Avrupa Bilim ve Teknoloji Dergisi
Sayı 13, S. 17-30, Ağustos 2018

Derleme Makale

www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
No. 13, pp. 17-30, August 2018

Review Article

Akan Veri Kümeleme Teknikleri Üzerine Bir Derleme

Ali Şenol[1*], Hacer Karacan[2]
[1]Ardahan Üniversitesi Yenisey Kampüsü, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, 75002, Ardahan, alisenol@ardahan.edu.tr
[2]Gazi Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, 06570 Maltepe Ankara, hkaracan@gazi.edu.tr

(İlk Geliş Tarihi 19 Temmuz 2018 ve Kabul Tarihi 27 Ağustos 2018)

(DOI: 10.31590/ejosat.446019)

Öz

Günümüz teknolojisinin gelişmesine paralel olarak bilgisayar ortamına aktarılmış olan veri miktarı inanılmaz boyutlara ulaşmış ve gün geçtikçe de artmaktadır. Bu nedenle veriyi işleme yöntemleri de değişmektedir. Klasik kümeleme yaklaşımlarında veri statiktir. Oysa günümüz teknolojisinde, verinin çok hızlı olduğu dünyada artık veriyi akarken kümeleyecek, kullanıcıya istediği zaman sonuç verebilecek uygulamalara ihtiyaç vardır. Bu anlamda ihtiyacı karşılayan akan veri kümeleme yaklaşımlarına olan talep gün geçtikçe artmaktadır. Çünkü akan veri kümeleme yaklaşımları bir defa okumalı, hızlı ve kendisini yeni gelen veriye uyarlama özelliğine sahiptir. Yani veri bir yandan akarken bir yandan kullanıcıya sonuç üretilebilmektedir. Bu çalışmada akan veri kümeleme alanında yapılan çalışmalar derlenmekte ve bu alana ilgi duyan araştırmacılara ışık tutulmaktır.

Anahtar Kelimeler: Akan veri kümeleme, akan veri kümeleme teknikleri, akan veri kümeleme alanındaki ihtiyaçlar

A Survey on Data Stream Clustering Techniques

Abstract

In parallel with the development of today's technology, the amount of data that has been transferred to the computer environment has reached incredible dimensions and is increasing day by day. For this reason, the methods of data processing are also changing. In classical data clustering approaches, data is static. However, in today's technology in which data streams very fast, there is a need for applications that can cluster data and show results while the data is streaming whenever the user wants. In this sense, the demand for data stream clustering approaches is increasing day by day. Because, the data stream clustering approaches read once, fast, and have the ability to adapt themselves to new data. In other words, the results are shown to the user on the one hand, while the data is streaming on the other hand. In this study, the proposed studies on the data stream clustering area are collected and the researchers who are interested in this field are enlighten.

Keywords: Data stream clustering, data stream clustering techniques, demands in data stream clustering area

1. Giriş

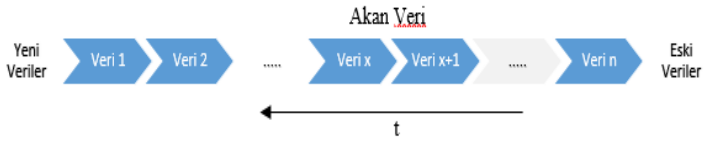
Teknolojinin hızlı bir şekilde gelişmesine paralel olarak bilgisayar, akıllı telefon veya sensor gibi veri üreten cihazların kullanımı yaygınlaşmıştır. Bu tür cihazlar kullanılarak "Büyük Veri" olarak tanımlanan inanılmaz bir veri miktarı üretilmektedir. Üretilen veriler çok hızlı bir şekilde arttığından klasik veri madenciliği yöntemleri pek çok açıdan yetersiz kalmaktadır. Bunların başında verileri bir yere kaydetme gelmektedir. Çünkü çok büyük veri miktarını kayıt altına almak için gereken kaynak ihtiyacı da çok büyüktür. Bir diğer yetersizlik nedeni çoğu uygulama için gerçek zamanlı sonuç üretmeye olan ihtiyaçtır. Örneğin banka gibi finansal kuruluşlar için analizlerin anlık yapılabilmesine ihtiyaç vardır. Çünkü dolandırıcılık gibi durumlar için 1 dk bile oldukça geç olabilir. Bu ve buna benzer nedenlerden dolayı gerçek zamanlı analizler üretebilecek uygulamalara olan ihtiyaç günden güne artmaktadır

Klasik veri kümeleme algoritmaları var olan bir veri seti üzerinde işlem yaparak bu veri setini kümelere ayırmaktadır. Kaç sınıf olacağını ya kullanıcıdan girdi olarak alır ya da rastgele belirleyerek işlem yapmakta ve sonuçta bu girdilere göre her zaman doğruluğu kesin olmayan bir kümeleme yapmaktadır. Yeni bir veri geldiğinde, dolayısıyla veri seti değiştiğinde kümeleme işlemini baştan yapar. Oysaki veri seti dinamik bir şekilde sürekli değişiyor da olabilir. Bu nedenle bu verileri dinamik bir şekilde, yeni veri geldikçe sonucu güncelleyecek yaklaşımlara ihtiyaç duyulmaktadır.

Son yıllarda akan veri üzerinde kümeleme yapmaya yönelik çeşitli çalışmalar yapılmış ve bu konu gün geçtikçe ilgi çekmektedir. Akan veriyi kümeleme olarak çevirebileceğimiz *data stream clustering* dinamik bir şekilde değişen veriyi kümelemeye yönelik yapılan çalışmalardır. Bu çalışmada akan veri kümeleme alanında yapılan çalışmalar derlenerek bu alanda çalışmayı düşünen araştırmacılara ışık tutmak hedeflenmektedir. Makalenin geri kalanı şu şekilde sıralanmaktadır. 2. bölümde akan verinin ne olduğu açılanmakta, 3. bölümde akan veri kümeleme alanında karşılaşılan problemler sıralanmakta ve 4. bölümde akan veri kümeleme yaklaşımlarının uygulama alanları üzerinde durulmaktadır. 5. bölümde bu alanda yapılan çalışmalarda kullanılan veri setleri açıklanmakta ve 6. bölümde veri özetleme metodolojileri üzerinde durulmakta iken 7. bölümde akan veri kümeleme yaklaşımlarının kümeleme başarısı değerlendirme yöntemleri açılanmaktadır. 8. bölümde akan veri kümeleme alanında yapılan çalışmaların sınıflandırılması yapılmakta ve 9. bölümde başlıca akan veri kümeleme yaklaşımları açıklanarak karşılaştırmaları yapılmaktadır. Son bölümde ise bu alanda yeni çalışmalara açık olan konular üzerinde durularak çalışma sonuçlandırılmaktadır.

2. Akan veri kümeleme

Akan veri kümeleme finansal, network izleme, telekomünikasyon veri yönetim, web, sensör ağ, meteorolojik, bilim ve mühendislik gibi pek çok alanda kullanılmaktadır [1, 2]. Adı geçen uygulamalarda yüksek hızda anlık veri akmaktadır. Geliştirilecek uygulamaların anlık olarak bu verileri kümelemesi gerekir. Geliştirilecek uygulamaların tek yönlü tarama yapan, on-line, çok seviyeli ve çok yönlü olması gerekir [1]. Şekil 1'de de görüldüğü gibi akan veride verinin sonu belli değil ve çoğunlukla da boyutu sonsuzdur. Bu nedenle veriyi bir yerde biriktirip işleme imkânı yoktur, yani sınırsız depolama ihtiyacı doğmaktadır.



Şekil 1. Akan veri örneği

Veri yüksek hızda aktığından bu veriyi kümeleyecek yaklaşımın da hızlı bir şekilde çalışması gerekmektedir. Bunun yanında akan bu verinin genellikle belli bir sınırı yoktur. Başı ve sonu önceden bilinemez. Dolayısıyla verinin geneli hakkında bilgi sahibi olma imkânı yoktur. Kısaca akan veri, boyut olarak büyük, sonsuz, devamlı ve ardışıktır [3, 4].

3. Akan veri kümeleme yaklaşımlarında karşılaşılan problemler

Pek çok açıdan bakıldığı zaman akan veri kümeleme klasik veri kümeleme yaklaşımlarından farklıdır. Geleneksel kümeleme yaklaşımları ve akan veri kümeleme yaklaşımları arasındaki farkları şu şekilde sıralayabiliriz [5]:

- Geleneksel yöntemde veriler statiktir; ancak akan veride veri dinamiktir, sürekli değişmektedir.
- Akan veride geleneksel yöntem gibi veriyi bir yere kaydedip tekrar tekrar işleme imkânı yoktur.
- Geleneksel yöntemde sonuçlar sabittir; ancak akan veride sonuç zamana bağlı olarak değişir.

Akan veri pek çok açıdan kısıtlayıcıdır. Geliştirilecek yöntemin kaynakları etkin kullanması ve tatmin edici sonuçları makul bir zamanda sunması ihtiyacı bunlardan bazılarıdır. Bu nedenle geliştirilecek yöntemlerin bu problemleri göz önünde bulundurması gerekir. Akan veri kümelemede en sık karşılaşılan problemleri şu şekilde sıralayabiliriz [5]:

- *Sonsuz boyut ve yüksek hız:* Akan veride verinin sonu genelde yoktur veya sınırları bilinmez.
- *Dinamizm:* Veri dinamiktir, her an değişir.
- *Veriye genel bakış:* Geleneksel kümeleme yaklaşımlarında veriye genel bir bakış attıktan sonra daha iyi sonuçlar elde edilebilmektedir; ancak akan veride verinin sonu belli olmadığından veri hakkında kestirimde bulunmak çok zordur.
- *Sapan veri:* Geleneksel kümeleme yaklaşımlarında veri ön işleme ile sapan veri olup olmadığını, var ise bunları bertaraf etme imkânı vardır; ancak akan veride bunu tespit edip düzeltmek oldukça zordur.
- *Çok boyutluluk*: Akan veride çok boyutlu veriyi işlemek geleneksel yönteme göre oldukça zordur.
- *Parametreleri belirleme*: Akan veride parametreleri belirlemek tüm veriye hâkim olunmadığından oldukça zordur.

Bir akan veri kümeleme yaklaşımının kabul görebilmesi için sağlaması gereken minimum kriterler vardır [5]. Bunlar:

- *Uyarlanabilirlik*: Akan veri kümeleme yaklaşımlarının her yeni gelen veriye göre baştan kümeleme yapmak yerine yeni gelen veriye göre kendisini uyarlayabilmesi gerekir.
- *Veriyi bir defa işleme*: Akan veri kümeleme yaklaşımlarının veriyi tekrar tekrar işlemek gibi bir imkânı olmadığından bir defa işleyerek sonuç üretebilmesi gerekir.
- *Zaman kısıtı ve alan kriteri*: Veri sürekli aktığından bu veriyi hafızaya kaydedip işleme imkânı yoktur. Bu nedenle veriyi çok kısa bir zamanda ve çok az kaynak harcayarak işlemek gerekir.
- *Kümeleme adaptasyonu (Concept Evolution)*: Veri sürekli değiştiğinden daha önce oluşturulmuş kümelerin gelen veriye göre adapte olabilmesi gerekir. Gerektiğinde var olan kümeler dışında yeni bir küme oluşturma, var olan bir kümeyi yok etme, var olan iki kümeyi birleştirme veya var olan bir kümeyi farklı sayıda kümelere ayırmak gerekebilir.
- *Devamlı kümeleme modeli*: Akan verinin sonuna kadar bekleyip kümeleme yapma imkânı yoktur. Bu nedenle veri akarken kümeleme modelinin de bir taraftan kümeleme işlemini gerçekleştirmesi gerekir. Kullanıcının istediği her an küme sonuçlarını sunabilmesi gerekir.

- *Sapan veri tespiti*: Geliştirilen kümeleme yaklaşımının sapan verileri tespit edebilmesi gerekir. Çünkü sapan veriler kümeleme başarısını doğrudan etkilemektedir.

4. Akan veri kümeleme yaklaşımlarının uygulama alanları

Kullanıcıya ait click verisi analizi [6], saldırı tespit sistemleri [7-9], sosyal medya [10-12], finansal uygulamalar [13], bilimsel araştırmalar [14], sağlık araştırmaları [15-17], mobil uygulamalar [18], nesnelerin interneti (IoT) [19] ve sensor ağ [20, 21] gibi pek çok alanda kullanılmaktadır. Nesnelerin interneti konusunun yaygınlaştığı günümüzde uygulama alanlarının daha da artacağını söylemek mümkündür.

Yukarıda da bahsettiğimiz gibi teknolojinin gelişmesine paralel olarak internet ortamına aktarılmış olan veri miktarı üssel olarak artmaktadır. Bu devasa verilerin anlamlı bilgiye dönüştürülmesine olan talep ve yeni ihtiyaçlar bu alanda yapılacak çalışmalara ilham vermektedir.

5. Akan veri kümeleme yaklaşımlarında kullanılan veri setleri

Akan veri kümeleme alanında yapılan çalışmaların büyük bir çoğunluğunda sentetik veriler kullanılmaktadır. Çünkü çalışmaların odaklandığı sapan veri tespiti veya çok boyutluluk gibi problemleri aynı anda sağlayan gerçek verileri bulmak çok zordur. Bu nedenle bu alanda çalışma yapan araştırmacılar genelde kendi verilerini üretmektedir. Bunun yanında performans veya başarı karşılaştırmaları için KDD-CUP '99, KDD-CUP '09 veya UCI'nin orman örtüsü verisi de sıkça kullanılmaktadır. Kullanıma açık akan veri kümeleme verilerini sunan kaynakları şu şekilde sıralayabiliriz [22]:

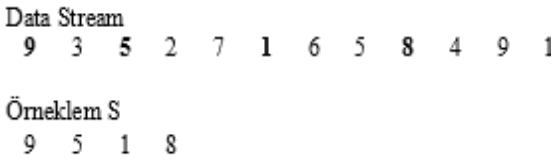
- UCI Knowledge Discovery in Databases Archive - http://kdd.ics.uci.edu
- KDD Cup Center - http://www.sigkdd.org/kddcup/
- UCR Time-Series Datasets - http://www.cs.ucr.edu/~eamonn/time_series_data
- Kaggle - https://www.kaggle.com/

Bu veri seti sunan kuruluşların dışında ABD gibi bazı ülkelerin FBI, nüfus bürosu ve sağlık bakanlığı gibi çeşitli kurum ve kuruluşları ve Amazon, Twitter, Facebook ve Instagram gibi web platformları çeşitli veri setleri sunmaktadır.

6. Akan veri kümeleme yaklaşımlarında temel veri özetleme metodolojileri

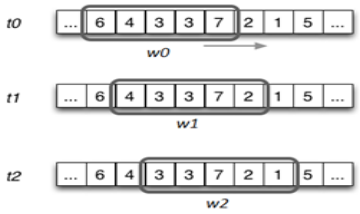
Sonsuz depolama imkânına sahip olunmadığından kümeleme başarısı ve depolama alanı arasında makul bir oranın tutturulması gerekir. Bütün veriyi depolamak ve kümeleme işlemine tabi tutmak kümeleme başarısı açısından daha iyi olabilir; ancak bu durum, büyük depolama ihtiyacı doğurmaktadır. Bu nedenle verinin tamamı yerine özetini almak hem kümeleme başarısı açısından makul bir sonuç almayı mümkün kılmakta, hem de alan açısından tasarruf sağlanmaktadır. Bu nedenle çeşitli özetleme algoritmaları geliştirilmiştir. Başlıca veri özetleme yöntemleri şunlardır:

Rastgele Örnekleme (Random Sampling): En basit özetleme yaklaşımıdır. Bu özetleme yaklaşımında belli aralıklarla örnek alınır. Bu özetleme yaklaşımının en büyük özelliği çoğu uygulama açısından kullanımı oldukça kolaydır. Şekil 2 örnek bir rastgele örnekleme örneğini göstermektedir.



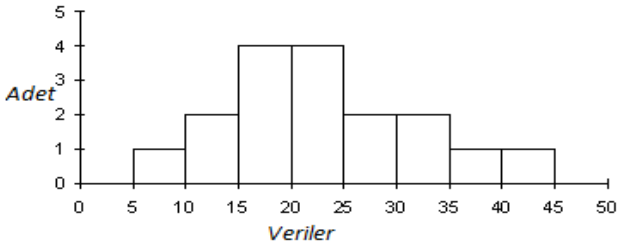
Şekil 2. Rastgele örnekleme örneği

Kayan Pencere (Sliding Window): Bu yaklaşımda örnekleme yapmak yerine sadece en son gelen veri alınır [23]. Şekil 3'te görüldüğü gibi herhangi bir *t* anında *w* kadar veri örnek olarak alınır. Stok veya sensör network uygulamaları için oldukça uygun bir yaklaşımdır. En son gelen verilerin daha önemli olduğu uygulamalar için kullanımı oldukça uygundur. Hafıza açısından oldukça tasarrufludur, sadece *w* kadar hafızaya gereksinim duyar.



Şekil 3. Sliding window örnekleme örneği

Histogramlar: Şekil 4'te de görüldüğü gibi veriye ait değerler için frekans değerlerine göre özet çıkaran bir yaklaşımdır. Bu yaklaşımda veriler kova (bucket) denilen parçalara bölünür. Kovaların boyutu gerekli hafıza miktarını belirler. Dezavantajı geriye dönük işlem yapamamasıdır.



Şekil 4. Histogram örnekleme örneği

Çok Çözünürlüklü Metodlar (Multiresolution Methods): Büyük veri miktarı ile baş etmek için geliştirilmiş bir yöntemdir. Bu yaklaşımda veri miktarını azaltmak için parçala ve fethet yaklaşımı izlenir. Bu yaklaşımın en önemli avantajı veriyi yönetme anlamında doğruluk ve kaynak açısından optimum sonuç vermesidir. Ayrıca veriyi farklı açılardan anlamaya yardımcı olur. Örnek olarak dengeli ikili ağaçlarını ele alırsak kökten yapraklara doğru indikçe her adımda sonuç detaylandırılmış olur. Bu yaklaşım da buna benzemektedir. İki çeşit çok çözünürlüklü metot vardır. Bunlar

a. *Mikro-kümeler (Micro-clusters)* [24]: En çok kullanılan özetleme yaklaşımlarından biridir. En büyük avantajı çok boyutlu veriler için oldukça kullanışlı olmasıdır.

b. *Dalgacık (Wavelets)* [25]: Dalgacık özetleme, veritabanlarında hiyerarşik verilerin ayrıştırılmasında ve özetlenmesinde sıkça kullanılan bir yöntemdir. Temelde hiyerarşik verileri ayrıştırmak için kullanılan bir yaklaşımdır.

Taslak (Sketches): En doğruya yakın bir cevabı elde etmeyi garanti eden bir yaklaşımdır. Veriyi bir defa işleme mantığına göre çalışır. Veriye ait sıklık momenti (frequency moment) sketch denilen özetler ile elde edilir.

Rastgele Hale Getirilmiş Algoritmalar (Randomized Algorithms): Çok boyutlu ve büyük verileri işlemek için geliştirilmiş bir yaklaşımdır. Rastgele örnekleme ve taslak yaklaşımlarının birleşiminden oluşur.

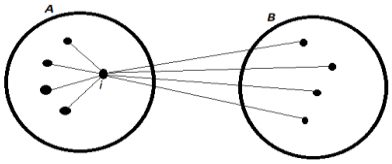
7. Akan veri kümeleme yaklaşımlarında kümeleme başarısını değerlendirme yöntemleri

Kümeleme yapan bir model ortaya konduktan sonra bu modelin başarısının ölçülmesi gerekmektedir. Akan veri kümeleme modellerinin başarasını ölçme yöntemleri yine klasik kümeleme yöntemlerini ölçme yöntemlerinde kullanılan yöntemlerle aynıdır. Purity Testi, F-Score, Accuracy, Rand index(RI), Adjusted Rand index(ARI) ve Silhouette index bu alanda kullanılan başlıca yöntemlerdir. Akan veri kümeleme modelini değerlendirirken bu parametrelerden sadece birini kullanmak başarıyı tam olarak ölçmek adına yeterli değildir. Bu nedenle bu parametrelerden birkaç tanesi kullanılmaktadır. Örneğin Purity-ARI, Purity-Accuracy-F-Score veya Purity-ARI-Silhouette İndex parametrelerinin kullanımı oldukça yaygındır. Temel anlamda kümeleme başarısı değerlendirme yöntemlerini iki gruba ayırabiliriz: Dahili ve harici yöntemler.

7.1. Dahili yöntemler

Dahili yöntemlerde veri setine ait bir sınıf etiketinin olmasına gerek yoktur. Çünkü kümeleme başarısını değerlendirmek için sınıflara atanmış veriler arasındaki benzerliklere ve diğer kümelere olan farklılığa bakar. Söz konusu bu benzerlik kriteri çoğu zaman uzaklıktır. Yani bir kümedeki veriler birbirine ne kadar yakınsa ve diğer kümelere de ne kadar uzaksa bu kümeleme modeli o kadar başarılı olarak değerlendirilir. Silhouette index ve SSE (Sum of Squarred Errors - Hataların Kareleri Toplamı), başlıca dahili küme başarısı değerlendirme yöntemleridir.

Şekil 5'te de görüldüğü gibi Silhouette index [26], her veri için iki uzaklığı baz alır. Bu uzaklıklardan ilki verinin bulunduğu kümeye ait diğer verilere olan uzaklıkların ortalamasıdır. Diğer uzaklık ise verinin ait olduğu kümeye en yakın kümenin elemanlarına olan uzaklıkların ortalamasıdır. Bu iki uzaklık üzerinden [-1, 1] aralığında bir değer hesaplar.



Şekil 5. Silhouette index örneği

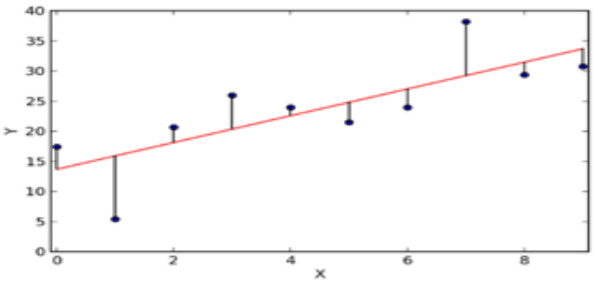
a herhangi bir i verisinin içinde bulunduğu kümedeki diğer verilere olan uzaklıkların ortalaması ve b, i verisinin en yakın olduğu kümeye ait verilere olan uzaklıkların ortalaması olmak üzere Silheouette index S_i Eş. 1 ile hesaplanır.

$$S_i = \frac{b-a}{\max(a,b)} \quad (1)$$

SSE, kendi içerisinde verilerin hata payını hesaplar. Verilerin hata paylarının karelerinin toplamını bulur. n veri sayısını, y_i, i. verinin değerini ve $\overline{y_i}$ tahminlerin ortalaması olmak üzere SSE Eş. 2 ile hesaplanır.

$$SSE = \sum_{i=1}^{n}(y_i - \overline{y_i})^2 \quad (2)$$

Şekil 6'da da görüldüğü gibi her verinin ortalama doğrusuna olan uzaklığı hata payını göstermektedir. Yani veri doğruya ne kadar yakın olursa hata paya o kadar azalır.



Şekil 6. SSE örneği

7.2. Harici yöntemler

Kümeleme barısını değerlendirirken kullanılan harici yöntemlerde temel iş modelin atandığı küme etiketinin veri setinde bulunan gerçek küme etiketleri ile karşılaştırılmasıdır.

Purity önerilen modelin yaptığı kümeleme yaklaşımının saflık derecesini hesaplar. Her küme için içerisinde barındırdığı verilerden sayısı en fazla olan verilerin toplam veri sayısına oranıdır. Bu işlemi tüm kümeler için yapar ve elde edilen sonuçların tamamını toplar. Bu toplamın toplam veri sayısına oranı Purity'dir. K küme sayısını, C_i^d, C_i kümesindeki baskın sınıfa ait veri sayısını ifade etmek üzere Purity Eş. 3 ile hesaplanır.

$$Purity = \frac{\sum_{i=1}^{K} \frac{|C_i^d|}{|C_i|}}{K} \quad (3)$$

Accuracy kümelenen verilerin gerçekte ne kadarının doğru sınıfa atandığını bulmaya yarayan bir kümeleme başarısı değerlendirme yaklaşımıdır. a_i i kümesine atanmış ve gerçekte de i kümesine ait olan verilerin sayısı ve n de veri setindeki toplam veri sayısı olmak üzere Accuracy Eş. 4 ile hesaplanır.

$$Accuracy = \frac{\sum_{i=1}^{k} a_i}{n} \quad (4)$$

Precision ve Recall en temel kümeleme başarısını değerlendirme yöntemleridir. Bir dokümandaki ilgili olan dokümanların sayısı $|R|$, R olduğu tahmin edilenlerin sayısı $|A|$

ve R'ye ait olduğu tahmin edilenlerden gerçekte R'ye ait olanlar $|R_a|$ olmak üzere Precision ve Recall değerleri Eş. 5 ve Eş. 6 ile hesaplanır.

$$Precision = \frac{|R_a|}{A} \quad (5)$$

$$Recall = \frac{|R_a|}{|R|} \quad (6)$$

Rand Index (RI) [28], bir kümedeki verinin değerini hesaplarken o veriyi aynı kümede olan diğer verilerle ikili olarak karşılaştırır. Bu işlemi aynı şekilde gerçek sınıf etiketleri için de yapar. Sonrasında bu yapılan işlemi ikili olarak karşılaştırır. Aynı mı yoksa farklı mı olduğunu bir tabloda tutar. Bu tablo üzerinden RI değerini hesaplar. $X_{Gerçek}$, X parçasına ait gerçek sınıf etiketleri ve X_{Tahmin} X parçasına ait tahmin, $Uyumlu_{Aynı}$ ikili karşılaştırmanın hem $X_{Gerçek}$'de hem X_{Tahmin}'de aynı olmasını, $Uyumlu_{Farklı}$ ikili karşılaştırmanın birinde farklı olmasını, $Uyumsuz_{Aynı}$ ikili karşılaştırmanın hem $X_{Gerçek}$'de hem X_{Tahmin}'de aynı olmasını, $Uyumsuz_{Farklı}$ ikili karşılaştırmanın birinde farklı olmasını ifade etmek üzere RI Eş. 8 ile hesaplanır.

$$RI = \frac{Uyumlu_{Aynı}+Uyumlu_{Farklı}}{Uyumlu_{Aynı}+Uyumlu_{Farklı}+Uyumsuz_{Aynı}+Uyumsuz_{Farklı}} \quad (8)$$

Adjusted Rand Index (ARI) [29], RI üzerinden hesaplanan bir kümeleme başarısı değerlendirme yöntemidir Eş. 9 ile hesaplanır.

$$ARI = \frac{RI-Beklenen\ RI}{Max\ RI-Beklenen\ RI} \quad (9)$$

Jaccard index [30], modelin etiketlediği verileri gerçek etiketler ile karşılaştırır. Modelin verileri atadığı kümenin etiketi A, söz konusu verilerin gerçek küme etiketi B olmak üzere Jaccard index (*Js*) iki kümenin kesişimlerinin birleşimlerine oranıdır ve Eş. 10 ile hesaplanır.

$$Js = \frac{A \cap B}{A \cup B} \quad (10)$$

Yukarıda saydığımız küme başarısı değerlendirme yöntemlerinin dışında Calinski-Harabasz index [31], I index [32], Dunn index [33], Davies-Bouldin index [34], Fowlkes & Mallows [35], BIC (Bayesian Information Cirtaria) index [36], NMI (Normalized Mutual Infprmation) [37] ve Entropy [38] gibi küme başarısı değerlendirme yöntemleri de vardır.

Küme başarısı değerlendirme yöntemleri olarak dahili yöntemler harici yöntemlere göre daha kullanışlı ve tutarlıdır. Çünkü dahili yöntemlerde bir küme etiketine ihtiyaç olmadığından modelin kümeleme yaparken verdiği etiketin bir önemi yoktur. Bu nedenle Gerçekte 1 numaralı küme olarak etiketlenmiş verilerin model tarafından 2 veya 3 olarak etiketlenmesinin bir farkı yoktur ve kümeleme başarısını etkilemez. Oysa ki harici yöntemlerde verilen etiketin gerçek küme etiketi ile uyuşmaması, sonucu doğrudan etkiler. Zaman karmaşılığı açısından bakıldığı zaman ARI, Jaccard index, Fowlkes & Mallows ve Silhouette index gibi kümeleme başarısı değerlendirme yöntemlerinin zaman karmaşıklığı diğer yöntemlerden daha yüksektir.

Kümeleme başarısını değerlendirme yöntemlerinden bir diğeri de F-Score'dur [27]. F-Score Precision ve Recall değerlerinin harmonik ortalamasıdır ve Eş. 7 ile hesaplanır.

$$F - Score = \frac{2.Precision.Recall}{Precision+Recall} \quad (7)$$

8. Akan veri kümelemede temel yaklaşımlar

Akan veri kümeleme yaklaşımları parçalamalı, hiyerarşik, yoğunluk-tabanlı, grid-tabanlı ve model tabanlı olarak beş ana gruba ayrılmaktadır [39]. Tablo 1'de de görüldüğü gibi her yöntemin hem avantajı hem de dezavantajı bulunmaktadır.

Parçalama temelli akan veri kümeleme yaklaşımında veri k-means gibi uzaklık tabanlı bir algoritma ile çeşitli sayıda kümelere ayrılır. Bu tür yaklaşımlarda küme şekli yuvarlaktır. Oysaki küme şekli verinin özelliğine bağlı olarak değişebilir. Bu yaklaşımlarda genelde sonuç gürültü ve sapan veriden etkilenir. STREAM [40] ve CluStream [24] bu yaklaşımlara örnek verilebilir.

Hiyerarşik kümeleme, veriyi kümelerden oluşan bir ağaç yapısında gruplar. Veri örnekleme ve veri görselleştirme açısından faydalı bir yaklaşımdır. Küme birleştirme (merge) ve küme bölme (split) işlemi sürekli yapılır. BIRCH [41] ve Chameleon [42] bu yaklaşımlara örnek verilebilir. Hiyerarşik kümelemenin başarısını arttırmak için diğer yöntemlerle birleştirilmiş yaklaşımlar da mevcuttur. Bu yaklaşımlara örnek ClusTree [43] verilebilir.

Grid tabanlı kümeleme, verinin dağılımından bağımsızdır. Veriyi grid denilen parçalara ayırır. Zaman açısından hızlı bir kümeleme yaklaşımıdır. STING [44], WaveCluster [45] ve CLIQUE [46] bu yaklaşımlara örnek verilebilir. Grid tabanlı yaklaşımların yoğunluk tabanlı yaklaşımlarla birleştirildiği çok sayıda çalışma da mevcuttur. D-Stream [47] ve MR-Stream [48] bunlara örnek verilebilir.

Model tabanlı kümeleme yaklaşımları, EM (Expectation Maximization) [49] gibi matematiksel modellerle veri arasında bir optimizasyon bulmaya çalışan kümeleme yaklaşımlarıdır. EM algoritması k-means algoritmasının değişik bir versiyonu olarak düşünülebilir. SWEM [50] algoritması bu yaklaşımlara örnektir.

Yoğunluk tabanlı kümeleme, yoğunluğu baz alarak kümeleme yapar. Bu yaklaşımlarda kümeler verinin yoğunlaştığı alanlarda yoğunlaşır. Temel yaklaşımı kümeyi verinin yoğunlaştığı yerlere doğru genişletmeye dayanır. Bu işlemi gerçekleştirirken belirli bir eşik değer kullanır. Böylece sapan verileri bertaraf etmek ve veriye göre şekiller elde etmek mümkün olabilmektedir. DBSCAN [51], OPTICS [52] ve DENCLUE [53] yoğunluk tabanlı kümeleme algoritmalarıdır. DenStream [54] akan veriyi kümelemeye yönelik geliştirilmiş yoğunluk tabanlı kümeleme algoritması olarak verilebilir.

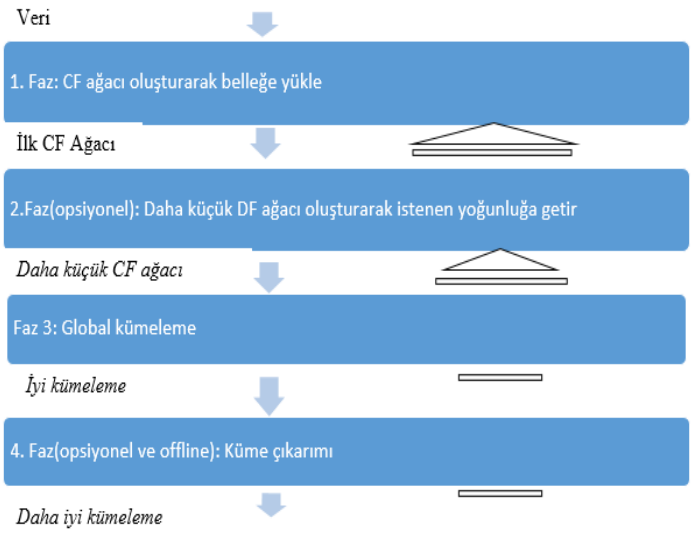
Tablo 1. Akan veri kümeleme yaklaşımlarının karşılaştırılması [55]

Metod	Avantaj	Dezavantaj
Parçalamalı	• Uygulaması kolaydır • Özyinelemeli bir şekilde kümeleri oluşturur	• Küme sayısı kullanıcı tarafından tanımlanmalıdır • Sadece dairesel şekiller elde edilebilir
Hiyerarşik	• Mesafe veya benzerliği ele alış şekli işi kolaylaştırır	• İşlemi tamamlama kriterlerinde problem olabiliyor

		• Zaman karmaşıklığı yüksektir
Grid-tabanlı	• İşlemleri hızlı bir şekilde yapar • Sapan verileri tespit edebilir	• Çok boyutlu veriler kullanılamaz • Grid boyutu tanımlanmak zorundadır
Yoğunluk-tabanlı	• Farklı şekle sahip kümeleri tespit edebilir. • Sapan verileri tespit edebilir	• Çok sayıda parametre tanımlanmalı • Çoklu yoğunluk durumlarında çalışmaz
Model-tabanlı	• Standart istatistiki bilgilere dayanarak küme sayısını otomatik bir şekilde tanımlayabilir • Sapan verileri tespit edebilir	• Model veya tanımlanmış yapıya bağlıdır

9. Başlıca akan veri kümeleme algoritmaları

STREAM [40], k-means tabanlı iki aşamadan oluşan bir yaklaşımdır. Böl ve fethet yaklaşımına dayanır. İlk aşamada data stream kova (bucket) denilen parçalara bölünür ve her kova için k-median uygulanarak k tane küme bulunur. Küme merkezleri kaydedilerek küme merkezleri sahip oldukları veri miktarlarına göre ağırlıklandırılır. Bu noktada asıl veriler göz ardı edilerek ağırlıklandırılmış veri merkezleri veri olarak kabul edilir. İkinci aşamada ağırlıklandırılmış veri merkezleri kümelenerek daha az sayıda küme elde edilir.



Şekil 7. BIRCH genel yapısı [41]

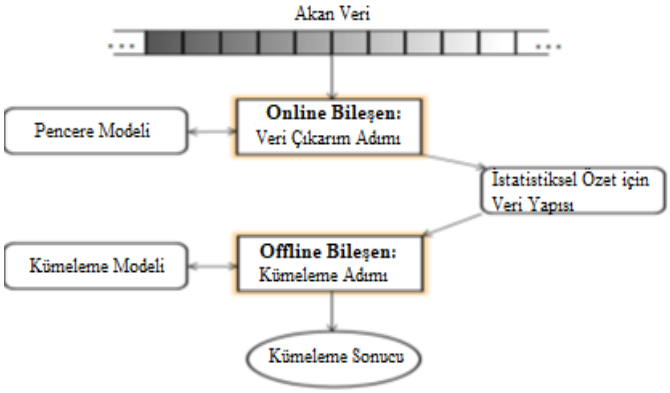
CluStream [24] bir online ve bir offline bileşenden oluşur. Online bileşen veriyi mikro-küme (micro-cluster) yapısında özetler. Burada mikro-kümeler BIRCH [41] (Şekil 7)'in kümeleme özelliğinin geçici genişletilmiş bir versiyonudur. Veriye ait özetleme istatistikleri belleğe kaydedilir. Bu durum, kullanıcıya zaman açısından esneklik sağlar. Offline bileşen ise k-means kümeleme uygulayarak mikro-kümeleri daha büyük kümelere dönüştürür. Bu algoritma dairesel olmayan şekilleri bulmakta çok verimli değildir. Ayrıca sapan veriden çok etkilenir. Bunun yanında yapısı gereği büyük veri setleri için çok başarılı değildir. CluStream algoritmasının gelişmiş bir versiyonu olan StreamSamp [56] algoritması geliştirilmiştir. Bu yaklaşım temel anlamda verimli bellek kullanımı ile ön plana çıkmaktadır. StreamSamp'te sabit boyutlu bellek kullanımı benimsenmiştir. İki çeşit hafızalama yoluna gidilmektedir. Kısa boyutlu ve daha uzun boyutlu hafızalama. Kısa boyutlu hafızalama küçük boyutlu periyodik hafızalama yaparken, uzun boyutlu hafızalama büyük boyutlu periyodik hafızalama yapmaktadır. StreamSamp akan veriyi hızlı bir şekilde özetleme yeteneğine sahip olmasına rağmen kümeleme başarısı zamana bağlı olarak azalmaktadır.

ClusTree [43] kümeleme işlemini yine online ve offline olmak üzere iki aşamada gerçekleştirir. Online kısım mikro-kümeleri bulmak için kullanılır. mikro-kümeler hiyerarşik yapıda birleştirilir. Böylece herhangi bir adımda offline bileşen uygulanabilir. Sistem kendinden uyarlamalıdır. Bu nedenle herhangi bir anda sistem sonuç üretebilmektedir.

HPStream [57], Fading Cluster Structure (FCS – Veri Ağırlığını Azaltma) kullanarak zamana bağlı olarak eski verilerin ağırlığını azaltır. CluStream algoritmasının gelişmiş bir versiyonudur. ClueStream algoritmasına göre avantajı çok boyutluluğu desteklemesidir. Çok boyutlu verilerde boyutların bir alt kümesini alır. Nitelik sayısı her küme için farklı olabilir. Nitelik sayısı güncellenebilmektedir. Yeni gelen verilere daha fazla önem verilmesi gereken uygulamalar için uygundur. Bu yaklaşım sensör network uygulamaları için oldukça faydalı bir yaklaşımdır. Çok boyutluluğu desteklemesine rağmen çok boyutluluk konusunda optimum bir sonuç verememektedir. Küme şekillerini belirlemekte problem çıkmaktadır. İyi performans için verinin tamamı hakkında yeterli bilgiye sahip olmak gerekir. Bu da akan veri için her zaman mümkün değildir.

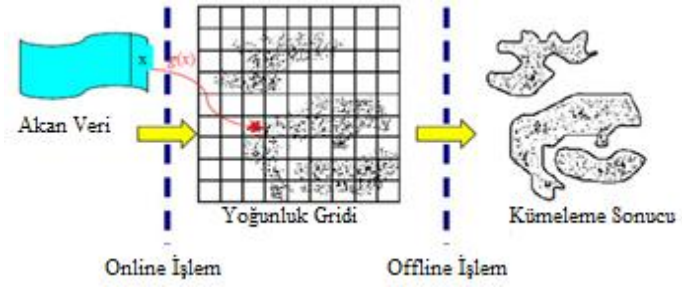
DUCstream [58] veriyi birbiri ile örtüşmeyen gridlere ayırır ve veriyi yığın (chunk) denilen parçalara ayırır. Her bir yığın M tane veriden oluşur. Gridler noktalar halinde haritalanır ve depth first search algoritması uygulanarak kümeler oluşturulur. Kümeleri bir grafın parçaları gibi birbirine bağlar. Kümeleme açısından oldukça elverişli bir algoritmadır. Ancak veriyi gridlere ayırmak için veri hakkında yeterli bilgiye sahip olmak gerekir.

DenStream [54] yoğunluk tabanlı bir kümeleme yaklaşımıdır. Bu yaklaşım da DBSCAN algoritmasının geliştirilmiş bir versiyonudur ve mikro-kümeler kullanır. Şekil 8'de de görüldüğü gibi online ve offline olmak üzere iki bölümden oluşur. Online bölümde veriler mikro-küme şeklinde temsil edilir. Offline bölümde mikro-küme yapısındaki veri DBSCAN [51] algoritması ile kümelenir. Bu algoritmanın en büyük avantajı sapan verileri tespit edebilmesidir. rDenStream [59] algoritması da DenStream algoritması gibidir. rDenStream algoritmasının en önemli farkı atılan verilere ikinci bir şans vermesidir.



Şekil 8. DenStream algoritması [54]

E-Stream [60], beş tip işlem gerçekleştirmektedir: Yeni bir kümenin ortaya çıkması, var olan bir kümenin yok edilmesi, büyük bir kümenin bölünmesi, iki kümenin birleştirilmesi, kümenin karakteristiğinin değiştirilmesi. Fading cluster yapısını ve histogram kullanır. Bu nedenle performansı HPStream [57] algoritmasından daha iyidir. Ancak kullanıcının pek çok parametreyi tanımlamasına ihtiyaç duyar [60].



Şekil 9. D-Stream kümeleme [47]

D-stream [47], yoğunluk tabanlı grid yapısı kullanan bir kümeleme algoritmasıdır. Veriyi gridlere böler. Şekil 9'da da görüldüğü gibi iki aşamadan oluşur. Online bölümde gelen veri gridlere göre haritalanır. Offline aşamada her grid için yoğunluk hesaplayarak veriyi atar. Son olarak grid yoğunluğuna göre bir kümeleme yapar. Eski gridlerin ağırlığını azaltmak için zamana bağlı olarak ağırlık azaltma (fading) fonksiyonu kullanır. Eğer belirlenen bir gridin yoğunluk değeri eşik değerinin altına düşerse ve yeni veriler eklenmez ise silinir. Nitelik sayısı (dimension) arttıkça grid sayısı üssel artar, bu nedenle çok boyutlu problemler için uygun değildir.

SE-Stream [61], E-Stream algoritmasının gelişmiş bir versiyonudur. Boyut problemini de ele alan bir algoritmadır. Amaç çok boyutlu veriler için performansı arttırmak. Boyut indirgeme işlemi yapılır. Bunu yaparken de sonuçla en fazla alakası olan nitelikler seçilir. E-Stream [60] algoritmasında MergeOvelapClustering ve LimitMaximumCluster işlemleri çok zaman almaktadır. SE-Stream algoritmasında bunlar optimize edilir.

DD-Stream [62], yoğunluk ve grid tabanlı yaklaşımları birleştiren bir yaklaşımdır. n-boyutlu veriyi gridlere ayırır ve özvektör (eigenvector) ile update eder. Özvektör grid merkezinin koordinatlarını, gridin en son güncellendiği zamanı, gridin grid merkezinden silindiği zamanı ve en son grid yoğunluğu gibi veri gruplarını içerir. Özvektör ile gridin yoğun mu yoksa seyrek bir grid mi olduğuna karar verir. Şekil 10'da da görüldüğü gibi yoğun gridlere ait kümeleri bulmak için yoğunluk tabanlı kümeleme yaklaşımı kullanılır.

HUE-Stream [63], E-Stream [60] algoritmasının geliştirilmiş bir versiyonudur. Kategorik niteliklerde belirsizliği ortadan kaldırmak için iki objenin uzaklık fonksiyonunu olasılık dağılımı ile kullanır. K üme yapısında değişiklik yapılıp yapılmayacağına adı geçen uzaklık fonksiyonu ile karar verir. Bu fonksiyon ile bir kümenin ikiye ayrılmasına veya iki kümenin birleştirilmesine karar verir.

STREAMKM++ [64], k-means ve Öklid uzaklığı tabanlı bir akan veri kümeleme algoritmasıdır. Eğer küme merkezlerinin sayısı fazla ise BIRCH [41] algoritmasından daha iyi sonuç vermektedir. Ancak performans açısından bakıldığı zaman BIRCH algoritmasına göre daha yavaş bir algoritmadır.

HDDStream [65], HPStream [57] benzeri bir algoritmadır. HDDStream üç aşamadan oluşur. İlk aşamada mikro-kümeler oluşturulur. Sonra online bölümde sapan veriler tespit edilir ve geri kalan veri kümelenir. Offline bölümde ise elde edilmiş kümeler daha büyük kümelere dönüştürülür. HDDStream algoritmasının HPStream algoritmasından farkı küme sayısının zamana bağlı olarak değişmesidir.

LeaDen-Stream [66], yoğunluk tabanlı bir akan veri kümeleme yaklaşımıdır. Şekil 11'de de görüldüğü gibi bu yaklaşımın yoğunluk tabanlı diğer çalışmalardan farkı, oluşturulmuş olan mikro-kümelerin içindeki verilerin dağılımını da göz önünde bulundurmasıdır. Burada mikro-kümelerin içinde verinin yoğunlaştığı yerde bulunan veri lider olarak seçilir. Offline bölümde lider noktalar kullanılarak sonuç kümeleri oluşturulur.

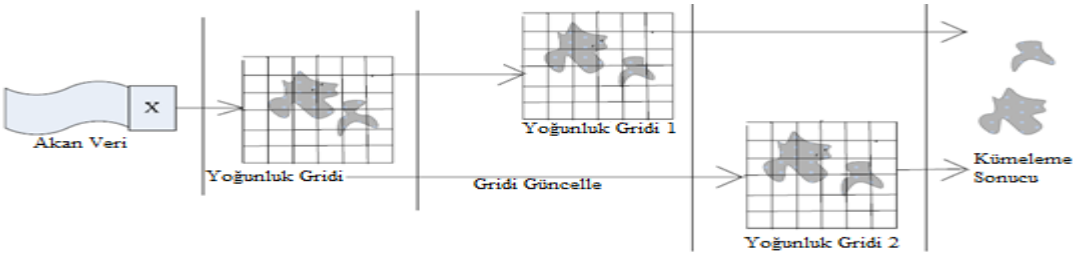
CL-Ant [68] ve CL-AntInc [69] algoritmaları karınca kolonisini örnekleyen akan veri kümeleme yaklaşımlarıdır. CL-Ant ve CL-AntInc algoritmalarında veriler karınca olarak temsil edilmektedir. Karınca kolonilerinin her biri de küme olarak kabul edilmektedir Algoritma başlangıçta k-means algoritması uygulayarak kümeleme yapmaktadır. Daha sonra dinamik bir graf yapısı oluşturularak dinamik bir kümeleme yapılmaktadır.

HSDStream [70], çok boyutlu verileri kümelemek amacıyla gerçekleştirilmiş bir algoritmadır. HSDStream üç aşamadan oluşmaktadır. Birinci aşamada veri sabit boyutlu mikro-kümelere(core-mc) dönüştürülür. İkinci aşama olan online aşamada sapan veriler tespit edilir. Offline bölümde ise sonuç kümeleri oluşturulur. Bu algoritmanın HDDStream [65] algoritmasından farkı çok boyutlu veriyi işleyebilmesidir. Yapılan deneysel çalışmalar HSDStream algoritmasının HDDStream algoritmasından daha iyi sonuçlar verdiğini ortaya koymuştur.

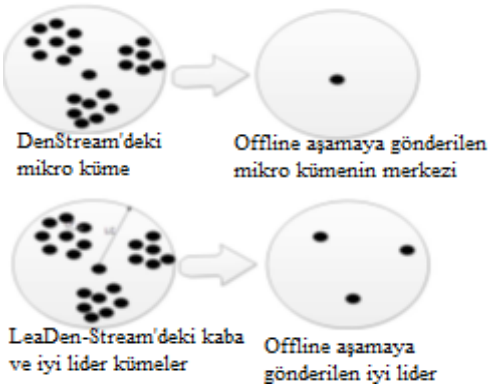
Şekil 12'de de görüldüğü gibi CODAS [67], yine yoğunluk tabanlı bir akan veri kümeleme yaklaşımıdır. CODAS'ın getirdiği en önemli yenilik tamamen online çalışmasıdır. Online olarak veriyi mikro-kümelere böler, sonra bu mikro-kümeler birleştirilerek asıl kümeler elde edilir. Bunu gerçekleştirmek için kümelemede hyper-spherical mikro-küme denilen yapıyı kullanır. Sapan verileri tespit etmek için bir eşik değer kullanılır. Bu kümeleme yaklaşımının en büyük eksiği online çalışmasından dolayı çok hızlı akan veriyi işlemede oluşabilecek problemler ve son gelen verileri daha önemli kılacak bir yapıya sahip olmamasıdır.

SOC [71], temelde hızlı bir şekilde ve online kümeleme yapmayı amaçlayan bir yaklaşımdır. k-means ve k-medoid gibi yaklaşımların aksine herhangi bir varsayımda bulunmaz. Kümeler

skeleton denilen parçalar ile temsil edilir. Skeleton denilen parçalar verinin yoğunluğa göre ağırlıklandırılmış şeklidir. Hızlı bir şekilde kümeleme yapmak için her yeni veri gelişinde kümeler güncellenir. Önerilen yaklaşım kümeleri otomatik bir şekilde tespit ederken sapan verileri de tespit etme yeteneğine sahiptir. Mevcut yaklaşımların çoğundan daha iyi sonuç verdiği tespit edilmiştir.



Şekil 10. DD-Stream algoritması [62]



Şekil 11. LeaDen-Stream algoritmasında mikro-makro kümeler[66]

STREAMLEADER [72], CODAS algoritması gibi offline bölüme gerek duymadan veriyi online olarak kümeleyen bir kümeleme yaklaşımıdır. Bu yaklaşımda kullanıcıdan sadece bir parametreyi belirlemesi beklenmektedir. Sistem MOA[1] platformuna adapte edilmiştir. Geliştirilen yöntem CluStream, DenStream ve ClusTree algoritmaları ile karşılaştırıldığında daha iyi sonuç verdiği tespit edilmiştir.

DBSTREAM [73] mikro-küme yapısını kullanırken mikro-kümeler arsındaki verileri de işleme alır. Online kısımda veri özetlenirken özeti alınmış veriler yoğunluğa göre ağırlıklandırılmaktadır. DBSTREAM veriyi mikro-kümelere bölerken sadece mikro-kümeler arası mesafeyi almaz aynı zamanda orijinal veriler arasındaki yoğunluk bağlantılarını da alır.

DCSTREAM [74], STREAM gibi böl ve fethet ve k-means tabanlı bir kümeleme algoritmasıdır. Büyük verileri kümelemek için geliştirilmiştir. Veriler *w* boyutunda parçalara ayrılır. w_i penceresine alınmış veri w_i+1 süresi içerisinde yok edilir. Yani geriye dönüş w süresince olabilir. Biri online ve biri offline olmak üzere iki bileşenden oluşur. Veri online bölümden önce veri temizleme, nitelik seçme, nitelik azaltma ve veri dönüşümü gibi işlemler gerçekleştirilir. Online bölümde veriler mikro-kümelere bölünür. Burada küme birleştirme ve ayırma (merge ve split) k-means ile yapılır. Offline bölümde w_i+1 süresi sonunda w_i yok olacağından son gelen verilere önem verilerek kümeleme yapılmış olur. Şekil 14 DCSTREAM algoritmasının framework'unu göstermektedir.

FEAC-Stream [75] küme sayısında sınırlama getirmeyen ve veriyi özetlerken verinin değişim noktalarını tespit etmeye çalışan böylece performansı arttıran bir akan veri kümeleme yaklaşımıdır. Yani oluşturulmuş kümeleri değiştirmeyi belirli şartlara bağlar aksi durumda kurulmuş olan yapı devam ettirilmektedir

DPStream [76] algoritması DPClust yapısını kullanan bir hiyerarşik veri kümeleme yaklaşımıdır. Verileri hiyararşik bir yapıda birleştiren DPStream bunu yaparken verileri yoğunluğa bağlı olarak en yakın yüksek yoğunluklu olarak birbirine bağlar. Mevcut yöntemler ile karşılaştırıldığında performans açısından çok hızlı çalışan bir algoritma olduğu söylenebilir.

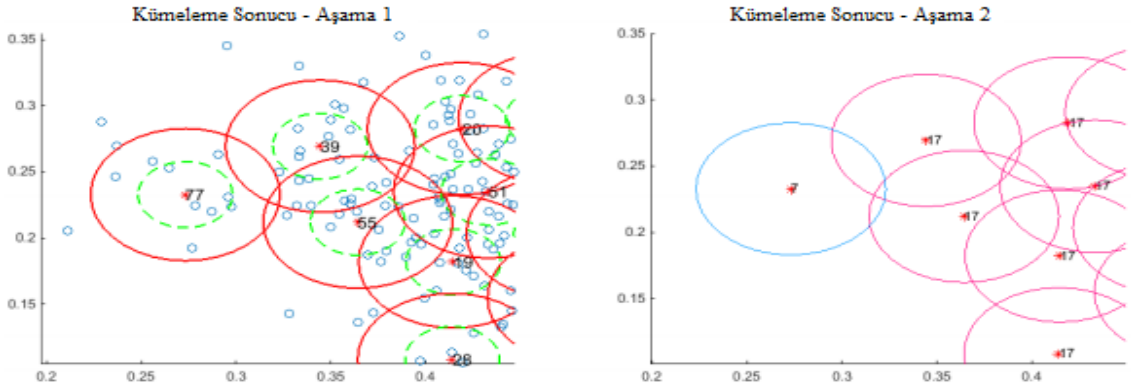
CEDAS [77] graf yapısını kullanan bir akan veri kümeleme yaklaşımıdır. Verileri ilk olarak mikro-küme yapısında birleştiren daha sonra bu mikro-kümelerin yakınlık derecesine bakarak birleştiren ve makro kümeleri oluşturan online çalışan ve farklı şekildeki kümeleri tespit etme yeteneğine sahip bir yaklaşımdır. Her veri gelişinde graf yapısı kontrol edilmekte ve eğer gerek varsa bu yapı güncellenmektedir.

LLDStream [78], DenStream ve DBSCAN [51] algoritmalarını temel alan yoğunluk tabanlı bir algoritmadır. LLDStream online ve offline olarak iki aşamadan oluşur. Online aşamada gelen verinin atanacağı mikro-kümeyi belirler. Offline aşamada ise kullanıcıya mikro-kümelerden oluşan makro kümeler sunulur. LLDStream boyut indirgemeyi LDA (Linear Discriminant Analysis)'dan faydalanarak yapar. Çok boyutluluğu ve farklı şekildeki kümeleri tespit edebilmesi avantajı olarak öne çıkmaktadır.

Bir diğer önerilen akan veri kümeleme yaklaşımı Xinxin ve ark. [79] sapan verileri tespit edebilen, en yakın komşu yaklaşımından güç alan yoğunluk ve grid tabanlı algoritmadır. Bu yaklaşım kayan pencere veri özetleme yaklaşımını kullanır. Özeti alınmış veri bir graf yapısında yapılandırılır. En yakın komşu yaklaşımına göre bir birine yakın olan veriler aynı kümenin elemanları olarak tanımlanır. Farklı şekildeki kümeleri ve sapan verileri tespit edebilmesi en güçlü yönleri olarak öne çıkarken,

[1] Massive Online Analysis: Yeni Zelanda'nın Waikato Üniversitesi'nin akan veri kümeleme için geliştirdiği açık kaynak kodlu bir framework

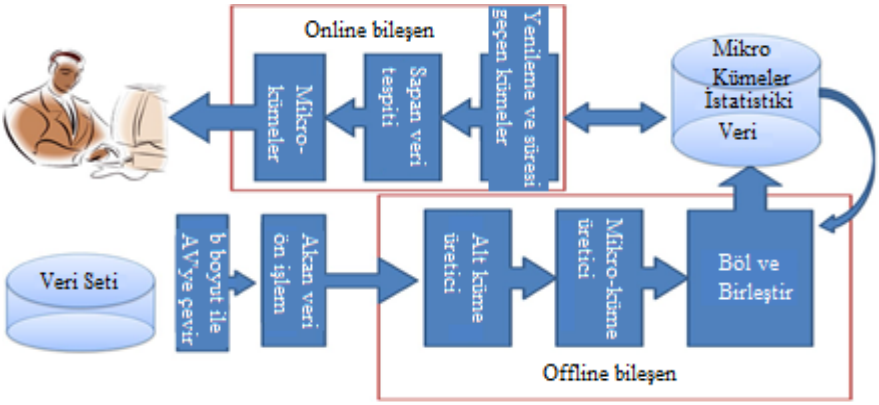
parametrelerin optimum belirlenmesi ihtiyacı zayıf tarafı olarak öne çıkmaktadır.



a. Çekirdek mikro-kümeler *b. Birleştirilmiş mikro-kümeler*

Şekil 12. CODAS algoritmasına ait mikro ve birleştirilmiş kümeler [67]

.



Şekil 13. DCSTREAM algoritmasına ait framework [74]

SPE-Cluster [80], oto-regresyon modelleme tekniğini kullanarak akan veriler arasındaki korelasyonu hesaplar. Bunun için akan veriden birbiri ile alakalı nitelikleri bulmak için frekans spektrumunu bulur. Online ve offline iki bölümden oluşur. Online bölüm kayan pencere ile akan verilerin spektral bileşenlerini hesaplar. Offline bölüm ise Dinamik k-means kullanarak akan veriyi kümeler. SPE-Cluster dinamik yapısı ile kümeleme adaptasyonu (concept evolution) ve küme sayısı problemlerine çözüm üretmektedir.

Jurgen ve ark. [81] paralel akan veriyi kümelemek için değişken tabanlı bir yaklaşım önermişlerdir. Veriyi özetlemek için veriler arasındaki uzaklığı ve Discrete Fourier Transform (DFT) katsayısını kaydeder. Akan veriyi işlemek için kayan pencere yapısını kullanır. Kümeleme işlemini gerçekleştirmek için k-means kümelemeyi kullanır. Kümeleme adaptasyonu gerçekleştirmek için küme sayısını birer birer artırır ve birer birer azaltır. Küme sayısını artırırken merkeze uzak veriyi yeni küme olarak tanımlar. Küme sayısını azaltırken var olan bir kümenin elemanlarını diğer kümelere atar.

ODAC [82], kümeleme adaptasyonunu desteklemek için ayırıcı ve kümeleyici bir yapıdadır. Üstten aşağıya (top-down) bir yapıyı destekler, hiyerarşik bir ağaç yapısındadır. Veriler arasındaki uzaklığı bulmak için korelasyona benzeyen bir hesaplama yaklaşımı vardır. Hiyerarşik ağaçta her bir kök (node - burada küme) uzaklığa bakarak ayırır. Aynı şekilde uzaklık korelasyonuna göre iki kökü birleştirir. Var olan kümelerin çapına bakarak kümelerin birleştirilmesine veya ikiye ayrılmasına karar verir. Bu işlem zamana bağlı olarak periyodik bir şekilde yapar.

POD-Clus [83], model tabanlı bir akan veri kümeleme yaklaşımıdır. Örnekleme için küme özetini kullanır. Bunun için ortalama, standart sapma ve her küme için nokta sayısına bakar. Kümelerin ayrıştırılması, iki kümenin birleştirilmesi, yeni bir küme oluşturulması veya var olan bir kümenin yok edilmesini destekler. Kümeleme başarısı tatmin edici olmasına rağmen işlem performansı kötüdür.

COMET-CORE [84], çoklu akan veriyi online manada kümelemek için geliştirilmiş bir yaklaşımdır. Benzerlik ölçütü olarak ağırlıklandırılmış korelasyon kullanır. ODAC [82]'ın aksine COMET-CORE veriyi periyodik olarak kümelemez. Belirli bir durum oluştuğunda kümeleri böler veya birleştirir. Akan veriler arasındaki ağırlıklandırılmış korelasyonu sürekli günceller.

Tablo 2, akan veri kümeleme alanında geliştirilmiş olan belli başlı yaklaşımları kullanılan parametreler, kümeleme adaptasyonunu destekleme, veri ağırlığını zamana göre azaltma (fading), kümeleme türü, avantaj ve dezavantajları açısından karşılaştırmaktadır.

Tablo 2. Akan veri kümeleme alanında geliştirilmiş algoritmalardan bazılarının karşılaştırılması

Algoritma	Yıl	Parametreler	Avantajlar	Dezavantajlar	Kümeleme Adaptasyonu Desteği Var mı?	Veri Ağırlığını Azaltma Var mı?	Kümeleme türü
STREAM	2001	Küme sayısı	Performansı	Küme sayısını sınırlama	Hayır	Hayır	Parçalamalı
SPE-Cluster	2001	Kayan ve temel pencerelerin tanımlanması gerekir	Küme sayısı belirlemede ve zamana bağlı küme adaptasyonunda avantajlıdır	Dairesel olmayan kümeleri bulmakta problemli	Evet	Hayır	Parçalamalı
CluStream	2003	Küme sayısı ve zaman penceresi	Özetleme sayesinde kullanıcıya esneklik sağlar	Dairesel olmayan kümeleri tespit etmekte problemli Sapan veriden etkilenir Büyük verilerde problemli	Evet	Hayır	Parçalamalı & Hiyerarşik
HPStream	2004	Max küme sayısı, ortalama boyut değeri	Çok boyutlu verileri destekler Yeni gelen verilerin ağırlığı daha fazla	Dairesel olmayan kümeleri bulmakta problemli	Evet	Evet	Parçalamalı & Hiyerarşik
DUCstream	2005	Grid hücrelerinin yoğunluk eşik değeri	Dairesel olmayan şekilleri bulabiliyor	Tüm verilerin ağırlığı aynı	Evet	Hayır	Yoğunluk & Grid tabanlı
DenStream	2006	Küme yarıçapı eşik değeri Veri ağırlık eşik değeri Sapan veri eşik değeri Küme ağırlığı Bozunum katsayısı (decay factor)	Farklı şekillerdeki kümeleri bulabilir Sapan verileri tespit edebilir	Sapan verileri bulma işlemi performansı düşürüyor Küme bölme ve birleştirme işlemlerini yapmıyor	Evet	Evet	Yoğunluk tabanlı
D-Stream	2007	Bozunum katsayısı Grid yoğunluk eşik değeri Veri ağırlığını azaltma oranı	Yoğunluğa bakarak kümeleri gerçek zamanlı olarak tespit edebiliyor Zamana bağlı olarak kümelerin dönüşümünü destekliyor	Algoritmanın zaman karmaşıklığı pek çok değişkenden etkilenir	Evet	Evet	Yoğunluk & Grid tabanlı
E-Stream	2007	Maksimum küme sayısı, yarıçap değeri, stream hızı, silme eşik değeri, aktif olma eşik değeri, birleştirme eşik değeri	Kümelerin zamana bağlı dönüşümünü destekler Zamanla küme bölme ve birleştirme işlemleri gerçekleşir	Dairesel olmayan kümeleri bulmakta problemli, çok sayıda parametrenin tanımlanması gerekir	Evet	Evet	Hiyerarşik
DD-Stream	2008	Grid hücrelerinin yoğunluk eşik değeri, yoğunluk fading oranı gibi parametreler	Farklı şekillerdeki kümeleri bulabilir Kümeleme başarısı tatmin edici	Çok boyutluluğu destekleme	Evet	Evet	Yoğunluk & Grid tabanlı
ODAC	2008	Kümeleri bölme eşik değeri	Kümeleri bölme ve birleştirme işleminde başarılı	Dairesel olmayan kümeleri bulmakta problemli	Evet	Hayır	Hiyerarşik
POD-Clus	2008	Küme bölme ve birleştirme eşik değerleri, veri ağırlığı azaltma oranı	Dairesel olmayan kümeleri tespit edebilir	İşlem performansı kötü	Evet	Evet	Model tabanlı
COMET-CORE	2009	Korelasyon eşik değeri, küme bölme ve birleştirme eşik değerleri	Çok boyutluluğu destekler	Dairesel olmayan kümeleri bulmakta problemli	Evet	Evet	Hiyerarşik
HUE-Stream	2011	Veri ağırlığı azaltma oranı ve E-Stream'in diğer parametreleri	Kümelerin zamana bağlı dönüşümlerini destekler, performansı E-Stream'e göre daya iyi	Dairesel olmayan kümeleri bulmakta problemli, çok sayıda parametrenin tanımlanması gerekir	Evet	Evet	Hiyerarşik
HDDStream	2012	Küme yarıçapı, küme ağırlığı, sapan veri eşik değeri, bozunum katsayısı	Farklı şekildeki kümeleri tespit edebilir, çok boyutlu verileri destekler, sapan verileri tespit edebilir,	Algoritmanın performansı düşük	Evet	Evet	Yoğunluk
LeaDen-Stream	2013	MMLC ve MMLCweight, MMLC ve MMLCCenter, MMLC ve MMLCradius	Kümeleme başarısı makul	Çok boyutlu veriyi desteklemiyor	Evet	Evet	Yoğunluk

			Sapan verilere karşı dirençli, performansı tatmin edici				
SE-Stream	2013	Silme eşik değeri, yarıçap değeri, veri akış hızı gibi çok sayıda parametre tanımlanmalıdır	E-Stream algoritmasına göre performans daha iyi, algoritma optimize edilmiştir.	Çok sayıda parametrenin tanımlanması gerekir	Evet	Evet	Hiyerarşik
HSDStream	2015	Pencere genişliği, yarıçap eşik değeri, sapan veri eşik değeri, veri sayısı eşik değeri, varyans eşik değeri, boyut indirgeme eşik değeri	Sapan verilere karşı dirençli, çok boyutlu verileri destekler, performans tatmin edicidir	Çok sayıda verinin tanımlanması gerekir	Evet	Evet	Yoğunluk
SOC	2015	Küme bölme ve birleştirme eşik değerleri	Sapan verileri tespit edebilir, kümeleme başarısı tatmin edici	Küme başarısı için belli sayıda verinin ulaşmış olması gerekir	Evet	Evet	Model tabanlı
DCSTREAM	2016	Küme bölme ve birleştirme eşik değerleri, sapan veri eşik değeri	STREAM algoritmasına göre çok daha başarılı, Çok boyutlu veriyi destekler	Dairesel olmayan kümeleri tespit etmek problemli	Evet	Evet	Parçalamalı
DBSTREAM	2016	Ağırlık azaltma oranı, yarıçap, α vs.	Mikro-kümeler arasında kalan verileri de kullanır buna göre gerekirse kümeler güncellenmektedir.	Mikro-kümeler arasındaki verileri de işlediği için zaman karmaşıklığı bir miktar artmaktadır.	Evet	Evet	Yoğunluk
FEAC-Stream	2017	InitSize, 10 k-means iterasyonu vs.	Küme sayısının ön tanımlı olmasına gerek yok	Eski oluşturulmuş kümeler silinir	Evet	Evet	Yoğunluk
DPStream	2017	Lokal yarıçap, global yarıçap, işleme alınan veri boyutu vs.	Tamamıyla online ve farklı şekilleri destekler. Ayrıca çok hızlı çalışan bir algoritmadır.	Çok sayıda parametre var ve parametrelerin çok doğru seçilmesi gerekir.	Evet	Hayır	Hiyerarşik ve Yoğunluk
CEDAS	2017	Yarıçap, küme yaşlanma oranı ve minimum eşik değeri	Tamamıyla online çalışan ve farklı şekildeki kümeleri tespit edebilir. Az sayıda parametre tanımlanmakta	Mikro kümelerin birleşimi ile oluşan makro kümelerin doğru tanımlanabilmesi optimum yarıçapın tanımlanması gerekir	Evet	Evet	Yoğunluk
LLDStream	2017	Yarıçap ve yoğunluk parametresi	Boyut indirgemeyi destekler, farklı şekilli kümeleri tespit edebilir, performansı iyi	Tamamen online değil ve seçilen parametreler sonucu etkiler	Evet	Evet	Yoğunluk
Xinxin ve ark. önerdiği algoritma	2018	Yoğunluk eşik değeri, yoğunluk katsayısı, en yakın komşu değeri paylaşım değeri	Farklı şekildeki kümeleri bulabilir, sapan verileri tespit edebilir, küme sayısında sınırlama yok	Optimum parametreleri belirlemek zor	Hayır	Hayır	Yoğunluk & Grid tabanlı

10. Sonuç ve öneriler

Akan veri kümeleme konusu son dönemin popüler konularından biridir. Veri akarken gerçek zamanlı kümeleme yapmak ve kullanıcıya anında bilgi sunmak oldukça önemlidir. Özellikle banka, sağlık kuruluşları, güvenlik ve sağlık kurumları gibi kuruluşlar için veriyi akarken anlamlandırmak gerekir. Çünkü bu tür kuruluşlarda zamana karşı bir yarış söz konudur. Bunun yanında bilim, ticaret, meteoroloji, teknoloji ve kamu yönetimi gibi pek çok alanda yaygın olarak kullanılmakta ve kullanım alanı gün geçtikçe artmaktadır.

Akan veriyi kümeleme statik veriyi kümelemekten çok daha zordur. Çünkü veriyi bir yere kaydetmek ve bu veriyi işlemek çok zordur. Zira büyük miktarda veri akmaktadır. Bu nedenle bu veriyi kaydetmek ve işlemek çok büyük kaynak ihtiyacını doğurmaktadır. Dolayısıyla akan veriyi kümeleyecek yaklaşımlara olan talep sürekli artmaktadır. Veri akarken bu veriyi hızlı bir şekilde işleyecek yaklaşımlara olan ihtiyaç güncel problemlerden birisidir. Literatürdeki çalışmaların çoğu online-offline olarak iki bileşenden oluşmaktadır. Tamamen online çalışan yöntemler de önerilmektedir. Ancak bu tür yöntemler ya istenen başarıyı vermemektedir ya da performansı arttırmak için özetleme yöntemleri kullanıldığından verinin önemli bir kısmı yok sayılmaktadır. Kısaca verinin nasıl özetleneceği de yine çalışmaya açık alanlardan birisidir. Bu nedenle bu alanda yeni yaklaşımların geliştirilmesine ihtiyaç vardır.

Akan veri kümeleme alanında yeni yöntemler geliştirilmesine açık alanlardan bir diğeri evrimsel kümeleme alanıdır. Veri dinamik bir şekilde akmaktadır. Dolayısıyla verinin özelliği de zamana bağlı olarak değişmektedir. Bu nedenle geliştirilen yöntemin de verinin özelliğine göre zamana bağlı olarak evirilmesi gerekir. Yani zamana bağlı olarak bir kümenin oluşturulması, silinmesi, iki kümenin birleştirilmesi veya bir kümenin ikiye bölünmesi ve kümenin yapısının değişmesi gibi

işlemlerin verinin özelliğine bağlı olarak gerektiğinde gerçekleştirilmesi gerekir.

Geliştirilen yöntemlerde kullanılan parametrelerin seçimi ve atanması da yine güncel problemlerden birisidir. Çoğu çalışmada çok sayıda parametrenin belirlenmesi gerekmektedir. Bu parametrelerin doğru belirlenmemesi başarıyı doğrudan etkilemektedir. Parametreleri azaltmak ve bu parametrelerin mümkün olduğunca model tarafından belirlenmesi de önemli problemlerden birisidir. Ayrıca sapan verileri tespit etmek ve performansı düşürmeden çok boyutluluğu desteklemek de güncel problemler arasındadır.

Kaynaklar

1. Ankleshwaria, T.B. and J.S. Dhobi, *Mining Data Streams: A Survey.* International Journal of Advance Research in Computer Science and Management Studies, 2014. **2**(2): p. 379-386.
2. Ikonomovska, E., S. Loskovska, and D. Gjorgjevik, *A survey of stream data mining*, in *Eighth International Conference with International Participation – ETAI 2007*. 2007: Ohrid, Republic ofMacedonia.
3. Aggarwal, C.C., *Data Streams: Models and Algorithms*. 1 ed. Advances in Database Systems. 2007: Springer US.
4. Bifet, A. and R. Kirkby, *Data stream mining a practical approach.* 2009.
5. Yogita and D. Toshniwal. *Clustering techniques for streaming data-a survey*. in *2013 3rd IEEE International Advance Computing Conference (IACC)*. 2013.
6. Antonellis, P., C. Makris, and N. Tsirakis, *Algorithms for clustering clickstream data.* Information Processing Letters, 2009. **109**(8): p. 381-385.
7. Yin, C., L. Xia, and J. Wang. *Application of an Improved Data Stream Clustering Algorithm in Intrusion Detection System*. in *Advanced Multimedia and Ubiquitous Engineering*. 2017. Singapore: Springer Singapore.
8. Yin, C., L. Xia, and J. Wang. *Data Stream Clustering Algorithm Based on Bucket Density for Intrusion Detection*. in *Advances in Computer Science and Ubiquitous Computing*. 2018. Singapore: Springer Singapore.
9. Li, Z.Q., *A New Data Stream Clustering Approach about Intrusion Detection.* Advanced Materials Research, 2014. **926-930**: p. 2898-2901.
10. Weiler, A., M. Grossniklaus, and M.H. Scholl, *Situation monitoring of urban areas using social media data streams.* Information Systems, 2016. **57**: p. 129-141.
11. Hawwash, B., *Stream-dashboard : a big data stream clustering framework with applications to social mediastreams*, in *Department of Computer Engineering and Computer Science*. 2013, University of Louisville.
12. Barddal, J.P., et al., *SNCStream: a social network-based data stream clustering algorithm*, in *Proceedings of the 30th Annual ACM Symposium on Applied Computing*. 2015, ACM: Salamanca, Spain. p. 935-940.
13. Hendricks, D., *Using real-time cluster configurations of streaming asynchronous features as online state descriptors in financial markets.* Pattern Recognition Letters, 2017. **97**: p. 21-28.
14. Aggarwal, C.C., *Data Streams: An Overview and Scientific Applications*, in *Scientific Data Mining and Knowledge Discovery: Principles and Foundations*, M.M. Gaber, Editor. 2010, Springer Berlin Heidelberg: Berlin, Heidelberg. p. 377-397.
15. King, R.C., et al., *Application of data fusion techniques and technologies for wearable health monitoring.* Medical Engineering & Physics, 2017. **42**: p. 1-12.
16. Gravina, R., et al., *Multi-sensor fusion in body sensor networks: State-of-the-art and research challenges.* Information Fusion, 2017. **35**: p. 68-80.
17. Manzi, A., P. Dario, and F. Cavallo, *A Human Activity Recognition System Based on Dynamic Clustering of Skeleton Data.* Sensors (Basel, Switzerland), 2017. **17**(5): p. 1100.
18. Tasnim, S., et al. *Semantic-Aware Clustering-based Approach of Trajectory Data Stream Mining*. in *2018 International Conference on Computing, Networking and Communications (ICNC)*. 2018.
19. Diaz-Rozo, J., C. Bielza, and P. Larrañaga, *Clustering of Data Streams with Dynamic Gaussian Mixture Models. An IoT Application in Industrial Processes.* IEEE Internet of Things Journal, 2018: p. 1-1.
20. Sabit, H., A. Al-Anbuky, and H. Gholam-Hosseini. *Distributed WSN Data Stream Mining Based on Fuzzy Clustering*. in *2009 Symposia and Workshops on Ubiquitous, Autonomic and Trusted Computing*. 2009.
21. Silva, A.d., et al., *A clustering approach for sampling data streams in sensor networks.* Knowl. Inf. Syst., 2012. **32**(1): p. 1-23.
22. Silva, J.A., et al., *Data stream clustering: A survey.* ACM Comput. Surv., 2013. **46**(1): p. 1-31.
23. Datar, M., et al., *Maintaining stream statistics over sliding windows: (extended abstract)*, in *Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms*. 2002, Society for Industrial and Applied Mathematics: San Francisco, California. p. 635-644.
24. Aggarwal, C.C., et al., *A framework for clustering evolving data streams*, in *Proceedings of the 29th international conference on Very large data bases - Volume 29*. 2003, VLDB Endowment: Berlin, Germany. p. 81-92.
25. Keim, D.A. and M. Heczko. *Wavelets and their Applications in Databases*. in *17th International Conference on Data Engineering (ICDE'01), Heidelberg, Germany, 2001*. 2001.
26. Rousseeuw, P.J., *Silhouettes: A graphical aid to the interpretation and validation of cluster analysis.* Journal of Computational and Applied Mathematics, 1987. **20**: p. 53-65.
27. Brun, M., et al., *Model-based evaluation of clustering validation measures.* Pattern Recognition, 2007. **40**(3): p. 807-824.
28. Rand, W.M., *Objective Criteria for the Evaluation of Clustering Methods.* Journal of the American Statistical Association, 1971. **66**(336): p. 846-850.
29. Hubert, L. and P. Arabie, *Comparing partitions.* Journal of Classification, 1985. **2**(1): p. 193-218.
30. Jaccard, P., *Distribution de la flore alpine dans le bassin des Dranses et dans quelques régions voisines.* Bulletin de la Société Vaudoise des Sciences Naturelles, 1901. **37**: p. 241-272.
31. Caliński, T. and J. Harabasz, *A dendrite method for cluster analysis.* Communications in Statistics, 1974. **3**(1): p. 1-27.
32. Maulik, U. and S. Bandyopadhyay, *Performance evaluation of some clustering algorithms and validity indices.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002. **24**(12): p. 1650-1654.
33. Dunn†, J.C., *Well-Separated Clusters and Optimal Fuzzy Partitions.* Journal of Cybernetics, 1974. **4**(1): p. 95-104.

34. Davies, D.L. and D.W. Bouldin, *A Cluster Separation Measure*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1979. **PAMI-1**(2): p. 224-227.
35. Wallace, D.L., *A Method for Comparing Two Hierarchical Clusterings: Comment.* Journal of the American Statistical Association, 1983. **78**(383): p. 569-576.
36. Raftery, A.E., *A Note on Bayesian Factors for Log-Linear Contingency Table Models with Vague Prior Information.* Journal of the Royal Statistical Society, Series B, 1986. **48**(B): p. 249-250.
37. Strehl, A. and J. Ghosh, *Cluster ensembles --- a knowledge reuse framework for combining multiple partitions.* J. Mach. Learn. Res., 2003. **3**: p. 583-617.
38. Shannon, C.E., *A mathematical theory of communication.* SIGMOBILE Mob. Comput. Commun. Rev., 2001. **5**(1): p. 3-55.
39. Amini, A., T.Y. Wah, and H. Saboohi, *On Density-Based Data Streams Clustering Algorithms: A Survey.* Journal of Computer Science and Technology, 2014. **29**(1): p. 116-141.
40. O'Callaghan, L., et al. *Streaming-data algorithms for high-quality clustering*. in *Proceedings 1st International Conference on Data Engineering*. 2002. San Jose, CA, USA, USA: IEEE.
41. Zhang, T., R. Ramakrishnan, and M. Livny, *BIRCH: an efficient data clustering method for very large databases.* SIGMOD Rec., 1996. **25**(2): p. 103-114.
42. Karypis, G., E.-H. Han, and V. Kumar, *Chameleon: Hierarchical Clustering Using Dynamic Modeling.* Computer, 1999. **32**(8): p. 68-75.
43. Kranen, P., et al., *The ClusTree: indexing micro-clusters for anytime stream mining.* Knowledge and Information Systems, 2011. **29**(2): p. 249-272.
44. Wang, W., J. Yang, and R.R. Muntz, *STING: A Statistical Information Grid Approach to Spatial Data Mining*, in *Proceedings of the 23rd International Conference on Very Large Data Bases*. 1997, Morgan Kaufmann Publishers Inc. p. 186-195.
45. Sheikholeslami, G., S. Chatterjee, and A. Zhang, *WaveCluster: a wavelet-based clustering approach for spatial data in very large databases.* The VLDB Journal, 2000. **8**(3): p. 289-304.
46. Agrawal, R., et al., *Automatic subspace clustering of high dimensional data for data mining applications.* SIGMOD Rec., 1998. **27**(2): p. 94-105.
47. Tu, L. and Y. Chen, *Stream data clustering based on grid density and attraction.* ACM Trans. Knowl. Discov. Data, 2009. **3**(3): p. 1-27.
48. Wan, L., et al., *Density-based clustering of data streams at multiple resolutions.* ACM Trans. Knowl. Discov. Data, 2009. **3**(3): p. 1-28.
49. Dempster, A., N.M. Laird, and D.B. Rubin, *Maximum Likelihood from Incomplete Data via the EM Algorithm*, in *Paper presented at the Royal Statistical Society at a meeting organized by the Research Section*. 1976.
50. Dang, X.H., et al. *An EM-Based Algorithm for Clustering Data Streams in Sliding Windows*. 2009. Berlin, Heidelberg: Springer Berlin Heidelberg.
51. Ester, M., et al., *A density-based algorithm for discovering clusters in large spatial databases with noise*, in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. 1996, AAAI Press: Portland, Oregon. p. 226-231.
52. Ankerst, M., et al., *OPTICS: ordering points to identify the clustering structure.* SIGMOD Rec., 1999. **28**(2): p. 49-60.
53. Hinneburg, A. and D.A. Keim, *An efficient approach to clustering in large multimedia databases with noise*, in *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*. 1998, AAAI Press: New York, NY. p. 58-65.
54. Cao, F., et al., *Density-Based Clustering over an Evolving Data Stream with Noise*, in *Proceedings of the 2006 SIAM International Conference on Data Mining*. p. 328-339.
55. Mousavi, M., A.A. Bakar, and M. Vakilian, *Data stream clustering algorithms: A review.* International Journal of Advances in Soft Computing and its Applications, 2015. **7**(Specialissue3): p. 1-15.
56. Csernel, B., F. Clerot, and G. Hébrail. *StreamSamp: DataStream Clustering Over Tilted Windows Through Sampling*. in *ECML PKDD 2006 Workshop on Knowledge Discovery from Data Streams*.
57. Charu, C.A., et al., *A framework for projected clustering of high dimensional data streams*, in *Proceedings of the Thirtieth international conference on Very large data bases - Volume 30 %@ 0-12-088469-0*. 2004, VLDB Endowment: Toronto, Canada. p. 852-863.
58. Gao, J., et al. *An Incremental Data Stream Clustering Algorithm Based on Dense Units Detection*. 2005. Berlin, Heidelberg: Springer Berlin Heidelberg.
59. Liu, L.x., et al. *rDenStream, A Clustering Algorithm over an Evolving Data Stream*. in *2009 International Conference on Information Engineering and Computer Science*. 2009.
60. Udommanetanakit, K., T. Rakthanmanon, and K. Waiyamai. *E-Stream: Evolution-Based Technique for Stream Clustering*. 2007. Berlin, Heidelberg: Springer Berlin Heidelberg.
61. Chairukwattana, R., et al. *Efficient evolution-based clustering of high dimensional data streams with dimension projection*. in *2013 International Computer Science and Engineering Conference (ICSEC)*. 2013.
62. Jia, C., C. Tan, and A. Yong. *A Grid and Density-Based Clustering Algorithm for Processing Data Stream*. in *2008 Second International Conference on Genetic and Evolutionary Computing*. 2008.
63. Meesuksabai, W., T. Kangkachit, and K. Waiyamai. *HUE-Stream: Evolution-Based Clustering Technique for Heterogeneous Data Streams with Uncertainty*. 2011. Berlin, Heidelberg: Springer Berlin Heidelberg.
64. Ackermann, M.R., et al., *StreamKM++: A clustering algorithm for data streams.* J. Exp. Algorithmics, 2012. **17**: p. 2.1-2.30.
65. Ntoutsi, I., et al. *Density-based Projected Clustering over High Dimensional Data Streams*. in *SIAM International Conference on Data Mining*. 2012.
66. Amini, A. and T.Y. Wah, *LeaDen-Stream: A Leader Density-Based Clustering Algorithm over Evolving Data Stream.* Journal of Computer and Communications, 2013. **1**: p. 26-31.
67. Hyde, R. and P. Angelov. *A new online clustering approach for data in arbitrary shaped clusters*. in *2015 IEEE 2nd International Conference on Cybernetics (CYBCONF)*. 2015.
68. Masmoudi, N., et al. *Incremental clustering of data stream using real ants behavior*. in *2014 Sixth World Congress on Nature and Biologically Inspired Computing (NaBIC 2014)*. 2014.
69. Masmoudi, N., et al., *CL-AntInc Algorithm for Clustering Binary Data Streams Using the Ants Behavior.* Procedia Comput. Sci., 2016. **96**(C): p. 187-196.
70. Ahmed, I., I. Ahmed, and W. Shahzad, *Scaling up for high dimensional and high speed data streams: HSDStream.* CoRR, 2015. **abs/1510.03375**.

71. Choromanski, K., S. Kumar, and X. Liu, *Fast Online Clustering with Randomized Skeleton Sets.* CoRR, 2015. **abs/1506.03425**.
72. Merino, J.A., *Streaming data clustering in MOA using the leader algorithm*, in *Department of Computer Science*. 2015, Universitat Polit`ecnica de Catalunya. p. 122.
73. Hahsler, M. and M. Bolaños, *Clustering Data Streams Based on Shared Density between Micro-Clusters.* IEEE Transactions on Knowledge and Data Engineering, 2016. **28**(6): p. 1449-1461.
74. Khalilian, M., N. Mustapha, and N. Sulaiman, *Data stream clustering by divide and conquer approach based on vector model.* Journal of Big Data, 2016. **3**(1): p. 1.
75. Silva, J.d.A., et al., *An evolutionary algorithm for clustering data streams with a variable number of clusters.* Expert Syst. Appl., 2017. **67**(C): p. 228-238.
76. Xu, J., et al., *Fat node leading tree for data stream clustering with density peaks.* Knowledge-Based Systems, 2017. **120**: p. 99-117.
77. Hyde, R., P. Angelov, and A.R. MacKenzie, *Fully online clustering of evolving data streams into arbitrarily shaped clusters.* Information Sciences, 2017. **382-383**: p. 96-114.
78. Laohakiat, S., S. Phimoltares, and C. Lursinsap, *A clustering algorithm for stream data with LDA-based unsupervised localized dimension reduction.* Information Sciences, 2017. **381**: p. 104-123.
79. Shao, X., M. Zhang, and J. Meng. *Data Stream Clustering and Outlier Detection Algorithm Based on Shared Nearest Neighbor Density*. in *2018 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS)*. 2018.
80. Keogh, E., et al. *An online algorithm for segmenting time series*. in *Proceedings 2001 IEEE International Conference on Data Mining* 2001. San Jose, CA, USA, USA: IEEE.
81. Beringer, J. and E. Hüllermeier, *Online clustering of parallel data streams.* Data & Knowledge Engineering, 2006. **58**(2): p. 180-204.
82. Rodrigues, P.P., J. Gama, and J. Pedroso, *Hierarchical Clustering of Time-Series Data Streams.* IEEE Transactions on Knowledge and Data Engineering, 2008. **20**(5): p. 615-627.
83. Chaovalit, P. and A. Gangopadhyay, *A method for clustering transient data streams*, in *Proceedings of the 2009 ACM symposium on Applied Computing*. 2009, ACM: Honolulu, Hawaii. p. 1518-1519.
84. Yeh, M.Y., B.R. Dai, and M.S. Chen, *Clustering over Multiple Evolving Streams by Events and Correlations.* IEEE Transactions on Knowledge and Data Engineering, 2007. **19**(10): p. 1349-1362.