METİN DOSYALARININ BİRLEŞTİRİLMESİNDE YAKINLIK ÖLÇÜTLERİNİN KULLANILMASI

ALPEREN ŞAHİN

YÜKSEK LİSANS TEZİ

BİLGİSAYAR MÜHENDİSLİĞİ

TOBB EKONOMİ VE TEKNOLOJİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

AĞUSTOS 2015

ANKARA

Fen Bilimleri Enstitü onayı

_____

Prof. Dr. Osman EROĞUL

Müdür

Bu tezin Yüksek Lisans derecesinin tüm gereksinimlerini sağladığını onaylarım.

_____

Doç. Dr. Erdoğan DOĞDU

Anabilim Dalı Başkanı

ALPEREN ŞAHİN tarafından hazırlanan METİN DOSYALARININ BİRLEŞTİRİLMESİNDE YAKINLIK ÖLÇÜTLERİNİN KULLANILMASI adlı bu tezin Yüksek Lisans tezi olarak uygun olduğunu onaylarım.

_____

Yrd. Doç. Dr. Hüsrev Taha SENCAR

Tez Danışmanı

Tez Jüri Üyeleri

Başkan :Prof. Dr. Ali Aydın SELÇUK _____

Üye : Yrd. Doç. Dr. Hüsrev Taha SENCAR _____

Üye : Yrd. Doç. Dr. Aybar ACAR _____

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, ayrıca tez yazım kurallarına uygun olarak hazırlanan bu çalışmada orijinal olmayan her türlü kaynağa eksiksiz atıf yapıldığını bildiririm.

Alperen ŞAHİN

Üniversitesi : **TOBB Ekonomi ve Teknoloji Üniversitesi**
Enstitüsü : **Fen Bilimleri**
Anabilim Dalı : **Bilgisayar Mühendisliği**
Tez Danışmanı : **Yrd. Doç. Dr. Hüsrev Taha Sencar**
Tez Türü ve Tarihi : **Yüksek Lisans – Ağustos 2015**

Alperen ŞAHİN

METİN DOSYALARININ BİRLEŞTİRİLMESİNDE YAKINLIK ÖLÇÜTLERİNİN KULLANILMASI

ÖZET

Parçalanmış dosyaların kurtarılması öncelikli olarak parçalanmış dosyalar arasındaki yakınlığın doğru bir şekilde değerlendirilmesine dayanır. Metin tabanlı dosyalar veriyi oldukça zayıf bir yapıda tuttuğu için parçaların birleştirilmesi işi zorlu bir iştir. Bu çalışmada, metin dosyalarının birleştirilmesinde kullanılan mevcut yakınlık ölçütlerini değerlendirdik. Aldığımız sonuçlara göre mevcut metotların her birinin de tek başına değerlendirildiğinde hedeflenen noktadan uzak olduğu görüldü. Daha sonra PTCR isimli yeni metodu tanıttık. Bu metot metin dosyaları içerisinde gerek dosya tanımı, gerek verilerin sunulması gerekse de verilerin işlenmesinde kullanılan oldukça sınırlı karakteristik yapılardan yakınlık değerleri çıkarmayı hedefleyen bir metottur. Yaklaşımımız, daha verimli yakınlık değerlendirmeleri elde etmek için dosya içerisindeki dosyaya özel olan etiket-kelimelerin sıralamaları üzerine istatiksel bir model kurmaktadır. Sonuçlara göre birleştirme performansı PTCR metodunun da katkısıyla dikkate değer bir iyileşme göstermiştir.

Anahtar Kelimeler: Dosya kurtarma, Metin dosyaları, Dosya parçalanması, Dosya birleştirme

University : **TOBB Economics and Technology University**
Institute : **Institute of Natural and Applied Sciences**
Science Programme : **Computer Engineering**
Supervisor : **Asst. Prof. Dr. Hüsrev Taha SENCAR**
Degree Awarded and Date : **M.Sc. – August 2015**

Alperen ŞAHİN

A STUDY ON ADJAJENCY MEASURES FOR REASSEMBLING TEXT FILES

ABSTRACT

Recovery of fragmented files relies on the ability to accurately evaluate the adjacency of two fragments. Text-based files typically organize data in a very weakly structured manner; therefore, fragment reassembly remains a challenging task. In this work, we evaluate existing adjacency measures that can be used for assembling fragmented test files. Our results show that individual performances of existing measures are far from adequately addressing this need. We then introduce a new approach that attempts to exploit the limited structural characteristics of text files which utilize constructs for description, presentation, and processing of file data. Our approach builds a statistical model of the ordering of file-type specific constructs and incorporates this information into adjacency measures for more reliable fragment reassembly. Results show that reassembly accuracy increases significantly with this approach.

Keywords: File carving, Text files, File fragmentation, File reassembly

TEŞEKKÜR

Öncelikli olarak danışman hocam Yrd. Doç. Dr. Hüsrev Taha Sencar'a değerli rehberliği ve önerileri, cesaretlendirmesi ve sabrı için en içten teşekkürlerimi ifade etmek isterim.

Ayrıca bu tezi okumalarındaki dikkatleri ve önerileri sebebiyle Prof. Dr. Ali Aydın Selçuk'a ve Yrd. Doç. Dr. Aybar Acar'a teşekkür ederim.

Ve eğitim hayatım boyunca eksilmeyen motivasyonları ve destekleri için aileme teşekkür ederim.

Son olarak, eşim Merve'ye çalışmalarımın devamında bana verdiği destekten dolayı teşekkür ederim.

İÇİNDEKİLER

ÇİZELGE LİSTESİ

Çizelge **Sayfa**

ŞEKİL LİSTESİ

Şekil **Sayfa**

1 GİRİŞ

Dijital cihazlar gün geçtikçe adli soruşturmalara daha çok konu olur hale gelmiştir. Bunun bir sonucu olarak, bu cihazlarda bulunan potansiyel dijital delilleri alabilmek ve analiz edebilmek bir gereklilik halini almıştır. Bu iş ile ilişkili en önemli sorunlardan biri cihazda bulunan dosya sistemi ulaşılamaz olduğunda cihazdaki dosyaların kurtarılması sorunudur. Çoğu durumda, bu bilgi ya dosyalar silindiği için ya da depolama birimi formatlandığı için kayıp olacaktır. Birçok diskte ve anlık bellek tabanlı depolama birimlerinde bu işlemler dosyaları tanımlayan ve depolama birimde nerede bulunduklarını gösteren dosya sistemi kayıtlarını silerler. Fakat ilgili dosyanın verileri, kaydedilen yeni bir dosya üzerine yazılmadığı sürece yerinde kalmaya devam edecektir.

Böyle durumlarda delilin alınabilmesi için ilgili dosyanın kurtarılması gerekir. Adli bilişimde dosya kurtarma terimi; bir sistemdeki dosyaların sistem meta-verileri olmadan dosya formatı karakterlerinin dikkate alınmasıyla geri getirilmesi işlemi için kullanılır.

Dosya kurtarma işlemleri basitçe dosya imzalarını tanımlama ile başlar ki bu dosya imzaları genel olarak üstbilgiler, altbilgiler ve bunun yanında dosyanın içerisindeki veriyi tanımlamak için kullanılan bütün dizgileri kapsar. Bu işlem sırasında depolama birimi baştan sona farklı dosya türlerine ait imzalar için sırasıyla taranır. Fakat bu aşamada iki farklı zorlukla karşılaşılır.

Bunlardan birincisi, depolanan dosyanın parçalanmış olma durumudur. Dosya parçalanmalarının yaygınlığı konusunda yapılan çalışmalar göstermiştir ki adli olaylarla ilgili dosyalar (birçok farklı metin dosyasını da içerir) çokça parçalamaya maruz kalır[5] ve bu dosyalardan en çok parçalananı ise yapılan çalışmalara göre bilgisayar günlükleridir[7]. Dosyanın parçalanması: dosyanın içeriğini tutan fiziksel depolama birimlerinin (ör. gruplar ya da bloklar) art arda gelmemesi ya da gerçek sıralarını kaybetmeleri ve fiziksel depolama birimi üzerinde rastgele yayılmaları

durumudur ki bu dağılan parçaların tespit edilip tekrar bir araya getirilmesine ihtiyaç vardır. Bu işlem ise, iki ayrık veri dizisinin birbirinin tamamlayıcısı olup olmadığını belirlemek için hızlı bir doğrulama yöntemi gerektirir.

Başa çıkılması gereken diğer problem ise kısmen üzerine yazılmış (ve bu yüzden bozulmuş) ve bir kısmı ya da büyük bir kısmı silinmiş olan dosyaların geri getirilmesidir. Bunun üstesinden gelinebilmesi için ise kısmi dosya verisinin algılanabilmesi gerekir ki bu, dosya parçalarının algılanmasından çok daha zor bir işlemdir.

Çoğu dosya kurtarma yöntemleri, kullanıcı cihazlarındaki yaygınlık sebebiyle kodlanmış ve sıkıştırılmış ikili dosyaları kurtarmaya odaklanmıştır. Bu dosyalar medya dosyaları, sıkıştırılmış arşiv dosyaları ya da yürütülebilir ikili dosyalar olabilir. Bu dosyalar çok sağlam yapılara sahip olduklarından dolayı birleştirmeden sonra doğruluğu otomatik bir şekilde kolaylıkla test edilebilir.

Fakat diğer taraftan, sistemlerde tutulan verilerin büyük bir kısmı; harfleri, rakamları ya da sembolleri temsil etmek için sayısal değerler halinde metin tabanlı formatlarda tutulur. (Burada şunu belirtmek gerekir ki metin dosyaları ile bütün insan-tarafından-okunabilir dosyalar ya da standart metin kodlama düzenlerinden biri tarafından oluşturulan dosyalar kastedilmektedir.) Bu tip dosyaların en açık örnekleri işlem kayıtları, düz metin dosyaları, e-posta kutuları, biçimlendirme dilleri (ör. HTML ve XML) , bazı veritabanları ve kaynak kodlarıdır. Metin dosyalarında ise diğerlerinde farklı olarak ayırt edici dosya yapıları olmak zorunda değildir. Bunun bir sonucu olarak, temelde dosya imzaları(üstbilgi, altbilgi ve diğer yapılar) kullanımına dayalı geleneksel dosya kurtarma yaklaşımları çoğunlukla etkisizdir.

Dahası, metin biçimli dosya parçaları kolayca tespit edilebilir ve anahtar kelime arama işlemi her dosya veri bloğunda bağımsız olarak yapılabilir. Bu nedenle, metin dosyaları kurtarma genellikle karmaşık olmayan bir görev olarak kabul edilir. Ancak, metin dosyalarının parçalanması göz önüne alınacak olursa, metin dosyası kurtarma işi çok zorlu bir probleme dönüşür. Mevcut yöntemle metin dosyasının birleştirmeden sonraki doğruluk testi ise mümkün değildir.

Metin dosyası parçalarını birleştirmede literatürde bugüne kadar tanımlanmış tek metot sonlu-düzen içeriğe dayalı modelleme tekniğidir. Bu teknikte ileride gelmesi muhtemel karakteri tahmin etmek için bir pencere parçalar üzerinde kaydırılır[10]. Bu ardışık olma olasılığı, her bir kayma pozisyonunda görülen n-gram frekanslarının çarpılarak birleştirilmesiyle elde edilmektedir. Bunun yanında, genel olarak metin parçacıklarının veya metin kümelerinin benzerliğini araştırmada kullanılan n-gram frekans profilleri de [3, 11], dosya kurtarma problemlerine uygulanabilirler.

Esasen, dosya parçaları yeniden birleştirme işlemi, birleştirilen verinin doğruluğunu kontrol etmek için otomatik bir teknik gerektirir. Metin dosyalarında ise sağlam bir dosya yapısının eksikliği sebebiyle, hangi parçanın nerede bulunması gerektiği konusunda çok az ipucu bulunabilir. Sonuç olarak, metin dosyaları için dosya birleştirme işlemleri büyük çoğunlukla manüel bir görev olarak kalmıştır.

Bu çalışmada; ilk olarak, parçalanmış metin dosyalarının kurtarılması probleminde uygulanan olası benzerlik ölçütlerinin doğruluğu ölçüldü. Daha sonra mevcut yöntemlerin dosya türlerine özel parametreleri tespit edildi. Bu işlemin ardından birleştirme performansını artırmak için metin tabanlı verilerin sahip oldukları yapıların kullanıldığı yeni metot tanıtıldı.

Metin dosyalarının sabit bir yapılarının olmamasına rağmen kesin olan karakteristik yapılara sahip oldukları söylenebilir. Bu noktada, daha çok işaretleme dilleri ve kaynak kodlara odaklanıldı. Çünkü bu diller, ön-tanımlı kelimeler kullanarak sahip olunan verinin hem gösteriliş şekillerini hem de verinin işleyiş şekillerini kontrol ederler. Önerilen yaklaşımda, sahip olunan bu ön-tanımlı kelimelerin birbirleri ile olan ilişkileri temel alınarak bir istatiksel model kullanılır ki bu model ile birlikte iki parçanın birbirini takip edip etmediği hakkında sayısal bir veri elde edilir. Böylece metin dosyalarının kurtarılması probleminde metin dosyalarının dosya türüne özel karakteristik yapılarını mevcut yöntemlerle birleştirerek daha otomatik dosya kurtarma teknikleri geliştirilebilir.

Alınan sonuçlar, birleştirme performanslarının önerilen yöntem ile birlikte yükseldiğini göstermiştir.

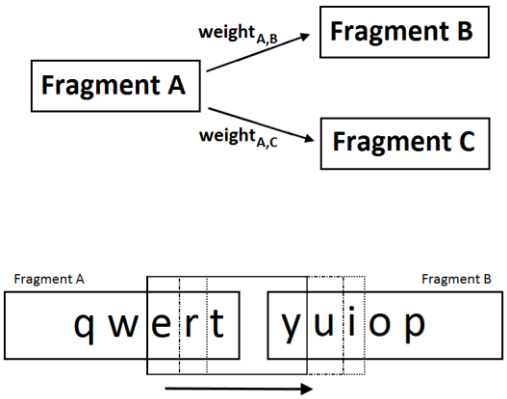
2 METİN DOSYALARININ YAKINLIĞININ ÖLÇÜLMESİNDE MEVCUT YÖNTEMLER

Literatür taraması yapıldığında, metin dosyalarının kurtarılması problemine üretilen tek çözümün bölüm 2.1'de anlatılan kısmi eşleştirme yoluyla tahmin yöntemi olduğu görülür. Fakat genellikle metin sınıflandırma işlemleri için kullanılan, bölüm 2.2'de anlatılan yöntemler de bu probleme uygulanabilirler. Bu bölümde, parçalanmış metin dosyalarının kurtarılması probleminde kullanılan/kullanılabilecek metin dosyalarının yakınlık ölçütlerine ve bu yöntemlerin farklı metin dosyası türleri üzerindeki performanslarına genel bir bakış yapacağız.

2.1 Kısmi Eşleştirme Yoluyla Tahmin Yöntemi

Kısmi eşleştirme yoluyla tahmin (prediction by partial matching - PPM), sonlu-düzen içeriğe dayalı bir modelleme tekniğidir ve kayıpsız veri sıkıştırma teknikleri için önemli bir ölçüttür. PPM' de temel fikir giriş iş akışından son birkaç karakter kullanılarak ardından gelebilecek karakterlerin tahmin edilmesidir [4]. PPM metodu $W_{A,B}$ ağırlık değerini parça A ile parça B arasındaki eşleşmenin başarısı olarak hesaplarken parça A'nın son baytları ile parça B'nin ilk baytları arasında kayan bir pencere ile hesaplar [8].

Şekil 2.1, kayan pencere boyutu *4* olduğu durumda iki parça arasındaki ağırlığın nasıl hesaplandığını göstermektedir. Pencere başlangıçta, parça B'nin başından bir karakter ve kalan kısmı parça A'dan olacak şekilde yerleştirilir. Daha sonra parça A'dan sadece bir karakter pencerenin içinde kalana kadar her seferinde bir karakter kaydırılır. Her bir kaydırmada, yeni bir n-gram kelime yaratılır (bu örnekte pencere boyutu 4 olduğu için 4-gram kelimeler çıkmaktadır.) ve yaratılan bu n-gram kelimenin frekansı daha önceki frekanslar ile çarpılarak $W_{A,B}$ ağırlık değerinin son hali hesaplanır.



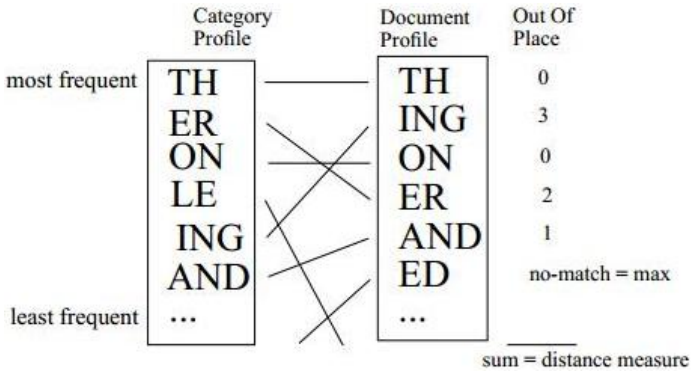
Şekil 2.1: Kayan pencere boyutu 4 olduğunda PPM ile ağırlık hesaplama

Genellikle, kendi kendine öğrenen algoritmaların kullanıldığı PPM modellerinde n-gram kelimelerin frekansları sadece o ana kadar işlenen karakter dizisi dikkate alınarak hesaplanır. İşlenen karakter dizisi yeterince uzun olduğunda avantaj sağlayan bu algoritmalar; uğraşılan veri kümesi sınırlı olduğunda, bu modeller ile elde edilen frekans değerleri güvenilir ve sağlam olmadıklarından pek işe yaramazlar. Bu sebeple önceden hesaplanmış n-gram frekanslarına ihtiyaç vardır.

2.2 N-gram İstatistiklerinin Kullanılmasıyla Yapılan Tahmin Yöntemleri

Metin dosyalarından elde edilen n-gram istatistiklerinin geniş bir yelpazedeki metin sınıflandırma işlerinde basit ve güvenilir bir yöntem olduğu ispatlanmıştır [4]. Bu sebeple, böyle bir yaklaşım birbirini takip eden metin parçalarının tespit edilmesi problemine de uygulanabilir. İki n-gram frekans profili arasındaki benzerlik ya da farklılığın ölçülmesi yersizlik ölçütü (out-of-place difference - OOP) ile mümkündür. Bunun için iki farklı metin parçasından elde edilen n-gram kelimeler ilk olarak kullanım sıklığına göre en çok kullanılan kelimeden en az kullanılana doğru sıralanır.

Daha sonra karşılaştırılan iki profil arasında benzer olan n-gram kelimelerin konumları arasındaki fark her bir n-gram kelimesi için toplanır. (İki profilde birden bulunmayan n-gram kelimeleri için ön tanımlı bir fark değeri eklenir) Şekil 2.2 bu işlemi anlatmaktadır.



Şekil 2.2: Cavnar & Trenkle (1994) tarafından tanıtılan yersizlik ölçütü ile iki n-gram frekans profilinin karşılaştırılması [3]

N-gram frekans profillerinin kullanılabileceği bir diğer benzerlik hesaplama yöntemi ise kosinüs benzerliği (cosine similarity - COS) yöntemidir. Bu durumda, n-gram profilleri en sık kullanılan k n-gram kelimesi içerir. Bu profiller, k-boyutlu uzayda bir vektör olarak gösterildiği zaman bu vektörler arasındaki açının kosinüs değeri bu profiller arasındaki benzerliği verir.

$$\begin{aligned} \cos(\theta) &= \frac{A \cdot B}{\|A\|\|B\|} \\ &= \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}} \end{aligned} \tag{2.1}$$

Birinci bölgede bulunan bu vektörlerin arasındaki açı (θ) 0 ile 90 derece arasında olacağından dolayı kosinüs değeri 0 ile 1 arasında olur. 1 değeri iki profilin tamamen aynı olduğunu gösterirken 0 değeri tamamen alakasız olduğunu gösterir.

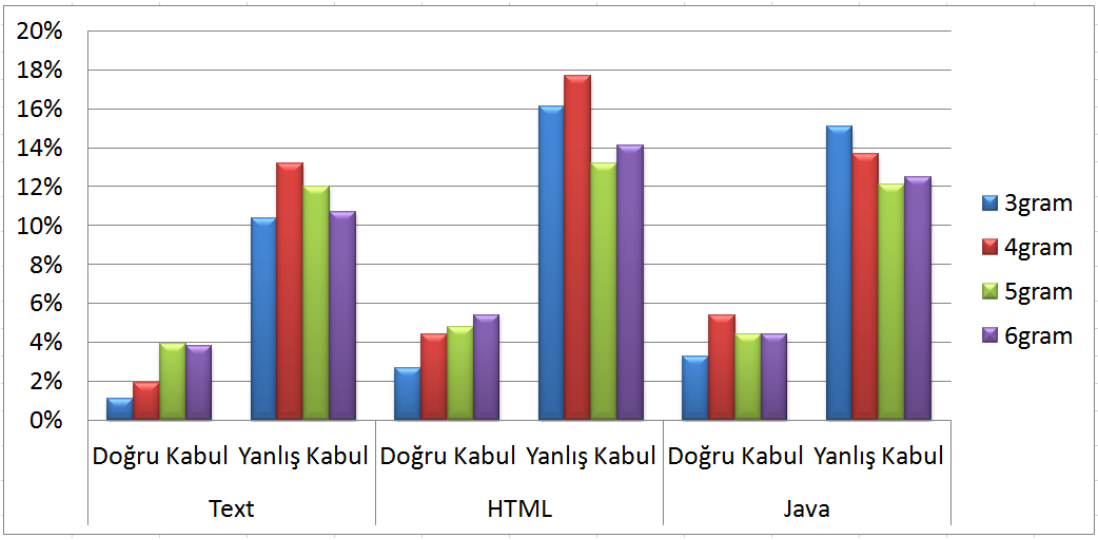
2.3 Performans Kıyaslamaları

Yapılan testler, bahsedilen metotların metin dosyaları birleştirmedeki başarılarını doğru bir şekilde ölçmek için geniş bir veri kümesi üzerinde yapıldı. Çalışılan veri kümesi içerinde çok sayıda yalın metin dosyaları, HTML dosyaları ve Java kaynak kodu dosyaları bulunmaktadır. Metin dosyaları parçaları *Project Gutenberg*' den rastgele elde edilen ve dünya klasiklerine ait *76MB* boyutundaki 100 e-kitaptan elde edilmiştir. HTML dosyaları için *510MB* boyutundaki 13.247 dosyadan oluşan ClueWeb09 [6] veri seti kullanıldı. Java kaynak kod dosyaları için ise açık kaynak kod paylaşım platformu olan *GitHUB* sisteminden elde edilen 42.435 Java dosyası kullanıldı. Bütün dosya türleri için dil; kullanım kolaylığı ve veri genişliği sebebiyle İngilizce olarak belirlendi.

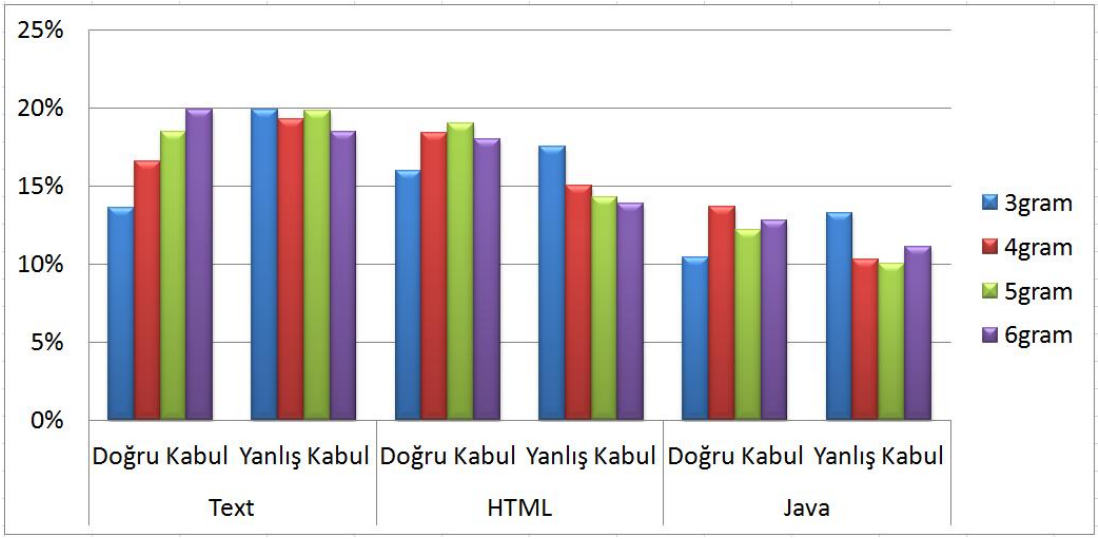
Her bir dosya türü için yapılan testlerde, bütün dosyalar önceden belirlenen miktarda baytlarda eşit olarak parçalandı. Her testte parçalardan oluşan bir havuz kuruldu ve bu havuza üretilen dosya parçaları rastgele olarak belli miktarda konuldu. Daha sonra birbirini gerçekten takip eden iki parçadan biri de bu havuza karıştırılarak karıştırılan bu parçanın diğerleri arasından bulunabilmesi bahsedilen yöntemler ile denendi. Farklı parça ve havuz boyutları ile yapılan testler her defasında 10.000 defa tekrarlanarak güvenilir ve doğru istatistiklerin alınması sağlandı.

PPM metodu ile birleştirmede, farklı n değerleri için n-gram frekansları hesaplandı. Bu frekansların hesaplanmasında daha önce de belirtildiği üzere kendi-kendine öğrenen bir algoritma yerine 190.000 kelime içeren bir sözlük [2] kullanıldı.

Havuz boyutunun 50 ve dosya parçası boyutunun 16KB olarak belirlendiği deney düzeneğinde bölüm 2.1 ve bölüm 2.2'de belirtilen metotlar sırasıyla test edilmiştir. Şekil 2.3, PPM metodunun belirtilen şekilde üç farklı dosya türü üzerindeki başarım grafiğini vermektedir. Bütün dosya türleri için yapılan testlerde yaklaşık %80 karar verilemezken doğru olarak verilen kararlar txt dosyalarında %4, HTML ve Java dosyalarında ise en yüksek %6 başarım değerine ulaşmıştır. PPM metodunun kalibrasyonu açısından ise txt, HTML ve Java dosyalarında alınan sonuçlara göre en yüksek n-gram değerleri sırasıyla 5-gram, 6-gram ve 4-gram olduğu görülmüştür.



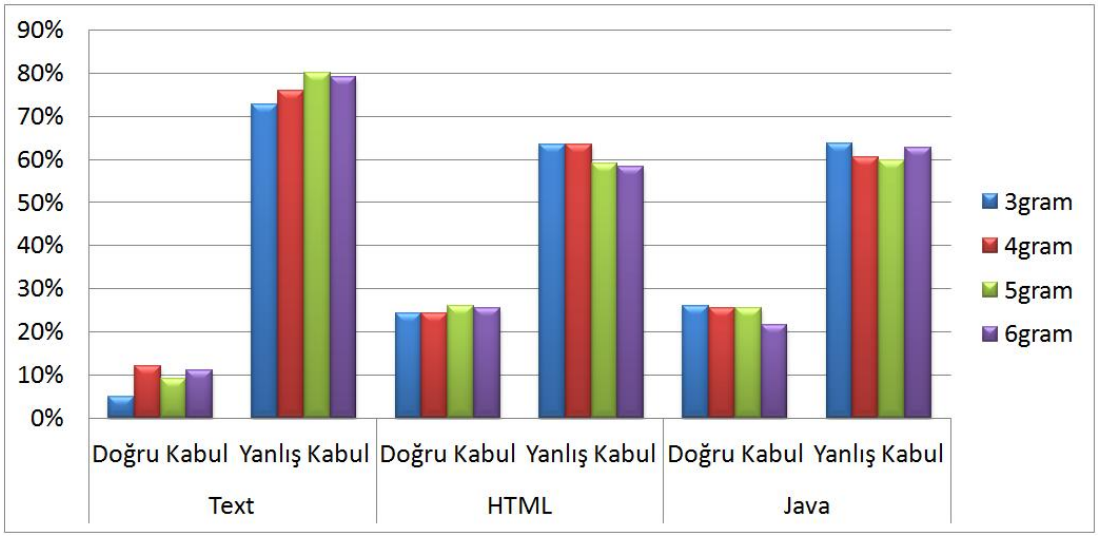
Şekil 2.3: PPM metodunun farklı metin dosyaları üzerindeki performansı



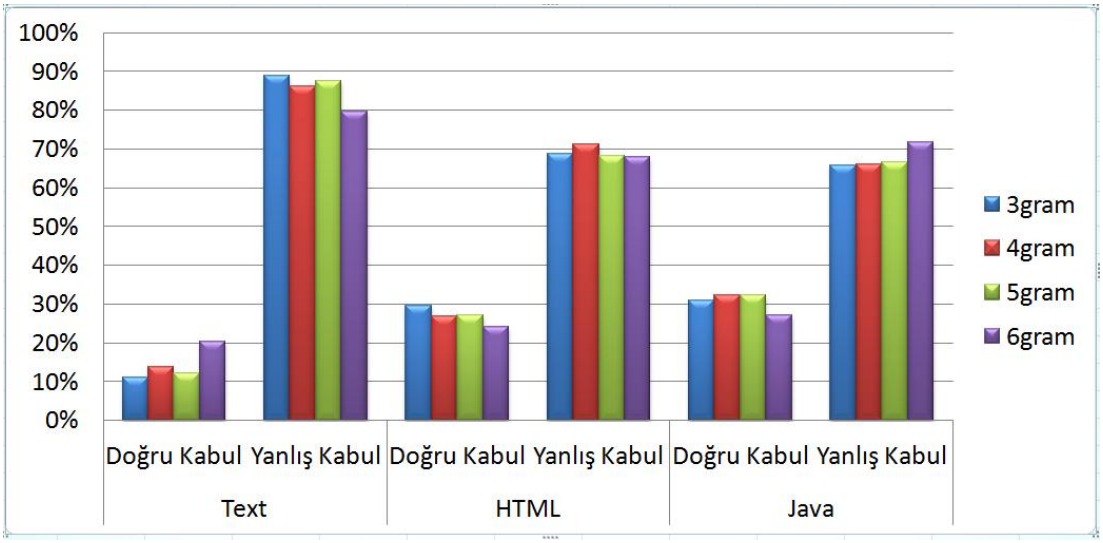
Şekil 2.4: Geliştirilmiş PPM (E-PPM) metodunun farklı metin dosyaları üzerindeki performansı

PPM metodu, tarif edildiği üzere, birleştirilen kelimenin sözlükte mevcut olup olmadığına bakmıyordu. Kullanılan sözlük İngilizce dili için yeteri kadar geniş bir sözlük olduğu için, birleştirilen kelimenin sözlükte bulunabilme zorunluluğu metoda dâhil edildiğinde sonuçlar gözle görülür bir şekilde iyileşmektedir. Şekil 2.4,

geliştirilmiş PPM (Enhanced PPM - EPPM) metodu dediğimiz bu yöntem ile alınan sonuçları göstermektedir. Alınan doğru kararların bir öncekine göre yaklaşık dört kat artmış ve txt dosyaları için %20 seviyelerine ulaşmış olduğu görülecektir.

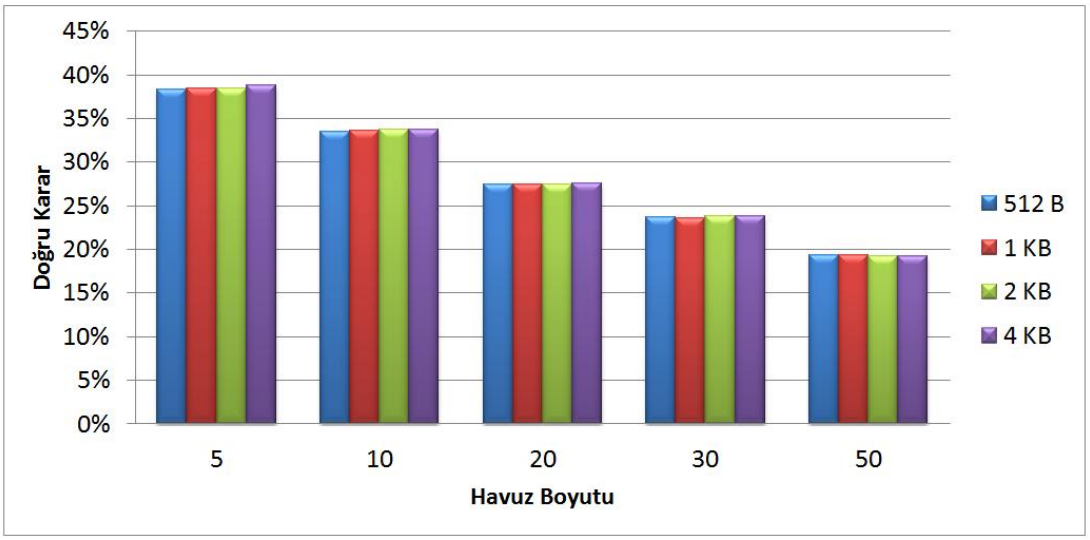


Şekil 2.5: Yersizlik ölçütü (OOP) metodunun farklı metin dosyaları üzerindeki performansı



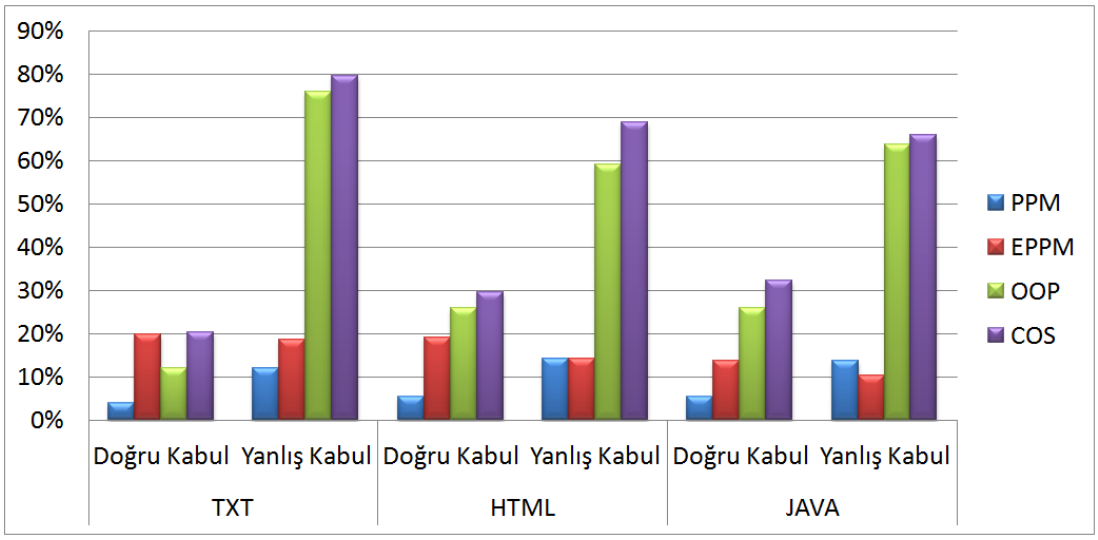
Şekil 2.6: Kosinüs benzerliği (COS) metodunun farklı metin dosyaları üzerindeki performansı

Yersizlik ölçütü (OOP) yöntemiyle mevcut üç dosya türü üzerinde yapılan test sonuçları ise şekil 2.5'te verilmiştir. Metodun yüksek oranda karar verme kabiliyetinin yanında tolere edilemeyecek kadar yüksek yanlış kararlar vermesi sadece bu metot ile yapılabilecek bir birleştirme işleminin mümkün olmadığını göstermektedir. Benzer sonuçları kosinüs benzerliği yönteminin kullanıldığı şekil 2.6'da da görebiliriz.



Şekil 2.7: PPM metodunun farklı havuz ve parça boyutları karşısındaki performansı

Yapılan testlerde parça boyutunun performans üzerinde kayda değer bir etkisinin olmadığı gözlendi. Bunun başlıca sebebi n-gram ile çalışan bir PPM metodunun ilk parçanın sadece son (n-1) karakterini ve ikinci parçanın da sadece ilk (n-1) karakterini almasıdır ki bu da parça boyutunu denklemden düşürür. Bu sebeple günümüz depolama aygıtlarının çoğunda blok boyutu olarak belirlenen 16KB'lık parçalar ile testler yapıldı. Fakat şekil 2.7'de gösterildiği üzere, problemin boyutu havuzun boyutu ile doğru orantılı olduğu için havuz boyutu arttıkça doğru verilen kararlarda gözle görünür bir azalma meydana gelmektedir. Bu azalma şekilden de görüldüğü gibi havuz boyutu 5 olduğunda %38 iken havuz boyutu 50 olduğunda %19'a düşmüştür.

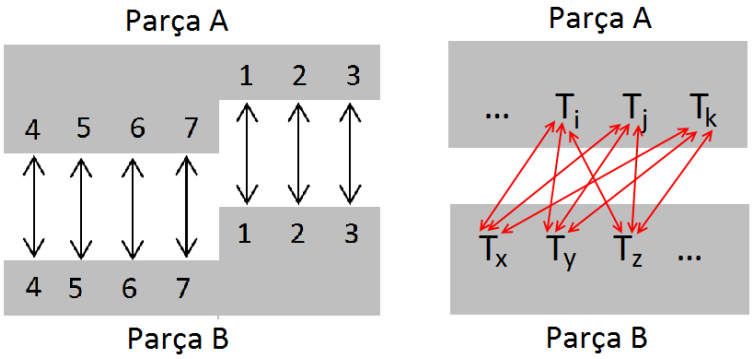


Şekil 2.8: Yakınlık ölçütlerinin farklı metin dosyaları üzerindeki en iyi performansları

Mevcut yakınlık ölçütlerinin genel olarak kıyaslandığı tablo şekil 2.8'de verilmiştir. E-PPM ile yakalanabilen en yüksek değer txt, HTML ve Java dosyaları için sırasıyla %19.90, %13.70 ve %19.00 olurken bu değerler yersizlik ölçütü için %12.08, %26.00 ve %25.90 ve kosinüs benzerliği için %20.36, %29.52 ve %32.30 olmuştur. PPM metodu ve metin dosyalarının sınıflandırılmasındaki başarılarına rağmen, n-gram frekans profilleri, metin parçalarının birleştirilmesinde oldukça sınırlı bir kullanım ve verime sahip olduğu yapılan testler neticesinde görülmüştür.

3 METİN DOSYALARININ YAKINLIĞININ ÖLÇÜLMESİNDE DOSYA YAPILARININ KULLANILMASI

Genel olarak, bir önceki test sonuçları göstermiştir ki ne metin parçalarının sınırlarını dikkate alan modeller ne de n-gram frekans istatistikleri yoluyla içeriği dikkate alan modeller metin dosyalarının birleştirilmesi problemine tatmin edici bir çözüm olamadılar. Bu bağlamda, metin dosyalarının yapılarını da işleme dâhil ederek hem PPM metodunu hem de bölüm 2.2'de tanıtılan metotları daha faydalı hale getirerek birleştirme performansını artıran yeni bir metot üzerinde çalıştık. Yaklaşımımız, resim dosyalarının kurtarılmasında %90'ın üzerinde bir başarım sağlayan teknikten[9] ilham almıştır.



Şekil 3.1: İki parça arasındaki aday ağırlıkları belirlerken piksel değeri karşılaştırma(sol) ve etiket-kelime ilişkisinden yararlanma(sağ)

Şekil 3.1 (solda) resim dosya parçalarının birleştirilmesi işlemini resmeder. Bu işlemde parçalar A ve B'nin potansiyel birleşme sınırları üzerinde bir süreklilik / pürüzsüzlük şartı aranır. Neredeyse bütün doğal resimler mekânsal pürüzsüzlük gösterdiğinden basitçe sınırlarda karşılıklı gelen piksel değerleri farkları toplamı iki parçanın birbirinden ne kadar uzak olduğunu gösterir. Bu gösterge, resim dosyalarının birleştirme işlemi için oldukça sağlam bir istatistik sunar.

Bu noktadan bakıldığında PPM metodunun ve n-gram frekans istatistikleri metotlarının neden düşük performans sergiledikleri de daha iyi anlaşılır. Esasen, mevcut metin dosyası kurtarma teknikleri ya birkaç karakter uzunluğundaki bir sınırda çalışır ki içeriği algılamak için çok kısadır ya da daha genel parça istatistikleri kullanır ki bu da parçalanma noktası etrafında yeterince yerel bir bilgi sağlamaz. Bu yüzden yaklaşımımızın püf noktası: metin dosyalarında oldukça sınırlı şekilde bulunan ama var olan yapıları kullanarak üzerinde çalışılan parçalanma noktası sınırlarını genişletmektir.

Metin dosyaları değişmez yapılara sahip olmasa da kesin karakteristik özelliklere sahiptir. Örneğin, posta kutusu dosyaları, bilgisayar günlüğü dosyaları, HTML, XML ve programlama dilleri gibi çok bilinen metin dosyaları bir takım yapılara sahiptir ve kullandıkları yapılar bu amaç doğrultusunda kullanılabilir.

Diğer bir deyişle, resim parçalarını birleştirirken eşleşen pikseller üzerinden değerlendirme yapar gibi, dosya türüne özgü etiket kelimeler arasındaki ilişkiler kullanılarak iki parçanın bitişikliği hakkında fikir yürütülebilir. Şekil 3.1 (sağ)'da resmedilen bu yaklaşıma "metin yapıları ilişkileriyle tahmin (Prediction through Text-File Construct Relations - PTCR) yöntemi" adını verdik.

Yaklaşımın verimini göstermek için, bu çalışmada, sadece biçimlendirme dillerine ve programlama dillerine odaklandık. Örnek dosyaların çokluğu ve kullanım yaygınlığı sebebiyle HTML ve Java türleri üzerinde çalıştık. İki dosya türü de metin tabanlı olmasına rağmen verilerin işlenmesi ve/veya gösterilmesi için belirli etiket kelimeler kullanırlar. Bu yaklaşım kolay bir şekilde benzer yapıları olan diğer dosya türlerine de uygulanabilir.

Herhangi bir biçimlendirme dilinde ve programlama dilinde ön-tanımlı etiket kelimelerin dosya boyunca rastgele değil belli bir yapıda dizildiğini söyleyebiliriz. Her ne kadar bu yapı kesin kurallarla ifade edilemese de bu kelimeler bulundukları dosyalarda diğer etiket kelimeler ile ilişkileri içinde stokastik bir şekilde incelenebilir. Yaklaşımımız kısaca, etiket kelimelerin dosyalar içerisindeki

sıralamalarını kullanarak istatiksel bir model oluşturur ve bu bilgi ile daha güvenilir birleştirme ölçütleri hesaplanmasına yardımcı olur.

3.1 PTCR Metodunun Uygulanması

Önceki bölümde belirtildiği üzere PTCR metodunun uygulanabilmesi için önce dosya türüne özel etiket kelimelerin tespit edilip bu kelimeler arasındaki ilişkilerin istatiksel bir modelde toplanmasına ihtiyaç vardır. Bunun istatiksel bir model olmasının sebebi, bu kelimeler arasında sıkı kuralların olmamasıdır. Öncelikli olarak belirlenen etiket kelime listeleri HTML ve Java dosya türü için ayrı ayrı çıkartıldı. $P(C\ |D,n)$ olasılığını bir dosya üzerinde D etiket kelimesinden n kelime sonra C etiket kelimesini görme olarak tanımladık ve $n = 1, 5, 10$ için istatistik tabloları oluşturduk. Ardından bir internet robotu ile birlikte işlenen HTML ve Java dosyaları istatistik tabloları istikrarlı hale gelene kadar çalıştırıldı. Bu tablolar istikrarlı hale geldiğinde sırasıyla 350.000 rastgele HTML bağlantısı ve 50.000 Java kaynak kodu dosyası işlenmişti. Böylece PPM metodunda n-gram frekanslarına bakmak için kullanılan başvuru çizelgesine benzer bir çizelge oluşturularak PTCR metodunun çalışabilmesi için gerekli olan istatiksel model tamamlandı.

Oluşturulan istatiksel model üzerinden bir yakınlık ölçümü için ihtiyaç duyulan matematiksel yapı olasılıkların birleştirilmesi[1] yöntemi ile hesaplandı. Önerilen olasılık birleştirme yönteminde, $D_1,\ D_2,.D_n$ bilgilerinin bağımsız olduğu durumda, C nin olma olasılığı denklem 3.1 ile hesaplandı.

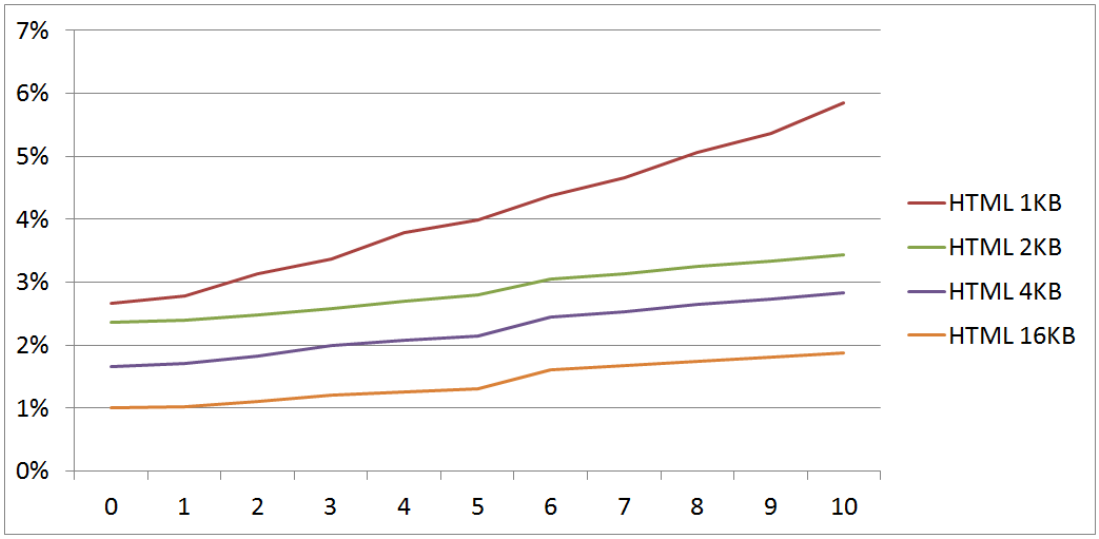
Denklem 3.1 ve denklem 3.2'de yer alan C_i ve D_i değerleri sırasıyla iki aday metin parçası üzerindeki etiket kelimeleri temsil etmektedir.

Denklem 3.1'de bulunan α değeri D_i değerlerine bağlı bir normalleştirme sabiti olduğu için kıyaslama işlemlerinde aynı olacağından dolayı ihmal edildi. Burada $C_1,\ C_2,.C_n$ bilgilerini de bağımsız kabul edersek denklem 3.2 elde edilmiş olur.

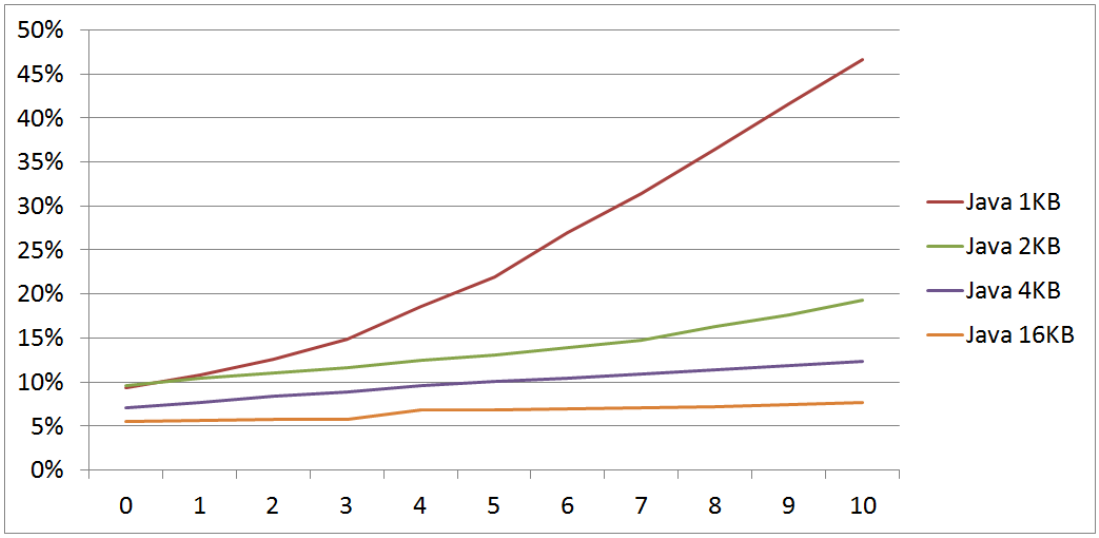
$$P(C \mid D_1, D_2, .., D_N) = \alpha \frac{\prod_{n=1}^{n=N} P(C \mid D_n)}{P(C)^{N-1}} \tag{3.1}$$

$$P(C_1, .., C_N \mid D_1, D_2, D_3) = \prod_{n=1}^{n=N} P(C_n \mid D_1, D_{2,} D_3) \tag{3.2}$$

Yine işlenen HTML ve Java dosya parçaları incelendiğinde sahip oldukları etiket kelime sayısı ile parça boyutları arasında doğru orantı olduğu görüldü. Şekil 3.2 ve şekil 3.3 sırasıyla HTML ve Java dosya parçaları üzerinde yapılan etiket kelime sayısı analizi grafiklerini göstermektedir. Düşük sayıda etiket kelime içeren parça sayısının dosya boyutu arttıkça iki dosya türü içinde azaldığı görülmektedir. Hiç etiket kelime bulunmayan parça oranı HTML parçalarında dosya boyutu *512B* olduğunda %4,5 iken *16KB* olduğunda bu değer %1,01'e düşmektedir. Benzer durum Java parçaları için de geçerlidir. Bu durum parçaların birleştirilmesinde bakılan sınırın genişletilmesinin doğal bir sonucudur. Ayrıca grafik incelendiğinde üç ve daha fazla sayıda etiket kelime içeren parça oranı üçten az etiket kelime içeren oranlara göre daha az olduğu görülecektir. Bu sebeple matematiksel formülde kullanılan N değeri için maksimum değer olarak 3'ün belirlenmesi metodun işleyişi açısından verimli olacaktır. Böylece bir taraftan ayrım yapmaya yetecek kadar sınırlar genişletilirken diğer taraftan da parçalanma noktasından fazla uzaklaşılarak yerel bilgi kaybedilmeyecektir.



Şekil 3.2: HTML dosya parçaları içinde bulunan etiket-kelime sayısı analizi



Şekil 3.3: Java dosya parçaları içinde bulunan etiket-kelime sayısı analizi

3.1.1 PTCR Metodunun Fraksiyonları

En başarılı fraksiyonun bulunabilmesi adına PPM metoduna benzer bir şekilde kayan pencere konsepti PTCR metoduna uygulandı. Dosya parçalarından en fazla üç etiket kelime alınmasının hataları minimize edeceği önceki bölümde açıklanmıştı. Bu sebeple oluşturulan fraksiyonlarda en fazla üç etiket-kelimenin kullanılmasına izin verilmiştir. Oluşturulan fraksiyonlar şu şekilde listelenebilir:

$PTCR_0$: Bu metot da, sadece sınırlardaki etiket kelimeler işleme dahil edilmiştir. İlk parçanın son etiket-kelimesi ile ardından gelen aday parçanın ilk etiket-kelimesi kullanılmıştır.

$$P_{PTCR0}(A,B) = P(T_x \mid T_k) \tag{3.3}$$

Şekil 3.4: $PTCR_0$ metoduna göre iki parçanın ilişkilendirilmesi

$PTCR_1$: Bu metot da, sabit bir pencere öngörülmüş ve ilk parçanın son üç etiket-kelimesi ile ardından gelen aday parçanın ilk etiket-kelimesi arasındaki eşleştirme kullanılmıştır.

$$P_{PTCR1}(A,B) = P(T_x \mid T_i, T_j, T_k) \tag{3.4}$$

Şekil 3.5: $PTCR_1$ metoduna göre iki parçanın ilişkilendirilmesi

$PTCR_2$: Bu metotta, büyüyen bir pencere öngörülmüştür. İlk parça üzerindeki kısmı sabit kalan pencerenin boyutu ikinci iterasyonda bir karakter büyütülmüştür. İlk parçanın son üç etiket-kelimesi ile ardından gelen aday parçanın ilk iki etiket-kelimesi arasındaki eşleştirme kullanılmıştır.

$$P_{PTCR2}(A,B) = P(T_x \mid T_i, T_j, T_k) * P(T_x, T_y \mid T_i, T_j, T_k) \quad (3.5)$$

Şekil 3.6: $PTCR_2$ metoduna göre iki parçanın ilişkilendirilmesi

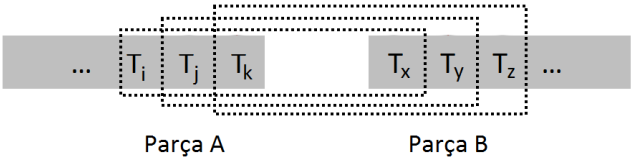
$PTCR_3$: Bu metotta, büyüyen bir pencere öngörülmüştür. İlk parça üzerindeki kısmı sabit kalan pencerenin boyutu her iterasyonda bir karakter büyütülmüştür. İlk parçanın son üç etiket-kelimesi ile ardından gelen aday parçanın ilk üç etiket-kelimesi arasındaki eşleştirme kullanılmıştır.

$$P_{PTCR3}(A,B) = P(T_x \mid T_i, T_j, T_k) * P(T_x, T_y \mid T_i, T_j, T_k) * P(T_x, T_y, T_z \mid T_i, T_j, T_k) \quad (3.6)$$

Şekil 3.7: $PTCR_3$ metoduna göre iki parçanın ilişkilendirilmesi

$PTCR_4$: Bu metotda, 4 boyutunda kayan bir pencere modeli uygulanmış ve ilk parçanın son üç etiket-kelimesi ile ardından gelen aday parçanın ilk üç etiket-kelimesi arasındaki eşleştirme yapılmıştır.

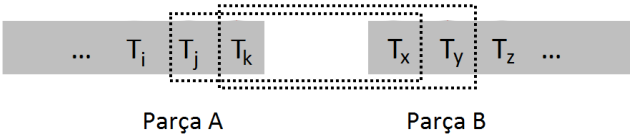
$$P_{PTCR4}(A,B) = P(T_x \mid T_i, T_j, T_k) * P(T_x, T_y \mid T_j, T_k) * P(T_x, T_y, T_z \mid T_k) \quad (3.7)$$



Şekil 3.8: $PTCR_4$ metoduna göre iki parçanın ilişkilendirilmesi

$PTCR_5$: Bu metotta, 3 boyutunda kayan bir pencere modeli uygulanmış ve ilk parçanın son iki etiket-kelimesi ile ardından gelen aday parçanın ilk iki etiket-kelimesi arasındaki eşleştirme yapılmıştır.

$$P_{PTCR5}(A,B) = P(T_x \mid T_j, T_k) * P(T_x, T_y \mid T_k) \quad (3.8)$$



Şekil 3.9: $PTCR_5$ metoduna göre iki parçanın ilişkilendirilmesi

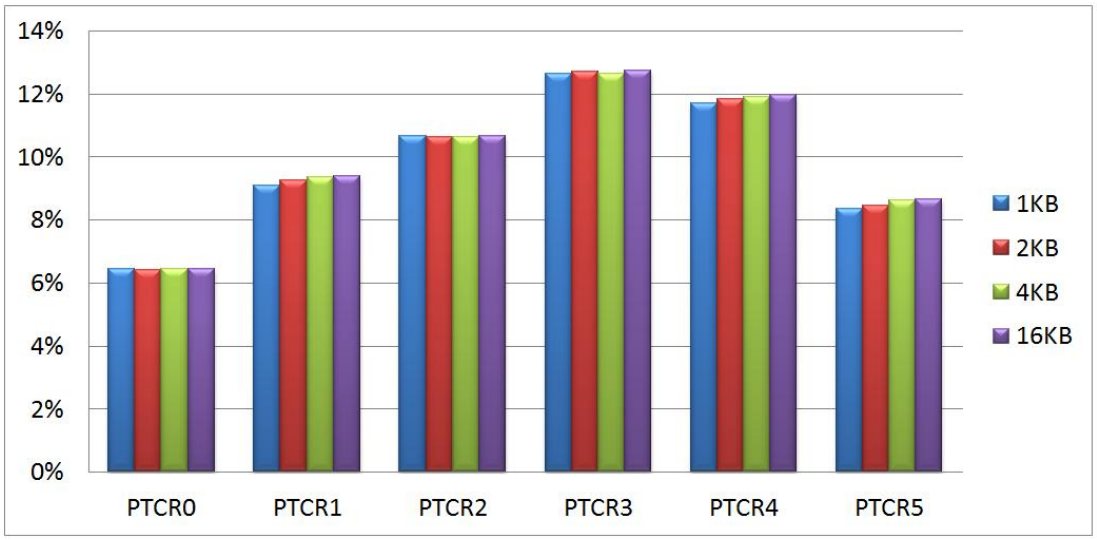
PTCR metodunun performansı bölüm 3.2'de gösterilmiştir.

3.2 PTCR Metodunun Performansı

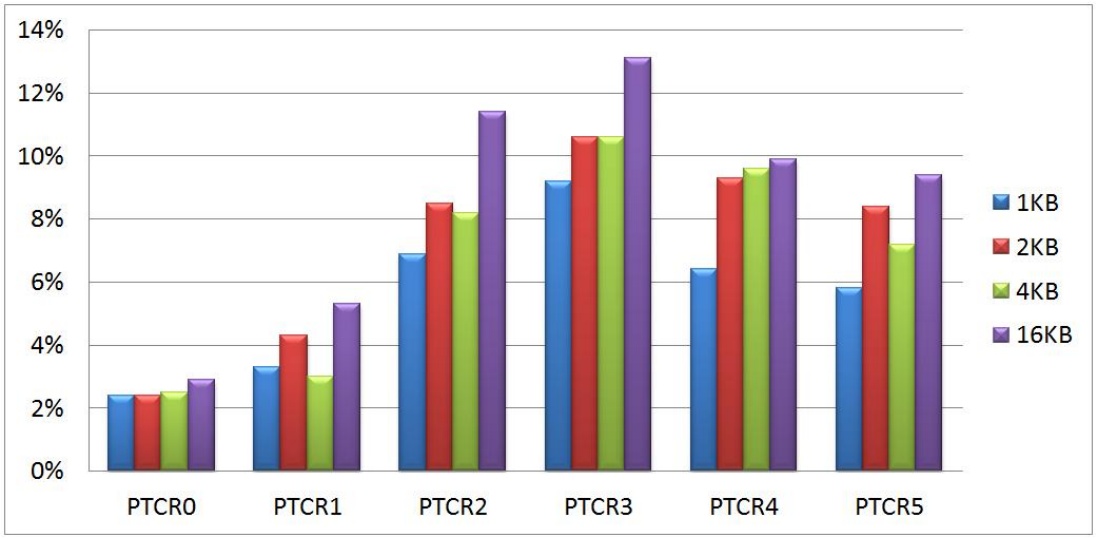
PTCR metodunu HTML dosyaları ve Java kaynak kodu dosyaları üzerinde test ettik. Test düzeneği için bölüm 2.3'te olduğu gibi ilgili dosya türüne ait parçalardan oluşan bir havuz kuruldu. Böylece ilgili metotlar kullanılarak, birbirini takip eden parçaların diğer ilgisiz olan parçalar arasından bulunması test edildi.

Şekil 3.10 ve şekil 3.11 incelendiğinde parça boyutunun metodun performansı üzerinde dikkate değer bir etkisinin olmadığı görülecektir. Bu durum şekil 2.7'deki durum ile aynı sebepten oluşmuştur. Çünkü PTCR metodunun çalışabilmesi için en fazla üç etiket-kelimeye ihtiyaç vardır. Bu kelimeler ise parçaların içeriğine bağlı olarak küçük boyutlu parçalarda da bulunmaktadır. *1KB*'lık HTML parçalarının %95'inden fazlasında üç ve daha fazla etiket kelime olduğu şekil 3.2'de

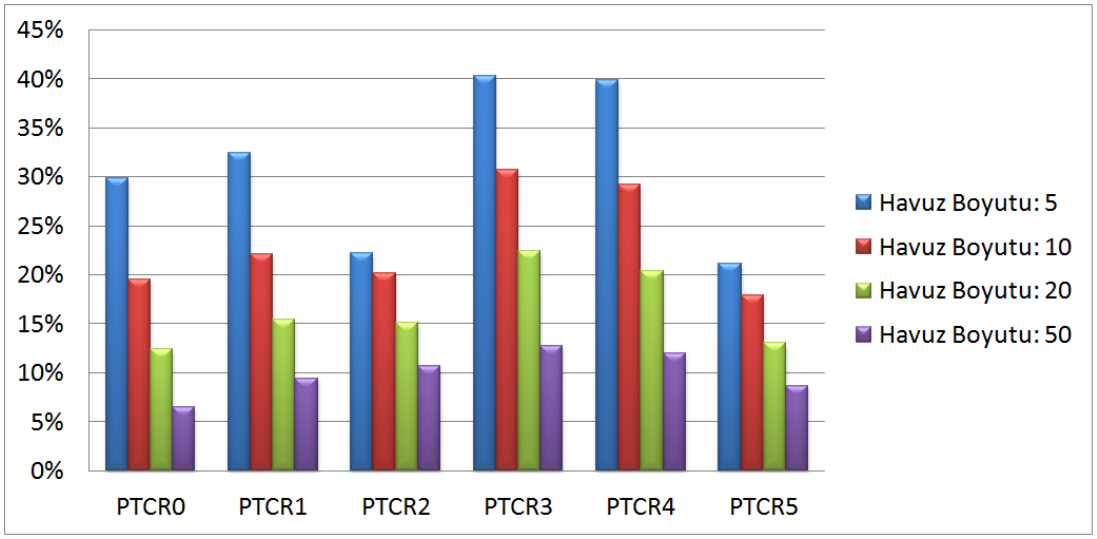
görülmektedir. Fakat bu durum *1KB*'lık Java parçalarında %85 civarında bir değer almaktadır ki bu da şekil 3.11'deki durumu açıklamaktadır.



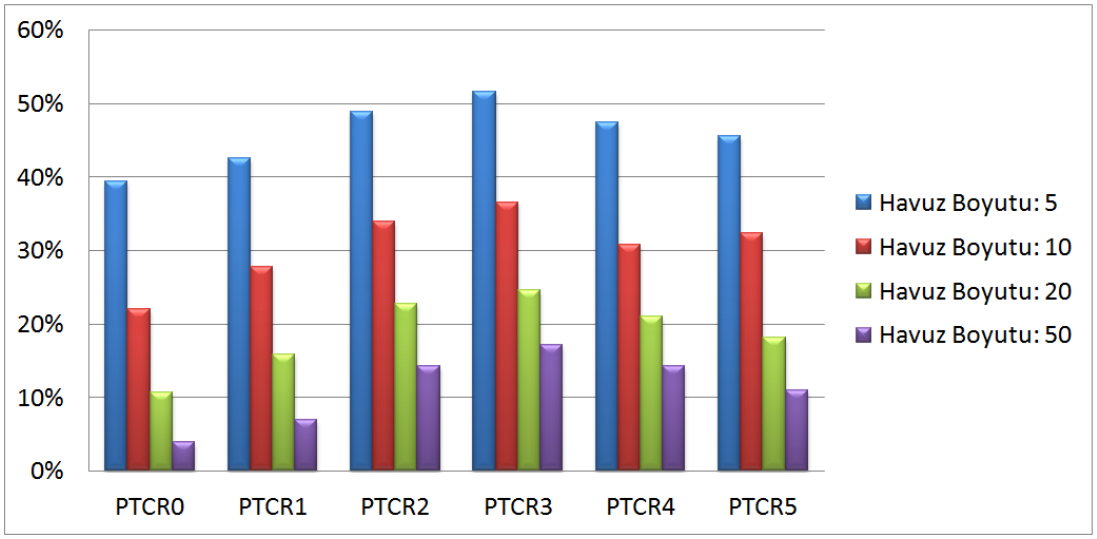
Şekil 3.10: PTCR metodunun havuz boyutu 50 iken farklı parça boyutları ile yapılan test sonuçları (HTML)



Şekil 3.11: PTCR metodunun havuz boyutu 50 iken farklı parça boyutları ile yapılan test sonuçları (Java)



Şekil 3.12: PTCR metodunun parça boyutu 16KB iken farklı havuz boyutları ile yapılan test sonuçları ve karşılaştırması. (HTML)



Şekil 3.13: PTCR metodunun parça boyutu 16KB iken farklı havuz boyutları ile yapılan test sonuçları ve karşılaştırması. (Java)

Şekil 3.12 ve şekil 3.12 gözden geçirildiğinde ise PTCR metodunun bütün fraksiyonları için performans değerinin havuz boyutu ile ters orantılı olarak azaldığı görülmektedir.

Çizelge 3.1 ve çizelge 3.2'de PTCR metodunun bölüm 2'de tanıtılan metotlar ile karşılaştırması verilmiştir. Metotlar parça boyutunun *16KB*, havuz boyutunun ise 50 olduğu durumda test edilmiştir. Metotların verdiği kararları doğru olup olmadığı ayrıca test edilmiş; "doğru karar" ve "yanlış karar" etiketleriyle etiketlenmiştir. Çizelgelerde yer alan bir diğer sütun ise metotların herhangi bir karar veremedikleri durumları anlatan "kararsız" sütunudur. Metotlar hiçbir sonuç alamadıkları durumlarda ya da en iyi eşleştirmeyi birden fazla parça için yaptıkları durumlarda "kararsız" sonucunu almışlardır. Bu metotların nasıl bir arada kullanılabileceği bölüm 4'de anlatılmıştır.

Çizelge 3.1: Yakınlık metotlarının HTML dosya parçaları üzerinde karşılaştırılması

Metotlar	Doğru Karar	Yanlış Karar	Kararsız
PPM	5,40%	14,10%	80,50%
E-PPM	19,00%	14,30%	66,70%
OOP	26,00%	59,10%	14,90%
COS	29,52%	68,87%	1,62%
PTCR	12,76%	54,64%	32,60%

Çizelge 3.2: Yakınlık metotlarının Java dosya parçaları üzerinde karşılaştırılması

Metotlar	Doğru Karar	Yanlış Karar	Kararsız
PPM	%5,40	%13,70	%80,90
E-PPM	%13,70	%10,30	%76,00
OOP	%25,90	%63,60	%10,50
COS	%32,30	%66,00	%1,70
PTCR	%12,28	%68,13	%19,59

4 MEVCUT YÖNTEMLER İLE ÖNERİLEN YÖNTEMİN BİR ARADA KULLANILMASI

Önceki bölümde mevcut yöntemler ile önerilen yöntemin tek başına HTML ve Java dosya parçaları üzerinde performansları çizelgeler 3.1 ve 3.2'de gösterilmişti. Belirtilen metotların tek başlarına dosya birleştirme işlemi için yetersiz oldukları açıkça görülmektedir. Bu noktada bu metotların beraber kullanıldığı bir sistem bu probleme çözüm olabilir. Fakat böyle bir sistemi oluşturabilmek ancak belirtilen metotlar birbirinin eksiğini kapatabildiği zaman mümkün olur. Burada şunu belirtmek gerekir ki, doğru kararları artırılması ne kadar önemliyse yanlış kararların azaltılması da o kadar önemlidir. Metotların çıktıları daha detaylı incelendiğinde özellikle PPM, E-PPM ve diğerlerine kıyasla PTCR metotlarında yüksek oranda kararsızlık durumuyla karşılaşıldığı görülür. Başlangıçta olumsuz gibi görünen bu durum üzerinde çalışılıp karar verilebilecek ve potansiyel olarak doğru karar oranını artıracak bir alan olduğundan birleşme açısından uygun bir zemin oluşturmaktadır. Diğer taraftan bahsedilen bütün metotların performanslarının havuz boyutu ile ters orantılı olduğu gösterilmişti. Dolayısıyla havuzdaki ilgisiz olan parçaların bir ön eleme sisteminden geçmesi de bütün metotların özellikle E-PPM ve PTCR metotlarının performansını artıracağı tahmin edilebilir.

Bütün bu veriler, kosinüs benzerliği ve yersizlik yöntemini aynı grupta değerlendirecek olursak, bahsedilen üç metodun bir şekilde birleştirilerek bir metodun çıktısının diğer metodun girdisi olacak şekilde bir boru hattı işlemesinin (pipelining) metotları birleştirme yöntemi olarak uygun bir yöntem olacağını göstermektedir. Bu sistemde metotların sıralamalarına ve üstlenecekleri görevlere karar vermek için metotların performanslarına farklı bir bakış ile tekrar baktık.

Metotların performanslarını doğru parçayı bulup bulamamaları yerine doğru parçayı kaçıncı sırada buldukları kıstasıyla tekrar karşılaştırdık.

Çizelge 4.1: Yakınlık ölçütlerinin düz metin parçaları arasında doğru parçayı ilk n sırada bulma performansları

Metotlar	5	10	15	20	25	30
PPM	%7,86	%11,11	%12,35	%13,58	%14,7	%15,26
E-PPM	%32,66	%37,93	%37,93	%38,27	%38,38	%38,38
OOP	%35,13	%52,12	%62,57	%67,39	%69,80	%73,13
COS	%50,73	%71,37	%82,02	%87,35	%91,21	%93,74

Çizelge 4.2: Yakınlık ölçütlerinin HTML parçaları arasında doğru parçayı ilk n sırada bulma performansları

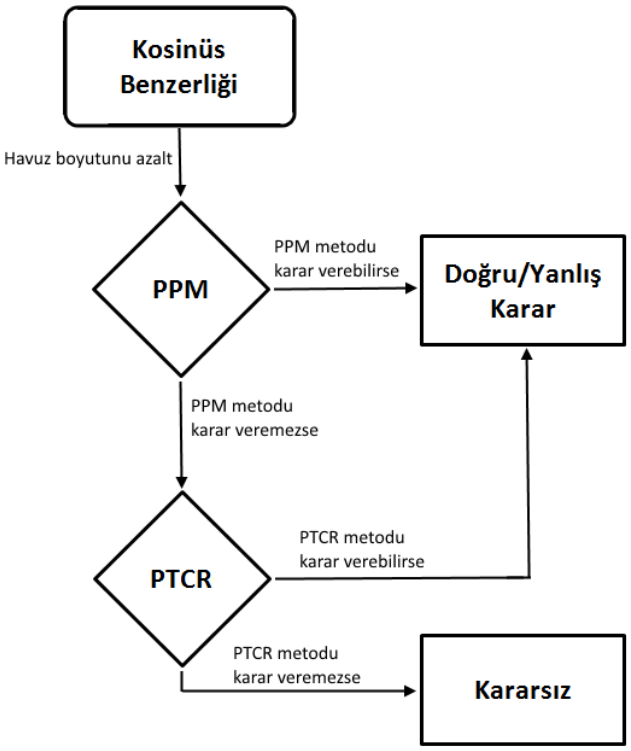
Metotlar	5	10	15	20	25	30
PPM	%17,76	%21,50	%23,36	%23,99	%23,99	%23,99
E-PPM	%31,30	%33,10	%33,10	%33,10	%33,10	%34,30
OOP	%63,90	%73,40	%77,60	%81,30	%83,50	%84,90
COS	%60,50	%71,30	%77,50	%82,00	%89,70	%94,00
PTCR	%33,66	%47,44	%54,86	%59,10	%62,06	%64,32

Çizelge 4.2: Yakınlık ölçütlerinin HTML parçaları arasında doğru parçayı ilk n sırada bulma performansları

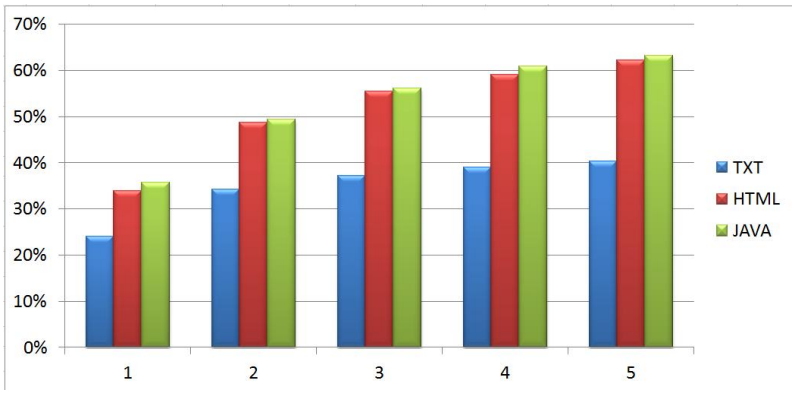
Metotlar	5	10	15	20	25	30
PPM	%12,60	%14,80	%17,00	%17,60	%17,90	%18,00
E-PPM	%20,90	%21,20	%21,20	%21,20	%21,20	%22,40
OOP	%60,10	%70,30	%76,80	%81,50	%85,30	%87,80
COS	%62,40	%75,30	%82,00	%85,90	%88,60	%90,70
PTCR	%33,29	%48,41	%59,38	%67,63	%73,86	%77,4

Çizelge 4.1, 4.2 ve 4.3'te ilgili metotların doğru parçayı kaçıncı sırada buldukları gözükmektedir. Çizelgeler incelendiğinde kosinüs benzerliği ile yersizlik ölçütünün yakın performe ettiği fakat kosinüs benzerliği yönteminin az bir farkla daha iyi olduğu görülecektir. Havuzdaki ilgisiz parçaların elenmesi adına yapılacak bir filtreleme görevini dolayısıyla kosinüs benzerliği yönteminin yapmasının yerinde olduğu açıktır. Boru hattı sistemine göre diğer aşamada yani kosinüs benzerliği yönteminin çıktısının girdisi olacağı sıradaki metot E-PPM metodu olmalıdır. PTCR

ve PPM metotlarının performansları incelendiğinde PPM ve E-PPM metodunun karar verdiği durumlardaki doğru karar oranı PTCR metoduna göre oldukça yüksektir. Buna mukabil, PPM metotlarının yüksek orandaki kararsızlık durumları PTCR metodunun devreye girmesiyle pozitif yönde değiştirilebilir. Böylece, metin dosyalarının yakınlığının ya da bitişikliğinin ölçülmesinde öngörülen sistem belirlemiş olunur. Şekil 4.1'de bu sistem anlatılmaktadır.



Şekil 4.1: Metin dosyalarının bitişikliğinin ölçülmesi için önerilen sistem



Şekil 4.2: Önerilen sisteminin farklı dosya türler üzerinde doğru parçayı ilk n sırada bulma performansı

Şekil 4.2'de önerilen sistem ile alınmış sonuçlar görülmektedir. Metotların ayrı ayrı performansları dikkate alındığında kayda değer bir iyileşme söz konusudur. Fakat yine de her ne kadar bitişikliğin ölçülmesinde başarılı sonuçlar alınmışsa da hedeflenen otomatik birleştirme kabiliyetinden yoksun olduğu görülecektir. Önerilen sistemin doğru parçayı ilk beş sırada verme performansının yüksek olması, önerilen sistemin yarı-otomatik birleştirme yapabilmesini mümkün kılmıştır.

5 SONUÇ

Dijital cihazlar ve bilgisayarlar her geçen gün hayatımıza daha çok girmekteler. Bu cihazlarda, posta kutusu dosyaları, bilgisayar günlükleri, biçimlendirme dillerine ait dosyalar ve kaynak kodu dosyaları gibi kullanıcılar için önemli olan dosyalar tutulmaktadır. Bütün bu dosya türlerinin ortak özelliği ise metin-tabanlı dosyalar halinde tutulmalarıdır. Bütün bunlara ek olarak cihazların ve bilgisayarların üzerinde çalıştığı birçok dosya harf, rakam ya da sembolleri tutmak için metin tabanlı dosyalar halinde depolanır. Diğer taraftan sistem arızaları, ani elektrik kesintileri ya da sert darbeler sonucu cihazlardaki dosya sistemi ya da dosyaların kendisi hasar görebilir. Bunlar metin dosyalarının kurtarılmasının kullanıcılar için gerekli ve önemli olduğunu göstermektedir.

Metin dosyalarının kurtarılmasındaki ilk aşama olan metin parçalarının birbirleri ile ilişkilendirilmelerinde mevcut yöntemler bölüm 2'de anlatıldı. Bölüm 3'de dosya yapıları üzerinde istatistiksel bir model kuran PTCR yöntemi tanıtıldı. Alınan sonuçlar incelendiğinde metotların tek başına performanslarının metin dosyalarının bitişikliğinin ölçülmesinde yeterli olmadığı görüldü. Bölüm 4'te mevcut yöntemler ile önerilen yöntemin bir arada kullanılabildiği sistem tanıtıldı. Bu sistem ile bitişiklik ölçülmesi performansı %10 - %20 aralığından %35 seviyesine çıktığı görüldü. Doğru parçanın ilk beş sırada olma performansına bakıldığında ise bu oranın %63 seviyelerinde olduğu gözlemlendi. Bu sonuçlar her ne kadar tam otomatik bir birleştirme işlemi için yeterli olmasa da daha önce manüel olan birleştirme işleminin yarı-otomatik olarak gerçekleştirilmesine olanak sağlamıştır.

KAYNAKLAR

[1] C. Bailer-Jones and K. Smith. Combining probabilities. GAIA-C8-TNMPIA-CBJ-053, July 2011.

[2] R. Beresford. *The UK Advanced Cryptics Dictionary*. Program documentation. Vers. 1.7. N.p., 20 Aug. 2000. Web. 15 May 2012.

[3] W. B. Cavnar and J. M. Trenkle. N-gram-based text categorization. In *In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161-175, 1994.

[4] J. G. Cleary, W. Teahan, and I. H. Witten. Unbounded length contexts for ppm. In *Data Compression Conference*, 1995. DCC '95. *Proceedings*, pages 52-61, Mar 1995.

[5] S. L. Garfinkel. Carving contiguous and fragmented files with fast object validation. *Proc. of Digital Investigation*, 4:2-12, 2007.

[6] LemurProject. The clueweb09 dataset, 2012.

[7] D. T. Meyer and W. J. Bolosky. A study of practical deduplication. *ACM Transactions on Storage* (TOS), 7(4):14, 2012.

[8] A. Pal and N. Memon. The evolution of file carving. *Signal Processing Magazine*, IEEE, 26(2):59-71, March 2009.

[9] H. T. Sencar and N. Memon. Identification and recovery of JPEG files with missing fragments. *Digital Investigation*, 6:88-98, 2009.

[10] K. Shanmugasundaram and N. Memon. Automatic reassembly of document fragments via context based statistical models. In *Computer Security Applications Conference, 2003. Proceedings. 19th Annual*, pages 152-159, Dec 2003.

[11] A. Strehl, E. Strehl, J. Ghosh, and R. Mooney. Impact of similarity measures on web-page clustering. In *Workshop on Artificial Intelligence for Web Search* (AAAI 2000, pages 58-64. AAAI, 2000.

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı : ŞAHİN, Alperen

Uyruğu : T.C.

Doğum tarihi ve yeri : 18.07.1989 Karabük

Medeni hali : Evli

Telefon : 0 (312) 292 40 00

e-mail : alperen.sahin@etu.edu.tr

Eğitim

Derece	**Eğitim Birimi**	**Mezuniyet tarihi**
Lisans	Bilkent Üniversitesi/Bilgisayar	2012

İş Deneyimi

Yıl	**Yer**	**Görev**
2012-2015	TOBB Ekonomi ve Teknoloji Üni.	Burslu Yüksek Lisans Öğrencisi

Yabancı Dil

İngilizce (İleri seviye)

Almanca (Başlangıç seviye)

Yayınlar

A. Sahin and H. T. Sencar. A study on adjacency measures for reassembling text files. *In Systematic Approaches to Digital Forensic Engineering (SADFE), 2015 10th International Conference on*, Sep 2015