Pamukkale Univ Muh Bilim Derg, 24(5), 857-863, 2018

Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi

Pamukkale University Journal of Engineering Sciences

Genelleştirilmiş regresyon yapay sinir ağı örüntü katman büyüklüğünü azaltmak için kümeleme tabanlı bir yaklaşım

A cluster based approach to reduce pattern layer size for generalized regression neural network

Mustafa ORAL[1], *Serkan KARTAL*[2*], *Buse Melis ÖZYILDIRIM*[3]

[1,2,3]Bilgisayar Mühendisliği Bölümü, Mühendislik Fakültesi, Çukurova Üniversitesi, Adana, Türkiye.
moral@cu.edu.tr, skartal@cu.edu.tr, mozyildirim@cu.edu.tr

Geliş Tarihi/Received: 14.04.2017, Kabul Tarihi/Accepted: 30.09.2017
doi: 10.5505/pajes.2017.76401
* Yazışılan yazar/Corresponding author
Araştırma Makalesi/Research Article

Öz

Genelleştirilmiş Regresyon Yapay Sinir Ağı (GRYSA) radyal tabanlı çalışan ve genellikle tahminleyici olarak kullanılan denetimli-öğrenimli bir yapay sinir ağı (YSA) modelidir. Kolay modellenebilmesinin yanında hızlı ve tutarlı sonuçlar üretmesi bu algoritmanın güçlü yanlarını oluşturmaktadır. Ancak GRYSA tahmin mekanizmasında, eğitim veri setindeki her örnek veri için örüntü katmanında bir adet nöron tutulmaktadır. Bu nedenle, eğitim veri setinin çok büyük olduğu çalışmalarda örüntü katman yapısı örnek verilerinin sayısıyla aynı oranda büyümekte, yapılan işlem sayısı ve bellek gereksinimi artmaktadır. Bu çalışmada, GRYSA algoritmasının işlem sayısını azaltmaya yönelik olarak literatürde daha önce de denenmiş olan k-ortalama kümeleme algoritması ön-işlemci olarak kullanılmış, literatürdeki çalışmalardan farklı olarak, bu çalışmaların performansını negatif anlamda etkileyen kümeler arasına düşen test verileri bulunarak aykırı veri oluşmasının önüne geçilmiştir. Böylece, örüntü katmanındaki bellek ihtiyacı ve işlem sayısı azaltılırken, kümeleme algoritmasının eklenmesi ile performansta ortaya çıkan negatif etki büyük oranda giderilmiş ve yaklaşık %90 daha az eğitim verisi ile neredeyse aynı tahmin sonuçları elde edilmiştir.

Anahtar kelimeler: Genelleştirilmiş regresyon yapay sinir ağı, Tahminleyici yapay sinir ağı, Örüntü azaltma, Azaltılmış veri seti

Abstract

Generalized Regression Neural Network (GRNN), is a radial basis function based supervised learning type Artificial Neural Network (ANN) which is commonly used for data predictions. In addition to its easy modelling structure, being fast and producing accurate results are the other strong features of it. On the other hand, GRNN employs a neuron in pattern layer for each data sample in training data set. Therefore, for huge data sets pattern layer size increases proportional to the number of samples in training data set, memory requirement and computational time also increase excessively. In this study, in order to reduce space and time complexity of GRNN, k-means clustering algorithm which had been used as pre-processor in the literature is utilized and outlier data emergence which affects the performances of previous studies negatively, is prevented by identifying test data located between clusters. Hence, while memory requirement in pattern layer and number of calculations are reduced, negative effect on the performance emerged by the use of clustering algorithm is significantly removed and almost the same prediction performances to that of standard GRNN are achieved by using 90% less training samples.

Keywords: Generalized regression neural network, Prediction neural network, Pattern reduction, Reduced dataset

1 Giriş

GRYSA Donald Specht tarafından 1991 yılında geri beslemeli yapay sinir ağlarına alternatif bir model olarak geliştirilmiştir [1],[2]. Geri beslemeli yapay sinir ağlarından farklı olarak tek bir geçişte tüm hesaplamaları yapabilmekte, hızlı bir şekilde sonuç üretebilmektedir. Ancak herhangi bir ön-işlemci algoritmasının kullanılmadığı Standart GRYSA (SGRYSA) da örüntü katmanındaki nöron sayısının, eğitim setindeki veri sayısına eşit olması sebebiyle, eğitim veri setinin büyük olduğu problemlerde ağ yapısı eğitim veri setiyle aynı oranda büyük olmakta ve yapılan işlem sayısı artmaktadır. Bu problemi ortadan kaldırmak için literatürde kümeleme algoritmaları gibi yöntemler GRYSA algoritmasından önce uygulanmış ve test verisinin tahmini sırasında kullanılabilecek en uygun veriler tespit edilmeye çalışılmıştır. Bu işlemler sonucunda, eğitim veri sayısının büyük ölçüde azaltılmasıyla beraber hem yapılan işlem sayısının hem de bellek gereksiniminin minimuma indirilmesi amaçlanmıştır.

Kümeleme algoritmalarının temel amacı, veri seti içerisinde bulunan benzer verileri gruplayarak bir araya getirmektir [3]-[5]. Böylece bir test verisinin sonucu GRYSA algoritması ile tahmin edilmek istenildiğinde, yalnızca eğitim verisinin ilgili grubu kullanılarak aynı sonucun elde edilebileceği düşünülmüştür. Buna ilave olarak, GRYSA algoritmasında sonuca etki eden tek parametre olan sigma (σ) parametresinin her veri kümesi için en uygun olacak şekilde yeniden ayarlanması ile tahmin işlemlerinin daha az hata ile yapılması amaçlanmıştır.

Kümeleme algoritmalarının birçok yöntem ve uygulamada kullanılmalarından dolayı, literatürde farklı ihtiyaç ve uygulamalara göre çeşitlilik gösteren birçok kümeleme algoritması bulunmaktadır [6],[7]. Bu nedenle, veriye en uygun kümeleme algoritmasının seçilmesi, işlemin performansı açısından büyük önem taşımaktadır

Literatürde kümeleme algoritmaları, bölümleme yöntemleri, hiyerarşik yöntemler, yoğunluk tabanlı yöntemler, ızgara tabanlı yöntemler ve model tabanlı yöntemler olmak üzere beş gruba ayrılmaktadır. Bölümleme yönteminde, veri seti önceden belirlenen sayıda alt kümelere ayrılır [8]. Verilerin her biri kullanılan benzerlik/farklılık fonksiyonuna bağlı olarak alt kümelerden birine dâhil edilir. Bu yöntemin en popüler algoritmaları k-ortalama, k-medoid ve CLARA-CLARANS algoritmalarıdır [9],[10]. Hiyerarşik yöntemlerde ise nesneler benzerliklerine uygun olarak Dendogram adı verilen

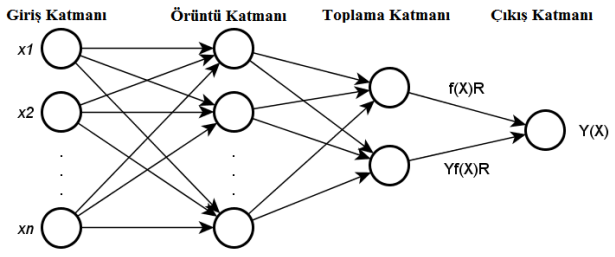
bir ağaç yapısı üzerine yerleştirilerek gruplandırılırlar [8]. Yoğunluk tabanlı yöntemlerde, bir yoğunluk fonksiyonu aracılığı ile verilerin doğal dağılımı tespit edilir ve eşik yoğunluğunu aşan bölgeler küme olarak adlandırılırlar [4]. Izgara tabanlı yöntemlerde, veri uzayı sonlu sayıda bölgelere ayrılmakta, daha sonra bölgelerden elde edilen istatiksel bilgilerle kümeleme işlemleri yapılmaktadır [7]. Eldeki verilerin matematiksel model ile ifade edilmeye çalışıldığı yöntemler ise model tabanlı yöntemler olarak adlandırılmaktadırlar [4]. İterasyonlar boyunca model içerisindeki parametreler, en iyi kümeleri verecek şekilde ayarlanırlar.

Her ne kadar kümeleme algoritmaları GRYSA algoritmasının işlem karmaşıklığını azaltmak için kullanılsalar da, literatürdeki çalışmalar incelendiğinde tahmin sonuçlarındaki hata miktarlarında artışlar olduğu da gözlemlenmiştir [11],[12]. Bu sorunun sebebi ise, GRYSA'nın aykırı verilerle (outlier) karşılaşması durumunda yüksek hatalı tahminler yapmasıdır. Kümeleme algoritmaları kullanıldığında, kümelerin sınır değerlerinde kalan test verileri, sadece bir kümeye ait eğitim verileri ile tahmin edilmekte ve o küme için aykırı veriler gibi davranarak yüksek miktarda hataya sebep olmaktadırlar.

Bu çalışmada, literatürde bulunan ön işlemci algoritmalarından farklı olarak sadece kümeleme ön işlemi yapılmamış, aynı zamanda, kümeler arasına düşen test verileri bulunarak aykırı veri oluşmasının önüne geçilmiştir. Önerilen algoritma, literatürde yaygın olarak kullanılan on beş farklı test fonksiyonu ve sekiz gerçek veri seti ile test edilmiş, elde edilen sonuçlar SGRYSA ve k-ortalama kümeleme algoritmasının ön işlemci olarak kullanıldığı GRYSA algoritmalarının sonuçları ile karşılaştırılmıştır. Elde edilen sonuçlar, öne sürülen yaklaşımın k-ortalama kümeleme ön-işlemci algoritmasının kullanılması ile ortaya çıkan hatayı büyük oranda yok ettiğini ve çok az miktarda eğitim verisi kullanılarak SGRYSA algoritmasıyla neredeyse aynı performansın elde edilebileceğini göstermiştir.

2 GRYSA'nın yapısı

GRYSA giriş katmanı, örüntü katmanı toplama katmanı ve çıkış katmanı olmak üzere dört katmandan oluşan ileri beslemeli bir YSA modelidir. Geri beslemeli YSA'lardan farklı olarak iteratif bir eğitim süreci gerektirmez. Yapısında bulunan her bir katman farklı sayılarda nöronlardan oluşup, katmanlar sırası ile bir sonraki katman ile bağlantılıdırlar [1]. GRYSA'nın genel yapısı Şekil 1'de verilmiştir.



Şekil 1: GRYSA'nın genel yapısı.

İlk katman olan giriş katmanında, nöron sayısı verilerin özelliklerinin sayısına, bir başka deyişle verinin boyutuna eşittir. Örüntü katmanında, önceden de belirtildiği gibi nöron sayısı eğitim setindeki veri sayısına eşittir. Bu katmanda bulunan nöronlarda, eğitim verisi ile test verisi arasındaki mesafeler hesaplanır ve sonuçlar σ değeri ile birlikte radyal tabanlı fonksiyondan geçirilerek ağırlık değerleri elde edilir. Örüntü katmanından elde edilen bu ağırlık değerleri, toplama katmanındaki pay ve payda nöronlarına beslenir. Payda nöronunda, tüm ağırlık değerleri direkt toplanırken; pay nöronunda, örüntü katmanındaki her nörondan gelen ağırlık değerleri ile ilgili nöronda bulunan eğitim verisinin çıkış değeri çarpılır ve bu çarpım değerlerinin toplamı, pay nöronunun çıkış değeri olarak kabul edilir. Son olarak, çıkış katmanında pay nöronundan gelen değer, payda nöronundan gelen değere bölünerek ve sonuç değeri elde edilir [1].

3 Önceki çalışmalar

GRYSA ile yapılan tahmin işlemi sırasında kullanılan eğitim verisi sayısını azaltmak için çeşitli yöntemler sunulmuştur [11]-[17]. İlk olarak GRYSA'nın kümeleme algoritmaları ile birlikte kullanılabileceği fikri Specht tarafından ortaya atılmıştır. Husain ve diğ. [11] ön-işlemci olarak bulanık c-ortalama kümeleme algoritmasını kullanmışlardır. Bu algoritmada önce tek geçiş ile veriler arasındaki benzerlik indeksi değerleri hesaplanmakta, daha sonra indeks değerleri bir eşik değeri ile karşılaştırılarak verilerin birbirlerine benzer olup olmadıklarına karar verilmekte ve benzer olan veriler aynı kümede toplanmaktadır. Mevcut veriler ile benzerlik göstermeyen bir veri ile karşılaşıldığında ise "aykırı veri" olarak adlandırılmakta ve bu veri için yeni bir küme oluşturulmaktadır. Elde edilen kümelerin merkezleri, GRYSA'da kullanılacak eğitim verilerini belirlemek için kullanılmaktadırlar. Öne sürülen algoritma iki farklı veri seti üzerinde test edilmiştir ve SGRYSA ile karşılaştırılmıştır. Elde edilen sonuçlar algoritmanın birinci veri setinde yaklaşık 5 kat, ikinci veri setinde ise %24 oranında daha kötü sonuçlar elde ettiğini göstermektedir.

Benzer şekilde, Zhao ve diğ. [12], örüntü katmanındaki nöronların sayısını düşürebilmek için bulanık ortalama kümeleme algoritmasını kullanmışlardır. Bu kümeleme algoritması, iki adımdan oluşmaktadır. Birinci adımda, her bir örnek verinin kümelere olan benzerliği benzerlik ölçüm metodu ile hesaplanmaktadır. İkinci adımda ise elde edilen en uygun benzerlik değeri eşik değeri ile karşılaştırılmaktadır. Değer eşik değerinin altında ise veri kümeye dâhil edilmekte, değilse veri yeni bir kümenin merkezi olarak kabul edilmektedir. Algoritma bir adet kıyaslama fonksiyonu üzerinde 300 adet örnek veri ile test edilmiştir ve SGRYSA ile karşılaştırılmıştır. Elde edilen sonuçlar, örüntü katmanındaki verilerin büyük ölçüde azaldığını ancak buna karşılık algoritmanın performansında %22.2'lik bir kötüleşme meydana geldiğini göstermektedir.

Bir başka çalışmada Zheng ve diğ. [13] SGRYSA algoritmasında kullanılan nöron sayısını düşürmek ve performansı arttırmak için k-ortalama kümeleme algoritması ile verileri kümelere ayırmış ve her kümeye ait farklı σ değerleri belirlemişlerdir. En iyi σ değerlerinin keşfi sırasında dağılım tahmin algoritmalarından faydalanmışlardır. Sunmuş oldukları algoritmayı, kendi özel veri seti üzerinde test etmişler ve hata miktarının SGRYSA'ya göre %1.62'den %1.24'e düştüğünü belirtmişlerdir.

Hautamaki ve diğ. [18] ise aykırı verileri tespit ederek k-ortalama kümeleme algoritmasının performansını arttırmaya çalışmışlardır. Yapılan çalışmada, küme merkezine maksimum uzaklığa sahip olan veri bulunmakta ve diğer verilerin uzaklık değerleri bulunan maksimum uzaklığa bağlı bir değer olarak hesaplanmaktadır. Bu hesap sonucunda maksimum uzaklığa

sahip olan veriye 1, diğer verilere de uzaklıkları ile orantılı olarak 1 den daha küçük değerler atanmaktadır. Daha sonra bir eşik değeri belirlenmekte ve bu eşik değerinden daha büyük değere sahip olan veriler aykırı veri olarak etiketlenmektedir. Aykırı verilerin kümeleme işlemlerinden çıkarılması ile kümeleme performansı arttırılmaya çalışılmıştır. Yapılan çalışma, üretilen özel veri setleri ve resimler üzerinde test edilmiştir. Çalışmada eşik değerinin dışarıdan girilmek zorunda olunması ve tüm verilerin aykırı veri olma değerlerinin kümeye ait en uzakta bulunan tek bir veriye bağlı olması, çalışmanın eksi yönleri olarak ortaya çıkmaktadır. Buna göre kümeye ait en aykırı veri çok uzak bir noktada ise, diğer aykırı veriler tespit edilemeyebilirler. Tam tersi durumda, yani en aykırı verinin diğer tüm verilere yakın olduğu durumlarda, istenenden çok daha fazla verinin aykırı veri olarak tespit edilmesi durumu ortaya çıkabilmektedir. Aynı metot 2015 yılında Marghny ve Taloba tarafından bu kez genetik k-ortalama algoritması üzerinde kullanılmıştır [19].

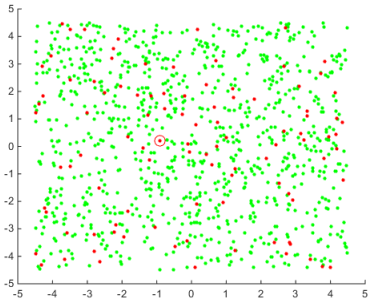
Bir başka çalışmada Ahmed ve Mahmood [20] yeni bir aykırı veri tespit algoritması öne sürmüşlerdir. Bu çalışmaya göre bir kümedeki aykırı veriler bulunurken ilk olarak tüm verilerin küme merkezine olan uzaklıkları alınmakta ve bu uzaklıkların ortalaması hesaplanmaktadır. Daha sonra her bir verinin küme merkezinden olan uzaklığı ortalama uzaklık cinsinden hesaplanmakta, dışarıdan girilen bir eşik değeri ile karşılaştırılmaktadır. Bu değerin üzerinde kalanlar aykırı veri olarak işaretlenmekte, veri kümesinden çıkarılmakta ve aynı işlemler tekrar etmektedir. Bu süreç, yeniden kümeleme işlemi sırasında, verilerin ait olduğu kümeleri değişmeyene kadar devam etmektedir.

Aykırı veri tespit algoritmaları, önermiş olduğumuz algoritmadan farklı olarak, tespit ettikleri verileri veri setinden çıkarmaya yönelik çalışmalardır. Ancak her gelen veri için sonuç üretilmesi gereken SGRYSA uygulamalarında, ya kullanılamamakta, ya da kullanılabilmeleri için yeniden düzenlenmeleri gerekmektedir. Bu çalışmada ise, literatürdeki aykırı veri tespit algoritmalarından farklı olarak, oluşan aykırı verilerinde en düşük hata ile tahmin edilebilmesi için yeni bir yöntem öne sürülmüştür.

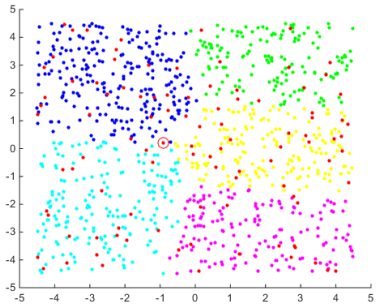
4 İç-aykırı veri tespit algoritması

Kümeleme algoritmalarının ön-işlemci olarak kullanıldığı GRYSA uygulamalarında test verilerinden farklı olan eğitim verilerinin sistemden çıkarılmasıyla, başka bir ifade ile test verilerinin kendilerine daha çok benzeyen eğitim verileri ile tahmin edilmesi nedeniyle hata miktarında büyük bir değişikliğin olmaması gerektiği düşünülmektedir. Ancak, elde edilen ön çalışma sonuçları, yapılan kümeleme işlemleri ile ortaya çıkan ve gözden kaçırılan yeni bir durum olduğunu göstermektedir. Değerlendirme işlemleri sırasında, hata değeri aniden büyüyen test verileri incelendiğinde bu verilerin, kümelerin sınır bölgelerine denk geldikleri tespit edilmiştir. Küme sınırlarına denk gelen test verilerinin, SGRYSA için aykırı veri durumunda olmasalar bile kümeleme sonucunda aykırı veri durumunu oluşturabildikleri ortaya çıkmıştır. Bu durum sonucunda, küme sınırına gelen test verisine benzeyen diğer eğitim verileri başka kümelerde yer aldığı için standart algoritmaya göre daha yüksek hatalı bir tahmin işlemi gerçekleşmiştir.

SGRYSA algoritması içerisinde aykırı veri olmayan ancak kümeleme algoritması sonucunda aykırı veri durumuna düşen bir test verisi sırası ile Şekil 2 ve 3'te gösterilmiştir. Bunun için f_1 fonksiyonunun arama uzayı üzerine rastgele dağılmış 1000 verilik bir veri seti oluşturulmuş ve 900'ü eğitim verisi 100'ü test verisi olacak şekilde rastgele seçilerek ayrılmıştır. Şekil 2 SGRYSA için verilerin nasıl kullanıldığını, Şekil 3 ise kümeleme algoritması sonucunda verilerin nasıl kümelere ayrıldığını ve bir test verisinin tahmini sırasında nasıl çalıştığını anlatmak için kullanılmışlardır.



Şekil 2: SGRYSA.



Şekil 3: K-ortalama kümeleme algoritması ile GRYSA.

Şekil 2'yi detaylı olarak inceleyecek olursak, bu şekil üzerinde bulunan ve yeşil ile verilen noktalar eğitim verilerini gösterirken, kırmızı ile verilen noktalar test verilerini göstermektedir. Kırmızı çember ile işaretlenmiş olan test verisi ise SGRYSA ile tahmin edilecek olan test verisi olsun. Bu durumda ilgili test verisi yeşil ile gösterilen tüm eğitim verileri kullanılarak tahmin edilecektir.

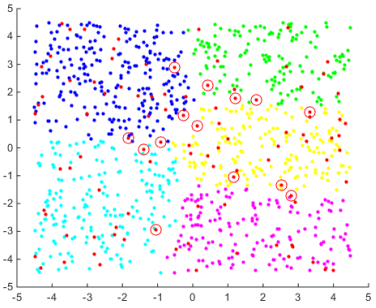
Benzer şekilde Şekil 3'ü inceleyecek olursak, kırmızı ile verilen noktalar test verilerini, onun dışında kalan renkler ise her bir kümeye ait olan eğitim verilerini göstermektedir. Şekilde görüldüğü üzere eğitim verileri k-ortalama kümeleme algoritması kullanılarak 5 kümeye ayrılmıştır. Yine kırmızı çember ile işaretlenmiş olan test verisinin bu sefer k-ortalama kümeleme algoritmasının ön işlemci olarak kullanıldığı GRYSA modeli ile tahmin edileceğini düşünelim. Bu test verisinin, kümeleme işlemi sonucunda mavi kümeye ait olduğu tespit edilmiş olsun. Bu durumda bu test verisi yalnızca mavi ile gösterilen eğitim verilerinin kullanılması ile tahmin edilmeye çalışılacak, yani bu test verisine benzeyen ve etrafında bulunan sarı ve turkuaz renkli kümelere ait diğer eğitim verileri tahmin işlemi sırasında kullanılamayacaklardır. Bu durumda bu test verisi, GRYSA modeli için kümeleme işlemi sonucunda bir iç aykırı veri durumuna geçmiş olacak ve tahmin işlemindeki hata miktarı artacaktır.

İç aykırı veri problemini ortadan kaldırmak için bu çalışmada literatürdeki diğer çalışmalardan farklı olarak kümeler arasında ortaya çıkan iç-aykırı veriler tespit edilmekte ve bu verilerin tahmini sırasında kendilerine komşu olan diğer veri kümeleri de kullanılmaktadır. Aykırı verilerin tespiti sırasında çok değişkenli normal olasılık yoğunluk fonksiyonundan faydalanılmıştır. Bu fonksiyon ile ilk olarak her test verisi için

kümelere olan üyelik değerleri hesaplanmaktadır. Hesaplanan üyelik değerlerinin toplamı 1 değerini vermektedir. Verilerin hangi kümeye ait olduklarına karar vermek için bir eşik değeri belirlenmektedir. Üyelik değerleri hesaplanan test verisi, üyeliği eşik değerinin üzerinde bulunan tüm kümelere üye olarak kabul edilmekte ve birden fazla kümeye aynı anda üye olan veriler iç-aykırı veriler olarak işaretlenmektedir. Eğer bir test verisinin hiçbir kümeye üyeliği bulunmazsa, yalnızca o test verisi için eşik değeri yarıya düşürülerek aynı işlemler tekrar yapılmakta, döngü test verisinin ait olduğu kümeler belirlenene kadar devam etmektedir. Kullanılan veri setlerinin farklı arama uzayı sınırlarına sahip olmalarından dolayı, yapılan ön çalışmalar doğrultusunda tüm veri setlerinde ortak olarak kullanılmak üzere belirlenen eşik değeri fonksiyonu Eşitlik 1'te verilmiştir.

$$ED = \left(\frac{1}{|al - ul| * 2}\right)^{n-1} \quad (1)$$

Fonksiyonda verilen *al* ve *ul* sırası ile arama uzaylarının alt ve üst limit değerlerini, *n*, verinin boyutunu, ED ise hesaplanan eşik değerini göstermektedir. Eşik değeri fonksiyonuna bağlı olarak tespit edilen iç-aykırı veriler Şekil 4 üzerinde gösterilmiştir.



Şekil 4. İç-aykırı verilerin tespiti.

5 Testler ve sonuçlar

SGRYSA ve öne sürülen k-ortalama GRYSA yaklaşımlarının performanslarını karşılaştırabilmek ve yorumlayabilmek için literatürde sıklıkla kullanılan on beş farklı kıyaslayıcı test fonksiyonu ve sekiz gerçek veri seti kullanılmıştır [21]. Veri setleri UCI kaynağından indirilmiştir [22]. Kullanılan test fonksiyonları üç boyutlu olup x1, x2 ve y değerlerinden oluşmaktadırlar. Her fonksiyonun arama uzayı birbirinden bağımsızdır. Kullanılan test fonksiyonları ve veri setleri için önemli olan bilgiler sırasıyla Tablo 1 ve Tablo 2'de verilmiştir. Gerçek veri setlerindeki tüm veriler kullanılmadan önce [0,1] aralığına normalize edilmişlerdir.

İlk olarak, SGRYSA ve k-ortalama GRYSA algoritmalarının tümü için en iyi sonucu verebilecekleri σ değerleri bulunmuştur. En iyi σ değerinin aranması sırasında 10-katmanlı çapraz doğrulama tekniği kullanılmıştır. σ değeri test fonksiyonları için 0.1'den başlatılmış, 3'e ulaşana kadar 0.1 büyüklükte adımlarla arttırılmıştır. Veri setlerinde ise normalizasyon işlemi uygulandığı için σ değeri 0.01'den başlatılmış ve 1'e ulaşana kadar 0.01 büyüklükte adımlarla arttırılmıştır. Bu işlem sırasında her katman için tüm σ değerlerindeki hatalar bulunmuş, daha sonra σ değerlerinin 10 katman için ürettikleri hatalar toplanarak toplam hataları karşılaştırılmıştır. En küçük toplam hatayı veren σ değeri, o veri kümesi için en uygun σ değeri olarak kabul edilmiştir. SGRYSA algoritmasında tüm veri seti için bir adet σ değeri bulunurken, kümeleme algoritmasının kullanıldığı GRYSA da her veri kümesi için en iyi σ değeri ayrı ayrı bulunmuştur.

Tablo 1: Test fonksiyonları.

Fonk.	Adı	Arama Uzayı	Min Noktası
f_1	Beale	[-4.5, 4.5]	0
f_2	Rastrigin	[-5.12, 5.12]	0
f_3	Schwefel	[-500, 500]	-837.9658
f_4	Ackley	[0, 0]	0
f_5	Goldstein-Price	[-2, 2]	3
f_6	Six- Hump Camel	[-5, 5]	-1.0316...
f_7	Bird	[-2*π, 2*π]	-106.7645
f_8	Leon	[-1.2, 1.2]	0
f_9	Griewangk	[-100, 100]	0
f_{10}	Giunta	[-1, 1]	0.0602
f_{11}	Booth	[-10, 10]	0
f_{12}	Bukin4	[-15, -5], [-5, 3]	0
f_{13}	Crossfunc	[-10, 10]	0
f_{14}	Himmelblau	[-5, 5]	0
f_{15}	Matyas	[-10, 10]	0

Tablo 2. Gerçek veri setleri.

Veri Kümesi	Veri Sayısı	Veri Boyutu
(V_1) Airfoil Self-Noise	1503	6
(V_2) Air Quality	9358	15
(V_3) Combined Cycle Power Plant	9568	5
(V_4) Concrete Comp. Strength	1030	9
(V_5) Energy efficiency 1	768	10
(V_6) Energy efficiency 2	768	10
(V_7) Wine Quality-Red	1599	12
(V_8) Wine Quality-White	4989	12

K-ortalama kümeleme algoritmasındaki en büyük sorun ise, küme sayısı olan k değerinin belirlenmesinin gerekmesi ve literatürde k'yı tahmin edebilecek verimli bir algoritmanın bulunmamasıdır. Bu nedenle k küme sayısı ön çalışmalara dayanılarak sırası ile 5, 10 ve 20 olarak alınmış, üç farklı değer ile güvenilir bir karşılaştırma yapılması amaçlanmıştır. Bu değerler ile örüntü katmanındaki nöron sayısında sırasıyla %80, %90 ve %95 oranlarında küçülmeler sağlanmıştır.

Değerlendirme işleminde, her bir test fonksiyonu için arama uzayı üzerinde 1000 rastgele veri üretilmiştir. Sonuçların karşılaştırılmasında, σ seçiminde olduğu gibi her GRYSA konfigürasyonu için test işleminde 10-katmanlı çapraz doğrulama tekniği kullanılmıştır. Bu işlem sırasında ilk olarak veri seti on alt gruba ayrılmaktadır. Dokuz alt grup eğitim verisi olarak geriye kalan bir alt grup ise test verisi olarak kullanılmaktadır. Bu işlem on defa, her bir alt grup bir defa test veri grubu olarak kullanılana kadar tekrarlanmaktadır. Test sonuçları üzerindeki rasgeleliğin etkisini ortadan kaldırmak ve sonuçların güvenilirliğini sağlamak için her bir konfigürasyon on farklı rasgele sayı üretici başlangıç değeri ile test edilmiş ve ortalama hata değerleri sunulmuştur.

Çalışmamızda, algoritmaların performanslarının karşılaştırılabilmesi için üç farklı hata değeri hesaplanmıştır: ortalama karesel hata (OKH), ortalama mutlak hata (OMH) ve ortalama mutlak yüzdesel hata (OMYH)'dır. OKH tahmin işlemlerinin sonucunu değerlendirmek için en sık kullanılan hata değeridir. İstenen değere olan uzaklığın karesi alınarak hesaplandığından, bu ölçüm miktarında aykırı verilerin

sonuca etkisi büyüktür ve bu yüzden önerilen algoritmanın değerlendirilmesinde önemlidir. OMH hataların mutlak değerlerinin ortalamasıdır ve ölçülen verilerle aynı ölçeği kullanır. Bu hata miktarı OKH ya göre aykırı verilerden daha az etkilenmektedir. OMYH ise tahmin sonuçlarının başarısını istatistiksel olarak yorumlamak için kullanılan bir ölçüm yöntemidir. Elde edilen sonucun gerçek sonuca göre ne kadarlık hata yüzdesi ile hesaplandığını vermektedir. Örnek olarak, gerçek sonuçların büyük değerlere sahip olduğu yerlerde, tahmin değeri ile aradaki karesel fark miktarı (OKH) büyük iken, hatanın gerçek değere olan oranı (OMYH) çok küçük olabilir. Bu yüzden bu çalışmada algoritmaların daha sağlıklı bir biçimde değerlendirilebilmesi amacı ile üç farklı yöntem ile de hata ölçümleri yapılmış, algoritmalar tüm sonuçlara bakılarak değerlendirilmişlerdir. Hata fonksiyonları, sırasıyla Eşitlik 2, 3 ve 4'te verilmişlerdir.

$$\text{OKH} = \frac{1}{n}\sum_{i=1}^{n}(A_i - P_i)^2 \tag{2}$$

$$\text{OMH} = \frac{1}{n}\sum_{i=1}^{n}|A_i - P_i| \tag{3}$$

$$\text{OMYH} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{A_i - P_i}{A_i}\right| * 100\% \tag{4}$$

Burada, n veri sayısını, A_i veri setindeki i. verinin gerçek değerini, P_i ise i. tahmin değerini göstermektedir.

SGRYSA ve k-ortalama kümeleme algoritmasının ön işlemci olarak kullanıldığı GRYSA algoritmalarının test sonuçları Tablo 3, Tablo A1 ve Tablo A2'de verilmiştir. Tabloda f_i i. test fonksiyonunu, V_i ise i. veri setini göstermektedir. Hata değişim miktarlarının negatif olması hata miktarındaki artış oranını, pozitif olması ise hata miktarındaki düşüş oranını, bir başka ifade ile performanstaki iyileşme oranını göstermektedir. SGRYSA ile elde edilen hata değerleri ile kümeleme algoritmasının ön-işlemci olarak kullanılmasıyla elde edilen hata değerleri arasındaki değişim yüzdesi yine aynı tablolarda gösterilmiştir.

Elde edilen sonuçlara göre kümeleme algoritması, her ne kadar veri setinin boyutunu büyük ölçüde düşürse de kümelemenin tüm fonksiyonların tahmin performansı üzerinde yarattığı negatif etki yadsınamaz. Özellikle küme sayısının artması ile hata miktarlarının da arttığı gözlemlenmiştir. Beş kümeli GRYSA algoritmasında OKH için hata miktarındaki artışın, standart algoritmaya göre %17'ye kadar çıkabildiği, küme sayısının ona çıkarılması ile hatadaki artış yüzdesinin %37'lere çıktığı, yirmiye çıkarıldığında ise hata miktarında %66.28 oranında artış olduğu görülmektedir. Benzer şekilde OMH ve OMYH için de hata miktarlarının benzer oranlarda hem test fonksiyonları hem de gerçek veriler için arttığı ve artışın küme sayısı ile orantılı olarak büyüdüğü net bir şekilde görülmektedir.

Testlerin ikinci aşamasında ise iç-aykırı verilerin tespiti ile aykırı veri durumlarının oluşmasının önüne geçilmiş, artan hata miktarları önlenmiştir. Ancak iç aykırı verilerin tespiti işleminden sonra, bu verilerin birden fazla kümeye ait eğitim verisi ile tahmin edilmesi, bu algoritmada kullanılan eğitim verisi miktarını da değiştirmektedir.

Tüm veri setlerinde ilgili küme sayısı için örüntü katmanında kullanılan verilerdeki azalma miktarları Tablo 4'te verilmiştir. Tüm test fonksiyonlarında kullanılmak üzere rastgele oluşturulan eğitim verilerinin örüntüsü aynı olduğundan, fonksiyonlarda kullanılan eğitim verilerinin miktarları da eşit olup, tablonun son satırında gösterilmiştir. Her ne kadar veri karmaşıklığındaki azalma miktarı normal kümeleme algoritmasına göre daha az olsa da kullanılan veri miktarında çok büyük oranda azalma sağlandığı görülmektedir.

Tablo 3: 10-katmanlı çapraz doğrulama için OKH sonuçları.

		Küme Sayıları ve Hata Değerleri			Hata Değişimi(%)		
Fonk.	SGRYSA	5	10	20	5	10	20
f_1	10898398	11545229	11602573	11875118	-5.94	-6.46	-8.96
f_2	49.89282	52.36921	53.63095	57.31444	-4.96	-7.49	-14.88
f_3	11269.23	11649.21	11986.75	12674.46	-3.37	-6.37	-12.47
f_4	0.425362	0.443211	0.449946	0.464853	-4.2	-5.78	-9.28
f_5	1.44E+08	1.5E+08	1.55E+08	1.76E+08	-4.09	-7.9	-22.07
f_6	32375.72	34569.43	36492.27	39989.43	-6.78	-12.71	-23.52
f_7	26.70286	30.34226	34.12043	40.73392	-13.63	-27.78	-52.55
f_8	349.7489	364.7196	375.2067	434.295	-4.28	-7.28	-24.17
f_9	3.150401	3.359533	3.591667	3.984392	-6.64	-14.01	-26.47
f_{10}	0.000402	0.000483	0.000489	0.000623	-19.94	-21.42	-54.75
f_{11}	434.1388	505.4886	597.2155	721.904	-16.43	-37.56	-66.28
f_{12}	668.9467	745.1108	824.7638	922.4584	-11.39	-23.29	-37.9
f_{13}	6.98E-11	7.26E-11	7.38E-11	8.95E-11	-3.96	-5.7	-28.2
f_{14}	290.8321	313.4133	334.0501	365.0672	-7.76	-14.86	-25.53
f_{15}	1.125342	1.317208	1.43761	1.760649	-17.05	-27.75	-56.45
V_1	0.01707	0.01842	0.02242	0.02152	-7.9	-31.33	-26.05
V_2	9.1E-07	9.2E-07	9.2E-07	9.3E-07	-1.34	-1.88	-3.08
V_3	0.00228	0.00235	0.00235	0.00236	-2.92	-2.97	-3.31
V_4	0.02761	0.02583	0.02527	0.02758	6.44	8.45	0.11
V_5	0.00365	0.00462	0.00509	0.00585	-26.84	-39.63	-60.42
V_6	0.00427	0.00489	0.00552	0.00656	-14.54	-29.33	-53.83
V_7	0.01814	0.01844	0.01938	0.02008	-1.64	-6.83	-10.7
V_8	0.01508	0.01543	0.01556	0.01571	-2.31	-3.14	-4.17

Tablo 4: Aykırı veri tespit algoritmasının kullanılması sonucu veri miktarındaki azalma yüzdeleri (%).

	Küme Sayıları ve Veri Azalma Yüzdeleri (%)		
Veri Kümesi	5	10	20
V_1	66	83	93
V_2	75	83	92
V_3	73	85	91
V_4	75	85	92
V_5	67	80	85
V_6	63	76	86
V_7	63	73	84
V_8	65	77	88
f	72	83	90

K-ortalama kümeleme algoritmasının iç-aykırı veri tespit algoritması ile geliştirildiği ve SGRYSA modelinde ön işlemci olarak kullanıldığı yeni modelin sonuçları Tablo 5, Tablo A3 ve Tablo A4'te verilmiştir. Önceki değerlendirme tablolarında olduğu gibi önerilen yaklaşımın sonuçları, SGRYSA algoritmasının sonuçları ile birlikte verilmiş ve hata oranlarındaki yüzdesel değişim miktarları gösterilmiştir. Elde edilen sonuçlar, öne sürülen yaklaşımın üç farklı değerlendirme ölçütü için, tüm test fonksiyonlarında ve gerçek veri setlerinde k-ortalama kümeleme algoritmasının ön-işlemci olarak kullanıldığı GRYSA'dan çok daha iyi sonuçlar verdiğini göstermektedir. Ayrıca algoritmanın test edildiği üç farklı küme sayısı için SGRYSA'ya çok yakın sonuçlar ürettiği görülmektedir. Yapılan ortalama hata miktarları test fonksiyonları ve gerçek veri setleri için ayrı ayrı hesaplandığında, küme sayısı 20 iken k-ortalama algoritmasının yalın haliyle kullanıldığı testlerde, OKH, OMH ve OMYH ölçümlerinde performansın sırasıyla %30.89-%20.18, %16.72-%12.03 ve %7.69-%5.26 oranlarında düştüğü görülmüştür. Aynı ortalama hata hesabı öne sürülen yaklaşım için yapıldığında OKH ve OMH için sırasıyla performansın %3.52-%11.52 ve %1.79-%7.92 oranlarında düştüğü, OMYH ölçümü için ise performansın test fonksiyonları için %4.4 oranında arttığı gerçek veri seti için ise sadece %1.91 oranında düştüğü ve bu sonuçların yalnızca yaklaşık %90 daha az eğitim verisi ile elde edildiği görülmüştür.

Özellikle test fonksiyonlarında hata miktarının önemli ölçüde azalması, bu fonksiyonlar için kullanılan veri setlerinde normal kümeleme işlemi ile çok fazla aykırı verinin ortaya çıktığını ve öne sürülen algoritma ile bu problemin büyük oranda çözüldüğünü göstermektedir. Gerçek veri setlerinde ise durumun biraz daha farklı olduğu görülmektedir. Değişim miktarının en az olduğu V_1 veri setinde küme sayısı 5 iken normal k-ortalama-GRYSA algoritması kullanılması sonucunda OKH için %7.9 kötüleşme olurken, öne sürülen algoritma ile bu oran %2.69'a düşürülmüştür. Ancak, küme sayısının arttığı durumlarda, örneğin küme sayısı 20' ye çıkarıldığında hata miktarının tekrar arttığı görülmektedir. Bu durum, aykırı veri eşik değerinin bu veri setinde büyük küme sayıları için çok uygun olmadığını göstermektedir.

Yapılabilecek özel bir çalışma ve bulunabilecek daha uygun bir aykırı veri eşik değeri ile sonuçlarda oluşan olumsuz etkinin çok daha makul seviyelere çekilebileceği düşünülmektedir. Veri setleri V_5, V_6, V_7 ve V_8 de ise hata oranlarının yaklaşık %50 oranında düşürüldüğü, hesaplanan aykırı veri eşik değerleri ile çok başarılı sonuçlar elde edildiği görülmektedir. Diğer taraftan V_2, V_3 ve V_4'te ise normal k-ortalama- GRYSA algoritması ile edilen hata oranlarının SGRYSA ile çok yakın olduğu görülmektedir. Bu sonuçlara bakarak bu veri setlerinde iç aykırı veri durumunun çok fazla ortaya çıkmadığını, hatta kümeler için bulunan özel sigma değerleri ile V_4 'ün performansında iyileşmeler olduğu, aykırı veri bulma yönteminin eklenmesi ile de sonuçların bir adım daha ileriye taşındığı gözlemlenebilmektedir.

Tablo 5: 10-katmanlı çapraz doğrulama için OKH sonuçları.

		Küme sayıları ve Hata değerleri			Hata değişimi(%)		
Fonk.	SGRYSA	5	10	20	5	10	20
f_1	1.1E+07	1.2E+07	1.2E+07	1.1E+07	-5.86	-5.7	-4.87
f_2	49.8928	50.3544	49.7612	50.455	-0.93	0.26	-1.13
f_3	11269.2	11342.9	11383.5	11380.6	-0.65	-1.01	-0.99
f_4	0.42536	0.41939	0.41111	0.41422	1.41	3.35	2.62
f_5	1.4E+08	1.5E+08	1.4E+08	1.5E+08	-1.78	-0.53	-2.92
f_6	32375.7	33744.1	34476.9	34414.3	-4.23	-6.49	-6.3
f_7	26.7029	27.9878	26.9844	27.9771	-4.81	-1.05	-4.77
f_8	349.749	358.306	354.857	359.247	-2.45	-1.46	-2.72
f_9	3.1504	3.21331	3.20674	3.19708	-2	-1.79	-1.48
f_{10}	0.0004	0.00043	0.00042	0.00043	-6.9	-3.58	-6.27
f_{11}	434.139	451.364	463.04	464.625	-3.97	-6.66	-7.02
f_{12}	668.947	681.905	693.013	688.228	-1.94	-3.6	-2.88
f_{13}	7E-11	7.1E-11	7.2E-11	7.2E-11	-1.75	-3.26	-2.74
f_{14}	290.832	300.889	307.144	303.694	-3.46	-5.61	-4.42
f_{15}	1.12534	1.19376	1.17401	1.20364	-6.08	-4.33	-6.96
V_1	0.01707	0.01753	0.02128	0.02128	-2.69	-24.63	-24.65
V_2	9.1E-07	9.2E-07	9.1E-07	9.3E-07	-1.1	-0.94	-2.48
V_3	0.00228	0.00229	0.00228	0.00228	-0.29	0.1	0.37
V_4	0.02761	0.02552	0.02592	0.02757	7.55	6.11	0.12
V_5	0.00365	0.00402	0.00404	0.0046	-10.21	-10.85	-26.09
V_6	0.00427	0.0046	0.00488	0.00564	-7.78	-14.33	-32.18
V_7	0.01814	0.01808	0.01852	0.01906	0.35	-2.09	-5.03
V_8	0.01508	0.01515	0.01522	0.01542	-0.44	-0.92	-2.22

Sonuç olarak tüm test fonksiyonları incelendiğinde, k-ortalama kümeleme algoritması literatürdeki çalışmalarda bulunan şekilde GRYSA algoritmasına ön işlemci olarak kullanıldığında eğitim verisi sayısını önemli ölçüde düşürdüğü, ancak bu iyileştirme karşısında performansta büyük kayıplar yaşandığı görülmüştür. Öte yandan, kümeleme algoritmasına iç-aykırı verilerin tespiti yöntemi uygulandığında, SGRYSA için kullanılan veri setinde %90'lara varan azalma sağlamasının yanı sıra k-ortalama kümeleme ön-işlemci algoritmasının kullanılması ile ortaya çıkan hatalar önemli ölçüde ortadan kaldırılmıştır.

6 Sonuç

GRYSA modeli her ne kadar verimli bir tahmin algoritması olsa da, eğitim verisinin çok büyük olduğu durumlarda ağ yapısı da aynı oranda büyümektedir. Bu yüzden özellikle eğitim verilerinin büyük olduğu problemlerde, veri setini küçültecek ön işlemci algoritmalara ihtiyaç duyulmaktadır. Bu çalışmada test verisinin tahmini işleminde en uygun verilerin seçilmesi için ön işlemci olarak k-ortalama kümeleme algoritması kullanılmıştır. Kullanılan algoritmanın performansının değerlendirilmesi için OKH, OMH ve OMYH adı altında üç farklı hata ölçüm kriterinden faydalanılmıştır. Kümeleme algoritmasının kullanıldığı hibrit GRYSA modeli SGRYSA modeli ile karşılaştırıldığında, eğitim veri setinde önemli ölçüde azalma sağlandığı ancak buna karşılık performansta da büyük oranda kayıplar yaşandığı gözlemlenmiştir. Yapılan çalışmalar ile kümeleme algoritmasının kullanılması ile ortaya çıkan iç-aykırı veri problemi tespit edilmiş ve sorun büyük ölçüde çözülmüştür. Böylece, hem veri setinde yaklaşık %90 oranında azalmalar sağlanmış, hem de performans üzerindeki kayıplar minimuma indirilmiştir.

7 Kaynaklar

[1] Specht DF, Shapiro PD. "Generalization accuracy of probabilistic neural networks compared with backpropagation networks". *IJCNN-91-Seattle International Joint Conference on Neural Networks*, Seattle, WA, USA, 08-12 July 1991.

[2] Specht DF. "A general regression neural network". *IEEE Transactions on Neural Networks*, 2(6), 568-576, 1991.

[3] Jain AK, Murty MN, Flyn PJ. "Data clustering: a review". *ACM Computing Surveys*, 31(3), 264-323, 1999.

[4] Fahad A, Alshatri N, Tari Z. "A survey of clustering algorithms for big data: taxonomy & empirical analysis". *IEEE Transactions on Emerging Topics in Computing*, 2(3), 267-279, 2014.

[5] Berkhin P. *Grouping multidimensional data*. Berlin, Germany, Springer, 2006.

[6] Harkanth S, Phulpagar BD. "A survey on clustering methods and algorithms". *International Journal of Computer Science and Information Technologies*, 4(5), 687-691, 2013.

[7] Kotsiantis SB, Pintelas PE. "Recent advances in clustering: a brief survey". *WSEAS Transactions on Information Science and Applications*, 1, 73-81, 2004.

[8] Rama B, Jayashree P, Jiwani S. "A Survey on clustering, current status and challenging issues". *International Journal on Computer Science and Engineering*, 2(9), 2976-2980, 2010.

[9] Kaufman L, Rousseeuw PJ. "*Clustering by Means of Medoids*". North-Holland, Elsevier, 1987.

[10] Wei CP, Lee YH, Hsu CM. "Empirical comparison of fast clustering algorithms for large data sets*". Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, Maui, Hawaii, 7 January 2000.

[11] Husain H, Khalid M, Yusof R. "Automatic clustering of generalized regression neural network by similarity index based fuzzy C-means clustering*". IEEE Region 10 Conference*, Chiang Mai, Thailand, 24-24 November 2004.

[12] Zhao S, Zhang J, LI X, Song W. "A Generalized Regression Neural Network Based on Fuzzy Means Clustering and Its Application in System Identification". *In International Symposium on Information Technology Convergence*, Jeonju, Korea, 23-24 Nov. 2007.

[13] Zheng LG, Yu MG, Yu SJ, Wang W. "Improved prediction of nitrogen oxides using GRNN with k-means clustering and EDA". *Fourth International Conference on Natural Computation*, Jinan, China 18-20 October 2008.

[14] Yuen RKK, Lee EWM, Lim CP, Cheng GWY. "Fusion of GRNN and FA for online noisy data regression". *Neural Processing Letters*, 19, 227-241, 2004.

[15] Mu T, Nandi AK. "Breast cancer detection from FNA using SVM with different parameter tuning systems and SOM-RBF classifier". *Journal of the Franklin Institute-Engineering and Applied Mathematics*, 344, 285-311, 2007.

[16] Seng TL, Khalid M, Tusof R. "Adaptive GRNN for the Modeling of Dynamic Plants". *Proc. of the 2002 IEEE International Symposium on Intelligent Control Vancouver*, Vancouver, Canada, 27-30 October 2002.

[17] Hamzacebi C. "Improving genetic algorithms' performance by local search for continuous function optimization". *Journal of Applied Mathematics and Computation-AMC*, 196(1), 309-317, 2008.

[18] Hautamäki V, Cherednichenko S, Kärkkäinen I, Kinnunen T, Fränti P. "Improving k-means by outlier removal". *In Scandinavian Conference on Image Analysis*, Joensuu, Finland, 19-22 June 2005.

[19] Marghny, MH, Taloba, AI. "Outlier detection using improved genetic k-means", *International Journal of Computer Applications*, 28(11), 33-36, 2011.

[20] Ahmed, M, Mahmood, AN. "A novel approach for outlier detection and clustering improvement". *In Industrial electronics and applications (ICIEA),* 2013 IEEE 8th Conference on Industrial Electronics and Applications (ICIEA), Melbourne, Australia, 19-21 June 2013.

[21] Tang K, Li X, Suganthan PN, Yang Z, Weise T. "Benchmark Functions for the CEC'2010 Special Session and Competition on Large Scale Global Optimization". Nature Inspired Computation and Applications Laboratory, University of Science and Technology of China, 85, 2009.

[22] Lichman, M. "Irvine, CA: University of California, School of Information and Computer Science". [http://archive.ics.uci.edu/ml], 2013.