

YEEM : Yazılım Projeleri Maliyet Tahminleme Ölçev Seti ve Modeli

Murat AYYILDIZ¹, Oya KALIPSIZ¹ ve Sırma YAVUZ³

^{1,2,3}Bilgisayar Mühendisliği Bölümü, Yıldız Teknik Üniversitesi
Beşiktaş, İstanbul. *e-posta: {f0100301, kalipsiz, sirma}@ce.yildiz.edu.tr*

Abstract—The metric-set selection has a vital role in software cost estimation studies; its importance has been ignored especially in neural network based studies. In this study we have explored the reasons of those disappointing results and implemented different neural network models using augmented new metrics. The results obtained are compared with previous studies using traditional metrics. To be able to make comparisons, two types of data have been used. The first part of the data is taken from the Constructive Cost Model which is commonly used in previous studies and the second part is collected according to new metrics in a leading international company in Turkey. The accuracy of the selected metrics and the data samples are verified using statistical techniques. The models presented here are based on Multi-Layer Perceptron and Elman Networks for both COCOMO'81 metric set and for the augmented metric set (YEEM). Another difficulty associated with the cost estimation studies is the fact that the data collection requires time and care. To make a more thorough use of the samples collected, k-fold, cross validation method is also implemented. It is concluded that, as long as an accurate and quantifiable set of metrics are defined and measured correctly, neural networks can be applied in software cost estimation studies with success

1. GİRİŞ

Ölçme gerçek dünyadaki varlıkların özelliklerine onları tanımlayabilmek için açıkça tanımlanmış kurallara göre sayı veya sembol atanması işidir. Bir varlığı ölçmek diye bir süreçten bahsedemeyiz. Bir varlığın bir özelliğini tanımlanmış bir kurala göre ölçme işinden bahsedebiliriz.

Bu çalışmada öncelikli olarak belirlenmiş yöntemlere göre yazılım maliyetleme çalışmalarında kullanılabilen yeni bir ölçev kümesi oluşturmuş ve sonra bir çok yapay sinir ağı modeli incelemesinden sonra bir MLP ve Elman kullanarak bir model ortaya koymuştur.

2. OLUŞTURULAN ÖLÇEV KÜMESİ

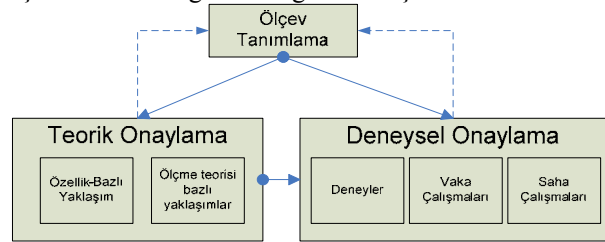
2.1 Ölçev Kümesi Belirleme Metodolojisi

Ölçev (Metric) bir sistemin veya bir parçanın verilen bir özelliğinin nicel ölçüm derecesidir. Ölçev hesaplanabilir veya birleşik bir gösterge olabilir. Bir yazılım ihtiyacının

büyükliğünün ölçülebileceği fikrini ilk ortaya çıkaran ve bir sistem kuran Allan Albrecht'tir. 1979'da IBM'de çalışan Albrecht işlev puan analizi adıyla bir method ortaya koymuştur. California'da bir danışmanlık firması olan TRW'de bulunan bir çok proje üzerinde çalışarak, Boehm COCOMO'yu ortaya koymuştur. COCOMO göreceli olarak dosdoğru bir modeldir. Yapay sinir ağı kullanarak yazılım maliyet tahminleme konusunda yapılmış olan birçok çalışmada COCOMO '81 ölçev kümesini kullanmıştır. Ancak kullanılabilen seviyede iyi sonuçlar alınmamıştır. Burada kritik nokta ölçev setinin oluşturulmasına yeteri kadar önem verilmemesidir. Bu yüzden biz çalışmadan önce bir ölçev seti oluşturmaya karar verdik. Oluşturduğumuz bu ölçev setine YEEM (Yıldız Effort Estimation Metrics) ismini verdik.

Birçok çalışmada COCOMO '81 ölçev kümesini kullanması dolayısıyla oluşturduğumuz ölçev kümesinde COCOMO '81'i içermesi bize çalışma sonuçlarını bir nebze karşılaştırabilme şansı tanıyacağına önemini ihmal etmedik. Oluşturulan ölçev kümesi temel olarak COCOMO 81, COCOMO II, 21 adet gerçek yazılım projesini, toplam 90 yıllık yazılım geliştirme tecrübesi olan 28 yazılım geliştirme proje yöneticisi öneri ve fikirleri üzerinde çalışılarak hazırlanmıştır.

Literatürde ölçev belirleme konusunda oldukça az çalışma vardır. Yazılım mühendisliği ölçevlerini belirlemek oldukça zordur. İyi tanımlanmamış hedefler, iyi tanımlanmamış hipotezler ve gözlemsel onaylama süreçleri eksikliği temel nedenleridir[1]. Ölçev belirleme çalışması metodolojik olarak tanımlanmalıdır. Şekil 1'de kullandığımız ölçev belirleme çalışmasının temel görevleri gösterilmiştir.



Şekil.1 : Yazılım ölçevlerinin tanımlama ve onaylama methodu

Biz net ölçüm hedefleri üzerinde bir ölçev seti oluşturmaya çalıştık. Bu yüzden hedefe yönelik ölçüm (GQM : Goal Oriented Measure) yaklaşımını ölçev seçme ve onaylama süreci için kullandık. Basili ve Weiss [2] veri toplama

metodolojisini YEEM'I oluşturmak için kullandık Bu metodolojinin geribeslemeli 6 adımı vardır. Bu altı adım:

- Yazılım projeleri maliyet göstergeleri veri toplaması ile ilgili hedefleri belirledik
- İlgi soru listelerini oluşturduk,
- Ölçev-Kümesi kategorilerini oluşturduk.
- Veriyi topladık ve onayladık
- Veriyi analiz ettik.

Yapılan belirleme çalışmasından sonra teorik ve deneysel onaylama (validation) sürecini tamamlamamız gerekti. Teorik onaylama süreci temelde iki türdür:

- Özellik bazlı yaklaşımlar ([3], [1])
- Ölçüm teorisi bazlı yaklaşımlar ([4], [5], [6])

Bu çalışmada YEEM'in teorik onaylama süreci için "distance framework" modelini kullandık. Bu model Poels ve Dedene[6] tarafından ortaya konmuş ölçme teorisi bazlı yazılım ölçev onaylama için kavramsal bir çatıdır. Bu çatı bu bildirinin kapsamı dışındadır. Detayları kaynaklar [6] kısmında bulabilirsiniz. Bu çatının 5 aktivitesi vardır. Bunlar (i) Bir ölçüm soyutlaması bulunur, (ii) Uzaklık tanımlanır, (iii) Ölçüm soyutlamaları arasında ki uzaklıklar nitelendirilir (iv) Referans bir soyutlama bul, (v) Bu özellik için bir ölçme tanımla. Deneysel onaylama süreci bu tür çalışmalar aslında gerçek bir kanıttır. YEEM ölçev kümesi için saha çalışmalar ve vaka çalışmaları yapılmıştır.

2.2 YEEM ölçev kümesi

Ölçev belirleme çalışmamızda dikkat ettiğimiz temel noktalar şunlardır:

- Ölçevler ayrık, yalıtılmış olmalıdır.
- Süreç, kişi, ürün bazında mümkün olduğunca çok ölçev bulup daha sonra yapılacak bir analiz ile en yararlıların bulunmasının sağlanmalıdır.
- Ölçevlerin birbirleriyle ilişkili olmaması gerekmektedir. Birbirleri üzerinde etkisi olmamalıdır veya çok az olmalıdır. Bu sürekli izlenmelidir.
- Ölçev bulma, çıkarma süreci sistematik ve düzenli olmalıdır. Sürekli iyileştirme yapmak gerekmektedir.
- Ölçevlerin doğruluğu gerçek hayat ile kontrol edilebilmelidir.
- İstatistiksel yöntemle ölçevlerin hata analizleri yapılmalıdır. Böylece gereksiz hatta zararlı ölçevler temizlenmelidir.
- Ölçev bazında mantıklı ve doğru karşılaştırmalar yapabilmek için kişi, süreç ve ürün benzerlikleri ve farklılıkları karşılaştırmaları iyi bilinmelidir.
- Ölçevleri toplarken ölçevlere bulaşanların farkında olmalıdırlar. Burada çok ünlü iki "H" vardır (Heisenberg etkisi ve Hawthorne etkisi.)
- Ölçevler zararlı olabilir. Daha doğrusu ölçevler yanlış kullanılıyor olabilirler. Buna sürekli dikkat edilmelidir.

Ölçev belirleme çalışması yaparken 21 adet gerçek proje, COCOMO 81, yapılmış çalışmalar ve 28 tane 3 yıldan fazla proje yönetim tecrübesi olmuş proje yöneticisiyle birebir görüşüldü. Proje yöneticisinin önerileri bu başlangıç matrisine eklenmiştir. Elde edilen verilen üzerinde dönüşüm matrisi oluşturuldu. Korelasyonlar belirlenip elenmiştir. Matematiksel, istatistiksel ve gözlemsel bir optimizasyon çalışmasından sonra 6 temel grupta toplanmıştır. Bunlar işin

büyüklüğü (ürün), kaynak, risk, teknoloji, ortam ile planlar ve tahminler gruplarıdır.

COCOMO'da belirtilen tüm ölçevler bu önerilen ölçev setinde kullanılmıştır. Bununla beraber bir hiyerarşik yapı getirilmiştir. Bu çalışmada aynı zamanda daha iyi sonuç alabilmek için bir melez model kullanılmıştır

Birçok ölçevin yanı sıra işlev puanı kullanımı ve baştan yapılan tahminlerinde girdi olarak kullanımı melez bir durum yaratmaktadır ve başarı ihtimalini artırmaktadır. Elde ettiğimiz ölçev kümesi Tablo 1'de gösterilmiştir. 6 Ana ölçev grubu ve bunlara bağlı 24 ölçevi bu tablodan görebilirsiniz. Bu 24 tane ölçevin bazılarında alt ölçevleri vardır. Bunlarla beraber aslında toplam 56 tane girdi ölçevi vardır. Bu alt ölçevleri bu tabloda gösterip tabloyu karmaşık hale getirmemek için tablodan sonra açıklamaya çalıştık.

Tablo 1. Önerilen Ölçev Seti

Ana Ölçev	Ölçev (Metric)
İşin Büyüklüğü (Ürün)	1. Karmaşıklık
	2. İşlev Puan (Function Point)
	3. Önem
	4. Ayrılan Bütçe
	5. Ürünün beklenen özellikleri
Kaynak	6. Çalışanın yeterlilikleri
	7. Çalışanların Projeye katılım oranları
	8. Çalışan Kişi Sayıları
	9. Donanım Durumu
Risk	10. Bütçe Değişme Riski
	11. Çalışan Riski
	12. Donanım Riski
	13. Ürünün tanımının ve kapsamının değişme riski
Teknoloji	14. Yaz. Geliştirme araçlarının kullanım kolaylığı
	15. Yaz. Geliştirme araçlarının kullanım tecrübesi
	16. Yaz. Geliştirme Araçlarının Kullanımı
	17. Modern Programlama Teknikleri
Ortam	18. Ortamın genel özellikleri
	19. Sahiplenilme (Her bir paydaş türünün projeyi sahiplenmesi)
	20. Baskı
	21. Zaman kullanım durumu
	22. Verimlilik durumu
Planlar ve Tahminler	23. Tahmin
	24. Planlar

i. İşin Büyüklüğü

6 temel gruptan biri olan işin büyüklüğünü bulabilmek için işlev puanı kullanmanın yanısıra ürünün karmaşıklığı, önemi, beklenen özellikleri almak büyüklük temel ölçevini daha doğru olmasını sağlayacaktır. Karmaşıklığın 3 alt ölçevi olarak Veritabanı büyüklüğü, Ürün karmaşıklığı ve yeniden kullanılabilirlik olarak belirledik. Bunların üçüde COCOMO'da DATA, CPLX ve RUSE olarak geçmektedir.

İşlev puanı bir yazılım uygulamasının büyüklüğünün ölçevidir. Bu büyüklük işlevsellik ölçümüdür. Yazılım dilinden, metodolojiden, proje takımının yeterliliğinden bağımsızdır. İşlev puanının nerede zorlandığını bilmek önemlidir. İşlev puanı bir uygulama için gereken çabayı ölçmek için mükemmel değildir. Ancak işin büyüklüğünü belirleme anlamında önemlidir. Çoğunlukla gerçek hayatta işin büyüklüğünü anlamak için benzetim kullanılır. İşlev puanı kullanmanın iki avantajı vardır. Bize ürünün büyüklüğünü ve ürünün görünen büyüklüğünü verir. Bunu girdi olarak kullanmak faydalı olacaktır. Dolayısıyla oluşturduğumuz ölçek setine işlev puanını da bir ölçek olarak ekledik.

Bir projeye ayrılan bütçe o projenin temel parçalarından biridir. Dolayısıyla projenin büyüklüğü ve önemi ile ilgili genel bir fikir verir. Bu açıdan ayrılan bütçe büyüklüğünü bir ölçek olarak kullanmak faydalı olacaktır. Bununla beraber nihai ürünün önemide direkt gereken çabayı etkileyecektir. Bu bize toplam baskıyı vermenin yanısıra o proje atanacak çalışanların yetkinlikleri ile ilgili bir bilgi vermiş olacaktır. Hatta yönetsel destek ile ilgili önemli bir parametre olması dolayısıyla projenin planlanan zamanda bitmesini etkileyecektir.

Ürünün beklenen özellikleri ölçeğinin alt ölçeklerini belirledik. Bunlar esneklik, güvenilirlik, kullanım kolaylığı, bakım yapılabilirlik, güvenlik ve dökümantasyon'dur.

ii. Kaynak

Kaynağın büyüklüğü ve niteliği direkt olarak sonucu etkilemektedir. Basit olarak bir işin maliyetini bulmak istersek ürünün büyüklüğünü ve dolayısıyla gereken kaynağın büyüklüğünün ne olduğunu bilmeye ihtiyacımız vardır.

Çalışan yetkinliklerini ise gruplara göre alt ölçeklerini oluşturduk. Bunlar proje yöneticinin yetkinliği, yazılım geliştiricilerin yetkinliği, sistem analistlerin yetkinliği ve test edenlerin yetkinlikleridir.

Çalışan kişi sayılarını gruplara göre alt ölçeklerde kapsamak faydalı olacaktır. Yazılım geliştirici sayısı, analist sayısı, test yapanların sayısı ve genelde bir olmasına rağmen olası durumlar için proje yöneticisi sayısı alt ölçeklerdir.

Donanım durumu ölçeğinin de alt ölçekleri vardır. Bunlar zaman kısıtı, depolama kısıtı, platform değişkenliğidir. Bunların üçüde COCOMO'da mevcuttur ve adları TIME, STOR ve PVOL'dur.

iii. Risk

Risk etkisi riskin oluşma olasılığı ve bunun maliyetinin çarpımı olarak tanımlanabilir. Dolayısıyla ürün ve kaynak risklerini ve etkilerine ihtiyacımız vardır. Dolayısıyla her birinin alt ölçeklerini belirledik. Her birinin iki alt ölçeği vardır. Bunlar olma olasılığı ve olunca yaratacağı etkidir.

iv. Teknoloji

Yazılım geliştirme araçları ve tecrübe oldukça önemlidir ve sonucu oldukça önemli bir biçimde etkiler. Yazılım geliştirme araçlarının kullanım kolaylığı yanında yer alan yazılım geliştirme araçlarındaki tecrübeninde alt ölçeğini belirledik. Bunlar platform tecrübesi, uygulama tecrübesi, yazılım dili ve çoklu ortam geliştirme durumu tecrübesidir.

Bunlar COCOMO'da PEXP, AEXP ve LTEX, SITE olarak geçmektedir.

v. Ortam

Yazılım geliştirilen kurumun yapısı maliyetleri direkt etkilemektedir. Bu kurumdaki ortalama gecikme, ortalama sapma, baskı, sahiplenme ve sorumluluk gibi durumlar sonucu direkt etkilemektedir. Ortamın genel özellikleri ölçeğinin alt ölçeklerini geciken proje oranı, kurumdaki ortalama gecikme oranı, ortalama gecikme miktarı, gecikmenin standart sapma miktarıdır.

Sahiplenme çok önemli bir ölçektir. Bunun alt ölçekleri proje yöneticisinin, yönetimin, müşterinin, proje çalışanlarının sahiplenme durumları alt ölçeklerdir.

Baskı miktarı projenin gecikip gecikmeyeceğine ve maliyete etkileyen önemli bir ölçektir. Bunun 3 alt ölçeği vardır. Bunlar pazar baskısı, müşteri baskısı, yönetim baskısıdır.

Zaman yönetimin alt ölçekleri bir çalışanın ortalama gün içinde kesilme (interrupt) sayısı, bir kesilmenin ortalama süresi, kesilme sonrası önceki duruma geçme ortalama süresi, bu kurumda ortalama yapılan fazla mesai süresidir.

vi. Planlar ve Tahminler

Uzman görüşü çoğu yerde en çok kullanılan yazılım maliyeti tahmin etme yöntemidir. Bu yöntem maalesef mevcut yöntemler arasında hala en başarılı olandır. Bu görüşüde bir ölçek olarak almak faydalı olacaktır. Bu uzmanların tahmin sapma oranlarında bir ölçek olarak aldığımız takdirde daha doğru sonuçlar alabileceğiz. Tahmin ölçeğinin alt ölçekleri tahmini zaman planı, Tahmini yapankişi veya kişilerin ortalama sapma oranıdır. Plan ölçeğinin alt ölçeği planlanan zaman, yanılma oranı ve hedeflenen zamanda bitmemesi durumunda katlanılabilirlik şeklindedir.

3. YAPAY SINIR AĞI MODELİ

3.1 Algoritma

Bu çalışmada biz çok katmanlı ileri beslemeli ve geri yayımlı yapay sinir ağı (MLP) ve Elman yapay sinir ağı modellerini kullandık. Birçok denemeden sonra lineer olmayan eğri uydurmada zaten başarısı sabitlenmiş olan Levenberg-Marquardt [5] eğitim algoritması olarak seçilip uygulanmıştır.

Yapay sinir ağı kullanarak yapılan yazılım maliyet tahminleme çalışmalarında genellikle çok katmanlı ileri beslemeli (MLP) ağlar kullanılmıştır. Ancak dinamik sistemleri modellemede yenilenen yapay sinir ağları (RNN) daha iyi sonuç verebilmektedir. RNN kısmen veya tamamen bağlı olabilir. Elman ağı [8], bilinen en iyi kısmi-bağılı RNN'lerden biridir. Bu çalışmada geri yayımlı Elman ağı modeli eğim iniş yöntemi kullanarak eğitilmesi de çalışılmıştır.

Her iki modelimizde hem COCOMO '81 hem de YEEM kullanılmıştır. Ölçek veri kümesi ise bu modele girdi olarak verilmiştir.

3.2 Veri İşleme

a. Rasgelelik ve istisnaların kontrolü

Henüz herhangi bir işlem yapılmaksızın verinin rasgeleliğini ve istisnaların incelenmesi oldukça önemlidir. Literatürde rasgeleliği ve istisnaları bulmaya yarayan bir çok test yöntemi vardır. Bu testler temel olarak teorik ve deneysel olarak 2

kategoriye ayrılabilir. Bu çalışmada eğitim ve test veri kümesi rasgelelik kontrolü "Run" testi [9] ile yapılmıştır. Yaptığımız çalışmalarda COCOMO '81'de 3 tane istisnai veri olduğu gözlemlenmiştir. COCOMO '81 63 adet projeden oluşmaktadır. Bu 3 tane istisnai verisini çıkarılmasıyla geriye kalan 60 tane veri ile çalışma yapılmıştır. YEEM için ise elimizde 109 adet veri mevcut idi. Bunlardan 9 tanesi istisnai olduğu görüldü ve çıkarıldı.

b. Verinin ön-işleme ve son-işlemesi

Bu çalışmada veri [0,1] aralığına çevrilerek kullanılmış ve bu aralığa getirmek için minumum-maksimum normalizasyon [10] yöntemi kullanılmıştır.

c. Veri Kümesinin organizasyonu

Bu çalışmada hem COCOMO '81 hemde YEEM veri kümesi kullanılmıştır. COCOMO '81 ile YEEM'i daha iyi ve gerçekçi karşılaştırmak için aynı veri sayısı içinde karşılaştırmaya karar verdik. Bu yüzden YEEM'i 60 adet verilik ve 100 adet verilik olarak iki küme olarak değerlendirmeye aldık. Dolayısıyla toplam 3 veri kümesi kullandık. Her veri kümesini eğitim ve test için iki gruba ayırdık. Bu çalışmada kullanılan veri kümeleri, veri sayısı, eğitim ve test için ayrılan veri sayısı tablo 1'de gösterilmiştir. Üçüncü veri kümesinin başarısı aslında bu çalışmada oryanta konan modeli gerçek değerini verecektir.

Tablo 2. Veri Kümelerinin Organizasyonu

	Veri Kümesi 1	Veri Kümesi 2	Veri Kümesi 3
Ölçev Kümesi	COCOMO	YEEM	YEEM
Veri Sayısı (Proje Sayısı)	60	60	100
Eğitim kümesi	45	45	75
Test kümesi	15	15	25

4. UYGULAMA SONUÇLARI

4.1 Değerlendirme Kriteri

Bir modeli test etmek için ve ne kadar iyi çalıştığı ölçmek için hata fonksiyonunu tanımlamak gerekiyor. Literatürde bu konuda az olan çalışmalarda genel kabul görmüş değerlendirme kriteri Ortalama bağıl hatadır (MMRE). Ortalama bağıl hata, bağıl hataların toplamının veri kümesinde bulunan toplam sayıya bölünmesi ile bulunabilir (1). Aşağıda gösterilen hesaplamada "N" kullanılan veri kümesinin veri sayısıdır.

$$MMRE = \frac{1}{N} \sum_i \frac{|Gerçekle\ şen_i - Hesaplanan_i|}{Gerçekle\ şen_i} \quad (1)$$

MMRE yeteri kadar iyi olmasına rağmen insan algısı için yüzdesel gösterimide faydalı olacaktır. Bu yüzden MMRE'nın 100 katınıda tablolarda göstermekteyiz.

4.2 Uygulama Sonuçları

3 veri kümesi için hem MLP hem de Elman ağıları kullanmak suretiyle bu çalışma için 6 değerlendirme yapılmıştır. Çalışma sonuçları tablo 2'de gösterilmiştir.

Tablo 3. COCOMO ve YEEM veri kümeleri ile MLP ve Elman çalışmaları sonuçları

Yapay Sinir Ağı Modeli	Veri Kümesi	Veri Sayısı (Proje Sayısı)	MMRE	Ort. % Hata
MLP	COCOMO	60	1.805	180.50
MLP	YEEM	60	0.280	28.00
MLP	YEEM	100	0.207	20.70
ELMAN	COCOMO	60	1.312	131.20
ELMAN	YEEM	60	0.240	24.00
ELMAN	YEEM	100	0.191	19.10

4.3 Çapraz Onaylama (Cross Validation)

Hata hesaplama teknikleri genel performansı göstermede yardımcı olmaktadır.[2] Bu çalışmada v-kat çapraz onaylama (v-fold cross validation) kullanılmıştır. v-kat çapraz onaylama ile elimizdeki veriyi daha iyi kullanmam sağlamaktadır. Örneğin 15-kat çapraz onaylamada, verinin 1/15'ini ayırıp daha sonra modeli aynı prosedüre göre yeniden inşası yapılmaktadır ve bu işlem 15 kez yapılmaktadır. Bu çalışmada 5,10 ve 15-kat çapraz onaylama yapılmıştır. Yapay sinir ağı çalışmalarını yaparken veri madenciliği konusunda çokça kullanılan WEKA adında araç kullanılmıştır. Üçüncü tabloda 5,10 ve 15-kat çapraz onaylama çalışmasının sonuçlarını görebilirsiniz.

Tablo 4. MLP kullanılarak Yapılan Çapraz Onaylama Sonuçları

	Ver Kümesi 1	Veri Kümesi 2	Veri Kümesi 3
Veri Kümesi	COCOMO	YEEM	YEEM
Veri Sayısı (Proje Sayısı)	60	60	100
Ort.% Hata (5-fold cv)	189.8	21.66	12.89
Ort.% Hata (10-fold cv)	189.0	22.26	12.01
Ort.% Hata (15-fold cv)	166.4	18.97	9.57

5. İLGİLİ ÇALIŞMALAR VE KARŞILAŞTIRMA

Yapay sinir ağı kullanarak yazılım maliyet tahminleme konusunda az sayıda çalışma yapılmıştır. Bu bölümde en başarılı olanları el almaya çalışacağız. Wittig and Finnie [11] bu konuda bir çalışma yapmışlardır. Bu çalışmada çok katmanlı perseptron ve veri olarak Desharnias ve Avusturya ölçev birliği verilerini (ASMA) kullanmışlardır. İki veri kümesininde 3 defa test etmişler ve 10 tane projeyi rasgelelik uygunsuzluğu nedeniyle çıkarmışlardır. MMRE değeri olarak Desharnias için %29 ve ASMA için %17'ye ulaşmışlardır. Diğer önemli çalışmaları özetleyecek olursak

Tablo5. Yapılmış çalışmalar

Çalışma Adı	Kullanılan Algoritma	Veri Kümesi	Proje Sayısı	Sonuç (MMRE)
Venkatachalam [13]	Back-Propagation	COCOMO	63	
Wittig & Finnie	Back-Propagation	Desharnias/ASMA	81/136	%17

Jorgenson [14]	Back-Propagation	Jorgenson	109	%100
Serluca [15]	Back-Propagation	Mermaid-2	28	%76
Karunanithi [17]	Cascade-Correlation			
Samson [12]	Back-Propagation	COCOMO O	63	%428
Srinivasan	Back-Propagation	Kemerer ve COCOMO O	78	%70
Hughes[16]	Back-Propagation	Hughes	33	%55

Bu çalışma’da kullandığımız Elman oldukça başarılı olmuştur. Gözlemliyebildiğimiz kadarıyla en Wittig&Finnie kadar hatta 15-kat çapraz-onaylama ile oldukça iyi bir sonuca ulaşmıştır.

6.SONUÇLAR

Bu çalışmada temel amaç yazılım maliyet tahminlemede kullanılabilinecek bir ölçek kümesi ve bir yapay sinir ağı modeli geliştirmektir. Yazılım maliyet tahminleme çalışmalarında ölçek kümesinin öneminin ihmal edilerek genellikle mevcut kümelerin kullanıldığını ve bu kullanımın sonuca ciddi biçimde etkilediği görülmüştür. Yeni bir çalışma ile yeni bir ölçek kümesi oluşturulmuştur. Bu ölçek kümesi oluşturduktan sonra yapay sinir ağı çalışmalarına geçilmiştir.

Yazılım maliyet tahminleme için yapay sinir ağları kullanımı konusunda literatürde bazı çalışmalar vardır. Bu çalışmalarda genellikle COCOMO ’81 veri kümesinin kullanıldığını görmekteyiz. Yapılan çalışmaların sonuçları günümüz için hala kabul edilebilir, kullanılabilir seviyelere henüz gelmemiştir.

Bu çalışmada karşılaştırma yapabilmek için hem COCOMO ’81 hem de yeni oluşturduğumuz YEEM ölçek kümelerini ayrı ayrı kullandık. COCOMO ’81 ile yaptığımız çalışmalarda yapay sinir ağı kullanarak yazılım maliyet tahminlemesi için bu ölçek kümesinin yetersiz kaldığını gözlemledik. COCOMO ’81 ile bir kaç değişik yapay sinir ağı modeli denememize rağmen sonuç olarak kullanılabilir seviyeye gelmedi.

Oluşturduğumuz yeni ölçek kümesi olan YEEM’i kullandığımızda kullanılabilir seviyede iyi sonuçlar alınmaktadır. Yeni ölçek kümesi ve verisi MLP ve Elman yapay sinir ağları kullanılarak test edilmiştir. Elman ağındaki başarı MLP’ye nazaran daha iyi seviyelerdedir. Ancak MLP’de çapraz-onaylama (cross-validation) yaptığımızda hata oranını %15’in altına indirdiğini görmekteyiz.

Değişik boyutta ölçek kümesi ile çalışıldığından veri sayısının modelin başarısı için önemli bir etken olduğunda görülmektedir.

KAYNAKLAR

- [1] Briand L., Morasca S. and Basili V. (2002). An Operational process for goal-driven definition of measures. IEEE Transactions on Software Engineering, 30(2), 120-140.
- [2] Fenton N. and Pfleeger S. (1997). Software Metrics: A Rigorous Approach. 2nd. edition. London. Chapman & Hall.
- [3] Weyuker E.J. (1988). Evaluating Software Complexity Measures. IEEE Transactions on Software Engineering, 14(9), 1357-1365.
- [4] Whitmire S. (1997). Object Oriented Design Measurement. John Wiley & Sons. Inc.
- [5] Reed, R. D. and Marks, R. J.: Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks. MIT Press. (1999)
- [6] Poels G. and Dedene G. (2000). Distance-based software measurement: necessary and sufficient properties for software measures. Information and Software Technology, 42(1), 35-46.
- [7] Krantz D., Luce R.D., Suppes P. and Tversky A. (1971). Foundations of Measurement. Vol. 1. Academic Press. New York.
- [8] Leung, H., Fan, Z.: In Handbook of Software Engineering and Knowledge Engineering (Ed, Chang, S. K.). Volume 2 World Scientific. (2002)
- [9] Knuth E. D.: The art of computer programming. 2nd ed. Addison-Wesley. (1981)
- [10] Pyle, D.: Data Preparation for Data Mining. Morgan Kaufmann. (1999)
- [11] Wittig, G., Finnie, G.: Estimating Software Development Effort with Connectionist Models. Information and Software Technology. Volume 39 (1997) 469-476
- [12] Samson, B., Ellison, D., Dugard P.: Software Cost Estimation using an Albus Perceptron (CMAC). In Proc Eight International COCOMO Est. Meeting. Pittsburgh. (1993)
- [13] Venkatachalam, A.R.: Software Cost Estimation Using Artificial Neural Networks. International Joint Conference on Neural Networks. Nagoya. (1993)
- [14] Jorgensen M. and Shepperd M.: A Systematic Review of Software Development Cost Estimation Studies. IEEE Transactions on Software Engineering. (2006)
- [15] Serluca, C.: An Investigation Into Software Effort Estimation using a Back-Propagation Neural Network. M.Sc. Thesis. Bournemouth University. (1995)
- [16] Hughes, R.T.: An Evaluation of Machine Learning Techniques for Software Effort Estimation. University of Brighton. (1996)
- [17] Karunanithi, N., Whitley, D., Malaiya, Y.K.: Using Neural Networks in Reliability Prediction. IEEE Software. Volume 9(4) (1992) 53-59