür. Devamındaki önemli çalışmalar (Pustejovsky ve Boguraev, 1993), (Bouillon, 1997) ve (Busa, 1996) tarafından sunulmuştur.

2.4.4 Derlem

Derlem belli prensipler çerçevesinde özel veya genel amaçlı metin ya da konuşma parça ya da bütünlerinin, üzerinde yapılacak araştırmaya uygun işaretlemelerle

beraber bir araya getirilmesinden oluşan bütündür (Kennedy, 1998). Derlemler genellikle dilbilim ve doğal dil işleme uygulamalarının geliştirilmesinde kullanılmaktadır. Bu amaç doğrultusunda kullanıldığında bir derlemde bulunması gereken birtakım özellikler vardır (Orhan, 2006):

- Çalışılan alandaki olası diğer verileri örneklemeli ve simgeleyebilmelidir.
- Kapsamı yeterince büyük, boyutu sınırlı ve statik olmalıdır.
- Uygulama yapılan makine tarafından *okunabilir* olmalıdır. Burada okunabilirlik kavramı sadece yazılı metinleri değil, makine tarafından algılanabilecek herhangi bir sayısal formu da içine almaktadır.
- Standart bir referansa sahip olmalı ve kullanmak isteyen bütün araştırmacılara açık olmalıdır.
- Tüm uygulamacıların uyması gereken tasarım kriterlerine sahip olmalıdır.
- Orijinal olmalı ve yapay olarak üretilmemelidir.

Kullanım alanları olarak makine çevirisi, otomatik metin özetleme, ses tanıma gibi bilgisayar mühendisliği uygulamaları verilebilir. Bunların yanı sıra bilişsel bilimler ve dilbilim açısından dilin sözdizimsel, anlamsal ve söylembilimsel öğelerinin incelenmesi ve teorik tezlerin desteklenmesi için deneysel veri toplanması; cinsiyet, yazın türü gibi öğelere göre kullanım farklılıklarının araştırılması ve elde edilen verilerin dil eğitiminde kullanılması gibi alanlarda da kullanılmaktadır. Özellikle doğal dil işleme alanında ağırlıklı olarak kullanılan istatistiksel modellerin başarılı kullanımı için bol miktarda veriye, yani bir derleme ihtiyaç duyulmaktadır (Manning ve Schütze, 1999).

Günümüz derlemlerinin elektronik ortamda tutuluyor olması araştırmacılara erişim ve kullanım kolaylığı sağlamıştır. Şimdiye kadar hazırlanmış birçok derlem bulunmaktadır. (Kucera ve Francis, 1967) tarafından Brown Üniversitesinde oluşturulan Brown Corpus bir milyon kelime ile en geniş kapsamlı olarak etiketlenmiş derlemlerden biridir. Lancester-Oslo-Bergen (LOB) derlemi ise Brown çalışmasının İngiliz İngilizcesine uyarlanmış şeklidir (Johansson, 1980). (Burnard, 1995) tarafından geliştirilen British National Corpus (BNC) derleminde yüz milyon kelime

bulunmaktadır. Bu derlemler bilim dünyasında kabul görmüş ve belli standartlara uygun geliştirilmiştir.

Türkçe dil çalışmalarında kullanılmak üzere ODTÜ Türkçe Derlem geliştirilmiştir. Bu derlem, ODTÜ-BAP ve TÜBİTAK tarafından desteklenmiş ve ODTÜ-Sabancı Üniversiteleri işbirliği ile gerçekleştirilmiştir. Çalışmada bir ana derlem oluşturulmuş; ayrıca farklı kullanımlar için bu ana derlemden bazı farklı özellikleri olan bir de ağaç bankası derlemi (ODTÜ Treebank) geliştirilmiştir (Oflazer vd., 2002). Derlemde kullanılan metinler 1990 yılı sonrası basılan eserlerden seçilmiştir. Derlemde yaklaşık olarak 2.000.000 kelime bulunmaktadır. 201 kitap, 87 makale ve 3 tane günlük gazeteden seçilmiş haberlerden oluşan 999 farklı yazılı metin kullanılmıştır. Derlemde bulunan metinlerin çoğunluğu biçimbirimsel olarak çözümlenmiştir. Fakat yapısal belirsizlikler tamamen çözülmemiş olduğu için kullanımda bazı problemlerle karşılaşılmaktadır.

2.5 Kelime Anlamı Belirginleştirmede Karşılaşılan Problemler

(Ide ve Veronis, 1998)'e göre KAB'da karşılaşılan üç ana problem vardır. Şimdi bu problemleri açıklayalım.

I. Bağlamın etkisi

Çok anlamlı bir kelimenin anlamını belirlemede bağlam önemli bir rol oynar. Bu sebeple bütün KAB yaklaşımları belirginleştirmede kullanılacak bilgiyi sağlayan hedef kelimenin bağlamına ihtiyaç duyar. Bağlam iki şekilde kullanılır:

- Hedef kelimenin yanındaki kelimelere bakılabilir. Burada hedef kelimeye belli bir uzaklıktaki kelimeler hedef kelimeye olan uzaklıkları ve dilbilimsel özellikleri vb. bakılmaksızın sadece birlikte bulunmasına bakılarak düşünülür.
- Hedef kelime ile yanındaki kelimeler arasındaki ilişkisel bilgiye bakılabilir. Bu ilişkiler uzaklık bilgisi, sözdizimsel ilişki, seçimsel öncelikler, eşdizimlilikler ve anlamsal kategoriler olabilir.

Çoğu belirginleştirme görevi, öncelikli bilgi kaynağı olarak bir kelimenin bulunduğu yerel bağlamı kullanır. Yerel veya mikro bağlam genellikle bir metindeki ya da bağlamdaki hedef kelimenin çevresinde bulunan sınırlı bir çerçeve içindeki kelimeler olarak düşünülür. KAB'da karşılaşılan önemli problemlerden birisi, hedef kelimenin çevresinde bulunan kelimelerin hangi uzaklığa kadar inceleneceğinin net olmamasıdır. Yapılan çalışmalarda tespit edilmiş bir en iyi ölçüm yoktur. Ancak farklı belirsiz kelimeler için farklı ilişki ölçümlerinin daha etkili olduğu söylenmiştir (Agirre ve Edmonds, 2006).

II. Anlamların ayrılması

Anlamların ayrılması KAB'daki bir diğer problemdir. Kelimenin anlam sayısı yapılan uygulamaya göre değişmektedir. MRD ve WordNet gibi sözlüklerden alınan anlamlar doğal dil işleme çalışmaları için çok geniş boyuttadır. Bazı anlamlar diğer anlamların özelleşmiş şeklidir. Kimi durumlarda sözlüklerde bulunmayan anlamların kullanılması da gerekebilir. Bu geniş anlam farkı KAB'da zorluklar çıkarmaktadır. Sözlük anlamlarının birleştirilmesi de bu problemi çözmemektedir.

Anlamların ayrılması kaba taneli (coarse-grained) ve ince taneli (fine-grained) seviyede yapılmaktadır. Kaba taneli anlam ayrımı, kelimenin birbiriyle ilgisi olmayan farklı anlamları olması durumunda yapılan ayrımdır. Yani kelimenin eş sesli olması durumu ile ilgilidir. Örnek olarak "kara" kelimesini ele alalım. Tablo 2.14'de verilen anlam ayrımları kaba taneli olarak yapılmış olan ayrımdır. Verilen örnek cümlelerde kullanılan "kara" kelimeleri eş seslidir.

Anlam no	**Anlamı**	**Örnek cümle**
1. anlam	Yeryüzünün denizle örtülü olmayan bölümü, toprak.	İçlerinden biri *kara* göründü diye bağırdı.
2. anlam	En koyu renk, siyah; ak, beyaz karşıtı.	Ben bir *kara* ağaç gölgesi buldum.

Tablo 2.14 "Kara" kelimesinin kaba taneli anlamları

Kelimenin birbirine yakın anlamları olması durumu göz önüne alınarak yapılan ayrım ince tanelidir. Kelimenin çok anlamlı olması durumunda yapılan bir ayrımdır. "Kara" kelimesi için yapılabilecek ince taneli anlam ayrımı Tablo 2.15'de verilmiştir.

Anlam no	**Anlamı**	**Örnek cümle**
1. anlam	En koyu renk, siyah; ak, beyaz karşıtı.	Ben bir *kara* ağaç gölgesi buldum.
2. anlam	Esmer.	Güneşte yanmış, *kara* bir çocuk gördüm.

Tablo 2.15 "Kara" kelimesinin ince taneli anlamları

Bu tabloda, "kara" kelimesinin 2. anlamı, 1. anlamına bağlı olarak ortaya çıkmış bir yan anlamdır.

III. Değerlendirme

KAB'da yapılan çalışmalarda hem kullanılan yaklaşımların hem de elde edilen sonuçların karşılaştırılması kolay değildir. Her çalışmada farklı metinler kullanılmaktadır ve bu metinler de belirli konularda olduğundan kelime anlamları da sınırlı kalmaktadır. Test için seçilen kelimeler de başarı oranını etkiler. Şöyle ki; test kelimeleri anlam ayrımı kolay yapılabilecek kelimelerden oluşuyorsa başarı oranı yükselmektedir. KAB sistemlerinin belli bir standartta değerlendirilmesi için Senseval çalıştayları[3] 1998 yılından beri yapılmaktadır. Dört senede bir yapılan Senseval çalıştaylarının sonuncusu 2010 senesinde gerçekleştirilmiştir.

[3] http://www.senseval.org

2.6 Kelime Anlamı Belirginleştirme Yaklaşımları

KAB için günümüze kadar yapılmış olan çalışmalarda birçok yaklaşım ortaya atılmıştır. Kullanılan kaynağa göre yapılan sınıflandırmada iki tür yaklaşım bulunmaktadır. Yapılan bu sınıflandırmada, (Agirre ve Edmonds, 2006) kaynağından yararlanılmıştır.

1. Bilgi-tabanlı KAB

2. Derlem-tabanlı KAB

2.6.1 Bilgi tabanlı kelime anlamı beliginleştirme

1980'li yıllara kadar geniş kapsamlı bilgi kaynaklarının yeterince bulunmaması ve mevcut kaynaklara erişimin zor olması nedeniyle, KAB çalışmaları için ihtiyaç duyulan bilginin elde edilmesinde sıkıntı yaşanmıştır. Ancak bu yıllardan itibaren profesyonel sözlük yazarları tarafından oluşturulmuş geniş kapsamdaki sözlüksel kaynakların elektronik sürümlerinin ortaya çıkması ile bu kaynaklara erişim imkânı kolaylaşmıştır. Bu da bilgi tabanlı KAB çalışmalarında artışa neden olmuştur. Yapılan çalışmalar sonucunda KAB için gerekli olan bilgiyi bu kaynaklardan otomatik olarak çıkaran farklı metotlar geliştirilmiştir. Bilgi tabanlı KAB için günümüze kadar sunulmuş olan dört farklı yaklaşım bulunmaktadır (Agirre ve Edmonds, 2006).

1. Sözlüklerden alınan kelime tanımlarını kullanarak bağlamsal örtüşmeyi hesaplayan metotlar.

2. Anlamsal ağlar üzerinden hesaplanan anlamsal benzerlik ölçümüne dayanan metotlar.

3. Seçimsel öncelikleri kullanan metotlar

4. Sezgisel tabanlı metotlar.

2.6.1.1 Sözlüklerden alınan kelime tanımlarını kullanarak bağlamsal örtüşmeyi hesaplayan metotlar

Sözlüklerden alınan kelime tanımlarını kullanarak bağlamsal örtüşmeyi hesaplayan metotlar, Lesk Algoritması ve bu algoritmanın varyasyonları olan Benzetilmiş Tavlama (Simulated Annealing), Basitleştirilmiş Lesk Algoritması (Simplified Lesk Algorithm) ile Uyarlanmış Lesk Algoritmasından (Adapted Lesk Algorithm) oluşmaktadır (Agirre ve Edmonds, 2006).

Lesk Algoritması: Lesk algoritması çok basit bir fikir ile ortaya çıkmıştır. Bu fikir şudur: Bir kelimenin sözlük tanımları, tanımladıkları anlamların iyi göstericileridir. (Lesk, 1986), belirsiz kelimenin sözlükteki tanımı ile bulunduğu bağlamdaki diğer kelimelerin sözlük tanımları arasındaki örtüşen kelimelerin sayısını hesaplayarak belirsiz kelimenin hangi anlamının seçileceğini bulan bir metot geliştirmiştir. Bu metot için makinece okunabilir sözlüklerden biri olan Oxford Advanced Learner's Dictionary (OALD)'i kullanmıştır. Lesk'in bu metottaki düşüncesi ile anlamların tüm kombinasyonlarını bulmaya çalışmak çok fazla hesap gerektirmektedir. Çünkü karşılaştırılması gereken yüksek miktarda veri bulunmaktadır. Ayrıca Lesk metodunda verilen kelimenin sözlükte bulunup bulunmaması durumu da sonucu büyük oranda etkiler. Bu sorunlara rağmen Lesk metodu kendisinden sonra gelen makinece okunabilir sözlükleri kullanan çalışmalara temel olmuş bir metottur.

Lesk'in vermiş olduğu klasik örnekte, "pine" ve "cone" kelimelerinin "pine cone" kelime çiftinde hangi anlamları ile kullanıldıkları gösterilmiştir. Bu kelimeler için OALD'den alınan anlam tanımları Tablo 2.16 ve Tablo 2.17'de verilmiştir (Hornby, 2000).

Anlam no	**Anlamı**
1. anlam	Seven kinds of *evergreen tree* with needle-shaped leaves.
2. anlam	Waste away through sorrow or illness.

Tablo 2.16 "Pine" kelimesinin anlamları

Anlam no	**Anlamı**
1. anlam	Solid body which narrows to a point.
2. anlam	Something of this shape whether solid or hollow.
3. anlam	Fruit of certain *evergreen trees*.

Tablo 2.17 "Cone" kelimesinin anlamları

"pine" ve "cone" kelimelerinin Tablo 2.16 ve Tablo 2.17'de verilen farklı anlam tanımları arasındaki örtüşen kelimelerin sayısını hesaplandığında Tablo 2.18'de verilen değerler elde edilmiştir.

cone / **pine**	1.anlam	2.anlam	3.anlam
1. anlam	0	1	2
2. anlam	0	0	0

Tablo 2.18 "Pine" ve "cone" kelimelerinin anlam tanımlarındaki örtüşme sayısı

Tablodan görüldüğü üzere tüm olası anlam kombinasyonları arasında pine" kelimesinin 1. anlam tanımı ve "cone" kelimesinin 2. anlam tanımı arasında "shape" kelimesi ile bir kelimelik örtüşme var iken, "pine" kelimesinin 1. anlam tanımı ve "cone" kelimesinin 3. anlam tanımı arasında "evergreen" ve "tree" kelimeleri olmak üzere toplam iki kelimelik örtüşme vardır. Lesk algoritması tarafından "pine cone" kelime çiftinin anlamları olarak, "pine" kelimesinin 1. anlam tanımı ve "cone" kelimesinin 3. anlam tanımı en fazla örtüşmeyi sağladığı için seçilir. Lesk OALD kullanarak yaptığı çalışmasında, belirsiz kelimelerden oluşan bir örnek üzerinde %50 ile %70 arası doğruluk oranında başarı elde etmiştir (Lesk, 1986).

Benzetilmiş Tavlama Tekniği: Lesk algoritmasındaki önemli problemlerden biri, bir bağlamda ikiden fazla anlamı belirsiz kelimenin bulunması durumunda belirginleştirme yapıldığında oluşan kombinasyonel artıştır. (Cowie, 1992), Lesk metodundaki bu

problemin üstesinden gelebilmek için Benzetilmiş Tavlama Tekniğini kullanmıştır. Bu tekniğin aşamaları şunlardır:

1. Verilen bir metindeki kelimelerin anlamları kombinasyonlarını veren bir E fonksiyonu tanımlama.
2. Elde edilen anlam kombinasyonları arasındaki en fazla örtüşmeyi veren anlam kombinasyonunu bulma.
    - E ile başla. Her kelime için en sık kullanılan anlamı bul.
    - Her iterasyonda, kümedeki rastgele bir kelimenin anlamını farklı bir anlamı ile değiştir ve E'yi ölç.
3. Anlamların konfigürasyonunda bir değişiklik olmazsa iterasyonu durdur.

Cowie'nin bu metodu elle oluşturulmuş 50 cümle için LDOCE sözlüğünü kullanarak anlam seviyesinde %47 başarı göstermiştir (Cowie, 1992).

Basitleştirilmiş Lesk Algoritması: Lesk algoritmasının farklı bir sürümü olan Basitleştirilmiş Lesk Algoritması ise kelime anlamı kombinasyonlarındaki üstsel artışı çözmek amacıyla oluşturulmuştur. Bu basitleştirilmiş sürümde girdi metnindeki her belirsiz kelime için ayrı belirginleştirme işlemi yapılır. Bir metindeki her kelimenin doğru anlamı o kelimenin sözlük tanımı ile geçerli bağlamı arasındaki en yüksek örtüşme hesaplanarak tek tek belirlenir. Bu algoritmaya ait adımlar aşağıda verilmiştir:

1. Belirginleştirilecek kelimenin tüm anlam tanımlarının makinece okunabilir sözlükten alınması.
2. Her anlam tanımı ile bağlamdaki kelimelerin örtüşme sayılarının hesaplanması.
3. En yüksek örtüşmeyi sağlayan anlamın seçilmesi.

Örnek olarak "pine" kelimesini ve bu kelimenin içinde geçtiği bir cümleyi ele alalım:

Pine cones hanging in a tree. (2.17)

Tablo 2.16'deki "pine" kelimesinin anlam tanımlarını oluşturan kelimeler ile (2.17) cümlesindeki kelimelerin örtüşme sayısı Tablo 2.19'daki gibi olur.

pine	(2.17) cümlesindeki kelimeler
1. anlam tanımını oluşturan kelimeler	1
2. anlam tanımını oluşturan kelimeler	0

Tablo 2.19 "Pine" kelimesinin anlam tanımlarındaki kelimeler ile (2.17) cümlesindeki kelimelerin örtüşme sayısı

Elde edilen bu hesaplamaya göre en fazla örtüşme kelimenin 1. anlam tanımı ile olmuştur. Burada örtüşme "tree" kelimesi ile gerçekleşmiştir. Sonuç olarak "pine" kelimesi verilen cümlede birinci anlamı ile kullanılmıştır denir.

(Vasilescu vd., 2004) tarafından yapılan karşılaştırmalı değerlendirmede, Basitleştirilmiş Lesk algoritması ile orijinal Lesk algoritmasından daha iyi duyarlılık ve etkinlik değerleri elde edilmiştir. Senseval-2'deki tüm İngilizce kelimeler için yapılan belirginleştirme uygulamasında bu algoritma ile elde edilen duyarlılık oranı %58'dir. Aynı veri üzerinde uygulanan orijinal lesk algoritması ile elde edilen %42'lik duyarlılık oranından fazladır.

Uyarlanmış Lesk Algoritması: (Banerjee ve Pedersen, 2002) hazırlamış oldukları Uyarlanmış Lesk algoritmasında bir ilişki ölçümü kullanarak KAB gerçekleştirmişlerdir. Bu algoritmanın adımları aşağıdaki gibidir:

1. Ortasında hedef kelimenin bulunduğu n kelime uzunluğunda bir bağlam penceresi seçilir.
2. Bağlamdaki her kelimenin aday anlamları bulunur.
3. Hedef kelimenin her aday anlamı için:
   - 3.1.1 Hedef kelimenin aday anlamının bağlamda çevresinde bulunan kelimelerle olan ilişkisi ölçülür.
   - 3.1.2 Her anlam kombinasyonu için ilgili sonuçlar toplanır.
   - 3.1.3 Bu toplam hedef kelimenin aday anlamına atanır.
4. İlişkisi en yüksek olan sonuç aday anlam olarak seçilir.

Bu algoritma Senseval-2'deki İngilizce verisi üzerinde uygulandığında %32'lik doğruluk değeri elde edilmiştir.

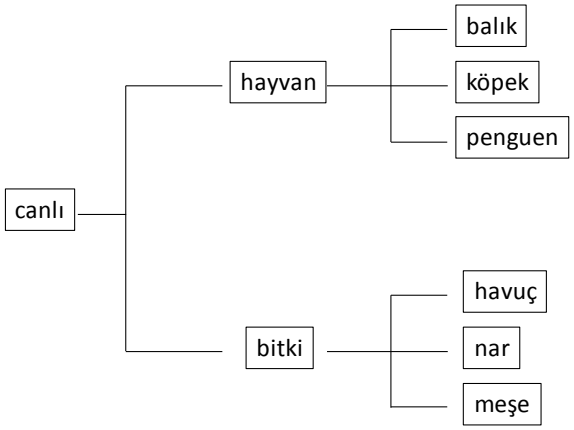
2.6.1.2 Anlamsal ağlar üzerinden hesaplanan anlamsal benzerlik ölçümüne dayanan metotlar

Anlamsal ağlar üzerinden hesaplanan anlamsal benzerlik ölçümüne dayanan metotlarda kavramlar arasındaki anlamsal benzerlik hesaplanır. Bu metotlardan bahsetmeden önce anlamsal benzerlik konusu ve ölçümleri hakkında biraz bilgi verelim.

Anlamsal benzerlik doğal dil işlemede önemli bir konudur. Bir söylemin mantıklı ve tutarlı olması için o söylemi oluşturan kelimelerin anlam olarak ilişkili olması gerekir (Halliday ve Hasan, 1976). Bu, insan diline özgü doğal bir özelliktir ve otomatik KAB işlemi için önemli kısıtlardan biridir.

İki kelime arasındaki anlamsal benzerliği hesaplamak için anlamsal benzerlik ölçümleri kullanılmaktadır. Anlamsal benzerliği ölçmede iki yaklaşım vardır: hesaplama tabanlı ve bilgi teorisi tabanlı metotlar. İlk yaklaşımda büyük bir derlem kullanılır ve bu derlemden anlamsal benzerliği tahmin etmek için istatistiksel bir veri elde edilir. İkinci yaklaşım, kelimeler arasındaki ilişkiyi ve sınıflandırmayı gösteren WordNet benzeri teknik sözlüklerin kullanılmasına dayanır.

ISA (is-a veya is-a-kind-of) hiyerarşi yapısındaki bir bilgi, kelimeler arasındaki uzaklığın belirlenmesinde önemlidir. Şekil 2.5'de ISA ilişkisini barındıran örnek bir hiyerarşik anlamsal bilgi yapısının bir bölümü görülmektedir.



Şekil 2.5 Örnek bir anlamsal bilgi yapısı gösterimi

Burada herhangi iki kelime arasındaki benzerliği bulmak için iki kelimeyi birleştiren en kısa yolun uzunluğuna bakılabilir (Rada vd., 1989). Örneğin "penguen" ile "balık" arasındaki yol *penguen → hayvan → balık* şeklindeyken, "penguen" ile "nar" arasındaki yol *penguen → hayvan → canlı → bitki → nar* şeklindedir. Bu durumda penguenin balığa nardan daha çok benzediği söylenebilir. Ancak iki kelime arasındaki en kısa yola bakma yaklaşımının doğru sonuç vermediği durumlarla da karşılaşılabilir. Bu durumlarda hiyerarşik anlamsal ağlardan faydalanılmaktadır. Hiyerarşik anlamsal ağlarda hiyerarşinin üst katmanlarındaki kavramların daha genel anlamları vardır ve aralarındaki benzerlik daha azdır. Alt katmanlardaki kavramların daha belirli anlamları vardır ve aralarındaki benzerlik fazladır (Li vd., 2003). Bu yüzden kavramların hiyerarşideki derinlikleri de hesaba katılmalıdır. Bundan başka kelimeler arasındaki benzerliği etkileyen bir diğer faktör de anlamsal ağlardaki bölgesel yoğunluktur.

Şekil 2.5'deki gibi bir ağacın olmadığı ya da yeterli olmadığı durumlarda iki kelime arasındaki anlamsal benzerliği bulmak için büyük metinlerden elde edilen istatistiklerin kullanılması önerilmektedir (Amasyalı, 2006).

Kelimeler arasındaki benzerlik kavramının ölçümü için araştırmacılar tarafından günümüze kadar birçok fikir ortaya atılmıştır ve bu fikirlerin çoğu sözlükteki kelimelere ait bilgi yapısını içeren bilgiye dayandırılmıştır. Bu ölçümlerin tümü WordNet'in sağladığı kavram hiyerarşisini temel alır. Günümüze kadar hazırlanmış anlamsal ölçümler şunlardır:

Resnik ölçümü: (Resnik, 1995) bilgi içeriğini (information content-IC) temel alan bir ilişki ölçümünü göstermiştir. Bilgi içeriği metinde bulunan işaretlerden oluşan bir hiyerarşideki her kavrama atanan bir değerdir. Bir kavramın bilgi içeriği bu kavramın büyük bir metindeki bulunma frekansı belirlenerek hesaplanır ve bu suretle olasılığı bir maksimum olasılık hesaplanması ile belirlenir. Resnik'e göre bu olasılığın negatif logaritma değeri kavramın bilgi içeriğini verir.

$$IC(kavram) = log\ (P(kavram)) \tag{2.1}$$

$$sim_R(c_1, c_2) = IC\big(lsc(c_1, c_2)\big) \tag{2.2}$$

Wu ve Palmer Ölçümü: (Wu ve Palmer, 1994)'a ait anlamsal benzerlik ölçümü yol uzunluğunu temel alır:

$$sim_{WUP}(c_1, c_2) = \max\left(\frac{2 * Depth(lcs(c_1, c_2))}{len(c)}\right) \quad (2.3)$$

Hirst ve St. Onge Ölçümü: Bahsedilen çoğu anlamsal benzerlik ölçümünde WordNet'te bulunan isimlere ait ISA ilişkileri kullanılmıştır. (Hirst ve St. Onge, 1998), WordNet'te bulunan diğer ilişki türleri de düşünerek bir ilişki ölçümü yapmışlardır ve ölçümü sadece isimler ile kısıtlamamışlardır:

$$rel_{HS}(c_1, c_2) = C - patlenght - kxd \quad (2.4)$$

Jiang ve Conrath Ölçümü: (Jiang ve Conrath,1997) Resnik tarafından tanımlanan bilgi içeriğini kullanmışlardır:

$$rel_{jcn}(c_1, c_2) \frac{1}{IC(c_1)+IC(c_2)-2*IC(lcs(c_1,c_2))} \quad (2.5)$$

Leacock ve Chodorow Ölçümü: (Leacock ve Chodorow,1998) tarafından sunulan bu benzerlik ölçümü ISA hiyerarşisindeki iki isim kavramı arasındaki en kısa yolu bulmaya dayanır:

$$sim_{LCH}(c_1, c_2) = \max\left(-log\ \frac{ShortestLen(c_1,c_2)}{2.TaxonomyDepth}\right) \quad (2.6)$$

Lin Ölçümü: (Lin, 1997), kavramlar arasındaki anlamsal ilişkinin ölçümünü kendi benzerlik teoremine dayandırmıştır:

$$related_{lin} = \frac{2x(lcs(c_1,c_2))}{IC(c_1)+IC(c_2)} \quad (2.7)$$

Bütün bu anlamsal ilişki ölçümlerinin uygulamaları WordNet Benzerlik Paketi'nde (WordNet Similarity Package) bulunmaktadır. Bu paket iki kelime veya iki anlam arasındaki benzerliği hesaplayabilir.

Anlamsal ağlar üzerinden hesaplanan anlamsal benzerlik ölçümüne dayanan metotlar uygulandıkları bağlamın boyutuna bağlı olarak iki kategoriye ayrılır.

- Sözdizimsel ilişkilerle veya yerel bir bağlamda bulunmaları ile birbirlerine bağlı olan kelimeleri belirginleştirmek için anlamsal ölçümlerin kullanıldığı *yerel bir bağlamda* uygulanabilir metotlar.
- Anlamsal benzerlik ölçümlerinden türeyen sözlüksel zincirlerin olduğu *genel bir bağlamda* uygulanabilir metotlar.

Yerel bir bağlamda anlamsal benzerliğin kullanılması: Kısıtlamaları olmayan bir metinde anlamsal benzerlik ölçümleri ile KAB kolay bir işlem olarak görülmemektedir. Bir metinde genelde ikiden fazla anlamı belirsiz kelime bulunduğu için genellikle bir kelime ile bağlamındaki diğer kelimeler arasında bulunan uzaklığın o kelimenin anlamını etkilediği belirsiz kelimeler kümesi ile çalışılmaktadır.

(Patwardhan vd.,2003) anlamsal benzerlik ölçümlerinden bazılarını Semeval-2 İngilizce sözlüksel örnek verisinden alınan 1.723 belirsiz ismin doğru anlamının belirlenmesi için kullanmışlardır. Yapılan deneyler sonucunda Jiang ve Conrath'ın ölçümü ile en yüksek doğruluğu elde etmişlerdir.

Genel bir bağlamda anlamsal benzerliğin kullanılması: Sözlüksel zincirler (lexical chains) anlamsal olarak ilişkili kelimelerden oluşan anlam yapılarıdır. Bu yapıların metin özetleme, metin gruplama, bilgi erişimi, akıllı yazım denetimi ve KAB gibi çeşitli uygulama alanları vardır. Sözlüksel zincirleri KAB alanında kullanan çalışmalardan bazıları aşağıda verilmiştir.

(Okumura ve Honda, 1994) ise bir Japonca teknik sözlüğünü kullanarak yaptıkları çalışmada sözlüksel zincirlerle %63,4 duyarlılık değerine ulaşmışlardır. (Mihalcea ve Moldovan, 2000) kendi sundukları sözlüksel zincirlemeye benzer bir yaklaşımla Semcor derleminde %90'ın üzerinde duyarlılık ve %60 oranında geriçağırım elde etmişlerdir. (Galley ve McKeown, 2003), Semcor'un bir alt kümesindeki isimler üzerinde bir sözlüksel zincirleme algoritması uygulamış ve %62,1 tutarlılık değerine ulaşmıştır.

2.6.1.3 Seçimsel öncelikleri kullanan metotlar

Bölüm 2.3'de ayrıntılı olarak anlattığımız seçimsel öncelikler, KAB çalışmaları için önemli bir kaynaktır. Ancak uygulanabilirliği ve kullanılabilirliği düşüktür.

(Agirre ve Martinez, 2001) kelimelerin seçimsel öncelikleri ile ilgili bilgiyi daha uygun ve erişebilir yapan bir metot sunmuştur.

(McCarthy ve Carroll, 2003) İngilizce için oluşturulan bir KAB sisteminde edinilen seçimsel öncelikleri otomatik olarak bir araya getiren detaylı bir değerlendirme yapmışlar ve Senseval-2 derleminde %52,3 duyarlılık ve %20 geriçağırım elde etmişlerdir.

2.6.1.4 Sezgisel tabanlı metotlar.

Kelime anlamlarının tahmininde en kolay yol büyük metinlerde ortaya çıkan dilin özelliklerinde bulunan sezgilere güvenmektir. Bu sezgisel durumlar şunlardır:

- En sık kullanılan anlam
- Her söyleme bir anlam
- Her eşdizimliliğe bir anlam

En sık kullanılan anlam: Bir kelimenin olası anlamları arasında genelde bir tanesi daha sık kullanılır. Bir kelimenin sıklık verisinin kullanılabileceği varsayımıyla her kelimeye en sık kullanılan anlamını atayan bir KAB metotu tasarlanabilir. Bu çok basit metot KAB için temel olarak alınmaktadır. (Gale vd., 1992a)'ne göre geliştirilen KAB sistemleri bu temelden daha başarılı olmalıdır.

Her söyleme bir anlam: (Gale vd., 1992b) söylem ve anlam arasında güçlü bir ilişki olduğunu fark etmişler ve şu hipotezi ileri sürmüşlerdir: "Bir kelime bir söylemde birden fazla geçiyorsa bu kelimeler aynı anlamları ile kullanılmış olabilirler.".

Her söyleme bir anlam hipotezi beş nesne ile uygulanmış bir deneyde iki taraflı belirsizlik ile test edilmiştir. Her nesne dokuz belirsiz kelime için bir grup anlamları ile verilmiştir ve bu kelimeler için toplamda 82 çift uyuşma satırı vardır. Uygulama sonucunda %98 olasılıkla iki kelimenin aynı söylemde aynı anlamda kullanılabileceğini göstermişlerdir.

Her söyleme bir anlam düşüncesinin KAB sistemlerinde bir kısıt olarak kullanılması başarımı arttırmak için kullanılabilir. (Yarowsky, 1995) her söyleme bir anlam ve her anlama bir eşdizimlilik sezgisel yöntemlerini kendi yinelemeli önyüklemeli (bootstrapping) algoritmasında kullanarak %90,6 olan performansı %96,5'e çıkarmıştır.

Her eşdizimliliğe bir anlam: Her eşdizimliliğe bir anlam sezgisel yöntemi özünde her söyleme bir anlam hipotezi ile benzerdir, ancak farklı bir kapsamı vardır. (Yarowsky, 1993) sunduğu bu yöntemde, bir kelimenin aynı eşdizimlilikte kullanıldığında anlamını sürdürmeye meyilli olduğunu söylemiştir. Diğer bir deyişle, bir hedef kelimenin anlamı için yanındaki kelimeler güçlü ve tutarlı ipuçları sağlar.

2.6.2 Derlem tabanlı kelime anlamı belirginleştirme

Derlem tabanlı KAB'da en başarılı yaklaşımlar derlemden sınıflayıcıyı veya istatistiksel modelleri öğrenmek için kullanılan istatistiksel algoritmalar ve makine öğrenmesi algoritmalarıdır. Derlem tabanlı KAB, genel olarak denetimli (supervised) ve denetimsiz (unsupervised) olmak üzere ikiye ayrılır. Denetimsiz KAB yönteminde anlam etiketli olmayan bir derlem kullanarak belirginleştirme gerçekleştirilir. Denetimli makine öğrenme ile gerçekleştirilen KAB'da ise anlam etiketli derlemler kullanılır. Bu yöntemlerden denetimsiz KAB yöntemi, üçüncü bölümde bir uygulama eşliğinde anlatılmıştır. Benzer şekilde, denetimli KAB yöntemi de dördüncü bölümde ayrıntılı olarak anlatılmıştır.

3. DENETİMSİZ DERLEM TABANLI KELİME ANLAMI BELİRGİNLEŞTİRME: BİR EŞDİZİMLİLİK UYGULAMASI

3.1 Denetimsiz Derlem Tabanlı Kelime Anlamı Belirginleştirme Yaklaşımı

KAB alanındaki araştırmalar çeşitli bilgi kaynaklarının kullanımına dayalı algoritmaların gelişmesine neden olmuştur. Bu algoritmalar, teknik sözlükler, eş anlamlılar sözlüğü ve ontolojilerin bulunduğu bilgi kaynaklarını kullanan bilgi açısından zengin teknikleri içermektedir. Bu tür kaynakların bulunamadığı veya yeterli olmadığı durumlarla da karşılaşılabilir. Böyle durumlarda kaynakların elle oluşturulması gerekmektedir ve bu oluşturma aşaması oldukça zor ve pahalı bir süreçtir. Bilgi edinmedeki bu dar boğazdan kurtulmak için denetimsiz yaklaşım öne sürülmüştür. Denetimsiz yaklaşımda anlamları elle etiketlenmemiş bir derlem kullanarak belirginleştirme gerçekleştirilir.

Denetimsiz yaklaşımda insan-tanımlı kelime anlamları kullanılmaz. Bunun yerine her kelime için anlamlar kümesi, eğitim setindeki her kelime örneğinden otomatik olarak oluşturulur. (Schütze, 1998) denetimsiz KAB çalışmasında kelime anlamlarını derlemden elde etmek için bir yığılmalı kümeleme algoritması kullanmıştır. Bu metot vektör-uzay modeline dayanmaktadır. Hedef kelimenin anlamları bu vektör gruplarıyla gösterilmiştir. Gruplandırmanın aşamaları şunlardır:

1. Eğitim verisinde olan her kelime için ayrı bir grup ata.
2. En benzer iki grubu birleştir.
3. Önceden belirlenen sayıda gruba ulaşıncaya veya belirli bir eşik değerine ulaşıncaya kadar grupları birleştirme işlemini tekrarla.

Schütze bu çalışması ile kelimeler bağlı olarak belirsiz örnekler üzerinde %89-97 doğrulukla belirginleştirme gerçekleştirmiştir.

(Yarowsky, 1995) tarafından sunulan denetimsiz yaklaşım, elle etiketleme gerektiren zaman alıcı denetimli yaklaşımlarla performans açısından yarışmaktadır. Bu

yaklaşım her söyleme bir anlam ve her eşdizimliliğe bir anlam gibi iki kısıtı temel alıyordu. Yarowsky kelimeleri Roget'in eş anlamlılar sözlüğündeki kategorilere göre sınıflandırmıştır. Grolier'in ansiklopedisini derlem olarak kullanmıştır. Kategorideki her kelime için 100 kelimelik bir bağlam çıkarmıştır. Bu bağlamlardaki kelimelerin oluşlarını kelimelerin sınıf gösterimleriyle hesaplamıştır. Çok anlamlı bir kelimenin anlamını belirginleştirmek için çok anlamlı kelimenin bulunduğu 100 kelimelik bir bağlam çıkarmıştır. En fazla ilgi çeken sınıf Bayes kuralında kullanılmak üzere seçilir. Bu metot 12 kelime üzerinde test edilmiştir ve %92 başarı elde edilmiştir.

(Resnik, 1995) bir denetimsiz KAB ortaya atmıştır. Bu metotta hedef kelimeyi modifiye eden fiiller, sıfatlar veya isimleri veri kaynağı olarak belirginleştirme işleminde kullanmıştır.

(Ng ve Lee, 1996) örnek tabanlı sınıflandırma yöntemini kullanan bir sistem olan LEXAS'i geliştirmişlerdir. Sistem giriş olarak doğru POS ile etiketlenmiş cümleleri alır. Bir bağlam kelimesi w için belirlenen olası anlamlar belirli bir POS içinde olacak şekilde azaltılır. Sonrasında her w örneği için, bir örnek tabanlı sınıflayıcı öğrenilir. Eğitim aşaması tamamlandıktan sonra aynı analizler test cümleleri için yapılır.

Takip eden alt bölümde tezin ilk altı aylık döneminde hazırlanmış olan bir KAB çalışmasından bahsedilecekdir. Bu çalışma eşdizimlilik bilgisini kullanarak Türkçe için hazırlanmış olan bir denetimsiz derlem tabanlı KAB uygulamasıdır (Aydın vd., 2007).

3.2 Uygulama

3.2.1 Uygulamanın aşamaları

1. *Kullanılacak yöntemin ve özelliklerin belirlenmesi:* Çalışmada herhangi bir öğrenme işlemine ihtiyaç duymayan algoritmalarla derlemlerden elde edilecek örnekler üzerinde işlem yapılmış ve bütün bir bağlam üzerinde değil cümle tabanlı olarak belirginleştirme yapılmıştır. Kullanılacak algoritmalar olarak başlangıç için iki

yaklaşım seçilmiştir. İlk yaklaşım ile cümle içinde bazı bölgelerde hiçbir sözdizim, anlambilim vb. özelliğe bakılmaksızın eşdizimlilik aranmıştır. İkinci yaklaşımda ilk yaklaşımın biraz daha geliştirilmiş şekli ile sözdizimsel bir özellik olan durum (case) bilgisi hem kelime hem de eşdizimlilik için kullanılmıştır.

2. *Kelime seçimi:* Kelimeler isim türünde seçilmiş ve çok anlamlı kelimeler olmasına özen gösterilmiştir. Seçilen kelimeler: "kahve", "saat" ve "pas" kelimeleridir. Bu kelimelere ait eşdizimlilikler ve anlamları TDK Türkçe Sözlük'ünden alınmıştır. Derlemlerden çıkarılan örnek cümleler içinde bu eşdizimlilikler aranmıştır. Aşağıda Tablo 3.1 ve Tablo 3.2'de "kahve" kelimesine ait eşdizimliliklere atanılan anlamları görebiliriz. Bu anlamların atanması elle yapılmıştır. Eşdizimlilikleri oluşturan belirginleştirmek istediğimiz kelime ile diğer kelimenin birbirlerine göre konumlarını da sınıflandırma işleminde göz önüne alınmıştır. Bu da diğer kelimenin hedef kelimenin öncesinde ve sonrasında olma durumu şeklinde yapılmıştır. Bunun yanında diğer kelimenin durum bilgisi de sınıflandırmada dikkate alınmıştır.

	İçecek	**Toz**	**Kahvehane**
yalın	sade	Kuru,hazır, çekirdek	
-i durum eki	şekerli, okkalı		

Tablo 3.1 Hedef kelimenin öncesinde kelime olması durumu

	İçecek	**Toz**	**Araç**	**Kahvehane**
yalın				
-i durum eki			Dibeği, fincanı, ocağı, takımı, dolabı, kaşığı, değirmeni, makinesi, tabağı, tepsisi, cezvesi	Parası, ağabeyi, ağası

Tablo 3.2 Hedef kelimenin sonrasında kelime olması durumu

3. *Örnekleme toplama:* Örnekleme verisi için ODTÜ derlemi (Say vd., 2002) ve Trakya derlemleri kullanılmıştır. ODTÜ Türkçe Derlem Türkçe dil çalışmalarında

kullanılmak üzerinde geliştirilmiştir. Ancak derlemde bazı yapısal problemler bulunmaktadır. Bu çalışmada derlemden cümle çıkarılması aşamasında karşılaşılan bu yapısal problemler çözülmeye çalışılmıştır. Trakya Derlemi ise ODTÜ derlemi gibi herhangi bir işaretleme vb. kullanılmaksızın doğrudan internetten elde edilen verilerle oluşturulmuş işlenmemiş bir yapıdadır.

4. *Örneklerin sınıflandırılması:* Cümle tabanlı bir KAB işlemi gerçekleştirilmesi için öncelikle mevcut derlemlerden seçilen "kahve", "saat" ve "pas" kelimelerinin içinde geçtiği cümleler çıkartılmıştır. ODTÜ derlemi paragraf tabanlı olarak ve XML tabanlı Corpus Encoding Standard (CES) işaretlemesi kullanılarak hazırlandığı için öncelikle bir ayrıştırma (parsing) işlemi kaçınılmazdır. Bu amaçla Java, Python ve C++ XML ayrıştırıcıları kullanılarak derlem ayrıştırılmaya çalışılmışsa da derlemdeki işaretleme problemleri sebebi ile başarılı olunamamıştır. Daha sonra AWK betikleri (script) yardımı ile (eksik işaretleme sebebi ile bazı cümlelerin tamamının ayrıştırılamaması gibi) en az hata ile ayrıştırma işlemi gerçekleştirilmiş ve her cümle bir satıra gelecek biçimde derlem düzenlemesi yapılmıştır. Trakya derlemi için de yine AWK betiği kullanılarak aynı biçimde derlem düzenlemesi yapılmıştır. Derlem düzenlemesinin ardından belirginleştirme işleminde kullanılacak olan kelimelerin içinde geçtiği cümlelerin her iki derlemden de çıkartılarak birleştirilip tek bir dosya haline getirilmesi gerekmiştir ve bu amaçla Python programlama dili ve düzenli ifadeler kullanılmıştır. Sonuçta hem ODTÜ hem de Trakya derlemlerinden seçilen her kelime için bu kelimelerin geçtiği cümlelerin bulunduğu ayrı dosyalar elde edilmiştir.

5. *Belirlenen metot için uygun veri yapısının tasarlanması ve ayrıştırılması:* Hesaplamalı bir işlem yapılacağı için XML tabanlı bir veri yapısı tasarlanmasının yapılacak işlemler ve programlama için en uygunu olacağı düşünülmüştür. Bu doğrultuda kullanılacak iki yaklaşım için XML yapıları tasarlanmıştır. Ortaya çıkan XML gösterimini elle giriş yapmak çok zahmetli ve hataya açık bir durum olduğu için sadece gerekli bilgileri toplayacak ve istenen XML dosyayı oluşturacak bir grafik arayüz programının yazımı gerekli olmuştur ve bu amaçla C++ ve Qt kullanılarak bir arayüz programı geliştirilmiştir. Oluşturulan bu veri yapısında

seçilen kelime, kelimeye ilişkin eşdizimlilik ve belirginleştirme için kullanılacak bilgiler bulunmaktadır. Bu örneklemeler üzerinde işlem yapılabilmesi için bu gösterimlerin ayrıştırılması gerekmektedir. Bu yüzden Python diline ait expat tabanlı XML ayrıştırıcısı hazırlanmıştır.

6. *Ayrıştırıcıdan elde edilen verilerin kullanımı için "Python" veri yapılarının tasarımı:* XML biçimindeki veriler ayrıştırıldıktan sonra dile ait uygun veri yapıları içinde saklanmaları gerekmektedir. Bu sebeple XML tasarımı ile birebir örtüşen Python veri yapılarının tasarlanması ve ayrıştırma işlemi sırasında içlerinin doldurulması gerekmiştir.

7. *Sınıflandırılan örneklemeler üzerinden analiz yapılıp sonuçların alınması:* Ayrıştırılan örneklemeler üzerinde XML biçiminde hazırlanan kelime ve eşdizimlilik bilgilerinden faydalanılarak analizler yine Python ve düzenli ifadeler kullanılarak yapılmıştır.

3.2.2 Değerlendirme

Uygulamaya aşağıdaki (3.1) örnekleme verisini girdiğimizde elde ettiğimiz veriler Tablo 3.3'deki gibi olmuştur.

Uzanıp kapatmış olduğum kahve fincanını eline aldı. (3.1)

		Cümle içindeki konumu
Hedef kelime	kahve	4. sıra
Bulunan kelime	fincanı	5. sıra

Tablo 3.3 (3.1) cümlesindeki kelimeler ve konumları

Anlam karşılığına baktığımızda “kahve fincanı” için doğru anlamın elde edildiğini görebiliriz. Benzer şekilde farklı bir örnekleme verisi uygulamaya girdiğimizde Tablo 3.4’deki sonuç elde edildi.

Tepsiyle kahve getirip, fincanı kibarca önüne koyuyor. (3.2)

		Cümle içindeki konumu
Hedef kelime	kahve	2. sıra
Bulunan kelime	fincanı	4. sıra

Tablo 3.4 (3.2) cümlesindeki kelimeler ve konumları

Bu örnekleme verisinden elde edilen anlam hatalı olmuştur. Karşılık gelen anlamın “içecek” olması gerekirken “araç” olarak elde edilmiştir. Burada dikkat edilmesi gereken eşdizimlilikleri oluşturan kelimelerin birbirlerine yakınlıklarıdır. Uygulamada cümle içinde eşdizimliliği oluşturan kelimelerin cümle içinde birbirlerine yakınlıkları önemsenmemiştir. Bu yakınlık ölçüsü de önemsendiği takdirde elde edilen anlam karşılıklarının doğruluk yüzdesi artacaktır.

Uygulama içinde kelimeler için biçimsel (morphological) analiz yapan bir program yoktur. Bu nedenle kelimelerin durum bilgilerine ilişkin veriler kullanıcı tarafından girilmiştir. Bu aşamada Zemberek gibi bir yazılımdan faydalanılarak biçimsel analiz yapan bir modül uygulamaya eklenebilir.

Takı analizi yapılamadığı için derlemden örnekleme çıkartılması ve analizde girilen eşdizimlilik için arama yapılaması sırasında karşılaşılan en temel problem “pas” kelimesi için “pası” gibi bir kelimenin bulunması istenirken “pastane” gibi bir kelimenin bulunmasının istenmemesidir. Bu sebeple bulunmaması gereken kelimeler için oluşturulmuş listedeki kelimelerin işleme girmemesi sağlanmıştır. Yani “pas” kelimesi düzenli ifade olarak aranırken bulunan kelimenin “pasta” gibi bir kelime olup olmadığı kontrol edilmiştir. Bu amaçla yaklaşık 26000 kelimelik bir Türkçe sözlükten faydalanılmıştır.

Eşdizimlilik ve sadece durum bilgisinden faydalanan KAB bilgileri oluşturulduktan sonra analiz işlemi derlemlerden elde edilen örneklemeler üzerinde gerçekleştirilmiştir. Analiz işlemi ile elde edilen sonuçlar tekrardan gözden geçirilerek doğruluk ve yanlışlıkları belirlenmiştir. Bunun sonucunda da çoğu durumda net biçimde doğru belirlemenin yapıldığı, bazı durumlarda örneklemenin yapısı sebebi ile rastlantısal biçimde doğru belirlemenin gerçekleştiği diğer durumlarda ise yanlış sonuçların elde edildiği görülmüştür. Bu sonuçlar aşağıda Tablo 3.5'de görülmektedir.

	Örnekleme Sayısı	**Analiz için seçilenler**	**Doğru**	**Rastlantısal doğru**	**Yanlış**
pas	91	9	4	3	2
kahve	655	89	52	7	30
saat	2642	238	106	4	128

Tablo 3.5 Analiz sonuçları

Tablodaki alanları sırası ile açıklamak gerekirse, ilk alan olan "örnekleme sayısı" daha önce bahsedilen derlemlerden elde edilen toplam örnekleme sayısıdır. Bu seçimde esas olan seçilen cümlenin hedef kelimeyi içermesidir. İkinci alan olan "analiz için seçilenler" analiz işlemi sırasında hedef kelimeyi içeren örneklemelerin arasından seçilen ve eşdizimli kelimeleri içeren örnekleme sayısıdır. Üçüncü alan olan "doğru" kısmı değerlendirmenin doğru yapıldığı örnekleme sayısını, dördüncü alan olan "rastlantısal doğru" ise örneklemenin yapısı sebebi ile yanlış değerlendirme yapıldığı halde doğru sonuç üretilmesi durumunu göstermektedir. Son alan ise yanlış biçimde belirleme yapılan örnekleme sayısını göstermektedir.

3.2.3 Sonuç

Uygulamadan elde edilen sonuçların hepsi "analiz için seçilen" örneklemeler düşünüldüğünde (son satır sayılmazsa) %50'nin üzerindedir. Bu oran özellikle

örnekleme sayısı arttıkça daha da artmaktadır. Son satırda “saat” için verilen değerler ilk iki örnekten daha düşük görünmektedir. Ancak bu durum hem seçilme “güneş” vb. eşdizimliliklerin çok genel olmasından hem de sözdizimsel filtrelerin kullanılmamış olmasından kaynaklanmaktadır. Böylece ilave filtrelerin kullanımı ile sonuçların iyileştirilebileceği ve bunun gerekliliği daha iyi anlaşılmaktadır. Açıklanan durum, çalışmanın devamında sözdizim, biçimsel vb. daha fazla özelliğin kullanılması ile çok daha rafine ve doğru sonuçların elde edileceği yönünde umut vericidir. Çünkü hemen hemen tüm yerel ve diğer özelliklerin KAB için faydalı olduğu bilinmektedir (Agirre ve Edmonds, 2006).

Elde edilen yanlış sonuçlar incelendiğinde bu durumun, eşdizimlilik olarak aranan kelimenin belli bir mesafe uzakta durma zorunluluğu konmamasının bir yan etkisi olarak ortaya çıktığı görülmüştür. Yani yanlış belirleme sayılarının da azaltılarak daha iyi sonuçlar alınması mümkündür.

4. DENETİMLİ DERLEM TABANLI KELİME ANLAMI BELİRGİNLEŞTİRME: BİR TÜMEVARIMLI MANTIK PROGRAMLAMA UYGULAMASI

4.1 Denetimli Derlem Tabanlı Kelime Anlamı Belirginleştirme Yaklaşımı

Eğer kelime anlamları elle etiketlenmiş verimiz varsa, anlam belirsizliği problemi için denetimli bir öğrenme yaklaşımı kullanabiliriz. Denetimli öğrenme ile gerçekleştirilen derlem tabanlı KAB'da bir sınıflayıcı yaratılır. Anlamların tahmininde yardımcı olacak özellikler metinden çıkarılır ve sonrasında bu özelliklerden doğru anlamın atanması için bir sınıflayıcı eğitilir. Eğitimin sonucunda bağlamdaki etiketlenmemiş kelimelere anlam etiketlerini atayabilen sınıflayıcı bir sistem elde edilir.

KAB bir tür sınıflandırma problemidir. Sınıflandırma hangi sınıfa ait olduğu bilinen geçmiş verilere dayanarak, yeni bir verinin hangi sınıfa ait olduğunun belirlenmesi işlemidir. Bir sınıflandırıcı (veya sınıflandırma tekniği) girilen veri kümesinden sınıflandırma modelleri geliştiren sistematik bir yaklaşımdır. Her sınıflayıcı giriş verisinin özellik kümesi ve sınıfı arasındaki ilişkiyi en doğru şekilde tanımlayan bir öğrenme algoritması kullanır.

Tablo 4.1'de omurgalıları memeli, kuş, balık, sürüngen veya amfibi kategorilerinden birinde sınıflandırmak için kullanılan örnek bir veri kümesini bulunmaktadır.

İsim	Vücut Sıcaklığı	Deri	Doğur-ganlık	Suda yaşama	Havada yaşama	Bacaklı	Kış uykusu	Sınıf etiketi
İnsan	Sıcakkanlı	Kıl	Evet	Hayır	Hayır	Evet	Hayır	Memeli
Piton	Soğukkanlı	Pul	Hayır	Hayır	Hayır	Hayır	Evet	Sürüngen
Somon_balığı	Soğukkanlı	Pul	Hayır	Evet	Hayır	Hayır	Hayır	Balık
Balina	Sıcakkanlı	Kıl	Evet	Evet	Hayır	Hayır	Hayır	Memeli
Kurbağa	Soğukkanlı	-	Hayır	İkisi de	Hayır	Evet	Evet	Amfibi
Komodo_ejderi	Soğukkanlı	Pul	Hayır	Hayır	Hayır	Evet	Hayır	Sürüngen
Yarasa	Sıcakkanlı	Kıl	Evet	Hayır	Evet	Evet	Evet	Memeli
Güvercin	Sıcakkanlı	Tüy	Hayır	Hayır	Evet	Evet	Hayır	Kuş
Kedi	Sıcakkanlı	Kürk	Evet	Hayır	Hayır	Evet	Hayır	Memeli
Leopar_Balığı	Soğukkanlı	Pul	Evet	Evet	Hayır	Hayır	Hayır	Balık
Kaplumbağa	Soğukkanlı	Pul	Hayır	İkisi de	Hayır	Evet	Hayır	Sürüngen
Penguen	Sıcakkanlı	Tüy	Hayır	İkisi de	Hayır	Evet	Hayır	Kuş
Kirpi	Sıcakkanlı	Diken	Evet	Hayır	Hayır	Evet	Evet	Memeli
Yılanbalığı	Soğukkanlı	Pul	Hayır	Evet	Hayır	Hayır	Hayır	Balık
Semender	Soğukkanlı	-	Hayır	İkisi de	Hayır	Evet	Evet	Amfibi

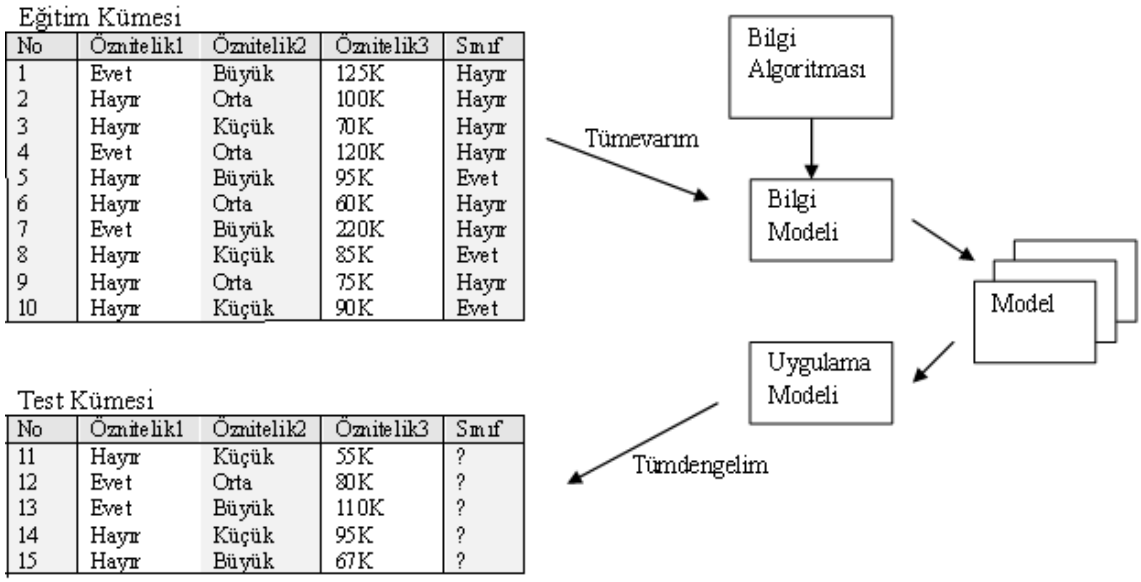
Tablo 4.1 Omurgalı canlılara ait veri kümesi

Tablo 4.2'de özellikleri verilen sınıf etiketi bilinmeyen bir omurgalının hangi sınıfa ait olduğunu Tablo 4.1'de görülen veri kümesinden oluşturulmuş bir sınıflandırma modelini kullanarak belirleyebiliriz.

İsim	Vücut Sıcaklığı	Deri	Doğur-ganlık	Suda yaşama	Havada yaşama	Bacaklı	Kış uykusu	Sınıf etiketi
X	Soğukkanlı	Pul	Hayır	Hayır	Hayır	Evet	Evet	?

Tablo 4.2 X canlısına ait veri kümesi

Öğrenme algoritması ile elde edilen bir model, giriş verilerini en iyi şekilde temsil etmeli ve işlenmemiş yeni verilerin sınıflarını doğru olarak tahmin edebilmelidir. Şekil 4.1'de bir sınıflandırma problemi için genel bir yaklaşım gösterilmektedir. Burada sınıfı belli olan örnekleri içeren bir eğitim kümesinden bir model oluşturulmaktadır. Sonrasında bu model sınıfları bilinmeyen test kümesi üzerinde kullanılır.



Şekil 4.1 Bir sınıflandırma modelinin oluşturulmasındaki genel yaklaşım

KAB işlemi bir sınıflandırma problemi olarak ele alındığında denetimli derlem tabanlı KAB için tümevarımlı öğrenme metotlarından faydalanılmaktadır. Bu makine öğrenmesi algoritmalarından bazıları; Naive Bayes sınıflandırması (Naive Bayes Classification), karar ağaçları (decision trees), k-en yakın komşuluk algoritması (k-nearest neighbor algorithm) ve destek vektör makineleridir (support vector machines)

4.1.1 Naive bayes sınıflandırması

Naive Bayes, hedef değişkenle bağımsız değişkenler arasındaki ilişkiyi analiz eden tahminci ve tanımlayıcı bir sınıflama algoritmasıdır. Naive Bayes sınıflandırıcılar, sınıflandırma kurallarını eğitim verisini inceleyerek öğrenirler. Bir Bayes sınıflandırıcı Bayes karar kuralını uygular ve hatayı en aza indirgemeye çalışır (Agirre ve Edmonds, 2006).

Özellik vektörü $(x_1, \dots, x_n)$ olan bir X örneği verildiğinde, Naive Bayes sınıflandırıcısı,

$$P(X|C) = P(x_1, \dots, x_n|C) \tag{4.1}$$

denklemini kullanarak benzerliği en yüksek yapan bir C sınıf etiketi arar.

Bayes kuralının KAB'da uygulanmasında kullanılan gösterimlerden bazıları Tablo 4.3'de verilmiştir (Orhan, 2006).

Kısaltma	**Anlamı**
W	belirginleştirilecek kelime
s_1, s_2,... s_k ,..., s_K	belirginleştirilecek kelime w'nin anlamları
c_1, c_2,... c_i ,... c_I	w'nin çevresindeki kelimeler
v_1, v_2,... v_j ,... v_J	w'nin anlamını belirginleştirmek için bağlamındaki kelimelerin özellikleri olarak kullanılan kelimeler

Tablo 4.3 Algoritmalarda kullanılan gösterimler

Bayes karar kuralı şu şekildedir: Eğer bir $s' \neq s_k$ için $P(s'|c) > P(s_k|c)$ ise anlam olarak s' seçilir. Genelde $P(s_k|c)$'yi bilmeyiz ancak Bayes kuralı ile hesaplayabiliriz:

$$P(s_k|c) = \frac{P(c|s_k)}{P(c)} P(s_k) \tag{4.2}$$

$P(c)$ tüm anlamlar için sabittir ve elenebilir.

$P(s_k)$ etrafındaki kelimeler bilinmeden s_k anlamının olasılığıdır.

Yapılmak istenen ise w için

$$s' = arg_{s_k} max P(s_k|c) \tag{4.3}$$

$$= arg_{s_k} max \frac{P(c|s_k)}{P(c)} P(s_k)$$

$$= arg_{s_k} max P(c|s_k)P(s_k)$$

$$= arg_{s_k} \max\left(logP(c|s_k) + logP(s_k)\right)$$

Naive Bayes'de kullanılan özelliklerin birbirinden bağımsız olduğu varsayılır ve yapı ile doğrusal sıralama önemsenmez. Buna göre Naive Bayes varsayımı:

$$P(c|s_k) = P\left(\{v_j|v_j\} \in c|s_k\right) = \prod_{v_j \in c} P(v_j|s_k) \quad (4.4)$$

olur.

Naive Bayes karar kuralı ise eğer,

$$s' = argmax(logP(s_k) + \sum_{v_j \in c} logP\left(v_j|s_k\right) \quad (4.5)$$

ise anlam olarak s' seçilir.

$P\left(v_j|s_k\right)$ve $P(s_k)$ en büyük olabilirlik (Maksimum-Likelihood) tahmini kullanılarak hesaplanır.

$$P\left(v_j|s_k\right) = \frac{C(v_j,s_k)}{C(s_k)} \quad (4.6)$$

$$P(s_k) = \frac{C(s_k)}{C(w)} \quad (4.7)$$

$C(v_j, s_k)$: v_j 'nin s_k anlamıyla beraber derlemde geçiş sayısı

$C(s_k)$: s_k anlamının derlemde geçiş sayısı

$C(w)$: w kelimesinin derlemde geçiş sayısı

Naive Bayes kulanılarak KAB alanında yapılmış çalışmalar vardır. Bunlardan bazıları; (Gale vd., 1992a), (Leacock vd., 1993), (Pedersen ve Bruce, 1997) ve (Escudero vd., 2000)'in çalışmalarıdır. (Gale vd., 1992b, 1992c) bu algoritmayı kullanan bir KAB sistemi ile Hansard derlemindeki altı isim (duty, drug, land, language, position ve sentence) için %90 başarı elde etmişlerdir.

Naive Bayes metodunu geliştirmek için farklı yöntemler denenmiştir. Ancak bu yöntemler genel olarak Naive Bayes'deki özelliklerin birbirinden bağımsız olduğu varsayımının zayıflatılması üzerinde durmaktadır (Friedman vd., 1997) (Wang ve Webb, 2002). Bu yöntemlerin uygulanması sonucu hata oranı azaltılmasına karşılık hesaplama zamanı çok artmıştır.

4.1.2 Karar ağaçları

Karar ağaçları öğrenme algoritmaları içinde ağaç yapısı ile kolay anlaşılabilir kurallar yaratabilen, bilgi teknolojileri işlemleri ile kolay bütünleşen başarılı yapılardan biridir.

Bir karar ağacı, karar düğümleri ve sonlandırıcı yapraklardan oluşmaktadır. Sınıflandırılması yapılacak olan örnek, bir yaprak düğüme ulaşıp da örneğe bir değer ataması yapılana kadar, test fonksiyonları düğümlerde özyinelemeli bir şekilde uygulanmaktadır. Dallanma için, her düğümde örneğin bir özelliği test edilmektedir. Ağacı oluşturmak için, seçilen özellikten meydana gelecek bilgi kazancı (information-gain) hesaplanmalı ve ağacın derinliğini azaltmak için önceden belirlenmiş sayıda, en çok bilgi sağlayan özellikler seçilmelidir (Güner, 2008). Aşağıdaki denklemler, bir karar ağacı olusturulurken "A" özelliğinin kullanılması durumunda elde edilen bilgi kazancını vermektedir:

$$I(P(v_i), \dots, P(v_n)) = \sum_{i=1}^{n} - P(v_i) log_2 P(v_i)) \tag{4.8}$$

$$Kazanç(A) = I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) - \sum_{i=1}^{v} \frac{p_i+n_i}{p+n} I\left(\frac{p_i}{p_i+n_i}, \frac{n_i}{p_i+n_i}\right) \tag{4.9}$$

Burada I bilgi içeriği, v_i özelliğin alabileceği sonlu sayıdaki olası değer sayısı, *p* ve *n* de sırasıyla eğitim kümesindeki pozitif ve negatif örnek sayılarıdır.

Karar ağaçları sembolik tümevarım algoritmalarıdır. Temelde ID3 (Induction Decision Tree) algoritmasından geliştirilmiş pek çok karar ağacı algoritması vardır. ID3

Ross Quinlan tarafından geliştirilen ilk karar ağacı algoritmasıdır ve denetimli bir yöntemdir (Quinlan, 1986). Bu algoritma öğrenme sonucunda bir karar ağacı üretmektedir. Entropi formülü sayesinde ağaç oluşturulurken özellikler önem sırasına göre dizilir. Örnek kümesinin özelliklerinden bilgi kazancını hesaplayarak örnek kümesini alt kümelere ayırır ve karar ağacını oluşturur. ID3 algoritmasında ilk aşamada en önemli ya da bilgi kazancı en yüksek olan özellik ilk düğüme yerleştirilir ve diğer aşamalarda ise bu özellik bir düğüm (alt küme için kök) olarak alınır ve buna göre ağaç dallandırılır. Böylelikle ağaç hem daha küçük olur, hem de daha başarılı bir model oluşturulur. İlk olarak ayrık özellikler için geliştirilen ID3 isimli bu algoritma ardından sürekli özellikler içinde uygulanabilmiş ve C4.5 ismini almıştır (Quinlan, 1993).

C4.5 algoritması da Quinlan tarafından geliştirilmiştir. ID3 algoritması gibi sınıflandırma sırasında en ayrıcı özelliği bulmak için entropi kavramından faydalanır. C4.5 algoritması ilk aşamada hesaplanan bilgi kazancı ile beraber alt kümelerde bulunan özelliklerin bilgi kazançlarını hesaplayarak en yüksek olan özelliği düğüm olarak seçer. Karar ağacındaki her dal bir sınıfa gelinceye kadar işlemleri sürdürür.

CART algoritması ise seksenli yılların başlarında (Breiman vd., 1984) tarafından geliştirilmiş olan bir karar ağacı algoritmasıdır. Çok değişkenli karar ağaçları CART algoritmasıyla elde edilir. Bu algoritma sayısal ve reel değerler üzerinde uygulanabilir. CART, her düğümündeki tüm özelikleri tek tek araştırır. Her bir özellik için en çok katkıyı sağlayan en iyi ayrıştırmayı bulur ve n adet aday içerisinden en iyi ayrıştırmayı seçer. Böl-ve- fethet öğrenme metoduna göre işlem yapmaktadır. Şu temel adımlardan oluşur: maksimum ağacı oluşturma, ağaç derinliğini belirleme ve test verilerini ağaca uygulama. Maksimum ağacı oluşturmada amaç en iyi bölmeyi sağlamaktır. Bu adımda GINI ve Twoing algoritmaları kullanılmaktadır. GINI algoritmasında amaç her adımda en büyük veri kümesini elde etmektir. Twoing algoritması ise, GINI' ye göre daha dengeli bir yapı sunar. Bunun nedeni ise her defasında ana ve çocuk düğümlerin %50'sini içermeye çalışmasıdır. Bundan dolayı GINI'ye göre bir yavaşlama olacaktır. CART aynı zamanda sürekli tipteki veriler üzerinde de çalışabilir. Bunun için regresyon yöntemiyle belli aralıktaki değerlere göre sınıflandırma yapılır. İkinci adım olarak ise ağaç derinliğinin belirlenmesi vardır. Bir düğümün sahip olacağı minimum kayıt sayısına n denilsin. Eğer bu değer, kullanıcı tanımlı n değerinden küçük olursa

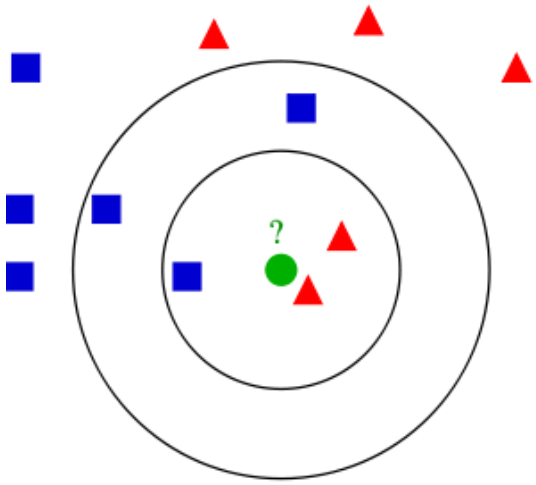
algoritma çalışmayı durdurur. Bu değer genellikle örnek veri kümesinin %10'u kadar seçilir. Daha büyük değerler seçilmesi algoritmanın çalışmasını hızlandırır, ancak bu durumda dengesiz ve yanlış karar veren bir karar ağacı oluşabilir. CART algoritmasının devamı olarak (Crawford, 1990) tarafından OC1 algoritması ile (Zhong vd., 2001) tarafından GDT-NR ve GDT-RS algoritmaları geliştirilmiştir.

Karar ağaçları öğe bulma (Marquez, 1999), makine çevirisi (Tanaka, 1996), çözümleme (Haruno vd., 1998), metin sınıflandırma (Weiss vd., 1999) gibi çeşitli alanlarda uygulanmıştır. (Black, 1988) KAB için bir karar ağacı modeli geliştirmiştir. 22 milyon 'token' dan oluşan bir derlem ve anlam-etiketli bir derlemden denetimli öğrenme yapmıştır. Bu çalışmasında elle yaklaşık olarak 2000 kelime listesi satırı anlam etiketlenmiştir. Bu yöntemdeki zorluk elle anlam etiketleme işlemidir. (Mooney, 1996) , C4.5 algoritmasının KAB'da kullanımını bazı makine öğrenmesi algoritmalarını kullanarak yapılan karşılaştırmalı deneylerle değerlendirmiştir. Bu değerlendirme sonucunda karar ağaçlarının KAB çalışmaları için en iyi performans gösteren yöntemlerden olmadığını savunmaktadır.

4.1.3 K-en yakın komşu algoritması

K-en yakın komşu algoritması örnek tabanlı bir sınıflandırma algoritmasıdır. Örnek tabanlı yöntemlerde eğitim işlemi yoktur. Test yapılacak her bir yeni örnek eğitim kümesindeki her örnekle kıyaslanır. Sınıflandırıcı yeni örneğe en yakın k adet komşu örneği seçer ve bu örneklerin dâhil oldukları sınıflara bakarak bu örneği bir veya daha fazla sınıfa atar. Bu atama işlemi için çeşitli yöntemler kullanılabilir. Örneğin, en yakın komşular arasında baskın olan sınıfa atama yapılabilir. Ya da bunlar arasında en iyi temsil edilen n sınıfa atanabilir. Örnek olarak Şekil 4.2'de ortadaki daire yeni bir örneği temsil eder. Bu örnek ya kare ya da üçgen olarak sınıflandırılacaktır. Peki, bu dairenin aslında kare mi yoksa üçgen mi olduğu nasıl tespit edilecek? Eğer k-en yakın komşu algoritmasındaki değeri 3 seçilirse şekilde de görüldüğü gibi içteki çember içinde kalan örneklere bakılır. Çünkü içteki çember 3 eski örneği çevreliyor. Bu

örneklerden ikisi üçgen biri kare olduğundan k-en yakın komşu algoritmasının kararı üçgen olur. Eğer k değeri 5 seçilirse 5 eski örneği çevreleyen dıştaki çembere bakılır. Bu çember içinde kalan örneklerin üç tanesi kare iki tanesi üçgen olduğundan k-en yakın komşu algoritmasının kararı baskın sınıf olan kare olur (Yıldırım, 2008).



Şekil 4.2 K-en yakın komşu algoritması

Örnekler arasındaki mesafe öklid uzunluğu (eulidean distance) denilen bir uzaklık formülüne göre hesaplanır. N-boyutlu bir uzayda $X = (x_1, \dots x_n)$ ve $Y = (y_1, \dots, y_n)$ gibi iki veri noktası arasındaki uzaklık aşağıdaki formüle göre hesaplanır.

$$\sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \ \dots + \ (x_n - y_n)^2} = \sum_{i=1}^{n} (x_i - y_i)^2 \qquad (4.10)$$

K-en yakın komşu yöntemine göre çalışan sınıflandırıcılarda eğitim işlemi çok kısadır ancak sınıflandırma işlemi görece olarak uzun sürmektedir.

(Ng ve Lee, 1996), k-en yakın komşu algoritmasını KAB için ilk olarak kullanmışlardır. Çalışmalarında her bir örnek cümle için bazı özellikler seçilmekte ve kaydedilmektedir. Daha sonra test edilecek her bir cümle için de bir özellik kümesi bulunmakta ve daha önce tespit edilen örneklerin özellikleriyle karşılaştırılmaktadır. Özellikleri en yakın olan örnek cümledeki anlam, test edilen cümledeki kelimenin de

anlamı olarak seçilmektedir. Algoritmanın en önemli kısmı özelliklerin seçilmesi ve özellikler arası uzaklığın hesaplanmasıdır (Orhan, 2006).

Bir f özelliğinin iki sembolik değeri v_1 ve v_2 arasındaki uzaklığı bulmak için şu formül kullanılmıştır (Ng ve Lee, 1996):

$$d(v_1, v_2) = \sum_{i=1}^{n} \left| \frac{C_{1,i}}{C_1} - \frac{C_{2,i}}{C_2} \right| \quad (4.11)$$

$C_{1,i}$ = f özelliği v_1 olan ve s_1 anlamında olan cümle sayısı.

$C_{2,i}$ = f özelliği v_2 olan ve s_1 anlamında olan cümle sayısı.

C_1 = f özelliği v_1 olan herhangi bir anlamda olan cümle sayısı.

C_2 = f özelliği v_2 olan herhangi bir anlamda olan cümle sayısı.

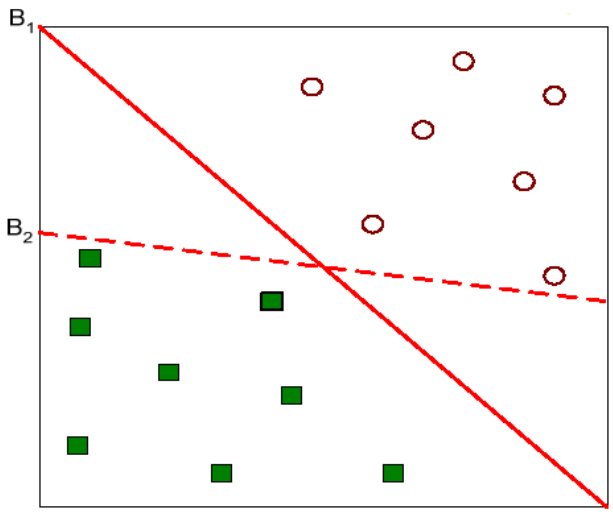
n = Toplam anlam sayısı

İki cümle arasındaki uzaklık bütün özellikler arasındaki uzaklığın toplamına eşittir. Eğer uzaklıkları eşit birden fazla örnek bulunursa herhangi biri rastgele seçilmektedir.

(Escudero vd., 2000), KAB için Naive Bayes ve k-en yakın komşu algoritmalarının karşılaştırılması ile ilgili literatürdeki aykırı sonuçlara odaklanmışlardır. K-en yakın algoritmasının özellik gösteriminde ve gereksiz özeliklerin varlığına karşı oldukça hassas olduğunu tespit etmişler ve bu sebeple yeterli ve etkili olan yeni alternatif gösterimler geliştirmişlerdir. (Stevenson ve Wilks, 2001) k-en yakın komşu algoritmasını farklı bilgi kaynaklarını bütünleştirmede kullanmışlardır ve LDOCE anlamları için yüksek duyarlılıkta sonuçlara ulaşmışlardır. Ayrıca (Hoste vd., 2001, 2002) Senseval-2'deki tüm İngilizce kelime görevleri için iyi performansla bir k-en yakın komşu sistemi kullanmışlardır.

4.1.4 Destek vektör makineleri

(Vapnik, 1963) tarafından önerilen destek vektör makineleri ileri yönde beslemeli bir ağ kategorisidir. İstatistiksel öğrenme teorisinde iyi şekilde kurulmuş bir teoriye sahiptir ve sınıflandırma problemlerine yaklaşım için uygun bir yapıdadır. Sınıfları birbirinden ayıran marjini en büyük, doğrusal bir ayırt edici fonksiyon bulunmasını amaçlar. Eğitim verileri kullanılarak hiperdüzlem bulunduktan sonra, test verileri sınırın hangi tarafında kalmışsa o sınıfa dâhil edilir. Doğrusal olarak ayrılamayan örneklerde veriler daha yüksek boyutlu başka bir uzaya taşınır ve sınıflandırma o uzayda yapılır.



Şekil 4.3 Destek Vektör Makineleri

Örnek olarak Şekil 4.3'de B_1 ve B_2 isimli iki doğru, sınıfları birbirinden ayırmaya çalışıyor. (Vapnik, 1963)'in optimal hiperdüzlem algoritması doğrusal bir sınıflandırıcıyken, daha sonra destek vektör makinelerinin, her nokta çarpımının doğrusal olmayan bir $K(x_i, x_j)$ çekirdek fonksiyonuyla yer değiştirilmesi sonucu doğrusal olmayan sınıflandırma problemlerini de çözebilecekleri gösterilmiştir (Boser vd., 1992).

(Strapparava vd., 2004) İngilizce sözlüksel örnek görevi için %72.6 geriçağırım elde etmiştir. Çalışmalarında heterojen bilgi kaynaklarının bir araya getirilmesinde bir çekirdek fonksiyon kullanmışlardır.

4.2 Tümevarımlı Mantık Programlama

4.2.1 Tümevarımlı mantık programlamanın tarihsel gelişimi

Diğer birçok bilimsel disiplin gibi tümevarım çalışması da felsefenin bir parçası olarak başlamıştır. Filozoflar genellikle tümevarımın deneysel bilimde oynadığı rol üzerinde odaklanmıştır. Bilgilenme sürecinde tümevarım yaklaşımının kullanılabileceği savı, öğretmeni Platon'un tersine nesneleri bir sanrı olarak kabul etmeyip gerçek bilgi kaynağı olarak gören Aristo'ya aittir (Aristotle, 1960). Fakat tümevarımlı düşünmeyi bilimsel düşüncenin temeli olarak belirleyip geliştirmek görevi Aydınlanma Çağı filozoflarına düşmüştür.

Ortaçağdan sonra filozof Francis Bacon, bilimsel araştırma yönteminin felsefi içeriğini saptayarak tümevarımı: "bilmek için sınamak, gözlemlemek, olayları çözümlemek ve sonra ayrı olaylardan genellemeler yapmak ve sonuçlar çıkarmak yöntemi" olarak tanımlamıştır (Bacon, 1620). Bacon'a göre eğer tümevarım yöntemi doğru bir şekilde kullanılırsa, doğal gerçeklere ulaşılabilir ve genellemeler yapılabilir. Sonraki yüzyıllarda tümevarım birçok filozof tarafından ele alınmıştır. David Hume tümevarım yöntemindeki bu akıl yürütmeyi eleştirmiştir. Hume'un eleştirisi iki madde halinde özetlenebilir (Hume, 1748).

1. Tümevarımlı çıkarımın gerekliliğini ortaya koyacak mantıksal kanıta sahip değiliz.
2. Tümevarımlı çıkarım konusunda deneysel hiçbir kanıt yoktur. Böylesi bir kanıt, kanıtlayacağı ilkenin kendisini ön koşul olarak gerektirecektir.

Hume'nin bu mantık hatasındaki eleştirisi gözlemlerden yola çıkarak bilgiye ve gerçeğe ulaşma bilim felsefesini oldukça sarsmıştır. Hume'un, bu problemi ortadan kaldırmak için önerisi şu olmuştur: Deneyimi göz önüne alınız, deneyimin gerçeklerini bir araya getirerek sonuçlara varınız ve sebepleri bu etkilerden çıkartınız. Bacon'dan Hume'e geçişteki en büyük değişiklik gözlemin yorumlanması olmuştur. Tümevarım üzerine önemli söylemlerde bulunan diğer filozoflardan bazıları John Locke, John Stuart Mill, Stanley Jevons ve Charles Sanders Peirce'tür.

20. yüzyılda tümevarım ile ilgilenenler yine formel mantığı geliştiren filozoflar ve matematikçiler olmuştur (Nienhuys ve Wolf, 1997). Tümevarımda ilgilendikleri konular olasılık teoremi ve deneysel veriden hipotez elde etme olmuştur.

(Bruner vd., 1956) kategorizasyon üzerinde durmuşlardır. Kategorizasyon insanın dünyayı anlamlandırma konusunda en önemli bilişsel temelli etkinliğidir. İnsanın fiziksel ve sosyal çevresini kategoriler halinde bölümlemesi ve çevredeki çeşitli öğeleri bu kategorilere yerleştirme etkinliği ve sürecidir. Bruner ve arkadaşları bu konudaki ilk çalışmalarından itibaren kategorizasyonun çeşitli işlevleri olduğunu vurgulamıştır: Çevrenin karmaşıklığını azaltmak; yeni şeyleri mevcut kategorilerimizden hareketle tanımak; davranış ve eylemlerimizi yönlendirmek; kategorilerimizi pekiştirmek; olay, kişi veya nesne sınıflarını düzenlemek ve birbiriyle ilişkilendirmek gibi. Çalışmalarında kavramlardan öğrenmeyi temel olan üç farklı öğrenme stratejisi üzerinde durmuşlardır. Bu kavram tipleri: bağlayıcı (conjunctive), ayrıştırıcı (disjunctive) ve ilişkiseldir (relational). Bu buluşlar ileriki yıllarda TMP'de kullanılmıştır.

1960'lı yılların başlarında örüntü tanıma alanında örüntü gösteriminin en iyi şekli bulunmaya çalışılmıştır. Bu da sembolik tanımlama dillerinin gelişmesini sağlamıştır. (Banerji, 1964) bu alanda birinci derecede mantık kullanımının örüntü tanımada gösterimsel bir araç olarak kullanılabileceğini göstermiştir.

1970 yılında Gordon Plotkin, kendi sunduğu en az genel genelleştirme (least general generalization - lgg) gösteriminin bir uzantısı olan ilgili en az genel genelleştirme (relative least general generalization - rlgg) gösterimini geliştirmiştir (Plotkin, 1970, 1971a, 1971b). Bu gösterimdeki düşünce verilen temel tümceciklerin

lgg'lerinin hesaplanarak genelleştirilmesine dayanıyordu. Bu buluş bu alanda çalışan diğer araştırmacıları da etkilemiştir. (Larson ve Michalski, 1977), mantık programlamayı temel alan INDUCE denilen bir öğrenme sistemi oluşturmuştur. Yine aynı dönemde (Cohen, 1978; Cohen ve Sammut, 1982) öğrenilen kavramları tekrar kullanabilen ilk sistem olan CONFUCIUS sistemini geliştirmişlerdir.

1980'lerin ilk döneminde görülmüştür ki öğrenilen kavramların bulunduğu ilk sistemler bir mantık programı olarak çalışabiliyordu. Bu sistemlerin TMP gerçekleştiren ilk sistemler olduğu söylenebilir. Bu sistemlerden biri Ehud Shapiro (Shapiro, 1983) tarafından geliştirilen MIS sistemidir.

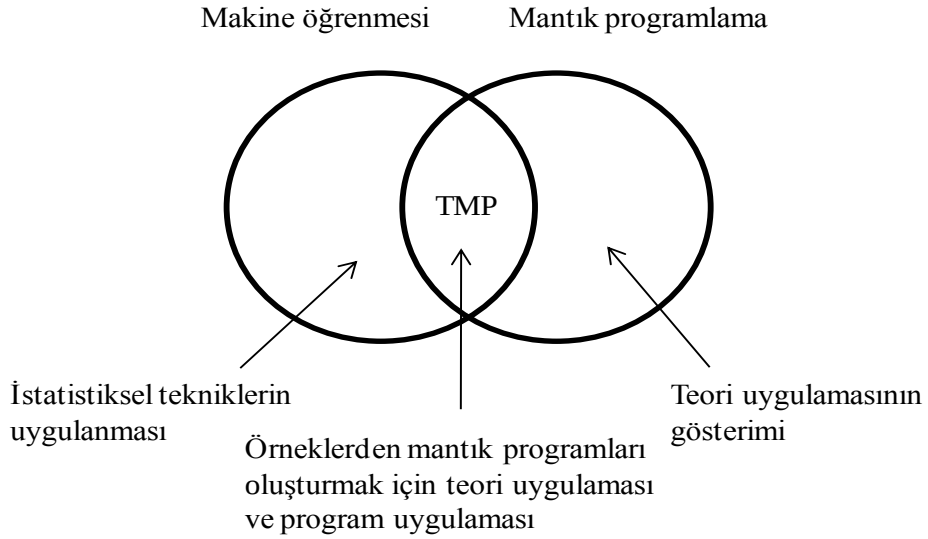
Mantık programlamadaki ve mantık programlamanın ilk ve en temel gerçekleştirmesi olan Prolog programlama dilindeki ilgi çekici gelişmeler sebebiyle TMP'ye olan ilgi, 1980'li yıllardan günümüze kadar artarak devam etmiştir. 1980'li yılların ikinci yarısında Wray Buntine artalan teorisi ile ilgili kapsama (subsumption) gösterimini genelleştirerek daha kullanılır kılmıştır (Buntine, 1986, 1988). Aynı dönemde Muggleton, Buntine ile birlikte ters çözümleme (inverse resolution) gösterimini sunmuştur (Muggleton ve Buntine, 1988). Bu gösterimi kendi hazırladıkları CIGOL sisteminde bir operatör olarak kullanmışlardır. Ters çözümleme kullanan diğer sistemler: FOIL (Quinlan, 1990), LINUS (Lavrac vd.,1991) ve GOLEM (Muggleton ve Feng, 1990) sistemleridir.

1990 yılında Muggleton, TMP'yi ilk olarak isimlendirmiştir ve bu alanı mantık programlama ile makine öğrenmesinin kesişimi olarak tanımlanmıştır (Muggleton, 1990; Muggleton, 1991). 1991 yılında, günümüze kadar her sene tekrarlanarak devam eden ilk uluslararası çalıştay yapılmıştır.

4.2.2 Tümevarımlı mantık programlamanın temelleri

TMP, makine öğrenmesi ve mantık programlamanın kesişimi olarak tanımlanabilir. Amacı gözlemlerden (örneklerden) hipotezler geliştiren ve artalan

bilgisinden (background knowledge) yeni bilgiler elde eden teknikler ve araçlar geliştirmektir. Bu sebeple TMP, makine öğrenmesi ve mantık programlama tekniklerinin her ikisini de kullanır (Muggleton ve De Raedt, 1994).



Şekil 4.4 Makine Öğrenmesi, Mantık Programlama ve TMP

TMP araştırmalarında amaç, Horn tümceciği[4] formunda tanımlanan örnekler ve artalan bilgisi ile mantık programlamadır. Öğrenilen mantık programları genellikle Prolog sözdiziminde tanımlanır ve mantık programlarındaki bildirimsel (declarative) olma özelliği TMP'nin etkili olmasının ana nedenidir.

TMP, hipotezler ve gözlemler için gösterimsel mekanizma olarak hesaplamalı mantığı kullanarak, klasik makine öğrenmesi tekniklerinde var olan iki önemli kısıtın üstesinden gelebilmektedir (Muggleton ve De Raedt, 1994):

1. Sınırlı bilgi gösterimi formalizmini kullanmak.

2. Öğrenme sürecinde çok miktarda artalan bilgisini kullanmadaki zorluklar.

[4] Horn tümceciği en fazla bir pozitif atomik önerme içeren, pozitif veya olumsuzlanmış atomik önermelerin 'veya' bağlacı ile bağlanması ile elde edilir.

İlk sınırlama önemlidir; çünkü bazı alanlardaki incelemeler sadece birinci dereceden yüklem mantığı ile açıklanır, önermeler mantığı ile açıklanamaz. Açıkça problem şudur ki; mantık programı alanı örneklerden sentez yapar ve çoğu mantık programı sadece önermeler mantığı ile tanımlanamaz. Artalan bilgisinin kullanımı önemlidir, çünkü yapay zekâdaki önemli buluşlardan biri zekâ davranışına ulaşmak için artalan bilgisi kullanımının gerekli olduğudur. Mantık bilginin gösterimine mükemmel bir formalizm sunar ve bundan dolayı tümevarımla bütünleşir (Nienhuys ve Wolf, 1997).

TMP, çıkarımda temel mod olarak tümevarımı inceleyerek hesaplamalı mantığı uygular ve teoriyi genişletir. Her ne kadar şimdiki hesaplamalı mantık, teorileri kullanıcıdan alınan mantık formülünden tümdengelimli çıkarım olarak tanımlansa da, TMP teorisi, örneklerden ve artalan bilgisinden mantık programlarının tümevarımlı çıkarım yapması olarak tanımlanır.

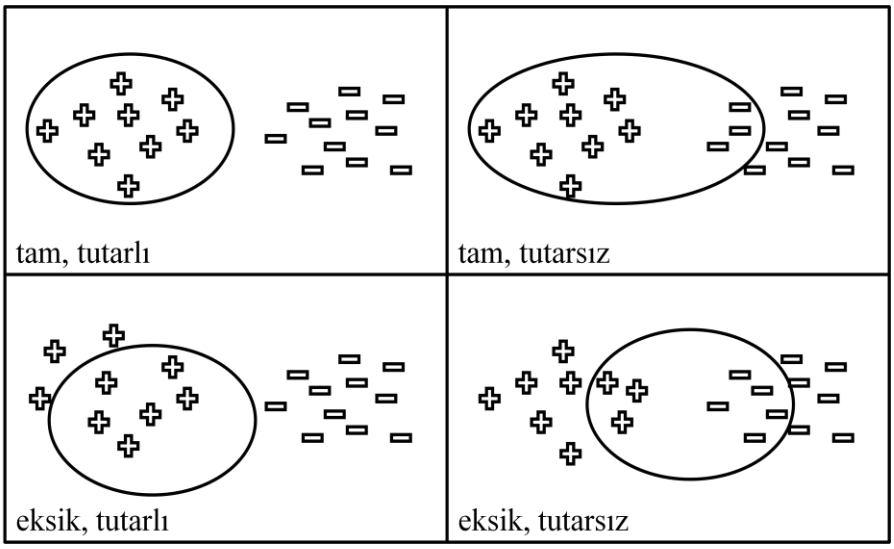
TMP'de kullanılan mantıksal terimlerin tanımları aşağıda açıklanmıştır:

- *B* sonlu sayıda tümcecik içeren artalan bilgisidir.
- *E* sonlu sayıda örnek setidir = $E^+ \cup E^-$
    - *Pozitif örnekler.* $E^+ = \{e_1, e_2, ...\}$ Boş olmayan tanımlı tümcecik kümesidir.
    - *Negatif örnekler.* $E^- = \{\overline{f}_1, \overline{f}_2, ...\}$ Horn tümceciği kümesidir (boş olabilir).
- *B* ve *E* aşağıdaki durumları sağladığında algoritmanın çıktısı *H* hipotezidir.

    - *Öncelikli kabul edilebilirlik (prior satisfiability).* $B \cup E^- \not\models \square$
    - *Sonraki kabul edilebilirlik (posterior satisfiability).* $B \cup H \cup E^- \not\models \square$
    - *Öncelikli gereklilik (prior necessity).* $B \not\models E^+$
    - *Sonraki yeterlilik (posterior sufficiency).* $B \cup H \models e_1 \wedge e_2 \wedge ...$

Elde edilen hipotezin kalitesini test etmek için kullanılan iki ölçüt vardır: tamlık (completeness) ve tutarlılık (consistency). Eğer hipotez tüm pozitif örnekleri kapsıyorsa tamdır (complete). Eğer hipotez tüm negatif örnekleri kapsamıyorsa tutarlıdır (consistent). Bu bağlamda, pozitif ve negatif örnekleri içeren bir örnek kümesi varsa,

hipotezin tamlık ve tutarlılığını gösteren Şekil 4.5[5]'deki dört durum oluşabilir (Lavrac ve Dzeroski, 1994).



Şekil 4.5 Tamlık ve tutarlılık

Tamlık ve tutarlılık ölçütleri birlikte *başarı kriteri* olarak tanımlanmaktadır (De Raedt, 1992).

TMP temelini örneklerle kavram öğrenmeden alır. Kavram öğrenme hem tam hem tutarlı olarak bir C kavramı için bir H hipotezi bulma görevi olarak tanımlanabilir. Makine öğrenmesi araştırmacıları tarafından kapsamlı olarak çalışılmış bu konu tümevarımın diğerlerine göre daha anlaşılır bir şeklidir. Birçok kavram öğrenme görevi yeterli gösterimsel güce sahiptir. 1980'lerde makine öğrenmesi araştırmacıları TMP'nin makine öğrenmesinin bir alt disiplini olarak kurulmasına neden olan mantık programlama gösteriminin kullanımı incelemeye başlamışlardır.

Belirli bir bakış açısıyla, tümevarımlı kavram öğrenme tüm olası kavram tanımları uzayı arasından doğru tanımı arama olarak tanımlanır. Güç problemler için

[5] Şekildeki + işaretler pozitif örnekleri, - işaretler negatif örnekleri ve elips şekli hipotezin kapsam alanındaki örnekleri göstermektedir.

olası kavram uzayı çok büyük olabilir. Arama uzayı kavrama ilişkin "artalan bilgisi" denilen tümceciklerin eklenmesi ile küçülebilir. Artalan bilgisi yardımıyla kavramlar insan düşüncesini tanımlamaya daha yakın bir şekilde ifade edilir. Artalan bilgisi Horn Tümceciği veya Birinci Dereceden Tümcecik Formu (First Order Clause Form) gibi farklı biçimlerde gösterilebilir.

Basit bir TMP problemi olarak ebeveyn ilişkisinin öğrenilmek istendiği bir örneği inceleyelim:

$$B = \begin{cases} dede(X,Y) \leftarrow baba(X,Z), ebeveyn(Z,Y) \\ baba(ali, ayşe) \leftarrow \\ anne(ayşe, ahmet) \leftarrow \\ anne(ayşe, suna) \leftarrow \end{cases}$$

Burada artalan bilgisi bir kuraldan (rule) ve üç gerçekten (fact) oluşmaktadır.

Kural: Eğer X bireyi Z bireyinin babası ve Z bireyi de Y bireyinin ebeveyni ise, X bireyi Y bireyinin dedesidir.

Gerçek-1: Ali Ayşe'nin babasıdır.

Gerçek-2: Ayşe Ahmet'in annesidir.

Gerçek-3: Ayşe Suna'nın annesidir.

Dede ve torunları arasındaki ilişkiyi gösteren pozitif örnekler aşağıda verilmiştir.

$$E^+ = \begin{cases} dede(ali, ahmet) \leftarrow \\ dede(ali, suna) \leftarrow \end{cases}$$

Ek olarak aşağıdaki negatif örneklerin de verildiğini varsayalım.

$$E^- = \begin{cases} \leftarrow dede(ahmet, ali) \\ \leftarrow dede(suna, ali) \end{cases}$$

İdeal bir TMP sisteminin, B'nin doğrulunu kabul edip, yeni E^+ ve E^- gerçekleri ile karşılaşıldığında aşağıdaki ilişkiyi tahmin etmesi beklenir:

$H = ebeveyn(X, Y) \leftarrow anne(X, Y)$

Burada elde edilen hipotez H'ye göre, X bireyi Y bireyinin annesi ise aynı zamanda ebeveynidir.

TMP'nin farklı uygulama alanları vardır. Bunlardan bazıları; yazılım mühendisliği, sistemlerin modellenmesi ve kontrolü, doğal dil işleme, veritabanından bilginin çıkarımı, veritabanı tasarımı, veri analizi, tıbbi teşhisler, çevresel izleme sistemleri ve moleküler biyolojidir (Dzeroski ve Bratko, 1995). Ayrıca TMP, bilimsel buluşların çeşitli aşamalarında; örneğin teori oluşturmada, deneysel tasarım ve teori doğrulamada bir araç olarak kullanılabilir (De Raedt, 1993).

4.2.3 Tümevarımlı mantık programlama teknikleri

Hipotez uzayında arama aşağıdan-yukarıya veya yukarıdan-aşağıya yönde gerçekleşebilir. Aşağıdan-yukarıya olacak şekilde hipotez uzayında arama genelleştirme (generalization) teknikleri ile yapılır. Genelleştirme teknikleri birkaç örnek ile yapılabilen etkileşimli ve değişken öğrenme için çok uygundur. Diğer taraftan özelleştirme (specialization) teknikleri, hipotez uzayını özelleşme operatörlerini kullanarak yukarıdan aşağıya doğru arar (Lavrac ve Dzeroski, 1994). Bu bölümde temel genelleştirme ve özelleştirme teknikleri anlatılacaktır.

4.2.3.1 Genelleştirme teknikleri

Genelleştirme, insanın günlük hayatta öğrenme sürecinde kullandığı temel bir iştir. Ancak bazı hatalardan kaçınmak için dikkatli olmak gerekir. Örneğin, şehirde yaşayan bir çocuğun bir serçeyi uçarken gördükten sonra, uçan bir güvercini de gördüğünü varsayalım. Sonuç olarak bu çocuk, kanadı olan her şeyin uçtuğunu düşünebilir. Kanadı olan her şeyin uçamayacağına inanabilmesi için, örneğin

devekuşunu negatif bir örnek olarak görmelidir. Genelde öğrenme için genelleştirme sürecini kısıtlamada kavrama ait negatif örneklere ihtiyaç duyulduğu sezgisel olarak görülmektedir. Ancak, Plotkin negatif örnekler olmadan sadece pozitif örneklerin lgg'lerini hesaplayarak genelleştirmenin kısıtlanabileceğini göstermiştir (Plotkin, 1970).

Genelleştirme teknikleri hipotez uzayını aşağıdan yukarıya doğru arar. Aşağıdan yukarıya öğreniciler verilen bir örneği kapsayan en genel belirli tümcecikten başlar ve başka negatif örnekleri kapsamaksızın genelleme yapana kadar tümceciği geneller. Aşağıdan yukarıya hipotez üretiminde c tümceciğinden c' üretimi θ-kapsama (θ-subsumption) tabanlı genelleştirme operatörü uygulanarak elde edilir.

θ-kapsama operatörü: Ancak ve ancak bir θ yerdeğiştirmesi varsa C_1 tümceciği C_2'ni θ-kapsar. Öyle ki; $C_1\theta \subseteq C_2$ olur. θ-kapsamayı gösteren bir örnek aşağıda verilmiştir:

$$C_1 = p(x, y)$$

$$C_2 = p(Ali, Ayşe)$$

$$\theta = \{x/\text{Ali}, y/\text{Ayşe}\}$$

Hipotez üretiminde önemli olan kapsama tabanlı genelleştirme teknikleri şunlardır (Bergadano ve Gunetti, 1996):

1. Plotkin'in en az genel genelleştirmesi
2. Ters çözümleme

I. Plotkin'in en az genel genelleştirmesi

TMP için en basit çıkarım modelidir. Plotkin'in lgg gösterimi TMP için önemlidir. Çünkü aşağıdan yukarıya θ-kapsama latis araması yapan genelleştirme algoritmalarının temeli olan bir formdur. Aşağıda tümcecikleri verilen bir genelleştirme örneğini ele alalım:

$$C_1: kanatları\_var(tweety) : -kuş(tweety).$$

$$C_2: kanatları\_var(birdy) : -kuş(birdy).$$

Eğer düşürme durum kuralını (dropping condition rule) ve sabitleri değişkenlere dönüştürme kuralını verilen tümceciklere uygularsak, aşağıdaki geneleme elde edilir.

C_3: $kanatları\_var(X)$.

Bu yüklem "herşeyin kanatları vardır" anlamına gelir ve bu da hatalı bir genelleme olur. Genelleştirme, eğer C_1 ve C_2 tümcecikleri doğru ise, benzer olarak $lgg(t_1, t_2)$'nin de doğru olduğunu varsayar. C_1 ve C_2 tümcecikleri için doğru lgg aşağıdaki gibidir:

C_4: $kanatları\_var(X) : -kuş(X)$.

Genelleştirme eğer C_1 ve C_2 tümcecikleri doğru ise, benzer olarak $lgg(t_1, t_2)$'nin de doğru olduğunu varsayar. İki tümceciğin lgg hesabı için terimlerin, atomların ve literallerin lgg'sinin öncelikle hesaplanması gerekir. Bu hesaplamalar ve ilgili örnekler (Lavrac ve Dzeroski, 1994)'den alıntıdır.

Terimlerin lgg'si $lgg(t_1, t_2)$

1. $lgg(t, t) = t$,
2. $lgg\big(f(s_1, \dots, s_n), f(t_1, \dots, t_n)\big) = f\big(lgg(s_1, t_1), \dots, lgg(s_n, t_n)\big)$,
3. $lgg(f(s_1, \dots, s_m), g(t_1, \dots t_n)) = V$, $f \neq g$ olduğunda ve *V* $lgg(f(s_1, \dots, s_m), g(t_1, \dots t_n))$'yi gösteren bir değişken olduğunda,
4. $lgg(s, t) = V$, $s \neq t$ ve s ve t'den en az biri bir değişken olduğunda; bu durumda V $lgg(s, t)$'yi gösteren bir değişkendir.

Terimlerin lgg'sinin hesaplanmasına ilişkin aşağıdaki örnekleri verebiliriz:

$lgg([a, b, c], [a, c, d]) = [a, X, Y]$.

$lgg\big(f(a, a), f(b, b)\big) = f\big(lgg(a, b), lgg(a, b)\big) = f(V, V)$. Burada *V*, $lgg(a, b)$'nin yerine geçmiştir.

Atomların lgg'si $\boldsymbol{lgg(A_1, A_2)}$

1. $lgg\big(p(s_1, \dots, s_n), p(t_1, \dots, t_n)\big) = p\big(lgg(s_1, t_1), \dots, lgg(s_n, t_n)\big)$, eğer atomlar aynı p yüklemine sahipse,
2. Eğer $p \neq q$ ise $lgg(p(s_1, \dots, s_m), q(t_1, \dots t_n))$ tanımsızdır.

Literallerin lgg'si $\boldsymbol{lgg(L_1, L_2)}$

1. Eğer L_1 ve L_2 atomlar ise, $lgg(L_1, L_2)$ yukarıdaki gibi hesaplanır.
2. L_1 ve L_2 negatif literaller ise, $L_1 = \overline{A_1}$ ve $L_2 = \overline{A_2}$, sonrasında

   $lgg(L_1, L_2) = lgg(\overline{A_1}, \overline{A_2}) = \overline{lgg(A_1, A_2)}$'dır,

3. Eğer L_1 pozitif ve L_2 negatif literal ise, $lgg(L_1, L_2)$ tanımsızdır.

Literallerin *lgg*'sinin hesaplanmasına ilişkin aşağıdaki örnekleri verebiliriz:

$$lgg\big(ebeveyn(ayşe, suna), ebeveyn(ayşe, onur)\big) = ebeveyn(ayşe, X).$$

$$lgg\big(ebeveyn(ayşe, suna), \overline{ebeveyn(ayşe, onur)}\big) = tanımsız$$

$$lgg(ebevey(ayşe, X), kız\_kardeş(suna, ayşe)) = tanımsız$$

Tümceciklerin lgg'si $\boldsymbol{lgg(c_1, c_2)}$

$c_1 = \{L_1, \dots, L_n\}$ ve $c_2 = \{K_1, \dots, K_m\}$ varsayalım. Sonrasında $lgg(c_1, c_2) = \{L_{i,j} = lgg(L_i, K_j) | L_i \in c_1, K_j \in c_2$ ve $lgg(L_i, K_j)$ tanımlanmıştır$\}$ olur.

Lgg gösterimi ile ilgili yukarıda verilen tanımlamaları yaptıktan sonra, şimdi *rlgg* gösterimi ile ilgili bilgi verebiliriz. Verilen bilgiler ve örnekler (Lavrac ve Dzeroski, 1994) ve (Bergadano ve Gunetti, 1996)'dan alınmıştır.

c_1 ve $c_2$tümceciklerinin rlgg'leri artalan bilgisi *B* ile $lgg(c_1, c_2)$'dir. Artalan bilgisi taban (ground) gerçekler içeriyorsa ve K bu gerçekler arasındaki bağlantıyı gösterirse, A_1 ve A_2 iki taban atomun (pozitif örnekler) *rlgg*'si artalan bilgisi *K* ile ilgili olarak aşağıdaki gibi tanımlanır:

$$rlgg(A_1, A_2) = lgg((A_1 \leftarrow K), (A_2 \leftarrow K))$$

Rlgg gösterimine duyulan ihtiyacın nasıl doğduğunu bir örnek üzerinden anlatalım. Bazen basit bir tümcecik kümesinin *lgg*'si TMP probleminde faydalı olmayabilir. Örneğin, aşağıdaki iki tümceciği ele alalım:

$$c_1 = amca(X,Y) :- erkek\_kardeş\big(X, baba(Y)\big).$$

$$c_2 = amca(X,Y) :- erkek\_kardeş\big(X, anne(Y)\big).$$

İlgili *lgg*:

$$lgg(c_1, c_2) = amca(X,Y) :- erkek\_kardeş(X,Z).$$

Buradaki *amca* ilişkisi tanımı yeterince bilgilendirici değildir. Gerçekte, bir artalan bilgisi veya teori için bir ilişkideki bir örnek kümesinin genelleştirmesini bulmak ile ilgilenilmektedir. Yani, yukarıdaki örnekte aynı zamanda *ebeveyn(baba(X),X))*'in ve *ebeveyn(anne(X),X))* artalan bilgilerinin doğru olduğunu da bilirsek, c_1 ve c_2 ilişkisi için lgg şu hale gelir:

$$c = amca(X,Y) :- erkek\_kardeş(X,U), ebeveyn(U,Y).$$

GOLEM sistemi *rlgg* gösterimini kullanan tek TMP sistemidir (Muggleton ve Feng, 1990). P gibi bir mantık programı (örneğin bir artalan bilgisi) ve $P \nvdash E_1$ile $P \nvdash E_2$ olan E_1 ve E_2 örnekleri verilsin. Bu P programı ile ilgili olarak E_1 ve E_2'nin lgg C'si bulmak istensin. Aranan C tümceciği $P \wedge C \vdash E_1 \wedge E_2$'dir ve C, E_1 ve E_2'nin her ikisinden birden bir defada oluşturulur. GOLEM sistemi aşağıdaki sıra ile ilerler:

$$P \wedge C \vdash E_1$$

$$C \vdash P \rightarrow E_1$$

$$\vdash C \rightarrow (\neg P \vee E_1)$$

$$\vdash C \rightarrow ((\neg p_1 \vee \neg p_2 \vee \ldots) \vee E_1)$$

Son adımda P, P'nin bir modeli olarak gösterilen $p_1, p_2, \ldots$ atomları ile yer değiştirir. Aynı adımlar E_2 için de uygulanır. Yerlerine koyarsak,

$C_1 = ((\neg p_1 \vee \neg p_2 \vee \ldots) \vee E_1)$

$C_2 = ((\neg p_1 \vee \neg p_2 \vee \ldots) \vee E_2)$

$\vdash C \rightarrow lgg(C_1, C_2)$ elde edilir. Bu $C = lgg(C_1, C_2)$ ile elde edilir.

Örnek olarak aşağıdaki P mantık programının olduğunu varsayalım:

$A: kuş(tweety).$

$B: kuş(birdy).$

ve iki örneğimiz,

$E_1: kanatları\_var(tweety).$

$E_2: kanatları\_var(birdy).$

olsun.

A ve B atomları P programı için taban modeldir. Yukarıdaki mantıktan yola çıkarak şunları elde ederiz:

$C_1 = kanatlar\_var(tweety) :- kuş(tweety), kuş(birdy).$

$C_2 = kanatlar\_var(birdy) :- kuş(tweety), kuş(birdy).$

Böylece C_1 ve C_2'nin lgg'si şu hale gelmektedir:

$lgg(C_1, C_2) = kanatlar\_var(X) :- kuş(tweety), kuş(birdy), kuş(X).$

Gereksiz literalleri kaldırısak, P'ye uyumlu olarak E_1 ve E_2 örneklerinin lgg'si,

$C = kanatları\_var(X) :- kuş(X).$

olur.

GOLEM klasik mantık programlarını pozitif ve negatif örnekleri rasgele seçerek birkaç saniyede öğrenebilmektedir. GOLEM ile bazı alanlarda başarılı sonuçlar elde edilmiştir. Bunlardan bazıları: proteinlerin aminoasit zincirindeki ikincil yapıların tahmini için kurallar öğrenme, ilaç tasarımında yapı-etkinlik ilişkilerini tahmin etme ve fiziksel sistemler için geçici tanılama kurallarını öğrenmedir.

II. Ters çözümleme

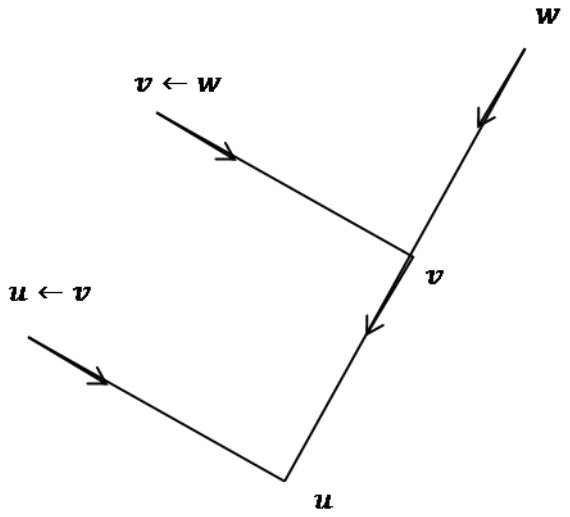
Tümevarım, tümdengelimin tersi olarak düşünülebilir. Şöyle ki, tümdengelimde özel durumlardan genel kurallar çıkarılırken, tümevarımda özel durumlardan genel kurallar elde edilmektedir. Yani, tümevarım çıkarım kuralları tümdengelim çıkarım kurallarının tersi olarak elde edilebilir. Bu düşünceyle, (Muggleton ve Buntine, 1988) *ters çözümlemeyi,* (Robinson, 1965) tarafından bulunan bir tümdengelimli çıkarım kuralı olan *çözümlemenin (resolotion)* tersi olarak sunmuşlardır. Ters çözümleme, günümüzde aşağıdan yukarıya TMP'de öne çıkan bir genelleştirme operatörüdür.

Şimdi, sırasıyla çözümleme ve ters çözümlemeyi (Lavrac ve Dzeroski, 1994)'den aldığımız farklı örnekler üzerinden açıklayacağız.

En basit çözümleme adımı, önermeler mantığındaki $p \vee \bar{q}$ ve $q \vee r$ öncüllerinden $p \vee r$ çözümünün çıkarılmasıdır. Bunun için bir önerme türetme ağacı örneğine bakalım.

Bir önerme türetme ağacı örneği:

Verilen $\daleth = \{u \leftarrow v, v \leftarrow w, w\}$ teorisinden u çıkarılmak istensin. w önermesi, v'yi veren $v \leftarrow w$ ile çözülür. Sonrasında u da $u \leftarrow v$ ile çözülür. Türetme ağacı Şekil 4.6'daki gibidir:



Şekil 4.6 Basit bir önerme türetme ağacı

Bu örnek önermeler durumundaki çözümdür. Birinci dereceden bir durumda çözümleme daha karmaşıktır, çünkü yerdeğiştirme (substitution) gerekir. Birinci dereceden bir türetme ağacı örneği aşağıda verilmiştir:

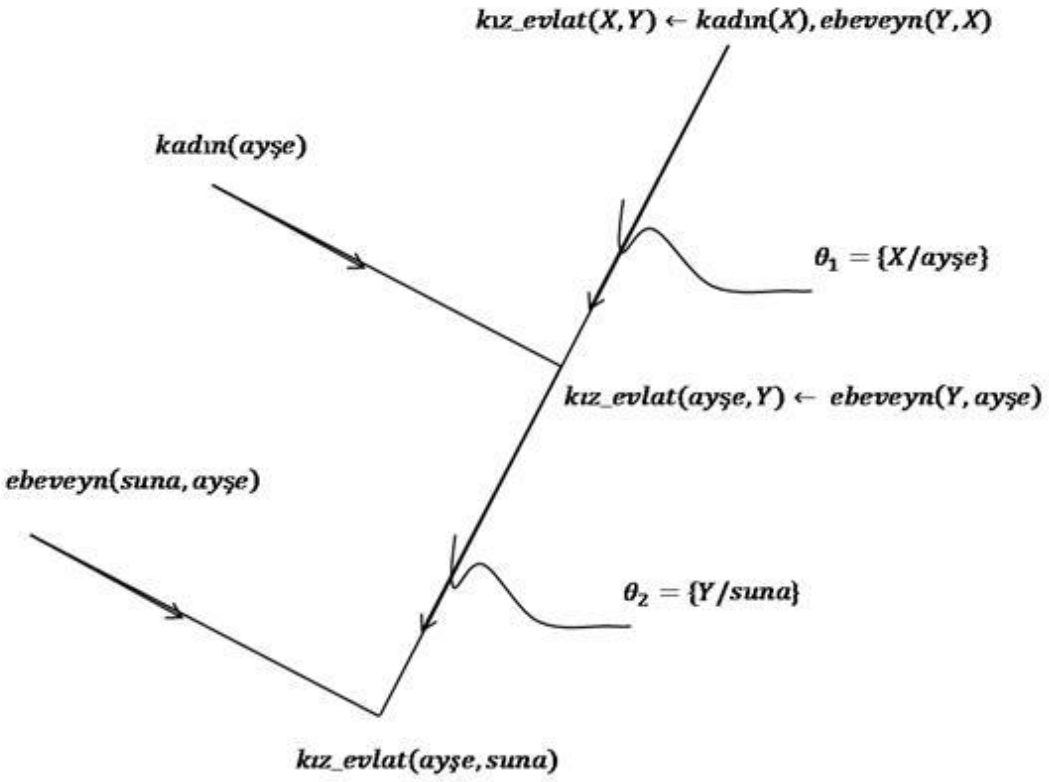
Birinci dereceden türetme ağacı örneği:

Birinci dereceden mantıktaki çözümlemeyi göstermek için bir aile örneğini ele alalım. Artalan bilgisi B'de, $b_1 = kadın(ayşe)$ ve $b_2 = ebeveyn(suna, ayşe)$ ve $H = \{c\} = kız\_evlat(X,Y) \leftarrow kadın(X), ebeveyn(Y,X)\}$. verilsin. $ד = H \cup B$ olsun. $e = kız\_evlat(ayşe, suna)$ gerçeğini ד'den elde etmek için aşağıdaki sırada ilerlenir.

- Öncelikle, $c_1 = çözüm(c, b_1)$ çözeni, $\theta_1 = \{X/ayşe\}$ yerdeğiştirmesi ile hesaplanır. θ_1 yerdeğiştirmesi $kız\_evlat(ayşe, Y) \leftarrow kadın(ayşe),\ ebeveyn(Y, ayşe)$ elde etmek için c tümceciğine uygulanır. Sonrasında bu çözen de önermeler durumu olarak b_1 ile çözülür. $kız\_evlat(X,Y) \leftarrow kadın(X),\ ebeveyn(Y,X)$ ve $kadın(ayşe)$ çözenleri böylece $c_1 = çözüm(c, b_1) = kız\_evlat(ayşe, Y) \leftarrow ebeveyn(Y, ayşe)$ olur.

- Sonraki çözen $c_2 = \text{çözüm}(c, b_2)$, $\theta_2 = \{Y/suna\}$ yerdeğiştirmesi ile hesaplanır. $kız\_evlat(ayşe, Y) \leftarrow ebeveyn(Y, ayşe)$ ve $ebeveyn(suna, ayşe)$ tümcecikleri $e = res(c_1, b_2) = kız\_evlat(ayşe, suna)$ ile çözülür.

Bu örneğe ait türetme ağacı Şekil 4.7'de verilmiştir.



Şekil 4.7 Birinci dereceden doğrusal türetme ağacı

Ters çözümleme, etkileşimli bir TMP sistemi olan CIGOL'da kullanılmıştır (Muggleton ve Buntine, 1988). Ters çözümleme sürecinde, çözümlemenin tersini gerçekleştirmek amacıyla ters yerdeğiştirmeyi (Buntine, 1988) temel alan genelleştirme operatörü kullanılır. Şimdi basit bir ters yerdeğiştirme örneğine bakalım:

Basit bir ters yerdeğiştirme örneği:

$c = kız\_evlat(X, Y) \leftarrow kadın(X), ebeveyn(Y, X)$ ve yerdeğiştirme

$\theta = \{X/ayşe, Y/suna\}$ olduğunda;

$c' = c\theta = kız\_evlat(ayşe, suna) \leftarrow kadın(ayşe), ebeveyn(suna, ayşe)$

olur.

Ters yerdeğiştirme $\theta^{-1} = \{ayşe/X, suna/Y\}$ uygulandığında asıl c tümceciği elde edilir:

$c = c'\theta^{-1} = kız\_evlat(X, Y) \leftarrow kadın(X), ebeveyn(Y, X)$

Yerler ile bir yerdeğiştirme örneği:

$W = sever(X, kız\_evlat(Y))$ ve $\theta = \{X/suna, Y/suna\}$ verilsin. Yerdeğiştirme, W'ye uygulandığında $W\theta = sever(suna, kız\_evlat(suna))$ olur. Ters yerdeğiştirme $\theta^{-1} = \{(suna, \{< 1 >\})/X, (suna\{< 2,1 >)/Y\}\}$, $W\theta$ teriminin X değişkeni ile yerdeğiştiren $suna$ alt teriminin ilk oluşu ($yer < 1 >$'de) ve Y ile yerdeğiştirmesindeki ikinci oluşu ($yer < 2,1 >$'de) belirler. θ^{-1}'de yerlerin kullanımı $W\theta\theta^{-1} = sever(X, kız\_evlat(Y)) = W$ sağlar.

Ters çözümlemeye bir örnek üzerinden bakalım. Burada $ters\_çözüm(c, d)$, c ve d tümceciklerinin ters çözümünü göstersin.

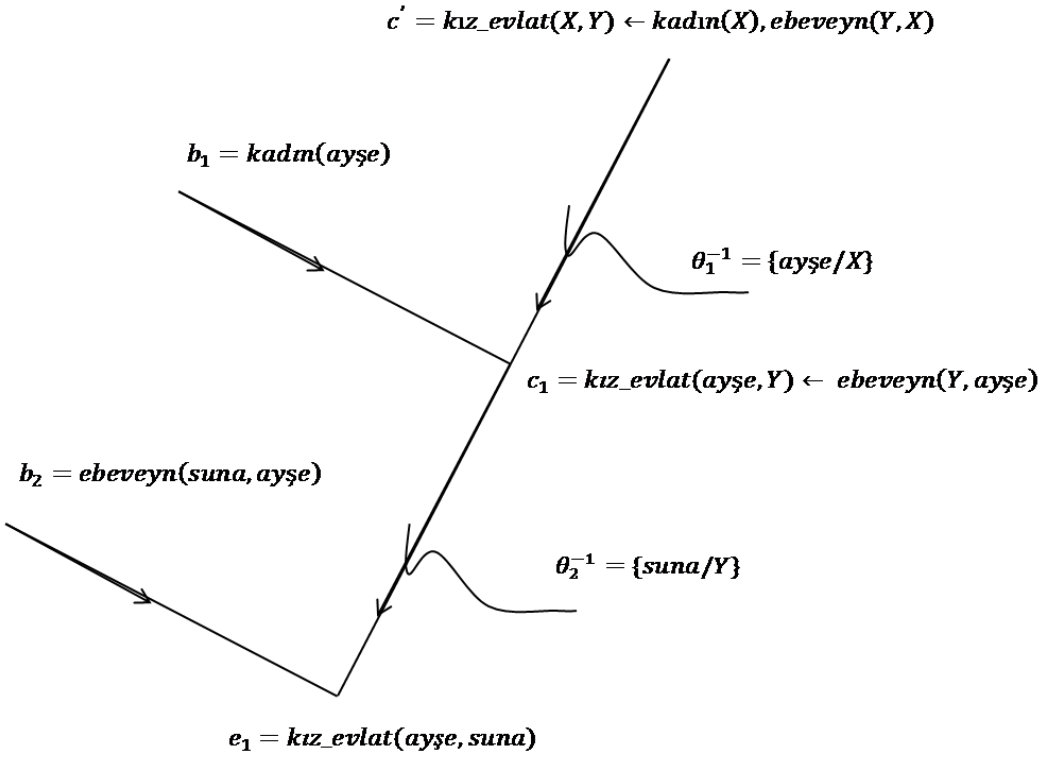
Ters çözümleme örneği:

Yukarıdaki örnekteki gibi B artalan bilgisi $b_1 = kadın(ayşe)$ ve $b_2 = evebeyn(suna, ayşe)$ tümceciklerini içersin. Geçerli hipotez $\mathcal{H} = \theta$ olsun. Öğrenicinin pozitif örnek $e_1 = kız\_evlat(ayşe, suna)$ ile karşılaştığını varsayalım. Bu durumda ters çözümleme süreci şu şekilde takip edilir:

- İlk adımda ters çözümleme b_2 ile birlikte e_1'i gerektiren ve geçerli $\mathcal{H}$ hipotezine e_1'in yerine eklenebilecek c_1 tümceciğini bulmaya çalışır. Ters yerdeğiştirme

$\theta_2^{-1} = \{suna/Y\}$ seçimi, bir ters çözümleme adımı $c_1 = ters\_çözüm(b_2, e_1) = kız\_evlat(ayşe, Y) \leftarrow ebeveyn(Y, ayşe)$ tümceciğini üretir. c_1 tümceciği geçerli hipotez $\mathcal{H}$olur, öyle ki $\{b_2\} \cup \mathcal{H} \vDash e_1$.

- Ters çözümleme sonrasında $b_1 = kadın(ayşe)$. ve geçerli hipotez $\mathcal{H} = \{c_1\} = \{kız\_evlat(ayşe, Y) \leftarrow ebeveyn(Y, ayşe)\}$ 'yi alır. $c' = ters\_çözüm(b_1, e_1)$ hesaplanarak, $\theta_1^{-1} = \{ayşe/X\}$ ters yerdeğiştirmesi kullanılarak, bu B artalan bilgisine dayanarak ve $c' = kız\_evlat(X, Y) \leftarrow kadın(X), ebeveyn(Y, X)$ verir. $\mathcal{H}$ geçerli hipotezinde c_1 tümceciği daha genel c' tümceciği ile yerdeğiştirebilir. Böylece çıkarılan hipotez $\mathcal{H} = kız\_evlat(X, Y) \leftarrow kadın(X), ebeveyn(Y, X)$ olur. İlgili ters doğrusal türetme ağacı Şekil 4.8'dedir.



Şekil 4.8 Ters doğrusal türetme ağacı

Ters çözümleme adımlarında kullanılan iki önemli operatör bulunmaktadır: V- ve W-operatörleri. C_1 ve R verildiğinde V-operatörü, R'nin C_1 ve C_2'nin

çözümlenmesinden oluşan bir örnek olacağı şekilde bir C_2 bulur. Böylece V-operatörü $\{C_1, R\}$'yi $\{C_1,C_2\}$'ye geneller. W-operatörü iki ayrı V-operatörünün birleşiminden oluşur. Eğer R_1, C_1 ve C_2'nin çözümlenmesinden oluşan bir örnek ve R_2, C_2 ve C_3'ün çözümlemesinde oluşan bir örnek ise $\{R_1,R_2\}$'yi $\{C_1,C_2,C_3\}$'e geneller,. W-operatörü ile yeni yüklemler bulunabilir.

Ters çözümlemede tanımlanan dört tümevarımlı çıkarım kuralı vardır: içerilme (absorption), özdeşleşme (identification), iç-yapılanma (intra-construction) ve ara-yapılanma (inter-construction). İçerilme ve özdeşleşme operatörleri tek adımlık çözümleme ile tersine çevirme işlemini gerçekleştirir. İki işlem ortaklaşa V-operatörleri olarak isimlendirilir. İç-yapılanma ve ara-yapılanma kuralları, yeni bir yüklem sembolü tanımlar. Yeni yüklemleri tanımlayan tümevarımlı çıkarım kurallarının *yüklem oluşturma (predicate invention)* gerçekleştirdiği varsayılır. İç-yapılanma ve ara-yapılanma operatörleri, W-operatörleri olarak adlandırılır. Şimdi V ve W-operatörlerini inceleleyelim. Bu operatörler için verilen tanımlarda küçük harfler önerme değişkenlerini, büyük harfler önerme değişkenlerini bağlayıcıları (conjunctions) göstermektedir.

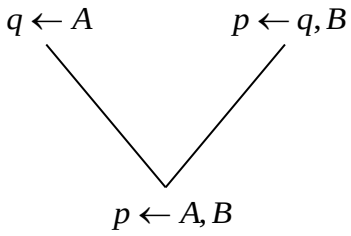
A. V-operatörleri

İçerilme ve özdeşleşme operatörleri, V-operatörleri olarak isimlendirilir.

İçerilme: Eğer $q \leftarrow A$ ve $p \leftarrow A, B$ tümceleri varsa, aşağıdaki oluşturma kuralı uygulanabilir. Verilen tümceler çizginin üstünde iken, çıkarım çizginin altında gösterilmiştir.

$$\frac{q \leftarrow A \ \& \ p \leftarrow A, B}{q \leftarrow A \ \& \ p \leftarrow q, B}$$

Burada yeni bir hipotezin ortaya çıktığını görüyoruz. A burada q ile değiştirilmiştir. Tümdengelimi göstermek için yukarıdaki çıkarımı baş aşağı çevirerek Şekil 4.9'daki V diyagramını yazabiliriz.

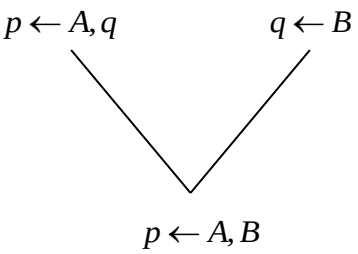


Şekil 4.9 İçerilme için V diyagramı gösterimi

Özdeşleşme: Eğer $p \leftarrow A, B$ ve $p \leftarrow A, q$ tümceleri varsa, $q \leftarrow B$ tümcesi ile açıklanabilen hipotez kurulabilir. Ters çözümleme için aşağıdaki yolu takip edebiliriz.

$$\frac{p \leftarrow A, B \ \& \ p \leftarrow A, q}{q \leftarrow B \ \& \ p \leftarrow A, q}$$

Bu ters çözümlemeyi Şekil 4.10'daki V diyagramı ile gösterebiliriz.



Şekil 4.10 Özdeşleşme için V diyagramı gösterimi

İçerilme ve özdeşleşme kuralları tek çözümleme adımında ters çevrilir.

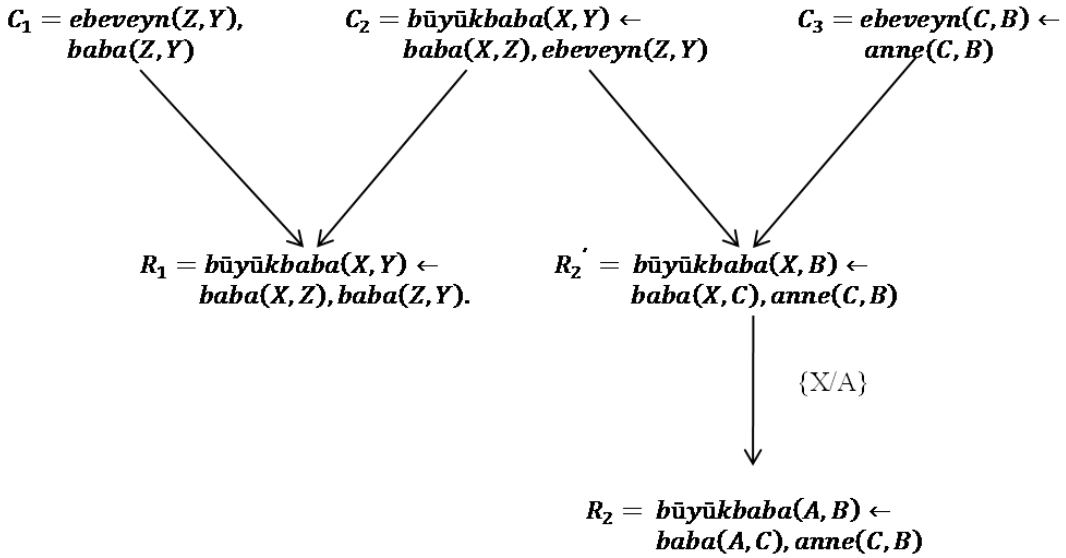
B. W-operatörleri

Yüklem oluşturma, yeni terimler tanımlayarak sistemin kendi sözlüğünü zenginleştirmesi yeteneğidir. Yeni bir terim, eğer oluşumu ve kullanımı teorinin öğrenebilirliğini etkilemiyorsa faydalı olarak sınıflandırılabilir. Yeni bir terim

olmadığında teori başarılı öğrenme kriterine göre öğrenebilir olamıyorsa terim gereklidir.

Yüklem oluşturma genellikle karmaşık teorilerin oluşturulmasında kullanılır. W-operatörleri ile formalize edilir. İç-yapılanma ve ara-yapılanma W-operatörleri olarak adlandırılır.

W-operatörünün arkasındaki düşünceyi kavramak amacıyla bir örnek verelim. İki tane Horn tümceciği $R_1 = büyükbaba(X,Y) :- baba(X,Z), baba(Z,Y)$ ve $R_2 = büyükbaba(A,B) :- baba(A,C), anne(C,B)$ bulunsun. Bu tümcecikler genelleştirilmek istensin. Bu durumda W-operatörü C_1, C_2 ve C_3 tümceciklerini oluşturur, öyle ki R_1, C_1 ve C_2'nin bir çözen örneğidir; R_2 de C_2 ve C_3'ün çözen örneğidir. Böylece W-operatörü $\{R_1, R_2\}$'yi $\{C_1, C_2$ ve $C_3\}$'e geneller. Bunu için olası bir çözüm Şekil 4.11'de görülmektedir.



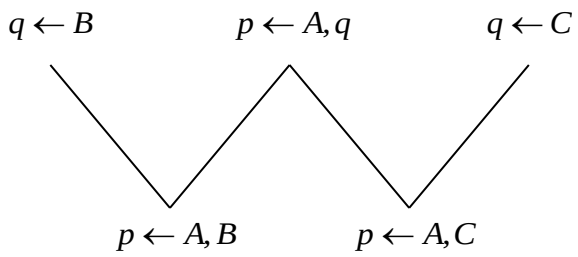
Şekil 4.11 $\{R_1, R_2\}$'nin $\{C_1, C_2$ ve $C_3\}$'e W-operatörü ile genelleştirilmesi

Şekilde görüldüğü üzere çözümü sağlamak için yeni bir yüklem olarak *ebeveyn* yüklemi oluşturulmuştur.

İç-yapılanma: Burada yeni bir sembol q vardır ve q yeni bir yüklemdir. Bu sürece yüklem oluşturma denir. *Eklemeli sıralamalı (Insertion sort)* gibi mantık programları oluşturulduğunda, CIGOL yeni bir *ekle (insert)* yüklemi oluşturmak için iç-yapılanma operatörünü kullanır (Muggleton ve Buntine, 1988). Yeni yüklem sonradan bir V-operatörü kullanılarak genelleştirilebilir.

$$\frac{p \leftarrow A,B \;\&\; p \leftarrow A,C}{q \leftarrow B \;\&\; p \leftarrow A,q \;\&\; q \leftarrow C}$$

Bu ters çözümlemeyi Şekil 4.12'deki W diyagramı ile gösterebiliriz.

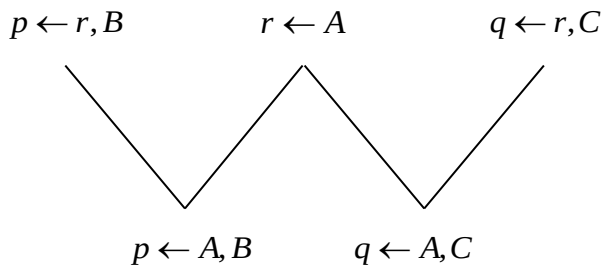


Şekil 4.12 İç-yapılanma için bir W diyagramı gösterimi

Ara-yapılanma: Başlangıç sembollerinin farklı yüklemler olması durumudur.

$$\frac{p \leftarrow A,B \quad \& \; q \leftarrow A,C}{p \leftarrow r,B \;\&\; r \leftarrow A \;\&\; q \leftarrow r,C}$$

Bunu W-diyagramı ile Şekil 4.13'deki gibi gösterebiliriz.



Şekil 4.13 Ara-yapılanma için bir W-diyagramı gösterimi

4.2.3.2 Özelleştirme teknikleri

Özelleştirme teknikleri, θ-kapsama tabanlı özelleştirme operatörleri kullanarak hipotez uzayını yukarıdan aşağıya tarar. Bir özelleştirme operatörüne genelde saflaştırma operatörü (*refinement operator)* denir.

Verilen bir L dilinde bir saflaştırma operatörü, p c'nin özelleştiği bir p(c) tümcecikleri kümesi için bir c tümceciğini kapsar. Bir saflaştırma operatörü tipik olarak sadece θ-kapsama altındaki bir tümceciğin minimum (en genel) özelleşmesini hesaplar. Bir tümcecik üzerinde iki temel sözdizimsel işleme başvurur.

- Bir tümceciğe bir yer değiştirme (substitution) uygulama
- Tümceciğin gövdesine bir literal uygulama.

Temel özelleştirme teknikleri hipotez uzayında yukarıdan aşağıya arama yapar. Yukarıdan-aşağıya öğreniciler en genel tümcecikten başlar ve negatif örnekleri kapsamayıncaya kadar tekrarlı şekilde saflaştırmaya devam eder. Tümceciklerin en az bir pozitif örneği kapsadığına emin oluncaya kadar arama sürer.

En bilinen iki öğrenme sistemi Model Çıkarım Sistemi (Model Inference System - MIS) ve Birinci Dereceden Tümevarımlı Öğrenici (First Order Clausel Learner –- FOIL) Sistemidir. Şimdi sırasıyla bu sistemlerden bahsedelim.

I. Model çıkarım sistemi

Ehud Shapiro 1983 yılında MIS sistemini geliştirmiştir (Shapiro, 1983). Bu sistem, mantık programlama sağlam bir disiplin olduktan ve çoğu insan bir programlama dili olarak Prolog ile ilgilenmeye başladıktan sonra mantık programlarının tümevarımlı sentezi için ilk belirgin yaklaşım olarak sunulmuştur. Şekil 4.14'de temel MIS algoritması verilmiştir:

Hipotez $H \leftarrow \emptyset$
loop
  Sonraki örneği işle
  while H tam veya tutarlı olmadığı sürece **do**
    if H bir negatif örnek e'yi kapsıyorsa **then**
      H'nin e'yi kapsayan tümceciklerini sil.
    end if
    if H'nin kapsamadığı bir pozitif örnek varsa **then**
      refinement grafiği boyunca derinlik-öncelikli bir arama ile e'yi kapsayan bir c tümceciği oluştur.
      H'ye bir c tümceciği ekle.
    end if
  end while
end loop

Şekil 4.14 MIS Algoritması

MIS sistemini aşağıdaki aile örneğini kullanarak açıklayalım:

Örnek:

$$B = \begin{cases} ebeveyn(ayşe, elif) \leftarrow \\ ebeveyn(ayşe, onur) \leftarrow \\ ebeveyn(onur, burcu) \leftarrow \\ ebeveyn(onur, ali) \leftarrow \\ kız(ayşe) \leftarrow \\ kız(elif) \leftarrow \\ kız(burcu) \leftarrow \\ erkek(onur) \leftarrow \end{cases}$$

$$E^+ = \begin{cases} kız\_evlat(elif, ayşe) \leftarrow \\ kız\_evlat(burcu, onur) \leftarrow \end{cases}$$

$$E^- = \begin{cases} kız\_evlat(onur, ayşe) \leftarrow \\ kız\_evlat(burcu, ayşe) \leftarrow \end{cases}$$

MIS etkileşimli bir sistem olduğu için örnekler sırasıyla işlenir. Hipotez kümesi başlangıçta boş tümcecik içerir. Bu örnekte sisteme önce pozitif örneklerin verildiğini varsayalım. İlk örnek $e_1 = kız\_evlat(elif, ayşe)$ işlendiğinde, *kız_evlat* yükleminin en genel tanımı olarak: $kız\_evlat(X, Y) \leftarrow$ eklenir. Bu durumda hipotez e_1 örneğini kapsayan tek bir tümcecik içerir:

$H = \{c\} = \{kız\_evlat(X, Y) \leftarrow \}$.

Sonrasında ikinci örnek olarak $e_2 = kız\_evlat(burcu, onur)$ verildiğinde bu örneğin c tümceciğini kapsadığı görülür. Sonraki negatif örnek $e_3 = kız\_evlat(onur, ayşe)$ işlenir. Bu negatif örnek de c tümceciğini kapsamaktadır. Bu durumda tümceciğin gövdesine bir literal eklenmesi gerekir. Eklenebilecek iki literal bulunmaktadır.

1. Tümceciğin baş kısmında olan değişkenlere sahip literallerdir.

$X = Y, kız(X), kız(Y), ebeveyn(X, X), ebeveyn(Y, Y), evebeyn(X, Y), ebeveyn(Y, X)$

2. Yeni değişkenler içeren literallerdir:

$ebeveyn(X, Z), ebeveyn(Z, X), ebeveyn(Y, Z), ebeveyn(Z, Y)$.

Öncelikle ilk bağımlı $X = Y$ denenir, ancak $kız\_evlat(X, Y) \leftarrow X = Y$ örneklerden hiçbirini kapsamadığı için bu literal elenir. Sonra ikinci literal denendiğinde $kız\_evlat(X, Y) \leftarrow kız(X)$ tümceciği elde edilir. Bu tümcecik $e_1$ve e_2 örneklerini kapsarken $e_3$örneğini kapsamaz. Bu durumda durulur ve elde edilen hipotez:

$H = \{c\} = \{kız\_evlat(X, Y) \leftarrow kız(X)\}$ olur.

Sonraki aşamada en dıştaki döngüye gidilir ve negatif örnek $e_4 = kız\_evlat(burcu, ayşe)$. işlenir. c tümceciği e_4 örneğini kapsadığı için bu hipotezden silinir ve arama önceki adımdan pozitif örnekleri kapsayacak şekilde tekrar başlatılır. $kız\_evlat(X, Y) \leftarrow$ saflaştırmalarından hiçbiri örneklerden ayrı tutulmaz, bu yüzden kendi çocuklarının saflaştırmaları da düşünülür. İlk olarak $kız\_evlat(X, Y) \leftarrow X = Y$ saflaştırması denenir. Ama hiçbirinin e_1 örneğini kapsamadığı görülür ve

bundan dolayı vazgeçilir. İkinci olarak $kız\_evlat(X,Y) \leftarrow kız(X)$ saflaştırması denenir Bulunan $kız\_evlat(X,Y) \leftarrow kız(X), ebeveyn(X,Y)$ tümceciği verilen örnek kümesine göre tam ve tutarlıdır ve hipoteze konur. Sonuç olarak kavramı doğru olarak tanımlayan hipotez:

$H = \{kız\_evlat(X,Y) \leftarrow kız(X), ebeveyn(X,Y)\}$ olur.

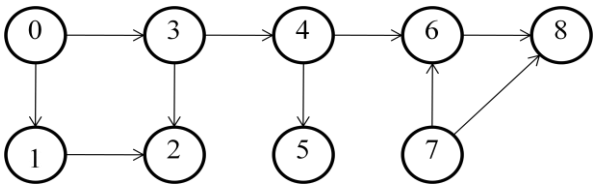
II. Birinci dereceden tümevarımlı öğrenici

FOIL öğrenme sistemi yaygın olarak TMP topluluğu ve makine öğrenmesi araştırmacıları tarafından kullanılan bir sistemdir (Quinlan, 1990). İçeriğini sezgisel (heurictic) aramadan alır.

Diğer yukarıdan aşağıya yaklaşımlar gibi FOIL de üç adımda işler:

1. Örnek kümesinin ön işlemesi.
2. Hipotez oluşturulması.
3. Hipotezin son işlemesi.

Eğer verilmediyse negatif örnekler ilk adımda oluşturulur. Yüklemi aynı olan birkaç tümcecikten oluşan hipotez temel kapsama algoritmasıyla oluşturulur. Son adımda gürültünün sebep olduğu hata artışı giderilir. Bu sistemi daha ayrıntılı açıklamak amacıyla (Bergadano ve Gunetti, 1996)'dan alınan bir örneğe ve bu örneğe ait Şekil 4.15'deki grafiğe bakalım:



Şekil 4.15 Bir yönlü grafik

Bu grafik *bağlantı* denilen bir ilişkinin parçası olan tüm oklar listenerek sembolik bir formda da gösterilebilir. Eğer n düğümünden m düğümüne giden yönde bir ok varsa

$< n, m >$ değişkenler grubu *bağlantı* ilişkisine aittir. Bu şekilde aşağıdaki sembolik tanımlama elde edilir:

$$bağlantı : - < 0{,}1 >, < 0{,}3 >, < 1{,}2 >, < 3{,}2 >, < 3{,}4 >, < 4{,}5 >, < 4{,}6 >, < 6{,}8 >, < 7{,}6 >, < 7{,}8 >.$$

Bu *bağlantı* ilişkisi bağlantı(n, m) tipinde Prolog gerçeklerine çevrildiğinde artalan bilgisinin basit bir formuna dönüşmektedir. Bundan sonra FOIL ile bu grafikten ulaşılabilecek kavramı öğrenme yapılabilir. Örneğin *ulaşabilir* ilişki için aşağıdaki pozitif ve negatif örnekler verilsin:

E^+ $ulaşabilir(0{,}1), ulaşabilir(0{,}2), ulaşabilir(0{,}3), \ldots (19\ örnek)$

E^- $ulaşabilir(0{,}0), ulaşabilir(0{,}7), ulaşabilir(1{,}0), \ldots (62\ örnek)$

FOIL *ulaşabilir* ilişkisi için aşağıdaki doğru Prolog tanımını üretebilir.

$ulaşabilir(X, Y) : - bağlantı(X, Y).$

$ulaşabilir(X, Y) : - bağlantı(X, Z), ulaşabilir(Z, Y).$

Aşağıda bu tanımın nasıl elde edildiği görünmektedir. Öncelikle ilk tümcecik ile başlanır. Tümceciğin başta gövdesi boştur ve ilk literal eklenmesi gerekir. Bu durum aşağıda şekilde gösterilebilir.

$ulaşabilir(X, Y) : - ?$

FOIL sadece örneklerde ve artalan bilgisinde olan yüklemleri ele alır. Eklenebilecek literaller aşağıdaki listede bazı ilişki bilgileri ile verilmiştir.

$bağlantı(X, Y)$	10 pozitif, 0 negatif örneği kapsar
$bağlantı(Y, X)$	0 pozitif, 10 negatif örneği kapsar
$bağlantı(X, Z)$	19 pozitif, 35 negatif örneği kapsar
$bağlantı(Z, Y)$	19 pozitif, 45 negatif örneği kapsar

$bağlantı(Z, W)$ $veya$ $ulaşabilir(Z, W)$ izinsiz(eski değişkenler dışında)

$ulaşabilir(X, Y)veya$ $ulaşabilir(Y, X)$ izinsiz (bitirme garantisi yok)

$ulaşabilir(X, Z)veya$ $ulaşabilir(Z, Y)$ izinsiz (bitirme garantisi yok)

Genellikle FOIL şu tipteki özyinelemeli tümceciklere izin verir:

$p(X_1, \dots, X_i, \dots, X_n) : - \cdots, q(X_i, Y), \dots, p(X_1, \dots, Y, \dots, X_n), \dots$

Sonraki aşamada tümceciğin gövdesine eklenecek literal seçilmelidir. Burada, ilk literal olan $bağlantı(X, Y)$ en iyi seçim gibi görünmektedir. Gerçekten, elde edilen ilk tümcecik 10 pozitif, 0 negatif örneği kapsamakta ve bu sebeple tutarlı ve faydalı bir ulaşabilme tanımı olmaktadır. $ulaşabilir(X, Y) : - bağlantı(X, Y)$ öğrenilen programa eklenir; bu ne kaldırılır ne de tekrar düşünülür. 10 kapsayan pozitif örnek kaldırılır. FOIL bundan sonra ikinci tümceciğin oluşması için devam eder. Tekrar tümcecik gövdesi başlangıçta boştur. Olası literaller yukarıdaki gibi tekrar listelenir. Ancak bu liste değişmiştir, çünkü kapsanan pozitif örnek sayısı değişmiştir:

$bağlantı(X, Y)$ 0 pozitif, 0 negatif örneği kapsar

$bağlantı(Y, X)$ 0 pozitif, 10 negatif örneği kapsar

$bağlantı(X, Z)$ 9 pozitif, 35 negatif örneği kapsar

$bağlantı(Z, Y)$ 9 pozitif, 45 negatif örneği kapsar

$bağlantı(Z, W)$ $veya$ $ulaşabilir(Z, W)$ izinsiz(eski değişkenler dışında)

$ulaşabilir(X, Y)veya$ $ulaşabilir(Y, X)$ izinsiz (bitirme garantisi yok)

$ulaşabilir(X, Z)veya$ $ulaşabilir(Z, Y)$ izinsiz (bitirme garantisi yok)

İlk iki literal kapsadığı pozitif örnek olmadığı için ele alınmaz. Üçüncü literal dördüncüden daha iyidir, çünkü daha az sayıda negatif örneği kapsamaktadır.

Sonrasında üçüncü literal ata tümceciğe eklenir, ancak işlem bitmez; kapsanan 35 negatif örneğin çıkarılması gerekmektedir. Oluşan durum şudur:

$ulaşabilir(X, Y) : -bağlantı(X, Z), ?$

Soru işareti yerine yeni bir literal konması gerekir ve olası seçenekler şunlardır:

$bağlantı(X, Y)$	0 pozitif, 0 negatif örneği kapsar
$bağlantı(Y, X)$	0 pozitif, 10 negatif örneği kapsar
$bağlantı(X, Z)$	izinsiz
$bağlantı(Z, Y)$	5 pozitif, 0 negatif örneği kapsar
$bağlantı(W, Y)$	9 pozitif, 24 negatif örneği kapsar
$bağlantı(Z, W)$	9 pozitif, 19 negatif örneği kapsar
$bağlantı(W, Z), bağlantı(X, W)$	9 pozitif, 19 negatif örneği kapsar
$bağlantı(W, T), ulaşabilir(W, T)$	izinsiz (bitirme garantisi yok)
$ulaşabilir(X, Y), ulaşabilir(Y, X), ulaşabilir(X, Z), ulaşabilir(X, W),$ $ulaşabilir(W, Y), ulaşabilir(Z, W), ulaşabilir(W, Z)$	izinsiz
$ulaşabilir(Z, Y)$	9 pozitif, 35 negatif örneği kapsar

Olabilecek iki literal olan $bağlantı(Z, Y)$ ve $ulaşabilir(Z, Y)$ tutarlı olan bir tümcecik üretebilirler, ancak kalan tüm pozitif örnekleri kapsadığı için $ulaşabilir(Z, Y)$ tercih edilir. Bu tümceciğin oluşturulmasından sonra biter, çünkü kapsanacak pozitif örnek kalmamıştır ve oluşturulacak başka bir tümceciğe ihtiyaç duyulmaz. Elde edilen program şudur:

$ulaşabilir(X, Y) : - bağlantı(X, Y)$

$ulaşabilir(X, Y) : - bağlantı(X, Z), ulaşabilir(Z, Y).$

FOIL için kapsama algoritması Şekil 4.16'da, özelleştirme algoritması Şekil 4.17'de verilmiştir.

$\varepsilon_{cur} \coloneqq \varepsilon$. ilk değer ver.
$H \coloneqq \emptyset$. ilk değer ver.
repeat {kapsama}
    $c \coloneqq T \leftarrow$. tümceciğine ilk değer ver.
    Bir c_{best} tümceciği bulmak için Özelleştirme Algoritması(c, ε_{cur}) çağır.
    c' elde etmek için gereksiz literalleri kaldırarak *c* son işle.
    H'ye c' ekleyerek yeni hipotez $H' \coloneqq H \cup \{c'\}$ elde et.
    Yeni bir eğitim seti $\varepsilon'_{cur} \coloneqq \varepsilon_{cur} - cover_{ext}(B, \{c'\}, \varepsilon^{+}_{cur})$ elde edinceye kadar
    c''nin kapsadığı pozitif örnekleri ε_{cur} 'dan kaldır.
    $\varepsilon_{cur} \coloneqq \varepsilon'_{cur}$, $H \coloneqq H'$ ata.
until $\varepsilon^{+}_{cur} = 0$ veya kodlama kısıtları ihlal edilince
Çıktı: *H* hipotezi

Şekil 4.16 FOIL Kapsama Algoritması

Yerel eğitim kümesine ilk değer ver $\varepsilon_i \coloneqq \varepsilon_{cur}$.
Geçerli tümceciğe ilk değer ver c_i:=c.
$i \coloneqq 1$ ilk değer ver.
while $\varepsilon^{-}_{cur} \neq 0$ veya kodlama kısıtları ihlal edildiğinde **do**
    $c_i \coloneqq T \leftarrow Q$'nin gövdesine eklemek ve $c_{i+1} \coloneqq T \leftarrow Q, L_i$. oluşturmak için en iyi L_i literalini bul.
    Yeni bir yerel eğitim seti ε_{i+1} oluştur.
    $c \coloneqq c_{i+1}$ ata.
    i'yi arttır.
end while.
Çıktı: *c* tümceciği

Şekil 4.17 FOIL Özelleştirme Algoritması

4.3 Uygulama

Bu bölümde tez kapsamında geliştirilen diğer bir KAB uygulaması anlatılacaktır. Yöntem olarak TMP'yi kullandık. TMP ile yapılan çalışmalarda elde edilen deneysel sonuçlar TMP'nin artalan bilgisini kullanmada başarılı olduğunu göstermiştir. TMP'nin bu yeteneğinin kelime anlamı belirginleştirme konusunda kullanılması bu alanda önemli bir gelişme sağlayabilir (Specia vd., 2007). Bu çalışmada amacımız denetimli derlem tabanlı KAB sistemi gerçekleştirmektir. Öncelikle uygulama için derlem olarak seçmiş olduğumuz Türkçe Sözlüksel Örnek Görevi'nden (Turkish Lexical Sample Task - TLST) bahsedeceğiz. Bu derlemi seçmemizin nedeni denetimli derlem tabanlı bir KAB uygulaması için uygun yapıda olmasıdır (Aydın ve Kılıçaslan, 2009).

4.3.1 Derlemin seçilmesi

TLST, SemEval-2'de sunulmuş bir değerlendirme çalışmasıdır (Orhan vd., 2007). Bu derlem eğitim ve test verisi olmak üzere iki farklı türde veri dosyası içermektedir. Bu veri setleri belirlenen 26 kelimeye ait örnek cümlelerden ve özelliklerden oluşmaktadır.

TLST'deki kelimelerin anlam sayısı bilgisi TDK'nın internet üzerindeki sözlüğünden alınmıştır. Eğitim ve test verisi oluşturulurken her kelime için en az 100 örnek etiketlenmesi yapılmıştır. Ancak bazı kelimelerin kullanım sıklığının azlığından yeterli sayıda örnek bulunmamaktadır. Toplam örnek setinin %70'i eğitim verisi %30'u test verisidir. Tablo 4.4'de TLST'de bulunan kelimeler ile ilgili kelimelere ait anlam sayısı, eğitim ve test örnek setinde bulunan cümle sayıları ve toplam örnek cümle sayısı verilmiştir.

Kelimeler	Anlam sayısı	Eğitim örneği sayısı	Test örneği sayısı	Toplam örnek sayısı
İsimler				
ara	7	192	63	255
baş	5	68	22	90
el	3	113	38	151
göz	3	92	27	119
kız	2	96	21	117
ön	5	72	23	95
sıra	7	85	28	113
üst	7	69	23	92
yan	5	65	31	96
yol	6	68	29	97
Ortalama	5	92	31	123
Fiiller				
al	24	963	125	1088
bak	4	207	85	292
çalış	4	103	61	164
çık	6	138	87	225
geç	11	164	90	254
gel	20	346	215	561
gir	6	163	84	247
git	13	214	120	334
gör	5	206	68	274
konuş	6	129	63	192
Ortalama	9.9	263.3	99.8	363.1
Zarflar ve Sıfatlar				
büyük	6	97	26	123
doğru	6	81	38	119
küçük	4	45	14	59
öyle	4	51	23	74
son	2	86	18	104
tek	2	40	10	50
Ortalama	4	66.7	21.5	88.2

Tablo 4.4 Kelimeler, anlam sayıları ve örnek sayıları

Derlem oluşturulmasında ODTÜ derleminden faydalanılmıştır. ODTÜ derlemi iki parçadan oluşmaktadır: ana derlem ve ağaç bankası derlemi. Ağaç bankası derlemi, ana derlemden alınan morfolojik olarak ayrıştırılmış, ancak üzerinde anlam belirginleştirilmesi yapılmamış cümlelerden oluşmaktadır. Bu cümleler XML formatındadır ve KAB için faydalı olabilecek sözdizimsel özelliklerden oluşmaktadır. KAB'da çok anlamlı kelimelerin seçimini yapabilmek için bu kelimelerin frekansları bulunmalıdır. Burada 5356 farklı kelime kökü bulunmaktadır. Bu kelimelerden 627'si 15 ve üstünde bulunurken diğerleri daha az bulunmaktadır.

TLST'deki eğitim ve test seti örnekleri oluşturulurken kelimelere ait cümleler için gerekli olan tüm sözdizimsel gösterimler bu derlemden alınmıştır. Ağaç bankası derlemi anlam etiketlemelerini içermemektedir. Bu yüzden TLST'deki anlam etiketlemeleri elle yapılmıştır. Anlam etiketleri key dosyalardan her kelime için alınmıştır. Bu key dosyalarda anlam gösterimleri satır satır bulunmaktadır. Her satırda dosya id, cümle ve oluşları belirlenen kelimeler için ince-taneli ve kaba-taneli anlamı ile verilmiştir. Bu key dosyalar, ağaç bankası XML dosyaları istenen kelime için bilgi almak amaçlı kullanılmıştır. Şekil 4.18'de ağaç bankası derlemindeki bir cümle için bulunan XML yapısı görülmektedir.

```
<?xml version="1.0" encoding="windows-1254" ?>
- <Set sentences="1">
- <S No="1">
    <W IX="1" LEM="" MORPH="" IG="[(1,"soğuk+Adj")(2,"Adv+Ly")]"
      REL="[2,1,(MODIFIER)]">Soğukça</W>
    <W IX="2" LEM="" MORPH="" IG="[(1,"yanıtla+Verb+Pos+Past+A1sg")]"
      REL="[3,1,(SENTENCE)]">yanıtladım</W>
    <W IX="3" LEM="" MORPH="" IG="[(1,".+Punc")]" REL="[,( )]">.</W>
  </S>
</Set>
```

Şekil 4.18 Ağaç bankası derleminde bulunan bir XML dosyası

Hedef kelimeler ve bağlamlarındaki seçilen kelimeler için küçük ölçekte bir ontoloji oluşturulmuştur. Türkçe WordNet ontoloji belirleme aşamasında yetersiz kalmıştır. Çünkü sadece fiiller için İngilizce WordNet'e uyumlu olarak ontolojiler hazırlanmıştır. İsimler, sıfatlar, zarflar ve diğer kelime türleri için ontoloji bulunmamaktadır. TLST'de belirginleştirmede etkili olacağı düşünülen üç seviyede

ontoloji bilgisi vardır. Birinci düzeyde kullanılan sınıf bilgileri Tablo 4.4'de, ikinci ve üçüncü düzeyde kullanılan sınıf bilgileri ise Tablo 4.5'de verilmiştir. Bu tablolar (Orhan, 2006)'dan alınmıştır.

Düzey 1	
Sınıf bilgisi	**Anlamı**
abstract entity	Soyut varlık
physical entity	Somut varlık

Tablo 4.5 Ontolojinin birinci düzeyinde sınıflar

Düzey 2		**Düzey 3**			
Sınıf bilgisi	**Anlamı**	**Sınıf bilgisi**	**Anlamı**	**Sınıf bilgisi**	**Anlamı**
Artifact	İnsan eliyle Yapılan ürün	*Animal*	Hayvan	*Measurement Unit*	Ölçü birimi
Change	Hal/durum değişimi	*Attitude*	Tavır/tutum	*Part*	Parça/bölüm
Cognition	Bilişsel etkinlik	*Attribute*	Özellik	*Perception*	Algı
Grouping	Grup	*Body*	Vücut bölümü	*Person*	Kişi
Location	Yer	*Business*	İş/ekonomi terimi	*Plant*	Bitki
Motion	Hareket	*Cause*	Sebep	*Possession*	Mal/mülk/iyelik
Organism	Organizma	*Communication*	İletişim etkinliği	*Process*	İşlem/süreç
Quality	Nitelik	*Container*	Kap	*Region*	Bölge
Quantity	Nicelik	*Delight*	Zevk verici madde	*Society*	Toplum/topluluk
Relation	İlişki	*Dress*	Kıyafet	*State*	Durum
Stative	Durum/oluş Bildiren etkinlik	*Drink*	İçecek	*Time*	Zaman
Substance	Madde	*Emotion*	Duygu	*Tool*	Alet/araç
Thing	Diğer	*Food*	Yiyecek	*Vehicle*	Taşıt
		Material	Materyal		

Tablo 4.6 Ontolojinin ikinci ve üçüncü düzeyindeki sınıflar

Tablo 4.7'de, TLST'de bulunan örnekler için verilen özellikler bir cümle örneği üzerinden aldığı değerlerle birlikte gösterilmiştir.

Özellik	Örnek
Dosya numarası	00002213148.xml
Cümle numarası	9
Sıra	0
Önceki kelimenin kökü	tap
Önceki kelimenin türü (POS)-düzeltilmiş	fiil
Önceki kelimenin 1.seviyedeki ontolojisi	soyut varlık
Önceki kelimenin 2.seviyedeki ontolojisi	özellik
Önceki kelimenin 3.seviyedeki ontolojisi	duygu
Önceki kelimenin türü (POS)	fiil
Önceki kelimenin türü (POS)-türetilmiş	zarf
Önceki kelimenin durum işaretçisi	?
Önceki kelimenin sahip olma durumu	fl
Önceki kelime-hedef kelime ilişkisi	niteleyen
Hedef kelime kökü	sev
Hedef kelime türü	fiil
Hedef kelime türü-türetme	isim
Hedef kelime durum işaretçisi	abl
Hedef kelime sahip olma durumu	tr
Hedef kelime-sonraki kelime ilişkisi	nesne
Sonraki kelimenin kökü	sıkıl
Sonraki kelimenin türü (POS)- düzeltilmiş	fiil
Sonraki kelimenin 1.seviyedeki ontolojisi	soyut varlık
Sonraki kelimenin 2.seviyedeki ontolojisi	özellik
Sonraki kelimenin 3.seviyedeki ontolojisi	duygu
Sonraki kelimenin türü (POS)	fiil
Sonraki kelimenin türü (POS)-türetilmiş	fiil
Sonraki kelimenin durum işaretçisi	?
Sonraki kelimenin sahip olma durumu	fl
Sonraki kelime-hedef kelime ilişkisi	cümle
İnce-taneli anlam sayısı	2
Kaba-taneli anlam sayısı	2
Cümle	#ne tuhaf şey; değil mi? İyi olmamdan; onu taparcasına sevmemden sıkıldı.#

Tablo 4.7 TLST'deki örnekler için verilen özellikler

4.3.2 Kullanılacak özelliklerin seçimi

Denetimli bir KAB uygulaması yapabilmek için kullanılacak özelliklerin seçimi önemlidir. Çünkü yeni karşılaşılan örneklerin sınıfını doğru olarak tahmin edecek iyi bir sınıflayıcı elde etmek için yeterli miktarda özellik gereklidir. İyi bir özellik farklı sınıflardan örnekleri birbirinden ayırabilmeli ve aynı sınıftaki örnekler için aynı değere sahip olmalıdır (Kononenko, 1994).

KAB işleminin kullanılan özelliklere karşı duyarlı olduğu yapılan çalışmalarda görülmüştür. Bazı özellikler belirginleştirmeyi etkilemezken, bazıları tek başına bile oldukça iyi sonuçlar verebilmektedir. Bazı özellikler de başka özellikler ile birlikte kullanıldığında başarıyı olumlu yönde etkilemektedir.

Uygun özellik kümesinin boyutunun bilinmediği durumlarda uygulanabilecek en basit yöntem olası bütün özellikleri kullanmak olarak dursa da bu yaklaşım beklenen sonucu vermemektedir (Tüysüz, 2010). (Almuallim ve Dietterich, 1991), bu konuda şunları söylemişlerdir:

"Örneğin, bir çok pratik uygulamada hangi özelliklerin ilgili olduğu ya da nasıl gösterileceği pek bilinmez. Kullanıcıların bu duruma doğal tepkisi, ilgili olabileceğini düşündükleri tüm özellikleri kullanmak ve öğrenme algoritmasının hangi özelliklerin gerçekten değerli olduğunu belirlemesidir. Diğer bir durum da, birçok farklı ikili fonksiyonları öğrenmek için aynı öğrenme verisinin kullanılması ve bu durumda bir çok ilgisiz özelliğin de bulunabilmesidir. Bu gibi durumlarda, verinin içinde bulunan özelliklerin bütün hedef fonksiyonları öğrenebilmek için yeterli olduğu garanti edilmelidir. Bununla birlikte, herbir fonksiyonu öğrenirken özelliklerin küçük bir alt setinin yeterli olması muhtemeldir."

Verilen alıntıdan da anlaşılacağı üzere, daha çok özellik daha fazla ayırdetme gücü sağlamamaktadır. Hatta bazı durumlarda çok özellik kullanmak işleyişi yavaşlatmaktan da öteye geçerek öğrenme algoritmasını yanıltıp performansı daha da kötüye götürmektedir. Bu duruma ilişkin (Yu ve Liu, 2004) aşağıdaki yorumu yapmışlardır.

"Klasik denetimli makine öğrenmesinde, etiketlenmiş sabit uzunluklu vektörler kümesi (örnekler) verilir. Bir örnekleme tipik olarak özellikler kümesine ve sınıf etiketine atanmış değerler olarak tarif edilir. Yapılması gereken iş yeni karşılaşılan örneklemelerin etiketlerini doğru şekilde tahminleyecek olan hipotezi (sınıflandırıcıyı) ortaya koymaktır. Sınıflayıcının öğrenilmesi özelliklerin aldıkları değerler tarafından belirlenir. Teoride, daha fazla özellik daha fazla ayırdetme gücü sağlamalıdır, fakat pratikte, sınırlı miktarda öğrenme verisi ile, fazla sayıdaki özellik sadece öğrenme sürecini yavaşlatmakla kalmayıp … ilgisiz ya da gereksiz veriler öğrenme algoritmasını yanıltabilmektedirler."

Bu doğrultuda TLST'de işaretlenmiş olan özelliklerden sadece bazılarını uygulamamız için seçtik. Uygulama için seçtiğimiz özellikler Tablo 4.8'de verilmiştir.

Özellikler
Önceki kelimenin kökü
Önceki kelimenin 1.seviyedeki ontolojisi
Önceki kelimenin 2.seviyedeki ontolojisi
Önceki kelimenin 3.seviyedeki ontolojisi
Önceki kelimenin türü
Önceki kelimenin durum işaretçisi
Önceki kelimenin sahip olma durumu
Önceki kelime-hedef kelime ilişkisi
Hedef kelime kökü
Hedef kelimenin türü
Hedef kelime durum işaretçisi
Hedef kelime sahip olma durumu
Hedef kelime-sonraki kelime ilişkisi
Sonraki kelimenin kökü
Sonraki kelimenin 1.seviyedeki ontolojisi
Sonraki kelimenin 2.seviyedeki ontolojisi
Sonraki kelimenin 3.seviyedeki ontolojisi
Sonraki kelimenin türü
Sonraki kelimenin durum işaretçisi
Sonraki kelimenin sahip olma durumu
Sonraki kelime-hedef kelime ilişkisi

Tablo 4.8 Uygulamada kullanılan özellikler

4.3.3 Kullanılacak tümevarımlı mantık programlama sisteminin belirlenmesi

Literatürde uygulaması ve testi yapılmış birçok TMP öğreniciler vardır. Bu sistemler girdi biçimine göre deneysel (empirical) ve etkileşimli (interactive) olarak veya arama yönüne göre aşağıdan-yukarıya (bottom-up) ve yukarıdan-aşağıya (top-down) olmak üzere sınıflandırılır (Lavrac ve Dzeroski, 1994). Deneysel sistemler giriş olarak örnek kümesini ve artalan tümcelerini alırlar ve hipotez üreterek bunu çıktı olarak verirler. Bu sistemlerden bazıları FOIL, MFOIL, GOLEM, ALEPH, PROGOL, LINUS, MARKUS ve MOBAL'dir. Etkileşimli sistemler bir örnek seti ile başlarlar ve bir hipotez üretirler ve bu hipotezi sistemdeki bir uzman tarafından yönlendirilen soru cevaplar ile güncelleştirirler. Etkileşimli TMP sistemlerine örnek olarak MIS, CLINT, CIGOL ve MARVIN verilebilir.

Uygulamamızda ALEPH sistemini kullandık. Temel ALEPH algoritması dört adımla tanımlanabilen çok basit bir süreç takip eder (Srinivasan, 1999). Bu adımlar:

1. *Örnek seçimi:* Genelleme yapılacak bir örnek seçilir. Eğer yoksa durulur ve sonraki sürece geçilir.

2. *En belirli öbeği oluşturma:* Seçilen örneğin gerektirdiği en belirgin tümceciğin oluşturulması aşamasıdır. Bu genellikle birkaç literalden oluşan bir belirli tümcecikdir (definite clause) ve *taban tümceciği (bottom clause)* olarak adlandırılır. Bu adıma *doygunluk (saturation)* adımı denilmektedir.

3. *Arama:* Taban tümcecikten daha genel bir tümcecik bulma aşamasıdır. Bu taban tümcecikteki bazı alt küme bağımlılarından "en iyi" skora sahip olanda arama yapılarak gerçekleştirilir. Bu aşamaya *indirgeme (reduction)* denir.

4. *Gereksiz olanları çıkarma:* En iyi skoru olan tümcecik geçerli teoriye eklenir, gereksiz görülen tüm örnekler kaldırılır. Bu adıma *örtü kaldırma (cover removal)* denir. Eğer örnek kaldıysa 1. adıma gidilir.

ALEPH kodu “aleph.pl” adında tek bir dosyadan oluşmaktadır. Bu dosyanın kullanılabilmesi için bir Prolog derleyicisinde derlenmesi gerekir. Uygulamada derleyici olarak SWI-Prolog kullanılmıştır. Teori oluşturma aşaması adımları sırasıyla aşağıdaki gibidir:

1. ALEPH’de teori oluşturmak için 3 veri dosyası gereklidir. Bunlar:

   - Artalan bilgisinin bulunduğu .b uzantılı bir dosya.
   - Pozitif örneklerin bulunduğu .p uzantılı bir dosya.
   - Negatif örneklerin bulunduğu .n uzantılı bir dosya.

   Bu dosyaların işlenmesi için isimleri aynı olmalıdır.

2. Bütün dosyalar *read_all(dosya-ismi)* komutu kullanılarak okunur.

3. *induce* veya *induce_tree* komutu kullanılarak tümevarım işlemi gerçekleştirilir ve sonucunda bir model oluşturulur. Bu model bir veya daha fazla kural içerebilir.

ALEPH’de artalan bilgisi Horn tümceciği formunda oluşturulmaktadır. Yüklem isimleri ve argümanları önceden tanımlanarak ALEPH’in etkin biçimde anlaşılır sonuçlar üretmesi sağlanır. Artalan bilgisi iki parçadan oluşur. Birisi her yüklem için *bildirim kısmı* diğeri ise hedefin içeriğini açıklayan *bilginin gösterimi* kısmıdır. Her yüklem için uygulamamızdaki bildirim kısmı aşağıdaki gibidir:

```
% Öğrenilecek hipotez

:- modeh(1,anlam(+cümle, -anlam_sayısı).

%Diğer modlar

:- modeb(1, yanında(+cümle, #kelime)).

:- modeb(1, konum(+ cümle, #kelimenin_konumu, #kelimenin_tür_bilgisi)).
```

:- modeb(1, tür(+cümle, # kelimenin_konumu, # kelime)).

:- modeb(1, ontoloji(+cümle, # kelimenin_konumu, #ontoloji_düzeyi, #ontoloji)).

:- modeb(1, durum(+cümle, # kelimenin_konumu, #kelimenin_durumu)).

:- modeb(1, ilişki(+cümle, # kelimenin_konumu, #ilişki_türü)).

:- modeb(1, sahip_olma(+cümle, # kelimenin_konumu, #sahip_olma_durumu)).

Bildirim kısmında, *modeh* ALEPH tarafından oluşturulacak teorinin baş kısmı olacak yüklemi ve yüklemin argümanlarını verirken, *modeb* gövde kısmını verir. Örneğin, *modeh(1, anlam(+cümle, -anlam_sayısı))* bildirimi *cümle* ve *anlam_sayısı* tiplerinde bir *anlam/1* yüklemini bildirir. "+" ve "-" sembolleri sırasıyla giriş ve çıkış değişkenlerini gösterir.

Bildirim kısmında ayrıca bir hipotezi oluşturmada kullanılacak yüklemlerin gösterildiği belirleme (determination) bölümü vardır. Örneğin, *determination(anlam/2, yanında/2)* bildirimi baş kısmı *anlam/2*, gövde kısmı *yanında/2* olan bir hipotez bildirir. Aşağıda uygulamamızda kullanılan determinationlar gösterilmiştir.

:- determination(anlam/2, yanında/2).

:- determination(anlam/2, konum/3).

:- determination(anlam/2, tür/3).

:- determination(anlam/2, ontoloji/4).

:- determination(anlam/2, durum/3).

:- determination(anlam/2, ilişki/3).

:- determination(anlam/2, sahip_olma/3).

Ayrıca ALEPH parametrelerini set etmek için kullanılan *set/2* yüklemi de artalan dosyasında verilmektedir. ALEPH'in ayarlar kısmında, *set/2* yüklemi belirli sayıdaki parametrelerin ayarları için temel formdur. Aşağıdaki kod parçasında uygulamada kullanılan *set/2* yüklemleri verilmiştir.

```
:- set(tree_type, classification).

:- set(classes, (anlam1,anlam2,…,anlamN)).

:- set(minpos, 2).

:- set(prune_tree, true).

:- set(confidence, 0.25).

:- set(evalfn, entropy).

:- set(dependent, 2).

:- set(i, 2).

:- set(test_pos, 'test_dosyası.f').

:- set(test_neg, 'test_dosyası.n').
```

Şimdi, *set/2* yükleminin alabildiği değerleri ve uygulamamız için yukarıda verilen kod parçasında aldığı değerlerin ne ifade ettiğini açıklayalım.

set(tree_type,+V): Kullanılan ağaç tipinin alabildiği değerler verilmiştir. *V* sınıflandırma (classification), class_probability (sınıf olasılığı), regression (regrasyon) veya model değerlerinden birini alabilir. Uygulamada aldığı değer *classification* olmuştur. Yani sınıflandırma yapılacağı söylenmektedir.

set(classes,+V): V ağacı öğrenen tarafından tahmin edilen sınıfların listesidir. Uygulamada ilgili kelimenin anlamları bu sınıf listesini oluşturmaktadır. Örneğin, üç

anlamı olan bir kelime için belirginleştirme yapılacaksa bu sınıf listesi *(anlam1, anlam2, anlam3)* olur.

set(minpos,+V): V burada pozitif bir sayısal değerdir (varsayılan olarak 1 değerini alır). Kabul edilebilir tümcecik tarafından kapsanan pozitif örnek sayısını veren en düşük değerdir. Eğer en iyi tümcecik bu sayı altındaki pozitif örnekleri kapsarsa, geçerli teoriye eklenmez.

set(prune_tree,+V): V ture veya false değerlerinden birini alabilir (varsayılan değeri false). Ağaç öğrenenin oluşturduğu kuralların pesimist budamaya maruz kalıp kalmadığını kontrol eder. Uygulamada aldığı değer *true* olmuştur, bu da budama yapıldığı anlamına gelmektedir.

set(confidence,+V): V (0.0,0.1) aralığındaki bir kayan noktalı sayıdır. Ağaç öğrenen tarafından budama kuralı için belirlenen güven (confidence) değerini belirler.

set(evalfn,+V): *V* coverage, compression, posonly, pbayes, accuracy, laplace, auto_m, mestimate, entropy, gini, sd, wracc, veya user değerlerini alabilir. Bir aramanın değerlendirme fonksiyonunu ayarlar. Uygulamada aldığı değer *entropi* olmuştur.

set(dependent,+V): V pozitif bir sayıdır. Örneklerdeki bağımlı değişkenin argümanını gösterir.

set(i,+V): V pozitif bir sayıdır (varsayılan değeri ikidir). Yeni değişkenin katmanlarında üst sınırı ayarlar.

set(test_pos,+V): V bir Prolog atomu veya Prolog atom listesidir. Test için pozitif örnekleri içeren dosya ismini veya dosya isim listesini set eder.

set(test_neg,+V): V bir Prolog atomu veya Prolog atom listesidir. Test için negatif örnekleri içeren dosya ismini veya dosya isim listesini set eder.

4.3.4 Eğitim ve test verisinin oluşturulması

Geliştirilen uygulamada TLST'den alınan 10 isim, 10 fiil ile 6 sıfat ve zarf üzerinde KAB gerçekleştirilmiştir. Bu kelimelere ait anlam sayısı, eğitim ve test örnek setinde bulunan cümle sayıları ve toplam örnek cümle sayısı Tablo 4.4'de verilmiştir. İlk aşamada bu kelimelere ait özelliklerin TLST'den alınma işlemini otomatik gerçekleştiren bir uygulama geliştirilmiştir. Bu uygulamanın geliştirilme sebebi ALEPH sistemine verilen dosyalarının elle oluşturulmasının zahmetli olmasıdır. Uygulama ALEPH sistemine verilen veri dosyalarının içeriğini oluşturmak için TLST'den ihtiyaç duyulan bilgileri çekmektir. Artalan bilgisinin bulunduğu dosya, örneklerin doğru anlamlarının verildiği pozitif örnek dosyası ve yanlış anlamların verildiği negatif örnek dosyası, eğitim ve test örnekleri için ayrı ayrı oluşturulmaktadır.

TLST'deki eğitim örnekleri anlamlara göre gruplanarak hazırlanmıştır. Bu tür bir düzenleme öğrenme aşamasında istenilen sonuçları vermeyebilir. Bu yüzden uygulamada ilk olarak eğitim örnekleri rastgele olarak karıştırılmıştır. Sonrasında eğitim örnekleri yüzde onluk dilimler halinde alınarak ALEPH dosyaları oluşturulmuştur. Bu şekilde ilk %10'luk cümleler ile eğitim yapıldıktan sonra test cümleleri ile performans değerlendirmesi yapılmaktadır. Bu şekilde sırası ile diğer yüzdelik dilimlerle eğitim yapıldıktan sonra test yapılmaktadır. Böylelikle bir çizge elde edilir. Buradaki amaç mutlu çizge (happy graph) elde edebilmektir. Elde edilmesi durumunda mevcut örnek sayısının öğrenme için yeterli olduğu sonucuna ulaşılır.

KAB uygulamamızı bir TMP sistemi olan ALEPH (A Learning Engine for Proposing Hypotheses) ile gerçekleştirdik.

ALEPH sistemine verilecek veri dosyalarının nasıl oluşturulduğunu bir örnek üzerinden açıklayalım (Aydın ve Kılıçaslan, 2010a):

Yeşilden maviye dönüşen iri gözlerini bize çevirmişti. (4.1)

Bu cümlede hedef kelime olarak anlamı belirsiz olan "göz" kelimesi seçilmiştir. Uygulamamızda kullandığımız yedi özellik, artalan dosyasında yer almaktadır. Bu özellikleri, verilen (4.1) cümlesi üzerinden açıklayalım:

1) *yanında* özelliği: Anlamı belirginleştirilecek hedef kelimenin sağında ve solunda bulunan kelimeler burada verilir. Yapılmış bazı çalışmalarda hedef kelimenin yanından alınan kelime sayısı beşe kadar çıkmaktadır. Ancak TLST içinde hedef kelimenin yanındaki ilk kelimeler verildiği için uygulamamızı tek kelime sınırladık. Bu özelliğin programdaki gösterimi aşağıdaki gibidir:

yanında(cümle_no, kelime).

yanında(cümle1, iri).

yanında(cümle1, çevir).

2) *konum* özelliği: Hedef kelimenin sağında ve solunda bulunan kelimeler ve bu kelimelerin hedef kelimeye göre konum bilgisi verilir.

konum(cümle_no, kelimenin_konumu, kelime).

konum(cümle1, solundaki_kelime, iri).

konum(cümle1, sağındaki_kelime, çevir).

3) *tür* özelliği: Hedef kelimenin tür bilgisi ile hedef kelimenin sağında ve solunda bulunan kelimelerin tür bilgisi ve hedef kelimeye göre konum bilgileridir.

tür(cümle_no, kelimenin_konumu, kelimenin_tür_bilgisi).

tür(cümle1, hedef_kelime, isim).

tür(cümle1, solundaki_kelime, sıfat).

tür(cümle1, sağındaki_kelime, fiil).

4) *ontoloji* özelliği: Hedef kelimenin sağında ve solunda bulunan kelimelerin ontolojik düzey bilgisidir. TLST'de ontoloji düzeyi üç olduğu için uygulamada ontololi düzeyi üç düzey ile sınırlanmıştır.

ontoloji(cümle_no, kelimenin_konumu, ontoloji_düzeyi, ontoloji).

ontoloji(cümle1, solundaki_kelime, 1, soyut_varlık).

ontoloji(cümle1, solundaki_kelime, 2, özellik).

ontoloji(cümle1, solundaki_kelime, 3, durum).

ontoloji(cümle1, sağındaki_kelime, 1, somut_varlık).

ontoloji(cümle1, sağındaki_kelime, 2, hareket).

ontoloji(cümle1, sağındaki_kelime, 3, hareket).

5) *durum* özelliği: Hedef kelimenin durum bilgisi ile hedef kelimenin sağında ve solunda bulunan kelimelerin durum bilgileri ve hedef kelimeye göre konum bilgileri burada verilir.

durum(cümle_no, kelimenin_konumu, kelimenin_durumu).

durum(cümle1, hedef_kelime, belirtme).

durum(cümle1, solundaki_kelime, _).

durum(cümle1, sağındaki_kelime, _).

6) *ilişki* özelliği: Hedef kelimenin sağında ve solunda bulunan kelimeler ile olan ilişki türü ve bu kelimelerin hedef kelimeye göre konum bilgileridir.

ilişki(cümle, kelimenin_konumu, ilişki_türü).

ilişki(cümle1, hedef_kelime, nesne).

ilişki(cümle1, solundaki_kelime, tümleyen).

ilişki(cümle1, sağındaki_kelime, cümle).

7) *sahip olma* özelliği: Hedef kelime ile hedef kelimenin sağında ve solunda bulunan kelimelerin iyelik ekine sahip olup olmamasına ilişkin bilgidir.

sahip_olma(cümle, kelimenin_konumu, sahip_olma_durumu).

sahip_olma (cümle1, hedef_kelime, doğru).

sahip_olma(cümle1, solundaki_kelime, yanlış).

sahip_olma(cümle1, sağındaki_kelime, yanlış).

Pozitif örnekler dosyasında, hedef kelimenin doğru anlamını veren bilgi kodlanır.

anlam(cümle, hedef_kelimenin_anlamı).

anlam(cümle1, anlam1).

Negatif örnekler dosyasında, hedef kelimenin doğru anlamını sağlamayacak şekilde diğer anlam etiketleri kullanılarak elde edilen bilgi verilir.

anlam(cümle, hedef_kelimenin_hatalı_anlamı).

anlam(cümle1, anlam2).

Uygulamada öncelikle hedef kelimenin eğitim kümesi örneklerinden alınarak oluşturulan artalan bilgisi, pozitif ve negatif örnek dosyaları ALEPH sistemine verilir. Sistemin veriyi işlemesi sonucunda kurallardan oluşan bir model elde edilir. Bu model test kümesinden alınan örneklerle oluşturulan artalan bilgisi, pozitif ve negatif örnek dosyaları sisteme verilerek test edilir.

ALEPH sistemi tarafından, verilen hedef kelime “göz” için eğitim örnekleri ile elde edilen kurallar aşağıdaki gibidir:

Kural 1:

anlam(A, B) :- ontoloji(A, sağındaki_kelime, 2, bilişsel),

sahip_olma(A, hedef_kelime, yanlış), B=2.

Kural 2:

anlam(A, B) :- not(sahip_olma(A, hedef_kelime, yanlış)), B=1.

Kural 3:

anlam(A, B) :- not(ontoloji(A, first_sağındaki_kelime, 2, bilişsel)), B=1.

Şimdi bu belirlenen kuralların ifade etmek istediklerini açıklayalım. Birince kurala göre, verilen bir cümlede hedef kelimenin sağındaki kelimenin 2. düzeyde ontolojisi "bilişsel" ve hedef kelimenin sahip olduğu iyelik eki yoksa, hedef kelime bu cümlede 2. anlamı ile kullanılmıştır. İkinci kuralda ise, verilen cümlede hedef kelimenin sahip olduğu bir iyelik eki varsa hedef kelime bu cümlede 1. anlamı ile kullanıldığı söylenmektedir. Son kural olan 3. kurala göre, verilen bir cümlede hedef kelimenin sağındaki kelimenin 2. düzeyde ontolojisi "bilişsel" değilse hedef kelimenin bu cümlede 1. anlamı ile kullanıldığı söylenmiştir.

5. DEĞERLENDİRME

Bu bölümde öncelikle KAB çalışmalarının değerlendirilmesinde kullanılan ölçütlerden bahsedilecektir. Sonrasında ise KAB sistemlerinin değerlendirmesinde kullanılan kavramlar ve değerlendirme için yapılan çalıştaylar hakkında bilgi verilecektir. Son olarak da uygulama sonucu elde edilen sonuçların değerlendirmesi yapılacaktır.

5.1 Kelime Anlamı Belirginleştirme İçin Performans Değerlendirme Ölçütleri

KAB için performans değerlendirme ölçütleri olarak; doğruluk (accuracy), duyarlılık (precision) ve geriçağırım (recall) kullanılmaktadır (Agirre ve Edmonds, 2006). Bu ölçütler sınıflandırma sonucu oluşan karmaşıklık matrisinden (confusion matrix) elde edilmektedir. Karmaşıklık matrisinde Tablo 5.1'de verilen dört farklı durum oluşmaktadır.

		Gerçek	
		Pozitif	**Negatif**
Tahmin	**Pozitif**	DP	YP
	Negatif	YN	DN

Tablo 5.1 Karmaşıklık matrisi

Çalışma alanında var olan bir öğeye var dendiğinde ya da gerçekte doğru olan bir şeyi, tasarlanan sistem de doğru olarak tahmin etmiş ise bu durum Doğru Pozitif (DP) olarak ifade edilir. Gerçekte doğru olan bir şey yanlış olarak tahmin edilirse, bu Yanlış Negatif (YN) olarak ifade edilir. Gerçekte yanlış olan bir şeyin doğru olduğu söylenirse, bu durum Yanlış Pozitif (YP) olarak ifade edilir. Gerçekte yanlış olan bir duruma sistem yanlış derse, bu durum Doğru Negatif (DN) olarak isimlendirilir.

Karmaşıklık matrisinde bulunan bu sayılarla elde edilen ölçütlerden bir tanesi doğruluk ölçütüdür. Bu ölçüt, tüm veri içinde doğru tahmin edilenlerin oranını ölçmeye imkan tanır. Bütün hata tiplerini dikkate alarak, pozitif ve negatif örnekleri aynı derecede önemsemeyi sağlar. Sınıflandırıcının toplam performansını değerlendirmeye yardımcı olur. Doğruluk formülü aşağıdaki gibidir:

$$Doğruluk = \frac{DP+DN}{DP+DN+YN+YP} \tag{5.1}$$

Fakat doğruluk ölçütü, veri kümesinde dengesiz dağılım var ise yeterli olmamaktadır. Bu durumda kullanılan *geriçağırım* ve *duyarlılık* ölçütleri, sırasıyla, pozitif örneklerin negatif olarak sınıflandırılmasından oluşan hatalar ile negatif örneklerin pozitif olarak sınıflandırılmasından oluşan hataları belirtirler. Bu ölçütler aşağıdaki formüllerle elde edilir:

$$Duyarlılık = \frac{DP}{DP+YP} \tag{5.2}$$

$$Geriçağırım = \frac{DP}{DP+YN} \tag{5.3}$$

Şimdi bu performans ölçütlerinin KAB için nasıl ifade edildiğine bakalım (Agirre ve Edmonds, 2006):

$$Doğruluk = \frac{Anlamı\ belirginleştirilen\ toplam\ örnekleme\ sayısı}{Toplam\ örnekleme\ sayısı}$$

$$Duyarlılık = \frac{Anlamı\ doğru\ olarak\ belirginleştirilen\ örnekleme\ sayısı}{Sistem\ tarafından\ belirginleştirilen\ toplam\ örnekleme\ sayısı}$$

$$Geriçağırım = \frac{Anlamı\ doğru\ olarak\ belirginleştirilen\ örnekleme\ sayısı}{Testteki\ toplam\ örnekleme\ sayısı}$$

5.2 Kelime Anlamı Belirginleştirme Sistemlerinin Değerlendirilmesi

Günümüze kadar yapılan çalışmalarda kullanılan bilgi kaynakları ve tekniklerin farklı birleşimlerinden çeşitli KAB yaklaşımları ortaya çıkmıştır. Doğal olarak akla bu yaklaşımlar ile oluşturulan sistemlerden hangisinin daha iyi olduğu sorusu gelmektedir. KAB sistemlerini geliştiren araştırmacılar da, bu sistemlerin değerlendirilmesi gerektiğini düşünmektedir. Ancak farklı KAB sistemlerinin karşılaştırılmalı değerlendirmesini yapmak kolay değildir. Bu sorun çok farklı diller için hazırlanmış KAB sistemlerinin değerlendirildiği Senseval değerlendirme çalışması ile çözülmeye başlanmıştır.

5.2.1 Değerlendirmede kullanılan temel kavramlar

KAB sistemlerinin değerlendirilmesinde kullanılan bazı anahtar kelimeler ve konular şunlardır (Palmer vd., 2006):

Altın standart: KAB işleminde, aynı test verilerini iki farklı kişinin işaretlemesi sonucu ortaya çıkan problemlerin çözülmesi sonucu elde edilen verinin durumu için kullanılan bir terimdir.

Anlam deposu: Değerlendirmedeki en önemli kriterlerden birisidir. Bir hesaplamalı sözlük veya makinece okunabilir sözlükteki her kelime için bulunan anlam parçalarıdır. İdeal bir anlam deposu her kelime için açık ve tutarlı anlam ayrımını yapabilmelidir.

Görev tanımı: KAB sistemlerinde iki tip değerlendirme yaklaşımı vardır. Birincisi *in vitro* denilen, herhangi bir başka uygulamaya bağımlılığı olmayan KAB görevleri için bir değerlendirme yaklaşımıdır. Diğer tip değerlendirme için *in vivo* denilen KAB görevinin bir doğal dil işleme uygulamasının performansına katkısını ele alan yaklaşımdır. Çoğunlukla *in vitro* yaklaşımına göre KAB görevlerinin değerlendirmesi yapılmaktadır.

KAB tek başına, bütün kelimeler (all-words) ve sözlüksel örnek görevi (lexical sample task) olarak iki alt göreve ayrılabilir. Bütün kelimeler görevinde KAB sistemlerinin metindeki veya bağlamdaki tüm kelimeleri etiketlemesi gerekir. Kelime türü işaretlemeye benzermiş gibi görünse de bütün kelimeler görevi tamamen farklı bir işaretleme etiketi kümesi gerektiğinden oldukça farklıdır. Sözlüksel örnek görevinde ise bir kelimeler örneği her kelime için derlem örnekleri ile birlikte sözlükten dikkatlice seçilir. Sistemlerin bu örnek kelimeleri metinde kısa açıklamalarla etiketlenmesi gerekir. Bütün kelimeler görevinden farklı olarak sözlük girişleri sadece seçilen kelimeler için gereklidirler, yani sözlük seçiminde daha fazla esneklik vardır.

Derlem: Derlemler sözlüksel örnek görevi yapısında ise veri, tipik olarak anlam deposundan anlam girişine göre bir işaretçi ile etiketlenmiş hedef kelimeyi içeren cümlelerden oluşur. Etiketlenmiş verilerin bir bölümü denetimli makine öğrenmesi sistemleri için eğitim verisi olarak kullanılabilirken, diğer bölümü test amaçlı kullanılabilir.

Puanlama: Belirli bir test öğesinde bir sistemi puanlamanın en basit yolu tam doğruluk (exact match) kriterini kullanmaktır. Hiyerarşik olarak organize edilmiş bir anlam deposu varsa puanlama üç seviyede tanımlanabilir: ince, kaba ve karışık. İnce seviyede puanlamada sadece benzer anlam etiketleri doğru sayılır. Kaba seviyede puanlama bütün anlam etiketleri altın standartta verilir ve atanan anlam ile doğru anlam en üst seviyede aynı kökten geliyorsa doğru sayılırlar. Karmaşık seviyede puanlama anlam deposu hiyerarşik yapıda ise uygulanabilir. Doğru anlamın çocuklarının ya da atasının seçilmesine göre olasılık hesabı yapılarak puanlama yapılır.

Diğer bazı değerlendirme kriterleri de bütün kelimelere anlam ataması yapan sistemlerde kullanılmaktadır. Bu kriterlerden bir sistemdeki *kapsama (coverage),* sistemin tahmin ettiği anlam etiketlerinin değerlendirme kümesindeki oranıdır. Bir sistemin *duyarlılık* kriteri ise sistemin tahmin ettiklerinden ne kadarının doğru olduğu bilgisidir. *Geriçağırım* doğru tahmin ettiklerinin değerlendirme kümesindeki toplam oranıdır. Anlam etiketlemesi için doğruluk geriçağırım olarak değerlendirilir.

Alt sınır: Değerlendirmelerde ortaya konulması gereken bir alt sınır olmalıdır. En basit temel sınır en sık kullanılan anlamın seçilmesidir (Gale vd., 1992a).

Üst sınır: Otomatik KAB sistemlerinde kavramsal üst sınır, aynı veya karşılaştırılabilir veri üzerinde etiketleme yapan insanların seviyesinde doğruluğa sahip olmaktır (Gale vd., 1992a). Çünkü sistemlerin tutarlılıklarının insanların tutarlılığını geçmesi beklenmemektedir.

5.2.2 Senseval çalıştayları

5.2.2.1 Senseval-1

Senseval, ARPA'nın desteklediği MUC ve TREC değerlendirmelerine yapı olarak benzemektedir. İlk Senseval 1998 yazında İngilizce, Fransızca ve İtalyanca için 23 araştırma grubunun katılımı ile gerçekleştirilmiştir. Bu çalıştayın amacı, KAB sistemlerinin farklı kelimeler ve kullanılan dilin farklı özellikleri için gerekli değerlendirmelerin yapılmasıydı. Senseval projesinde kullanılan sözlük ve derlem 1990 yılında Oxford University Press and Digital projesi olarak geliştirilmiş olan HECTOR (Atkins, 1993) sözlüksel veritabanıdır. Bu veritabanında bir sözlükle ilişki bir derlem bulunmaktadır. Senseval da bu veritabanın seçilme nedeni ihtiyaç duyulan özellikleri taşıması ve maliyet açısından ise elle etiketleme için gerek duyulan zaman ve parayı azaltıyor olmasıdır.

Senseval'da üç çeşit veri kullanılabilmektedir:

- *Kuru çalışma verisi:* Bu veri sözlüksel girişler ve elle etiketlenmiş derlem örneklerinden oluşur. Eğitim ve test verilerine benzer şekilde örneklenmektedir.

- *Eğitim verisi:* Değerlendirilmesi yapılacak sözlüksel örnek için sözlüksel girişler ve elle etiketlenmiş derlem örneklerinden oluşur. Sözlüksel bilgi, sistemlerin kendilerini bu bilgiye adapte etmeleri ve gerektiğinde ekleme yapmaları için oluşturulur. Derlem örnekleri sağlanmıştır, böylece denetimli eğitim sistemleri sözlüksel örnekteki kelimeler için eğitilebilir.

- *Değerlendirme verisi:* Değerlendirme her görev için basitçe derlem örneklerinin bir kümesini içerir. Her kelime için en az üç işaretleme yapılmış ve bu işaretlemeler araştırmaya katılanlara verilmemiştir.

Değerlendirmede öncelikle bir kelime grubu seçilmekte ve bu kelimelerin kullanılan derlemlerin bazı cümlelerindeki anlamları işaretleyiciler tarafından belirlenmektedir. Sonrasında katılımcı, KAB sistemleri oluşturulmuş derlemlerdeki aynı kelimelerin hangi anlamda kullanıldıklarını tahmin eder. Farklı sistemlerin atadıkları anlamlar, eğer işaretleyicilerin belirlediği anlamlarla tam örtüşüyorsa doğru, kısmen örtüşüyorsa kısmen doğrudur. Bu sistemlerin atadıkları anlamlar işaretçilerin belirlediği anlamlardan tamamen farklı ise yanlış olarak kabul edilmiş ve toplamda aldıkları sonuçlara göre değerlendirilmişlerdir.

Senseval-1'de İngilizce dili için yarışan 3 farklı kategoride sınıflandırılmış 17 sistem vardır[6]:

1. Bütün kelimeler sistemi (A), bütün bağlam kelimelerini belirginleştirir.
2. Denetimli eğitim sistemleri (S), belirginleştirecekleri her kelime için 30'dan fazla anlam etiketli örneğe ihtiyaç duyar.
3. Diğer eğitim sistemleri (O), 30'dan az anlam etiketli örneğe ihtiyaç duyan, ancak her kelime için öğrenme aşaması gerektiren sistemlerdir.

Sonuçlar farklı sistemler için karşılaştırmalı olarak Tablo 5.2'de[7] verilmiştir.

	İnce-taneli	**Karışık-taneli**	**Kaba-taneli**
İnsan	0.965 (0.963)	0.968 (0.967)	0.970 (0.968)
En iyi sistem	0.771 (0.771)	0.797 (0.797)	0.814 (0.813)
Sistemlerin ortalaması	0.550 (0.376)	0.632 (0.410)	0.661 (0.426)
En kötü sistem	0.205 (0.162)	0.315 (0.248)	0.338 (0.267)
En iyi dayanak	0.691 (0.689)	0.720 (0.719)	0.741 (0.739)

Tablo 5.2 İngilizce Sözlüksel Örnek Görevi için Senseval-1'deki sonuçlar (İlk değer duyarlılık, parantez içindeki değer geriçağırımdır.)

[6] http://www.senseval.org/

[7] Sonuçlar http://www.itri.brighton.ac.uk/events/senseval/ARCHIVE/RESULTS/senseval-summary.html adresinden alınmıştır.

5.2.2.2 Senseval-2

İkinci Senseval çalıştayı olan Senseval-2, 2001 baharında 10 farklı dil için 37 araştırma grubu ile yapılmıştır[8]. Senseval-2 üç tipte görev için değerlendirilmiştir (Edmonds, 2002):

- Bütün kelimeler (AW - All Words): İngilizce, Hollondaca, Çek dili ve Estonya dilindeki tüm kelimeler için metinlerde geçen hemen hemen bütün kelimeler belirginleştirilmeye çalışılmıştır.
- Sözlüksel Örnek (LS – Lexical Sample): Bask dili, Çince, Danimarkaca, İngilizce, İtalyanca, Japonca, Korece, İspanyolca ve İsveççe için sözlüksel örnek görevidir. Seçilen bir kelime grubunun anlam belirginleştirilmesi verilen metinlerde gerçekleştirilmeye çalışılmıştır.
- Çeviri (TL – Translation Memory): Japonca için çeviri görevidir. Kelime anlamlarının farklılığını ayırdetmek için çevirilerden faydalanılmıştır.

Dil	Görev	Sunumlar	Takımlar	IAA	Dayanak	En iyi sistem
Çek dili	AW	1	1	-	-	0.94
Bask dili	LS	3	2	0.75	0.65	0.76
Estonya dili	AW	2	2	0.72	0.85	0.67
İtalyanca	LS	2	2	-	-	0.39
Korece	LS	2	2	-	0.71	0.74
İspanyolca	LS	12	5	0.64	0.48	0.65
İsveçce	LS	8	5	0.95	-	0.70
Japonca	LS	7	3	0.86	0.72	0.78
Japonca	TL	9	8	0.81	0.37	0.79
İngilizce	AW	21	12	0.75	0.57	0.69
İngilizce	LS	26	15	0.86	0.51	0.64

Tablo 5.3 Senseval-2 için sonuçlar (Geriçağırım değerleri verilmiştir.)

[8] http://www.senseval.org

Senseval-1'deki değerlendirme metodu Senseval-2'de de kullanılmıştır. Her görev için bir anlam deposu, elle etiketlenmiş bir derlem ve opsiyonel anlam hiyerarjisi vardır. İngilizce için anlam deposu olarak WordNet 1.7; İspanyolca, İtalyanca ve Estonya dili için EuroWordNet kullanılmıştır. Sonuçlar gönderildikten sonra otomatik olarak değerlendirme yapılmıştır. Senseval-1'de ortaya konan değerlendirme işlemi küçük bazı değişikliklerle burada da kullanılmıştır. Her dil için farklı görevlerdeki değerlendirme sonuçları Tablo 5.3'de verilmiştir.

5.2.2.3 Senseval-3

Senseval-3 2004 yılında yapılmıştır. Çekirdek KAB ve yanısıra anlamsal rollerin tanımlanması, çok dilli açıklamalar, mantıksal formlar ve alt-ulamlama elde etme gibi konular için 16 farklı görev içermektedir[9].

- İngilizce ve İtalyanca dilleri için tüm kelimeler.
- Bask dili, Çince, İngilizce, İtalyanca, Katalanca, Romence, İspanyolca ve İsveçce için sözlüksel örnekler.
- Otomatik alt-ulamlama elde etme.
- Çok dilli sözlüksel örnekler.
- WordNet sözlükleri için KAB.
- Anlamsal roller.
- Mantıksal formlar.
- İsveçce için anlamsal roller.

Senseval-3'de 160'dan fazla sistem çalıştaya katılmıştır. Tablo 5.4'de bu sistemler içinde performansı en yüksek olan ilk 15 sistem geriçağırım değerine göre sıralanmış bir şekilde verilmiştir (Synder ve Palmer, 2004).

[9] http://www.senseval.org/senseval3

Sıra	Sistem	Tip	Duyarlılık	Geriçağırım
1	GAMBL-AW	Denetimli	0.651	0.651
2	SenseLearner	Denetimli	0.651	0.642
3	Koc University	Denetimli	0.648	0.639
4	R2D2:English-all-words	-	0.626	0.626
5	Meaning-allwords	Denetimli	0.625	0.623
6	Meaning-simple	Denetimli	0.611	0.610
7	LCCaw	-	0.614	0.606
8	Upm-shmm-eaw	Denetimli	0.616	0.605
9	UJAEN	Denetimli	0.601	0.588
10	IRST-DDD-00	Denetimsiz	0.583	0.582
11	University of Sussex-Prob5	-	0.585	0.568
12	University of Sussex-Prob4	-	0.575	0.550
13	University of Sussex-Prob3	-	0.573	0.547
14	DFA-Unsup-AW	Denetimsiz	0.557	0.546
15	KUNLP-Eng-All	Denetimsiz	0.510	0.496

Tablo 5.4 Senseval-3'deki en iyi 15 sistem

5.2.2.4 SemEval-1/Senseval-4

SemEval-1 2007 Uluslararası Anlamsal Değerlendirme çalıştayının dördüncüsüdür. Komite tarafından seçilen 18 görev şunlardır[10]:

- Çapraz Dil Bilgi Erişimi
- Kelime Anlamı Çıkarımı ve Ayırıcılığı Sistemlerinin Değerlendirilmesi
- Nominaller arasındaki Anlamsal İlişkilerin Sınıflandırılması
- Çok dilli Çince-İngilizce Sözlüksel Örnek Görevi
- Edatlar için Kelime Anlamı Belirginleştirme

[10] http://nlp.cs.swarthmore.edu/semeval/

- Bütün kelimeler için kaba taneli İngilizce
- SemEval 2007'de Ad Aktarması Çözümü
- Katalonca ve İspanyolca'daki Çok aşamalı Anlamsal Açıklama
- SemEval 2007'de İngilizce Sözlüksel Yerdeğiştirme Görevi
- İngilizce-Çince Paralel Metinleri ile Sözlüksel Örnek Görevi
- Türkçe Sözlüksel Örnek Görevi
- Web İnsan Arama, Etkisel Metin
- TemEval: Zaman-Olay Geçici İlişki Tanımlama Değerlendirmesi için Bir Öneri
- Bilgi kaynaklarının geniş kapsamlı değerlendirmesi
- İngilizce Sözlüksel Örnek
- İngilizce SRL ve İngilizce Bütün kelimeler Görevi
- Arapça Anlamsal Etiketleme
- Çerçeve Anlamsal Yapı Çıkarılması

Çalıştaya 14 sistem katılmıştır. Bu sistemlerden İngilizce bütün kelimeler görevi için ince taneli sonuçlara göre ilk 10 sistem Tablo 5.5'de, kaba taneli sonuçlara göre ilk 10 sistem ise Tablo 5. 6'da verilmiştir (Navigli vd., 2007; Pradhan vd., 2007).

Sıra	**Sistem**	**Sınıflama tekniği**	**F-ölçütü**
1	PNNL	Maksimum Entropi	0.591
2	NUS-PT	Destek Vektör Makinaları	0.587
3	UNT-Yahoo	Bellek-tabanlı	0.583
4	NUS-ML	Naive Bayes	0.576
5	UBC-ALM	kNN	0.544
6	UBC-UMB-2	kNN	0.540
7	PU-BCD	Üssel Model	0.539
8	RACAI	Denetimsiz	0.527
9	UPV-WSD	Denetimsiz	0.469
10	JU-SKNSB	Denetimsiz	0.402

Tablo 5.5 SemEval-1 için ince taneli İngilizce bütün kelimeler görevi sonuçları

Sıra	Sistem	Deneme	Duyarlılık	Geriçağırım	F-ölçütü
1	NUS-PT	100%	0.825	0.825	0.825
2	NUS-ML	100%	0.815	0.815	0.816
3	LCC-WSD	100%	0.814	0.814	0.814
4	GPLSI	100%	0.796	0.796	0.796
5	UPV-WSD	100%	0.786	0.786	0.786
6	TKB-UO	100%	0.702	0.702	0.702
7	PU-BCD	90.1%	0.697	0.628	0.661
8	RACAI-SYNWSD	100%	0.657	0.657	0.657
9	SUSSX-FR	72.8%	0.717	0.522	0.604
10	USYD	95.3%	0.588	0.560	0.574

Tablo 5.6 SemEval-1 için kaba taneli İngilizce bütün kelimeler görevi sonuçları

5.2.2.5 SemEval-2

Son olarak düzenlenen değerlendirme çalıştayı olan Semeval-2, 2010 yılında düzenlenmiştir ve mevcut 17 görev bulunmaktadır[11]:

- Çoklu dillerde Eşgönderge (coreference) Çözümü
- Çapraz-dilsel Sözlüksel Değiştirim
- Çapraz-dilsel KAB
- VP Elipsis-Tesbiti ve Çözümü
- Bilimsel Makalelerden Otomatik Anahra Cümle Çıkarma
- Argüman Seçimi ve Baskısı
- Ad kökenli kelime çiftleri arasındaki Anlamsal İlişkilerin Çok-yollu Sınıflandırılması
- İsim olan bileşik kelimelerin Fiiller Kullanarak Yorumlanması

[11] http://semeval2.fbk.eu/semeval2.php

- Etkinlik ve Katılımcılarının Bağlamda Bağlanması
- Çince Haber Cümlelerindeki Etkinlik Tesbiti
- Metinsel Gerektirim Kullanarak Ayrıştırıcıların Eğitimi ve Değerlendirilmesi
- Kuzey Çin Lehçesi için Metni Konuşmaya Çeviren Sistemlerde Az Bulunan Anlamı Saptama
- Japonca KAB
- Belirli bir alanda Bütün-kelimeler için KAB
- Hassas Çokanlamlı Sıfatların Belirginleştirilmesi

  Bu çalıştay için başkaca ayrıntılı bilgiye erişilememiştir.

5.3 Performans Sonuçları

TMP ile gerçekleştirdiğimiz KAB uygulamamızda elde ettiğimiz duyarlılık ve geriçağırım değerleri Tablo 5.7'de bulunmaktadır. Tablo 5.8'de ise TLST kullanılarak Naive Bayes benzeri istatistiksel bir yöntemle elde edilmiş başka bir çalışmanın sonuçları vardır (Orhan vd., 2007). Bu iki çalışma da kelimelerin kaba taneli anlamları ile elde edilen sonuçlardır.

Kelimeler	**P**	**R**
İsimler	0.71	0.71
Fiiller	0.65	0.79
Diğerleri	0.92	0.84
Ortalama	0.76	0.78

Tablo 5.7 TMP ile elde edilen sonuçlar

Kelimeler	**P**	**R**
İsimler	0.65	0.43
Fiiller	0.56	0.50
Diğerleri	0.57	0.44
Ortalama	0.59	0.46

Tablo 5.8 Naive Bayes benzeri bir yöntemle ile elde edilen sonuçlar

Bu tablolara bakarak TMP yönteminin diğer yönteme göre daha fazla oranda başarılı olduğu söylenebilir. Tablo 5.9'da isim, Tablo 5.10'da fiil, Tablo 5.11'de ise zarf ve sıfat türlerindeki kelimeler için uygulamamızda ile elde ettiğimiz duyarlılık, geriçağırım ve doğruluk değerleri verilmiştir.

İsimler	**Duyarlılık**	**Geriçağırım**	**Doğruluk**
ara	0.51	0.91	0.73
baş	0.68	0.71	0.70
el	0.76	0.83	0.84
göz	0.77	0.75	0.76
kız	1	0.5	0.52
ön	0.87	0.65	0.70
sıra	0.61	0.89	0.77
üst	0.52	0.66	0.61
yan	0.81	0.63	0.66
yol	0.55	0.57	0.57
Ortalama	0.71	0.71	0.69

Tablo 5.9 İsimler için elde edilen duyarlılık, geriçağırım ve doğruluk değerleri

İsimler içinde en yüksek doğruluk değeri "el" kelimesiyle elde edilirken, en düşük doğruluk değeri "kız" kelimesinde elde edilmiştir.

Fiiller	**Duyarlılık**	**Geriçağırım**	**Doğruluk**
al	0.51	0.81	0.69
bak	0.71	0.85	0.74
çalış	0.92	0.58	0.63
çık	0.72	0.79	0.76
geç	0.64	0.76	0.72
gel	0.53	0.72	0.74
gir	0.60	0.83	0.74
git	0.58	0.95	0.78
gör	0.69	0.96	0.83
konuş	0.55	0.67	0.64
Ortalama	0.65	0.79	0.65

Tablo 5.10 Fiiller için elde edilen duyarlılık, geriçağırım ve doğruluk değerleri

Fiil kategorisindeki kelimeler için örnek sayısı genel olarak iyidir. Ancak fiilerin anlam sayıları diğer kelime türlerine göre yüksektir ve bu da belirginleştirme işlemini zorlaştırmaktadır.

Zarflar ve Sıfatlar	**Duyarlılık**	**Geriçağırım**	**Doğruluk**
büyük	0.46	0.63	0.64
doğru	0.79	0.75	0.76
küçük	1	0.5	0.5
öyle	0.74	0.68	0.70
son	0.72	0.72	0.72
tek	0.9	0.9	0.9
Ortalama	0.92	0.84	0.70

Tablo 5.11 Zarflar ve sıfatlar için elde edilen duyarlılık, geriçağırım ve doğruluk değerleri

Kelime türlerine göre elde edilen doğruluk değerlerinin ortalamalarına bakacak olursak birbirlerine yakın olduğunu söyleyebiliriz. En yüksek doğruluk değeri zarflar ve sıfatlar için olmuştur. Bu doğruluk değerine yakın bir değer isim kategorisinde elde edilmiştir. En düşük doğruluk değeri ise fiillerde olmuştur. Zarflar ve sıfatlar için TLST'de bulunan örnek sayısı düşüktür. Özellikle "tek" ve "son" kelimelerinde azdır. Örnek sayısı az olmasına rağmen "tek" kelimesi için en yüksek doğruluk değerine ulaşıldığı görülmektedir. Ancak bu değer yanıltıcıdır, çünkü bu kelimenin anlamları için dengeli bir örnek dağılımı yapılmamıştır.

Şimdi, uygulamada karşılaşılan bir durumdan bahsedilecektir. Bazı hedef kelimeler için yapılan öğrenme sonucu elde edilen hipotezlerden biri aşağıdaki gibi olmuştur:

anlam(A,B):- true, B=1.

Bu hipotez her ne kadar kural gibi görünüyor olsa da aslında aşağıdaki gerçeğe denktir.

anlam(A,1).

Bu kural bize verilen bir cümlede hiçbir özelliğe bakılmaksızın hedef kelime için anlam karşılığının 1. anlamı olacağını söylemektedir. Örneğin "göz" kelimesi test edildiğinde Tabo 5.12'deki değerler elde edilmektedir.

		Gerçek		
Tahmin		**Pozitif**	**Negatif**	
	Pozitif	21	7	28
	Negatif	6	20	26
		27	27	54

Tablo 5.12 "Göz" kelimesi için test performansı

Tabloya bakıldığında test dosyasında bulunan 27 örnek cümle için, pozitif dosyada anlam(A,1) hipotezini gerçekleyen 21 örnek var iken, diğer anlamları için 6 örnek olduğu görülmektedir. Benzer şekilde aynı hipotez negatif dosyada 7 örnek için birinci anlam değerini alırken 20 örnek için de diğer anlamlarını sağlamaktadır.

Sonuç olarak öğrenmeden sonra kelimelerin özelliklerine bağımlı olmayan bir prolog gerçeği ile ifade edilmiş bir genellemeye ulaşılmıştır. Bunun nedeni de elimizdeki kelimelerin birinci anlamının diğer anlamlara oranla daha fazla sayıda bulunmasıdır. Bu dengesizlik, ALEPH sisteminin özelliklerden bağımsız olarak her kelime için "her kelime birinci anlamı taşır." biçiminde aşırı bir genelleme yapmasından kaynaklanmaktadır. Bu aşırılıktan kaçınmak için daha dengeli örnek kümeleri ile çalışılmalıdır. Yapılan uygulamada bazı kelimeler için aşırı genelleme sonucu istenmeyen sonuçlar elde edilmiştir. Çözüm olarak bu kelimelerin aşırı genellemeye neden olan anlamlarına ait örnek sayısının azaltılması yoluna gidilmiştir (Aydın ve Kılıçaslan, 2010b). Bu şekilde yapılan öğrenme sonucu elde edilen sonuçlar Tablo 5.13'de verilmiştir.

Bu tabloda bazı kelimelerin önüne "*" işareti konmuştur. Bu işaretin bulunduğu "baş", "el", "kız", "üst", "bak", "gel", "gir", "git", "büyük" ve "son" kelimeleri aşırı genelleme sebebiyle eğitim örnek sayısı azaltılan kelimelerdir. Diğer kelimeler için aşırı genelleme problemi oluşmadığı için eğitim örnek sayısında bir değişiklik yapılmamıştır.

“Tek” kelimesinin aşırı genellemeye neden olan anlamlarına ait örnek sayısının azaltılması işlemi bu problemin giderilmesini sağlamamıştır. Eğitim örneklerinin az olması ve kelimenin verilen anlamları için belirginleştirmeyi sağlayacak sayıda örneğin bulunmamasıdır. Özellikle 1. anlamı için verilen eğitim örneği sayısı fazladır. “tek” kelimesinin aşırı genelleme problemi çözülemediğinden bu kelime için başarım değerlendirmesinde bulunamayacağız.

Tablo 5.13’e bakarak kelimelerin verilen anlamları içinde öğrenilen anlamlarının miktarı açısından bir değerlendirme yapabiliriz. Bazı kelimelerin tüm anlamları belirlenebilirken, bazı kelimelerde için öğrenilen anlam sayıları düşük olmuştur. Tüm anlamları belirlenebilen kelimeler “kız” ve “son” kelimeleridir. Bu kelimelerde verilen iki anlam da öğrenilebilmiştir ve belirginleştirmenin bu açıdan başarılı olduğunu söyleyebiliriz. “Kız” ve “son” kelimelerinin verilen anlam sayısının az olmasının bu başarıyı elde etmede etkili olduğu söylenebilir. Ayrıca, verilen her anlam için eğitim örneklerinde belirginleştirmeyi sağlayacak miktarda veri olduğu sonucuna da varılabilir. Bu başarıya etki eden bir diğer faktör, seçilen özelliklerin diğer kelimelere göre bu kelimelerin anlamını belirginleştirmede daha etkili olması düşünülmektedir.

Öğrenilen anlamları yüksek olan diğer kelimeler “baş”, “el” ve “göz” kelimeleri olmuştur. Bunların öğrenilen anlamlarının oranı %50’nin üstündedir. Bu kelimelere baktığımız zaman verilen anlam sayılarının düşük olduğu görülmektedir. Anlam sayısının az olması bu başarımda etkili olmuş denilebilir.

Öğrenilen anlamlarının verilen anlamlarına oranı %50 olan, yani anlamlarının yarısı öğrenilebilen kelimeler “yol”, “bak”, “çalış”, “çık”, “büyük”, “doğru”, “küçük”, “öyle” ve “tek” kelimeleridir.

%30-%50 arasında anlamları öğrenilen kelimeler “ara”, “sıra”, “ön”, “üst”, “yan”, “gir”, “gör” ve “konuş” kelimeleridir. Bu kelimelerin anlam sayısı 3 ile 6 arasında değişmektedir. Anlam ayrımları fazla değildir. Bunun da yine belirginleştirilebilen anlam sayısında etkili olduğu söylenebilir.

Kelimeler	Anlam sayısı	Anlamlar[12]	Öğrenilen anlamlar	Eğitim örneği sayısı	Test örneği sayısı	Toplam örnek sayısı
İsimler						
ara	7	1,2,3,4,10,12,13	2.10,12	192	63	255
*baş	5	1,3,4,7,10	1,7,10	56	22	78
*el	3	1,2,5	1,2	106	38	144
göz	3	1,2,4	1,2	92	27	119
*kız	2	1,3	1,3	83	21	104
ön	5	1,2,3,4,6	1,6	72	23	95
sıra	7	1,3,4,5,6,8,9	4,5	85	28	113
*üst	7	1,2,3,4,6,8,9	1,2,4	60	23	83
yan	5	1,2,5,6,10	2,6	65	31	96
yol	6	1,2,5,7,8,11	7,8,11	68	29	97
Fiiller						
al	24	1,2,3,4,5,7,8,9,11,12,13,14,19,21,25,26,27,28,30,32,33,35,37,38	4,7,38	963	125	1088
*bak	4	1,9,17,19	1,17	181	85	266
*çalış	4	1,2,3,5	1,3	102	61	163
çık	6	1,2,4,13,35,43	1,13,35	138	87	225
geç	11	1,2,8,11,17,18,22,28,30,36,39	1,22	164	90	254
*gel	20	1,3,6,7,8,9,11,12,13,15,16,17,21,22,24,26,27,31,38,39	1,11,24,27	244	215	459
*gir	6	1,3,10,12,13,15	1,3	144	84	228
*git	13	1,2,3,4,5,6,7,8,10,14,15,16,21	1,4	143	120	263
gör	5	1,2,4,5,17	1,4	206	68	274
konuş	6	1,2,3,4,5,11	3,4	129	63	192
Zarflar ve Sıfatlar						
*büyük	6	1,2,3,4,5,6	2,4,5	83	26	109
doğru	6	1,2,3,7,8,9	2,8,9	81	38	119
küçük	4	1,2,3,4	1,2	45	14	59
öyle	4	1,2,3,4	1,2	51	23	74
*son	2	1,2	1,2	32	18	50
tek	2	1,2	1	40	10	50

Tablo 5.13 Kelimeler ve öğrenilen anlamları

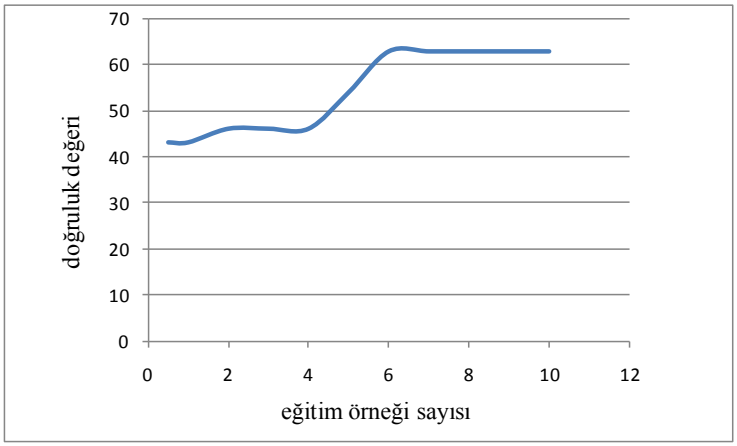
[12] Bu tablodaki anlamlara verilen numaralar TDK Türkçe Sözlüğü'ndeki anlam numaraları ile aynıdır.

Anlamları en düşük belirlenen kelimeler ise “al”, “geç”, “gel” ve “git” olmuştur. Bu kelimelerin hepsi fiil kelime türündedir. Genel olarak fiillerin anlam ayrımları diğer kelime türlerine (isim, zarf, sıfat) göre daha fazladır. Tablodan görüleceği üzere kelimelerin verilen anlam sayıları çok fazladır ve belirginleştirmede bunun olumsuz etkisi olmuştur. Ayrıca buradan kelimelerin belirlenemeyen diğer anlamları için eğitim örneklerinin yeterli olmadığı sonucu da çıkarılabilir.

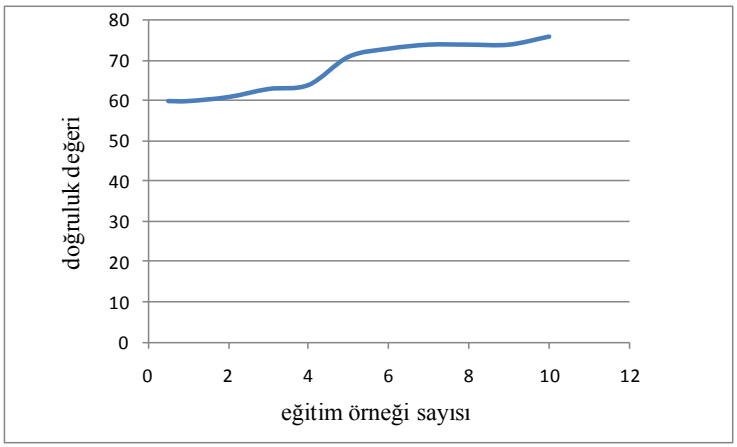
Anlamlara ilişkin yapacağımız değerlendirme için sonuç olarak şunları söyleyebiliriz. Başarımın olumsuz olmasına neden olarak, belli anlamların fazlaca kullanılması ve diğer anlamlara ait örneklerin çok az veya hiç bulunmaması verilebilir. Başka bir deyişle, anlamlar arasındaki dağılım dengesizliği belli bir anlamın seçilmesine sebep olmaktadır. Kelimelerin anlamları için yeterli örnek bulunması durumunda hem başarım oranı artmakta, hem de farklı anlamlar için yeterli örnek bulunduğundan elde edilen sonuçlar daha sağlıklı olmaktadır.

Uygulamada önce eğitim örneklerinin ilk %10’luk dilimindeki örnekler ile öğrenme yapılmaktadır. Sonrasında eğitim sonucu elde edilen kurallar ile tüm test örnekleri üzerinde test işlemi yapılır. Bu test işleminden sonra eğitim örneklerinin ilk %20’lik dilimi ile öğrenme gerçekleştirilir ve elde edilen kurallar ile yine test işlemi gerçekleştirilir. Bu öğrenme ve test işlemleri her adımda eğitim örneklerine %10’luk dilimi eklenerek varolan eğitim örnekleri bitene kadar devam eder. Bu işlemlerin sonunda her kelime için eğitim örnek sayısına göre elde edilen doğruluk değerlerinden grafikler oluşturulmuştur. Bu grafiklerde eğitim örnekleri arttıkça doğruluk değerinin yükseldiğini gösteren bir eğri varsa, bu varolan eğitim örneklerinin sayısının öğrenme için yeterli olduğu anlamına gelir. Bu şekilde yükselen bir eğrinin bulunduğu bir grafikler mutlu çizge olarak isimlendirilir.

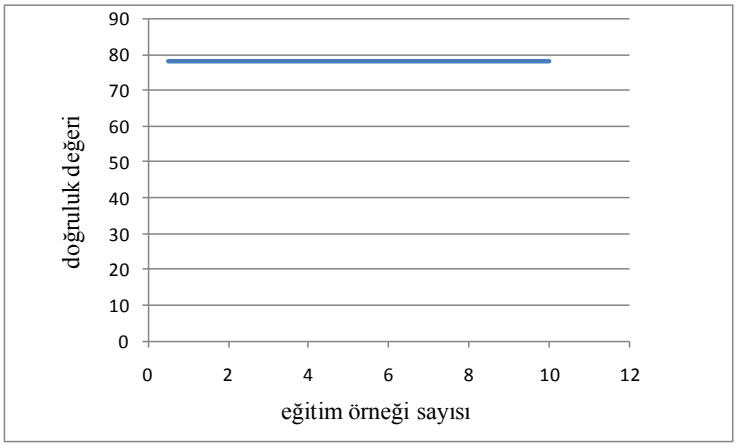
Şimdi, belirginleştirmesini yaptığımız kelimelerden bazıları için elde ettiğimiz grafikleri vereceğiz. Öncelikle fiil kelime türünde bulunan “çalış”, “çık” ve “git” kelimeleri için öğrenme sonucunda elde edilen grafiklere bakalım.



Şekil 5.1 "Çalış" kelimesinin eğitim örneği sayısı ile test örneklerinden elde edilen doğruluk değerleri arasındaki ilişki



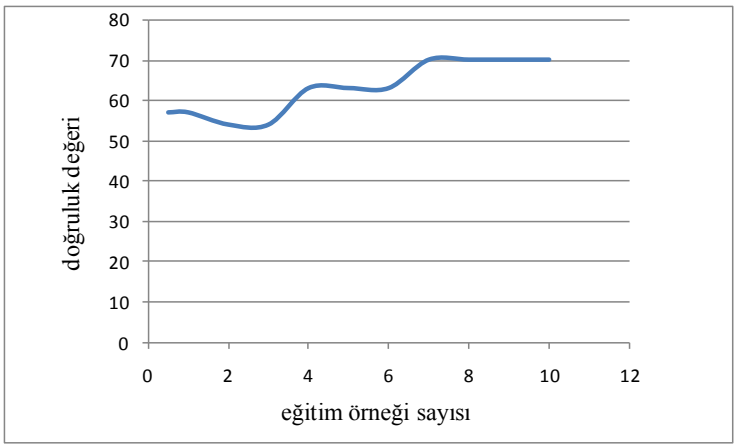
Şekil 5.2 "Çık" kelimesinin eğitim örneği sayısı ile test örneklerinden elde edilen doğruluk değerleri arasındaki ilişki



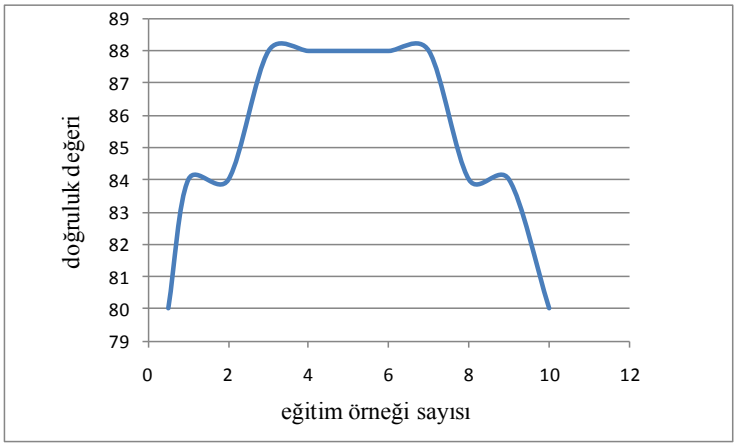
Şekil 5.3 "Git" kelimesinin eğitim örneği sayısı ile test örneklerinden elde edilen doğruluk değerleri arasındaki ilişki

Şekil 5.1 ve Şekil 5.2'deki grafiklerden görüleceği üzere "çalış" ve "çık" kelimeleri için mutlu çizge elde edilmiştir. Buradan eğitim örnek sayısının bu kelimeler için yeterli olduğu sonucu çıkmaktadır. Fakat aynı sonuç, Şekil 5.3'de grafiği verilen "git" kelimesi için geçerli olmamıştır. Eğitim örneklerinin ilk %10'luk dilimi ile tamamı için yapılan öğrenmede doğruluk değerlerinde hiçbir değişiklik olmamıştır.

İsim kelime türündeki "ön" ve "el" kelimeleri için elde edilen grafikler sırasıyla Şekil 5.4 ve Şekil 5.5'de verilmiştir.



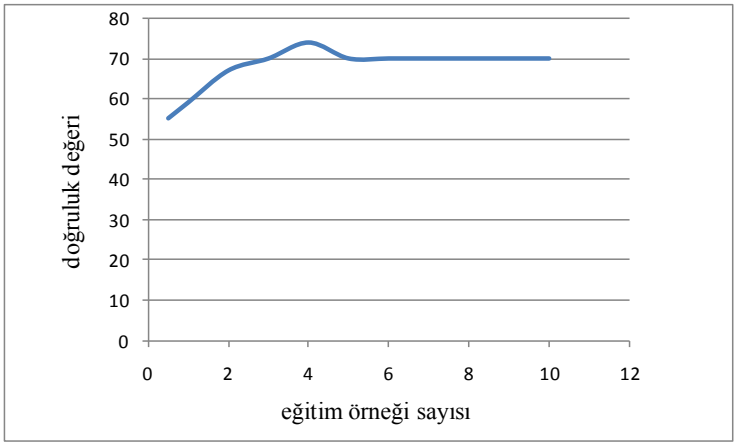
Şekil 5.4 "Ön" kelimesinin eğitim örneği sayısı ile test örneklerinden elde edilen doğruluk değerleri arasındaki ilişki



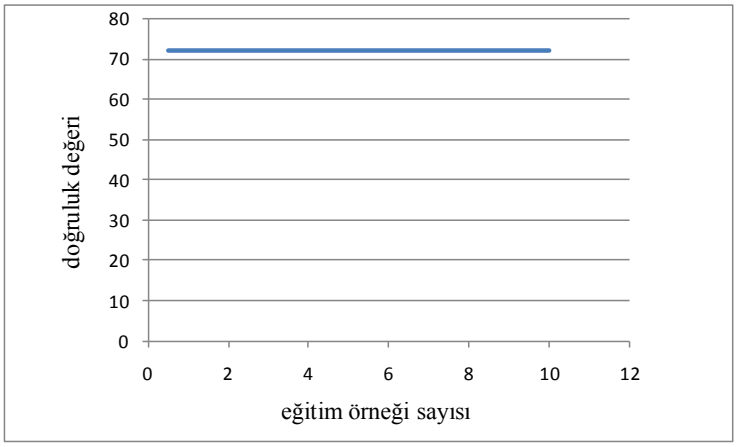
Şekil 5.5 "El" kelimesinin eğitim örneği sayısı ile test örneklerinden elde edilen doğruluk değerleri arasındaki ilişki

"Ön" kelimesi için yapılan öğrenmede eğitim örnek sayısı yeterli olarak görülmektedir. Diğer taraftan "el" kelimesine ait grafiğe bakacak olursak öğrenmenin belli bir değerden sonra düştüğünü görmekteyiz. Bu verinin içinde gürültü (noise) barındırdığını göstermektedir. Gürültü veri birçok tutarsızlığı beraberinde getirmektedir. Veride gürültülü bilginin bulunması aşırı uyum (overfitting) ve yetersiz uyum (underfitting) sorunlarının oluşmasına neden olur. Bu sorunlar makine öğrenmesindeki en önemli sorunlardandır. Üretilen modelin kendisini oluşturan veriye aşırı bir şekilde bağımlı olması ve genellikten uzaklaşması aşırı uyuma sebep olur. Özellikle makine öğrenmesi teknikleri içinde, k-en yakın komşu ve karar ağacı algoritmalarında aşırı uyum sorunu çoğunlukla oluşmaktadır. Çünkü bu iki algoritmanın veriyi ifade etme gücü yüksektir. İfade güçleri o kadar yüksektir ki ne sayıda örnek verildiğine bağlı olmaksızın, sınıf belirsizliği olmadığı sürece, mükemmel sınıflandırma yapabilirler. Bu tabii ki gürültü yaratan ve ait olduğu sınıfın geneline uzak (outlier) örneklere büyük olasılıkla uyum sağlama, yani aşırı uyum yoluyla olacaktır. Bu sorunun giderilmesi için k- en yakın komşu algoritmasında K değerinin arttırılması, karar ağaçlarında ise budama işlemini yapılması gerekmektedir. Uygulamamızda kullandığımız makine öğrenmesi tekniği karar ağaçlarıdır, ancak budama işlemi yapıldığı için aşırı uyum problemi ile karşılaşılmamıştır. "El" kelimesinin grafiğindeki düşüş gürültünün neden olduğu yetersiz uyum sorunudur.

Zarf ve sıfat kelime türündeki “öyle” ve “son” kelimeleri için elde edilen grafikler sırasıyla Şekil 5.6 ve Şekil 5.7’de verilmiştir.



Şekil 5.6 “Öyle” kelimesinin eğitim örneği sayısı ile test örneklerinden elde edilen doğruluk değerleri arasındaki ilişki



Şekil 5.7 “Son” kelimesinin eğitim örneği sayısı ile test örneklerinden elde edilen doğruluk değerleri arasındaki ilişki

Bu grafiklere göre,“öyle” kelimesi için “yapılan öğrenmede eğitim örnek sayısı yeterli olarak görülmektedir. Ancak “son” kelimesinin örnek sayısının yeterli olmadığı söylenebilir.

Sonuç olarak, KAB uygulamalarında kullanılan yöntemlerin başarısını etkileyen çeşitli faktörlerin olduğu söylenebilir. Bu faktörlerden bazıları kelime için verilen eğitim kümesindeki örneklerin sayısı, kelimenin anlamları, örneklerin dağılımı ve kullanılan özellikler olabilmektedir. Yapılan değerlendirmelerde bütün bu faktörler göz önünde bulundurulmalıdır.

6. SONUÇ

Bu çalışma, Türkçe metinlerdeki anlamı belirsiz kelimelere uygulanan KAB işlemi için bazı yaklaşımlarla yapılan uygulamaları anlatmakta ve sonuçlarını sunmaktadır. İlk uygulanan denetimsiz derlem tabanlı KAB yaklaşımı ile elde edilen sonuçlar başarılı görülmemiştir. Bu sebeple doğal dil işleme uygulamalarında başarımı yüksek olan denetimli bir yaklaşımla farklı bir uygulama geliştirilmiştir. Bu uygulamada klasik makine öğrenme yaklaşımlarının artalan bilgisini kullanmadaki ve tümdengelim çıkarım yapabilmedeki yetersizliğini gideren bir yöntem olan TMP kullanılmıştır. Bilgi kaynağı olarak uygulamada TLST'den faydalanılmıştır. Bu kaynağı oluşturanlar tarafından Naive Bayes benzeri bir istatiktiksel yöntemle geliştirilmiş bir KAB uygulaması ve sonuçları bulunmaktadır. Ayrıca TMP'nin KAB uygulamamızındaki başarımını değerlendirmek amacıyla, bu uygulamanın sonuçları ile kendi uygulamamızın sonuçlarının kıyaslaması yapılmıştır. Sonuç olarak TMP yönteminin diğer yönteme göre daha fazla oranda başarılı olduğu görülmüştür. TMP ile elde edilen doğruluk değerleri isim, fiil, zarf ve sıfat kelime türlerindeki kelimeler için diğer uygulamadan daha yüksektir.

Bilgi tabanlı öğrenme modelleri çoğunlukla tümevarımlı mantıkla tasarlanır. Bu şekilde oluşturulan bazı makine öğrenme modelleri ile çeşitli uygulama alanlarında başarılı sonuçlar elde edildiği halde, şu bir gerçektir ki tümevarımlı mantık paradigması artalan bilgisinin öğrenme sistemleri ile birleşmesine izin vermemektedir. Bu makine öğrenme modelleriyle, artalan bilgisinde üstü kapalı olarak bulunan önemli miktardaki bilginin çıkarımı engellenir. TMP'de tümevarımlı ve tümdengelimli mekanizmaların birlikte kullanılması bu probleme bir çözüm getirmiştir.

KAYNAKLAR

Agirre E., Martinez D., 2001, “Learning class-to-class selectional Preferences”, Proceedings of the InternationalConference on Natural Language Learning, Toulouse, France, 15-22.

Agirre, E., Edmonds, P., 2006, “Word Sense Disambiguation: Algorithms and Applications”, Springer.

Almuallim H., Dietterich T. G., 1991, “Learning with many irrelevant features”, Proceedings of the 9th National Conference on Artificial Intelligence, AAAI-91, 547-552.

Amasyalı, M. F., 2006, “Arama motorları kullanarak bulunan anlamsal benzerlik ölçütüne dayalı kelime sınıflandırma”, SIU 2006.

Aristotle, 1960, “Posterior Analytics”, Loeb Edition, translated by Hugh Tredennick.

Atkins, S., 1993, “Tools for computer-aided corpus lexicography: The Hector Project”, Acta Linguistica Hungarica 41, 5-72.

Aydın, Ö., Tüysüz, M. A. A., Kılıçaslan, Y., 2007, "Türkçe için bir Kelime Anlamı Belirginleştirme Uygulaması", XII. Elektrik, Elektronik, Bilgisayar, Biyomedikal Mühendisliği Ulusal Kongresi, Eskişehir Osmangazi Üniversitesi, Eskişehir.

Aydın, Ö., Kılıçaslan, Y., 2009, "Tümevarımlı Mantık Programlamanın Kelime Anlamı Belirginleştirmeye Uygulanabilirliğinin İncelenmesi", IV. İletişim Teknolojileri Ulusal Sempozyumu, Adana, 135-140.

Aydın, Ö., Kılıçaslan, Y., 2010a, "Tümevarımlı Mantık Programlama ile Türkçe için Kelime Anlamı Belirginleştirme Uygulaması", Akademik Bilişim 2010, Muğla.

Aydın, Ö., Kılıçaslan Y., 2010b, “Resolving Lexical Ambiguity in Turkish within a Paradigm Involving Both Inductive and Deductive Techniques: an ILP Application”, INISTA-2010, Kayseri.

Bacon F., 1620, “Novum Organum”.

Banarji, R. B., 1964, “A language for the description of concepts”, General Systems, 9, 135-141.

Banerjee, S., Pedersen T., 2002, “An adapted Lesk algorithm for word sense disambiguation using WordNet” . In: Proceedings of the third international conference on intelligent text processing and computational linguistics, Mexico City, February 17–23, 136–145.

Bergadano, F., Gunetti, D., 1996, “Inductive Logic Programming: From Machine Learning to Software Engineering”, MIT Press.

Bilgin, O., Çetinoğlu, Ö., Oflazer, K., 2004, “Morphosemantic Relations In and Across Wordnets: A Study Based on Turkish”, Proceedings of the Second Global WordNet Conference (GWC 2004), January, Brno, Czech Republic.

Black, E., 1988, “An Experiment In Computational Discrimination Of English Word Senses”, BM Journal Of Research And Development, 32(2), 185-194.

Boser, B. E., Guyon, I.M., Vapnik, V.N., 1992, “A training algorithm for optimal margin classifiers”, In D. Haussler, editor, 5th Annual ACM Workshop on COLT, Pittsburgh, PA. ACM Press. 144-152.

Bouillon, P., 1997, "Polymorphie et semantique lexical: le case des adjectifs", Ph.D., Paris VII. Paris.

Breinman, L., Friedman, L., Olshen, R., Stone, C., 1984, “Classification and Regression Trees”, Wadsworth Inc., Belmont, California.

Briscoe, T., 1991, “Lexical Issues in Natural Language Processing.”, In Klein, Ewan H. and Veltman, Frank (Eds.)., Natural Language and Speech. Springer-Verlag, Berlin, 39-68.

Bruner, J.S., Goodnow, J. J., Austin, G. A., 1956, “A Study of Thinking”, Wiley.

Buntine, W., 1986, “Generalised Subsumption”, In Proceedings of European Conference on Artificial Intelligence. London.

Buntine, 1988, “Generalized Subsumption and its Applications to Induction and Redundancy”, Artificial Intelligence, 36, 149-176.

Burnard, L., 1995, “British National Corpus: User's Reference Guide for the British National Corpus”, Oxford: Oxford University Computing Service.

Busa, F., 1996, “Compositionality and the Semantics of Nominals”, Ph.D. Dissertation, Brandeis University.

Chapman, R., 1977, “Roget’s International Thesaurus (Fourth Edition)”, Harper and Row, New York.

Cohen, B. L., 1978, “A Theory of Structural Concept Formation and Pattern Recognition”, Ph.D. Thesis, Department of Computer Science, University of New South Wales.

Cohen, B. L., Sammut, C. A., 1982, “Object Recognition and Concept Learning with CONFUCIUS”, Pattern Recognition Journal, 15(4), 309-316.

Cowie, J., Guthrie, J., Guthrie, L., 1992, “Lexical Disambiguation using Simulated Annealing”, Computing Research Laboratory, Las Cruces NM, Fifth DARPA Workshop on Speech & Natural Language.

Crawford, S. L., 1990, “Extensions to the CART algorithm”, Machine Learning and Uncertain Reasoning - Knowledge-Based Systems, Vol. 3, Gaines, B and Boose, J. (Eds), Academic Press, London, 15-35.

De Raedt, L., 1992, “Interactive Theory Revision: An Inductive Logic Programming Approach”, Academic Press.

De Raedt, L., 1993, “Inductive logic programming and scientific discovery”, Technical report, Katholieke Universiteit Leuren, Leuren, Belgium.

Dzeroski, S., Bratko, I., 1995, "Applications of inductive logic programming", In:De Raedt,L.(ed.). Advances in ILP. Amsterdam: IOS Press :Ohmsha, 65-81.

Edmonds, P., 2002, "SENSEVAL: The evaluation of word sense disambiguation systems", ELRA Newsletter, Vol. 7 No. 3, 5-14.

Escudero, G., Màrquez, L., Rigau, G., 2000, "Naive Bayes and Exemplar-Based approaches to Word Sense Disambiguation Revisited", In Proceedings of the 14th European Conference on Artificial Intelligence, ECAI'00. Berlin, Germany.

Fellbaum, C., 1998, "WordNet: An Electronic Lexical Database", The MIT Press.

Friedman, N., Geiger, D., Goldszmidt, M., 1997, "Bayesian Network Classifiers", Machine Learning, 29 (2), 131-163.

Gale W., Church K. W., Yarowsky D., 1992a, "Estimating upper and lower bounds on the performance of word-sense disambiguation programs", Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics (ACL), Newark, Delaware, 249-256.

Gale, W., Church K., Yarowsky D., 1992b, "A method for disambiguating word senses in a large corpus", Computers and the Humanities 5-6, 415-439.

Gale, W., Church K. ve Yarowsky D.,. 1992c, "One sense per discourse", In Proceedings of the DARPA Speech and Natural Language Workshop.

Galley, M., McKeown, K., 2003, "Improving word sense disambiguation in lexical chaining", Proceedings of the International Joint Conference in Artificial Intelligence (IJCAI), Acapulco, Mexico, 1486-1488.

Gonçalo Oliveira, H., 2009, "Ontology learning for portuguese", In 2nd Doctoral Symposium on Artificial Intelligence.

Grishman, R., MacLeod, C., Meyers, A., 1994, "COMLEX syntax: Building a Computational Lexicon.", Proceedings of the 15th International Conference on Computational Linguistics, COLING'94, Kyoto, Japan, 268-272.

Güner, E.S., 2008, "Türkçe İçin Derlem Tabanlı Bir Anafor Çözümleme Çalışması", Yüksek Lisans Tezi, Trakya Üniversitesi Fen Bilimleri Enstitüsü.

Güzel Türkçemiz İnternet Sitesi, http://guzelturkcemiz.org.

Halliday, M.A.K., Hasan, R., 1976, "Cohesion in English", London: Longman.

Haruno, M., Shirai, S., Ooyama, Y., 1998, "Using Decision Trees To Construct A Practical Parser", In Proceedings Of The Joint 17th International Conference On Computational Linguistics And 36th Annual Meeting Of The Association For Computational Linguistics (COLING-ACL), Montreal, Canada, 1136-1142.

Herman, I., 2007, "3C Semantic Web Activity", Retrieved June 12, 2007, from World Wide Web Consortium, web site: http://www.w3.org/2001/sw/

Hirst, G., St-Onge, D., 1998, "Lexical chains as representations of context for the detection and correction of malapropisms", In: Fellbaum, Christiane, ed., WordNet: An Electronic Lexical Database, MIT Press.

Hornby, A.S., 2000, "Oxford Advanced Learner's Dictionary", Oxford University Press, Oxford.

Hoste, V., Kool, A., Daelemans, W., 2001, "Classifier Optimization and Combination in the English All Words Task", Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems, SENSEVAL-2, Toulouse, France, 83-86.

Hoste, V., Daelemans, W., Hendrickx, I., van den Bosch, A., 2002, "Evaluating the Results of a Memory-Based Word-Expert Approach to Unrestricted Word Sense Disambiguation", In: Proceedings of the Workshop on Word Sense Disambiguation: Recent Successes and Future Directions, 95-101.

Hume, D., 1748, "An Enquiry Concerning Human Understanding".

Ide, N., Veronis, J., 1998, "Word Sense Disambiguation: The State of the Art", Computational Linguistics.

Jiang, J.J., Conrath, D.W., 1997, "Semantic similarity based on corpus statistics and lexical taxonomy", In: Proceedings of ROCLING X, Taiwan.

Johansson, S., 1980, "The LOB corpus of British English texts: presentation and comments.", ALLC Journal, 1(1), 25-36.

Jurafsky, D., Martin, J.H., 2009, "Speech and Language Processing", Second Edition, Pearson, Prentice Hall.

Kennedy, G., 1998, "An Introduction to Corpus Linguistics", Longman.

Kılıçaslan, Y., 1998, "A Form-Meaning Interface for Turkish", Ph.D. Dissertation, University of Edinburgh.

Kılıçaslan Y., Güner, E.S., Uçar, E., 2010, "Using Formal Concept Analysis to Improve Machine Translation", Proceedings of International Symposium on Computing & Engineering (ISCSE), Kuşadası.

Kipke, U., Wille, R., 1987, "Formale Begriffsanalyse erläutert an einem Wortfeld", LDV–Forum, 5.

Kononenko I., 1994, "Esimating Attributes : Analysis and Extensions of RELIEF", European Conference on ML, 171-182.

Kucera, H., Francis, W. N., 1967, "Computational Analysis of Present-Day American English", Brown University Press, Providence.

Kurudayıoğlu, M. ve Karadağ, Ö., 2005, "Kelime Hazinesi Çalışmaları Açısından Kelime Kavramı Üzerine Bir Değerlendirme" GÜ, Gazi Eğitim Fakültesi Dergisi, C.25, S.2(2005), 293-307.

Larson, J., Michalski, R.S., 1977, "Inductive inference of VL decision rules," SIGART Bull., 38-44.

Lavrac, N., Dzeroski, S., Grobelni, M., 1991, “Learning nonrecursive definitions of relations with LINUS”, In Y. Kodratoff, editor, Proceedings of the 5th European Working Session on Learning, volume 482 of Lecture Notes in Artificial Intelligence, 265-281.

Lavrac, N., Dzeroski, S., 1994, “ Inductive Logic Programming: Techniques and Applications”, Ellis Horwood, New York.

Leacock C., Towell G., Voorhes E., 1993, “Towards Building Contextual Representations of Word Senses Using Statistical Models”, Proceedings of SIGLEX workshop “Acquisition of Lexical Knowledge from Text” ACL.

Leacock, C., Chodorow, M., Miller, G., 1998, “Using corpus statistics and WordNet relations for sense identification”, Computational Linguistics, 24(1), 147-165.

Lenat, D., Guha, R., 1990, “Building Large Knowledge-based Systems”, Addison-Wesley, Reading, Massachusetts.

Lesk, M., 1986, “Automated SenseDisambiguation Using Machine-readable Dictionaries”, Proceedings of the SIGDOC Conference.

Li, Y., Bandar, Z. A., McLean, D., 2003, “An approach for measuring semantic similarity between words using multiple information sources”, IEEE Transactions on Knowledge and Data Engineering, 15 (4), 871–882.

Lin, D., 1997, “Using syntactic dependency as local context to resolve word sense ambiguity”, In 35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics (ACL/EACL 1997), 64-71.

Manning, C. D., Schütze. H., 1999, “Foundations of Statistical Natural Language Processing”, Cambridge MA: MIT Press.

Marquez, L., 1999, “Part-Of-Speech Tagging: A Machine Learning Approach Based On Decision Trees”, Ph.D. Thesis, Dep. Llenguatges I Sistemes Informatics, Universitat Politecnica De Catalunya.

Masterman, M., 1957, “The Thesaurus in Syntax and Semantics.”, Mechanical Translation, 4, 1-2.

McCarthy, D., Carroll, J., 2003, “Disambiguating nouns, verbs and adjectives using automatically acquired selectional preferences”, Computational Linguistics, 29(4), 639-654.

Michiels, A., Mullenders, J., Noel, J., 1980, “Exploiting a large data base by Longman”, In: Proceedings of the 8th conference on Computational linguistics, Morristown, NJ, USA, Association for Computational Linguistics, 374-382.

Mihalcea, R., Moldovan, D., 2000, “An iterative approach to word sense disambiguation”, Proceedings of Florida Artificial Intelligence Research Society, Orlondo, U.S.A., 219-223.

Miller, G. A., 1985, "WordNet: A dictionary browser. In: Information in Data", Proceedings of the First Conference of the UW Centre for the New Oxford English Dictionary. Waterloo, Canada: University of Waterloo, 25-28.

Mooney, R.J., 1996, "Comparative Experiments on Disambiguating Word Senses: An Illustration of the Role of Bias in Machine Learning", In Proceedings of the 1st Conference on Empirical Methods in Natural Language Processing, EMNLP'96, 82-91.

Muggleton, S., Buntine, W., 1988, "Machine invention of first order predicates by inverting resolution.", In Proceedings of the 5th International Workshop on Machine Learning, 339-351.

Muggleton, S., Feng, C., 1990, "Efficient induction of logic programs", In Proceedings of the First Conference on Algorithmic Learning Theory, Arikawa, S., Goto, S., Ohsuga, S. and Yokomori, T., Eds. Japanese Society for Artificial Intelligence, Tokyo, 368-381.

Muggleton, S., 1990, "Inductive logic programming", In: Proc. 1st Conference on Algorithmic Learning Theory, Ohmsma, Tokyo, Japan, 43–62.

Muggleton, S., 1991, "Inductive logic programming", New Generation Computing, 8, 295-318.

Muggleton, S., De Raedt, L., 1994, "Inductive logic programming: Theory and methods.", J. Logic Program.

Navigli, R., Litkowski, K., Hargraves, O., 2007, "Coarse-grained english all-words task", In Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007), Prague, Czech Republic, 30–35.

Nida, E., 1975, "A Framework for the Analysis and Evaluation of Theories of Translation", in Brislin, R. W. (ed) (1975), Translation Application and Research, Gardner Press, New York.

Ng, H. T., Lee, H.B., 1996, "Integrating Multiple Knowledge Sources to Disambiguate Word Sense: An Exemplar-based Approach", In Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics, ACL'96.

Nienhuys, S. H., Wolf, R.de, 1997, "Foundations of Inductive Logic Programming", Springer-Verlag, New York.

Oflazer, K., Say, B., Tur, D. Z. H., Tur, G., 2003, "Building A Turkish Treebank, Invited Chapter In Building and Exploiting Syntactically-Annotated Corpora", Anne Abeille Editor, Kluwer Academic Publishers.

Okumura, M., Honda, T., 1994, "Word sense disambiguation and text segmentation based on lexical cohesion", Proceedings of the International Conference on Computational Linguistics (COLING), Kyoto, Japan, 775-761.

Old, L. J., Priss, U., 2001, "Metaphor and Information Flow", In Proceedings of the 12th Midwest Artificial Intelligence and Cognitive Science Conference, 99-104.

Old, L. J., Priss, U., 2001, "Metaphor and Information Flow", In Proceedings of the 12th Midwest Artificial Intelligence and Cognitive Science Conference, 99-104.

Olney, J., Revard, C., Ziff , P., 1967, "Summary of some computational aids for obtaining a formal semantic description of english", In Proceedings of the 1967 conference on Computational linguistics.

Orhan, Z., 2006, "Türkçe Metinlerdeki Anlam Belirsizliği Olan Sözcüklerin Bilgisayar Algoritmaları ile Anlam Belirginleştirmesi", Doktora Tezi, İstanbul Üniversitesi Fen Bilimleri Enstitüsü.

Orhan, Z., Çelik, E., Demirgüç, N., 2007, "SemEval-2007 Task 12: Turkish Lexical Sample Task", Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007), Prague, Czech Republic, 59-63.

Quinlan, J. R., 1986, "Induction of Decision Trees", Machine Learning, Vol.1, No.1, 81-106.

Quinlan, J. R., 1990, "Learning logical definitions from relations.", Machine Learning, 5(3), 239–266.

Quinlan, J.R., 1993, "C4.5: Programs For Machine Learning", Morgan Kaufmann.

Palmer M., Ng H. T., Dang H. T., 2006, "Evaluation of WSD Systems in Agirre and Edmonds (eds.) Word Sense Disambiguation: Algorithms and Applications", Springer, 75-106.

Patrick, A. B., 1985, "An Exploration of Abstract Thesaurus Instantiation", MSc. Thesis, University of Kansas, Lawrence, Kansas.

Patwardhan, S., Banerjee, S., Pedersen, T., 2003, "Using measures of semantic relatedness for word sense disambiguation", In Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics, Mexico City.

Pedersen T., Bruce, R., 1997, "A New Supervised Learning Algorithm for Word Sense Disambiguation", Proceedings of the Fourteenth National Conference on Artificial Intelligence, AAAI'97, Providence, RI.

Plotkin, G. D., 1970, "A Note on Inductive Generalization", In B. Meltzer & D. Michie (Eds.), Machine Intelligence 5., Edinburgh University Press, 153-163.

Plotkin, G. D., 1971a, "Automatic Methods of Inductive Inference", Ph.D. Thesis, Edinburgh University.

Plotkin, G. D., 1971b, "A further note on inductive generalization", In B. Meltzer & D. Michie (Eds.), Machine Intelligence 6. New York: Elsevier.

Pradhan, S., Loper, E., Dligach, D., Palmer, M., 2007, "English lexical sample, SRL and all words", In Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007), Prague, Czech Republic, 87–92.

Priss, U., 2005, "Linguistic Applications of Formal Concept Analysis", Ganter; Stumme; Wille (eds.), Formal Concept Analysis, Foundations and Applications, Springer Verlag, LNAI 3626, 149-160.

Pustejovsky, J. ve B. Boguraev., 1993, "Lexical Knowledge Representation and Natural Language Processing", in Artificial Intelligence, 63, 193-223.

Rada, R., Mili, H., Bickell, E., Blettner, B., 1989, "Development and application of a metric on semantic nets", IEEE Transactions on Systems, Man and Cybernetics, 19, 17-30.

Resnik, P., 1995, "Using information content to evaluate semantic similarity in a taxonomy", In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, Montreal.

Resnik, P., 1995, "Disambiguating Noun Groupings with Respect to WordNet Senses", In Third Workshop on Very Large Corpora, ACL.

Robinson, J. A., 1965, "A Machine-Oriented Logic Based on the Resolution Principle", Journal of the ACM (JACM), Volume 12, Issue 1, 23–41.

Sanderson, M., 1996, "Word Sense Disambiguation and Information Retrieval", Ph.D.Thesis, University of Glasgow, Glasgow.

Say, B., Zeyrek, D., Oflazer K., Özge, U., 2002, "Development of a Corpus and a Treebank for Presentday Written Turkish", in Proceedings of the Eleventh International Conference of Turkish Linguistics.

Schütze, H., 1998, "Automatic word sense discrimination", Computational Linguistics, 24(1), 97-123.

Sedelow, S. Y., Sedelow, W. A. Jr., 1969, "Categories and procedures for content analysis in the humanities.", In Gerbner, George; Holsti, Ole; Krippendorf, Klaus; Paisley, William J.; and Stone, Philip J. (Eds.), The Analysis of Communication Content, John Wiley & Sons, New York, 487-499.

Shapiro, E.Y., 1983, "Algorithmic Program Debugging", The MIT Press.

Snyder B., Palmer M., 2004, "The english all-words task", Senseval-3: The Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text, Barcelona, Spain, 41-43.

Sparck Jones, K., 1964, "Synonymy and semantic classification", Ph. D. Thesis, University of Cambridge, Cambridge, England.

Sparck Jones, K., 1986, "Synonymy and semantic classification", Edinburgh, Edinburgh University Press, England.

Specia, L., Nunes, M.G.V., Srinivasan, A., Ramakrishnan, G., 2007, "Word Sense Disambiguation using Inductive Logic Programming", Proceedings of the 16th International Conference on ILP, Springer-Verlag.

Srinivasan, A., 1999, "The Aleph User Manual", http://www.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/.

Strapparava, C., Gliozzo, A., Giuliano, C., 2004, "Pattern abstraction and term similarity for word sense disambiguation: IRST at Senseval-3", Proceedings of Senseval-3: Third International Workshop on the Evaulation of Systems for the Semantic Analysis of Text, Barcelona, Spain, 229-234.

Stevenson, M., Wilks, Y., 2001, "The interaction of knowledge sources in word sense disambiguation", Computational Linguistics, 27(3), 321–349.

Tanaka, H., 1996, "Decision Tree Learning Algorithm with Structured Attributes: Application to Verbal Case Frame Acquisition", In Proceedings of the 16th International Conference on Computational Linguistics, COLING'96, 943-948,

TDK Türkçe Sözlük, http://www.tdk.gov.tr

Trakya Derlemi, http://tbbt.trakya.edu.tr/download/corpus.htm

Tüysüz, M.A.A., 2010, "Denetlemeli Kelime Anlamı Belirginleştirmede Kullanılan Özelliklerin Ayırtediciliğinin Biçimsel Kavram Analizi Yardımı İle Değerlendirilmesi", Doktora Tezi, Trakya Üniversitesi Fen Bilimleri Enstitüsü.

Vapnik, V., Lerner, A., 1963, "Pattern Recognition using Generalized Portrait Method", Automation and Remote Control. 24:6.

Vasilescu, F., Langlais, P., Lapalme, G., 2004, "Evaluating variants of the Lesk approach for disambiguating words", LREC 2004.

Vauquois, B., 1968, "A survey of formal grammars and algorithms for recognition and transformation in mechanical translation", IFIP Congress (2) 1968, 1114-1122.

Wang, Z., Webb, G. I., 2002, "Comparison Of Lazy Bayesian Rule And Tree-Augmented Bayesian Learning", In Proceedings Of The IEEE International Conference On Data Mining, ICDM-2002, Maebashi, Japan, 775–778.

Weiss, S. M., Apte, C., Damerau, F. J., Johnson, D. E., Oles, F. J., Goetz, T., Hampp, T., 1999, "Maximizing Text-Mining Performance", IEEE Intelligent Systems, 14(4), 63-69.

Wu, Z., Palmer, M., 1994, "Verb semantics and lexical selection", In Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics, Las Cruces, New Mexico.

Yarowsky, D., 1992, "Word-sense Disambiguation Using Statistical Models of Roget's Categories Trained on Large Corpora", Proceedings of 14th International Conference on Computational Linguistics, COLING-92.Nantes, 454-460.

Yarowsky, D., 1993, "One sense per collocation", Proceeding of ARPA Human Language Technology Workshop, Princeton, New Jersey, 266-271.

Yarowsky, D., 1995, "Unsupervised word-sense disambiguation rivaling supervised methods", In Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL '95), 189–196.

Yıldırım, S., 2008, "Türkçe Derlemlerdeki Artgönderimlerin Tümdengelimli ve Tümevarımlı Yöntemlerle Çözümlenmesi", Doktora Tezi, Trakya Üniversitesi Fen Bilimleri Enstitüsü.

Yu L., Liu H., 2004, "Efficient Feature Selection via Analysis of Relevance and Redundancy", Journal of Machine Learning Research.

Zargan İngilizce Sözlük, http://www.zargan.com/

Zhong, N., Dong, J., Ohsuga, S., 2001, “Rule discovery by soft induction techniques”, Neurocomputing 36, 171-204.

ÖZGEÇMİŞ

Özlem AYDIN 1979 yılında Konya'da doğdu. İlk ve orta öğrenimini Tekirdağ'da tamamladıktan sonra 1997 yılında girdiği Trakya Üniversitesi Mühendislik-Mimarlık Fakültesi Bilgisayar Mühendisliği Bölümü'nden 2001 yılında mezun oldu. 2001 yılında Trakya Üniversitesi Mühendislik Mimarlık Fakültesi Bilgisayar Mühendisliği Bölümü'nde Araştırma Görevlisi kadrosuna atandı ve 2002 yılında Trakya Üniversitesi Fen Bilimleri Enstitüsü'ne bağlı olarak Yüksek Lisans eğitimine başladı. Yüksek lisansını 2005 yılında başarıyla bitirdi. 2005 yılında Trakya Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Ana Bilim Dalı'nda doktora çalışmalarına başladı. Özlem AYDIN halen Trakya Üniversitesi Mühendislik Mimarlık Fakültesi Bilgisayar Mühendisliği Bölümü'nde Araştırma Görevlisi olarak çalışmaktadır.