# ÖZELLEŞTİRİLMİŞ ANALİTİK BULUT MİMARİLERİNDE DAĞITIK DOSYA SİSTEMLERİ İLE PERFORMANS İYİLEŞTİRMESİ

**MUHAMMED AKİF AĞCA**

**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.

_____

Prof. Dr. Erdoğan DOĞDU

Anabilim Dalı Başkanı

Muhammed Akif AĞCA tarafından hazırlanan ÖZELLEŞTİRİLMİŞ ANALİTİK BULUT MİMARİLERİNDE DAĞITIK DOSYA SİSTEMLERİ İLE PERFORMANS İYİLEŞTİRMESİ adlı bu tezin Yüksek Lisans tezi olarak uygun olduğunu onaylarım.

_____

Prof. Dr. Erdoğan DOĞDU

Tez Danışmanı

Tez Jüri Üyeleri

Başkan : Prof. Dr. Mehmet Ali AKÇAYOL _____

Üye : Doç. Dr. Bülent TAVLI _____

Üye : Prof. Dr. Erdoğan DOĞDU _____

## 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.

MUHAMMED AKİF AĞCA

**Üniversitesi** : **TOBB Ekonomi ve Teknoloji Üniversitesi**
**Enstitüsü** : **Fen Bilimleri**
**Anabilim Dalı** : **Bilgisayar Mühendisliği**
**Tez Danışmanı** : **Prof.Dr. Erdoğan DOĞDU**
**Tez Türü ve Tarihi** : **Yüksek Lisans – AĞUSTOS 2015**

# ÖZELLEŞTİRİLMİŞ ANALİTİK BULUT MİMARİLERİNDE DAĞITIK DOSYA SİSTEMLERİ İLE PERFORMANS İYİLEŞTİRMESİ

**MUHAMMED AKİF AĞCA**

## ÖZET

Teknoloji ve sosyal medyanın hızlı gelişimiyle veri hızı, hacmi ve çeşitliliği artış göstermektedir. Biriken veriye anlık olarak erişim ve karar desteği sağlanması mevcut teknolojiler ile mümkün olmamaktadır. Toplanan verilerin anlık olarak analiz edilmesi ve metin verilerinden bilgi çıkarımları standart veri tabanları ile yapılamamaktadır. Mevcut çözüm ve yöntemler de Türkçe metin için kısıtlı analiz yetenekleri bulunmaktadır. Bu çalışmada veri yoğun, işlemci yoğun uygulamalar için özelleştirilmiş dağıtık analitik sistem ve uygulamaları geliştirilmektedir.

Bu sistemde dağıtık dosya sistemlerinin kullanımı ile performans iyileştirmeleri yapılmıştır. Tasarlanmış olan tek düğümlü ve çok düğümlü sistemlerde performans iyileştirmeleri gözlemlenmiştir. Dağıtık analitik sistemin dağıtık dosya sistemleriyle tasarlanmasıyla hızlı sonuçlar elde edilebileceği gözlemlenmiştir. Mikroblog metin analitiği için özelleştirilmiş platformda farklı algoritmaların performans ve doğruluk değerlendirmeleri yapılmıştır. Mikroblog metin analitiği için dağıtık skorlama algoritmasının k-means kümeleme algoritmasına göre daha hızlı çalıştığı gözlemlenmiştir. Metin analitiği için geliştirilmiş dağıtık algoritmalar tek düğümlü ve çok düğümlü sistemlerde performans olarak karşılaştırılmıştır. Küme performansında bellek kısıtlarının kritikliği gözlemlenmiş ve sistemin bellek ihtiyaçları değerlendirilmiştir.

Geliştirilen dağıtık analitik sistem sayesinde büyük verinin hızlı sorgulanmasına imkân sağlanmaktadır. Uygulamalar için jenerik ve ölçeklenebilir depolama katmanları sağlanmaktadır. Dağıtık analitik uygulamalar için dağıtık mimari kullanımı önerilmektedir. Dağıtık dosya sistemlerinin ölçeklenebilir otomatik düğüm ekleme çıkarma özellikleri sayesinde donanımlar maksimum verimlilikte kullanılmakta ve ölçekleme minimum donanım ve zaman maliyeti ile yapılabilmektedir. Sonuç olarak, dağıtık dosya sistemlerinin özelleştirilmiş analitik bulut mimariler üzerinde analitik işlemler için önemli performans iyileştirmeleri sağladığı ve analitik işlemler için verimliliği arttırdığı gözlemlenmiştir.

**Anahtar Kelimeler:** Dağıtık sistem, Dağıtık dosya sistemi, Akan veri analizleri, Dağıtık analitik, İnteraktif analitik, Büyük veri, Mikroblog metin

**University** : **TOBB Economics and Technology University**
**Institute** : **Institute of Natural and Applied Sciences**
**Science Programme** : **Computer Engineering**
**Supervisor** : **Professor Dr. Erdoğan DOĞDU**
**Degree Awarded and Date** : **M.Sc. – AUGUST 2015**

# PERFORMANCE IMPROVEMENT VIA DISTRIBITED FILE SYSTEMS ON PRIVATE ANALYTIC CLOUDS

**MUHAMMED AKİF AĞCA**

## ABSTRACT

Improvements on current technologies and social media cause increase in data volume, variety, and velocity. Instant access to stored data and providing decision support is very hard with current technologies. Standard data base technologies cannot analyze the data and retrieve information from text data. Current solutions and methodologies have restricted analysis capabilities for Turkish texts. In this study a distributed analytical system and applications are developed for data bound and CPU bound applications.

Performance improvements via distributed file systems are implemented on the system. The improvements are observed on single node and multi node systems. Faster results are obtained via distributed file systems on distributed analytical system. Different algorithms are evaluated in terms of performance and correctness for microblog text analytics on private distributed analytical system. Distributed scoring algorithm gives faster results than k-means clustering algorithm for microblog text analytics. The distributed algorithms developed for text analytics are implemented on single node and multi node systems and compared in terms of cluster performance. Memory constraints are observed on cluster performance and minimum memory requirement of the system is evaluated.

Faster querying on big data is provided via the distributed analytical system. Generic and scalable storage layer is provided for applications. Distributed architecture usage is proposed for distributed analytical applications. Hardware can be used with maximum efficiency, and node replacement can be done at minimum time and minimum hardware cost with the scalability and automated node replacement features of distributed file systems. To sum up, it is observed that distributed file systems provide important performance improvements and improve efficiency for analytical operations on private analytical clouds.

**Keywords:** Distributed system, Distributed file system, Stream data analytics, Distributed analytics, Interactive analytic, Big data, Microblog text

## TEŞEKKÜR

Çalışmalarım boyunca değerli yardım ve katkılarıyla beni yönlendiren hocalarım Prof. Dr. Erdoğan DOĞDU'ya, Doç. Dr. Bülent TAVLI'ya yine kıymetli tecrübelerinden faydalandığım TOBB Ekonomi ve Teknoloji Üniversitesi Bilgisayar Mühendisliği Bölümü öğretim üyelerine, araştırmalarım için sağladığı burstan dolayı TOBB Ekonomi ve Teknoloji Üniversitesi'ne teşekkürü bir borç bilirim.

# İÇİNDEKİLER

**Sayfa**

# ŞEKİL LİSTESİ

**Şekil** **Sayfa**

# ÇİZELGE LİSTESİ

**Çizelge** **Sayfa**

## KISALTMALAR

| Kısaltmalar | Açıklama |
|---|---|
| **OLAP** | Online Analytical Processing |
| **OLTP** | Online Transaction Processing |
| **HDFS** | Hadoop Distributed File System |
| **VPN** | Virtual Private Network |
| **IaaS** | Infrastructure as a Service |
| **PaaS** | Platform as a Service |
| **SaaS** | Software as a Service |
| **ACID** | (Atomicity, Consistency, Isolation, Durability) |

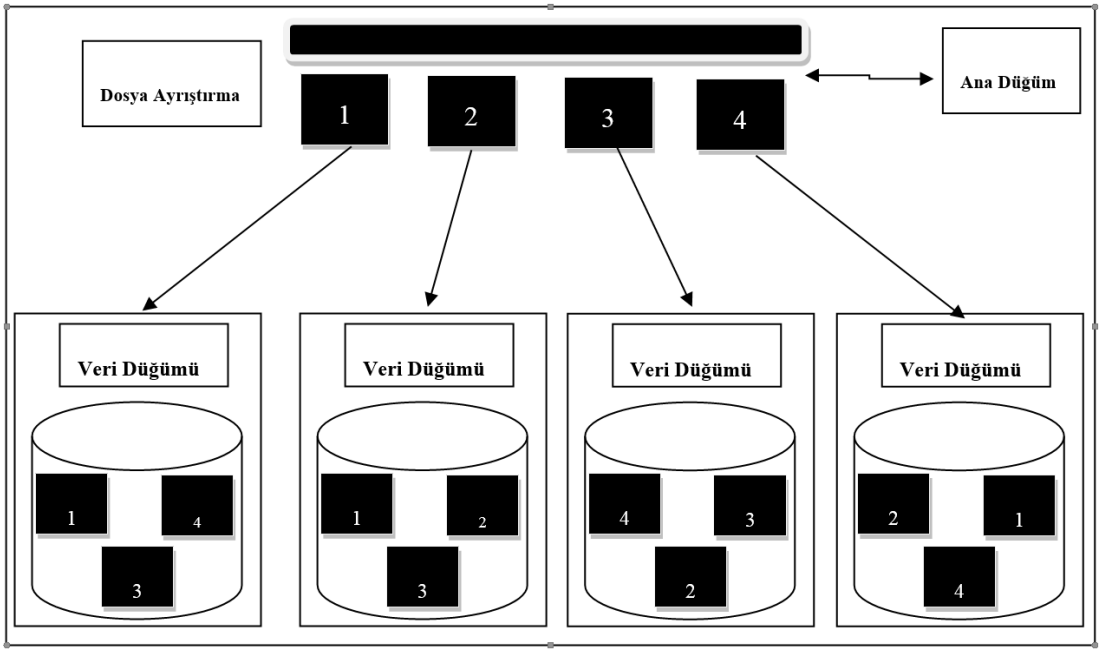
# 1. GİRİŞ ve ÇALIŞMANIN AMACI

Değişken hacimli, değişken hızda ve çeşitliliği fazla olan verilerin yönetimi için klasik veri tabanı yöntemleri yeterli olmamaktadır. Veri hacminin büyük olduğu ve verinin yapısal olduğu durumlarda veri yönetimi için disk ve işlemcisi güçlü makineler yeterli olabilmektedir. Verinin yapısına göre dağıtık veri tabanı sistemleri ile paralel işleme özellikleri kullanılabilmektedir.

Verinin yapısal olmadığı durumlarda bu sistemler yetersiz kalmaktadır. Çünkü verinin standardize edilerek ilişkisel bir tabloda tutulması yapısal olmayan verilerde mümkün değildir. İlişkisel veri tabanlarında ACID (Atomicity, Consistency, Isolation, Durability) özelliği, özellikle verinin atomik olarak saklanabilmesi standart format sağlanabilmesi ile mümkündür. Bu yapıda verinin atomik olarak depolanabilmesi, sık değişim göstermemesi, diğer verilerden ayrıştırılabilmesi ve uzun süre standart olarak kullanılabilmesi önemlidir. Bu sayede veri tabanları ilişkisel olarak oluşturulup kullanıma sunulabilmektedir. Bu özellikleri klasik veri tabanlarında sağlamak çeşitlilik ve hacimden dolayı artık mümkün olmamaktadır.

Yüksek hacim ve çok çeşitlilikte verinin verimli analizi için dağıtık veri analizi platformları kullanılmaktadır. Bu platformlar veriyi küme (cluster) içerisinde düğümlerde (node) özelleştirilmiş dosya sistemlerinde tutmaktadırlar. Bu dosya sistemleri verinin küme üzerinde çoklu kopyalarını tutarak veri kaybı riskini de minimum düzeye indirmektedirler. Verini verimli yönetim için kümedeki düğümlere ana düğüm ve çalışıcı düğüm fonksiyonları atanmaktadır. Ana düğüm veriyi orantılı olarak çalışıcı düğümlere dağıtmakta ve gerekli durumlarda veriye erişimleri yönetmektedir. Herhangi bir düğümde verinin veya düğümün zarar görmesi durumunda verinin diğer düğümlerdeki yedekleri ile zarar gören düğümün tüm fonksiyonları diğer düğümlere aktarılmaktadır. Ana düğümün zarar görmesi durumunda ise en uygun düğüm ana düğüm olarak seçilmekte ve çalışan düğümler kendi durumlarını ana düğüme bildirerek sistem güncellenmektedir.

Verinin analizi için dağıtık programlama metotları kullanılmaktadır. Son yıllarda Hadoop temelli platformlar öne çıkmaktadır. Hadoop veriyi kendi dosya sisteminde ( HDFS - Hadoop Distributed File System) dağıtık yapıda tutmaktadır. Şekil 1'de görüldüğü gibi veri küme üzerinde çoklu kopyalar halinde farklı düğümlerde tutulmaktadır. Düğümdeki verinin zarar görmesi durumunda diğer kopyaları kullanılmaktadır. Verinin dağıtık olarak analiz edilebilmesi için Map-Reduce iskeletleri (framework) kullanılmaktadır. Bu iskelete ile dağıtık analiz programları verimli olarak geliştirilebilmektedir




Şekil 1: HDFS ( Hadoop Distributed File System), Hadoop Dağıtık Dosya Sistemi

Bu yapıda herhangi türde veri saklanabilmektedir. Verinin farklı düğümlerde kopyaları tutulması ile düğüm bozulması durumlarında ki veri kaybı engellenmektedir. Şekil 1'de görüldüğü gibi veri master – slave mantığı ile depolanmaktadır. Veri standart olarak 512MB bloklar halinde tutulabilmektedir. İhtiyaca göre blok büyüklükleri değiştirilebilmektedir.

Blok kaybının engellenmesi için her blok üç kopya olarak çoklanarak her biri farklı düğümlerde tutulmaktadır. Bu dosya sistemindeki verinin analizi için Map Reduce denilen dağıtık programlama metotları kullanılmaktadır. Koordinatör düğümler, işin tüm düğümlerde hesaplanıp sonucun kullanıcıya dönülmesini sağlamaktadır. Bu yapı ile çok büyük ölçekte veriler analiz edilebilmektedir. Yahoo tarafından yapılan testlere göre Hadoop 4000 düğüme kadar [40] verimli olarak çalıştığı ifade edilmektedir. Büyük hacim ve çeşitlilikte verinin analizine imkân veren dağıtık dosya sistemleri disk temelli çalıştığından veri analizi çok yavaş yapılmaktadır.

Veri analizinin daha hızlı yapılabilmesi için büyük veriye hızlı erişimin sağlanması gerekmektedir. Büyük veriye erişim dağıtık dosya sistemleri ile mümkün olmaktadır. Yerellik ilkeleri esas alınarak geliştirilen metotlarda güncel veriler bellekte tutulmakta ve bellek-disk arası verimli eşleme ile veriye hızlı erişim sağlanmaktadır. Bu dosya sistemleri ile büyük hacim, hız ve çeşitliliğe sahip veriler gerçek zamanlı olarak analiz edilebilmektedir. Büyük veri analizine gerçek zamanlı erişim sağlayan

dağıtık dosya sistemleri kullanılarak birçok farklı alanda ürün geliştirilebilmektedir. Çeşitliliğin fazla olması uyumlu çalışma problemlerine yol açmaktadır. Bu ürünlerden uygun olanlarının entegresi ile Büyük Veri Analiz platformları geliştirilebilmektedir. Ayrıca bu ürünler esas alınarak yeni ürünler geliştirilebilmektedir.

Dağıtık mimari kritik verilerin analizleri ve depolanması için kullanılabilecektir. Verinin güvenli olarak depolanabilmesi için sistem özel ağ (Private Network) olarak tasarlanacaktır. Sisteme dışardan ve mobil cihazlardan erişim için Sanal Özel Ağlar – ( Virtual Private Network - VPN ) kullanılacaktır. İhtiyaca göre özelleştirilecek olan VPN tünelleri ile sisteme verimli erişim sağlanmaktadır. Tünel protokollerinin özelleştirilmesi ile istenen hızlar sağlanabilerek ve güvenli erişim gerçekleştirilmektedir.

Bu yapı ile büyük veri analizleri için en çok ihtiyaç duyulan özelliklerin bir arada bulunduğu Dağıtık Veri Analiz Sistemi geliştirilmekte sosyal medya verilerinin hızlı analizlerine imkân sağlamaktadır. Geliştirilen dağıtık analitik sistemi uygun paketlerin bütünleştirilmesi ile aşağıda belirtilen temel işlevler için kullanılabilmektedir.

- **SQL Sorgulama Ara Yüzleri:** Yapısal sorgulama dilleri (SQL – Structural Querying Language) ilişkisel veri tabanlarında ACID özelliklerine göre standart formatta tutulan verilerin sorgulanmasına imkân sağlanmaktadır. Fakat tablolardaki satır ve sütun sayıları büyüdüğünde bu yapıların kullanımı mümkün olmamaktadır. Bu platform sayesinde SQL Like denilen, SQL e benzer ara yüzler ile büyük tablolar oluşturulabilmekte ve veriler hızlı sorgulanabilmektedir.

- **Çizge Analizleri (Graph Computation):** Çizge analizleri nesneler arasında ilişkilerin tanımlanıp, birbirleri ile olan mümkün fakat bilinmeyen bağlantıların çıkarılmasını sağlamaktadırlar. Sosyal medyada kişiler ve nesneler arasındaki ilişkiler, şebeke bağlantıları için en uygun seçenekler, coğrafi konumlar arası erişim için en uygun yolun bulunabilmesi bu metotlar ile sağlanabilmektedir. Bu analizler veri yoğun ve hesaplama yoğun olabilmektedir. Dağıtık veri analitik platformu sayesinde gerekli hesaplamalar hızlı olarak yapılabilmektedir.

- **Makine Öğrenmesi Ara yüzleri (Machine Learning):** Makine öğrenmesi metotları ile otomatik olarak öğrenip, kendisini geliştiren sistemler tasarlanabilmektedir. Metotlar öğreticili ve öğreticisiz olarak geliştirilebilmektedir. Öğreticili metotlarda örnek veriler ile sistemin öğrenmesi sağlanıp yeni gelen veriler ile sistemin tecrübeye dayalı olarak kabiliyetleri geliştirilmektedir. Öğreticisiz metotlar daha çok olasılıksal metotlara dayanmaktadır. Örnek verilerin elde edilemediği durumlarda sistem olasılıkları değerlendirerek kendisini geliştirmektedir. Bu metotlar özellikler sınıflama ve tahmin uygulamalarında kullanılmaktadır. Görüntü ve video içerisinde nesne, örüntü tanıma, metin analizlerinde kullanılabilmektedir. Ayrıca geleceğe yönelik tahmin uygulamalarının temellerini oluşturmaktadır.

- **Akan Veri İşleme (Stream Data Processing):** Akan veri işleme metotları akan görüntü, resim, metin, dijital sinyal gibi farklı kaynaklardan gelen verilerin paralel olarak işlenmesine olanak sağlarlar. Bant genişliğinin yüksek olduğu ve birçok kaynaktan gelen verinin işlenmesi gerektiği durumlarda veriye hızlı erişim ile hızlı analizler yapılabilmesine olanak sağlanabilmektedir. Yüksek boyutlara Multispektral, hiperspektral, ultraspektral görüntülerin ve videoların paralel olarak hızlı işlenmesine olanak sağlanabilmektedir. Ayrıca yüksek başarımlı hesaplama gerektiren karmaşık vaka analizleri (CEP - Complex Event Processing) için hızlı hesaplama platformları sağlanabilmektedir.

- **Gerçek Zamanlı Dağıtık Hesaplama (Real Time Distributed Computing):** Gerçek zamanlı sistemler verinin belirli zamanda işlenmesi ve sonucun döndürülmesini garanti ederler. Zamanlamanın kritik olduğu takip sistemlerinde (surveillance) yoğun olarak kullanılmaktadırlar. Bu sistemler sayesinde büyük miktarda veriler gerçek zamanlı olarak işlenebilmektedir. Büyük hacim ve çeşitlilikteki verilerin hızlı olarak analiz edilip sonuçların belirli bir aralıkta dönülebilmesi için dağıtık analitik platformlarına ihtiyaç duyulmaktadır. Dağıtık mimari ile tasarlanan karar destek sistemleri kritik verileri veri merkezinde analiz ederek kullanıcıya maksimum hızda cevap dönebilecektir. Gerekli durumlarda veri dağıtık olarak işlenip, sonuçlar merkeze aktarılabilecektir. Verinin belirlenen kısımlarının merkeze aktarılmadan kaynak üzerinde güvenli olarak işlenebilmesi için TrustZone mimarileri kullanılacaktır. TrustZone [41] güvenlik kritik işlemler için donanımsal olarak ayrılmış güvenli bölgelerde işlemleri gerçekleştirmektedir. Kullanıcı ihtiyacına göre bu işlemler Trusted Real Time Operating Systems (TRTOS) ile yapılıp gerçek zamanlı güvenli hesaplama sağlanmaktadır.

Güncel teknolojilerin hızlı gelişimi ile veri hızı, hacmi ve çeşitliliği artış göstermektedir. Biriken verilere anlık olarak erişim ve karar desteği sağlanması mevcut teknolojiler ile mümkün olmamaktadır. Güncel sistemlerde bu verilere bulut mimarileri ile erişim sağlanıp web tabanlı çözümler sağlanması tercih edilmektedir. Ancak güvenlik kritik, veri yoğun ve işlemci yoğun sistemlerde internet tabanlı mimariler performans ve güvenlik problemlerine yol açmaktadırlar.

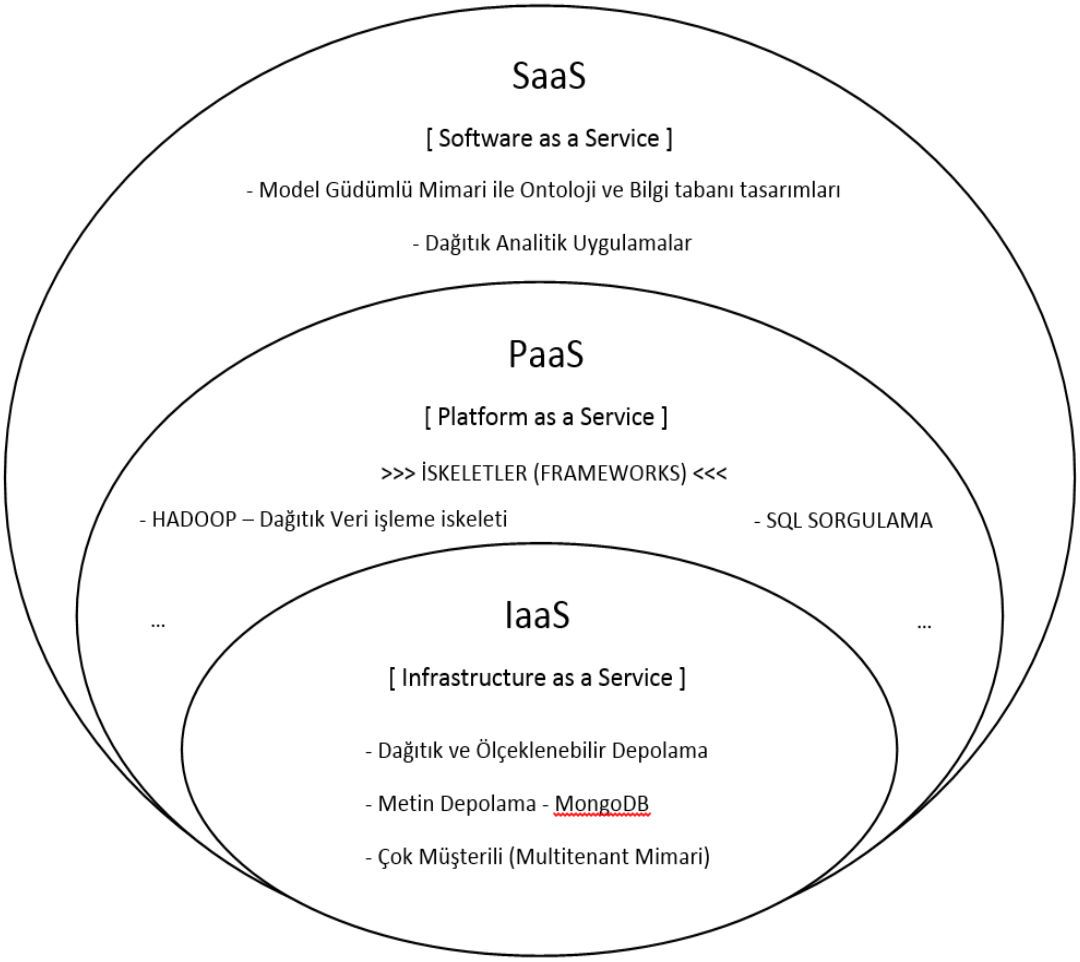
Bu çalışmada veri yoğun, işlemci yoğun uygulamalar için dağıtık analitik sistem geliştirilmektedir. Bu sistemde veri yoğun ve işlemci yoğun uygulamalar için dağıtık dosya sistemlerinin kullanımı ile performans iyileştirilmeleri yapılmıştır. Tasarlanmış olan tek düğümlü ve çok düğümlü sistemlerde performans iyileştirmeleri gözlemlenmiştir. Dağıtık mimarinin dağıtık dosya sistemleriyle tasarlanması ile hızlı sonuçlar elde edilebileceği gözlemlenmiştir.

Mikroblog metin analitiği için özelleştirilmiş sistemde farklı algoritmaları performans ve doğruluk değerlendirmeleri yapılmıştır. Mikroblog metin analitiği için skorlama algoritmasını k-means kümeleme algoritmasına göre daha hızlı çalıştığı gözlemlenmiştir. Geliştirilen dağıtık analitik mimari sayesinde büyük verinin hızlı sorgulanmasına imkân sağlanmaktadır. Dağıtık veri analitik sisteminde analitik uygulamalar için jenerik ve ölçeklenebilir depolama katmanları sağlanabilmektedir.

Veri modelleri oluşturulması ve uygulamaların birbirleri ile etkileşimleri uygulama katmanlarında sağlanmaktadır. Bu soyutlamalar ile uygulamaların çok-müşterili (multi-tenant) çalışması sağlanabilmektedir. Veri tabanına erişim uygulama katmanında soyutlamamaktadır. Dağıtık dosya sistemlerinin ölçeklenebilir otomatik düğüm ekleme çıkarma özellikleri sayesinde donanımlar maksimum verimlilikte kullanılmakta ve ölçekleme minimum donanım ve zaman maliyeti ile yapılabilmektedir.

## 1.1 Önerilen Çözüm Ve Sistem Alt Yapısı

Geliştirilen sistem bileşenleri açık kaynak olarak geliştirilmektedir. Geliştirilen bu ürünlerin uygun şekilde entegresi ve geliştirilmeleri ile hatasız çalışan platformlar üretilebilmektedir. Dağıtık dosya sistemi HDFS üzerine, SQL analizleri için Shark, Akan Veri (Stream Data) analizleri için Spark, Çizge (Graph) analizleri için GraphX, Makine Öğrenmesi analizleri için MLBase, Gerçek Zamanlı (Real Time) veri analizleri için Storm vb. araçlar geliştirilmektedir. Dağıtık analitik sistem geliştirmesi kapsamında ürünler kullanılıp gerekli yerlerde geliştirmeler yapılarak amaca yönelik platformlar geliştirilebilmektedir.




Şekil 2: Dağıtık Analitik Sistem Mimarisi

Dağıtık veri analiz sistemi öncellikli olarak veriye hızlı, ölçeklenebilir erişimi sağlamayı hedeflemektedir. Şekil 2'de görüldüğü gibi veri, disk üzerinde IaaS katmanında tutularak veri kaybı riskleri minimize edilmektedir. Sistem model güdümlü mühendislik yaklaşımı ile planlandığından, önce uygulama modelleri kullanım senaryolarına göre oluşturulmakta ve buna göre sistem tasarımları sağlanmaktadır. Dağıtık dosya sistemlerinin ölçeklenebilir yapısı sayesinde verinin düğümler üzerinde yönetimi ve ihtiyaca göre yeni düğümler eklenmesi minimum maliyetle yapılmaktadır.

Bu mekanizma ile bütünleşmiş geliştirilen hesaplama kütüphaneleri (computation engine) ile hızlı analitik kabiliyetler kazanılmaktadır. Bu analitik kabiliyetler ile amaca yönelik uygulamalar geliştirilebilmektedir.

Öncelikli olarak, küme üzerinde mikroblog metinlerin hızlı analizleri için metin analitiği uygulamaları geliştirilmektedir. Metinlerin küme üzerinde depolama işlemleri MongoDB üzerinde yapılmakta ve dağıtık olarak sorgulama için Map-Reduce iskeleti ve HDFS (Hadoop Distributed File System) kullanılmaktadır.

Sistem üzerinde, büyük ölçekli veri yoğun ve işlemci yoğun gerçek zamanlı ve akan veri analizleri dağıtık olarak yapılabilmektedir. Bu işlemler için platform üzerinde uygun iskeletler (framework) entegre edilerek kullanılabilmektedir. Dağıtık hesaplama için verinin bazı kısımlarını kaynakta işleyip sonuçlarını veya ilgili kısımlarının merkeze aktarmak gerekmektedir. Bu durumda güvenli hesaplama için TrustZone teknolojileri kullanılacaktır. Uygulama ihtiyacına göre OBC (On Board Computer), SBC (Single Board Computer) tercih edilebilmektedir.

### 1.3 Tezin Organizasyonu

Tez organizasyonu şu şekilde düzenlenmiştir; ikinci kısımda dağıtık sistemler hakkında ayrıntılı bilgi verilmektedir. Dağıtık dosya sistemlerinin temel özellikleri açıklanmaktadır. Dağıtık dosya sistemleri üzerinde dağıtık programlama için kullanılan map-reduce iskeletleri ve kullanımları özetlenmiştir. Kart üzerinde işleme ve gerçek zamanlı sistemler hakkında kısaca bilgi verilmiş ve sistemin bu alanlarda kullanımları hakkında açıklama yapılmıştır. Genel ve özel bulut mimarileri ayrıntılı olarak açıklandıktan sonra bu mimariler üzerinde dağıtık analitik platformu kullanımları hakkında bilgi verilmektedir. Bu platformların kullanım alanı olarak analiz ve analitik kavramları açıklanıp dağıtık analitik sistemde görsel analitik ve öneri sistemleri kullanımı özetlenmiştir.

Tezin üçüncü kısmında büyük veri analizi ve hızlı analiz sistemleri hakkında ayrıntılı bilgi verilmektedir. Büyük veri analizi kavramları ve kullanım alanları ayrıntılı olarak açıklanmaktadır. Büyük veri üzerinde hızlı analitik kavramları ayrıntılı olarak anlatıldıktan sonra büyük veri sistem entegrasyonları ve büyük veri kaynaklarının dinamik yönetimi hakkında bilgi verilmektedir.

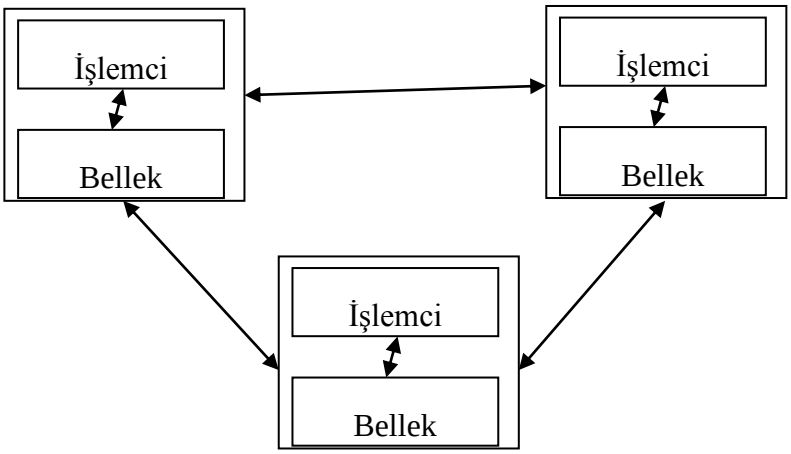
Tezin dördüncü kısmında geliştirilen dağıtık analitik sistemin uygulama alanı olarak mikroblog metinler üzerinde görüş madenciliği ayrıntılı olarak anlatılmaktadır. Bu kısımda kullanılan veri kümesi hakkında bilgi verilip, dağıtık skorlama ve dağıtık kümeleme algoritmaları ayrıntılı olarak anlatılmaktadır. Dördüncü kısmın sonunda dağıtık analitik sistemin metin madenciliğinde nasıl kullanıldığı sistemin özellikleri belirtilerek anlatılmaktadır.

Tezin beşinci kısmında sistem ve geliştirilen uygulama ve algoritmaları hakkında değerlendirmeler sunulmaktadır. Tezin son kısmı altıncı kısımda çalışma hakkında genel olarak özet bilgiler verilmekte ve gelecek çalışmalar ve potansiyeller anlatılmaktadır.

## 2. İLGİLİ ÇALIŞMALAR

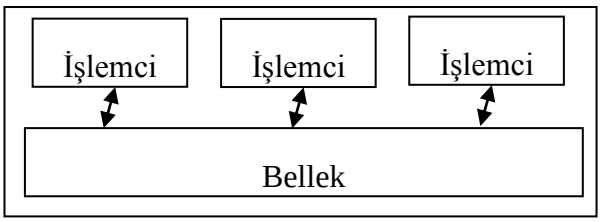
### 2.1. Dağıtık Sistemler

Dağıtık sistemler birbirlerinden fiziksel olarak ayrı konumlarda bulunan bilgisayarların mesaj iletişim protokolleri ile birbirleri ile etkileşimli olmasının sağlanması ile kurulmuş olan sistemlerdir. Sistemdeki düğümler aynı görevi farklı işlemci ve bellek mimarileri ile işleyebilir ve koordinasyonlarını kümeyi koordine eden ara katmanlarla sağlayabilirler. Bu sistemler üzerinde çalışan programlara dağıtık programlar, bu yazılımların geliştirilmesine dağıtık programlama denmektedir. Dağıtık algoritmalar farklı bellekler ve işlemcilerle ve belleklerle çalışırlar. Her düğümün kendisine ait bellek ve işlemcisi bulunmaktadır.




Şekil 3: Dağıtık Sistem

Şekil 3'te görüldüğü gibi sistemdeki her düğümde işlemci ve bellek bulunmaktadır. Her düğümde işlemin bir parçası bağımsız olarak işlenmektedir. Her işlemci yalnız kendi bellek alanına erişebilmektedir. Gerekli durumlarda düğümler birbirleri ile iletişim protokolleri ile iletişim kurmaktadırlar.




Şekil 4: Paralel Sistem

Şekil 4'te görüldüğü gibi paralel sistemde ortak bir bellek alanı bulunmaktadır. Her işlem ortak bellek alanına erişim sağlayabilmektedir. Paralel algoritmalar çok işlemcilerin paralelleştirilmiş algoritmaların ortak bellek alanını kullanımı ile yüksek başarımlı hesaplamalar gerçekleştirebilmektedir. Dağıtık sistemlerde ise bu algoritmalar dağıtık olarak farklı bellek alanlarında işlenmektedir.

### 2.1.1 Dosya Sistemleri

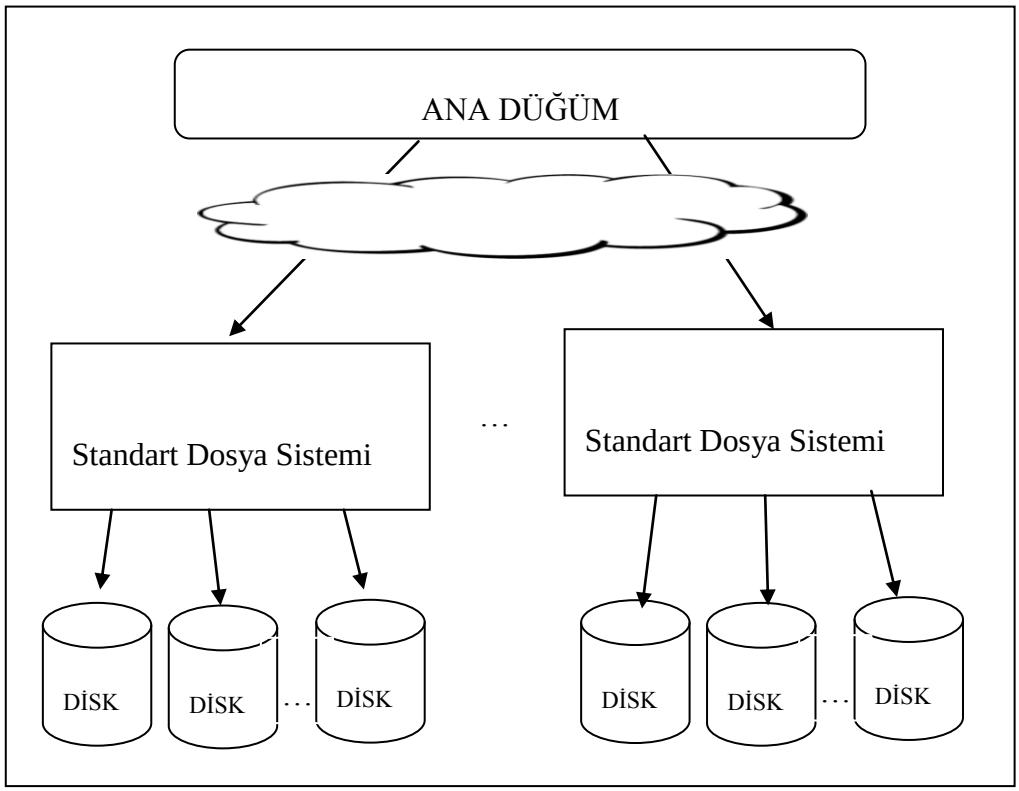
Dosya sistemleri işletim sistemleri tarafından kullanılan, verinin disk üzerinde nerelerde tutulacağını adresleyen mekanizmalardır. Bu sayede veriler bloklar halinde saklanıp, verinin gerekli kısmına ihtiyaç duyulduğu zaman erişim sağlanabilmektedir. Her cihazın donanımsal mimarisine göre özelleştirilebilen ve özelleştirilmesi gereken bu sistemler amaca yönelik olarak tasarlanmaktadırlar. Cihaz mimarilerine göre özelleştirilmiş dosya sistemleri ile cihaz içerisinde dosya yönetimleri sağlanabilmektedir. Veri tabanı dosya sistemleri kavramları ile hiyerarşik dosya depolama metotlarına ek olarak, dosya tipi, konusu, yazarı, karakteri gibi bilgiler saklanarak veri yönetimi daha verimli yapılabilmektedir. İşlemsel dosya sistemleri kavramları ile işlemlerin durumları ve geçmişleri takip edilerek işlemler daha verimli yönetilebilmektedir. Ağ dosya sistemleri yapıları ile ağ protokolleri kullanımı sayesinde dosyaların birden çok bilgisayarda tutulması ve ağ üzerinden erişimleri sağlanmaktadır.

Paylaşımlı disk dosya sistemleri (SAN-Storage Area Network) birden çok bilgisayara blok düzeyinde direk disk erişimi sağlamaktadırlar. Paylaşımlı disk dosya sistemleri mimari olarak iki temel türdedir. Tam dağıtık mimaride dosya bilgileri tüm sunucularda tutulur, her sunucu dosya bilgilerine kendi üzerinden erişir. Merkezi olarak metadata sunucusu kullanan mimarilerde dosya bilgilerine metadata sunucuları üzerinden erişim sağlanır.

Dağıtık dosya sistemlerinde ise dosyalara blok düzeyinde erişim sağlanmaz, bunun yerine ağ protokolleri ile dosya erişim ve kullanımları sağlanmaktadır. Tasarlanana ağ protokolleri ile dosya erişim kuralları belirlenmekte, kullanıcılara bu kurallara göre erişim yetkileri tanımlanabilmektedir. Bunlar ağ dosya sistemleri olarak ta bilinmektedirler. NFS (Network File System) yapıları ile herhangi bir kullanıcı (client) kendi üzerindeki adres blokları sayesinde dosya üzerinde olmasa bile dosya varmış gibi davranıp, işlem yapılmasını sağlamaktadırlar. Dosyalara ihtiyaç olduğu durumda ağ üzerinden tanımlanmış ağ protokolleri ile dosya transfer işlemlerini gerçekleştirmektedirler. Ağ hızına göre dosya aktarımı gerçekleştirilmekte, kaynak üzerinde ilgili işlevler tanımlandıktan sonra sonuçlar ağ üzerinden aktarılmaktadır. Güvenli hesaplamaların gerektiği durumlar için dosyalar üzerinde şifreleme yapılmakta ve güvenli ağ transfer işlemleri yapılmaktadır.

Dağıtık dosya sistemlerinin mimari yapısı sayesinde kullanıcılar dosyaların kümede dağıtılmış olduğunun farkına varmadan, yerel düğümlerine erişir gibi ağdaki dosyalarına erişebilmektedirler. Şekil 5'te görüldüğü gibi ana düğüme erişim sağlayarak küme üzerinde bulunan tüm disklere erişilebilmektedir. Ana düğüm diğer düğümlerin isim ve adres bilgilerini hafızasında tutar ve gerekli yönetim işlevlerini

gerçekleştirir. Farklı dağıtık dosya sistemleri ile de bütünleşik olarak farklı dağıtık kümelerin bütünleşik olarak çalışması sağlanabilmektedir. Her disk kendi yerel dosya sistemleri ile düğüm üzerinde yönetebilir ve dağıtık dosya sistemleri vasıtası ile küme üzerinde veri okuma yazma işlemlerini gerçekleştirebilmektedir.




Şekil 5: Dağıtık Dosya Sistemi Mimarisi

Şekil 5'te görüldüğü gibi, farklı dosya sistemleri ile bulut yapısı oluşturulabilmektedir. Bu yapı ile farklı dosya sistemlerinin birbirleri ile bütünleşik çalışması ve farklı sistemlerin uyumlu olarak veri paylaşımı gerçekleştirmeleri sağlanmaktadır. Bulut yapısının tasarımı kullanıcı taleplerine göre değişim göstermektedir. Kullanıcılar veriyi kendi bünyesinde tutmak istedikleri durumlarda özel (private) bulutlar tasarlanmakta ve kullanıcının veri merkezindeki merkezi sunucular üzerinde veri okuma, yazma ve düzenleme işlevleri gerçekleştirilir. Kullanıcı depolama hizmetlerini dış bulutlardan alması gerektiği durumlarda genel (public) bulut mimarileri kullanılmaktadır. Bu mimarilerde veriler internet üzerinden gerekli durumlarda transfer edilmekte ve ilgili işlevler internet protokolleri ile gerçekleştirilmektedir. İnternet temelli sistemler güvenlik kritik olmayan durumlarda tercih edilmekte ve internet hız sınırları ve kısıtları ile işlevler gerçekleştirilebilmektedir.

### 2.1.2 Map-Reduce İskeletleri ( Framework )

Map-Reduce programlama modeli büyük veri kümelerini dağıtık olarak işleyebilmek için geliştirilmiş standart donanımlarla çalışabilen çatı yazılımlardır. Dağıtık hesaplamaları standart donanımlar ile yapabilmeleri süper bilgisayar işlevlerini görmeye başlamıştır. Hızlı ve verimli yapısı ile son dönemde popüler hale gelmişlerdir. Dağıtık dosya sistemleri üzerinde çalıştıklarından ölçekleme ve düğüm değişimleri minimum zaman ve donanım maliyeti ile yapılabilmektedir. Kullanıcılar key/value ikililerini işlemek ve key/value ikilisi kümelerini oluşturmak için map fonksiyonu tanımlarlar. Her düğüm map fonksiyonunu kendi yerel verisi üzerinde uygular ve çıktıyı geçici bir depolama alanına koordinatör düğüm tarafından koordine edilmek üzere yazar. Reduce aşamasında worker düğümler key ler ile alakalı her çıktı grubunu paralel olarak işleyip birleştirirler. Bu modelde paralelleştirme ve kümeye dağıtıp, toplama işlemleri otomatik olarak gerçekleştirilmektedir.

**Map-Reduce Algoritması**

```
function map (key, value)

    valueList [];
    foreach ( v in valueList )
        if ( listTest == True )
            valueList.append ( ( key, v ) )

     return valueList;

function reduce ( key, valuesList )

    result = 0;
    foreach ( v in  valuesList )
        result += x

    return (key, result)
```

Algoritma 1: Map-Reduce Algoritması [42].

Map algoritmasında her girdi değer (value) için referans (key) değeri tanımlanır. Kullanıcı tarafından her girdi değeri üzerinde uygulanmak üzere fonksiyon tanımlanır. Girdi amaca göre işlenip ayrıştırılması (parsing) gerekebilmektedir. Bu key/value değerleri ile key/value ikilileri listesi üretilir.

Reduce aşamasında key/value ikilileri işlenmektedir. İkililerin sıralaması key değerlerine göre yapılmaktadır. Her key/value ikilisinin iteratif olarak işlenmesi ile sonuçlar toplanıp, sonuç key/value ikilileri olarak dönülür.

### 2.1.3 Kart Üzerinde İşleme (On-Board Processing) Ve Tek Kartlı Bilgisayarlar (Single Board Bilgisayarlar)

Kart üzerinde veri işleme (on-board processing) herhangi bir sistem üzerindeki verinin toplanmasını, aktarılmasını, depolanmasını, sıkıştırılmasını ve ilgili veri merkezleri ile paylaşımlarını kapsamaktadır. Güncel sistemlerde yüksek miktarda veriler üretilmektedir. Bunların hepsinin veri merkezleri ile paylaşılması ağ trafiğini gereksiz olarak meşgul etmektedirler. Sinyal işleme, görüntü işleme uygulamalarında verinin bir kısmının kart üzerinde işlenip, anlamlı verinin veri merkezi ile paylaşımı ile ağ trafiği verimli olarak düzenlenebilmektedir.

Tek kartlı bilgisayarlar işlemci, bellek ve girdi/çıktı birimleri ile tek kart üzerinde tasarlanmış özel amaçlı bilgisayarlardır. Tek kartlı bilgisayarlar gömülü bilgisayar kontrol sistemlerinde yaygın olarak kullanılmaktadırlar. Özellikle görev kritik uygulamalarda (mission ciritical) zaman ve fonksiyonel kısıtları sağlamak için yoğun olarak amaca yönelik tasarlanmaktadırlar. Özelleştirilmiş analitik bulut mimarileri bu sistemler ile anlık olarak veri alışverişi gerçekleştirebilmektedirler. Veri transferlerinin verimli olarak yapılabilmesi için analitik işlemler için ihtiyaç duyulan anlamlı verilerin kaynakta tespiti ve transferi tek kaynaklı bilgisayarlar ile verimli olarak yapılabilmektedir. Hesaplama yoğun sistemlerde işlemcileri desteklemek üzere belirlenen analitik fonksiyonları gerçekleştirmek üzere tek kartlı bilgisayarlar sisteme entegre edilerek verimlilikleri arttırılabilmektedir.

### 2.1.4 Gerçek Zamanlı Sistemler

Gerçek zamanlı sistemler (Real Time Systems) belirlenen görevlerin (mission, task) belirlenen zaman aralıklarında (deadline) çalışma prensipleri ile çalışırlar. Komuta kontrol sistemleri, hava trafik kontrol sistemleri, multimedya sistemleri gibi sistemler gerçek zamanlı olarak çalışmaktadırlar. Gerçek zamanlı sistemler belirlenen görevlerin gerçekleştirilmediği durumdaki kritikliklerine göre sınıflandırılırlar. Görevin gerçekleştirilmediği durumda felaket senaryoları oluşuyorsa (hard-real time systems), büyük kayıplara yol açılıyorsa (soft-real time systems) olarak adlandırılmaktadırlar. Komuta kontrol sistemleri (hard-real time systems), hava trafik kontrol sistemleri (soft-real time systems) olarak adlandırılmaktadırlar.

Gerçek zamanlı sistemlerin zamanlaması statik ve dinamik olarak yapılabilmektedir. Statik zamanlamalar derleme sırasında (compile time) ve sistem pasifken (off-line) yapılmaktadır. Dinamik zamanlama sistem aktifken yapılmaktadır ve zamanlama testleri ile görevlerin belirlenen zaman aralıklarında yapılıp yapılmadığını kontrol etmektedirler. Güncel teknolojiler ile geliştirilen özelleştirilmiş analitik bulut

mimarileri bu sistemlerin karar destek mekanizmalarını güçlendirmek için kullanılmaktadırlar. Sisteme gerçek zamanlı olarak analitik kararlar sağlanabilmektedir. Sisteme bu kararlar ile kendi durumunda güncelleme ve durumlar arasında geçişler gerçekleştirebilmektedir.

## 2.2. Public Ve Private Cloud Mimarileri

Bulut bilişim ağ tarafından paylaşılan ve her yerden erişim sağlanabilen konfigüre edilebilir, hesaplama aygıtlarından oluşan hesaplama modelidir. Tasarlanan ağ türüne göre mimari model ve cihaz seçimleri gerçekleştirilmektedir. Bulut mimarisi sayesinde kaynaklar ağdaki tüm cihazlar tarafından paylaşımlı olarak kullanılabilmektedir. Bulut mimarileri temelde ön uç (front-end) arka uç (back-end) bileşenlerinden oluşmaktadır. Ön uç bileşenleri kullanıcılar için servis ve uygulamalara erişim sağlamaktadırlar. Arka uç bileşenleri depolama ve sunucu hizmetlerini ağ üzerinden sağlamaktadırlar. Uygulama ve servisler belirlenen politikalara göre bu kaynaklara erişim sağlayarak kullanıcılara hizmet verebilmektedir. Bulut mimarileri sunum şekline göre genel (public) ve özel (private) olarak ikiye ayrılmaktadırlar. Genel bulut mimarileri internet üzerinden hizmet vermektedirler.

Genel bulutlarda kullandıkça öde politikaları uygulanmaktadır. Kullanıcılar kullandıkları işlemci, depolama ve ağ hizmetlerinden birim fiyatlar üzerinden ücretlendirilmektedirler. Özel (private) bulut mimarileri kullanıcılara dedike edilerek (on-premise) intranet mantığı ile geliştirilmektedirler. Özel bulutlar kullanıcıların kendi veri merkezlerini oluşturup kendi iç ağları üzerinden kendi kullanıcılarına erişim sağlayarak hizmet vermelerine imkân sağlamaktadırlar. Genel ve özel bulut mimarilerinin entegresi ile hibrit bulut mimarileri de oluşturulabilmektedir. Bu mimari ile hem internet temelli servisler kullanılabilmekte hem de intranet ağı tasarımları ile verinin daha güvenli erişimi sağlanabilmektedir. Bulut mimarileri alt yapı, platform ve uygulama katmanları olarak soyutlanmaktadırlar.

### 2.2.1 IaaS (Infrastructure As a Service)

Alt yapı katman fiziksel donanımları içermektedir. Sanallaştırma, sunucu, depolama, ağ yönetimi, sistem yönetimi bu katmanda sağlanmaktadırlar. Kullanıcılar genel (public) bulut mimarileri ile bu katmanı internet üzerinden kiralayarak veri merkezi maliyetlerini azaltabilmektedirler. Özel (private) bulut mimarileri ile de kullanıcılar bu hizmeti kendilerinde dedike (on-premise) olarak alıp, kendi kullanıcılarına daha güvenli intranet hizmeti sağlayabilmektedirler.

### 2.2.2 PaaS (Platform As a Service)

Platform katmanında kullanıcılara uygulamaları için platform ve veri tabanı servisleri sağlanmaktadır. Uygulamaların ihtiyaçlarına göre platformlar özelleştirilmekte ve platform hizmetler genel (public) veya özel (private) bulut mimarileri üzerinden sağlanabilmektedir.

### 2.2.3 SaaS (Software As a Service)

Uygulama katmanı uygulamaların servis olarak sunulmasını sağlayarak kullanıcıların yazılım kurma maliyetlerinden kurtarmayı hedeflerler. Uygulamalar bulut üzerinde kurulur kullanıcılar (client) bulut üzerinden uygulamayı kullanırlar. Uygulamalar genel (public) ve özel (private) bulut mimarileri üzerinde amaca göre tasarlanarak kullanılabilmektedirler. Örneğin; güvenlik kritik uygulamalar için internet temelli servisler güvenlik riski oluşturduğundan, intranet ağları üzerinden özel (private) bulut mimarisi ile sunulması ve kullanılması tercih edilmektedir.

### 2.2.4 Bellek Merkezli Dağıtık Analitik Platformu

Veri analizi için kapsamlı platformlar sağlanmaktadır. Ham verilerin yüklenip, temizlenip, amaca göre özelleştirildikleri çok çeşitli teknolojiler bulunmaktadır. Gelişen teknolojilerle paketlerin bütünleşmiş edilerek analitik fonksiyonlarının platform olarak sunulması sağlanabilmektedir. Bu platformlarında depolama işlemleri için bellek merkezli dosya sistemleri (Tachyon), SQL sorgulama işlemleri için Shark, çizge hesaplamaları için GraphX, makine öğrenmesi için MLib, akan veri analizleri için Storm, DStreams, dağıtık hesaplama için Spark araçları bütünleşmiş edilerek ve özleştirilerek amaca yönelik Bellek Merkezli Analitik Platformları geliştirilmektedir.

## 2.3. Analiz Ve Analitik

### 2.3.1 Veri Analizi Ve Analitiği

Veri bilginin işlenmemiş halidir, kimse için anlam ifade etmemektedir. Verinin anlamlı olabilmesi için veri gerekli kaynaklardan toplanır ve amaca uygun formatta bilgiye çevrilir. Bu bilgilerde daha sonra yetenekleri geliştirmek için kullanılabilmektedir. Bilginin herhangi bir sorunu çözmek için kullanılabilen, kabiliyet olarak kullanılabilen kısmı bilgi, yetenek (knowledge) olarak adlandırılmaktadır. Veri analizi ise bir süreçtir. Veri uygun kaynaklardan toplanır, verideki kirlilikler temizlenir, karakteristiği anlaşılır ve veri için uygun modeller uyarlanır.

Veri modellendikten sonra karar destek sistemleri için anlamlı olmaktadır. Karar destek sistemleri modellenmiş formattaki verileri karar destek sistemleri süreçlerinde kullanmaktadırlar. Analiz edilmiş, modellenmiş ve temizlenmiş veri karar destek süreçleri için kullanılabilmektedir. Sistemin analitik kabiliyetlerini geliştirmek için bilgiler üzerine analitik metotlar uygulanmaktadır. Güncel veriler üzerinde analitik metotları uygulayabilmek için matematik, bilgisayara bilimleri ve istatistiksel metotlar bir arada bütünleşmiş edilerek kullanılabilmektedir. Bu bilgiler üzerinde örüntü algılama işlemlerimde uygulanarak veri üzerinde tahmin modelleri geliştirilmektedir. Bu şekilde yetenekler geliştirilmektedir.

### 2.3.2 Veri Görselleştirme Ve Görsel Analitik

Analiz ve analitik sonuçları kullanıcıya görsel bileşenlerle sunulmaktadır. Tablolar, çizgeler, grafikler, haritalar vb. bileşenler görselleştirme amaçlı kullanılmaktadırlar. Bu bileşenler aracılığı ile kullanıcılarla görsel iletişim metotları geliştirilmektedir. Verinin bilgi ve yeteneğe dönüşüm süreci bu bileşenlerle gerçekleştirilmektedir. Bilgisayarlar ve otomasyon yöntemleri yeterli olmadığı durumlarda insan gücünden yararlanılmaktadır. Bu süreçte insan bilgisayar etkileşimi metotları kritik hale gelmektedir. Hangi süreçlerin otomatize edileceği, hangilerinin insan gücü ile yapılacağı insan-bilgisayar etkileşimi konusu olarak değerlendirilmektedir. Verinin miktarı, türü ve hızı her geçen gün artmaktadır.

Veriyi anlamlı hale getirmek için interdisipliner metotlar kullanılmaktadır. Keim [1], çalışmasında görsel analitik konusunda ortaya çıkan konuları ayrıntılı olarak adreslemektedir. Bu çalışmada görsel analitik metotlarının veri madenciliği ve istatistik konusunda öne çıkan çalışma konularından bir olduğu söylenmektedir.

Kullanıcılar ile bilgiler üzerinde etkileşimli analitik yapabilmek için yeni teknoloji ve kütüphaneler geliştirilmektedir. D3 [43] java script dili ile geliştirilmiş, HTML sayfaları ile uyumlu örneklerden biridir. Bu ve benzer kütüphaneler mobil, PC, dizüstü, vb. herhangi platform ile görsel analitik işlemlerinin etkileşimli olarak çok daha verimli yapılabilmesini sağlamaktadırlar.

### 2.3.3 Öneri Sistemleri

Öneri sistemleri büyük miktarda veriyi hızlı analiz ederek kullanıcı profiline uygun bilgileri kullanıcıya uygun şekilde sunarlar. Kullanıcı profillerinin belirlenebilmesi, kullanıcı için uygun seçeneklerin çıkarılabilmesi için büyük miktarda veri üzerinde analiz yapmaları gerekmektedir. Kullanıcılara anlık olarak geri dönütler sunmak için çeşitliliği fazla olan veri üzerinde dinamik olarak yapısal olmayan sorguları çalıştırmaları gerekmektedir. Bu sistemler genellikle internet üzerinden alışveriş, karar destek sistemleri, bilgisayar destekli tasarım sistemleri gibi anlık olarak hızlı kararlar verilmesini gerektiren sistemlerde kullanıldığından veri yoğun ve işlemci yoğun büyük verinin anlık olarak gerçek zamanlı analiz edilmesini gerektirirler.

Öneri sistemlerinde kullanıcı profilleri belirlerme ve veri toplama işlemleri için ajan temelli sistemler kullanılır. Öneri sistemlerinin gerekli bilgiyi elde edebilmeleri için ajanlar belirlenen sensör fonksiyonlarına göre etraflarında sürekli veri toplarlar. Toplanan veriler doğrultusunda bilgi tabanlarını güncellerler. Bilgi tabanlarına hızlı erişim ile daha dinamik ajan tasarımları gerçekleştirilebilmektedir. Akıllı ajanların etkin tasarımları ile öneri motorları daha verimli olarak çalışmaktadır. Toplanan veriye anlık olarak erişim ve ilgili veri tabanları, bilgi tabanlarının anlık olarak güncellenebilmesi için hızlı ve interaktif analitik fonksiyonlara ihtiyaç duyulmaktadır.

Bilgi kümesi içerisinden kullanıcı profiline uygun bilgi seçimlerini gerçekleştirmek ve bu ajanların etkileşimli olarak çalışabilmesi ve öneri sistemleri için etkin kararlar verilmesininin sağlanması için hız ve tutarlılık önemlidir. Öneri sistemlerinin kullanıcı ile etkileşimli olarak ara yüz üzerinden tasarlanabilmesi ve büyük veri kümlerini bilgi tabanı olarak kullanabilmeleri için bellek merkezli dağıtık analitik sistemler kullanılmaktadır. Bu sistemler sayesinde kullanıcılar interaktif olarak veri, bilgi tabanlarını tasarlayabilmektedir. Veri, bilgi tabanları için gerekli güncellemler dağıtık sistemler üzerinde anlık olarak yapılabilmektedir. Son kullanıcılar da görsel ara yüzler vasıtasıyla sistem düzeyinde güncellemeleri gerçekleştirebilmektedirler. Böylece alan uzmanları ve sistem tasarımcıları öneri sistemleri tasarımlarını verimli olarak gerçekleştirebilmekte ve büyük veri üzerinde anlık olarak karar verilebilmektedir.

# 3. BÜYÜK VERİ ANALİZİ ve HIZLI ANALİZ SİSTEMLERİ

## 3.1 Büyük Veri Analizi

Büyük veri standart sistemlerle depolanıp, yönetilip, analiz edilemeyen ve hacim, çeşitlilik, hız karakteri olan veri kümeleridir. Büyük veriler resim, video, görüntü, ses, metin vb. dosyaları bir arada bulundurur, anlamlı olarak sorgu sonuçlarının alınabilmesi için bunların hızlı analizlerini gerektirirler. Örneğin toplanmış video görüntüleri içerisinden anlık olarak kişi veya nesnelerin analizi, takibi gerekmektedir. Sorguların hızlı ve anlık olarak işlenebilmesi için dağıtık platformların kullanımı gerekmektedir.

Standart veri merkezi ve OLAP (Online Analytical Processing) küp yaklaşımları veriyi disk üzerinde depolar ve analitik işlemler için uzun süren sorgular çalıştırırlar. OLAP mantığında standardize edilmiş ve normalizasyon kuralları uygulanmış ilişkisel veri tabanı tabloları (RDBMS – Relational Database Management System) denormalizasyon işlemleri ile tek tablo haline dönüştürülürler. Bu sayede veriye daha hızlı erişim sağlanır ve analitik fonksiyonlar uygulanabilir. Ancak tablo büyüklüğü çok arttığından okuma yazma işlemleri verimsiz olarak yapılır.

OLTP (Online Transaction Processing) verisinde en çok kullanılan veriler tutulur. Veri standart ilişkisel (RDBMS – Relational Database Management System) yöntem ile tutulur. Tekrar hesaplama yapmamak için sonuçlarında saklarlar. Bu şekilde sık ihtiyaç duyulan veriye hızlı erişim sağlanır. OLAP ve OLTP yaklaşımları ile veri merkezi çözümleri geliştirilir ve hantal çözümler oluşur. Veri merkezlerindeki sorgular uzun sürede çalışır ve sorguların hızlandırılması, güncellemelerin yapılması yüksek donanım ve insan kaynağı maliyetleri ile gerçekleşir.

Büyük veri ve bellekte çalışan çözümler ile hızlı çözümler daha düşük maliyet ile sağlanabilmektedir. Dağıtık dosya sistemlerinin ölçeklenebilir yapısı sayesinde verimli ölçekleme sağlanabilmektedir. Depolama IaaS katmanında sağlanmaktadır. Bu katman üzerinde uygulama geliştirme, MVC modeli ile gerçekleştirilebilmektedir. Bean sınıflar ve DAO nesneler sayesinde SaaS katmanında uygulamaların veri erişimi sağlanmaktadır. Bu sayede sistem teknoloji bağımsız olarak modellenebilecek ve teknolojiye özel (.NET; JAVA, C++ vb.) olarak model uyarlamaları yapılabilmektedir. Büyük veri için tasarlanan dağıtık analitik sistemlerde her alan için teknoloji bağımsız bileşen modelleri oluşturup iş modelleri olarak hedef teknolojiler ile entegre edilmektedir. Bu sayede teknoloji değişimler ve uyarlamalar minimum maliyet ile sağlanabilmektedir.

### 3.2 Büyük Veri Üzerinde Hızlı Analitik

Verinin büyüklüğü ve çeşitliliği artış gösterdikçe veriyi taramak ve içerik tespitleri yapmak zorlaşmaktadır. Veriye analitik fonksiyonlar uygulanması çok zaman almaktadır. Büyük ölçekte verinin interaktif sorgulanması için yeni yöntemler geliştirilmektedir. Milla [16], coğrafi veriler üzerinde zaman serileri ile interaktif analizler yapmak için yeni yöntemler geliştirmektedir. Daha fazla istatistiksel fonksiyonlar kullanabilmek için açık kaynak istatistiksel programlama dili R kullanarak görsel analitik metotlarını uygulamaktadırlar.

Standart veri tabanları ve dosya sistemleri farklı kaynaklardan bütünleşmiş edilen bu verileri analiz edebilmek için yeterli hız ve işlem kapasitesine sahip değillerdir. Bu verileri hızlı olarak analiz edebilmek ve görsel analitik fonksiyonlarını veri üzerinde uyarlayabilmek için yeni dosya sistemleri ve depolama mekanizmaları kullanılmaktadır. Zhang ve arkadaşları [17] büyük veriyi interaktif sorgulayabilmek ve görsel analitik fonksiyonlarını büyük veri üzerinde uygulayabilmek için bellek üzerinde çalışan dosya sistemlerini değerlendirmektedirler. Zhao standart veri tabanlarının veri yoğun bilimsel uygulamalar için yetersizliklerini vurgulamaktadır. Çalışmalarında veri yoğun işlemler için FusionFS [18] dosya sistemini önermektedirler.

Zinn [19] çevresel verileri Microsoft Azure tabanlı geliştirdikleri bulut platformunda analiz etmektedirler. Akan uydu verilerini direk olarak bulut ortamında kullandıkları sanal makinelere aktarmaktadırlar. Gerçekleştirdikleri iş akışları süreçlerinde bulut depolama, katmanlama ve işleme mekanizmaları ile %130 - %160 arasında performans iyileştirmeleri elde ettiklerini vurgulamaktadırlar. Sistemde veriyi BLOB nesneler halinde tutmaktadırlar ve REST mimarisi ile kullanıcı ara yüzleri sağlamaktadırlar. Veriyi dağıtık dosya sistemlerinde depolama ise başka bir çözüm olarak önerilmektedir. Hadoop temelli analitik uygulamalar ile daha verimli sorgular yazılabileceği önerilmektedir. Örneğin Skybox Şirketi [20], görüntü işleme yeteneklerini analitik fonksiyonlar ile iyileştirdiklerini belirtmektedirler. C ve C++ dillerinde geliştirmiş oldukları bilimsel algoritmalarını JAVA işlemleri olarak Hadoop mimarisinde işlemektedirler.

Chang ve diğerleri [21], bellek merkezli iletişim mimarisini tekrar konfigüre edilebilir hesaplamada kullandıklarını söylemektedirler. Belleği çoklu bellek birimlerine bölmektedirler ve her bellek birimi için tek port atamaktadırlar. Tasarladıkları mimari ile standart mimarilerde %76 daha iyi performans aldıklarını belirtmektedirler. Beric [22], çalışmasında iki seviyeli bellek hiyerarşisine sahip alana özel bellek alt sistemi önermektedir. Mimari hesaplama yoğun ve yüksek bant

genişliği isteyen uygulamalar için geliştirilmiştir. L0 veriyi hızlı çekmek için, L1 bellek için bant genişliği ihtiyacını minimize etmek için kullanmaktadırlar.

Büyük miktarda akan veri işlemler veri yoğun ve işlemci yoğun olarak gerçekleşmektedir. Disk ve işlemciyi çok yoğun olarak kullanmaktadırlar ve işlemler paralel işleme yöntemlerine uygundurlar. Yao [23] ve diğerleri çok çekirdekli gerçek zamanlı sistemleri bellek merkezli yaklaşımlar ile zamanlamaktadırlar.

Akıllı ajanların büyük veri üzerinde bilgi tabanları oluşturması ve interaktif analitik uygulama tekniklerinin uygulanması literatürde de görüldüğü gibi standart veri tabanı yaklaşımlarıyla mümkün olmaktadır. Bu işlemler için dosya sistemleri ve depolama yaklaşımlarında güncellemeler ön görülmektedir. Dosya sistemleri ve depolama mekanizmalarının optimize edilmesi ve mimarilerin değişimi gerekliliği ön plana çıkmaktadır.

Dosya sistemleri verinin depolama katmanları üzerinde nerede tutulacağına karar verirler ve dosyanın depolanıp getirilmesi işlemlerini gerçekleştirirler. Dosyalar tek düğümlü veya dağıtık olarak çok düğümlü mimaride depolanabilirler. Sistemin kendine has ihtiyaçlarına göre dosya sistemleri mantıksa yerleştirme ve geri getirme metotlarını belirlerler. Gerçek zamanlı sistemlerde bu kurallara göre kritik uygulamalar çalıştırılabilir ve hayati önem taşıyan kararla verilebilir.

Tek düğümlü sistemler dosyaları işlemci ön belleğinde, bellekte veya disk üzerinde yönetirler. İsimlendirme kuralları ve hiyerarşik depolama metotları belirlerler. Windows, Macintosh ve UNIX sistemler dosyaları ağaç yapısında depolamaktadırlar. Dosyalar ağaç yapısı üzerinde takip edilmektedirler. Dağıtık mimaride ise, ağ erişim konuları dosyalara erişim kurallarını belirlemek için kritiklik göstermektedirler. Hitz ve diğerleri [24] ağ üzerinden dosya erişimleri için mimarilerini optimize etmektedirler. Dosyayı ağ üzerinde takip edebilmek için ağ protokolleri önem arz etmektedirler. Sistemin performansı bant genişliğine, paket yapılarına ve algoritmaların performanslarına göre değişim göstermektedirler.

Ağ temelli sistemler; TCP, UDP benzeri iletişim protokolleri kullanmaktadırlar. Düğümler arası iletişim için RPC (Remote Procedure Call) metotları genellikler kullanılmaktadır. Daha büyük ölçekli, binlerce kullanıcıya hizmet veren uygulamaları desteklemek için ağ protokolleri ve depolama/geri çağırma metotlarının uyarlanması ve geliştirilmesi gerekmektedir. Callaghan ve Brent [25, 26, 27] çalışmalarında ağ temelli dosya sistemleri üzerinde (NFS – Network File System) yapılan iyileştirmelerle geliştirilen webNFS dosya sistemleri üzerindeki iyileştirmeleri ayrıntılı olarak anlatmaktadırlar.

Ağ erişim protokolleri ve okuma/yazma işlemleri üzerindeki iyileştirmelerini ayrıntılı olarak tanımlamaktadırlar. Her okuma/yazma işlemi için bağlantıyı açıp kapatmama ve büyük dosyaları bölmeler halinde indirmeye imkân tanımaları webNFS için yaptıkları en önemli iyileştirmeler arasında yer almaktadırlar.

Ön bellekleme yönetimi dosya sistemleri konularından en önemlileri arasında yer almaktadır. Ön belleğe alınan dosyaların tekrarlı olarak kullanımı önem arz etmektedir. Andrew Dosya Sistemi [28] yerel olarak ön belleğe alınan dosyayı dağıtık hesaplama ortamlarında verimli olarak kullanmaktadır. Sunucu üzerinde bir dosya için yapılan istek üzerine dosya yerel düğümde ön belleğe alınmakta ve yerel ön bellekte tutulmaktadır. Aynı dosyanın ikinci defa istenilmesi durumunda dosya yerel ön bellek üzerinden istek yapan işleme sağlanmaktadır. Bu yaklaşım büyük ölçekli kümeler için verimli bir dağıtık hesaplama ortamı sunmaktadır.

Küme üzerindeki düğüm sayısı arttıkça dosyaların senkronize edilmesi, tekrar çağrılması, zarar gören dosyaların onarımı zorlaşmaktadır. Merkezi olarak senkronizasyon sağlanması veya dağıtık olarak kilit mekanizmalarının kullanılması kritik kararlar arasında yer almaktadır.

IBM'in büyük kümeler için kullandığı dosya sistemi [29] yapısında senkronizasyon, dosyaların tekrar çağrılması ve zarar gören dosyaların süper bilgisayarları da içeren dağıtık mimaride nasıl kurtarıldığı ayrıntılı olarak anlatılmaktadır. Peta Byte'lar düzeyinde veriyi yönetmek için pahalı çözümler bulunmaktadır. Büyük şirketler süper bilgisayarlara benzer yapıları kullanmaktadırlar. Ancak, standart donanım çözümleri çoğu durumlar için yeterli olabilmekte ve hatta daha iyi çözüm olarak sunulabilmektedir.

Profesyonel kullanıcılar standart dosta sistemleri yaklaşımlarını tekrar gözden geçirmektedirler. Google dağıtık depolama ve dağıtık uygulamalar için kullandığı kendi dosya sistemi yapısını GFS [30] açıklamakta ve kendi dağıtık depolama ve dağıtık uygulama ihtiyaçları için yeterli olduğunu belirtmektedirler. GFS tasarımı basit tutmak için tek yönetici düğüm yaklaşımını kullanmaktadırlar. Bu sayede yönetici düğüm üzerindeki görevler minimize edilmekte ve yönetici düğüm en az düzeyde meşgul edilmektedir. GFS bellek üzerinde çalışan veri yapıları sayesinde sonuçlarını bellek hızında sağlayabilmektedirler GFS büyük ölçekli, veri yoğun uygulamaları standart donanımlar ile desteklemektedir.

Son yıllarda büyük ölçekli paralel ve dağıtık uygulamaların popülerliği artmaktadır. Araştırma ve üretim amaçlı olarak paralel ve dağıtık programlama standart donanımlar üzerinde uygulanmaktadırlar. Google dağıtık hesaplama yeteneklerini dağıtık hesaplama için geliştirdikleri Map-Reduce [31] iskeleti (framework) ile geliştirmektedir. GFS üzerinde Map-Reduce uygulamalarını geliştirmektedirler. Bu

birleşim sayesinde standart donanımlar daha değerli hale gelmektedirler. Standart donanımlar üzerinde veri yoğun, büyük ölçekli uygulamalar çalıştırılabilmektedir. Fakat bu iskelet tüm platformlar ile uyumlu değildir, telif haklar Google'a aittir. Linux temelli (C, C++) platformlar desteklenmektedir.

Açık kaynak çözüm olarak ta Yahoo tarafından tüm platformlar ile uyumlu versiyonları (JAVA) geliştirilmektedir. Hadoop Dağıtık Dosya Sistemi (HDFS – Hadoop Distributed File System) [32] Google'ın yaklaşımını herkese açık hale getirmektedir. HDFS veriyi dağıtık mimari üzerinde depolamaktadır. Map-Reduce paralel programlama iskeletini (framework) büyük ölçekli veri yoğun uygulamalar için desteklemektedirler.

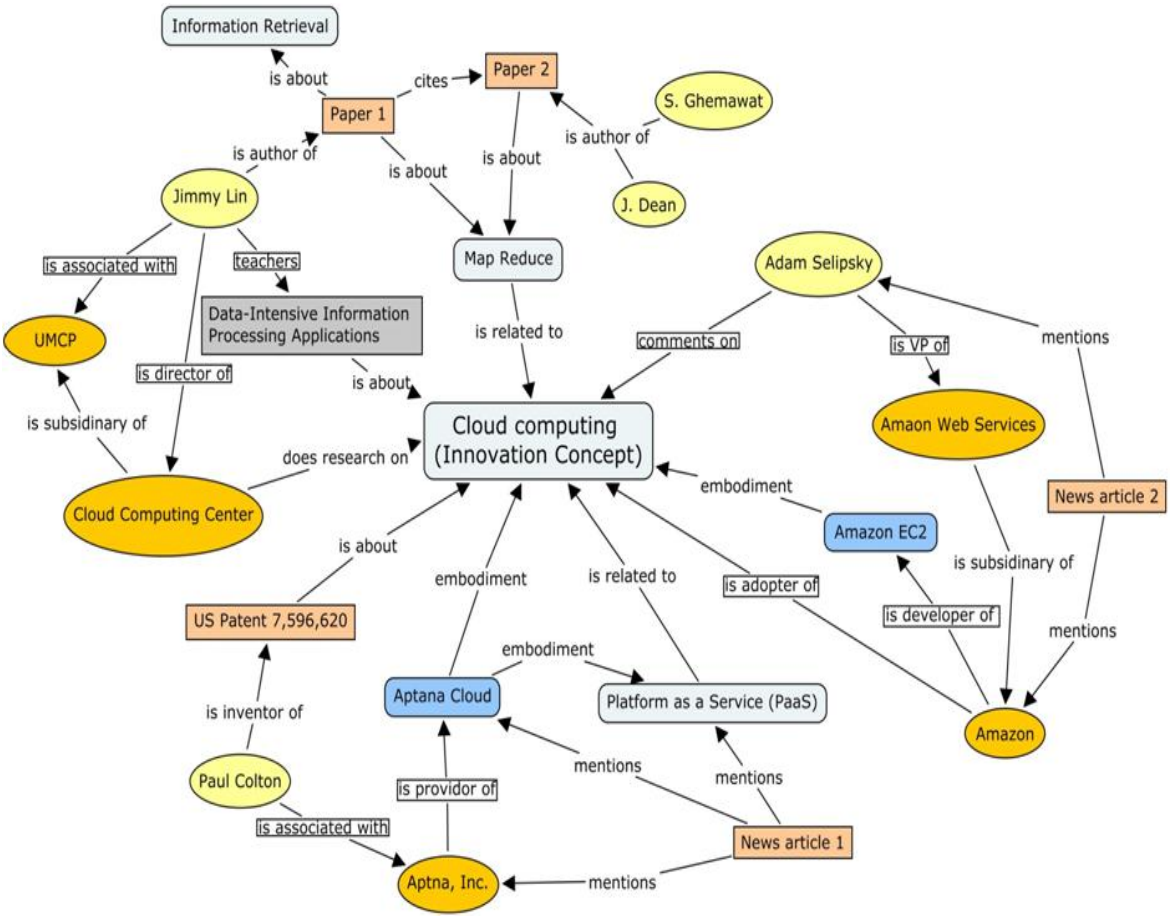
Ancak HDFS disk temelli çalışmaktadırlar. Çok fazla okuma yazma olduğu için veri erişimi çok yavaş yapılabilmektedir. HDFS dosya sistemlerinin ön bellekleme mekanizmalarının geliştirilmesi ile büyük veriye hızlı erişim ve büyük veri üzerinde interaktif sorgulamalar yapılması mümkün olmaktadır.

Tachyon dağıtık dosya sistemi TachyonFS [33] sağladığı ön bellekleme mekanizmaları ile büyük veriye bellek hızında erişim ve büyük veri üzerinde interaktif sorgulamaya imkân sağlamaktadır. Büyük veri üzerinde interaktif SQL sorgulamaları yazılması Shark [34] ile mümkün olmaktadır.

Büyük çizge (Graph) verilerinin interaktif olarak yüklenip, sorgulanabilmesi GraphX [35] ile yapılabilmektedir. Makina öğrenmesi algoritmalarının büyük veri üzerinde interaktif olarak uygulanabilmesini Mlbase [36] sağlamaktadır. Akan büyük veri üzerinde interaktif sorgular ise D-Streams [37] metodu ile gerçekleştirilebilmektedir.

### 3.3 Büyük Veri Sistemleri Entegrasyonu

Büyük veri kaynakları farklı sistemlerden farklı formatlarda toplandığından uyumsuzluk problemleri yaşanmaktadır. Farklı sistemlerin anlamsal olarak standardizasyonları ve ortak veri iletişim protokolleri oluşturulması gerekmektedir. Ortak iletişim için anlamsal olarak bütünleştirme gerekmetkedir. Bu bütünleştirmeler için ontoloji tasarımı ile sağlanabilmetkedir [44]. Ontoloji varlıkların, türlerin, özelliklerin isimlendirilip aralarındaki ilişkilerin modellendiği yapılardır. Varlıklar konu (subject), ilişki türü (predicate), ve nesne (object) üçlüleri olarak adlandırılırlar. Varlıklar arasındaki ilişkiler bu üçlülerin birbirleri ile bağlantılarını gösteren bağlı veri tabanları (linked data) olarak gösterilirler.




Şekil 6. Büyük veri sistemleri için örnek ontoloji [45]

Şekil 6 da görüldüğü gibi ontolojide varlıklar belirlenen kurallara göre adlandırılırlar. Bu sayede varlıklar arasında ortak, standartlaştırılmış bir dil oluşturulur. Standardizasyon ile faklı alandaki sistemlerin birbirleri ile bütünleştirilmesi ve bütünleşik çalışması sağlanabilmektedir. Tasarlanan bu ontolojiler uygulamalar için geliştirilmiş akıllı ajanlar için bilgi sağlamakta ve daha verimli anlamsal arama metotları geliştirilebilmektedir. Bu sayede herhangi bir alanda bilgi tabanlı akıllı sistemler geliştirilebilmektedir.

Bu yaklaşım ile büyük veri üzerinde varlıklar arasındaki (named-entity) ilişkiler bulunup, sınıflandırılabilmektedir. Her uygulama alanı için mümkün olduğunca jenerik bilgi tabanları oluşturulabilmektedir. Dağıtık analitik sistem ile ajanlar sql, çizge, makine öğrenmesi, akan veri analitiği sensörleri ile jenerik kabiliyetlerini büyük veri analizi için bilgi tabanı olarak kullanılabilmektedirler.

Her alan için detaylı model özelleştirmeleri uygulama geliştirme esnasında yapılabilmektedir. Bu mimari sayesinde varlık ayrıştırmaları (Named Entity Disambiguation) alanlar için teknoloji bağımsız özelleştirilip tekrar kullanımı ve farklı teknolojilerle bütünleştirilmesi minimum maliyetler gerçekleştirilebilmektedir. Varlıkların birbirleri ile ilişkilendirilmesi ve gruplandırılması (Entity-Linkage) yapısal olmayan veya yarı yapısal verilerde de minimum maliyetle hızlı olarak gerçekleştirilebilmektedir.

### 3.4 Büyük Veri Kaynaklarının Dinamik Yönetimi

Teknolojinin hızlı gelişim ile bilgi kaynakları çok hızlı değişmektedir. Bilgi toplamak ve bunları dinamik olarak yönetmek gerekmektedir. Teknoloji ve bilgi kaynağı türüne göre bilginin formatı güncellenmektedir. Depolanan ve aktif kullanılan bilgi üzerindeki değişimler yüksek maliyetlere neden olmaktadır. Bilgi formatlarının dinamik olarak tasarlanıp, yeni gelen bilgilerin yeni teknolojiler ile hızlı entegre olmasının sağlanması gerekmektedir. Farklı alanlardan teknik veya alan uzmanlarının ortak çalışması ile oluşturulan bilginin dinamik yönetimi için dinamik ontoloji geliştirme araçları kullanılmaktadır. DynamOnt [46] vb. araçlar ontolojilerin dinamik olarak tasarımı ve yönetimini sağlamaktadırlar. Bu metot ve araçlar ile dinamik bilgi yönetimi ve teknoloji entegrasyonları minimum maliyet ile gerçekleştirilebilmektedir.

Ajanlar çevrelerini sensörleri vasıtasıyla algılarlar ve kendi içerisindeki karar mekanizmalarına göre aksiyon gerçekleştirirler. Aksiyonlarını eyleyiciler vasıtasıyla çevrelerine aktarırlar. Ajanlar hedeflerine göre aksiyonlarını optimize edip, karar mekanizmalarını geliştirebilirler. Akıllı ajanlar ise insanlara karar vermelerine yardımcı olabilen, gerekli durumlarda insanlar yerine karar verebilen yazılımlardır.

Akıllı ajanalar sürekli tekrarlanan görevleri yerine getirebilir, karmaşık veri kümelerini standardize edebilir, hatırlatmalarda bulunabilir, yeni bilgiler öğrenebilir ve önerilerde bulunabilirler. Akıllı ajanlar, internet arama motorları, internet alışverişleri, müşteri yardım masaları, sağlık tarama sistemleri gibi alanlarda amaca yönelik olarak tasarlanıp ihtiyaca göre geliştirilebilmektedirler.

Akıllı ajanların değişen çevre şartlarını dinamik olarak güncelleyip, hedeflerine ulaşmak için oluşturdukları planları anlık olarak güncellemeleri gerekmektedir.Aksiyonlarını dinamik olarak yönetebilmeleri için büyük çizgeler içerisinde dinamik olarak arama yapmaları gerekmektedir. Likhachev ve arkadaşları büyük çizgeler içerisinde herhangi bir zamanda dinamik olarak arama yapılabilmesi için dinamik çizgeler üzerinde herhangi bir zamanda arama algoritması geliştirmişledir [38].

Likhachev diğer bir çalışmasında ajanların planlarını herhangi bir zamanda güncel tutabilmeler için tekrar planlama A* algoritması geliştirmişlerdir [39]. Geliştirilen bu algoritmaların büyük miktarda veri üzerinde anlık karar verebilmesi gerekmektedir. Büyük ölçekli çizgelerin hızlı olarak sorgulanabilmesi için küme üzerinde hızlı olarak dağıtık sorgulama algoritmalarının geliştirilmesi gerekmektedir. Küme üzerinde hızlı sorgulama metotları ile dağıtık analitik uygulamaların sorguları hızlı ve dinamik olarak çalıştırılabilmektedir.

# 4. MICROBLOG METİNLER İLE GÖRÜŞ MADENCİLİĞİ

Dağıtık dosya sistemleri ile büyük veri küme üzerinde hızlı ve ölçeklenebilir olarak yönetilebilmekte ve analitik işlemler hızlandırılabilmektedir. Sistemler alana ve veri türüne göre özelleştirilerek ölçeklenebilir analitik platformlar oluşturulabilmektedir. Bu çalışma kapsamında tasarlanan dağıtık büyük veri analitik sistemi üzerine mikroblog metinler ile görüş madenciliği uygulamaları geliştirilmiştir. Sosyal medya üzerinde biriken metinlerde görüş madenciliği yapılabilmesi için mikroblog metinlerin ölçeklenebilir sistemde tutulması ve dağıtık işleme iskeleti (framework) ile işlenmesi gerekmektedir. Bu kısımda mikroblog metin analitiği için geliştirilen dağıtık analitik sistem anlatılmaktadır ve görüş madenciliği uygulamasında kullanılan dağıtık skorlama ve k-means kümeleme algoritmaları ayrıntılı olarak açıklanmaktadır.

## 4.1 Görüş Madenciliği

Twitter, Facebook, blogs, vb. web teknolojilerinin hızlı artması ile görüş madenciliği konusunun popülerliği hızla armaktadır. İnsanların duygularının, olayların, varlıkların, sosyal medya kullanarak otomatik olarak anlaşılması değer kazanmakla birlikte araştırmacılar ve şirketler bu yönde çalışmalarını yoğunlaştırmaktadırlar.

Bu çalışmada sözlük tabanlı duygusal skorlama ve kümeleme tabanlı sınıflandırma içeren bir metot sunulmaktadır. Duygusal skorlama algoritması mikroblog metinler üzerine sözlük tabanlı skorlama algoritması uygulamaktadır. Skorlama sonuçlarına göre metinle pozitif, negatif, nötr olarak sınıflandırılmaktadırlar. K-means kümeleme algoritması her grup için benzer karakteristikleri bulmakta ve buna göre sınıflandırmalar gerçekleştirmektedir. Yeni gelen her metin en benzer gruba atanmaktadır. Pozitif, negatif ve nötr olmak üzere üç adet etiketlenmiş grup bulunmaktadır. Her grup için örnekler ve dağılımlar önceden belirlenmiştir.

Veri kümesinde çok miktarda mikroblog metin buluduğundan standart veri işleme yaklaşımları yerine dağıtık analitik mimari önerilmektedir. Veri kümesi bağımsız olduğundan yığıt olarak paralel işlemeye uygunluk göstermektedir. Güncel Hadoop ve NoSQL teknollojileri verinin karakteristiğine göre modifye edilip bütünleşmiş edilmişlerdir. Her yeni gelen mikroblog metin NoSQL veri tabanında JSON formatında saklanıp map-reduce algoritmaları ile işlenmektedir. Sonuçlar tekrar NoSQL veri tabanında tutulmaktadır.

Thelwall geliştirdiği SentiStrength algoritması ile sözlüksel yaklaşım kullanarak metin içindeki kelimeleri pozitif, negatif, nötr olarak ayrıştırarak duygusal skor hesaplamaktadır. Fakat metin içerisinde herhangi bir gramer kuralına bakamamaktadırlarç [2].

Ozsert ve Özgür diğer bir çalışmada Wordnet kullanmaktadırlar, her kelime birbirine ilişkisel çizge ile bağlıdır, kelimelerin pozitif küme veya negatif kümeye olan mesafesi ile kutuplaşmaları ifade edilerek sınıflandırılmaktadır [3]. Bu çalışma İngilizce ve Türkçe kelimeler için uygulamaktadır. Ayrıca bazı çalışmalarda büyük ölçekli dağıtık hesaplama sistemleri metin analitiği için önerilmektedir. Khuc'ın önerdiği dağıtık sistemde " cümle kurucu ve duygusal sınıflandırıcı " olarak iki bileşen bulunmaktadır. Bu sistemi Twitter metinlerinin duygusal analizleri için kullanmaktadırlar [4]. Lin makine öğrenmesi algortimalarını büyük ölçekli Hadoop temelli, Pig merkezli platformda uygulamaktadırlar [5]. Bu platformda makine öğrenmesi metotları, veri örnekleme, özellik üretme, veri eğitimi ve testler toplanan tweet ler üzerinde uygulanmaktadırlar.

Pang ve Lee [6] görüş madenciliği üzerine yapılan çalışmaları özetlemektedirler. İnsanların düşünce ve hislerini bulmaya yönelik çalışmalara odaklanmaktadırlar. Verilerin paralel olarak işlenebilmesi için çoklu Hadoop görevlerinin konfigüre edilip eş zamanlı olarak çalışması gerekmektedir. Lee [7] çalışmasında bu metodun artı ve eksilerini değerlendirmektedir. Kümeleme algoritmalarının performans iyileştirmesi için paralel işleme metotları iyi sonuçlar vermektedir. Zhao ve arkadaşlarıda k-means algortimasını paralel olarak mapreduce iskeleti ile çalıştırmaktadırlar [8]. Go ve arkadaşları [9] çalışmalarında farklı makina öğrenmesi metotlarını kullanarak mikroblog metinlerin otomatik olarak pozitif, negatif, nötr olarak sınıflandırmasını sağlamaktadırlar. Zhou ve arkadaşları büyük ölçekli veri kümelerinin kümelemesinde paralel işleme ile daha iyi sonuçlar aldıklarını çalışmalarında değerlendirmektedirler [10].

Bermingham and Smeaton mikroblog metin sınıflandırmasını uzun blok metinlere karşılaştırmalı olarak kullanmışlardır. Alışılmışın dışında olarak mikroblog metinler kısa metinler ve az bilgilerden dolayı duygu analizleri için uygunluk göstermektedirler. Yaptıkları çalışmada mikroblog duygu analizleri blog duygu analizlerinden daha iyi sonuç vermektedir [11]. Ayrıca çalışmalarında öğreticili öğrenme metotlarının kullanımlarının kullanım senaryoları ve zorluklarından bahsederek potansiyel çalışmaların bulunduğunu ve çalışılması gerektiğini ifade etmektedirler. O'Connor müşteri güvevnini, politik görüş anketlerini analiz etmektedir ve mikroblog metinlerdeki duygu sıklıklarında korelasyon bulmaktadırlar [12]. Çalışmlarında akan metin analitiğini geleneksel anketleme metotları yerine önermektedirler.

Pak and Paroubek dökümanları pozitif, negatif, nötr olarak sınıflandırmak için semantik sınıflandırıcı geliştirmişlerdir. Duygusal sınıflandırıcıyı eğitmek için otomatik olarak derleme yapmaktadırlar. Söz dizimsel yapılar genelde duyguları ve

durumları belirlemek için kullanılmaktadırlar. Metinlerde duyguları belirlemek için POS-tags kullanımını önermektedirler. Multinomial Naïve Bayes sınıflandırıcı ile N-gram ve Pos-tas karakterlerini çıkarmaktadırlar. Çalışma İngilizce için geliştirilmiştir ve diğer dillerede uygulanabilmektedir [14].

Kouloumpis sözdizimsel karakter algılama özelliklerini Twitter metinlerinde duygu analitiği için kullanmaktadırlar. Mevcut sözdizimsel kaynakların problemlerini incelemektedirler. Problem için öğreticili bir yaklaşım önermektedirler. Twitter hashtag lerini eğitim verisi üretimi amacı ile örnek olarak kullanmaktadırlar [15]. Martinez-Camara Twitter metinleri üzerinde duygu analizlerini kapsamlı olarak incelemekte ve literatür taramasını ayrıntılı olarak ifade etmektedirler [13]. Yaptıkları literatür araştırması sonucunda çalışmaların temel olarak odaklandıkları problemler ve çözümleri sınıflandırmaktadırlar. Temel olarak, (1) kutuplaşma sınıflandırması, (2) olay tahmini; kullanılan metotlar (1) öğreticisiz, (2) öğreticili ve (3) hibrit; kullanılan algoritmalar çok çeşitlilik göstermektedirler. Örneğin (1) SVM, (2) Naïve Bayes, (3) time series, (4) çizge tabanlı vb.

Sonuç olarak, mikroblog metinler üzerinde sözcük tabanlı analitik duygu analizleri için yaygın olarak önerilmekte ve kullanılmaktadırlar. Metotlar genellikle iteratif yöntemleri uygulamaktadırlar. Veriye çoğu zaman tekrar tekrar ulaşım gerektirmektedirler, tekrarlı erişim gerçekleştirmektedirler. Klasik veri tabanları bu metotları işlemek için çok zaman harcamaktadırlar. Güncel olarak geliştirile Dağıtık Veri Analitik Sistemleri büyük veriye paralel olarak erişebilmektedirler ve yığın sorgulama işlemlerini gerçekleştirebilmektedirler. Ayrıca makine öğrenmesi algoritmalarının bu platformlara uyarlanması ve optimize edilmesi sayesinde daha hızlı analitik fonksiyonlar geliştrilebilmekte ve sonuçlar daha hızlı elde edilebilmektedir.

Bu çalışma kapsamında mevcut sistemler incelenmiş ve Türkçe Metinler için özlleştirişmiş bir dağıtık analitik sistem ve ilgili algoritmalar geliştirilmiştir. Algortimalar öğreticili ve öğreticisiz olarak hibrit tasarlanmışlardır. Skorlama ve kümeleme algoritmaları karşılaştırmalı olarak sunulmaktadır. Metin analitiği için özelleştirilmiş analitik platform özel bulut mimarisi olarak sunulmaktadır.

Tasarlanan platform da mikroblog metinler dağıtık olarak ölçeklenebilir platformda tutulmaktadırlar. Dağıtık dosya sistemlerinin ölçekleme özelliği sayesinde platform ihtiyaca göre minimum zaman ve donanım maliyeti ile genişletilebilmektedir. Düğüm değişimleri için dağıtık dosya sistemlerinin veriyi üç kopya olarak tutması sayesinde otomatik değişim hızlı yapılabilmektedir. Düğüm üzerindeki fonksiyonlar otomatik olarak yedeklerine aktarılmakta ve değişimden sonra düğüme yeni fonksiyonlar otomatik olarak tanımlanabilmektedir.

### 4.2 Skorlama Algoritması

Skorlama algoritması istatistiksel tahminlerin kesinliğini belirlenen skorlama fonksiyonu veya skorlama kuralına göre belirler. Tahmin sonuçlarının somut çıktılara olasılıksal değerler atayabildiği durumlar için kullanılabilmektedir. Mümkün olan çıktılar 1,0 şeklinde ikili bitler veya sınıflandırmalar olarak ifade edilebilmektedir. Skorlama algoritması temel olarak şöyle çalışır; skorlama fonksiyonu veya kuralı her x çıktısı için v olasılıksal vektörünü belirler. x olayının gerçekleşmesi için skorlama fonksiyonu S (v,x) değeri olarak gösterilebilir.

Mikrolblog metinlerin olumlu, olumsuz, nötr olarak sınıflandırılması için skor vektörleri ve fonksiyonları belirlenmiştir. Her mikroblog metin skorlama fonksiyonu sonucunda olumlu, olumsuz, nötr olarak sınıflandırılmaktadır. Skorlama vektöründe her kelimeye Çizelge 1 de görüldüğü gibi -5 ile +5 arasında skor atanmıştır. Skorlama vektörü toplamda 2159 kelimeden oluşmaktadır. Sentiwordsnet [47] teki skorlama kelimeler Türkçe ye çevrilmiş ve İngilizce olarak ta kullanılmaktadır. Bu sayede algoritma Türkçe ve İngilizce için de çalışmaktadır.

Çizelge 1. Örnek duygusal sözcükler

| Score | English | Turkish |
|---|---|---|
| -2 | Abduction | Kaçırma |
| -3 | Abhor | İğrenmek |
| 2 | Ability | İktidar |
| 2 | Ability | Yetenek |

Skorlama fonksiyonu her mikroblog metin içerisinde skorlama vektörü anahtar kelimelerini atamakta ve her kelime için skor değeri üretmektedir. Skorlama fonksiyonu Formül 1 de görüldüğü gibidir.

Değişken t, bir mikroblog metni (örneğin tweet) olmak üzere ve words(t) bu metindeki kelimeler kümesini veren fonksiyon olmak üzere, t'nin skoru aşağıdaki gibi hesaplanır:

$$\text{skor}(t) = \sum_{w \in words(t)} Skor(w) \qquad (4.1)$$

Formül 4,1'de görüldüğü gibi her mikroblog metin için duygu vektörü anahtar kelimeleri: "w" her mikroblog metin içinde aranmakta ve bulunduğu durumda skor değerine eklenmektedir. Toplam skor değeri S > 0 ise pozitif, S < 0 ise negative, S = 0 ise notr olarak değerlendirilmektedir.

**Algoritma 2 Skorlama Algoritması**

**Input:**
*MicroblogTextList* (*t*) *t* = (*microblogText, tweetID*)
*DictionaryList* (*d*) *d* = (*keyword, keyScore*)

**Output:**
*TweetList* (*t*) *t*= (*microblogText, tweetID, tweetScore*)

**function map** (**key:** *MicroblogText_ID*, **Value:** *MicroblogText*)

Tokanize *Tweet[] token* ∈ *TweetList*
**while** (*token* has more *tokens*)
**if**(i=0,1,,,,n ($d_i$) **and** $d_i$ ∈
*DictionaryList*, $d_i$ contains t)
*t.tweetScore* ++= *d.score*
**context**.write(*t.tweetScore*)

**return** *MicroblogTextList;*

**function reduce** (**key:** *MicroBlogText_ID*, **Value:** *MicroBlogText* )

*val* = 0;
**while** (**IntWritable** *val* **in Context**)
**if** (*val* > 0)
*positive* ++;
**else if** (*val* = 0)
*neutral* ++;
**else if** (*val* < 0)
*negative* ++;
*totalTweet* ++;

**context.write**(*t. microblogText*);

**return** (**key:** *MicroblogText_ID*, **result:** *MicroblogText*); // Labeled Results for each text.

Algoritma 2. Mikroblog metinler için dağıtık skorlama algoritması.

Skorlama algoritması mikroblog metinleri dağıtık küme üzerinde sınıflandırarak işlemektedir. Algoritma şu şekilde çalışmaktadır;

1. Bütün mikroblog metinler dağıtık dosya sistemi üzerine key/value ikilileri olarak (key: ID, value: microblogText) dağıtılır.
2. Duygusal skorlama için key/value ikilisi olarak (key: sentimentWord, valu: score) oluşturulmuş sözlük dağıtık dosya sistemine yüklenir.
3. Her mikroblog metin için skor hesaplaması yapılır.
    1. tweet score = Sum (w) (score (w)) where word is a word in dictionary and w is mentioned in t.
4. Skoru hesaplanan her metin pozitif/negatif/nötr olarak etiketlenir.

**Dağıtık Skorlama Uygulaması Örneği**

```
public void map (LongWritable key, Text value, Context context) InterruptedException
{
      String line = value.toString();
      JSONObject jsn;
           num++;
           Long tweetid = (long) 0;
                    try {
                              jsn = new JSONObject(line);
                              line = (String) jsn.get("tweet");
                              tweetid = (Long) jsn.get("tweetid");
                     }
                     catch (JSONException e) {
                              e.printStackTrace();
                     }
      StringTokenizer tokenizer = new StringTokenizer(line);
       IntWritable result = new IntWritable(0);
       int say = 0;
       while (tokenizer.hasMoreTokens())
       {
           String token =  tokenizer.nextToken();
           if(dictList.containsKey(token)){
                    say += dictList.get(token).intValue();
           }
        }
       result = new IntWritable(say);
       word.set("\""+line.replace("\"", "\\\"") +"\" , \"TweetId\":" + tweetid);
       context.write(word, result);

   }

public void reduce(Text key, Iterable<IntWritable> values, Context context)

                    throws IOException, InterruptedException

{

      int sum = 0;

      for (IntWritable val : values) {
         sum += val.get();
         counter++;
         IntWritable result = new IntWritable(0);
               if(sum>0){
                    result = new IntWritable(1);
                    ++pos;
                }else if(sum<0){
                    result = new IntWritable(-1);
                    ++neg;
                }else{
                    result = new IntWritable(0);
                    ++neut; }
              totalTweet++;
          String jsonFormat = "{ \"Tweet\":"+key.toString()+" , \"Score\":\""+result+"\"  }";
          context.write(new Text(""), new Text(jsonFormat));

       }
}
```

Uygulama 1. Dağıtık Skorlama Algoritması Map-Reduce Program Kodu.

Yukarıdaki program kodunda görüldüğü gibi skorlama algoritması girdi olarak mikroblog metinler listesini ve duygu sözlüğünü almaktadır. Metinler JSON formatında tutulmakta ve duygu tespiti için ayrıştırılmaktadırlar.

Duygu vektöründeki her kelimeye manuel olarak puan verilmiş ve bu değerlere göre kelimenin metin içerisindeki toplam duygu skoruna olan etkisi hesaplanmaktadır. Duygu sözcükleri program içerisinde sözlük formatında tutulmaktadır. Sözlük listesi dağıtık dosya sistemi üzerinde tutulmakta ve her düğüm üzerinde çalışan fonksiyona göre dosya sistemi tarafından yönetilmektedir.

Her metin için skorlama fonksiyonu dağıtık olarak çalıştırılmaktadır. Küme üzerinde dağıtılan skorlama fonksiyonu ile her metnin skoru hesaplanmaktadır. Reduce aşamasında metnin skor sonuçları atanmaktadır ve metin pozitif, negatif, nötr olarak sınıflandırılmaktadır.

### 4.3 Kümeleme Algoritması

K-means kümeleme algoritması vektörleri sayısallaştırmak için kullanılan yaygın bir metottur. Veri madenciliğinde küme analizleri için yaygın olarak kullanılmaktadır. K-means kümeleme algoritması x sayıda gözlemi k kümeye yakınlık ölçütüne göre atayarak çalışmaktadır. K-Means kümeleme algoritması herhangi n gözlem kümesini ($x_1$, $x_2$, ..., $x_n$) ( her gözlem d boyutlu bir vektördür) kümeye atamayı hedefler (k<= n) S = ($s_1$, $s_2$, .... ,$s_k$). Formül 4.2' de görüldüğü gibi minimum s küme değerini bulmaktadır.

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x}\in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 \qquad (4.2)$$

$\mu_i$ = Si noktalarının mean değeri.

Formül 2. K-means kümeleme algoritması [48].

Kümeleme algoritması ile metinleri pozitif, negatif, notr olarak sınıflandırmak için x gözlem kümesi olarak pozitif, negatif ve nötr kümeler belirlenmiştir. Her yeni gelen mikroblog metin hangi kümeye daha yakın vektörel puanlama elde ediyorsa o sınıfa atanmaktadır. Vektör atamaları bit değeri olarak 1,0 şeklinde gerçekleştirilmektedir. Her mikroblog metin için olumlu, olumsuz, notr vektör kümeleri ile karşılaştırılmaları 100 bit vektörler ile gerçekleştirilmektedir. Mikroblog metinlerde en çok bit benzerliği olan vektörler aynı sınıflarda (pozitif, negatif, nötr) tutulmaktadır. Çizelge 2'de görüldüğü gibi her vektör için anahtar kelimeler belirlenmiş ve bu kelimelerin mikroblog metinlerde bulunup bulunmama durumuna göre her mikroblog metin için 100 bit vektör oluşturulmaktadır. Oluşan vektörlerin benzerlik karşılaştırılmaları ile de sınıflandırmaları gerçekleştirilmektedir.

Çizelge 2. Örnek duygusal vektörler

| **İngilizce** | **Türkçe** |
|---|---|
| Ugly | Çirkin |
| Uneasy | Huzursuz |
| Uneasy | Rahatsız |
| Uneasy | Mutsuz |
| Unjust | Haksız |

**Algoritma 3 K-Means Kümeleme Algoritması**

```
Input:
MicroblogTextList (t)  t =  (microblogText, tweetID)

PositiveSamples (p ) p = (microblogText, tweetID)
NegativeSamples (n ) n = (microblogText, tweetID)
ZeroSamples (z ) z = (microblogText, tweetID)
PositiveVectors (pv) pv = (microblogText, tweetID)
NegativeVectors (nv ) nv = (microblogText, tweetID)
ZeroVectors (zv ) zv = (microblogText, tweetID)

Output:
MicroblogTextList  (t) t= (microblogText, twetID, tweetScore)

fınction map (key: MicroblogText_ID, Value: MicroblogText)

  Vectorize SampleVectors sv ∈ SampleTweetList st
     if (sv ∈ p) { pv.add(sv) }
     if (sv ∈ n)  { nv.add(sv) }
     if (sv ∈ z)   { zv.add(sv) }

      foreach tweet in tweetList
         Vectorize tweet convert to v
            if (sv ∈ p) { pv.add(sv); v.tweetScore = positive; }
            if (sv ∈ n)  { nv.add(sv) ; v.tweetScore = negative; }
            if (sv ∈ z)   { zv.add(sv) ; v.tweetScore = neutral; }

      context.write(v.tweetScore)

 return MicroblogTextList;


function reduce  (key: MicroBlogText_ID, Value: MicroBlogText )

       val = 0;
       while (IntWritable val in Context)
          if (val  > 0)
               positive ++;
          else if (val  =  0)
               neutral ++;
          else if (val  <  0)
               negative ++;
          totalTweet ++;

       context.write(t. microblogText);

 return (key: MicroblogText_ID, result: MicroblogText); // Labeled Results for each text.
```

Algoritma 3. Mikroblog metinler için dağıtık k-means kümeleme algoritması.

K-means kümeleme algoritması mikroblog metinleri dağıtık küme üzerinde sınıflandırarak işlemektedir. Algoritma şu şekilde çalışmaktadır;

1. Bütün mikroblog metinler dağıtık dosya sistemi üzerine key/value ikilileri olarak (key: ID, value: microblogText) dağıtılır.
2. Mikroblog metin içerisinde geçen tüm kelimeler istatistiksel olarak sayılır ve metinler içerisinde en çok geçen 100 kelime seçilir ve 100 bit en çok kullanılan kelimeler vektörü oluşturulur.
3. Pozitif, negatif, nötr örnekler için 100'er tane metin seçilir ve her sınıf için 100'er örnek metin sınıfları oluşturulur.
4. En çok kullanılan kelimeler vektöründen pozitif, negatif, nötr örnek grubundaki her metin için 100 bit vektör oluşturulur. 100 bit vektörlerin her biti en çok geçen kelimeyi içeriyorsa 1, içermiyorsa 0 olarak atanır.
5. Her mikroblog metin pozitif, negatif, nötr vektörler ile karşılaştırılır. En fazla benzerlik gösterdiği vektör sınıfına atanır ve mikroblog metin pozitif, negatif veya nötr olarak etiketlenir.

**Dağıtık K-means Kümeleme Uygulaması Örneği**

```
public void map(LongWritable key, Text value, Context context)
                    throws IOException, InterruptedException
{
        //taking one line at a time and tokenizing the same
           String line = value.toString();
           String tweetId = "";
           JSONObject jsn;
            try {
                  jsn = new JSONObject (line);
                  line = (String) jsn.getString("tweet");
                  tweetId = (String) jsn.getString("tweetid");
                 }
                   catch (JSONException e) {
                      e.printStackTrace();
                  }

                  String tweetVec = vectorize(line);
                  SentimentVector sVec = new
                  SentimentVector(tweetVec,tweetVec.split(",").length);
                  String clusterName = clusterTweet(tweetVec,line);
                  String clusterType = clusterName.split("##")[0];
                            if(clusterType == "neutral")
                                      zeroVecs.add(sVec);
                            else if(clusterType == "positive")
                                      posVecs.add(sVec);
                            else if(clusterType == "negative")
                                      negVecs.add(sVec);


          context.write(new Text(clusterType), new Text(tweetId));
}

public String clusterTweet(String tweetVec,String tweet)
```

```
{
            String result = "";

            int pos = 0; // Number of same bits
            for(SentimentVector sv : posVecs )
            {
                        if(sv.compareVec(tweetVec)>pos)
                                    pos = sv.compareVec(tweetVec);
            }

            int neg = 0; // Number of same bits
            for(SentimentVector sv : negVecs )
            {
                        if(sv.compareVec(tweetVec)>neg)
                                    neg = sv.compareVec(tweetVec);
            }

            int zero = 0; // Number of same bits
            for(SentimentVector sv : zeroVecs )
            {
                        if(sv.compareVec(tweetVec)>zero)
                                    zero = sv.compareVec(tweetVec);
            }
           int max = Math.max(zero,Math.max(pos, neg));

           //SentimentVector twVec = new SentimentVector(100,tweet);

           if(max == zero)
           {
                        result = "neutral";
           }
           else if(max == pos)
           {
                        result = "positive";
           }
           else if(max == neg)
           {
                        result = "negative";
           }

            return result+"##"+pos+","+zero+","+neg;
}
public void reduce(Text key, Iterable<Text> values, Context context)
                        throws IOException, InterruptedException
{
          while (values.iterator().hasNext())
          {
                        totalTweet++;
                        String twt = values.iterator().next().toString();
                        String jsonFormat = "{ \"TweetID\":\""+twt+"\" ,
                                                \"Score\":\""+key.toString()+"\"  }";




                        İf(key.toString().equals("positive"))
                                                pos ++;
                        else if(key.toString().equals("negative"))
                                                neg ++;
```

```
                    else if(key.toString().equals("neutral"))
                                        zero ++;
                    context.write(new Text(""), new Text(jsonFormat));
                }
    }
```

Uygulama 2. Dağıtık K-Means Kümeleme Algoritması Map-Reduce Program Kodu.

Yukarıdaki program kodunda görüldüğü gibi kümeleme algoritması girdi olarak mikroblog metinler listesini ve duygu vektörlerini almaktadır. Metinler JSON formatında tutulmakta ve duygu tespiti için ayrıştırılmaktadırlar.

K-means kümeleme algoritması pozitif, negatif ve nötr 100 bit önceden oluşturulmuş 100'er adet vektör kümelerini girdi olarak almaktadır. Yukarıdaki program kodunda görüldüğü gibi veri setinde belirtilen met mikroblog metin için duygu vektöründeki metinleri içerip içermeme durumuna göre 100 bit vektör oluşturulmaktadır. Her metin için oluşturulan 100 bit lik vektörler küme üzerine dağıtılarak karşılaştırmaları yapılmaktadır.

Reduce aşamasında vektörleri benzerlik sonuçları toplanmaktadır. Her mikroblog metin pozitif, negatif, nötr vektör kümelerinden en çok hangisine yakınlık göstermiş yani benzer ise onunla etiketlenmektedir. Bu şekilde tüm mikroblog metinler küme üzerinde dağıtık olarak örnek kümeleri ile karşılaştırılarak sınıflandırılmaktadırlar.

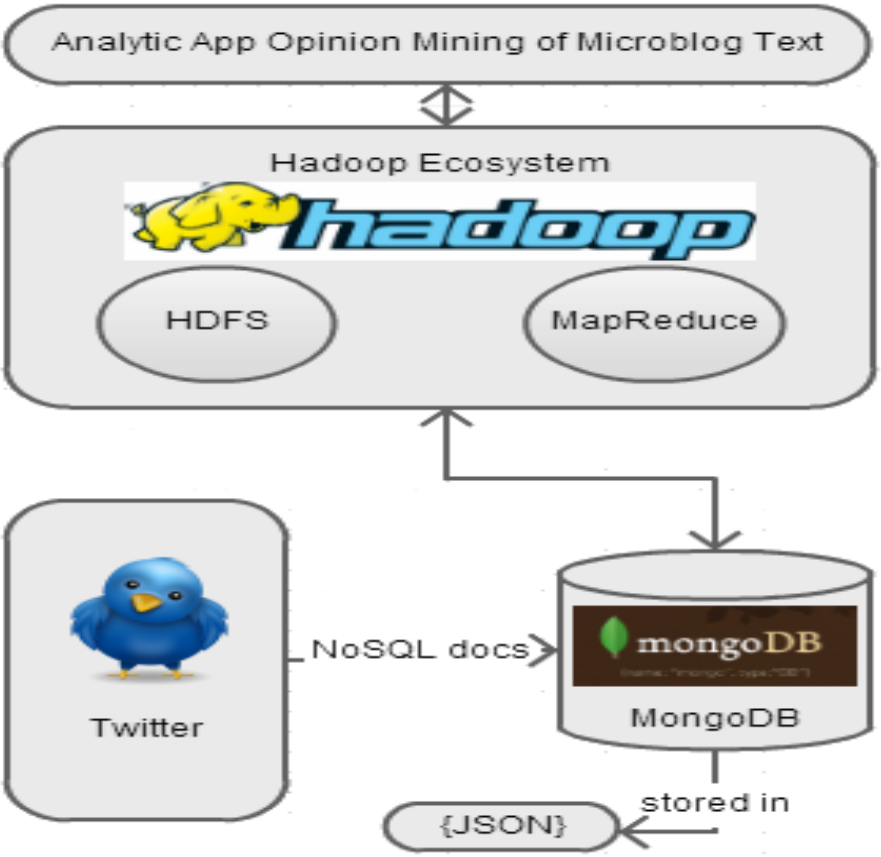
## 4.4 Mikroblog Metin Analitiği

Metin analitiği doküman içeriklerini algılamak ve doküman sınıflandırması için kullanılmaktadır. Doküman analizleri büyük metinler üzerinde standartlaşmış ve geçerliliği kabul edilmiş metot ve araçlarla yapılabilmektedir. Web teknolojilerinden önce kullanılan standart ve büyük doküman içerikleri metin olarak incelenmekte ve sınıflandırılmaktaydı. Web teknolojilerinin gelişimi ile dokümanların uzak konumlardan da erişimine ihtiyaç duyulmaktadır. Dokümanların web üzerinde erişim ve arama işlemleri uzun sürmekteydi. Bu yüzden dokümanlar web ve ağ erişim protokolleriyle hızlı ve verimli olarak erişilebilecek formatlarda depolanmaya başlandı. İnternet üzerinde erişim ve düzenleme imkânları sağlanmasıyla daha kullanışlı doküman yönetim sistemleri geliştirilmeye başlandı.

Sosyal medya gelişimiyle doküman boyutları ve içerikleri değişim göstermektedir. Kısa metin ve çeşitli içeriklerden oluşan dokümanlar üzerinde anlık karar destek mekanizmaları çalıştırılması gerekmektedir. Standart doküman analiz yöntemleri küçük ve sayısı fazla mikroblog dokümanlarda iyi performans vermemektedir. Verimli anlık analizler için bu metotlar küçük metinler için özelleştirilip, dağıtık platformlarda işlenmesi ve analitik olarak incelenmesine yönelik fonksiyonlar geliştirilmektedir. Standart doküman içeriği analizleri için geliştirilmiş olan sistemlerde içeriklerin sınıflandırılması büyük dokümanların konularına göre düzenlenebilmektedir. Ancak sayı arttıkça bu şekilde yönetim veri ve işlem yoğunluğunu arttırmaktadır. Mikroblog metinler için geliştirilen sistemler ile bu metinler üzerinde anlık olarak karar desteği sağlanabilmektedir.

## 4.5 Metin Madenciliği İçin Dağıtık Analiz Sistemi

Metin analitiği işlemlerinin mikroblog veriler üzerinde hızlı olarak gerçekleştirilebilmesi için metin verilerinin tüm küme üzerinde tutulabilmesi ve her metnin kümede dağıtık olarak işlenebilmesi gerekmektedir. Geliştirilen sistemde mikroblog metinler her düğümde dağıtık olarak tutulmakta, hesaplama işlemleri düğümlere map-reduce iskeleti aracılığıyla dağıtılmaktadır. Metin analitiği için geliştirilmiş analitik uygulama modeli ile dağıtık analiz sistemi metin tabanlı analitik işlemlerde jenerik olarak kullanılabilecektir. Sistemin genel mimarisi Şekil 7'de görüldüğü gibidir.




Şekil 7. Mikroblog metin analitiği için dağıtık analiz sistem.

Şekil 7'de görüldüğü gibi depolama katmanında dağıtık dosya sisteminin ölçeklenebilir yapısı kullanılmaktadır. Veri JSON formatında mongoDB de tutulmakta ve analitik işlemler ve gerekli durumlar için HDFS (Hadoop Distributed File System) dağıtık dosya sistemine aktarılmaktadır. Mikroblog metinler üzerinde dağıtık hesaplama yapabilmek için MapReduce iskeleti kullanılmaktadır. Bu iskelet ile analitik uygulamalar için belirlenen fonksiyonlar küme üzerinde düğümlerin rollerine göre dağıtılmakta ve işlenebilmektedir.

# 5. DEĞERLENDİRME

## 5.1 Test Verileri

Veri kümesinde 1-12 Nisan 2014 tarihleri arasında 573,794 adet mikroblog metin bulunmaktadır. Metinler Ankara'daki üniversiteler hakkında görüşleri içermektedir. Metinler Twitter API (Application Programming Interface) aracılığı ile toplanmıştır. Çizelge 3'te görüldüğü gibi metinler JSON formatında NoSQL veri tabanlarında tutulmaktadır.

Çizelge 3. Örnek mikroblog metinler.


{ "_id" : { "$oid" : "531ca82684ae0ac0b4b74a85" },

"tweet":"Dünyanınensaygın 100 üniversitesindetekTÜRKüniversitesi\nODTÜdiyeyazılrOrtadoğuTeknikÜniversitesidiyeokunr \nhttp://t.co/GCisdEmlbN",

"date" : { "$date" : 1394299521000 },

"userid" : 2174545004,

"tweetid" : 442350370007052288,

"retweetcnt" : 0,

"subjectid" : "1"

}

{ "_id" : { "$oid" : "531ca84184ae0ac0b4b74ce2" },

"tweet" "HacettepeÜniversitesi'ninteknolojiyevemühendislereverdiğiöneminresmidir. ÇöplükveTeknokentyanyana! http://t.co/7XolIoZY2W",

"date" : { "$date" : 1394366532000 },

"userid" : 88445878, "tweetid" : 442631433618993152, "retweetcnt" : 0,

"subjectid" : "15",

"pictureurl":"http://pbs.twimg.com/media/BiSK8YxIMAAE-Pa.jpg"

}


Veri küme üzerinde JSON metin formatında tutulmaktadır. Her metin için tek ID bulunmaktadır. Metinlerin içeriği, tarihi ve ilgili linkler saklanmaktadır. Metinleri depolamak için MongoDB veri tabanı kullanılmaktadır. Her metin analiz işlemleri

için Hadoop dağıtık dosya sistemine aktarılmakatıdır ( Hadoop Distributed File System – HDFS).

### 5.2 Test Ortamı

Bu kısımda dağıtık analitik sistemlerin performans olarak kıyaslamaları sunulmaktadır. Ayrıca bu kümelerde geliştirilen analiz uygulamalarının doğruluk olarak karşılaştırılmaları sunulmaktadır. Karşılaştırma yapılan tek düğümlü ve çok düğümlü sistemlerde donanımsal özellikleri Çizelge 4'te sunulduğu gibidir.

Çizelge 4. Tek düğüm ve çok düğümlü sistemlerin donanımsal özellikleri.

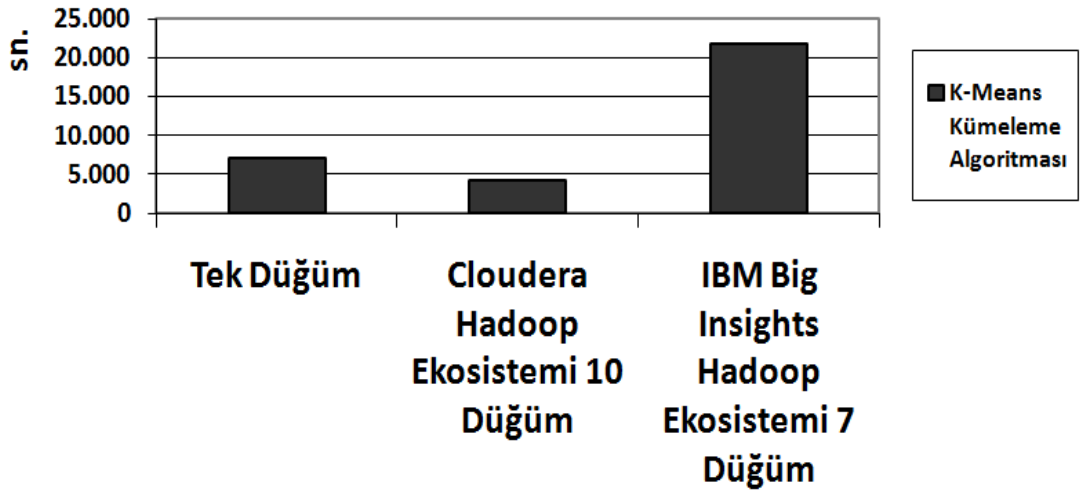
| **Düğüm Sayısı** | **Donanım Özellikler** |
|---|---|
| 1 düğüm | a) Processor:Intel(R)Core(TM)i72670QMCPU@2.20GHz<br>b) RAM: 7 GB<br>c) Operating System: Linux Enterprise Server 11 SP2 |
| 10 düğüm | a) Processor:Intel®Core™i5-2400CPU@3.10GHz × 4<br>b) RAM: 7 GB<br>c) Operating System: Ubuntu 12.04 LTS, Desktop |
| 7 düğüm | a) Processor:Dual-CoreAMDOpteron(tm)Processor 2210*4<br>b) RAM: 4 GB<br>c) Operating System: x86_64GNU/LinuxCentOS release6.5 |

Çizelge 4'te görüldüğü gibi analitik işlemlerin farklı, dağıtık sistemlerde performans değerlerinin yapılması ve bellek kısıtlarının gözlemlenebilmesi için tek düğümlü ve çok düğümlü sistemler tasarlanmış ve karşılaştırılmaktadır. Tek düğümlü sistemde standart işlemci ve bellek bulunmaktadır. 10 düğümlü sistemde 7 GB RAM ve standart işlemci kullanılmaktadır. 7 düğümlü sistemde 4 GB ve standart işlemci kullanılmaktadır. Kümeler arasındaki RAM miktarındaki farklılıklar ile analitik uygulamalar için RAM kısıtları ve uygulama performansına etkisi gözlemlenmiştir.

### 5.3 Testler ve Sonuçlar

Veri yoğun analitik işlemler için çok düğüm ve tek düğümlü mimarilerde bellek kritikliği ve yeterliliği konusunda değerlendirmeler ve analitik sistemlerde gözlemler sunulmaktadır.

Özelleştirilmiş analitik sistemlerde K-means ve skorlama algoritmaları ayrı ayrı test edilmiştir. Tek düğüm, çok düğüm performansları bulunmaktadır. Algoritma testler performans/hız ve doğruluk olarak değerlendirilmektedir. Şekil 8'de görüldüğü gibi K-means algoritması tek düğümde 6,000 sn.de çalışırken 10 düğümlü her düğümde 7 GB olan makinede performans iyileşmesi görülmüş ve 4,000 sn.de çalışmıştır. Fakat 7 düğümlü her düğümün 4 GB belleğe sahip olduğu durumda çalışması 21,000 sn. sürümüş ve performans iyileşmesi görülmemiştir. Sistem dağıtık olarak daha fazla belleğe sahip olmasına rağmen her düğümün 4GB belleğe sahip olduğundan dolayı performans iyileşmesi gözlenmemektedir.




Şekil 8. Metin analitiği için özelleştirilmiş analitik sistemde Kümeleme Algoritması yürütme zamanı performans karşılaştırılması.

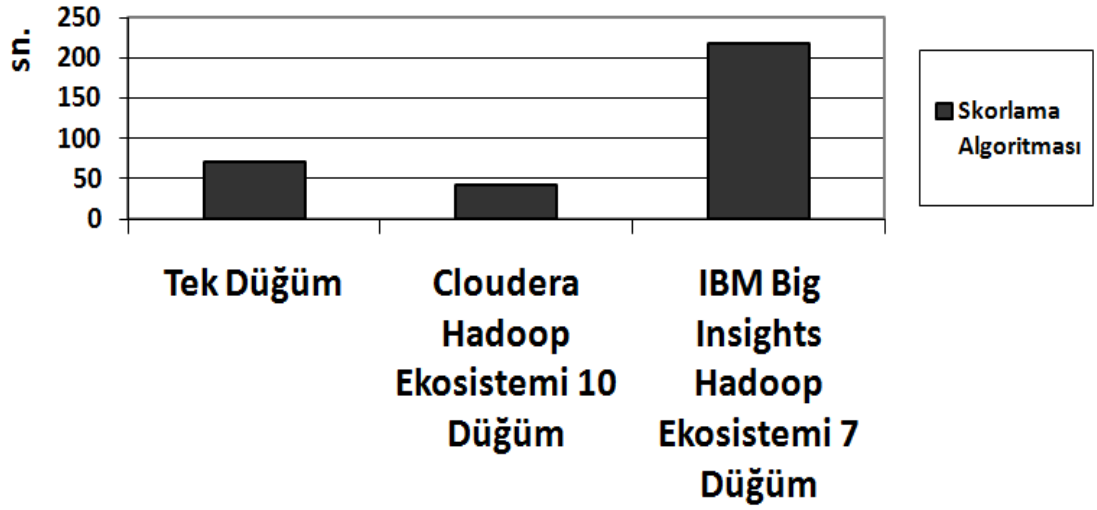
Kümeleme algoritmasının çalıştırıldığı dağıtık sistemler tek düğümlü ve çok düğümlü olarak tasarlanmıştır. Tek düğümlü sistem analitik işlemler için gerekli paketleri bulundurmaktadır. Çok düğümlü sistemlerde tek düğümün kopyaları aktarılmış ve düğüm fonksiyonlara göre özel paketler yüklenmiştir. Duygu vektörleri ve mikroblog metinlerin vektörlere aktarımı dağıtık dosya sistemi üzerinde gerçekleştirilmektedir.

Ağ yönetimi için ana düğüm olana makinalara küme yönetim fonksiyonları tanımlanmış ve web tabanlı ara yüzler ile kümenin yönetimine imkân sağlanmıştır.

Küme için metin depolamaları ve sonuçların saklanması JSON formatında MongoDB de tutulmuştur. Gerekli durumlarda HDFS (Hadoop Distributed File System) üzerine analitik işlemler için aktarımlar yapılmıştır. Metin analitiği için tasarlanan dağıtık sistemlerde her düğümde minimum 7GB RAM kısıtı olduğu gözlemlenmiştir.

Şekil 9'da görüldüğü gibi skorlama algoritması tek düğümde 50 sn.de çalışırken 10 düğümlü her düğümde 7 GB olan makinede performans iyileşmesi görülmüş ve 25 sn.de çalışmıştır. Fakat 7 düğümlü her düğümün 4 GB belleğe sahip olduğu durumda çalışması 130 sn. sürümüş ve performans iyileşmesi görülmemiştir. Sistem dağıtık olarak daha fazla belleğe sahip olmasına rağmen her düğümün 4GB belleğe sahip olduğundan dolayı performans iyileşmesi gözlenmemektedir. Metin analitiği için tasarlanan dağıtık mimaride her düğümde minimum 7GB RAM kısıtı olduğu gözlemlenmiştir.

Skorlama algoritmasının kümeleme algoritmasında göre daha hızlı çalıştığı ve dağıtık mimaride minimum 7GB bellek kısıtı şartı ile daha iyi performans verdiği gözlemlenmiştir.




Şekil 9. Metin analitiği için özelleştirilmiş analitik sistemde Skorlama Algoritması yürütme zamanı performans karşılaştırılması.

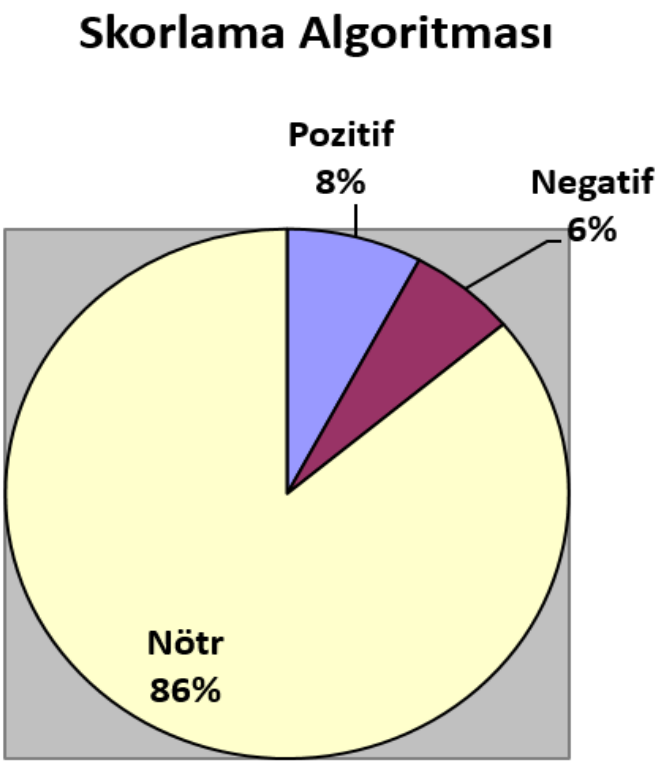
Dağıtık skorlama algoritmasının çalıştırıldığı dağıtık sistemler tek düğümlü ve çok düğümlü olarak tasarlanmıştır. Tek düğümlü sistem analitik işlemler için gerekli paketleri bulundurmaktadır. Çok düğümlü sistemlerde tek düğümün kopyaları aktarılmış ve düğüm fonksiyonlara göre özel paketler yüklenmiştir. Skorlama için oluşturulmuş sözlük dağıtık dosya sisteminde tutulmakta ve uygulama esnasında dağıtık dosya sisteminde tutulan her mikroblog metinle karşılaştırılmaktadır.

Ağ yönetimi için ana düğüm olana makinalara küme yönetim fonksiyonları tanımlanmış ve web tabanlı ara yüzler ile kümenin yönetimine imkân sağlanmıştır. Küme için metin depolamaları ve sonuçların saklanması JSON formatında MongoDB de tutulmuştur. Gerekli durumlarda HDFS (Hadoop Distributed File System) üzerine analitik işlemler için aktarımlar yapılmıştır. Metin analitiği için tasarlanan dağıtık sistemlerde her düğümde minimum 7GB RAM kısıtı olduğu gözlemlenmiştir.

Çizelge 5. Skorlama ve K-means kümeleme algoritmaları sonuçları.

| **Algoritma** | **Pozitif Mikroblog Metin Sayısı** | **Negatif Mikroblog Metin Sayısı** | **Nötr Mikroblog Metin Sayısı** |
|---|---|---|---|
| **Skorlama** | 44,612 | 35,373 | 493,809 |
| **K-means Kümeleme** | 33,832 | 44, 648 | 495,314 |

Skorlama algoritması sözlük tabanlı çalışmaktadır. Oluşturulan duygu sözlüğü ve önceden atanmış puan değerleri ile her mikroblog metin için toplam puan bulunmakta ve pozitif, negatif, nötr kümelerinden birine atanmaktadır. Çizelge 5 te görüldüğü gibi, skorlama algoritması 573,794 adet mikroblog metni 44,612 pozitif, 35,713 negatif, 493,809 nötr olarak sınıflandırmıştır. Kümeleme algoritması pozitif, negatif ve nötr vektör kümeleri ile benzerlik kıyaslamaları yaparak çalışmaktadır. Örnek kümelerinin daha iyi seçimi ve örnek sayısının artırımı ile daha iyi sonuçlar alınacağı değerlendirilmektedir. Çizelge 5'te görüldüğü gibi 573,794 adet mikroblog metni 33,832 pozitif, 44,648 negatif, 495,314 nötr olarak sınıflandırmıştır. Dağıtık Skorlama ve dağıtık k-means kümeleme algoritmaları ile sınıflandırma yaklaşık olarak aynı sonuçları vermektedir. Kümeleme algoritması vektör atama işlemi yaparak çalıştığından daha yavaş çalışmaktadır. Performans kritik durumlarda skorlama algoritması kullanımı tercih edilmektedir. Her iki algoritmada da daha iyi örnek seçimi ve daha kapsamlı duygusal sözlük oluşturulması ile sonuçlar iyileştirilebilmektedir.

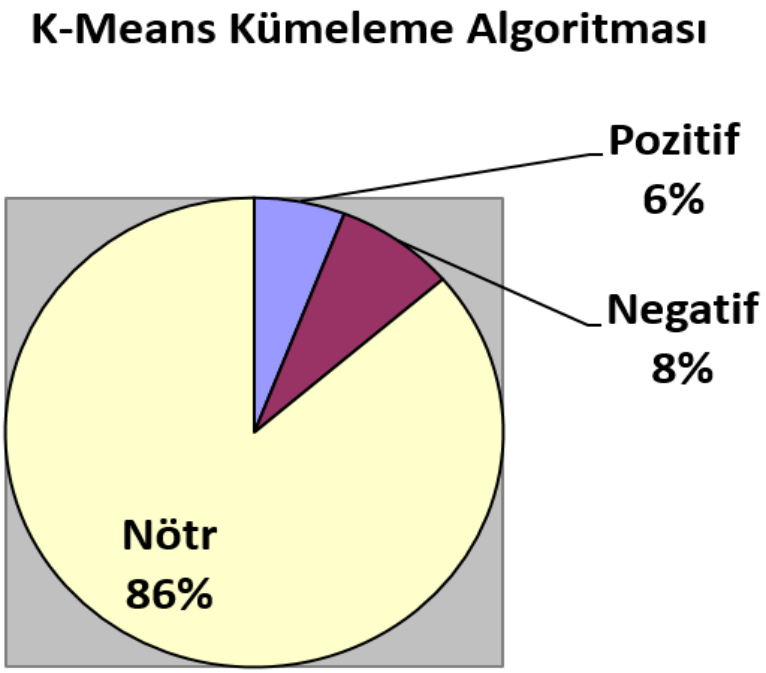



Şekil 10. Dağıtık Skorlama algoritması analiz sonuçları.

Mikroblog metin sayısı: 573,794 pozitif mikroblog metin sayısı: 44,612 negatif mikroblog metin sayısı: 35,373 notr mikroblog metin sayısı: 493,809

Şekil 10'da görüldüğü gibi metinler skorlama algoritması ile pozitif, negatif ve nötr olarak sınıflandırılmıştır. Toplam 573,794 mikroblog metinden 44,612 tanesi pozitif, 35,373 adeti negatif, 493,809 adedi nötr olarak sınıflandırılmıştır.

Dağıtık skorlama algoritmasında mikroblog metinler sözlük temelli olarak sınıflandırılmaktadırlar. Sözlük te her kelime için duygu puanı manuel olarak belirlenmektedir. Sözlük kapsamına ve genişliğine göre veri kümesi içerisindeki metinlerin pozitif, negatif nötr olarak sınıflandırma ve algılama oranı değişim göstermektedir. Veri kümesi içerisinde geçen kelimelerin algılanabilmesi için her kelimenin mümkün olan kombinasyonlarını sözlük kapsamına almak gerekmektedir. Güncel metin depolama sistemlerinde bulanık (fuzzy) arama sağlanmadığından bunu manuel olarak yapmak gerekmekte ve vakit almaktadır. Veri kümesi için belirlenen kombinasyonlar ile toplam %8 pozitif, %6 negatif, %86 nötr olarak algılanmıştır. Sosyal medya ortamındaki veri kirliliği fazla olduğundan otomatik algılanamayan metinler için manuel olarak sözlük tanımlamak gerekmektedir. Sözlük manuel olarak iyileştirildikçe Metinler daha iyi algılanmakta ve daha iyi sonuçlar elde edilmektedir.




Şekil 11. Dağıtık Kümeleme algoritması analiz sonuçları.

Mikroblog metin sayısı: 573,794 pozitif mikroblog metin sayısı: 33,832 negatif mikroblog metin sayısı: 44, 648 notr mikroblog metin sayısı: 495,314

Şekil 11'de görüldüğü gibi metinler kümeleme algoritması ile pozitif, negatif ve nötr olarak sınıflandırılmıştır. Toplam 573,794 mikroblog metinden 33,832 tanesi pozitif, 44,648 âdeti negatif, 495,314 adedi nötr olarak sınıflandırılmıştır. Dağıtık kümeleme algoritmasında sınıflandırma için örnek kümeleri manuel olarak oluşturulmaktadır. Mikroblog metinler içerisinden seçilen metinler içerisinden pozitif, negatif ve nötr kümeler için 100'er adet örnekler seçilmiştir. Her mikroblog metnin bu örneklere benzerlikleri tespit edilmekte ve en benzer olduğu sınıfa ataması yapılmaktadır. Küme içerisinde örnek seçimleri manuel olarak yapıldığından vakit almaktadır. Metinlerin otomatik olarak sınıflandırılması bu örnekler üzerinden yapılmaktadır.

Sosyal medyadaki veri kirliliğinden dolayı uygun örneklerin bulunması ve her metin içerisinde geçen kelimelerin kombinasyonlarının sisteme tanımlanması veri kümesi için özelleştirilmesi gerekmektedir. Metin analiz sistemlerinin bulanık (fuzzy) arama özellikleri kısıtlı olduğundan bu kombinasyonlar manuel olarak sisteme tanıtılmaktadır. Veri kümesi için özelleştirilmiş örnekler ve belirlenen kombinasyonları ile %6 pozitif, %8 negatif ve %86 nötr olarak algılanmıştır. Örneklerin daha iyi seçilmesi ve kelime kombinasyonlarının daha çok tanımlanması ile daha iyi sonuçlar elde edilebilmektedir.

K-means algoritması skorlamadan farklı olarak %6 pozitif, %8 negatif algılamıştır. Örnek kümelerinin seçimi ve kelime kombinasyonlarının belirlenmesine göre sonuçlar farklılık göstermektedir ancak yakın değerler elde edilebilmektedir. Daha iyi örnek kümesi oluşturulması ve kelime kombinasyonları belirlenmesi ile daha iyi sonuçlar elde edilebilmektedir.

Skorlama ve k-means kümeleme algoritmaları doğruluk performansı örnek seçimi ve sözlük oluşturma doğruluğu ile doğru orantılı olarak artmaktadır. Skorlama algoritması için sözlük oluşturma işlemi otomatik yapılamamaktadır. Her sözcük için puan değerleri atanmaktadır. Bu değerlerin doğruluğu insan hesaplaması yardımı ile belirlenmektedir. Bu süreçlerin hızlandırılması için analistlerden yardım alınmaktadır. K-Means kümeleme algoritmasında pozitif, negatif, nötr örnek seçimleri insan hesaplaması yardımı ile yapılmaktadır. Örnek sayısı ve doğruluğu arttıkça algoritmaların doğruluğu da artmaktadır.

Algoritmaların doğrulanması için mikroblog metinler arasından 500 adet rastgele örnek metinler seçilmiştir. Seçilen her metin Skorlama ve K-means kümeleme algoritmalarının verdiği sonuçlar ile karşılaştırılmıştır. Çizelge 6'da görüldüğü gibi doğruluk yüzdeleri verilmiştir. Mikroblog metinlerin otomatik sınıflandırılması için yapılan çalışmalar ve sektörden alınan talepler doğrultusun da otomatik sınıflandırma için %70 civarı kabul edilebilir olarak değerlendirilmektedir. Örneğin Çetin ve Amasyalı çalışmalarında kullanmış oldukları farklı metotlar ile %60 civarında doğrulukla otomatik sınıflandırma yapmaktadırlar [49].

Skorlama algoritması sözlük oluşturma temelli çalıştığından veri kümesi içerisinde geçen kelimeler sisteme daha hızlı tanıtılabilmektedir. Veri kümesindeki kelime çeşitliliği incelenerek sözlük iyileştirilebilmektedir. Kümeleme algoritmasında sözlükler kümeye benzerliklerine göre sınıflandırılmaktadır. Vektörel olarak benzerlikler tespit edildiği için işlem daha yavaş yapılabilmektedir. Sonuçları iyileştirmek için veri kümesi içerisinde pozitif, negatif, nötr örnekleri iyileştirmek gerekmektedir. Bu neden skorlama algoritmasında sonuçları iyileştirme daha hızlı yapılabilmekte ve daha iyi sonuçlar alınabilmektedir. Sınıflandırılan mikroblog metinlerin otomatik analiz sonuçları kabul edilebilir değerlerde olduğundan her iki algoritma da kabul edilebilir doğrulukta çalışmaktadırlar.

Çizelge 6. Dağıtık Skorlama ve Dağıtık K-means Kümeleme algoritmalarının doğruluk karşılaştırmaları.

| **Algoritma** | **Örnek Mikroblog Metin Sayısı** | **Doğru Sınıflandırılan Mikroblog Metin Sayısı** | **Doğruluk Oranı** | **Ortak Doğru Sınıflandırılan Mikroblog Metin Sayısı** | **Ortak Doğruluk Oranı** |
|---|---|---|---|---|---|
| **Skorlama** | 500 | 454 | %91 | 406 | %81 |
| **K-Means** | 500 | 429 | %86 | 406 | %81 |

# 6. SONUÇ ve GELECEK ÇALIŞMALAR

Teknoloji ve sosyal medyanın hızlı gelişimiyle veri hızı, hacmi ve çeşitliliği artış göstermektedir. Biriken veriye anlık olarak erişim ve karar desteği sağlanması mevcut teknolojiler ile mümkün olmamaktadır. Toplanan verilerin anlık olarak analiz edilmesi ve metin verilerinden bilgi çıkarımları standart veri tabanları ile yapılamamaktadır. Mevcut çözüm ve yöntemler de Türkçe metin için kısıtlı analiz yetenekleri bulunmaktadır. Bu çalışmada veri yoğun, işlemci yoğun uygulamalar için özelleştirilmiş dağıtık analitik sistem ve uygulamaları geliştirilmektedir.

Bu sistemde dağıtık dosya sistemlerinin kullanımı ile performans iyileştirmeleri yapılmıştır. Tasarlanmış olan tek düğümlü ve çok düğümlü sistemlerde performans iyileştirmeleri gözlemlenmiştir. Dağıtık analitik sistemin dağıtık dosya sistemleriyle tasarlanmasıyla hızlı sonuçlar elde edilebileceği gözlemlenmiştir. Mikroblog metin analitiği için özelleştirilmiş platformda farklı algoritmaların performans ve doğruluk değerlendirmeleri yapılmıştır. Mikroblog metin analitiği için dağıtık skorlama algoritmasının k-means kümeleme algoritmasına göre daha hızlı çalıştığı gözlemlenmiştir. Metin analitiği için geliştirilmiş dağıtık algoritmalar tek düğümlü ve çok düğümlü sistemlerde performans olarak karşılaştırılmıştır. Küme performansında bellek kısıtlarının kritikliği gözlemlenmiş ve sistemin bellek ihtiyaçları değerlendirilmiştir.

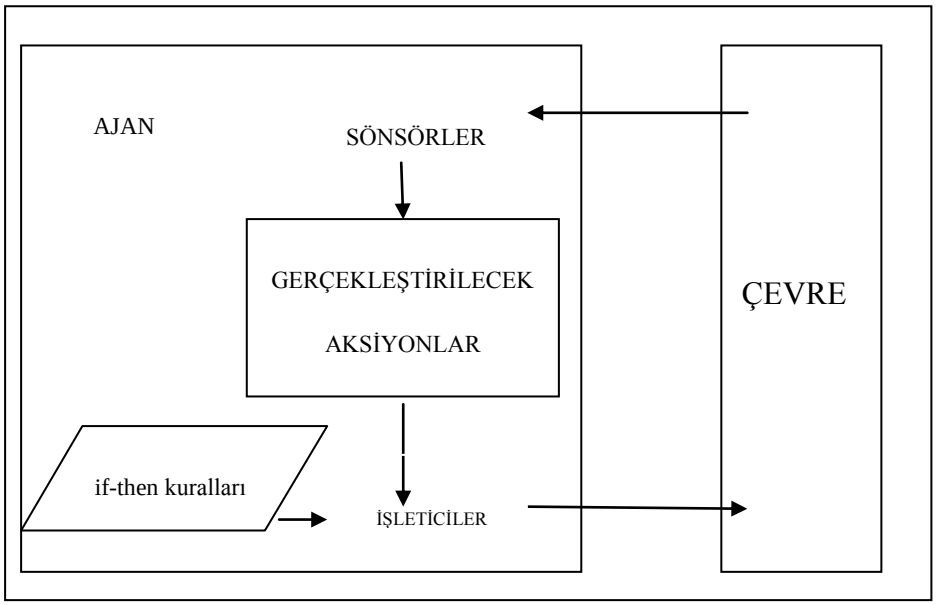
Skorlama algoritması ve K-means kümeleme algoritması doğruluk olarak yaklaşık aynı sonuçları vermektedirler. Skorlama algoritması sözlük tabanlı çalışmaktadır. Sözlük vektöründe eleman sayısı ve kelime kombinasyonları arttırıldıkça daha iyi sonuçlar vereceği değerlendirilmektedir. Skorlama algoritması sözlük temelli çalıştığından sözlükteki kelime vektörünün eleman sayısı ve kelime kombinasyonlarının daha ayrıntılı seçimi ile daha iyi sonuç verebilmektedir. K-means kümeleme algoritması pozitif, negatif, nötr vektörlerin benzerliklerine göre kıyaslamalar gerçekleştirdiğinden örnek sınıf vektörlerinin seçimine göre benzerlik kıyaslamalarını daha iyi gerçekleştirebilmektedir.

Geliştirilen dağıtık analitik sistem sayesinde büyük verinin hızlı sorgulanmasına imkân sağlanmaktadır. Uygulamalar için jenerik ve ölçeklenebilir depolama katmanları sağlanmaktadır. Dağıtık analitik uygulamalar için dağıtık mimari kullanımı önerilmektedir. Dağıtık dosya sistemlerinin ölçeklenebilir otomatik düğüm ekleme çıkarma özellikleri sayesinde donanımlar maksimum verimlilikte kullanılmakta ve ölçekleme minimum donanım ve zaman maliyeti ile yapılabilmektedir. Sonuç olarak, dağıtık dosya sistemlerinin özelleştirilmiş analitik bulut mimariler üzerinde analitik işlemler için önemli performans iyileştirmeleri sağladığı ve analitik işlemler için verimliliği arttırdığı gözlemlenmiştir.

Gelecek çalışmalar kapsamında büyük verinin interaktif sorgulanmasını sağlayan bellek merkezli dağıtık dosya sistemleri üzerinde analitik platform oluşturma işlemleri gerçekleştirilecektir. Görsel analitik uygulamalarının büyük veri üzerinde anlık olarak uygulanabilmesi dağıtık ön bellekleme (distributed caching) mekanizmaları ile mümkün olmaktadır. Büyük veriye bellek hızında erişimin sağlanması sayesinde uygulamalar için dinamik olarak veri yönetimi sağlayan ajanlar tasarlanabilmektedir.

**Akıllı Ajanlar**

Akıllı ajan çevresini sensörleri vasıtasıyla gözetleyen ve işleticileri vasıtasıyla çevresinde hareket eden otonom varlıklardır. Aktivitelerini amaçlarını başarma başarma doğrultusunda modifye eder ve ona göre düzenlemelerde bulunurlar. Akıllı ajanlar tecrübeler ile öğrenebilirler ve öğrendikleri bilgileri amaçlarını gerçekleştirmek için kullanabilirler. Şekil 12'de görüldüğü gibi akıllı ajanlar, algılayıcı ve işleticilerden oluşurlar. Algılayıcı bilgilerine göre işletim senaryolarını gerçekleştirirler.




Şekil 12. Akıllı ajan genel yapısı

Akıllı ajanlar uygulama ve alana göre özel olarak tasarlanırlar. Ajan temelli olarak geliştirilen yapay zekâ algoritmaları ile uygulamalar veriyi hızlı analiz edip anlık karar verebilmektedirler. Karar destek sistemlerinin ajan temelli olarak tasarlanmasıyla sistemlerin otonom karar vermeleri sağlanabilmektedir. Uygulama alanlarına göre özelleştirilecek ajan yapıları ile büyük veri üzerinde karar desteği sağlayan otonom sistemler geliştirilecektir.

Büyük veri üzerinde çalışan interaktif görsel analitik uygulamalar, ajan temelli otonom sistemler ile büyük veriyi bilgi ve veri tabanı olarak kullanabilmektedirler. Uygulama ihtiyaçlarına göre özelleştirilecek ajan yapıları ile platform üzerinde dağıtık analitik uygulamalar geliştirilecektir. SQL sorgulama, çizge (sorgulama), makine öğrenmesi (machine learning), akan veri analizleri (stream processing) iskeletleri (framework) ve ihtiyaca göre eklenecek diğer iskeletler ile sistemin analitik kabiliyetleri geliştirilecektir.

Güvenlik kritikliği durumuna göre public (genel) veya private (özel) bulut mimarileri kullanılarak uygulama güvenliği sağlanacaktır. Uygulamada ajanlar arası güvenli iletişim için TrustZone [41] vb. teknolojiler kullanılacaktır. Bu teknolojiler sayesinde analitik uygulamalar veri merkezleri ile güvenli iletişim sağlayabilmektedirler. Özel ve genel bulut mimarilerinin IaaS (Infrastructure as a Service), PaaS (Platform as a Service) ve SaaS (Software as a Service) katmanları ile ölçeklenebilir depolama, platform ve uygulama hizmeti sunabilmektedir. Bu katmanların amaca göre özelleştirilmeleri ile amaca yönelik analitik sistemler geliştirilebilmektedir. Gelecek çalışmalarda kullanım amaçlarına göre yapılacak özelleştirmelerle alana özel dağıtık analitik bulut sistemleri geliştirmeleri yapılacaktır.

## KAYNAKLAR


[1] Keim, Daniel A., et al. Visual analytics: Scope and challenges. Springer Berlin Heidelberg, 2008, pp. 76-90.

[2] Thelwall, Mike, Kevan Buckley, and Georgios Paltoglou. "Sentiment strength detection for the social web." Journal of the American Society for Information Science and Technology 63.1, 2012 , pp. 163-173.

[3] Özsert, Cüneyd Murad, and Arzucan Özgür. "Word polarity detection using a multilingual approach." Computational Linguistics and Intelligent Text Processing. Springer Berlin Heidelberg, 2013, pp. 75-82.

[4] KHUC, Vinh Ngoc, et al. Towards building large-scale distributed systems for twitter sentiment analysis. In: Proceedings of the 27th annual ACM symposium on applied computing. ACM, 2012. p. 459-464.

[5] LIN, Jimmy; KOLCZ, Alek. Large-scale machine learning at twitter. In: Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data. ACM, 2012. p. 793-804.

[6] Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis.Foundations and trends in information retrieval, 2(1-2), 1-135.

[7] Lee, K. H., Lee, Y. J., Choi, H., Chung, Y. D., & Moon, B. (2012). Parallel data processing with MapReduce: a survey. ACM SIGMOD Record, 40(4), 11-20.

[8] ZHAO, Weizhong; MA, Huifang; HE, Qing. Parallel k-means clustering based on mapreduce. In: Cloud Computing. Springer Berlin Heidelberg, 2009. p. 674-679.

[9] GO, Alec; BHAYANI, Richa; HUANG, Lei. Twitter sentiment classification using distant supervision. CS224N Project Report, Stanford, 2009, 1: 12.

[10] Zhou, P., Lei, J., & Ye, W. (2011). Large-Scale Data Sets Clustering Based on MapReduce and Hadoop. Journal of Computational Information Systems, 7(16), 5956-5963.

[11] BERMINGHAM, Adam; SMEATON, Alan F. Classifying sentiment in microblogs: is brevity an advantage?. In: Proceedings of the 19th ACM international conference on Information and knowledge management. ACM, 2010. p. 1833-1836.

[12] O'Connor, B., Balasubramanyan, R., Routledge, B. R., & Smith, N. A. (2010). From tweets to polls: Linking text sentiment to public opinion time series. The International Conference on Weblogs and Social Media (ICWSM), 11, 122-129.

[13] Martínez-Cámara, Eugenio, M. TERESA MARTÍN-VALDIVIA, L. ALFONSO UREÑA-LÓPEZ, and A. RTURO MONTEJO-RÁEZ. "Sentiment analysis in twitter." Natural Language Engineering 20, no. 01 (2014): 1-28.

[14] PAK, Alexander; PAROUBEK, Patrick. Twitter as a Corpus for Sentiment Analysis and Opinion Mining. In: Language Resources and Evaluation Conference (LREC). 2010. p. 1320-1326.

[15] Kouloumpis, E., Wilson, T., & Moore, J. (2011). Twitter sentiment analysis: The good the bad and the omg!. The International Conference on Weblogs and Social Media (ICWSM), 11, 538-541.

[16] SEPPELT, R. Z. M.; BLOCK, C.; RETSIOS, V. Geovisual analytics of Satellite Image Time Series. In: International Congress on Environmental Modelling and Software. 2012.

[17] Hao Zhang, Gang Chen, Beng Chin Ooi, Kian-Lee Tan, Meihui Zhang: In-Memory Big Data Management and Processing: A Survey. IEEE Trans. Knowl. Data Eng. 27(7): 1920-1948

[18] D. Zhao, X. Zhou, et al. "FusionFS: Toward supporting data-intensive scientific applications on extreme-scale high-performance computing systems." Big Data (Big Data), 2014 IEEE International Conference on. IEEE, 2014.

[19] ZINN, Daniel, et al. Streaming satellite data to cloud workflows for on-demand computing of environmental data products. In: Workflows in Support of Large-Scale Science (WORKS), 2010 5th Workshop on. IEEE, 2010. p. 1-8.

[20] "Skybox şirketi erişim linki", http://www.skyboximaging.com/technology, erişim tarihi: 25 Ağustos 2015.

[21] K. Chang, K. Choi. "Memory-centric communication architecture for reconfigurable computing." Reconfigurable Computing: Architectures, Tools and Applications. Springer Berlin Heidelberg, 2010. 400-405.

[22] A. Beric, V. Meerbergen, G. Haan, R. Sethuraman, "Memory-centric video processing." Circuits and Systems for Video Technology, IEEE Transactions on 18.4 (2008): 439-452.

[23] G. Yao, R. Pellizzoni, S. Bak, E. Betti, M. Caccamo, "Memory-centric scheduling for multicore hard real-time systems." Real-Time Systems 48.6 (2012): 681-715.

[24] Hitz, Dave, James Lau, and Michael A. Malcolm. "File System Design for an NFS File Server Appliance." USENIX winter. Vol. 94. 1994. 50-55.

[25] Callaghan, Brent. "WebNFS server specification." (1996).

[26] Callaghan, Brent. "WebNFS-The Filesystem for the Internet." (1997).

[27] Callaghan, Brent. "WebNFS server specification." (1998).

[28] Howard, John H. An overview of the andrew file system. Carnegie Mellon University, Information Technology Center, 1988.

[29] SCHMUCK, Frank B.; HASKIN, Roger L. GPFS: A Shared-Disk File System for Large Computing Clusters. In: FAST. 2002. p. 19.

[30] GHEMAWAT, Sanjay; GOBIOFF, Howard; LEUNG, Shun-Tak. The Google file system. In: ACM The Special Interest Group on Operating Systems (SIGOPS) operating systems review. ACM, 2003. p. 29-43.

[31] Dean, Jeffrey, and Sanjay Ghemawat. "MapReduce: simplified data processing on large clusters." Communications of the ACM 51.1 (2008): 107-113.

[32] SHVACHKO, Konstantin, et al. The hadoop distributed file system. In: Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on. IEEE, 2010. p. 1-10.

[33] LI, Haoyuan, et al. Tachyon: Reliable, memory speed storage for cluster computing frameworks. In: Proceedings of the ACM Symposium on Cloud Computing. ACM, 2014. p. 1-15.

[34] ENGLE, Cliff, et al. Shark: fast data analysis using coarse-grained distributed memory. In: Proceedings of the 2012 ACM Special Interest Group on Management Of Data (SIGMOD) International Conference on Management of Data. ACM, 2012. p. 689-692.

[35] XIN, Reynold S., et al. Graphx: A resilient distributed graph system on spark. In: First International Workshop on Graph Data Management Experiences and Systems. ACM, 2013. p. 2.

[36] KRASKA, Tim, et al. MLbase: A Distributed Machine-learning System. In: CIDR. 2013. p.5-10.

[37] Z. Matei, et al. Discretized streams: A fault-tolerant model for scalable stream processing. No. UCB/EECS-2012-259. CALIFORNIA UNIV BERKELEY DEPT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE, 2012.

[38] Likhachev, Maxim, et al. "Anytime search in dynamic graphs." Artificial Intelligence 172.14 (2008): 1613-1643.

[39] LIKHACHEV, Maxim, et al. Anytime Dynamic A*: An Anytime, Replanning Algorithm. In: The International Conference on Automated Planning and Scheduling (ICAPS). 2005. p. 262-271.

[40] “HDFS için Yahoo firması tarafından yapılan değerlendirmelerin erişim adresi”, https://developer.yahoo.com/blogs/hadoop/scalability-hadoop-distributed-file-system-452.html, erişim tarihi: 25 Ağustos 2015.

[41] ”Güvenli veri iletişimi için kullanılan Trust Zone teknolojisi erişim adresi”, http://www.arm.com/products/processors/technologies/trustzone/index.php, erişim tarihi: 25 Ağustos 2015.

[42] “Dağıtk hesaplama için kullanılan MapReduce algoritması erişim adresi”, http://web.cs.wpi.edu/~cs4513/d08/OtherStuff/MapReduce-TeamA.ppt, erişim tarihi: 25 Ağustos 2015.

[43] “D3 Java Script kütüphanesi erişim adresi”,http://d3js.org/, erişim tarihi: 25 Ağustos 2015.

[44] ”Büyük veri sistemleri entagrasonu için ontoloji kullanımı erişim linki”, https://en.wikipedia.org/wiki/Ontology-based_data_integration , erişim tarihi: 25 Ağustos. 2015.

[45] ”Büyük veri sistemlerinde kullanım için geliştirilmiş örnek ontoloji erişim linki”, http://stick.ischool.umd.edu/newsite/innovation_ontolgy, erişim tarihi: 25 Ağustos 2015.

[46] ”Dinamik ontoloji tasarımı için geliştirilmiş yazılım aracı erişim linki”, http://www.salzburgresearch.at/en/projekt/dynamont_en/, erişim tarihi: 25 Ağustos 2015.

[47] ”Duygusal sözlük kütüphanesi erişim linki”, http://sentiwordnet.isti.cnr.it/, erişim tarihi: 25 Ağustos 2015.

[48] ”Duygusal vektörler için kullanılan K- Means Dağıtık kümeleme algoritması erişim linki”, https://en.wikipedia.org/wiki/K-means_clustering, erişim tarihi: 25 Ağustos2015.

[49] ÇETIN, M.; AMASYALI, M. F. Eğiticili ve Geleneksel Terim Ağırlıklandırma Yöntemleriyle Duygu Analizi. In: Proceedings of Signal Processing and Communications Applications Conference (SIU). 2013.

# ÖZGEÇMİŞ

## Kişisel Bilgiler


Soyadı, Adı : AĞCA, Muhammed Akif
Uyruğu : T.C.
Doğum tarihi ve yeri : 17.09.1988 Ankara
Medeni hali : Bekar
Telefon : 0 (505) 899 57 71
Faks : 0 (312) 292 40 91
e-mail : akif.agca@etu.edu.tr, agca.akif@gmail.com


## Eğitim

| Derece | Eğitim Birimi | Mezuniyet tarihi |
|---|---|---|
| Yüksek Lisans | **TOBB Ekonomi ve Teknoloji Üniversitesi/** Bilgisayar Mühendisliği | 2012 - 2015 |
| Lisans | **Ortadoğu Teknik Üniversitesi/** Bilgisayar ve Öğretim Teknolojileri Eğitimi | 2006 - 2011 |

## İş Deneyimi

| Yıl | Yer | Görev |
|---|---|---|
| 2014 –Halen | HAVELSAN | Yazılım Mühendisi |

## Yabancı Dil

İngilizce, İleri
Arapça, Orta
Almanca, Başlangıç

**Yayınlar**

- MEMCA [Memory CentricAnalytics], VLDB 2015, **On Progress**

- **Muhammed Akif Ağca**, Emre Başeski, Serhan Gökçebağ "MEMCA [Memory CentricAnalytics] forSatelliteand Space Data", 7th International Conference on Recent Advances in Space Technologies-RAST2015.

- Emre Başeski, Serhan Gökçebağ, Alim Rüstem Aslan, Osman Ceylan, Ahmet Erdem, Şaban Gökhan Erbay, Mücahit Akyol, Kamil Arslankoz, İsmail Arslan, **Muhammed Akif Ağca**, Yusuf Burak Aydın, " HAVELSAT: A Software Defined Radio Experimentation CubeSat ", 7th International Conference on Recent Advances in Space Technologies-RAST2015.

- **M. Akif, Ağca**, Şenol Ataç, M. Mert Yucesan, Gokhan Y. Kucukayan, A. Murat Özbayoglu and Erdoğan Doğdu, "Opinion Mining of Microblog Texts on Hadoop Ecosystem", International Journal of Cloud Computing (in press).

- **Muhammed Akif Ağca**, Şenol Ataç, M. Mert Yucesan, Gokhan Y. Kucukayan, A. Murat Özbayoğlu, Erdoğan Doğdu, "Opinion Mining of Microblog Texts on Hadoop Ecosystem", 2nd IBM Cloud Academy Conference, ICA CON 2014.