YILDIZ TEKNİK ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

Türkçe Met. Ben. Hes. için Yeni bir Yön.

YÜKSEK LİSANS TEZİ

Bünyamin Dursun

2008

YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

TÜRKÇE METİNLERİN BENZERLİĞİNİN HESAPLANMASI İÇİN YENİ BİR YÖNTEM

Bilgisayar Mühendisi Bünyamin DURSUN

FBE Bilgisayar Mühendisliği Ana Bilim Dalında
Hazırlanan

YÜKSEK LİSANS TEZİ

Tez Danışmanı : Prof Dr. A. Coşkun Sönmez

İSTANBUL, 2008

YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

TÜRKÇE METİNLERİN BENZERLİĞİNİN HESAPLANMASI İÇİN YENİ BİR YÖNTEM

Bilgisayar Mühendisi Bünyamin DURSUN

F.B.E Bilgisayar Bilimleri Ana Bilim Dalında Hazırlanan

YÜKSEK LİSANS TEZİ

Tez Danışmanı : Prof Dr. A. Coşkun Sönmez

Jüri Üyesi : Yrd. Doç. Dr. Banu Diri

Jüri Üyesi : Yrd. Doç. Dr. Şule Gündüz Öğüdücü

İSTANBUL, 2008

İÇİNDEKİLER

SİMGE LİSTESİ

s	N-Gram Uzunluğu yöntemine göre metinlerin benzerlik oranı
m	İki metinde de birbirini tutan 3lü karakter katar sayısı
a	Birinci metindeki 3lü karakter katar sayısı
b	İkinci metindeki 3lü karakter katar sayısı
s	Jaro-Winkler yöntemine göre metinlerin benzerlik oranı
m	Jaro-Winkler yöntemine göre eşleşen karakter sayısı
a	Birinci metinin uzunluğu
b	İkinci metinin uzunluğu
t	Transpozisyon sayısı

KISALTMA LİSTESİ

BME	Bulanık Metin Eşleme
BMET	Bulanık Metin Eşleme Teknikleri
DDİ	Doğal Dil İşleme
ED	Edit Distance
EDS	Edit Distance Similarity
JW	Jaro-Winkler
LED	Levenshtein Edit Distance
MB	Metin Benzerlik
ME	Metin Eşleme
MBHT	Metin Benzerlik Hesaplama Teknikleri
MET	Metin Eşleme Teknikleri
KT	Konuşma Tanıma
OCR	Optical Character Recognition

ŞEKİL LİSTESİ

Sayfa

ÇİZELGE LİSTESİ

Sayfa

ÖNSÖZ

Başta, bu tez çalışmam süresince bana her türlü desteği veren ve gerekli yönlendirmeleri yapan, bilgi ve deneyimi ile bana her zaman yol gösteren ve tezime son şeklini vermemde yardımcı olan değerli hocam Prof.Dr. A.Coşkun Sönmez'e çok teşekkür ederim.

Tez çalışmamım içeriğinin zenginleştirilmesi için değerli fikirlerini esirgemeyen Yrd.Doç.Dr. Banu Diri'ye ve yüksek lisans eğitimim süresince bana emeği geçen diğer tüm Yıldız Teknik Üniversitesi Bilgisayar Bölümü öğretim üyelerine de teşekkürü bir borç bilirim.

Eğitimim konusunda her türlü özveriyi gösteren ve her zaman destek veren aileme ve sevgili eşime...

ÖZET

Verilen iki metinin birbirleri ile benzerlik oranını hesaplamak için çok sayıda yöntem bulunmaktadır. Bu yöntemlerin bazıları doğrudan metinlerin benzerliklerini klasik yöntemlere göre hesaplarken, diğer bazı yöntemler ise daha akıllı bir şekilde çalışarak daha doğru ve insan zekasına yakın benzerlikler hesaplayabilmektedirler. Bu ikinci kısım yöntemler genel olarak Bulanık Metin Benzerliği olarak adlandırılmaktadır.

Bulanık metin eşleme yöntemleri genellikle İngilizce dili ve İngilizce metinler düşünülerek geliştirildiğinden, İngilizce metinler için yüksek başarı gösterseler bile Türkçe metinlerde çoğu kez bu kadar başarılı sonuçlar üretememektedirler.

Bu nedenle bu çalışmada Türkçe metinlerin eşlenmesinde ve benzerliklerinin hesaplanmasında sık karşılaşılan bazı hata durumları modellenerek yeni bir benzerlik hesaplama yöntemi geliştirilmiştir. Bu yöntem özellikle yazım yanlışlarını algılayıp, metinlerin benzerliklerini daha tutarlı bir şekilde hesaplamaktadır. Burada *metin* olarak ifade edilen kavram, birkaç harften oluşan bir kelime olabileceği gibi yüzlerce kelimeden oluşan paragraf gibi uzun bir metin parçası da olabilir.

Geliştirilen bu yöntemin başarısını ölçmek için farklı seviyelerde ve farklı özelliklere sahip bilgisayar kullanıcılarından, farklı şekillerde veri girişi yapmaları talep edilerek, kullanıcıların hatalı girdikleri bu veriler kullanılmıştır. Bu kullanıcıların hatalı girdikleri metinler ve bu metinlerin doğru hallerinden oluşan metin çiftlerinin benzerlik oranları, Geliştirilen yöntem, Edit Distance Benzerliği ve Jaro-Winkler Benzerliği olmak üzere 3 farklı yöntem ile hesaplanarak, karşılaştırmalı olarak başarıları ölçülmüştür. Ayrıca bahsedilen 3 yöntemi kullanarak, herhangi bir Oracle veritabanı sisteminde bulunan tablolardaki tekrarlı veya benzer kayıtları bulan bir yazılım gerçekleştirilmiştir.

Yapılan bu çalışma Türkçe Doğal Dil İşleme, veritabanı sistemlerinde bulunan benzer kayıtların bulunması, Türkçe işletim sistemi, Türkçe arama motorları, entegrasyon projeleri ve e-Devlet çalışmalarında faydalı olabilir.

Anahtar Kelimeler: Türkçe, Türkçe doğal dil işleme, Metin benzerliği, Yazım hataları, Oracle, Tekrarlı kayıt, Entegrasyon, Klavye diziliş

ABSTRACT

To compute similarity rate of two texts, there are many methods. While some of these methods compute the similarity of texts by classical methods, some others can compute similarities more correctly and more closely to human brain by working more intelligently. The ones mentioned as the second are called Fuzzy String Similarity.

Fuzzy string matching methods are generally improved through thinking English and texts in English. Thus, even they are succesfull for the texts in English, they usually may not be perform as the same in Turkish texts.

Therefore, in this work a new similarity computing technic has been improved by modeling some cases oftenly encountered in matching Turkish texts and computing similarities. Especially, an intelligent method has been developed that perceives slips in spelling, arranges them, and computes similarity. Here, the concept mentioned as *text* may be both a word composed of a few letters and a long text composed of hundreds of words.

In order to measure the success of the method improved, different computer users who have different specialities, at different levels, were asked to type different texts, and then these wrongly typed inputs were used. The similarity percent of the texts users typed false and their correct forms was computed by 3 different methods; The improved method, Edit Distance Similarity and Jaro-Winkler Similarity. So, their success was measured comparatively. Also, a software that finds repeated or similar records on the tables in any Oracle database by using 3 methods mentioned above was developed.

It is hoped that this work may contribute to the common good of e-government workings with the aim of Turkish Natural Language Processing, finding similar records available in database systems, Turkish operating system, Turkish search engines, integration projects and providing integration between establishments.

Keywords: Turkish, Turkish natural language processing, String similarity, Slips in spelling, Oracle, Repeated records, Integration, Keyboard layout

1. GİRİŞ

Bilişim dünyasında en sık yapılan işlemlerden bir tanesi belki de en sık yapılanı daha önce ilgili sisteme kaydedilmiş bir veriye veya bilgiye erişmektir. Aranılan bu bilgiye erişmek için ise genellikle ilgili sistemde, sistemin bize sunduğu arayüz üzerinden arama yapılmaktadır. Bu arama işlemleri sırasında ise metin arama, metin bulma, metin eşleme ve metin benzerlik hesaplama tekniklerinden faydalanılır. Başta işletim sistemi olmak üzere birçok Windows uygulaması, Web uygulaması ve veritabanı uygulaması bu gibi teknikleri kullanır. Mesela bir işletim sisteminde veya dosya sisteminde dosya arama işlemi yapılırken metin arama ve metin eşleme teknikleri kullanıldığı gibi, Google gibi günümüzün popüler internet arama motorları ve kütüphaneler gibi arşiv tarama sistemleri yine metin eşleme ve metin benzerlik hesaplama tekniklerini kullanarak çalışır. Ve yine günümüzün vazgeçilmez teknolojilerinden olan veritabanı sistemlerinde özellikle ilişkisel veritabanı sistemlerinde metin eşleme, metin arama ve metin benzerlik hesaplama teknikleri çok yoğun bir şekilde kullanılmaktadır.

Verilen iki metinin birbirleri ile benzerlik oranını hesaplamak için çok sayıda yöntem ve algoritma bulunmaktadır. Bu yöntemlerin bazıları doğrudan metinlerin benzerliklerini klasik yöntemlere göre hesaplarken, bazı yöntemler ise daha akıllı bir şekilde çalışarak daha doğru ve insan zekasına yakın benzerlikler hesaplayabilmektedirler. Bu ikinci kısım yöntemler genel olarak Fuzzy String Similarity (Jin ve Li, 2005) veya dilimizde Bulanık Metin Benzerliği Hesaplama olarak adlandırılmaktadır.

Bulanık metin eşleme yöntemleri genellikle İngilizce dili ve İngilizce metinler düşünülerek geliştirildiğinden İngilizce metinler için yüksek başarı gösterseler bile Türkçe metinlerde çoğu kez bu kadar başarılı sonuçlar üretememektedirler.

Bu nedenle bu çalışmada Türkçe metinlerin eşlenmesinde ve benzerliklerinin hesaplanmasında sık karşılaşılan bazı durumlar modellenerek yeni bir benzerlik hesaplama yöntemi geliştirilmiştir. Bu yöntem özellikle yazım yanlışlarını algılayıp düzelttikten sonra benzerlik hesaplayan daha akıllı bir yöntemdir. Bu çalışmada *metin* olarak ifade edilen kavram, birkaç harften oluşan bir kelime olabileceği gibi yüzlerce kelimeden oluşmuş bir büyük metin parçası da olabilir. Yani burada amaç, değişik sayıda kelimeden oluşan ve değişik uzunlukta metinlerin birbirleri ile olan benzerliğinin hesaplanabilmesi için yeni bir yöntem geliştirmektir.

Bu çalışmanın teknik olarak daha önce İngilizce metinler veya diğer dillerdeki metinler üzerinde yapılan çalışmalardan çok önde olması beklenmemektedir. Ama Türkçe metinler üzerinde daha iyi sonuçlar verecek metodlar geliştirilmiştir. Bu nedenle bu çalışma aslında Bulanık Metin Eşleme Tekniklerinin (BMET) bir lokalizasyonu çalışması olarak tanımlanabilir.

Böyle bir çalışmaya duyulan ihtiyaçlardan en önemlileri büyük veriler içerisinde aranılan bir veriye daha kolay ulaşmayı sağlamak ve değişik bilgi sistemlerinde istenmeyen veri tekrarını azaltmaktır. Mesela günlük hayatta sabit diskimizdeki bir veriye ulaşmaya çalışırken veya kütüphane gibi bir arşiv sistemindeki bir kayıta ulaşmaya çalışırken genellikle aradığımız şeyi ilk seferinde bulamayız. Ayrıca özellikle banka, holding veya kamu kuruluşlarında bulunan büyük verileri daha kolay ilişkilendirmek veya tekrarların önüne geçmek isteriz. Aradığımız veriye ulaşmayı sağlayacak anahtar kelimelerin bazı harflerini değiştirerek veya çoğu zamanda anahtar kelimeyi değiştirerek tekrar deneriz. Bunu bilgisayar kullanan her kişinin günde birkaç sefer karşılaştığı bir durum olduğu varsayımı altında milyonlarca bilgisayar kullanıcısının sırf bu nedenle kaybettiği vakit göz önünde bulundurulduğunda ortaya şaşırtıcı büyüklükte vakit kayıpları çıkacaktır. Bu nedenle aranılan bilgiye en az hamlede erişmenin önemi gittikçe artmaktadır.

1.1 Metin Benzerlik Hesaplama Tekniklerine Duyulan İhtiyaç

İnsan doğadaki varlıkları birbirleri ile benzerlik veya zıtlıklarına göre ilişkilendirip anlamlandırdığı ve kullandığı gibi dijital dünyada da objeler (metin, resim, imaj, vs.) birbirleri ile ilişkilendirilerek kullanılmaktadır. Örnek olarak bilgisayarımızın sabit diskinde bulunan bir veriye ulaşabilmek için ya doğrudan dosya adı ile ya da dosya adında geçen bazı karakterlere göre ya da dosyanın içerisinde geçen bazı metin parçalarına göre arama yaparak aradığımız dosyaya ulaşırız. Bir kütüphaneye girdiğimizde veya benzeri bir arşivde aradığımız veriye ulaşabilmek için yine metin eşleme ve metin arama tekniklerinden faydalanmaktayız. Yine günlük hayatımızın artık bir parçası haline gelmiş olan internetten daha iyi faydalanmak amacıyla oluşturulmuş olan internet arama motorları yine temelde metin eşleme ve metin arama tekniklerine dayanmaktadır.

Metin eşleme sözlü iletişim veya yazılı hayatımızda farkında olarak veya farkında olmadan çok sıkça kullandığımız bir işlemdir. İnsan sözlü iletişimde duyduğu, gördüğü veya okuduğu metini aklı veya hisleri aracılığıyla kolayca yorumlayabilse ve metinler arasında benzerlikler

kurabilse de bilişim dünyasında metin eşlemenin daha farklı ve hassas bir yere sahip olduğu düşünülebilir. Çünkü çoğu programatik işlemler bu metin eşleme fonksiyonlarından dönen değerlere göre yapıldığından metin eşleme ve akıllı metin eşleme çalışmaları günümüz bilişim dünyasında artan bir öneme sahip olmaktadır.

1.2 Metin Benzerlik Hesaplama Tekniklerinin Kullanım Alanları

Metin Eşleme (ME) ve Metin Benzerlik Hesaplama Teknikleri (MBHT)'nin kullanıldığı çok sayıda yer vardır. Her bir kullanım alanında bu tekniklerden bir veya birkaçı beraberce kullanılarak istenilen fonksiyonalite sağlanır. Biz bu bölümde örnek olması açısından ME ve MBH tekniklerinin kullanıldığı yerlerden en önemli birkaçını açıklayacağız.

1.2.1 İşletim Sistemi Fonksiyonları

Bilgisayarımızda bulunan herhangi bir dosyaya erişmek istediğimizde işletim sistemi üzerinden bu işlemimizi gerçekleştiririz. Aradığımız dosya veya klasöre erişmek için genellikle iki seçenekten birisi kullanılır. Bu seçeneklerden birincisi, eğer aradığımız dosya veya klasörün diskteki mantıksal yerini biliyorsak doğrudan bu yolu takip ederek bu dosya ya da klasöre erişiriz. Eğer aradığımız dosyanın tam olarak yerini bilmiyorsak işletim sisteminin sağladığı dosya arama fonksiyonlarını kullanırız. Bunun için de ilgili dosya veya klasörün mantıksal adında anahtar kelimeler ile arama yaparız veya ilgili dosyada geçmesi muhtemel kelimeler ile arama yaparak bu işlemimizi yaparız. İkinci yol izlendiği takdirde metin eşleme, metin benzerlik hesaplama teknikleri kullanılıyor demektir. Bu nedenle işletim sistemlerinde metin arama, metin eşleme, metin benzerlik hesaplama teknikleri önem arzetmektedir.

1.2.2 İnternet Arama Motorları

Başta günümüzün en popüler internet arama motoru olan Google olmak üzere bütün arama motorları, Metin Eşleme ve Metin Benzerlik Hesaplama Teknikleri üzerine kurulmuştur. Aranılan kelime veya kelime gruplarına en yakın metinleri içeren web sayfaları bularak kullanıcının karşısına getiren bu sistemler çok gelişmiş metin eşleme ve metin arama teknikleri kullanmaktadırlar. Bu tip arama motorlarında klasik metin eşleme metin benzerlik hesaplama teknikleri kullanıldığı gibi akıllı arama teknikleri, hızlı arama teknikleri ve hızlı indeksleme teknikleri kullanılarak bu kadar büyük veride en etkin aramayı yaparak kullanıcının karşısına en tutarlı sayfaları getirmeye çalışmaktadır.

1.2.3 Konuşma Tanıma (KT)

Metin eşleme tekniklerinin kullanıldığı bir başka alan ise Konuşma Tanıma çalışmalarıdır (Nivre, 2000). Ses verisinin çözümlenmesi ile elde edilen metinin anlamlı bir metin olup olmadığına veritabanında tanımlı metinlere benzerliğine göre karar verilir ki, bu da yine metin eşleme anlamına gelmektedir. Konuşma tanıma çalışmaları sadece Metin Benzerlik Teknikleri ile sınırlı olmayıp, Doğal Dil İşleme ve ilgili metinlerin anlamsal ve biçimsel yapılarının analizine de dayanmaktadır.

1.2.4 Optik Karakter Tanıma (OCR)

Herhangi bir resimden veya imajdan OCR programları vasıtasıyla içerdiği metin bilgileri çıkarılabilmektedir. OCR çalışmalarında, tam olarak tanınamayan kelime veya kelime gruplarının tanınması için tanınmaya çalışılan kelimeye en yakın kelimeler araştırılarak, en doğru sonuçların verilmesi amacıyla yine Metin Eşleme Teknikleri ve Metin Benzerlik Hesaplama Tekniklerinden faydalanılır (Liu vd., 2004; Yanıkoğlu ve Kholmatov, 2003).

1.2.5 Doğal Dil İşleme (DDİ)

Metin Eşleme ve Metin Benzerlik Hesaplama Tekniklerinin bir diğer kullanım alanı ise Doğal Dil İşleme çalışmalarıdır. DDİ çalışmaları semantik ve sentaks kuralları da kullanılarak metinlerin yazılımlar, dolayısıyla bilgisayarlar tarafından anlamlandırılması çalışmalarıdır (Külekçi, 2000). İnsan genellikle işittiği hatalı söylenen bir cümle veya ifadenin ne olduğunu kafasında yorumlayarak, her ne kadar işittiği şeyin gerçekte söylenmek istenen ifadeden şekil ve anlam olarak farklı olsa da yine de asıl anlatılmak isteneni anlayabilmektedir. Fakat bilgisayar sistemleri ve bu sistemler için geliştirilmiş olan yazılım ürünleri insandaki bu kabiliyete henüz sahip değillerdir (Amasyalı ve Diri, 2005). Doğal Dil İşleme alanında yapılan çalışmalarda metinlerin hem anlamsal yakınlıklarının bulunabilmesi hem de biçimsel benzerliklerinin hesaplanabilmesi için metin eşleme teknikleri bu konuda da çok önemli olmaktadır (Kardeş vd., 2003).

1.2.6 Veritabanı Uygulamaları

Veritabanı sistemlerinde de ME ve MBHT çok önemlidir. Özellikle ilişkisel veritabanlarında en önemli şey ilgili kayıtların birbirleri ile ilişkilendirilerek kullanılabilmesidir. Yani ilgili veritabanı sisteminde bulunan tablo veya tablolardaki verilerin birbirleri ile ilişkilendirilmesi

bu tablolarda bulunan kolonlardaki veriler üzerinden olmaktadır (Bilenko ve Money, 2003). Bu veriler ise genellikle numerik veriler veya metin tipinden veriler olmaktadır. Numerik verilerde genellikle daha az hatalı yazım ile karşılaşılsa da metin alanlarında çok fazla ve farklı şekillerde hatalar ve hatalı yazımlar ile karşılaşılabilmektedir. Dolayısıyla veritabanı sisteminde aranılan kriterlere uyan kayıtların bulunması için yine Metin Eşleme, Metin Benzerlik Hesaplama tekniklerinden faydalanılır. Özellikle günümüzde bütün kurumsal şirketlerin ve kurumların kullandıkları vazgeçilmez teknojilerden birisinin veritabanı sistemleri oldukları düşünüldüğünde Metin Eşleme ve Metin Benzerlik Hesaplama Tekniklerinin bu alandaki önemi daha iyi anlaşılmış olacaktır.

1.3 Metin Benzerlik Hesaplama Konusu Üzerine Yapılmış Çalışmalar

ME ve MBHT çalışmalarında, iki farklı metine bakan bir insanın kolaylıkla bu metinlerin birbirlerine ne kadar benzeştiğini anladığı gibi, verilen metinlerin birbirlerine ne kadar benzediklerini yazılımlar ve bilgisayarlar tarafından buldurmaya yönelik birçok çalışma yapılmıştır ve yöntemler geliştirilmiştir. Özellikle bu tip çalışmalar yurtdışında ve günümüzde en yaygın teknoloji dili olan İngilizce üzerinde yapılmış olsa da son yıllarda ülkemizde de bu konu üzerinde Türkçe üzerinde yapılan çalışmaların sayıları artmaktadır.

1.3.1 En Çok Kullanılan Metin Benzerlik Hesaplama Algoritmaları

Bu bölümde Metin Eşleme/Metin Benzerlik Hesaplamak üzere geliştirilmiş yöntemlerden en önemlileri hakkında kısaca bilgi verilecektir. Genel olarak birbirleri ile tam olarak aynı olmayan metinlerin benzerliklerini hesaplayabilmek için Bulanık Metin Eşleme/Arama yöntemleri geliştirilmiştir.

Günlük hayatımızda, sözlü iletişimlerimizde yine farkında olmadan ve adını koymadan sıkça Bulanık Metin Eşleme yöntemlerini kullanırız. Mesela Türkiye'nin farklı illerinden farklı kişiler aynı kelimeyi farklı şekillerde telaffuz etseler de ve hatta telaffuzları sırasında harf veya hece yanlışları da yapsalar biz bu insanların demeye çalıştıkları şeyi doğru anlarız. İnsan aklı, zekası, önceki deneyimleri ve bilgi birikimleri sayesinde bu hatalı telaffuzlardan bile doğru metini kolayca anlayabilir.

Ama yazılımların yetenekleri ve kısıtları ile sınırlı bilişim dünyasında, metin eşleme işlemleri daha zor olmaktadır. Çünkü genellikle metin eşleme demek programın yazıldığı C, Pascal, Java gibi herhangi bir programlama dilinde metin karşılaştırma fonksiyonları veya "= ="

operatörleri ile yapılmaktadır ki, bu fonksiyon ve operatörlere gönderilen iki metine birbirlerinin aynı diyebilmesi için tamamen aynı olmaları gerekmektedir. Hatta bu metinlerin birisi "İSTANBUL" olsa diğeri ise "İstanbul" olsa bile yine bu iki metin aynı metin olarak değerlendirilmezler. Öncelikle her iki metin de büyük harfe veya her iki metin de küçük harfe çevrildikten sonra bu fonksiyon veya operatör tarafından kullanılırsa aynı metinler diye sonuç döndürürler.

Bunun için bilişim dünyasında sıkça kullanılan *Bulanık*, *Akıllı* gibi kavramlardan hareketle Bulanık Metin Eşleme veya Akıllı Metin Eşleme teknikleri üzerinde çalışılmaya başlanmıştır. Bu çalışmalarda temel amaç insan zekasının kolaylıkla birbirine benzer veya birbirinin aynısıdır dediği ama gerçekte en az bir karakterinin farklı olduğu iki metinin yazılım tarafından da benzer veya aynı metinler olarak algılanmasını sağlamaktır. Bunu söyleyebilmek için ise Yapay Zeka kavramlarından faydalanılmaktadır.

Metin benzerlik hesaplama yöntemleri genel olarak üç kısıma ayrılabilir (Cohen vd., 2003). Bunlar;

- Edit Distance Benzerliğine Dayanan Yöntemler
- Token Tabanlı Benzerlik Yöntemleri
- Hibrit Yöntemler

Edit Distance benzerliğine dayanan yöntemler metinlerde bulunan aynı veya yakın lokasyonlarda bulunan karakterlerin farklılaşması, değişmesi, düşmesi veya fazlalaşmasına dayanır. Örnek olarak; Levenshtein Benzerliği, Monge-Elkan Benzerliği, Smith-Waterman Benzerliği, Jaro-Winkler Benzerliği verilebilir.

Token tabanlı yöntemlerde ise benzerlikleri bulunacak metinler kelime veya kelime gruplarına ayrılır ve buna göre toplam benzerlik hesaplanır. Örnek olarak; Jaccard Benzerliği, Cosine Benzerliği, Jensen-Shannon Uzaklığı, FS (Fellegi ve Sunter) uzaklığı verilebilir.

Hibrit yöntemlerde ise önceki iki tip benzerlik yöntemleri beraberce kullanılır. Örnek olarak; softCosine Benzerliği verilebilir.

Bundan sonraki bölümde en çok bilinen Metin Eşleme/Benzerlik Hesaplama yöntemleri kısaca tanıtılacaktır.

1.3.1.1 Tam Metin Eşleme

Tam metin eşleme, adından da anlaşılacağı gibi iki metinin birbirinin tamamıyla aynı olması esasına dayanan metin eşleme ve benzerlik hesaplama yöntemidir. Burada verilen iki metine aynı metinler diyebilmek için, iki metindeki harflerin büyüklük ve küçüklükleri ve metinlerin baş veya sonlarındaki boşluklar da dahil olmak üzere tamamen aynı olmalıdır.

1.3.1.2 Hamming Uzaklığı

Hamming Uzaklığı aynı uzunluktaki iki metinin kaç pozisyonda ilgili karakterin farklılaştığının ölçüsüdür. Mesela "Kaburga" ve "Kadırga" metinleri arasındaki Hamming Uzaklığı 2 dir. Çünkü iki metinin 3 ve 4. sırada bulunan karakterler birbirinden farklıdır. Bu yöntem doğal dil işleme gibi çalışmalar için çok kullanışlı olmamaktadır. Çünkü benzerlikleri bulunacak metinlerin aynı uzunlukta olmaları kısıtını getirmektedir. Oysa günlük hayatta kullanılan metinler ve benzerliği araştırılacak metinlerin aynı uzunlukta olmaları şartını aramak çok pratik ve gerçekçi olmamaktadır.

1.3.1.3 Levenshtein Edit Distance

Edit Distance kavramı genel olarak iki metinin birbirine kaç birim hamle sonrasında eşit olacaklarının bir ölçüsüdür[2]. Bu birim hamleler *ekleme/değiştirme/silme* işlemlerinden herhangi birisi olabilir. Yani iki metini birbirlerine eşitlemek için metinlerin herhangi birisinin bir karakter lokasyonuna yeni bir harf eklenmesi, bu lokasyondaki harfin başka bir harf ile değiştirilmesi veya bu harfin bu metinden çıkarılması işlemlerinden kaç tanesinin peş peşe yapılması gerektiğinin bir ölçüsüdür.

Örnek olarak "katip" ve kitap kelimelerini birbirlerine eşit metinler haline getirmek için gerekli işlem sayısı 2 dir. Yani bu iki metinin Edit Distance'ı 2 dir. Çünkü;

katip != kitap (iki metin eşit değil)

kitip != kitap (ilk metinde bulunan a harfi i harfi ile değiştiriliyor, işlem sayısı 1)

kitap = kitap (ilk metinde bulunan ikinci i harfi a harfi ile değiştiriliyor, işlem sayısı 1)

Yani toplamda 2 adet *değiştirme* işlemi sonucunda iki metin birbirinin aynı oldu.

Levenshtein Edit Distance ise, Levenshtein adındaki bir bilim adamının verilen iki metinin edit distance değerini hesaplamak için geliştirdiği Dinamik Programlama kullandığı bir

algoritmadır. Normalde metinlerin uzunlukları arttıkça klasik yöntemlerle Edit Distance hesaplama, işlem yükü olarak çok fazla olacağından Dinamik Programlama Tekniklerinden faydalanılmaktadır. Dinamik Programlama, karmaşıklığı ve işlem yükü fazla olan hesaplamaları çok küçük karmaşıklıklarda problemlere indirgeyen bir programlama tekniğidir.

1.3.1.4 Needleman-Wunch Distance

Bu yöntem Temel Edit Distance yöntemine benzemektedir. Temel Edit Distance yöntemi olan Levenshtein Edit Distance (LED)'den farklı olarak, edit distance yönteminde *ekleme/değiştirme/silme* gibi işlemlerdeki maliyet hesaplamasına ek olarak bu yöntemde boşluklara da bir maliyet ayarı yapılabilmektedir. LED'de boşluk karakteri özel olarak ele alınmamakta ve herhangi bir karakter gibi değerlendirilmektedir. Bu yöntemde ise *ekleme/değiştirme/silme* işlemi yapılan karakter boşluk karakteri ise farklı bir maliyet verilmekte, boşluk karakterinin dışında bir karakter ise daha farklı bir maliyet verilebilmektedir.

1.3.1.5 Gotoh-Distance

Gotoh-Distance yöntemi Levenshtein Edit Distance yöntemini baz alarak geliştirilmiş bir yöntemdir. Genellikle biyologların ihtiyacı olan protein zincirlerindeki DNA parçalarının benzerliklerini araştırma amacıyla geliştirilmiştir. Özellikle boşluklu metinler üzerinde iyi sonuçlar veren bir yöntemdir. Levenshtein Edit Distance'tan farklı taraflarından biri, Boşluk karakterlerine ekleme ve silme işlemi için ayrı maliyetlerin verilebilmesidir.

1.3.1.6 Monge-Elkan Distance

Monge Elkan yöntemi de aslında Gotoh Distance yönteminin geliştirilmiş bir versiyonudur. Daha önce bahsedilen Gotoh yönteminin yeteneklerine ek olarak metinlerin veya metinlerin içerdiği alt metinlerin bazı semantik benzerliklerini de göz önünde bulundurmaktadır.

1.3.1.7 Block Distance

Bu yöntem L1 Distance veya City Block Distance olarak da adlandırılmaktadır. Vektör tabanlı bir yöntem olup, metinler iki boyutlu vektörel alanlarda tanımlanmakta ve metinlerin benzerlikleri, oluşturulan bu iki boyutlu yapı üzerinden hesaplanmaktadır.

1.3.1.8 Soundex Distance

Soundex, fonetik hatalarını ve telaffuz hatalarını ele alan bir yöntemdir. Özellikle İngilizce gibi okunuşu genellikle yazılışından farklı olan diller için faydalı bir yöntemdir. Eşlenecek metinlerin ses benzerliklerine dayanır. Mesela *John*, *Johne* and *Jon* isimleri aynı şekilde telaffuz edilir. Genellikle benzerlikleri bulunacak metinlerde bulunan sessiz harflerin benzeşmesine göre çalışır. Türkçe için genellikle bu tip hatalar ile pek karşılaşılmamaktadır. Çünkü Türkçe okunduğu gibi yazılan ve yazıldığı gibi okunan bir dildir. Türkçe metinler üzerinde bu türden hatalara benzer olarak genellikle boşluklardan kaynaklanan ve boşluğu telaffuzda hızlı geçerek iki kelimeden oluşan bir metinin tek kelime gibi tellafuz edilmesinden kaynaklanan durumlar oluşur.

1.3.1.9 Euclidian Benzerliği

Bu yöntem de vektör uzayında çalışan bir yöntemdir. Vektör girdilerindeki Direkt Euclidian Uzaklığına göre metinlerin benzerliklerini bulur. Benzerlik hesaplaması yapılırken, Cosine Similarity yönteminden farklı olarak vektörler arasındaki açının cosinüsü kullanılarak hesaplanmaz. Benzerlikleri hesaplanacak vektörlerin Direkt Euclid uzaklığına göre hesaplanır.

1.3.1.10 Cosine Benzerliği

En genel vektör tabanlı benzerlik yöntemlerindendir. Benzerlikleri bulunacak metinler vektör uzaylarına dönüştürüldükten sonra vektörlerin arasındaki Euclidian Cosine kuralı uygulanarak benzerliklerini hesaplar. Genellikle diğer token tabanlı yöntemler ile beraber kullanılarak boyut probleminde bir sınırlandırma sağlanır. Yöntem genellikle terim sayısı kadar boyut üzerine çalışır.

1.3.1.11 Jaccard Benzerliği

Bu yöntem de Cosine Benzerliği gibi token tabanlı vektör uzayları üzerinde çalışan bir yöntemdir. Karşılaştırılacak kelime çiftlerinin benzerliklerini hesaplayarak toplam benzerlik hesaplanır. Genellikle kimyasal bileşimlerin benzerliklerini hesaplamak için kullanılmaktadır.

1.3.1.12 Variational Benzerlik

Metinler üzerindeki farklılıkların benzerliklerini hesaplamak için olasılık dağılımlarından faydalanan bir yöntemdir. Diğer yöntemlere göre daha matematiksel bir yöntemdir.

1.3.1.13 N-gram Uzaklığı

N-Gram benzerliği veya n değerini örnekleyecek olursak 3-gram (Trigram) benzerliği iki metinde de bulunan 3'lü bitişik karakter katarlarının sayısına dayanan bir benzerlik hesaplama yöntemidir. İki metinin de başı ve sonuna birer boşluk karakteri eklenir ve her iki metin de 3'lü karakter katarlarına parçalanır. Sonuçta iki metinin benzerliği şöyle hesaplanır;

$$s = \frac{2*m}{a+b} \text{ dir.} \qquad (1.1)$$

Burada,

m: iki metinde de birbirini tutan 3'lü karakter katar sayısı,

a: birinci metindeki 3'lü karakter katar sayısı

b: ikinci metindeki 3'lü karakter katar sayısıdır.

Mesela "işlemsel" ve "işlevsel" metinlerinin Trigram benzerlikleri;

$$s = \frac{2*7}{8+8} = \frac{14}{16} = 0{,}875$$

olarak hesaplanır.

1.3.1.14 Jaro-Winkler Benzerliği

Verilen iki s1, s2 metinleri için bu iki metinin Jaro-Winkler benzerliği:

$$s = \frac{m}{3*a} + \frac{m}{3*b} + \frac{m-t}{3*m} \qquad (1.2)$$

olarak hesaplanır.

Burada ;

m: eşleşen karakter sayısı

a: s1'in uzunluğu

b: s2'nin uzunluğu

t: transpozisyon sayısıdır.

İki karakter birbirinden

$$\frac{\max(a,b)}{2}-1 \tag{1.3}$$

den daha uzakta değilse bu iki metin eşleşiyor olarak kabul edilir.

s1 deki ilk eşleşen karakter s2 deki ilk eşleşen karakter ile karşılaştırılır, s1 deki ikinci eşleşen karakter s2 deki eşleşen karakter ile karşılaştırılır vb. Eşleşmeyen karakter sayısı 2 ye bölünerek transpozisyon sayısı bulunur.

Geliştirilmiş Jaro-Winkler metodu 1/3 ten daha farklı ağırlıklar kullanır. OCR hataları veya klavye yazım yanlışları gibi bazı hata tipleri daha düşük puanlandırılarak daha iyi hesaplamalar yapılır.

Örnek:

MÜKEMMEL ve MUKEMEL metinlerinin benzerliği

$$s=(\frac{6}{8}+\frac{6}{7}+\frac{6-1}{6})*\frac{1}{3}=0{,}81$$

olarak hesaplanır. (Eşleşen karakterler: M, K, E, M, E, L; 1 transpozisyon).

1.4 Metin Benzerlik Hesaplama Yöntemlerinin Türkçe Metinler Üzerindeki Başarısı

Metin eşleme ve benzerlik hesaplama çalışmaları üzerine yapılan çalışmalar genellikle İngilizce dili üzerine yapıldığından İngilizce metinler için başarılı sonuçlar vermektedir. Çünkü, metin eşlemede muhtemel durumlar oluşturulurken İngilizce kelimeler ve İngilizce metinlerde sık karşılaşılan durumlar düşünülerek modellemeler yapılmakta ve çözümler aranmaktadır. Fakat bilindiği gibi her dilin kendine göre hususiyetleri ve muhtemel hata durumları bulunmaktadır. Bu nedenle İngilizce için başarısı çok yüksek olan bazı yöntemler, Türkçe metinlerde düşük başarı gösterebilmektedir. Bu nedenle belirtilen yöntemlerin Türkçe metinler üzerinde daha iyi sonuçlar vermesini sağlamak amacıyla ya bu yöntemlerin

lokalizasyon çalışmaları yapılmalı ya da Türkçe metinler için mevcut yöntemlerin yeteneklerini de içeren yeni yöntemler geliştirilmelidir. Bu çalışmada var olan yöntemlerden bağımsız olarak Türkçe hata durumları modellenmiş ve yeni bir yöntem geliştirilmeye çalışılmıştır.

1.5 Veritabanı Teknolojilerinde MBH Tekniklerinin Kullanılması

Veritabanı sistemleri özellikle İlişkisel Veritabanı Sistemleri adından da anlaşılacağı gibi birbiriyle ilişkili tablo ve dolayısıyla birbiri ile ilişkilendirilmiş olan verilerden oluşur. Yani veritabanı sistemlerinde ilişkiler tabloların alanlarındaki değerler üzerinden olur. Numerik alanlarda bu ilişkilerin araştırılması daha kolay olurken farklı türlerden metin türü veriler içeren alanlar üzerinde ilişkilendirmede genel metin eşleme çalışmalarında karşılaşılan problemler ile karşı karşıya kalınılır. Çünkü bir şekilde herhangi bir nedenden dolayı aynı verinin farklı bir formatta veya hatalı bir şekilde yazılmış olmasından dolayı birbiri ile gerçekte ilişkili olan tablo verileri ilişkilendirilememektedir.

Günümüzde şirketlerin, bankaların, kurum ya da kuruluşların veritabanlarındaki kayıtlar sürekli büyüdüğünden, sürekli olarak büyüyen tablolardaki gereksiz tekrarlardan arındırılması önemli olmaktadır. Mesela birleşen iki bankanın müşteri kayıtlarının birleştirilmesi esnasında tekrarların önüne geçilmesi gerekmektedir. Bir müşteri sadece bir kere tanımlı olmalıdır. Veya bir ürün için yapılan anketlerin değerlendirilmesi esnasında ankete katılan kişinin daha önceden kendi müşteri veritabanında kayıtlı olup olmadığını bilmek yine o firma için önemli olmaktadır. Veri tabanlarında tekrardan arındırılmasının önemli olacağı bir diğer yer de sayısal bilgileri toplayıp, değerlendirip şirketlere ticari veri olarak sunan veri madenciliği yapan şirketlerden veri satın alan firmalardır. Bu firmalar satın aldıkları verilerin sadece kendi veritabanlarında olmayanları için para ödedikleri düşünüldüğünde olayın önemi daha iyi anlaşılacaktır.

Bu sebeplerden ötürü veritabanı sistemlerinde metin eşleme ve metin benzerlik hesaplama yöntemleri çok büyük önem taşımaktadır. Ve bu konuda yapılan her türlü çalışma ve geliştirilen yöntem veritabanı sistemlerinde pratik olarak kullanım alanı bulmaktadır.

Veritabanı sistemlerinde istenmeyen tekrarın başlıca iki nedeni vardır, birincisi farklı veri kaynaklarından gelen verilerin birleştirilmesi, ikincisi ise grafik arayüzünden girilen benzer kayıtlar için kullanılan sistemin yeni bir ID üretmesidir.

Oracle ve Microsoft SQL Server gibi veritabanı sistemleri tabloların veya kayıtların

ilişkilendirilmesi için gerekli olan metin eşleme ve metin benzerlik hesaplama yöntemlerini çok gelişmiş şekilde kullanmaktadır. Bu nedenle her iki sistemde builtin olarak Fuzzy String Search ve Fuzzy String Similarity yöntemlerini içermektedir. Ve yine önceki bölümde bahsedilen Metin Benzerlik Yöntemlerini gerçekleyen fonksiyon ve prosedürleri de kullanmakta ve kullanıcının kullanımına da sunmaktadır. Örnek olarak Oracle 10g veritabanı sisteminde UTL_MATCH.JARO_WINKLER_SIMILARITY fonksiyonu iki metinin Jaro-Winkler yöntemine göre benzerliğini hesaplar (Austin, 1998).

2. TÜRKÇE METİNLER İÇİN YAPILMIŞ ÇALIŞMALAR

Türkçe yazıldığı gibi okunan bir dil olması ve telaffuzdan kaynaklanan hatalar ile daha az karşılaşılması nedeniyle Doğal Dil İşleme ve Metin Benzerlik Hesaplama çalışmaları için son derece uygun bir dildir. Bu nedenle İngilizce veya bir başka dil için sık karşılaşılan bazı zorluklar veya hata durumları Türkçe için daha az söz konusu olabilir. Buna karşın Türkçe'nin de kendine has durumları ve problemleri olması doğaldır (Cebiroğlu, 2006).

Her ne kadar Türkçe yazıldığı gibi okunan bir dil olsa da metinler söylendiği gibi yazılmamış olabilir. Yani eşleme yaparken, metinin birisi tarafından başka birisine söylenmiş ve söylenirken veya kaydedilirken yanlış girilmeler de olabileceği gözönünde bulundurularak benzerlik hesaplama işleminin kalitesi artırılmalıdır. Türkçe Bulanık Metin Eşleme çalışmaları üzerinde çok fazla çalışma yapılmamış bir konudur. Aslında bu konularda akademik veya ticari birçok çalışmalar yapılmış olsa da bu çalışmaların yeterli olmadıkları aşikardır.

Türkçe Metinler üzerine yapılmış çalışmalardan en çok bilinen çalışmalardan bazıları örnek olarak tanıtılacaktır. Daha önce bahsedildiği gibi burda bahsedilenlerin dışında bu konuları da kapsayan birçok akademik veya ticari çalışmalar da yapılmaktadır. Mesela Türkiye'de bazı bankalar veritabanlarındaki tekrarlı kayıtları bulmak veya anketler yoluyla gelen kayıtları mevcut veritabanı kayıtları ile eşleştirmek için Türkçe hata durumlarını da içeren çalışmalar yapmaktadır.

2.1 Zemberek

Zemberek, Tspell projesinin geliştirilmiş halidir[5]. Türkçe diline ilişkin çeşitli bilgi islem problemlerinin çözümlenmesi icin açık kaynak kodlu, platform bağımsız bir Doğal Dil İşleme kütüphanesi oluşturulma amacı ile başlatılmıştır. Türkçe kelime denetleme, kelime çözümleme, kelime önerme, oluşturma, Türkçe karakter kullanılmadan yazılan yazıların dönüştürülmesi, heceleme, kodlama hatası yüzünden bozulmuş metinlerin düzeltilmesi gibi işlemleri gerçekleştirmektedir. Yaklaşık 20.000 kök ve 6000 tane özel isim içeren bir sözlük kullanmaktadır.

2.2 Pardus

Pardus, TÜBİTAK-UEKAE (Ulusal Elektronik ve Kriptoloji Araştırma Enstitüsü) bünyesinde

yürütülen bir işletim sistemi geliştirme projesidir. Pardus açık kaynak kodlu ve GPL (GNU Genel Kamu Lisansı) ile dağıtılan özgür bir yazılımdır [3]. Linux'un Türkçe olarak derlenmiş özel bir versiyonudur. Zemberek gibi Pardus da bazı Türkçe bilgi işlem problemlerine çözüm sunmaktadır. Türkçe dili kullanan yazı yazma uygulamaları üzerinde (OfisaMessenger, e-mail, vs.) Türkçe imla ve yazım denetimi yapılabilmektedir. Bu denetim yapılırken Zemberek çalışmasından faydalanılmıştır. Dosya ve dizin adlarında Türkçe karakter kullanılması durumunda doğan bazı problemler çözüme kavuşturulmuştur. Çekinmeden Türkçe alfabenin tüm harflerinin kullanılabildiği bir işletim sistemidir. Kullanıcıyı güzel bir Türkçe kullanmaya teşvik eder.

2.3 Türkçe WordNet

Türkçe kavramsal bir veritabanı oluşturma çalışmasıdır. Mesela "kira sözleşmesi" diye bir metin arandığında "kira kontratı" şeklinde olan metinleri de getirir. Çok dilli kaynaklarda tek bir dilde arama yapabilmeyi hedeflemiştir. BalkaNet pojesi için Türkçe bir veritabanı hazırlanarak oluşturulmuştur (Durgar vd., 2004; Haynes, 2002).

2.4 Diğer Çalışmalar

Yukarıda bahsedilenlerin dışında Türkçe metinler üzerine çok sayıda akademik çalışma yapılmıştır. Bu çalışmaların birçoğu Yapay Zeka, Doğal Dil İşleme, kelimelerin kökenlerine inebilme, Türkçe kelimeler için kurallar tanımlayabilme gibi konuları kapsamaktadır. Türkçe metin eşleme, metin benzerlik hesaplama konuları üzerine özel olarak yapılmış bir çalışma ve geliştirilen yeni bir yönteme rastlanılmamıştır.

3. YAPILAN ÇALIŞMA

Önceki bölümlerde Metin Eşleme ve Metin Benzerlik Hesaplama üzerine yapılmış çalışmalardan bahsedilmiştir. Bu bölümde ise yapılan çalışma ve Türkçe Metinler gözetilerek geliştirilen yöntemden bahsedilecektir.

3.1 Türkçe Metinlerde Sıkça Karşılaşılan Hata Durumlarının Tesbit Edilmesi

Genel metin eşleme ve metin benzerlik hesaplanması teknikleri araştırıldıktan sonra Türkçe metinlerde sık karşılaşılan hata durumları ve metinlerin benzerliğinin hesaplanması veya metin eşleme yapılırken karşılaşılan problemler ve durumlar incelendi. Bunun için değişik seviyede çok sayıda bilgisayar kullanıcısına en çok karşılaştığı durumlar soruldu ve Türkçe metin eşleme ve benzerlik hesaplanması çalışmalarında en çok karşılaşılan 11 tane durum belirlendi. Bu durumların bazıları sadece Türkçe metinler için geçerli olmayıp genel sorunlardır. Fakat çalışmanın daha sağlıklı olması açısından geliştirilen yöntemde bu durumlar da göz önünde bulundurulduu.

Sonraki bölümlerde daha detaylı olarak ele alınacak bu durumların herbirisi ile karşılaşılma olasılığı farklıdır ve bu olasılıklar genellikle kullanıcıdan kullanıcıya göre farklılık göstermektedir. Mesela genellikle İngilizce klavye ile yazı yazmaya alışkın bir kişinin karşılaşacağı durumlar Türkçe klavye ile yazı yazmaya alışkın olan kişiye göre farklılık gösterebilecektir.

Ayrıca bu 11 durum ile karşılaşılma olasılığı bilgisayar sistemine, veri giriş şekline, kaynağına göre farklılık gösterebilir. Mesela günlük hayatta sık yaşandığı üzere bir kişinin sesli olarak söylediği bir veriyi diğer bir kişinin bilgisayar sistemine girerken, giren kişinin yanlış işitmesi veya işittiği şeyi yazarken yanlış girmesi nedeniyle bu hata durumlarından bazıları oluşabilir.

Yine bu hata durumlarının oluşma sebeplerinden birisi de üzerinde çalışılan verinin kaynağıdır. Mesela Dil ve Bölgesel Ayarları farklı olan bir bilgisayardan diğer bir bilgisayara veri taşındığında veride farklılıklar ve bozulmalar olabilir. Bu durum da yine belirtilen 11 tane durumdan bir veya birkaçının oluşma nedeni olabilir.

Ayrıca OCR veya KT gibi işlemler neticesinde bu işlemi yapan yazılım ürününün doğruluk veya hassasiyetine göre bu 11 durumdan bazıları ile karşılaşılması mümkündür.

Sonuç olarak yapılan çalışmada her ne kadar bu 11 tane durum beraber ele alınsa da bu

durumlar ile gerçekte karşılaşma olasılığının tam olarak tesbit edilmesi zordur. Bu nedenle bu hata durumlarından bir, iki veya üç tanesinin beraberce olabileceği ihtimalini gözönünde bulunduran bir yapı ortaya konulmaya çalışıldı. Daha sonra yapılabilecek bir başka çalışmada da bu durumların olasılıkları araştırılıp, puanlanabilir ve kendi aralarında bir öncelik oluşturularak daha parametrik ve matematiksel bir temele oturtulabilir.

Aşağıda Türkçe metinler içeren metinlerde sık karşılaşılan 11 hata durumundan bahsedilecektir.

3.1.1 Büyük Küçük Harf Farklılıkları

Herhangi iki metin birbirinin aynı olsa bile bu metinlerde bulunan karakterlerden tamamı veya bir kısmı büyük veya küçük harfle yazılmış olabilir. Böylece birbirinden farklı olarak yazılmış bu iki metin, programlama dillerindeki "= =" operatörü tarafından aynı metinler olarak değerlendirilmezler.

Bu büyük küçük harf farklılaşması bazen metinlerin farklı metinler olarak algılanmasına neden olurken, bazen de bu durum göz ardı edilerek bu metinler aynı metinlermiş gibi davranılır. İçerisinde harflerinin büyüklük küçüklükleri dışında hiç bir farklılığın olmadığı iki metinin aynı metin mi yoksa farklı metin mi olduklarının kararı bu metinlerin kullanım yerine ve amacına bağlı olarak değişebilir. Zaten çoğu yazılım ürününde de Büyük Küçük Harfe Duyarlılık bir seçenek olarak kullanıcıya sunulmaktadır.

Büyük/küçük harfe duyarlı olmayı gerektirmeyen durumlarda iki metinin aynı metin olup olmadıklarının veya benzerlikleri hesaplanmadan önce bu iki metin ya beraberce büyük harfe çevrilmeli ya da beraberce küçük harfe çevrilmelidir.

Bir metini büyük harfe veya küçük harfe çevirirken de çeviren fonksiyonun dil desteği olması gerekmektedir. Genellikle bu çevirme işlemini yapan fonksiyonlar İngilizce dili düşünülerek yapıldığından Türkçe için yanlış sonuçlara neden olmaktadır. Mesela "I" karakteri İngilizce için yazılmış bir küçük harfe çevirme fonksiyonundan geçirildiğinde "i" olmaktadır, oysaki Türkçe bir metin için bu dönüşüm "ı" olmalıydı.

Büyük küçük harf farklılıklarına örnek verecek olursak; eşlenecek metinler, "istanbul" ve "İSTANBUL" olsun. Bu iki metin aynı metinler olmasına rağmen sadece harflerin büyüklük ve küçüklükleri farklıdır. Eğer biz bu iki metini programlama dillerinde bulunan "= = " (eşit mi) operatörüne verirsek veya metin karşılaştırma fonksiyonlarına parametre olarak

gönderirsek sonuç negatif olacaktır, yani metinler farklı olarak algılanacaktır. Yine bu iki metini Türkçe desteği olmayan İngilizce metinler için yazılmış bir Büyük_Harfe_Çevir fonksiyonundan geçirdikten sonra yine tekrar "= =" operatörüne veya string compare fonksiyonlarına gönderirsek sonuç yine negatif olacaktı. Çünkü "istanbul" kelimesi Türkçe desteği olmayan Büyük_Harfe_Çevir fonksiyonundan "ISTANBUL" olarak geri döneceğinden ve ISTANBUL != İSTANBUL olduğundan bu iki metin birbiri ile eşlenememiş olacaktır.

3.1.2 Metin Başı ve Sonundaki Boşluklar

Bu durum genellikle verinin tutulduğu sistemin yapısından dolayı oluşabileceği gibi yine elle girilen verilerde metinin başı veya sonunda fazladan bir veya daha fazla boşluk karakterinin girilmesi nedeniyle de oluşabilir. Genellikle veritabanlarında bulunan veriler bir başka veri kaynağından alındığı için bazı format farklılıkları olabilmektedir.

Mesela Oracle veritabanı için bu durumu örnekleyecek olursak, Oracle veritabanı sisteminde CHAR ve VARCHAR2 olmak üzere birbirine benzer iki tane veri tipi bulunmaktadır (Austin, 1998). Farklı iki tabloda bulunan iki tane "Şehir" alanı olduğunu varsayalım ve bu Şehir alanı birinci tabloda CHAR(15) olacak şekilde ve ikinci tabloda da VARCHAR2(15) olacak şekilde tanımlanmış olsun. Tanımlardan da anlaşılacağı üzere her iki tablodaki alanın maksimum uzunluğu 15 olarak verilmiştir. Ve her iki tabloda bulunan birer kayıta bu alana değer olarak "İSTANBUL" verisi girilmiş olsun. Daha sonra bu iki veri "= = " operatörüne verilirse sonuç negatif olacaktır. Çünkü CHAR alanının özelliği, eğer alanın maksimum uzunluğundan daha küçük uzunlukta bir veri girilirse girilen verinin sonunda boşluklar olacak şekilde veri saklanır. Yani, 15 uzunluklu bir CHAR alana girilen "İSTANBUL" verisi veritabanında "İSTANBUL" olacak şekilde sonunda 7 tane boşluk karakteri olacak şekilde saklanır. VARCHAR2 tipinden alanlar ise maksimum uzunluğu ne olursa olsun içerdiği veri, verinin uzunluğu kadar bir metin olarak tutulur, yani sonuna boşluk karakteri eklenmez. Dolayısıyla belirtilen iki metin farklı metinler olarak karşımıza çıkar.

Bu nedenle eşleme fonksiyonuna sokulacak metinlerin benzerliğine bakılmadan önce bu metinler, verilen metinin başı ve sonundaki boşlukları silen Kırpma(Trim) fonksiyonundan geçirilmelidir.

3.1.3 Türkçe Karakterlerin Kullanılmasından Kaynaklanan Durumlar

Metin eşleme çalışmalarında sorun olarak karşımıza çıkan bir başka durum da Türkçe

alfabede bulunan "ı", "ü", "ş", "ğ", "ö", "ç" harflerinin yerine bunların benzerleri olan "i", "u", "s", "g", "o", "c" harflerinin kullanılması veya tersi durumdan kaynaklanan sorunlardır. Mesela "İSTANBUL" kelimesi bir başka veri kaynağında "ISTANBUL" olarak yazılmış olabilir. Aslında bu iki metin de aynı olmasına rağmen yine metin eşleme fonksiyonlarınca veya "= =" operatörünce farklı metinler olarak değerlendirilir.

Bu durumun bir sebebi genellikle verilerin bulundukları bilgisayar sistemlerinin dil ayarlarının farklı olması veya bazı yazılım ürünlerinin Türkçe dil desteklerinin olmamasıdır. Bir diğer neden ise dijital dünyada çok sık karşılan "Karakter Uyumsuzluğu" veya "Türkçe Karakter Problemi" gibi sorunlar ile karşılaşmamak için çoğu kullanıcının bilinçli olarak "ı", "ü", "ş", "ğ", "ö", "ç" harfleri yerine sırasıyla "i", "u", "s", "g", "o", "c" harflerini tercih etmeleridir. Mesela çoğu bilgisayar kullanıcısı mail ortamında karşıdaki mail alıcısının Internet Explorer ayarlarının doğru olduğundan emin olamadığından dolayı, mail metninin bozuk çıkma ihtimaline karşı bilinçli olarak Türkçe karakter kullanmamaktadırlar. Ve yine birçok yazılım ürünü dosya veya dizin adında Türkçe karakter ile karşılaşırsa hata durumlarına düşmekte veya belirtilen dosyanın bulunamadığı hata bilgisini geri döndürmektedir. Yine bu yüzden birçok kullanıcı dosya, klasör veya herhangi bir şeye isim verirken Türkçe alfabeye özgü "ı", "ü", "ş", "ğ", "ö", "ç" harflerini bilinçli olarak kullanmamaktadırlar. Bütün bu durumlar da metin eşleme çalışmalarında bir problem olarak karşımıza çıkmaktadır.

Bu tip durumların oluşmasının bir başka nedeni ise verilerin bazen anketörlerin yaptığı gibi önce kağıt üzerine yazılıp, sonra veri girişinden sorumlu kişiler tarafından veritabanına elle geçirilmesi esnasında yanlış okumadan dolayı bu karakterlere en yakın karakterlerin tercih edilmesidir.

3.1.4 D-T, B-P Harf Çiftlerinin Birbirlerinin Yerine Kullanılmasından Kaynaklanan Durumlar

Türkçe metinlerin eşlenmelerinde karşılaşılan bir diğer sorun da "d" harfinin yerine "t" harfinin ve "b" harfinin yerine "p" harfinin kullanılmasından veya tam tersi durumlardan kaynaklanmaktadır.

Türkçe kelimelerin telaffuzunda bu harfler birbirlerine benzediğinden ve Türkçe dilbilgisi kuralları arasında bulunan "Sert Sessizlerin Yumuşaması" gibi nedenlerden dolayı metinlerde bu harfler birbirlerinin yerine sıkça kullanılabilmektedir (Cebiroğlu, 2006).

Örnek olarak "kasap" kelimesi yanlışlıkla "kasab" kelimesi ile ifade edilebilir. Normalde bu iki metin *b,p* harfleri dışında aynı olmasına rağmen yine bu iki metin "= =" operatörünce farklı metinler olarak değerlendirilir.

3.1.5 Aynı Harfin Peşpeşe İkilemesinden Kaynaklanan Durumlar

Türkçe metin eşleme çalışmalarında (diğer dillerde de bu böyledir) karşılaşılan bir başka durum da peşpeşe gelen iki karakterin bir tanesinin eşlenecek metinlerin birinde unutulması veya yazılmaması durumudur. Örnek olarak "müddet" kelimesi yazılırken "müdet" olarak yazılmış olabilir. Gerçekte bu iki metinin Edit Distance'larına bakarsak 1 (bir) olduğunu görürüz. Ama insan gözü ile bu iki metine baktığımızda bu iki metinin aynı metinler olduğu, sadece ikinci metinin yazılırken bir şekilde ikinci "d" harfinin yazılmadığını kolaylıkla söyleyebilmekteyiz. Bu nedenle metin eşleme çalışmalarında bu durumun da olabileceğini göz önünde bulunduran yöntemler kullanılmalıdır.

3.1.6 Metinlerin İkinci Harflerinin "i" , "ı" veya "u" Olmasından Kaynaklanan Durumlar

Metin eşleme çalışmalarında karşılaşılan bir başka durum da ilk harfi sessiz olup, normalde ikinci harfi *i-ı-u* gibi seslileri bulundurmadığı halde bu kelimelerin telaffuz edilirken olduğu gibi sanki bu seslilerin varmış gibi yazıya geçirildiği durumlardır.

Örnek olarak "spor-sipor", "prenses-pirenses" , "grup-gurup" ve "klavuz-kılavuz" gibi kelime çiftleri verilebilir.

3.1.7 Kısaltma Kullanılmasından Kaynaklanan Durumlar

Metin eşleme çalışmalarında karşılaşılan bir durum da birden çok kelimeden oluşan bir metinin içerdiği kelimelerin bir veya daha fazlasının yerine bu kelimelerin baş harflerinin veya ilk bir kaç harfinin kullanılmasından kaynaklanan problemlerdir. Örnek olarak "Mustafa Kemal Mahallesi" yerine "M. Kemal Mahallesi" veya "M.Kemal Mahallesi" veya "M Kemal Mahallesi" olarak yazılabilmektedir.

Burada en büyük sorun kısaltmalarda bir standart olmayışıdır. Kısaltılan kelime veya harflerin herbirisinin sonuna nokta konulabildiği gibi, nokta yerine kısaltılan ifadelerin arasına boşluk karakteri de tercih edilebilmektedir. Bu nedenle içerisinde kısaltma içeren metinlerin eşlenmesi veya benzerliklerinin bulunması bir derece daha zor olabilmektedir.

3.1.8 "tuvar" , "tuar" veya "hane" Sözcüklerini İçeren Metinlerdeki Hatalar

Türkçe metinlerde karşılaşılan bir yanlış yazım durumu da içerisinde "tuvar", "hane" gibi ifadeler içeren metinlerin yanlış yazılmasından kaynaklanan durumlardır.

Örnek olarak, "labarauvar-labaratuar", "hastahane-hastane" kelime çiftleri genellikle sürekli olarak yanlış yazılan Türkçe kelimelerdendir.

3.1.9 Bitişik Hecelerdeki Harflerin Yerdeğiştirmesinden Kaynaklanan Durumlar

Yine Türkçe metin eşleme çalışmalarında karşılaşılan bir diğer sorun bitişik hecelerde bulunan harflerin ve genellikle de sessizlerin yanlışlıkla yer değiştirmesinden kaynaklanan durumlardır. Örnek olarak, "yaprak-yarpak", "toprak-torpak", "kirpik-kiprik" gibi kelime çiftleri verilebilir. Bu durumda genellikle bilgisayar kullanıcısı kelimeyi yazarken harflerin sırasını zihninden doğru olarak kodlarken parmaklar ile zihin arasında oluşan faz farkından kaynaklanabilmektedir. Ayrıca bu durumun oluşma nedenlerinin bir diğeri ise bazı kelime veya ifadelerin farklı kültür ve şivelere göre farklı telaffuz edilmeleri ve bu farklı telaffuzun da yazıya yanlış geçirilmesi olarak gösterilebilir.

3.1.10 Karakterlerin Klavyedeki Dizilişlerinden Kaynaklanan Durumlar

Bu durumlar genellikle Türkçe metin eşleme çalışmalarında veya herhangi bir yazı yazma, kayıt girme işlemi sırasında en çok rastlanılan veya düşülen hata durumudur. Genellikle insanlar bilgisayarda (eskiden de daktilo aynı şekilde düşünülebilir) yazı yazarken daha hızlı yazma (Shieber ve Nelken, 2005) çabası gösterilirken klavye üzerinde bir tuşa basacağı yerde yanlışlıkla klavye üzerinde diziliş konumu olarak bu tuşa yakın bir başka tuşa basabilmektedirler.

Örnek olarak "Erzurum" kelimesi yazılacakken "z" harfi yerine yanlışlıkla klavyede "z" harfine bitişik olan "x" harfine basılırsa metin "Erxurum" halini alır. Metin eşleme çalışmalarında bu durum ciddi bir sorun olarak karşımıza çıkmaktadır. Çünkü bu sebepten oluşabilecek hataların sayısı klavyedeki komşuluklar adedince olabilir ve kullanılan klavyenin türüne göre de değişebilir. Mesela Türkçe Q klavyede yazı yazarken bu sebeple oluşabilecek hata durumları Türkçe F veya İngilizce klavyede yazarken karşılaşılabilecek durumlardan farklı olması normaldir.

Bu hata durumu da genel olarak iki farklı şekilde oluşabilmektedir. Aşağıda bu durumun

oluşum şekilleri açıklanmıştır.

3.1.10.1 Bir Karakter Yerine Yakınındaki Farklı Bir Karaktere Basılması Durumu

Klavyeden herhangi bir veri girişi esnasında genellikle hızlı yazmaya çalışmaktan veya dikkatsizlikten bir karaktere basılacağına klavyede bu karaktere bitişik tuşlardan birisine basılmış olabilir. Yani kullanılan klavyenin türüne göre (Türkçe Q, Türkçe F veya İngilizce klavye) bir karakter yerine başka bir komşu karakter kullanılmış olabilir.

Örnek olarak Türkçe Q bir klavyede "Ferhat" kelimesi yazılmaya çalışılırken "h" harfine basılacakken, yanlışlıkla "h" harfi yerine bu harfin bitişiğindeki "j" harfine basılmış olabilir. Yani "Ferhat" kelimesi yerine "Ferjat" yazılmış olabilir.

Bu tip durumların olma olasılığı kullanılan klavyenin fiziksel yapısına göre değişebilir. Mesela "h" harfinin yerine "j" harfine basma ihtimali "u" harfine basma ihtimalinden daha yüksektir.

3.1.10.2 Bir Karaktere Basılırken Bir de Fazladan Yakınındaki Bir Karaktere Basılması Durumu

Bir önceki maddede açıklanan durumun farklı bir şekli ise bir metin yazılırken yine hızlı yazma gibi problemlerden dolayı bir tuşa basarken fazladan bir de basılmak istenen tuşun komşusu olan tuşlardan birisine basılmasından kaynaklanır.

Örnek olarak "kalemlik" kelimesi yazılacakken "kjalemlik" yazılmış olabilir. Bu durum da yine farklı klavye türlerinde farklı hatalı yazımlara neden olabilmektedir.

3.1.11 Farklı Format Kullanılmasından Kaynaklanan Durumlar

Yine metin eşleme veya metin benzerlik hesaplama çalışmalarında karşılaşılan bir diğer genel sorun da bazı verilerin ya da metinlerin çok farklı formatlarda yazılabilmeleridir. Mesela adres, tarih, telefon numarası gibi bilgiler çok farklı formatlarda yazılması mümkün olan bilgilerdir. Bir ailenin bireylerine oturdukları evin adresini bir yere not etmelerini istersek çok büyük bir ihtimalle bireylerden en az birkaçı aynı adresi farklı şekillerde yazacaktır. Veya aynı kişiye birkaç saat veya bir kaç gün aralıkla oturduğu evin adresini sorsak büyük ihtimalle her seferinde adresi farklı bir şekilde ifade edecektir.

3.1.11.1 Adres Verisinin Farklı Yazılmasından Kaynaklanan Durumlar

Adres türünden alanlar genellikle çok farklı yazılabilmektedir. Örnek olarak aynı adres aşağıdaki gibi farklı şekillerde yazılmış olabilir;

"Gümüşyolu Cad. No: 41/9 Bağlarbaşı Üsküdar/İstanbul" veya

"İcadiye Mahallesi Gümüşyolu Cad. No: 41/9 Bağlarbaşı Üsküdar/İstanbul" veya

"İcadiye Mah. Gümüşyolu Cad. No: 41/9 Bağlarbaşı Üsküdar/İstanbul" veya

"İcadiye M Gümüşyolu Cad. No: 41/9 Bağlarbaşı Üsküdar/İstanbul" veya

"İcadiye Mahallesi Gümüşyolu Caddesi Bina No: 41 Daire: 9 Bağlarbaşı Üsküdar-İstanbul" gibi yazılmış olabilir.

Bu nedenle adres alanı eşleme işleminin ancak akıllı ön işlem ve eşleme fonksiyonları ile yapılması kaçınılmaz olmaktadır. Adres alanları akıllı şekilde tokenlara parse edilmeli ondan sonra eşleme yapılmalıdır.

3.1.11.2 Tarih Verisinin Farklı Yazılmasından Kaynaklanan Durumlar

Yine çok farklı formatlarda yazılabilen veri tiplerinden birisi de tarih verileridir. Tarih kullanılan bilgisayar sisteminin ayarlarına bağlı olarak veya bu tarih bilgisini giren insanın alışkanlıklarına bağlı olarak farklı şekillerde yazılabilir.

Bu nedenle adres tipi verilerinde olduğu gibi tarih tipinden alanlar üzerinde de eşleme işlemi yapılmadan önce akıllı önişlem fonksiyolarından geçirilmelidir.

Ayrıca tarih bazen "date-time" bazen "date" şeklinde tutulabileceğinden yani bazı tarih ve saat veya sadece tarih şeklinde yazılmış olabilir.

Örnek olarak aynı tarih aşağıdaki gibi yazılmış olabilir;

"10/10/1980 21 37 00" veya

"10/10/1980" veya

"10.10.1980" veya

"10 Ağustos 1980" veya

"10 Ağu 1980" şeklinde yazılmış olabilir

3.1.11.3 Telefon Numarası Verisinin Farklı Yazılmasından Kaynaklanan Durumlar

Telefon numarası bilgisi ise farklı formatlarda yazılabildiğinden, mesela alan kodu ile telefon numarası bitişik veya ayrı ayrı olarak yazılabildiğinden, yine birbiri ile gerçekte aynı olan iki telefon numarası metin eşleme fonksiyonlarında veya "= =" operatörünce farklı olarak değerlendirilebilir.

Bu nedenle telefon/faks numarası türünden veriler eşleme fonksiyonlarına sokulmadan önce adres ve tarih tipi verileri gibi bir önişlem fonksiyonundan geçirilerek benzer formatlara getirilmelidir.

Mesela aynı telefon numarası aşağıdaki gibi farklı formatlarda yazılmış olabilir;

"+90 216 315 37 90" veya

"90 216 315 37 90" veya

"0 216 315 37 90" veya

"0216 315 37 90" veya

"+90 216 3153790" veya

"0 216-315 37 90" veya

"216-315 37 90" veya

"02163153790" şeklinde yazılmış olabilir.

3.2 Çalışmada Modellenen Veri Türleri

Günümüzde kullanılan veri tipleri çok çeşitlidir. Ad, adres, telefon numarası, tarih gibi genel veri/bilgi tipleri olduğu gibi değişik ihtiyaçlar sonucu ortaya çıkmış, şahsa özgü, şirkete özgü veya kuruma özgü veri tipleri bulunması da normaldir. Bizim burada bu veri tiplerinin hepsini listelememizin mümkün olmadığı ve farklı türde Türkçe metinlerin daha etkin ve doğru eşleme veya benzerliklerinin hesaplanmasına gayret edilen bu çalışmada bütün veri tiplerini göz önünde bulundurmamızın da zor olacağı açıktır. Bu nedenle biz bu çalışmada eşlenecek metinlerin veya veritabanı tablolarında bulunan verilerin tiplerinin bir kaç tip ile sınırlı tutarak çalışmamıza devam ettik. Bu veri tipleri aşağıda listelenmiştir.

3.2.1 İsim Tipinden Veriler

Sık kullanılan veri tiplerinin birisi belki de en çok kullanılanı İsim verileridir. Çünkü günlük hayatımızda çok sıkça ya birilerinin ismini sorarız ya ismimizi soran kişiye ismimizi söyleriz ya da herhangi bir yerde ismimizi yazmak durumunda kalırız. İsim tipi verileri genellikle de soyisimler ile beraber kullanılmaktadır. İsim verileri yazılırken çok farklı şekillerde girilebilmektedir. Mesela "Halil İbrahim Canikoğlu" ismi aşağıdaki gibi farklı şekillerde yazılmış olabilir.

HALİL İBRAHİM CANİKOĞLU

Halil İbrahim Canikoğlu

H.İbrahim Canıkoğlu

H.İbrahim Canikoğlu

Halil İbrahim Canikoglu

İbrahim Canikoğlu

Bunlar genellikle isim/soyisim tipinden veri tutan veya bu tipten veri üzerinde çalışan şirket veya kuruluşların sıkça karşılaştığı durumlardır. Mesela bir şirketin yaptığı anketlerde genellikle ya ankete katılan kişi bu şekilde girişler yapabilir ya da ankete girilen veriler ilgili sisteme taşınırken bu şekil modifikasyonlara uğramış olabilir.

Yapılan çalışmada İsim tipinden veriler üzerinde bazı kurallar işletildikten sonra muhtemel yazım yanlışlarını da göz önünde bulunduran bir algoritma geliştirilmiştir. Aslında İsim tipinden verileri daha iyi eşleyebilmek ve benzerliklerini bulabilmek için işletilen kuralların bir kısmı sonraki maddelerde belirtilecek veri tipleri ile ortak olacaktır.

3.2.2 Adres Tipinden Veriler

Bu çalışmada adres tipinden verilerin eşlenmesi için yöntemler geliştirilmiştir. Adres tipinden verilerde karşılaşılabilecek durumlar modellenmiş ve karşılaşılan sorunlara çözümler aranmıştır. Adres tipinden veriler hem çok sayıda kelimeyi ve anahtar kelimeyi içerisinde bulundurabileceğinden ve farklı formatlarda yazılabileceğinden tek kelimelik veya birkaç kelimelik İsim ve Kısa Metin verilerinden farklıdır. Bu nedenle daha sonraki bölümlerde ele alınacak bu veri tipi üzerinde eşleme işlemlerinde farklı yol ve yöntemler izlenmiştir.

3.2.3 Kısa Metinler

İsim veya adres tipi olmayan herhangi bir türden Türkçe veri içeren metinlerin eşlenmesi için, Türkçe metinlerde karşılaşılan ve Bölüm 3.1'de detaylı olarak açıklanan hata durumları modellenmiş ve her bir durumda ilgili hatanın metinler üzerinde etkisini düzeltecek yöntemler geliştirilmiştir. Ve bu yöntem ve kuralların işletilmesi sonucu özellikle yazım yanlışlarından dolayı eşlenemeyen veya benzerlikleri daha düşük çıkan metinlerin eşlenmesi ve hesaplanandan daha büyük benzerlik gösterdiklerini gösteren yöntemler ve adımlar geliştirilmiştir.

Aslında Kısa Metin diye bir veri tipi olmadığı bir gerçektir. Her bir verinin kendine göre bir anlamı vardır. Ama biz Kısa Metin derken ilgili metinin ne türden (mesela mezun olunan Okul, Baba mesleği, vb.) bir veri içerdiğine bakmadan bu verilerin en fazla bir kaç kelimeden oluşan bir metin olduğunu varsayarak çalışmamıza devam ettik.

3.2.4 Uzun Metinler

Uzun metinler ise çok sayıda kelimeden oluşan metin alanları olarak düşünülmüştür. Çok sayıda kelimeden oluşması yönüyle Adres tipinden verilere benzemesine rağmen *Mahalle*, *Cadde* gibi anahtar kelimelerin olmaması nedeniyle de Adres tipinden verilerden farklılaşmaktadır. Yine Adres verilerinde olduğu gibi uzun metin alanlarında da eşleme çalışmaları için, iki metinin birbirine ne kadar benzediğinin hesaplanması için Edit Distance algoritmalarının kullanılmasının mantıklı olmayacağı kesindir. Bu nedenle bu tip verilerin benzerliklerinin hesaplanması için yine Türkçe metinlerde karşılaşılabilecek hata durumlarını da göz önünde bulunduran harf yerine kelime (aslında token) tabanlı bir benzerlik hesaplaması yapılacaktır. Tabii burada token'ların kendi içerisinde benzerlikleri söz konusu olacaktır.

3.3 Eşleme Sorunlarına Getirilen Çözümler ve Geliştirilen Yöntem

Önceki bölümlerde Türkçe metinler için bahsedilen hata durumlarını çözmek için bu hata durumları modellendi. Daha sonra bu hata durumlarından en fazla 3 tanesini içeren metin çiftlerinin benzerliklerini bu hataları giderdikten sonra hesaplayan bir yöntem geliştirildi. Yani geliştirilen yöntem hali hazırda en fazla 3 tane hatayı bulabilecek ve giderebilecek şekilde tasarlandı. Yöntem yine daha önce belirtilen "Ad", "Adres", "Genel Kısa Metin", "Uzun Metin" türünden veriler üzerinde çalıştırıldı. Herhangi bir metin çiftinde bu hata

durumlarından bir veya bir kaçı beraber bulunduğunda bu metinlerin nasıl farklılaşacağının karakteristiği belirlendi. Bir alanda bu hata durumlarından en fazla 3 tanesinin oluşabileceği kısıtı, varsayımı konuldu. Ad ve Kısa Metin tipinden veriler için benzer bir yol, Adres ve Uzun Metin tipinden veriler içinse ilkinden daha farklı bir yol izlendi. Bütün veri tipleri için, öncelikle metinlerin baş ve son kısımlarında bulunan muhtemel boşlukları yok etmek için eşlenmeye çalışılan veya benzerlikleri araştırılan iki metin de Kırpma(Trim) fonksiyonundan geçirildikten sonra ve Türk alfabesi için sorunsuz çalışan bir Küçük_Harfe_Çevir fonksiyonundan geçirildikten sonra her bir veri tipi için önceden tanımlı adımlar veya yöntemin kullanıcıya sunduğu parametrelere uygun olacak şekilde çalıştırılarak bahsedilen işlem gerçekleştirilmiş olacaktır.

Ad ve Kısa Metin tipinden metinlerin benzerliklerini hesaplamak için şöyle bir yol izlendi;

Öncelikle belirtilen 11 durumdan 9 tanesinden herhangi bir tanesinin -iki tanesi zaten boşluk ve büyük/küçük harf farklılığı idi– bu iki metinin herhangi birisinde olması durumu ele alındı. Bu iki metinde sırası ile bu durumlardan sadece bir tanesi kullanılarak metinlerin aynı metinler haline gelip gelmediği kontrol edilmektedir. Eğer herhangi bir hata durumunun tersine işletilmesi ile metinler eş metinler haline geliyorsa buna göre benzerlik hesaplanır. Eğer hata durumlarından herhangi bir tanesi metinleri eş metinler haline getirmeye yetmiyorsa bu hata durumlarından iki tanesinin beraber oluştuğu varsayımı ile algoritma devam ettirilir. Eğer bu durumların herhangi ikili kombinasyonu ile metinler eş metin haline geliyorsa benzerlik bu iki durumun işlettirilmesi sonucu hesaplanır. Eğer ikili kombinasyonlar ile de metinler eş metin haline getirilemiyorsa bu sefer de son olarak hata durumlarının üçlü kombinasyonları denenir ve herhangi üç durumun peşpeşe işlettirilmesi ile metinler eş metin haline getirilebiliyorsa benzerlik buna göre hesaplanır ve fonksiyon sonuçlanır.

Yöntem işletilirken her adımda metinler üzerinde, bu 9 durumdan kaynaklanan modifikasyonlu halleri de bellekte bir tabloda tutulmaktadır. Bu tablonun yapısı Tablo 3.1'deki gibidir.

Tablo 3.1 Hata durumlarına göre modifiye edilmiş metin çiftlerini tutan tablonun yapısı

METIN_1	METIN_2	SIRA
...	...	...

Bu tablo ise şöyle doldurulmaktadır. Öncelikle benzerlikleri hesaplanacak metin çifti ilk kayıta "0" (sıfır) sıra numarası ile atılır. Eğer bu kayıttaki iki metin aynı metinler ise işlem sonuçlanır. Eğer bu iki metin aynı değil ise sırası ile bu iki metinde bahsedilen 9 hata durumu oluşturularak oluşan modifiye metin çiftleri tabloya sıra numarası 1 olacak şekilde kayıt olarak eklenir. Ve eğer sıra numarası 1 olan kayıtlardaki metinler birbirlerinin aynı olmuş ise işlem başarı ile sonuçlanır. Eğer yine herhangi bir kayıtta bu iki metin aynı metinler haline gelmemişse bu sefer de sıra numarası 1 olan kayıtlarda bahsedilen 9 hata durumu tekrar işletilerek iki tane hata barındıran modifiye kelime çiftleri oluşturulmuş olur. Bu kayıtlar da sıra numarası 2 olacak şekilde tabloya eklenir. Sıra numarası 2 olan kayıtlarda da bu iki metinin aynı olduğu en az bir kayıt varsa işlem başarı ile sonuçlanır. Yoksa sıra numarası 2 olan kayıtlardaki metinlerde 9 hata durumu tekrar işletilerek 3 hatalı modifiye kelime çiftleri oluşturulur ve ilgili tabloya sıra numarası 3 olacak şekilde eklenir. Ve eğer sıra numarası 3 olan kayıtlarda da bu iki alandaki metinler birbirinin aynı olmuşsa işlem yine başarılı sonuçlanır. Eğer 3 hata durumununda işlettirilmesine rağmen metinler aynı metin haline gelmemişlerse bu sefer sıra numarası 0, 1, 2 ve 3 olan kayıtlardaki kelime çiftlerinden benzerlik oranı en fazla olan çiftin benzerliği geri döndürülmektedir. Bu adımdaki benzerlik oranı hesabı ise Levenstein Edit Distance Similarity yöntemine göre hesaplanmaktadır.

Adres tipinden metinlerin benzerliklerini hesaplamak için şöyle bir yol izlendi;

Öncelikle Adres tipinden veriler çok farklı formatlarda yazılabildiğinden, ilgili adres metinleri bir ön işlemden geçirilerek "Mahallesi", "Mah.", "C.", "Cad.", "Caddesi" gibi farklı şekillerde yazılma ihtimali bulunan anahtar kelimeler yok edildi. Daha sonra kalan metinler kelime bazında tokenlarına ayrılarak, birinci metindeki kelimeler bir tabloya, ikinci metindeki tablodaki kelimeler ise diğer bir tabloya alınmaktadır. Ve daha sonra bu iki tablodaki aynı veya benzer olan kelimelerin sayısına göre bir benzerlik hesaplaması yapılmaktadır. Benzerlik hesaplaması yapılırken Ad ve Kısa Metinler için yapılan yöntem burada da işlettirilerek birbirine benzer olan kelimeler bulunmaktadır. Yani Adres tipinden metinlerin benzerliklerinin hesaplanması yapılırken her bir kelime çifti için daha önce bahsedilen metin benzerlik hesaplama tekniği kullanılarak çalışmanın çekirdeği olan Türkçe metinler için hata durumları bu veriler üzerinde de gözönünde bulundurulmuştur.

Uzun Metin tipinden metinlerin benzerliklerini hesaplamak için şöyle bir yol izlendi;

Adres tipinden verilerdekine benzer bir yöntem izlenmesine rağmen bu tip verilerde metinler

ön işlemden geçirilmeden doğrudan tokenlarına ayrılarak aynı veya benzer kelimelerin sayısına göre iki metinin benzerliği hesaplanmaktadır. Burada ön işlemden geçirilmeme sebebi genel Uzun Metinler için tanımlı olan ve farklı formatlarda yazılabilecek anahtar kelimelerin bulunmayışıdır.

Yukarda belirtilen Ad, Adres, Kısa Metin ve Uzun Metin tipinden veriler içeren Türkçe metinlerde karşılaşılan 11 hata durumunu ilgili metinlerden gidermek için her bir hata durumu için atomik çözümler üretilmiştir. Bu atomik çözümlerin bir, iki veya en fazla üç tanesi bir metin çifti üzerinde çalıştırılarak beklenen benzerlik sonucu bulunacaktır. Daha önce de belirtildiği gibi bu 11 durumun tamamı Türkçe metinlere özgü veya Türkçe'ye özgü problemler değildir, bunlardan bazıları ile Türkçe dili dışındaki metinlerde de karşılaşılabilir ve bu durumlara farklı farklı çözümler sunulmuştur. Bu çalışmanın önemi Türkçe metinlerde karşılaşılabilecek eşleme veya daha tutarlı benzerlik hesaplama sorunlarının beraberce ele alınıp çözüm aranmasıdır.

Aşağıda her bir durum için belirtilen çözümler açıklanmıştır.

3.3.1 Metin Başındaki ve Sonundaki Boşluklara Getirilen Çözüm

Kolayca tahmin edileceği üzere metinler herhangi bir işleme sokulmadan önce baş ve sonlarında bulunan boşlukları kırpan Kırpma(Trim) fonksiyonundan geçirildi.

3.3.2 Büyük-Küçük Harf Farklılıklarına Getirilen Çözüm

Metinlerin Büyük Küçük harf farklılıklarından dolayı benzerliklerinin düşük hesaplanmaması için her iki metinin de ya bütün harfleri büyük harfe ya da bütün harfleri küçük harfe çevrilir. Biz çalışmamızda genellikle küçüğe çevirmeyi tercih ettik.

Burda dikkat edilen en önemli husus, bu çevirme işleminin Türkçe metinler için doğru çeviriyi yapmasıdır. Çünkü genellikle verilen bir metini Büyük harfe veya Küçük harfe çeviren fonksiyonlar Türkçe metinler üzerinde doğru çalışmamaktadırlar. Mesela "ISPARTA" kelimesi Türkçe dil desteği olmayan bir Küçük harfe çevirme fonksiyonundan geçirilirse sonuç hatalı olarak "isparta" olur. Çünkü İngilizce'de büyük "İ" harfi olmadığından "I" harfinin küçük hali "i" olur ki bu da Türkçe metinler için yanlış sonuçlar doğurur.

3.3.3 Türkçe Karakter Kullanılmasından Kaynaklanan Durumlara Getirilen Çözüm

Yine önceki bölümlerde üzerinde durulduğu gibi herhangi bir nedenden dolayı bir metinde bulunan ve Türkçe alfabeye mahsus olan "ı", "ü", "ğ", "ş", "ç", "ö" harflerinin yerine bunların eşleri olan "i", "u", "g", "s", "c", "o" harfleri bulunabilmektedir. Bu nedenle iki metin arasında farklılığa neden olan bu durumu ortadan kaldırmak için bu Türkçe alfabeye özgü harfler bunların eşleri ile Yer Değiştirilirler.

Tabii buradaki *Yer Değiştirme* tam Yer Değiştirme fonksiyonu değil bu çalışma için geliştirilmiş özel bir fonksiyondur. Çünkü eşlenecek veya benzerliği hesaplanacak metinlerde bulunan bu harf çiftleri karşılıklı olarak bulundukları metin içerisinde aynı lokasyonlarda iseler Yer Değiştirme işlemine tabii tutulurlar. Aksi takdirde hatalı bir yer değiştirme işlemi yapılma ihtimali doğar. Yine iki metindeki Yer Değiştirmeler direkt harflerin indislerine göre de yapılması uygun olmaz çünkü işlem yapılacak iki metinde daha farklı hata durumları da oluşmuşsa harflerin metinler içerisinde indisleri karşılıklı olarak aynı olmayabilir. Bu nedenle belirtilen harf çiftlerinin doğru lokasyonlardaki çiftlerinin birbirleri ile yer değiştirilmesi için şöyle bir yöntem izlendi;

Birinci metinde "ı", "ü", "ğ", "ş", "ç", "ö" harflerinden herhangi birisi varsa bu harflerin metin içerisinde öncesinde (varsa tabii) ve sonrasındaki 2'şer harfi de alınarak en fazla 5 harften oluşan bir metin parçası oluşur. Bu metin m1 = xx**t**xx şeklinde oluşmuş olabilir. Bu notasyonda x herhangi bir harfin yerine geçebilir, t ise Türkçe karakterlerden herhangi birisidir. Daha sonra m2 = xx**t'**xx şeklinde ikinci bir metin parçası oluşturulur. Ve ikinci metinde m2 metini varsa m1 ile yer değiştirilir ve böylelikle iki metinin ilgili kısımları eşit hale gelir ve bu yer değiştirme işlemi, değiştirilecek harfin önündeki ve sonundaki 2 şer harf ile beraber yapıldığından yanlış lokasyondaki Türkçe karakterinin değiştirilmesi önlenmiş olur.

Burada belirtilen harften önceki ve sonraki karakter sayısı olan 2 aslında çalışmada parametrik tutulmuştur. Yani herhangi bir eşleme için bu 2 rakamı 1'e düşürülerek algoritma çalıştırılabilir. Ama bu rakamı 2 den 1'e düşürmek geliştirilen yöntemin bazı durumlar için hatalı çalışmasına neden olabileceği düşünüldüğünden bu değerin varsayılan değeri 2 olarak belirlenmiştir.

Bu bölümde belirtilen ve çalışma için kullanılan bu özel Yer Değiştirme fonksiyonu iyi anlaşıldıktan sonra ileriki bölümlere geçilmelidir, çünkü birçok durum için benzer bir şekilde *Yer Değiştirme* fonksiyonu kullanılacaktır.

3.3.4 D-T ve B-P Harf Çiftlerinin Birbirlerinin Yerine Kullanılmasından Kaynaklanan Durumlara Getirilen Çözüm

Yukarda Türkçe alfabelerdeki karakterler için yapılan işlemlerinin bir benzeri de "d-t", "b-p" çiftleri için de yapılarak bu durum çözülmüştür. Sadece yukardakinden farklı olarak m1=xx**t**xx ve m2=xx**t'**xx şeklinde oluşturulan Değişim metinlerinde t ve t' "d-t", "b-p" çiftleri olacaktır. Bahsedilen Yer Değiştirme fonksiyonu bu tip bir hatayı içeren metinler üzerinde çalıştırılınca metinlerin birbirlerine olan benzerlikleri artacaktır.

3.3.5 Aynı Harfin Peşpeşe İkilemesinden Kaynaklanan Durumlara Getirilen Çözüm

Metinlerin birinde herhangi bir harfin çiftlemesi olmuşsa yani peşpeşe gelmişse ve ikinci metinde de bu çiftleme yoksa m1 = xx**tt**xx ve m2 = xx**t**xx şeklinde Değişim metinleri oluşturulur. Yine burada *x* karakteri herhangi bir karakter (birbirinden bağımsız) ve *t* karakteri de çiftleyen karakterdir. Burda ikinci metinde bulunan xx**t**xx metini ilk metinden oluşturulan xx**tt**xx metini ile değiştirilerek metinler aynı metinler haline getirilir en azından benzerlik oranları daha artmış olur.

3.3.6 Metinlerin İkinci Harflerinde "i", "ı" veya "u" Harfinin Bulunmasından Kaynaklanan Durumlara Getirilen Çözüm

Bu durumda da eğer birinci metinin ikinci harfi belirtilen seslilerden birisi ise ve ikinci metin, birinci metinin ilk harfi ve üçüncü harfinin uçuca eklenmesi ile başlıyorsa ilk metinin ilk üç harfi ikinci metinin ilk iki harfi ile yer değiştirilerek metinler benzeştirilir. Mesela ilk kelime "gurup" ve ikinci kelime "grup" ise ilk kelimedeki ilk üç harf olan "gur" yerine ikinci metinin ilk iki harfi olan "gr" yazılarak metinler benzeştirilir.

3.3.7 Kısaltma Kullanılmasından Kaynaklanan Durumlara Getirilen Çözüm

Kısaltmalar için bu çalışmada bir çözüm sunulmamıştır. Daha ileri bir çalışmaya dahil edilmesi uygun görülmüştür.

3.3.8 Tuvar-Tuar-Hane Sözcüklerini İçeren Metinlerdeki Hatalara Getirilen Çözüm

Burada da "tuar", "tuar", "hane", "ane" içeren kelime çiftleri uygun şekilde düzelten bir fonksiyondan geçirilerek metinler benzeştirilir. Mesela metinlerden birisi labaratuvar ve

diğeri ise labaratuar ise ilk metinde bulunan "tuvar" metini ikinci metinde bulunan "tuar" metini ile Yer Değiştirilerek metinler benzeştirilir.

3.3.9 Bitişik Hecelerdeki Harflerin Yerdeğiştirmesinden Kaynaklanan Hatalara Getirilen Çözüm

Metinlerin orta hecelerinde bulunan ve bitişik hecelerde bulunan iki harfin yanlışlıkla ters sırada yazılmasından dolayı oluşan hatalar için yine Yer Değiştirme fonksiyonumuz uygun şekilde çalıştırılarak metinler benzeştirilir. Burada birinci metin ortalarında "tr" şeklinde bir harf çifti bulundursun, ikinci metin ise belirtilen lokasyonda "tr" yerine "rt" şeklinde yazılmış olsun. Yine ilk metinde bulunan "tr" metin parçasının öncesindeki ve sonrasındaki 2'şer harfler ile beraber alınarak m1 = xx**tr**xx ve m2 = xx**rt**xx şeklinde Değişim metinleri oluşturulur ve ikinci metindeki xx**rt**xx metini xx**tr**xx metini ile değiştirilerek metinler birbirlerine benzetilir.

3.3.10 Karakterlerin Klavyedeki Dizilişlerinden Kaynaklanan Hatalara Getirilen Çözüm

Çalışmanın önemli bir kısımlarından biri bu kısımdır. Genellikle en çok yapılan hatalar bu tip hatalardır. Daha hızlı yazılmaya çalışılan bir metin içerisindeki bulunması gereken harfler yerine klavyede bu harfe bitişik veya yakın dizilen başka bir karaktere basılmaktadır.

Bu çalışma için sadece Türkçe Q klavye üzerinden giriş yapıldığını ve bir klavyede bir karakter yerine bu harfe komşu bütün karakterlere yanlışlıkla basılma olasılığının aynı olduğunu varsayılmıştır. Daha ileriki bir çalışma için daha farklı klavye türlerinde ve farklı olasılıklar oluşturularak çalışma tekrarlanabilir.

3.3.10.1 Bir Karakter Yerine Yakınındaki Farklı Bir Karaktere Basılması Durumuna Getirilen Çözüm

Bu tip hataların etkisini metinlerden gidermek için şöyle bir yöntem geliştirildi; Birinci metinin baştan itibaren sırasıyla her seferinde bir karakteri alınacak şekilde bütün karakterleri teker teker alındı ve yine önceki durumlar için bahsedilen, önceki ve sonrasında bulunan 2 şer karakter ile beraber alacak şekilde m1 = xx**t**xx şeklinde bir metin oluşturulmaktadır. (Burada

ilk metinin ilk harfi için önceki 2 karakter diye bir şey söz konusu olmayacağından m1 = **t**xx şeklinde bir metin olacağı tahmin edilebilir). Daha sonra alınan t karakteri için bu karakterin Türkçe Q klavye dizilişinde bulunan komşuları sırasıyla alınarak m2 = xx**k**xx şeklinde ikinci Değişim metini oluşturulur. Ve ikinci metinde eğer xx**k**xx şeklinde bir kısım varsa bu kısım xx**t**xx ile değiştirilerek metinler birbirlerine benzeştirilirler.

3.3.10.2 Bir Karaktere Basılırken Bir de Fazladan Bitişiğindeki Bir Karaktere Basılması Durumuna Getirilen Çözüm

Bu durum için de bir önceki bölümde anlatılana benzer bir yol izlenmiştir. Ama burada bir karaktere basılırken bir de fazladan bir komşu karaktere daha basılması durumuna çözüm arandığından oluşturulacak Değişim metinleri biraz farklı olacaktır. Yöntem bu sefer şöyle çalışacaktır;

Birinci metinin baştan itibaren sırasıyla her seferinde bir karakteri alınacak şekilde bütün karakterleri teker teker alındı ve yine önceki durumlar için bahsedilen önceki ve sonrasında bulunan 2'şer karakter ile beraber alacak şekilde m1 = xx**t**xx şeklinde bir metin oluşturulmaktadır. Daha sonra alınan t karakteri için bu karakterin Türkçe Q klavye dizilişinde bulunan komşuları sırasıyla alınarak m2 = xx**tk**xx ve m3 = xx**kt**xx şeklinde iki adet Değişim metini oluşturulur. Ve ikinci metinde eğer xx**kt**xx veya xx**tk**xx şeklinde bir kısım varsa bu kısımlar xx**t**xx ile değiştirilerek metinler birbirlerine benzeştirilirler. Burada iki tane Değişim metinin oluşturulma sebebi kolayca tahmin edileceği gibi fazladan basılan karakterin ilgili karaktere bastıktan önce veya bastıktan sonra basılabileceği ihtimalinin bulunmasındandır.

3.3.11 Farklı Format Kullanılmasından Kaynaklanan Durumlara Getirilen Çözümler

Aslında format farklılıklarından dolayı içerik olarak aynı olan metinlerin farklı metinler gibi olması önceki durumlardan temelde farklı olmasına rağmen metin eşleme ve metin benzerlik hesaplama çalışmalarında benzer sorunlara neden olduklarından bu durumlar da çalışmaya dahil edilmiştir. Bu çalışmada farklı şekilde formatlanabilecek veri türlerinden sadece Adres verileri ele alınmıştır. Aynı şekilde Tarih ve Telefon Numarası gibi alanlar de benzer bir çalışmada ele alınabilir.

3.3.11.1 Adres Verilerinin Farklı Formatta Yazılmalarına Getirilen Çözüm

Adres bilgisi çok farklı şekillerde yazılabilen bir bilgi tipi olduğundan verilen iki adres yazım olarak birbirinin aynı olmasa bile gerçekte aynı adresi işaret ediyor olabilir. Bunun için şöyle bir yöntem izlendi; Adres verileri nasıl yazılırsa yazılsın akıllı bir şekilde parse edilerek belirli bir formata sokan bir ön işlemden geçirildikten sonra benzerlikleri araştırılacaktır. Önişlemden geçirilen metinler daha sonra tokenlarına ayrılarak, tokenların benzerliğine göre adreslerin benzerliği hesaplanmaktadır. Burada ilgili tokenlarda, Türkçe Hata Durumlarının da var olma ihtimali gözetilerek benzerlik araştırılacaktır. Mesela aynı adres birinde farklı bir formatta diğerinde de farklı formatta yazılmış olabileceği gibi aynı zamanda yazılırken sokak ismide yanlışlıkla hatalı yazılmış olabilir.

3.3.11.2 Tarih Verilerinin Farklı Formatta Yazılmalarına Getirilen Çözüm

Tarih verilerinin bir formata getirilerek eşlenme veya benzerliklerinin bulunması daha ileri bir çalışmaya bırakılmıştır. Bu çalışmanın kapsamına dahil edilmemiştir.

3.3.11.3 Telefon Numarası Verilerinin Farklı Formatta Yazılmalarına Getirilen Çözüm

Telefon numarası verilerinin bir formata getirilerek eşlenme veya benzerliklerinin bulunması daha ileri bir çalışmaya bırakılmıştır. Bu çalışmanın kapsamına dahil edilmemiştir.

3.4 Geliştirilen Yöntemin Örnek Veriler Üzerinde Denenmesi

Geliştirilen yöntem temelde farklı türde bilgi içeren Türkçe iki metinin benzerliklerini daha iyi hesaplayabilecek bir yöntemdir. Geliştirilen yöntemin başarısını test etmek için metin eşleme yöntemlerine çokça ihtiyaç duyulan veritabanı tablolarında tekrarlı kayıtları veya birbirine benzer kayıtları bulmak üzere bir uygulama yazılımı gerçekleştirildi.

Veritabanı olarak Oracle 10g seçildi ve uygulamaya arayüz geliştirmek için Microsoft Visual Studio 2005 C#.Net platformu seçildi. Yazılan uygulama Türkçe metinler içeren Oracle tablolarındaki tekrarlı kayıtları veya benzer kayıtları araştırıp bulduğu için Oracle DetecTR olarak adlandırıldı.

Uygulama 2 modda çalışmaktadır. Ya bağlanılan veritabanında kullanıcının seçtiği tek bir veri tablosunda belirtilen alanlara göre minimum benzerlik kriterlerinin üstünde benzerlik

gösteren alanlara göre tekrarlı kayıtları bulan tek tablo modu veya kullanıcının seçtiği iki tablonun yine belirtilen alanlarına göre tekrar sayılabilecek kayıtlarını getiren çift tablo modu olmak üzere iki modda çalışmaktadır. Yazılan uygulama programı hakkında daha detaylı bilgi Bölüm 3.4.2 te verilecektir.

3.4.1 Geliştirilen Yöntemin Başarısının Ölçülmesi

Yapılan çalışmada en çok zorlanılan konulardan birisi de yapılan çalışmanın başarısının nasıl ölçüleceğiydi. Geliştirilen yöntemin başarısını ölçmek için, içerisinde belirtilen ve çalışmada modellenen Türkçe Hata durumlarının bir veya bir kaçını içeren metin çiftlerinin bulunması gerekmekteydi. Bunun için öncelikle şöyle bir yol denendi;

Örnek bir Türkçe sözlüğü ve örnek bir Türkçe kişi isimleri sözlüğü elde edildi. Daha sonra buradaki kelime, sözcük veya kişi isimlerinden rastgele seçimler yaparak ve gerektiğinde bu seçilen kelimelerden bir kaçı beraber kullanılarak 1000 kayıtlı bir tablo oluşturuldu. Bu tablonun bazı alanları çalışmada modellenen Ad, Adres, Genel Kısa Metin ve Uzun Metin tipinden olacak şekilde oluşturuldu ve belirtilen 1000 kayıt ilgili alanın tipine uygun veriler ile doldurulduu.

Daha sonra bu tablodan iki adet yeni tablo türetildi. Türetilen bu tabloların ilki orjinal tablodaki satırlara ek olarak bazı satırların içerdikleri verilerin belirtilen Türkçe Hata Durumlarının bir veya birkaçını içerecek şekilde bozulmaları ile oluşan kayıtlardan teşekkül ettirilmiştir. Ve bu tablo çalışmanın bir parçası olan bir veritabanı tablosundaki tekrarlı kayıtların bulunması için kullanılacaktır. İkinci tablo ise orjinal tablodaki bazı kayıtların yine benzer şekilde bozularak oluşturulan yeni kayıtlardan oluşturulmuştur. Bu tabloda orjinal tablo ile karşılaştırılıp iki tablo arasındaki ortak veya benzer kayıtları bulmak için kullanılacaktır.

Fakat çalışmanın ilerleyen safhalarında bu şekilde elde edilen verilerin gerçekçi veriler olmadığı ve test verisi olmak üzere kullanılmalarının doğru bir adım olmadığına karar verildi. Çünkü gerçekte modellenen hata durumlarına göre kalıp hataların oluştuğu gözlemlendi. Gerçek kullanıcıların sebep oldukları hataların ise, bu hatalara göre çok daha çeşitli ve farklı dağılımlarda oldukları tesbit edildi. Bu nedenle geliştirilen yöntemin başarısını ölçmek için çok sayıda kullanıcının farklı veri giriş yöntemleri kullanarak girdikleri hatalı metin ve doğru metin çiftleri toplanarak test verisi olarak kullanıldı.

Türkçe metinlerin benzerliğinin daha tutarlı bir şekilde hesaplanması için geliştirilen yöntemin başarısını ölçmek için farklı seviyelerde ve farklı özelliklere sahip bilgisayar kullanıcılarından farklı şekillerde kısa metinler girilmesi istendi. Girilen kısa metinler arasından hatalı olanlar seçilerek yapılan çalışma için test verisi olarak kullanıldı.

Bahsedilen kişiler tarafından farklı şekillerde girilen hatalı 344 kısa metin ve bu metinlerin doğru halleri elde edildi. Daha sonra bütün bu metin çiftleri için aşağıdaki 3 yönteme göre benzerlik oranları hesaplandı ve bu 3 yöntemin bu veriler üzerindeki başarısı karşılaştırıldı.
Bu yöntemler;

- Standart Edit Distance Benzerliği yöntemi
- Jaro-Winkler yöntemi
- Bizim tarafımızdan geliştirilen yöntem

Bu çalışmada başarı kriteri, hesaplanan Metin Benzerlik Oranının büyüklüğüdür. Çünkü burada her 3 yönteme gönderilen farklı yazılmış metin çiftlerinin gerçekte aynı metinlerin hatalı yazım nedenleri dolayısıyla farklılaşmış halleri olduğundan bu metinler aynı metinlerdir. Dolayısıyla hangi yöntem daha yüksek benzerlik hesaplarsa o yöntem daha başarılı sayılmıştır.

Burda bahsedilen diğer iki yöntemi biraz daha açacak olursak;

Standart Edit Distance Benzerliği: Verilen iki metinin en az kaç Ekleme/Silme/Değiştirme işlemi sonucunda aynı metinler haline geleceğinin ölçüsü olan Levenshtein Edit Distance değerine dayanan bir benzerlik hesaplama yöntemidir.

Jaro-Winkler Benzerliği: Jaro-Winkler standart Edit Distance Benzerliğinden farklı olarak daha akıllı bir şekilde muhtemel yazım yanlışlarını da göz önünde bulundurarak daha gerçekçi benzerlik oranı hesaplayan bir yöntemdir. Normal olarak birbirine daha az benzer gibi görünen iki metin bu yönteme göre daha yüksek oranda benzerlik gösterirler.

Burada ayrıca geliştirilen yöntemin başarısını test etmek kullanılacak olan, Jaro-Winkler ve Edit Distance Similarity yöntemleri için Oracle 10g *UTL_MATCH* packaginde bulunan *Jaro_Winkler_Similarity* ve *Edit_Distance_Similarity* fonksiyonlarının gerçeklemeleri Büyük Küçük harfe duyarlı olduklarından test verilerinde büyük küçük harf farklılıkları olan metin çiftlerinde başarıları iyice düştüğünden, bu fonksiyonlara gönderilen test verileri öncelikle

Türkçe için yazılmış küçük harfe çevirme fonksiyonundan geçirilmiştir. Yani testin daha bu aşamasında bile belirtilen iki yöntemin Türkçe metinler için eksik kaldıkları gözlemlenmiştir.

3.4.2 Geliştirilen Uygulama (Oracle DetecTR)

Daha önce de bahsedildiği gibi geliştirilen Türkçe Metin Benzerlik Hesaplama yönteminin başarısını örnek bir veritabanında denemek için bir uygulama programı yazıldı ve Oracle 10g veritabanında tekrarlı veya benzer kayıtları bulmak üzere tablolar oluşturuldu. Ayrıca yazılan uygulama sayesinde, istenirse herhangi bir kullanıcı kendi veritabanına bağlanarak programı kendi tabloları üzerinde çalıştırabilir.

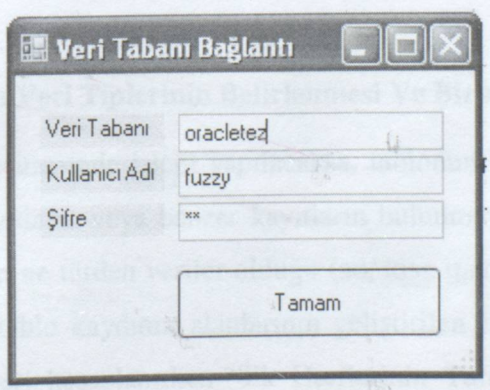
Programa, Oracle veritabanı üzerinde bulunan Türkçe metinler içeren tablolardaki tekrarlı kayıtları araştırdığı için Oracle DetecTR isminin koyulması uygun bulundu.

Genel olarak geliştirilen metin benzerlik hesaplama işlemleri veritabanı tarafında PL/SQL dili kullanılarak package içerisinde gerçekleştirildi. VS.Net tarafında yazılan uygulama ise kullanıcı ara yüzüdür. Yani algoritmanın çalışması için gerekli parametrelerin bu veritabanı packagine göndermek için yazılmıştır. Çalışmanın çekirdek kısmı belirtilen PL/SQL packagidir.

Programın çalıştırılması esnasında yapılan veya yapılması gereken işlemler programın çalıştırılması ile alınan ekran görüntüleri de eklenerek anlatılacaktır.

3.4.2.1 Veritabanına Bağlanma

Öncelikle tekrarlı kayıtların bulunduğu Oracle veritabanı sistemine uygun parametreler verilerek bağlanılır.

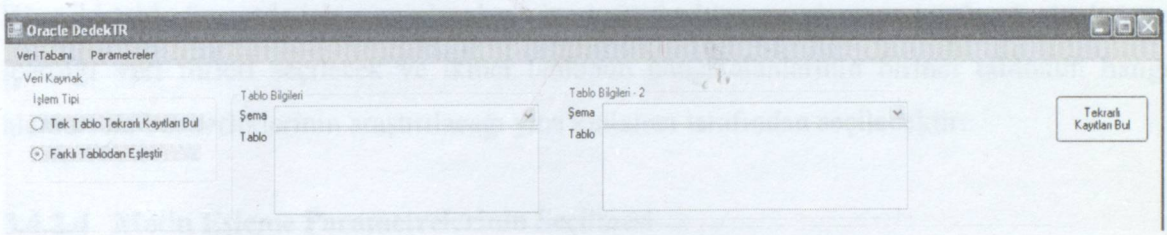


Şekil 3.1 Veritabanına Bağlanma Formu

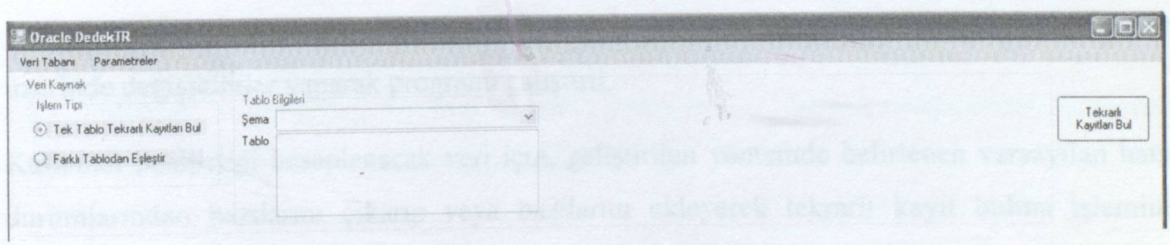
Burada bağlanılacak veritabanın Global Adı ve Kullanıcı Adı ve Şifre bilgileri doğru bir şekilde girilmelidir.

3.4.2.2 Çalışma Modunun ve Üzerinde Çalışılacak Tabloların Seçilmesi

Bu aşamada programın ilgili arayüzünden iki tablo arasındaki tekrarlı kayıtların mı yoksa tek tablo içerisindeki tekrarlı kayıtların mı bulunmak istendiğine dair çalışma modu ve bu modda çalıştırılmak üzere ilgili veribanı tablo veya tabloları seçilir. Eğer tek tablo üzerinde işlem yapılacaksa sadece bir tablo seçilecek, eğer iki tablo üzerinde işlem yapılacaksa iki tane veritabanı tablosu seçilecektir.



Şekil 3.2.a Veri kaynak seçimi – Çift tablo tekrarlı kayıt arama

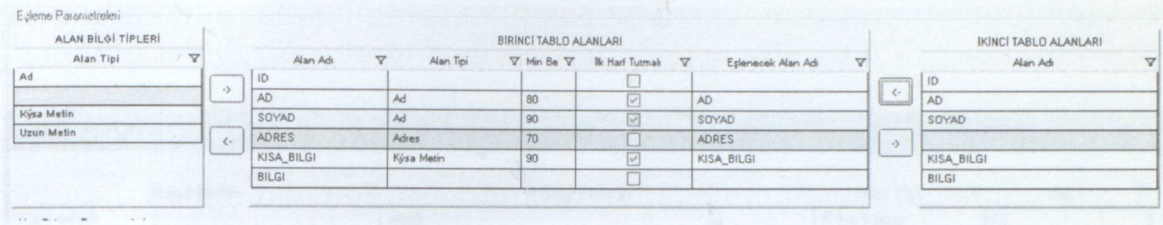


Şekil 3.2.b Veri kaynak seçimi – Tek tablo tekrarlı kayıt arama

3.4.2.3 Tablo Alanlarının Veri Tiplerinin Belirlenmesi Ve Birbirlerine Map Edilmesi

Bu aşamada eğer tek tablo üzerinde işlem yapılacaksa, tablonun hangi alanları göz önünde bulundurularak tekrarlı kayıtların veya benzer kayıtların bulunmaya çalışılacağı ve bu tablo alanlarının içerdiği verilerin ne türden veriler olduğu (ad, kısa metin, adres, vb.) seçilecektir. Ayrıca karşılaştırılacak 2 tablo kaydının alanlarının geliştirilen metin benzerlik hesaplama yöntemine göre benzerlikleri hesaplanırken "İlk Harflerinin Tutup/Tutmama" zorunluluğu seçeneği ve minimum benzerlik puanları parametrik olarak seçilmektedir. Normalde bu iki

parametrenin varsayılan değerleri tablonun içerdiği veri tipi seçilince otomatik olarak gelmektedir. Kullanıcı isterse bu parametreleri değiştirerek işlemine devam eder.



Şekil 3.3 Eşleme Tabloları Mapping

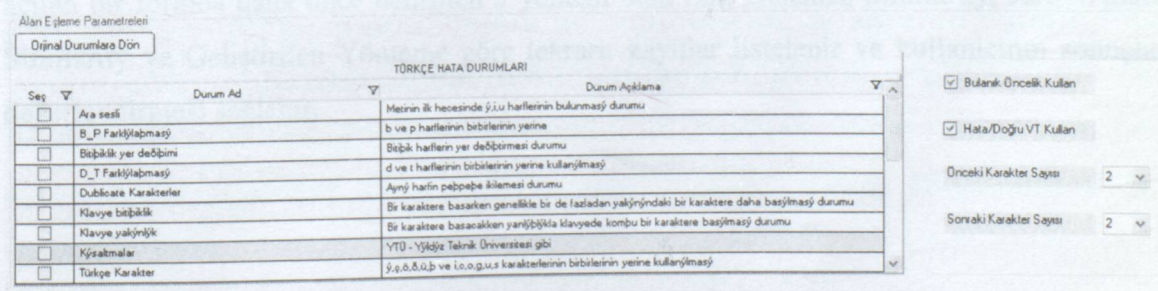
Eğer iki tablo üzerinde işlem yapılacaksa yine birinci tablonun eşlemeye katılacak alanlarının içerdiği veri türleri seçilecek ve ikinci tablonun hangi alanlarının birinci tablonun hangi alanları ile benzerliklerinin araştırılacağı yine kullanıcı tarafından seçilecektir.

3.4.2.4 Metin Eşleme Parametrelerinin Seçilmesi

Programın bu adımında iki alan arasındaki benzerlik hesaplanırken, çalışmaya temel teşkil eden Türkçe Hata Durumlarından hangilerinin kullanılacağı ve bu durumlar için yöntemin hangi parametrelerle çalıştırılacağı seçilmektedir.

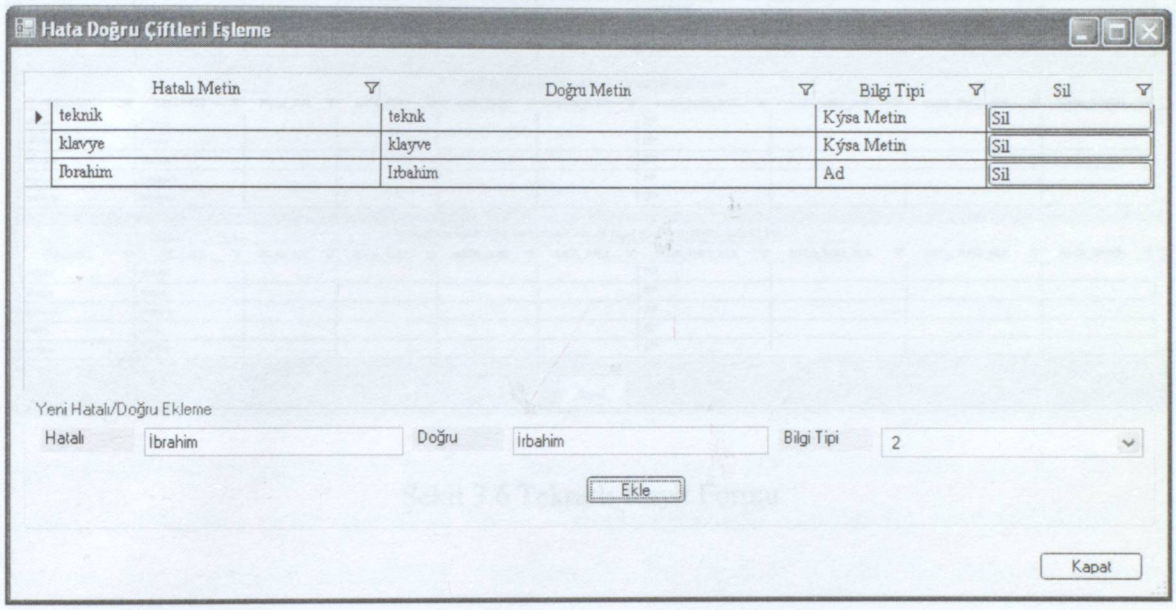
Kullanıcı isterse varsayılan parametre değerleri ile programı çalıştırır, isterse bu parametreler üzerinde değişiklikler yaparak programı çalıştırır.

Kullanıcı benzerliği hesaplanacak veri için, geliştirilen yöntemde belirlenen varsayılan hata durumlarından bazılarını çıkarıp veya bazılarını ekleyerek tekrarlı kayıt bulma işlemini gerçekleştirebilir.



Şekil 3.4 Eşleme Parametrelerinin Seçimi

Yine bu aşamada programın daha önceki çalıştırıldıklarında karşılaşılan ve bir kelime veya metinin yanlış yazılmış hali ve doğrusunun beraber bulundukları bir tabloyu kullanıp kullanmamasına dair parametreyi seçer. Kullanıcı isterse ilgili arayüzden bu tabloya kayıt ekleyip çıkarabilir.

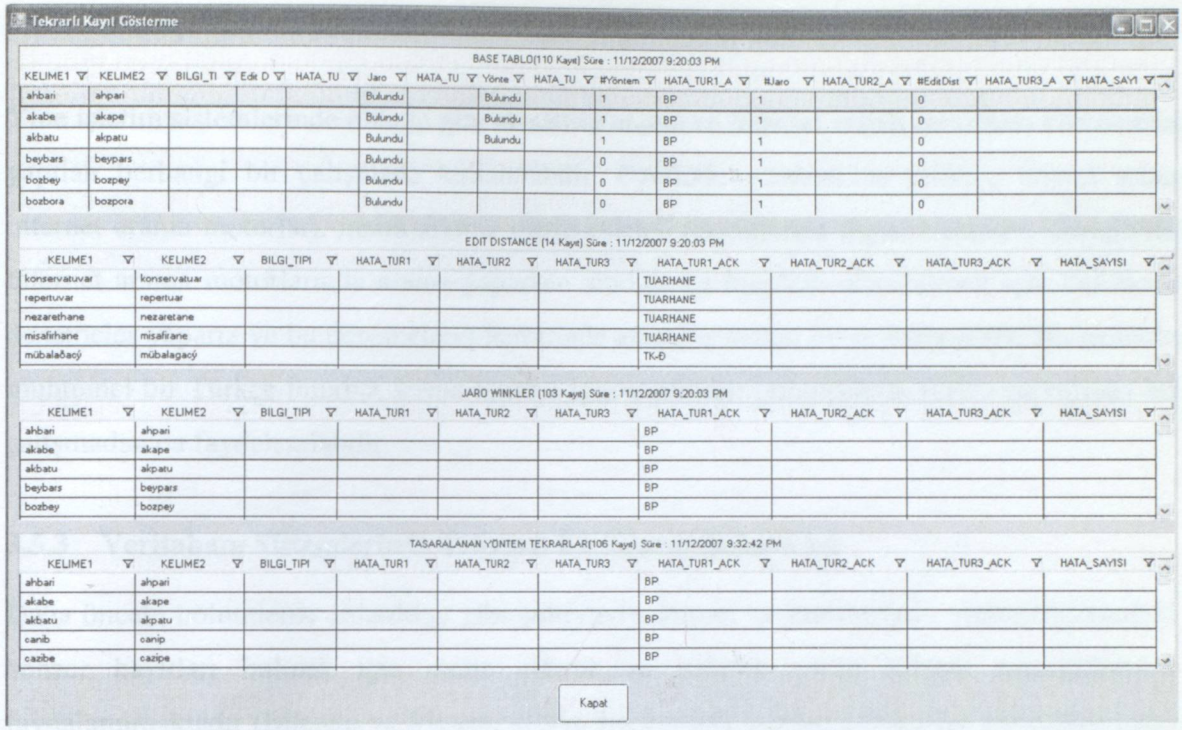


Şekil 3.5 Hatalı/Doğru Tablosu Düzenleme Formu

Ve yine metin benzerlik hesaplanırken önceki bölümlerde anlatılan özel Yer Değiştirme fonksiyonunda kaç önceki ve kaç sonraki karakter ile beraber değiştirme işleminin gerçekleştirileceğine dair parametre değerlerini de seçer.

3.4.2.5 Tekrarlı Kayıtların Bulunması

Kullanıcı bütün seçimlerini yaptıktan sonra son olarak tekrarlı kayıtları görüntüler. Programda açılan bir formda daha önce belirtilen 3 yöntem olan Edit Distance Similarity, Jaro-Winkler Similarity ve Geliştirilen Yönteme göre tekrarlı kayıtlar listelenir ve kullanıcının sonuçları değerlendirmesi sağlanır.



Şekil 3.6 Tekrarlı Kayıt Formu

3.5 Geliştirilen Yöntemin Muhtemel Kullanım Alanları

Geliştirilen yöntem Türkçe metin işleme konuları ile kesişen herhangi bir alanda kullanılabilir. Yöntem tam olarak bütün ihtiyaçlar için yeterli bir çalışma olmasa bile belirtilen ihtiyaç durumları için uyarlanabilir veya geliştirilen yöntemin bazı yetenekleri direk veya dolaylı olarak başka yöntemlere eklenebilir.

3.5.1 Türkçe İşletim Sistemleri

İşletim sistemlerinin çalışması ve çok sayıda fonksiyonu metin eşleme ve metin benzerlik hesaplamalarına dayanır. Çünkü aranılan bir dosya veya dizine ulaşma, içerisinde aranan metinin bulunduğu dosyaları getirme, iki dosyanın içeriğini birbiri ile karşılaştırma gibi temel birçok işlem metin eşleme tekniklerini kullanır. Bu çalışmada Türkçe metin arama, metin benzerlik hesaplama için yeni bir yöntem geliştirildiğinden Türkçe işletim sistemlerinde kullanılabilir.

3.5.2 Türkçe Arama Motorları

Yine işletim sistemlerinde olduğu gibi masaüstü arama ve internet arama motorları konusunda yapılan herhangi bir çalışmada kullanılabilir. Özellikle günümüzde gittikçe önemi artan internet arama motorları, metin arama, metin eşleme tekniklerine dayanmaktadır. Genellikle internet arama motorlarında arama yaparken aradığımız bilgilere ulaşabilmek için defalarca denemeler yaparız ve bu denemelerin herbirinde aranılan metini biraz değiştiririz. Bu nedenle muhtemel bir Türkçe internet arama motoru için yapılacak çalışmalarda bizim yaptığımız bu çalışmadan da faydalanılabilir.

3.5.3 Veritabanı Sistemlerinde Tekrarlı Kayıtların Bulunması

Daha önceki bölümlerde anlatıldığı gibi şirketlerin, kurum ve kuruluşların veritabanlarındaki benzer kayıtları bulmak için metin eşleme ve bulanık metin eşleme tekniklerinden faydalanılmaktadır (Bilenko ve Money, 2003). Bu konuda yapılan çalışmalar genellikle ticari uygulamalardır ve Türkiye'de bu konu üzerine yapılmış çok fazla çalışma yoktur. Bir kaç banka, anketlerden gelen kayıtların kendi veritabanlarındaki kayıtlar ile eşleştirebilmek için kuruma has uygulama programları yazdırmıştır. Bu nedenle yapılan bu çalışma, hem bu konuda yapılacak akademik çalışmalara öncülük edecektir hem de ticari uygulamalarda da faydalanılabilecek bir kaynak olacaktır.

3.5.4 Konuşma Tanıma

Konuşma tanıma çalışmaları için Türkçe çok elverişli bir dildir. Çünkü Türkçe okunduğu, telaffuz edildiği gibi yazılan bir dildir. Fakat konuşmayı tanıyan sistemin hassasiyeti ve kalitesi, konuşmayı yapan kişinin şivesi, ses tonu veya konuşmanın yapıldığı ortamdaki gürültü nedeniyle bu konuşma yazıya çevrilirken bazı hatalı durumlar oluşabilir.

Dolayısıyla bu hatalı durumların gözönünde bulundurularak ve muhtemel hata durumları giderilerek, gerektiğinde sesten metin olarak elde edilen ifadenin sözlük yardımı ile en çok benzeştiği kelime veya kelime gruplarını bulan bir çevirme işlemi daha başarılı olacaktır. Bizim yaptığımız çalışmada ise Türkçe'deki muhtemel hatalı durumlar giderilerek iki metinin benzerliğini daha başarılı şekilde hesaplayan bir eşleme yöntemi geliştirildiğinden konuşma tanıma çalışmalarında bu çalışmadaki kazanımlardan faydalanılabilir.

3.5.5 Türkçe OCR Çalışmaları

Yine konuşma tanımada olduğu gibi OCR çalışmalarında da bu çalışmadan faydalanılabilir. Çünkü günümüzde gelişmiş OCR programları, karakter karakter yerine bütün bir kelimeyi tanıyıp, belirtilen dilde yazım kontrolü yapıp ve ilgili sözlükteki metinler ile karşılaştırarak daha başarılı tanıma işlemi yapmaktadır. Yine bizim çalışmamız Türkçe metinlerin benzerliğini hesaplamaya dayalı bir çalışma olduğundan bu sözlük veritabanı ile karşılaştırma yaparken bizim çalışmamızdaki kazanımlar da göz önünde tutularak daha başarılı Türkçe dil destekli OCR ürünleri geliştirilebilir.

4. GELECEKTE YAPILABİLECEK ÇALIŞMALAR

Bu çalışmada Türkçe metinlerinin eşlenmesi ve birbiri ile olan benzerliklerin hesaplanması için yeni bir yöntem geliştirilmiştir. Burada esasen konuyla ilgili bir çerçeve oluşturulmaya çalışılmıştır. Ve geliştirilen yöntem parametrik bir yapıya sokularak sonraki çalışmalara bir başlangıç noktası olabilecek hale getirilmiştir. Mevcut çalışmada modellenen Türkçe hata durumları sayısı artırılabilir veya varsayılan parametre değerleri ile oynanarak farklı ihtiyaçlara cevap verecek şekilde daha tutarlı hale getirilebilir.

Ayrıca bu çalışmada bahsedilen, yapılabilecek denilip de çalışmanın kapsamı içerisine dahil edilmeyen şeyler de ileriki çalışmalara dahil edilebilir. Mesela;

- Türkçe Q klavye dışındaki klavye türleri için de hata modelleme yapılması ve geliştirilen yöntemin onlar üzerinde çalışıyor olması sağlanabilir.

- Klavyelerde yazarken komşuluğa ek olarak Shift, Alt, Alt Gr ve Num Lock gibi tuşlara basılması veya basılı kalması durumları da göz önünde bulundurulabilir.

- Tarih ve telefon gibi veri tiplerinin hatalı yazım durumları veya çok farklı formatta yazılmış olmaları durumu bu çalışmaya dahil edilmemiştir. Özellikle telefon tipinden veri alanının Türkçe metinler ile bir alakası olmadığından bu çalışmada yer bulmamıştır. Fakat veritabanlarındaki tekrarlı kayıtlar aranırken bu tiplerden alanlara göre de sorgulamalar yapılabileceğinden sonraki çalışmalara tarih ve telefon tipinden veri alanları dahil edilebilir.

- Benzerlik hesaplama işlemlerinin ve tekrarlı kayıt bulma işlemlerinin performansı artırılabilir ve eşleme süresi ve ilgili veritabanına olan yükü azaltılması yönünde iyileştirmeler yapılabilir.

- Bahsedilen hata durumları puanlanabilir, öncelikler belirlenebilir ve bu öncelikler ve puanlamalar dinamik olabilir. Hatta bu puanlama ve öncelik belirleme kısmı öğrenen bir yapıda olabilir; yani Makine Öğrenmesi çalışmalarından faydalanılabilir. Belirtilen hata durumları için istatistik tutularak puanlamanın kalitelileştirilmesi sağlanabilir.

- Çalışmada modellenen 4 bilgi tipinden birisi olan uzun metin verileri için Türkçe word-gram (kelimelerin peşpeşe gelme olasılıkları) lardan faydalanılabilir.

- Bu çalışmada geliştirilen yöntem bir veritabanı uygulaması olmasına rağmen bir sözlük kullanmamaktadır. Değişik türden Türkçe sözlükler de kullanılarak sözlük destekli bir hale getirilebilir.

- Bu tip bir çalışmada Türkçe harf, bigram ve trigram istatistiklerinden faydalanılarak matematiksel bir temele de dayandırılabilir (Diri ve Karslıgil, 2001).

- Yine bu çalışma Türkçe metinler için yapılmış bir çalışma olduğundan Türkçe dilbilgisi kurallarından da faydalanılarak sistemin başarısı artırılabilir.

- Son olarak bu çalışma Doğal Dil İşleme kavramlarına dayalı olmayan, yani anlamdan çok yazıma dayalı bazı istatistiksel çalışmalara dayanmaktadır. Bu nedenle bu konuda yapılabilecek daha ileri ve daha güzel bir çalışma da bu çalışmanın Türkçe Doğal Dil İşleme çalışmaları ile birleştirilmesi olabilir. Ayrıca yapılan çalışma Türkçe Dili üzerine yapılan başarılı çalışmalara örnek olarak gösterilebilecek Zemberek ve Pardus gibi program ve işletim sistemi projelerine de dahil edilebilir.

5. SONUÇLAR VE DEĞERLENDİRMELER

Sonuç olarak bu çalışmada metin eşleme ve metin benzerlik kavramlarının ne olduğu, metin eşleme ve metin benzerlik hesaplama yöntemlerine hangi alanlarda ve hangi durumlarda ihtiyaç duyuldukları araştırılmıştır. Daha sonra genel metin eşleme teknikleri ve algoritmaları incelenmiş ve bunların Türkçe metinler üzerindeki başarısı araştırılmıştır. Çalışmada daha sonra Türkçe metinler için yapılmış Zemberek gibi örnek çalışmalar incelenmiştir.

Bütün bu araştırmalar yapıldıktan sonra Türkçe metinlerin eşlenmesi ve benzerliklerinin hesaplanmasında karşılaşılan problemler ve hata durumları gözlemlenmiş ve yaklaşık on bir tane hata durumu modellenerek Türkçe metinlerin karşılaştırılmasında ve benzerliklerinin hesaplanmasında kullanılabilecek yeni bir yöntem geliştirilmiştir. Geliştirilen bu yöntem temelde kelimelerin anlamları ile ilgilenmek yerine özellikle muhtemel hatalı yazım veya farklı yazım formatlarından kaynaklanan hata durumlarını giderdikten sonra karşılaştırma yapan veya benzerlik hesaplayan bir yöntemdir.

Geliştirilen bu yöntemi kullanan, metin eşleme çalışmalarının kullanıldığı bir alan olan veritabanlarında tekrarlı veya benzer kayıtların bulunmasını sağlamak için Oracle veritabanı üzerinde çalışacak ve daha sonraki benzer çalışmalara da temel teşkil edebilecek bir uygulama programı yazıldı.

Türkçe metinlerin benzerliğinin daha tutarlı bir şekilde hesaplanması için geliştirilen yöntemin başarısını ölçmek için farklı seviyelerde ve farklı özelliklere sahip bilgisayar kullanıcılarından farklı şekillerde metin girilmesi istendi. Girilen metinler arasından hatalı olanlar seçilerek yapılan çalışma için test verisi olarak kullanıldı. Daha sonra bu veriler üzerinde birisi kendi geliştirdiğimiz yöntem, diğer ikisi de önceki bölümlerde anlatılan Levenshtein Edit Distance Benzerliği ve Jaro-Winkler Benzerliği yöntemleri olmak üzere 3 farklı yönteme göre metin benzerlik hesaplaması yapılarak, bu veriler üzerinde 3 yöntemin başarısı karşılaştırıldı.

Test verisi girmek üzere 5 farklı gruptan kişiler aşağıdaki gibi seçildi;

I. Genellikle İngilizce Q klavye kullanmaya alışmış olan, İngilizce'yi iyi bilen bilgisayar mühendisi–yazılımcı
II. Genellikle Türkçe Q klavye kullanmaya alışmış olan, İngilizce'yi iyi bilen bilgisayar mühendisi-yazılımcı
III. Orta seviyede İngilizce bilen ve yazabilen makine mühendisi
IV. Ortaokul mezunu az tecrübeli bilgisayar kullanıcısı

V. İngilizceyi çok iyi bilen ve sürekli olarak İngilizce metinler yazmaya alışmış mütercim

Yukarda verilen özelliklerdeki kişiler test verisi olarak veri girerken aşağıda listelenen farklı yollardan birisi veya bir kaçını kullanarak istenilen verileri girdiler. Bu giriş yöntemleri şöyledir;

I. Bir kişinin sessiz ortamda söylediği metinleri yazma
II. Bir kişinin gürültülü ortamda söylediklerini yazma
III. Eldeki bir kitaptan okuduğu metinleri yazma
IV. Bilgisayarda dijital ortamda bulunan bir kaynaktaki metinleri okuyup başka bir dosyaya yazma
V. Ezberden aklına gelen metinleri yazma

Yapılan bu testler esnasında farklı kullanıcıların farklı metin giriş yöntemlerini kullandıklarında genellikle farklı türden hatalar yapmaya meyilli oldukları gözlemlenmiştir;

- İngilizce mütercim olan kişinin genellikle Türkçe karakterler yerine bunların İngilizcelerini yazma eğiliminde olduğu
- İngilizce klavye kullanmaya alışmış olan bilgisayar mühendisinin İngilizce mütercime benzer bir şekilde Türkçe karakter kullanmamaya çalıştığı, bu yüzden hatalı metinler yazdığı
- Türkçe klavye kullanan bilgisayar mühendisinin 10 parmak kullanarak hızlı yazmaya çalıştığından, genellikle harflerin klavyedeki dizilişinden kaynaklanan hatalı yazımlar yaptığı
- Makine mühendisinin ve ortaokul mezunu kullanıcıların ise hata çeşitlerinin diğer kullanıcılara göre daha fazla olduğu gözlemlenmiştir.

Yukarda bahsedilen kullanıcıların hatalı olarak girdikleri 344 adet metin ve bu metinlerin doğru halleri "Edit Distance Similarity", "Jaro-Winkler Similarity" ve bizim tarafımızdan geliştirilen yöntem ile benzerlikleri hesaplanarak geliştirilen yöntemin diğer iki yönteme göre başarısı araştırıldı. Üç yöntemden hangisinin belirtilen hatalı yazımlı kelime çiftleri için en yüksek benzerlik oranı verdiği test edildi. Buna göre Tablo 5.1'deki gibi sonuçlar elde edildi;

Tablo 5.1 Kısa metinler üzerinde yöntemlerin başarılarının karşılaştırılması

Kelime Çifti Sayısı	En Yüksek Başarılı Yöntem(ler)	İkinci En Başarılı Yöntem(ler)	Başarısı En Düşük Yöntem(ler)
226	Geliştirilen Yöntem	Jaro-Winkler Benz.	Edit Distance Benz.
9	Geliştirilen Yöntem	Edit Distance Benz.	Jaro-Winkler Benz.
7	Geliştirilen Yöntem	Edit Distance Benz. = Jaro-Winkler Benz.	
77	Jaro-Winkler Benz.	Geliştirilen Yöntem	Edit Distance Benz.
25	Jaro-Winkler Benz.	Geliştirilen Yöntem = Edit Distance Benz.	

Tablo 5.1'de aynı hücrede birden çok yöntemin adının bulunması, hücrede adı geçen yöntemlerin başarılarının, o satır için aynı olması demektir. Yani 3. satırın anlamı; 7 metin çifti için en başarılı yöntem Geliştirilen Yöntemdir, bu metin çiftleri için diğer iki yöntem olan Jaro-Winkler Benzerliği ve Edit Distance Benzerliği yöntemlerinin hesapladıkları benzerlik oranları birbirlerine eşittir, demektir.

Tablo 5.2 'de ise 344 tane hatalı yazılmış metin çiftleri için her üç yöntem ile hesaplanan ortalama benzerlik yüzdeleri verilmiştir.

Tablo 5.2 Kısa metinler üzerinde yöntemlerin hesapladıkları benzerlik puanlarının ortalaması

Yöntem	Ortalama Benzerlik Yüzdesi
Geliştirilen Yöntem	95.280
Jaro-Winkler Benzerliği	93.577
Edit Distance Benzerliği	83.931

Geliştirilen yöntemin uzun metinler üzerindeki başarısını ölçmek için şöyle bir yol izlendi; Türkçe yazılmış bir tez dökümanının farklı uzunluklardaki 110 paragrafı, her bir paragraf bir uzun metin olmak üzere alındı. Bu şekilde elde edilen paragrafların en kısası 10 kelimeden en

uzunu ise 228 kelimeden oluşmaktadır. Sonra bu paragraflarda bulunan kelimelerden bazıları değiştirilerek, bu paragrafların hatalı yazılmış halleri elde edildi. Bu şekilde elde edilen doğru yazılmış uzun metinler ve bunların değiştirilmesiyle elde edilen hatalı hallerinin, kısa metinlerde olduğu gibi bahsedilen 3 yöntem kullanılarak benzerlikleri hesaplanarak Tablo 5.3 ve Tablo 5.4'teki sonuçlar elde edildi.

Tablo 5.3 Uzun metinler üzerinde yöntemlerin başarılarının karşılaştırılması

Paragraf Sayısı	**En Yüksek Başarılı Yöntem(ler)**	**İkinci En Başarılı Yöntem(ler)**	**Başarısı En Düşük Yöntem(ler)**
12	Geliştirilen Yöntem	Jaro-Winkler Benz.	Edit Distance Benz.
21	Geliştirilen Yöntem	Edit Distance Benz.	Jaro-Winkler Benz.
39	Geliştirilen Yöntem = Edit Distance Benz.	Jaro-Winkler Benz.	
3	Geliştirilen Yöntem	Edit Distance Benz. = Jaro-Winkler Benz.	
1	Jaro-Winkler Benz.	Geliştirilen Yöntem	Edit Distance Benz.
1	Edit Distance Benz.	Geliştirilen Yöntem = Jaro-Winkler Benz.	
11	Edit Distance Benz.	Jaro-Winkler Benz.	Geliştirilen Yöntem
21	Edit Distance Benz.	Geliştirilen Yöntem	Jaro-Winkler Benz.

Tablo 5.4 Uzun metinler üzerinde yöntemlerin hesapladıkları benzerlik puanlarının ortalaması

Yöntem	**Ortalama Benzerlik Yüzdesi**
Geliştirilen Yöntem	98.554
Edit Distance Benzerliği	97.027
Jaro-Winkler Benzerliği	94.192

Sonuç olarak, Tablo 5.1, Tablo 5.2, Tablo 5.3 ve Tablo 5.4'deki değerlere bakıldığında genel olarak yapılan çalışmada geliştirilen yöntemin bahsedilen test verileri için Jaro-Winkler Benzerliği ve Edit Distance Benzerliği yöntemlerine göre daha başarılı olduğunu söylenebilir. Bu çalışmanın bu konuda veya benzer konularda yapılacak çalışmalar için iyi bir prototip çalışma olduğunu söyleyebiliriz.

KAYNAKLAR

Amasyalı M.F., Diri B., "Bir Soru Cevaplama Sistemi: Baybilmiş", Türkiye Bilişim Vakfı Bilgisayar Bilimleri ve Mühendisliği Dergisi, sayfa:37-51, 2005/1

Austin D. Using Oracle 8, Que Press, USA, 1998

Bilenko M., Mooney R.J., "Adaptive Duplicate Detection Using Learnable String Similarity Measures", "Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2003)", Washington DC, 2003

Cebiroğlu G. "Türkçe'nin Bağlılık Araştırması", İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Doktora Tezi, İstanbul, 2006

Cohen W. W., Ravikumar P., and Fienberg S. E., A comparison of string distance metrics for name-matching tasks. In Proceedings of the IJCAI, 2003

Diri B., Karslıgil M.Y., "Design of a Genetic Algorithm for Constructing the Mathematical Model of Frequency Distribution of Words in Turkish Documents", 2001 ISCIS-The Sixteenth International Symposium on Computer and Information Sciences, Antalya, Turkey, 2001

Durgar I, El-Kahlout, Oflazer K, "Use of WordNet for Retrieving Words from Their Meanings in Proceedings of the Global WordNet Conference", Masaryk, Czech Republic, 2004

Haynes S.K, "Word Sense Disambiguation Using A Biased Corpus Of Wordnet Examples", Doctor of Philosophy in Computer Science , PHd Thesis, Illinois Institute of Technology, Chicago, Illinois, 2002

Jin L., Li C., "Selectivity Estimation for Fuzzy String Predicates in Large Data Sets", 31st VLDB Conference, Trondheim, Norway, 2005

Kardeş O, Güngör T, Kipman E, "Bir Uygulama Alanında Türkçe Metnin Anlambilimsel Gösterimi", Boğaziçi Üniversitesi Bilgisayar Mühendisliği Bölümü Yüksek Lisans Tezi, İstanbul, 2003

Külekçi M. O, "Türkçe Hasta Hikayelerinden Anahtar Kelime Analiziyle Akıllı Teşhis Sistemi", Unpublished paper,2000

Liu C.L, Jaeger S, Nakagawa M, "Online Recognition of Chinese Characters: The State-of-the-Art", IEEE Transactions On Pattern Analysis And Machine Intelligence, Vol. 26, No. 2, pg 198-213, 2004

Nivre J, "Sparse Data and Smoothing in Statistical Part-of-Speech Tagging", Journal of Quantitative 2000, Vol. 7, No. 1, sayfa 1–17, Sweden, 2000

Shieber S M, Nelken R, "Abbreviated Text Input Using Language Modeling", Natural Language Engineering 1 (1): 1–11, Cambridge University Press, United Kingdom, 2005

Yanıkoğlu B, Kholmatov A, "Türkçe İçin Geniş Sözcük Dağarcıklı Döküman Tanıma Sistemi", SİU, 2003

İNTERNET KAYNAKLARI

[1]. http://tspell.sourceforge.net

[2]. http://www.levenshtein.net

[3]. http://www.pardus.org.tr

[4]. http://www.tdk.gov.tr

[5]. https://zemberek.dev.java.net

EKLER

Ek-1 Türkçe Hata Durumları Tablosu

TÜRKÇE HATA DURUMLARI TABLOSU		
ID	**DURUM_AD**	**DURUM_ACIKLAMA**
1	Türkçe Karakter	ı,ç,ö,ğ,ü,ş ve i,c,o,g,u,s karakterlerinin birbirlerinin yerine kullanılması
2	D_T Farklılaşması	d ve t harflerinin birbirlerinin yerine kullanılması
3	B_P Farklılaşması	b ve p harflerinin birbirlerinin yerine
4	Dublicate Karakterler	Aynı harfin peşpeşe ikilemesi durumu
5	Kısaltmalar	YTÜ - Yıldız Teknik Üniversitesi gibi
6	Klavye bitişiklik	Bir karaktere basarken genellikle bir de fazladan yakınındaki bir karaktere daha basılması durumu
7	Klavye yakınlık	Bir karaktere basacakken yanlışlıkla klavyede komşu bir karaktere basılması durumu
8	Tuvar_hane	İçerisinde tuvar, tuar, hane ifadelerini bulundurduğu durumlar
9	Ara sesli	Metinin ilk hecesinde ı,i,u harflerinin bulunması durumu
10	Bitişiklik yer değişimi	Bitişik harflerin yer değiştirmesi durumu

Ek-2 Türkçe Karakterler Tablosu

TÜRKÇE KARAKTERLER TABLOSU		
ID	**KARAKTER**	**ESLENIK_KARAKTER**
1	ı	i
2	ü	u
3	ö	o
4	ğ	g
5	ş	s
6	ç	c

Ek-3 Türk Alfabesindeki Harfler Tablosu

TÜRK ALFABESİNDEKİ HARFLER TABLOSU	
HARF	**FREKANS**
A	11,68
B	2,95
C	0,97
Ç	1,26
D	4,87
E	9,01
F	0,44
G	1,34
Ğ	1,13
H	1,14
I	5,20
İ	8,27
J	0,01
K	4,71
L	5,75
M	3,74
N	7,23
O	2,45
Ö	0,87
P	0,79
R	6,95
S	2,95
Ş	1,94
T	3,09
U	3,43
Ü	1,99
V	0,98
Y	3,37
Z	1,50

Ek-4 Alan Bilgi Tipleri Tablosu

ALAN BİLGİ TİPLERİ TABLOSU			
ID	**TIP**	**MIN_BENZERLIK**	**ILK_HARF_TUTMALI**
1	Adres	90	0
2	Ad	90	1
3	Kısa Metin	90	1
4	Uzun Metin	80	0

Ek-5 Klavye Türleri Tablosu

KLAVYE TÜRLERİ TABLOSU		
ID	**KLAVYE_TUR_ADI**	**ACIKLAMA**
1	Türkçe Q – Dizüstü Bilgisayar	Dizüstü bilgisayarlar için Q klavye
2	Türkçe Q – Standart	Masaüstü bilgisayar için standart Q klavye
3	Türkçe F – Dizüstü Bilgisayar	Dizüstü bilgisayarlar için F klavye
3	Türkçe F – Standart	Masaüstü bilgisayar için standart F klavye
4	İngilizce	İngilizce klavye

Ek-6 Alan Bilgi Tip Hata Durum Tablosu

ALAN BİLGİ TİP HATA DURUM TABLOSU			
ID	**ALAN_BILGI_TIP_ID**	**HATA_DURUM_ID**	**ALAN_BILGI_TIP**
1	1	1	Adres
2	1	2	Adres
3	1	3	Adres
4	1	4	Adres
5	1	5	Adres
6	1	6	Adres
7	1	7	Adres
8	1	8	Adres
9	1	9	Adres
10	1	10	Adres
11	2	1	Ad
12	2	2	Ad
13	2	3	Ad
14	2	4	Ad
15	2	5	Ad
16	2	6	Ad
17	2	7	Ad
18	2	8	Ad
19	2	9	Ad
20	2	10	Ad
21	3	1	Kısa Metin
22	3	2	Kısa Metin
23	3	3	Kısa Metin
24	3	4	Kısa Metin
25	3	5	Kısa Metin
26	3	6	Kısa Metin
27	3	7	Kısa Metin

28	3	8	Kısa Metin
29	3	9	Kısa Metin
30	3	10	Kısa Metin
31	4	1	Uzun Metin
32	4	2	Uzun Metin
33	4	3	Uzun Metin
34	4	4	Uzun Metin
35	4	5	Uzun Metin
36	4	6	Uzun Metin
37	4	7	Uzun Metin
38	4	8	Uzun Metin
39	4	9	Uzun Metin
40	4	10	Uzun Metin

Ek-7 Klavye Karakter Komşuluğu Tablosu

KLAVYE KARAKTER KOMŞULUĞU TABLOSU				
ID	**KLAVYE_TUR_ID**	**KARAKTER1**	**KARAKTER2**	**KLAVYE_TUR**
1	1	a	q	Türkçe Q – Dizüstü
2	1	a	s	Türkçe Q – Dizüstü
3	1	a	z	Türkçe Q – Dizüstü
4	1	b	n	Türkçe Q – Dizüstü
5	1	b	h	Türkçe Q – Dizüstü
6	1	b	g	Türkçe Q – Dizüstü
7	1	b	v	Türkçe Q – Dizüstü
8	1	c	x	Türkçe Q – Dizüstü
9	1	c	d	Türkçe Q – Dizüstü
10	1	c	f	Türkçe Q – Dizüstü
11	1	c	v	Türkçe Q – Dizüstü
12	1	ç	ö	Türkçe Q – Dizüstü
13	1	d	e	Türkçe Q – Dizüstü
14	1	d	s	Türkçe Q – Dizüstü
15	1	d	f	Türkçe Q – Dizüstü
16	1	d	x	Türkçe Q – Dizüstü
17	1	d	c	Türkçe Q – Dizüstü
18	1	e	w	Türkçe Q – Dizüstü
19	1	e	r	Türkçe Q – Dizüstü
20	1	e	s	Türkçe Q – Dizüstü
21	1	e	3	Türkçe Q – Dizüstü
22	1	e	4	Türkçe Q – Dizüstü
23	1	e	d	Türkçe Q – Dizüstü
24	1	f	d	Türkçe Q – Dizüstü
25	1	f	r	Türkçe Q – Dizüstü
26	1	f	t	Türkçe Q – Dizüstü
27	1	f	g	Türkçe Q – Dizüstü
28	1	f	v	Türkçe Q – Dizüstü

29	1	f	c	Türkçe Q – Dizüstü
30	1	g	h	Türkçe Q – Dizüstü
31	1	g	v	Türkçe Q – Dizüstü
32	1	g	b	Türkçe Q – Dizüstü
33	1	g	t	Türkçe Q – Dizüstü
34	1	g	y	Türkçe Q – Dizüstü
35	1	h	g	Türkçe Q – Dizüstü
36	1	h	j	Türkçe Q – Dizüstü
37	1	h	b	Türkçe Q – Dizüstü
38	1	h	n	Türkçe Q – Dizüstü
39	1	h	y	Türkçe Q – Dizüstü
40	1	h	u	Türkçe Q – Dizüstü
41	1	h	t	Türkçe Q – Dizüstü
42	1	ı	k	Türkçe Q – Dizüstü
43	1	ı	j	Türkçe Q – Dizüstü
44	1	ı	u	Türkçe Q – Dizüstü
45	1	ı	9	Türkçe Q – Dizüstü
46	1	ı	8	Türkçe Q – Dizüstü
47	1	i	ş	Türkçe Q – Dizüstü
48	1	i	ğ	Türkçe Q – Dizüstü
49	1	i	ü	Türkçe Q – Dizüstü
50	1	i	,	Türkçe Q – Dizüstü
51	1	i	.	Türkçe Q – Dizüstü
52	1	i	ç	Türkçe Q – Dizüstü
53	1	j	h	Türkçe Q – Dizüstü
54	1	j	k	Türkçe Q – Dizüstü
55	1	j	y	Türkçe Q – Dizüstü
56	1	j	u	Türkçe Q – Dizüstü
57	1	j	n	Türkçe Q – Dizüstü
58	1	j	ı	Türkçe Q – Dizüstü
59	1	j	m	Türkçe Q – Dizüstü
60	1	j	h	Türkçe Q – Dizüstü
61	1	k	j	Türkçe Q – Dizüstü

62	1	k	l	Türkçe Q – Dizüstü
63	1	k	u	Türkçe Q – Dizüstü
64	1	k	ı	Türkçe Q – Dizüstü
65	1	k	m	Türkçe Q – Dizüstü
66	1	k	ö	Türkçe Q – Dizüstü
67	1	k	ç	Türkçe Q – Dizüstü
68	1	k	o	Türkçe Q – Dizüstü
69	1	l	k	Türkçe Q – Dizüstü
70	1	l	ş	Türkçe Q – Dizüstü
71	1	l	ı	Türkçe Q – Dizüstü
72	1	l	o	Türkçe Q – Dizüstü
73	1	l	ö	Türkçe Q – Dizüstü
74	1	l	ç	Türkçe Q – Dizüstü
75	1	l	p	Türkçe Q – Dizüstü
76	1	m	n	Türkçe Q – Dizüstü
77	1	m	ö	Türkçe Q – Dizüstü
78	1	m	j	Türkçe Q – Dizüstü
79	1	m	k	Türkçe Q – Dizüstü
80	1	n	m	Türkçe Q – Dizüstü
81	1	n	b	Türkçe Q – Dizüstü
82	1	n	h	Türkçe Q – Dizüstü
83	1	n	j	Türkçe Q – Dizüstü
84	1	o	9	Türkçe Q – Dizüstü
85	1	o	0	Türkçe Q – Dizüstü
86	1	o	ı	Türkçe Q – Dizüstü
87	1	o	ş	Türkçe Q – Dizüstü
88	1	o	l	Türkçe Q – Dizüstü
89	1	o	k	Türkçe Q – Dizüstü
90	1	o	p	Türkçe Q – Dizüstü
91	1	ö	ç	Türkçe Q – Dizüstü
92	1	p	ğ	Türkçe Q – Dizüstü
93	1	p	*	Türkçe Q – Dizüstü
94	1	p	i	Türkçe Q – Dizüstü

95	1	p	ş	Türkçe Q – Düzüstü
96	1	p	l	Türkçe Q – Düzüstü
97	1	p	o	Türkçe Q – Düzüstü
98	1	r	e	Türkçe Q – Düzüstü
99	1	r	t	Türkçe Q – Düzüstü
100	1	r	4	Türkçe Q – Düzüstü
101	1	r	d	Türkçe Q – Düzüstü
102	1	r	f	Türkçe Q – Düzüstü
103	1	r	g	Türkçe Q – Düzüstü
104	1	r	5	Türkçe Q – Düzüstü
105	1	s	d	Türkçe Q – Düzüstü
106	1	s	q	Türkçe Q – Düzüstü
107	1	s	w	Türkçe Q – Düzüstü
108	1	s	z	Türkçe Q – Düzüstü
109	1	s	x	Türkçe Q – Düzüstü
110	1	s	e	Türkçe Q – Düzüstü
111	1	s	a	Türkçe Q – Düzüstü
112	1	ş	i	Türkçe Q – Düzüstü
113	1	ş	I	Türkçe Q – Düzüstü
114	1	ş	o	Türkçe Q – Düzüstü
115	1	ş	ç	Türkçe Q – Düzüstü
116	1	ş	ğ	Türkçe Q – Düzüstü
117	1	ş	p	Türkçe Q – Düzüstü
118	1	ş	.	Türkçe Q – Düzüstü
119	1	t	6	Türkçe Q – Düzüstü
120	1	t	r	Türkçe Q – Düzüstü
121	1	t	g	Türkçe Q – Düzüstü
122	1	t	f	Türkçe Q – Düzüstü
123	1	t	y	Türkçe Q – Düzüstü
124	1	t	5	Türkçe Q – Düzüstü
125	1	u	ı	Türkçe Q – Düzüstü
126	1	u	y	Türkçe Q – Düzüstü
127	1	u	k	Türkçe Q – Düzüstü

128	1	u	j	Türkçe Q – Dizüstü
129	1	u	h	Türkçe Q – Dizüstü
130	1	u	8	Türkçe Q – Dizüstü
131	1	u	7	Türkçe Q – Dizüstü
132	1	ü	-	Türkçe Q – Dizüstü
133	1	ü	,	Türkçe Q – Dizüstü
134	1	ü	i	Türkçe Q – Dizüstü
135	1	ü	ğ	Türkçe Q – Dizüstü
136	1	v	c	Türkçe Q – Dizüstü
137	1	v	g	Türkçe Q – Dizüstü
138	1	v	f	Türkçe Q – Dizüstü
139	1	v	b	Türkçe Q – Dizüstü
140	1	y	6	Türkçe Q – Dizüstü
141	1	y	7	Türkçe Q – Dizüstü
142	1	y	u	Türkçe Q – Dizüstü
143	1	y	j	Türkçe Q – Dizüstü
144	1	y	h	Türkçe Q – Dizüstü
145	1	y	g	Türkçe Q – Dizüstü
146	1	y	t	Türkçe Q – Dizüstü
147	1	z	x	Türkçe Q – Dizüstü
148	1	z	s	Türkçe Q – Dizüstü
149	1	ö	m	Türkçe Q – Dizüstü
150	1	m	ö	Türkçe Q – Dizüstü
151	1	ö	k	Türkçe Q – Dizüstü
152	1	k	ö	Türkçe Q – Dizüstü
153	1	l	ö	Türkçe Q – Dizüstü
154	1	ö	l	Türkçe Q – Dizüstü

ÖZGEÇMİŞ

Doğum tarihi	10.08.1980	
Doğum yeri	Erzurum	
Lise	1995-1998	Erzurum Fen Lisesi
Lisans	1998-2002	İstanbul Teknik Üniversitesi Elektrik Elektronik Fak. Bilgisayar Mühendisliği Bölümü
Yüksek Lisans	2002-	Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Bölümü

İş Tecrübesi

2002-2003	SAP-Abap Programcısı
2003-2008	Veritabanı Programcısı - Oracle Danışmanı


00325034