Yazılım Geliştirme Projelerinde Maliyet Tahminleme Çalışmasında Kullanılabilinecek Bir Ölçev Kümesi Ve Bir Yapay Sinir Ağı Topolojisi Önerisi

Oya KALIPSIZ[1], Murat AYYILDIZ[1]

[1]Yıldız Teknik Üniversitesi, Bilgisayar Mühendisliği Bölümü, Barbaros Bulvarı Yıldız, İstanbul

[1]e-posta : {kalipsiz, f0100301}@yildiz.edu.tr

Özet

Yazılım projeleri çaba maliyeti genellikle karmaşık ve pahallı olarak görünmektedir. Yazılım üretimi bir kriz içindedir.Aşırı maliyetlerden müstariptir. Yazılım üretimi çoğu kez kontrol dışındadır çünkü ölçmüyoruz. Eğer ölçemezsen kontrol edemezsin. Bu makale yazılım geliştirme maliyet kestirim çalışmaları için bir ölçev kümesi önermektedir. Bu ölçev kümesi yazılım geliştirme projelerinden, çalışmalardan ve tecrübelerden türetilmiştir. Ölçev kümesi seçiminin aslında yazılım maliyet kestirim çalışmalarında hayati bir rolü vardır. Birçok model COCOMO ölçev kümesi tabanlıdır.Maalesef ölçev kümesinin önemi bir çok çalışmada göz ardı edilmiştir. Ölçev kümesi üzerine çok az çalışma vardır. Aynı zamanda, bu makale önerilen ölçev kümesini girdi parametresi olarak alan çok katmanlı perceptron ağ topolojisi önermektedir.

Abstract

Software project effort estimation is frequently seen as complex and expensive for individual software engineers. Software production is in a crisis. It suffers from excessive costs. Software production is often out of control. It has been suggested that software production is out control because we do not measure. You cannot control what you cannot measure. This paper suggests a metric-set for software development cost estimation studies. This metric-set has been derived from software development projects, studies and experience The metric-set selection is actually have a vital role in software cost estimation studies. Most of the model based on cocomo metric-set. Unfortunately importance of the metric set has been ignored in most of the studies. There is few studies on metric set. Also, this paper suggests a multi-layer perceptron network topology which takes the sugested metric-set as input parameter.

1. Giriş

Mühendisliğin temel yapı taşlarından biri ölçmedir. Yazılım mühendisliğinin bir disiplinden çok bir ideoloji gibi kalmasının temel nedeni yazılım mühendisliğine yön veren kişilerce ölçümün neredeyse tamamen göz ardı edilmesidir. Yazılım mühendisliğini gerçek bir mühendisliğe çevirme anahtarı ölçmedir.

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.

”Yazılım mühendisliği” bir yazılım ürünü inşaa etmek için kullanılan teknikler topluluğunu tanımlamak için kullanılan bir terimdir. Burada ”mühendislik” yaklaşımıyla yönetim, bütçeleme, planlama, modelleme, tasarlama, uygulama, test etme ve bakım konularını içine almaktadır.
Günümüzde yazılım projeleri geliştirirken ölçülebilir hedefler koymakta başarılı olunamamaktadır. Örneğin bir yazılıma başlarken onun “güvenilir”, “bakım yapılabilir”, “kullanıcı dostu” gibi olacağı söylenmekte, net tanımlar verilmemektedir. Ölçme işinin açık ve net hedefleri olmalıdır. Yazılım mühendisliğinde ölçevlerin kullanımı teorik ve pratik çalışmalar arasındaki aralığı küçültecek bir anahtardır. Ölçme ve değerlendirme yapabilmek için sistematik ölçüm yapmak gerekmektedir.
Bir iş için gereken çabayı baştan bilmek çok önemlidir. Bunu bilmeden işleri yönetmek mümkün değildir. Yönetmek için bu tahminleri kullanıp projeleri baştan kabul etme veya reddetmek çok daha faydalıdır. Çok uzun sürebilecek ve çok kaynak gerektirebilecek bir yazılım geliştirme işine girmemek veya girilme kararı verilse bile baştan gerekli iş gücünü organize etmek verimi her aşamada artıracaktır. Bügünün dünyasında yazılım maliyet kestiriminin önemi sürekli artmakta ve yayılmaktadır. Bu konuda yapılmış çalışmalar olmasına rağmen hala bu konu tam olarak aydınlatılabilinmiş değildir ve üzerinde bir çok çalışma yapılması gerekmektedir. Bu konu şu an ihtiyaç duyulandan çok daha yavaş olarak ilerlemektedir. Ölçüm yapabilmeyi başarabildikten sonra aslında temel problem ölçülmesi gerekenleri ortaya çıkarmaktır.
Yazılım projeleri zaman planına ve bütçe planına uymama yönünde kötü ün salmış projelerdir. Bunun temel nedeni gereken iş gücünü baştan yanlış tahmin etmektir. Bütün planları bu yanlış tahminler üzerine kurmakla devam eden bu yanlışlık zinciri yüksek maliyetlere neden olmaktadır. Yazılım mühendisliğinde yazılım projelerinin gereken iş gücünü, zamanı ve bütçeyi tahmin etme yönünde yapılacak çalışmalar bu problemleri azaltacaktır.
Yazılım maliyet tahminlemesi bir yazılım sisteminin inşaası için gereken çaba miktarının tahminleme sürecidir. Yazılım mühendisliği henüz olgunlaşmamış bir disiplindir. Diğer disiplinler gibi ispatı henüz mümkün olmayan tahminleme, plan ve kontrolleri içerir. 1990’larda Standish Group 8000 yazılım projesini incelemiştir. Bu projeler hem kamu hemde özel projelerden oluşmakta idi. Bu projelerin %90’ının bütçe ve zaman aşımı nedeniyle başarısız oldukları görülmüştür. Bunların %33’ü daha tamamlanmadan iptal edilmiştir. Yaklaşık üçte biri ilk tahmin edilen ve bütçelenen maliyeti %150 ile %200 arasında ve ortalama olarak %189 aştığı görülmüştür. Yaklaşık üçte biri ise %200 ile %300 arasında ve ortalamada %222 aştığı görülmüştür. [1]
Bu makalede amaç yazılım geliştirme maliyet tahminleme çalışmalarında kullanılabilinecek bir ölçev seti ve bir yapar sinir ağı topolojisi önermektir. Bu ölçev seti gerçek proje çalışmalarından, proje yöneticileri ile görüşmelerden ve tecrübelerinden alınan veriler üzerinde çalışma yapılarak çıkarılmıştır.

2. Ölçev Kümesi Önerisi

Ölçev (Metric) bir sistemin veya bir parçanın verilen bir özelliğinin nicel ölçüm derecesidir. Ölçev hesaplanabilir veya birleşik bir gösterge olabilir. Bir yazılım ihtiyacının büyüklüğünün ölçülebileceği fikrini ilk ortaya çıkaran ve bir sistem kuran Allan Albrecht’tir. 1979’da IBM’de çalışan Albrecht işlev puan analizi (Function Point Analysis) 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. Orjinal COCOMO modeli 1981’de yayınlanmıştır.[1] Daha sonra bu geliştirilmiş ve COCOMO II adını almıştır.
Bu makale yazılım maliyet tahminlemesinde kullanılabilinecek bir ölçev kümesi önermektedir. Bu konuda yapılmış olan birçok çalışmada COCOMI ’81 ölçev kümesini kullanmıştır. Dolayısıyla

önereceğimiz ölçev kümesinde COCOMO '81'i içermek bize çalışma sonuçlarını bir nebze karşılaştırabilme şansı tanıyacaktır. Bu önerilen 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.

Ölçev belirleme çalışmamızda dikkat ettiğimiz temel noktalar şunlardır:

a. Bir yazılım mühendisliği ölçevi ayrık, yalıtılmış olması avantajlıdır.
b. Süreç, kişi, ürün bazında mümkün olduğunca çok ölçev bulup daha sonra yapılacak bir analiz ile en yararlıların bulunmasının sağlanması faydalıdır.
c. Ölçevlerin birbirleriyle ilişkili olmamaması gerekmektedir. Birbirleri üzerinde etkisi olmamalıdır veya çok az olmalıdır. Bu sürekli izlenmelidir.
d. Ölçev bulma, çıkarma süreci sistematik ve düzenli olmalıdır. Sürekli iyileştirme yapmak gerekir.
e. Ölçevlerin doğruluğu gerçek hayat ile kontrol edilebilmelidir.
f. İstatistiksel ve başka yöntemlele ölçevlerin hata analizleri yapılmalıdır. Böylece gereksiz hatta zararlı ölçevler temizlenmelidir.
g. Ölçev bazında mantıklı ve doğru karşılaştırmalar yapabilmek için kişi, süreç ve ürün benzerlikleri ve farklılıkları karşılaştırmaları iyi bilinmelidir.
h. Ölçevleri toplarken ölçevlere bulaşanların farkında olmalıdırlar. Burada çok ünlü iki "H" vardır (Heisenberg etkisi ve Hawthorne etkisi.)
i. Ölçevler zararlı olabilir. Daha doğrusu ölçevler yanlış kullanılıyor olabilirler. Buna sürekli dikkat edilmelidir.

Yazılım maliyet tahminleme çalışmalarında çoğunlukla COCOMO 81 ölçev seti kullanılmıştır. Bu yüzden önerilen ölçev seti aslında COCOMO bazlı bir ölçev kümesidir. 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 matrise 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 makalenin temel amacı yazılım geliştirme projelerinin maliyet tahminlemesi çalışmalarında kullanılabilinecek ve daha iyi sonuç alınmasını sağlayacak ölçev seti önermektir. Bu makale aynı zamanda daha iyi sonuç alabilmek için bir melez model önermektedir. Birçok ölçevin yanı sıra işlev puanı kullanımı ve baştan yapılan tahminlerinde girdi olarak kullanımı melez bir durum yaratmaktadır ve başarı ihtimalini artırmaktadır. COCOMO'da kullanılmayan bazı ölçevlerde eklenmiştir. Elde ettiğimiz ölçev kümesi Tablo 1'de gösterilmiştir. 6 Ana ölçev grubu ve bunlara bağlı 24 ölçevi bu tablodan görebilirsiniz. Bu 24 tane ölçevin bazılarınında 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. Bu ölçevler ve alt ölçevler için çoğu kez direk bir sayı elde etmek mümkün olmayabilir. Bu durumda bulanık mantık kümeleri kullanmak mümkün olabilir.

Tablo 1. Önerilen Ölçev Seti

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

A. 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 kullanılabilinirlik 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, methodolojiden, proje takımının yeterliliğinden bağımsızdır. İşlev puanın nerede zorlandığını bilmek önemlidir. İşlev puan bir uygulama için gereken çabayı ölçmek için mükemmel değildir. Ancak işin büyüklüğünü belirleme anlamında önemlidir. Çoğunlukla gerçek hayatta işin büyüklüğünü anlamak için benzetim kullanılır. Örneğin 1000 m^2 bir ev yapmak genellikle 2000 m^2 bir ev yapmaktan daha ucuza mal olmaktadır. Ancak mermer banyo, karo fayans gibi birçok parametrenin aslında durumu tamamen değişmesi mümkündür. Dolayısıyla 1000 m^2 bir evin yapımının 2000 m^2 bir ev yapmaktan daha ucuza mal olabilir veya daha pahallıya mal olabilir. Dolayısıyla sadece evin büyük olması bize çaba ile ilgili bir fikir verse bile bunun üzerine maliyeti

hesaplamamız pek mümkün gözükmemektedir. İşlev puan'da bunun gibi bize ürünün büyüklüğünü vermesine karşın tek başına maliyeti bulmamız mümkün değildir. İşlev puanı kullanmanın iki avantajı vardır. Bize ürünün büyüklüğünü ve ürünün görünen büyüklüğünü verir. Bunu girdi olarak kulanmak faydalı olacaktır. Dolayısıyla önereceğimiz ölçev setine işlev puanıda bir ölçev olarak ekledik.
Bir projeye ayrılan bütçe o projenin temel parçalarından biridir. Dolayısıyla projenin büyüklüğü ve önemi ile iligli genel bir fakir verir. Bu açıdan ayrılan bütçe büyüklüğünü bir ölçev olarak kullanmak faydalı olacaktır. Bununda beraber nihai ürünün önemide direk gereken çabayı etkileyecektir. Bu bize toplam baskıyı vermenin yanısıra o proje atanacak çalışanların yetkinlikleri ile ilgili bir bilgi vermiş olacaktır. Hatta yönetimsel destek ile ilgili önemli bir parametre olması dolaysıyla projenin planlanan zamanda bitmesini etkileyecektir.
Ürünün beklenen özellikleri ölçevinin alt ölçevlerini belirledik. Bunlar esneklik, güvenilirlik, kullanım kolaylığı, bakım yapılabilirlik, güvenlik ve dökümantasyon'dur.

B. Kaynak

Kaynağın büyüklüğü ve niteliği direk 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 ölçevlerini oluşturduk. Bunlar proje yöneticisin yetkinliği, yazılım geliştiricilerin yetkinliği, system analistlerin yetkinliği ve test edenlerin yetkinlikleridir.
Çalışan kişi sayılarını gruplara göre alt ölçevlerde kapsamak faydalı olacaktır. Yazılım geliştirici sayısı, analist sayısı, test yapanların sayısı ve genelde bir olmasına rağmen olası durumlar için proje yöneticisi sayısı alt ölçevlerdir.
Donanım durumu ölçevinin de alt ölçevleri vardır. Bunlar zaman kısıtı, depolama kısıtı, platform değişkenliğidir. Bunların üçüde COCOMO'da mevcuttur ve adları TIME, STOR ve PVOL'dur.

C. 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 ölçevlerini belirledik. Her birinin iki alt ölçevi vardır. Bunlar olma olasılığı ve olunca yaratacağı etkidir.

D. 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 ölçevini belirledik. Bunlar platform tecrübesi, uygulama tecrübesi, yazılım dili ve çoklu ortam geliştirme durumu tecrübesidir. Bunlar COCOMO'da PEXP, AEXP ve LTEX,SITE olarak geçmektedir.

E. Ortam

Yazılım geliştirilen kurumun yapısı maliyetleri direk etkilemektedir. Bu kurumdaki ortalama gecikme, ortalama sapma, baskı, sahiplenme ve sorumluluk gibi durumlar sonucu direk

etkilemektedir. Ortamın genel özellikleri ölçevinin alt ölçevlerini geciken proje oranı, kurumdaki ortalam gecikme oranı, ortalama gecikme miktarı, gecikmenin standart sapma miktarıdır.

Sahiplenme çok önemli bir ölçevdir. Bunun alt ölçevleri proje yöneticisinin, yönetimin, müşterinin, proje çalışanlarının sahiplenme durumları alt ölçevlerdir.

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

Zaman yönetimin alt ölçevleri 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.

F. 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 ölçev olarak almak faydalı olacaktır. Bu uzmanların tahmin sapma oranlarınıda bir ölçev olarak aldığımız takdirde daha doğru sonuçlar alabileceğiz. Tahmin ölçevinin alt ölçevleri tahmini zaman planı, Tahmini yapankişi veya kişilerin otalama sapma oranıdır. Plan ölçevinin alt ölçevi planlanan zaman, yanılma oranı ve hedeflenen zamanda bitmemesi durumunda katlanılabilinirlik şeklindedir.

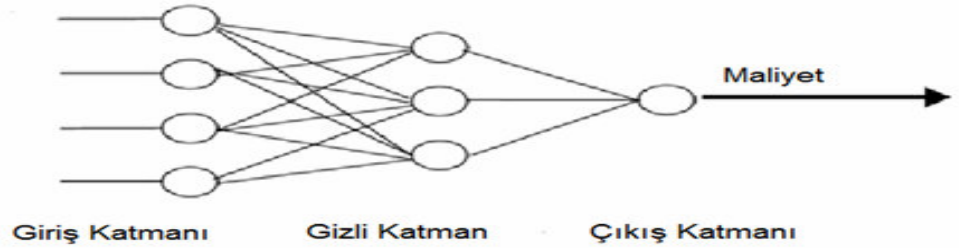
3. Yapay Sinir Ağı Önerisi

Temelde iki tür maliyet tahmin yöntemi vardır : algoritmik ve algoritmik olmayan. Algoritmik modellerin bazıları istatistiği kullanarak basit aritmetik formüller kullanmaktadır. Algoritmik model bir takım değişkenleri kullanan bir fonksiyon (1) olarak düşünebiliriz.

$$\text{Maliyet} = f(x1, x2, \ldots, xn) \qquad (\mathbf{1})$$

Burada x1,x2,..xn', maliyet faktorleri olarak düşünebiliriz. Bir çok maliyet çalışmasında kullanılan COCOMO II modelinin 4 temel maliyet faktörü vardır : ürün faktörleri, bilgisayar faktörleri, personel faktörleri ve proje faktörleri. İkinci kısımda önerdiğimiz ölçev kümesi COCOMO'nun tüm ölçevlerini kapsamaktadır. Ancak bu 4 temel faktör yerine 6 temel faktör getirmiştir.

Maliyet tahminleme aslında oldukça karmaşık bir problemdir ancak bugune kadar gereken ilgiyi görmemiştir. Araştırmacılar değişik yaklaşımlar getirmeye çalışmışlardır. Son zamanlarda yeni yeni yapay zeka yaklaşımları düşünülmeye başlanmıştır.Temel ve basit gösterimle durum şekil 1'de gösterilmiştir. Yapay sinir ağları kullanılarak yapılan bazı çalışmalar vardır. Bu konu henüz yeni gelişen bir konudur. Bu konuda yapılmış çalışmaları tablo 2'de görebilirsiniz.



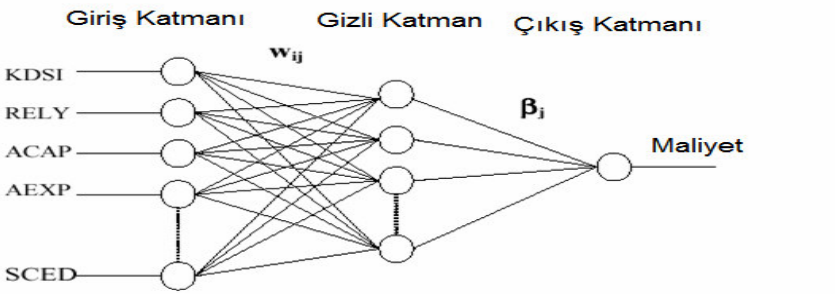
Şekil 1. Yapay Sinir ağı (YSA) Topolojisi

Tablo 2. Yapay sinir Ağı kullanarak Yazılım Maliyet Tahminlemesi yapma konusunda yapılmış çalışmalar

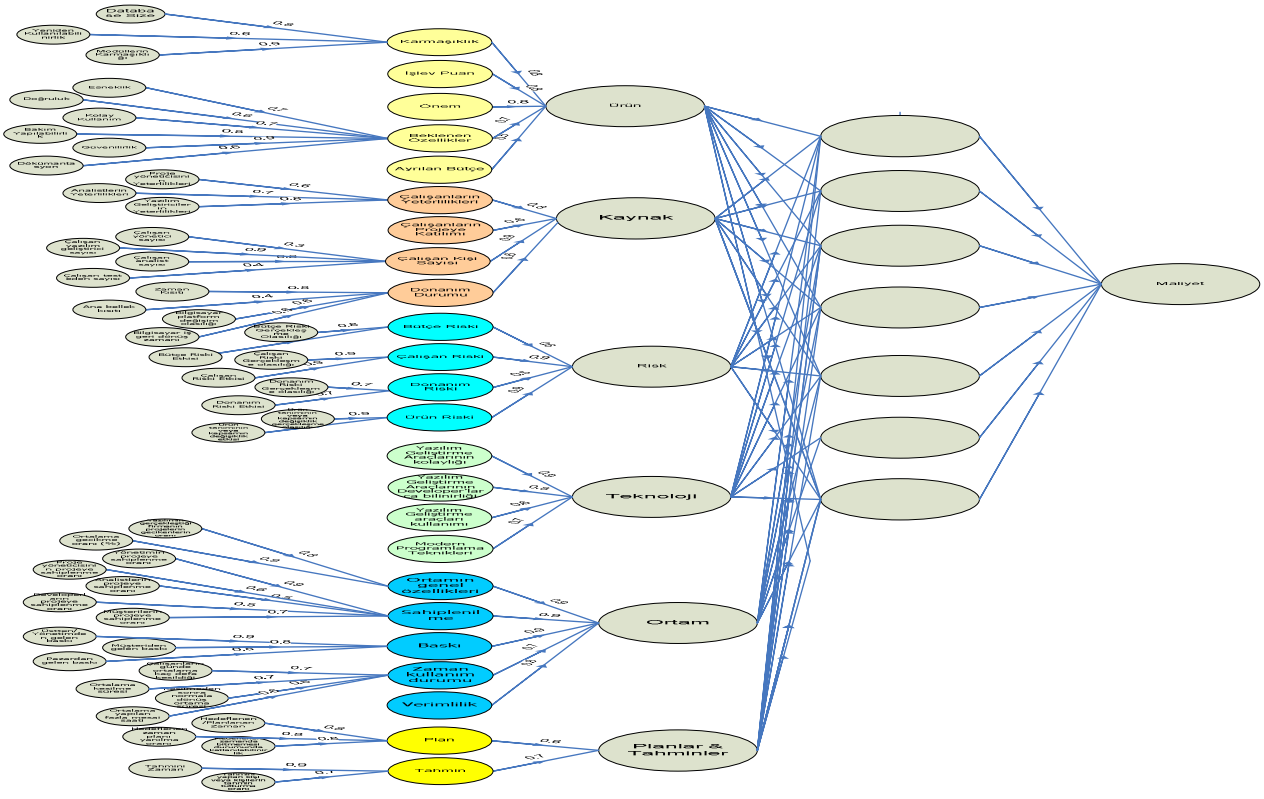
Çalışma	Kullanılan Algoritma	Ölçev Kümesi	Proje Sayısı	Sonuç (MMRE)
Wittig & Finnie [6]	Back-Propagation	ASMA ve Desharnais	136 ve 81	17%
Venkatachalm [7]	Back-Propagation	COCOMO	63	
Jorgenson [8]	Back-Propagation	Jorgensen	109	100%
Serluca [9]	Back-Propagation	Mermaid-2	28	76%
Karunanithi et al [10]	Cascade-Correlation			
Samson et al [11]	Back-Propagation	COCOMO	63	428%
Srinivasan & Fisher [12]	Back-Propagation	Kemerer ve COCOMO	78	70%
Hughes [13]	Back-Propagation	Hughes	33	55%

Yapay sinir ağı kullanarak maliyet tahminlemede iyi bir tahminleme oranı yakalayacağımızı düşünüyoruz. Yapılan çalışmalarda başarıyı artırmak için tahminleme çalışılmış ve girdiler çok önemsenmemiştir. Burada bir farklılık yaratarak önemli bir sonuç farkı olabileceğini düşünmekteyiz. Ölçev kümesinin yazılım maliyet tahmininde çok önemli bir rolü vardır. Bir çok model ölçev kümesi üzerinde çok çalışma yapmadan COCOMO'yu kullanmayı tercih edip çoğunlukla tahminleme yöntemi üzerinde çalışmayı tercih etmişlerdir. Ancak ölçev kümesi sonucu direk etkilemektedir. Dolayısıyla iyi bir tahminleme çalışması yapmadan önce iyi bir ölçev kümesi ile işe başlamak önemlidir. Bu konuda yapılan en ünlü çalışma Wittig ve Finnie'nin yaptığı çalışmadır. [10] Avusturya Yazılım Metric Birliğinin (ASMA) verilerini kullanmışlardır.[6]

Göreceli hatanın miktarının ortalaması (MMRE) göz önüne alınarak en başarılı olanın Wittig & Finnie çalışmasıdır. Bu çalışmada COCOMO ölçev kümesini direk girdi olarak kullanmıştır. (Şekil 1) Bu çalışmayı baz alarak bizim önerdiğimiz ölçev setini bu yapı üzerine oturtabiliriz. Bu durumu şekil 2'de görebilirsiniz. Bu çok katmanlı perceptron ağ topolojisi bizim önerimizdir.



Şekil 2. Wittig & Finnie çalışmasında kullanılan yapay sinir ağı topolojisi



Şekil 3. Önerdiğimiz yapay sinir ağı topolojisi

4. Sonuç

Yazılım projeleri maliyet tahmini yapmadan önce ölçev kümesi oluşturmak durumundayız. Ölçev kümesi çalışmasını gerektiği kadar önemsemeden yazılım projeleri maliyeti tahminleme yöntemleri üzerinde yapılan çalışmalar yeteri kadar başarılı olamamaktadırlar.

Belirlediğimiz ölçevleri düzgün kullanılabilinirsek bize sağlayacağı yararlar :
1. Nicel olarak başarıyı ve başarısızlığı tanımlayabileceğiz. Bir ürünün, bir sürecin veya bir kişinin başarı veya başarısızlığın derecesini nicel olarak tanımlayabileceğiz.
2. Ürünlerimizde, süreçlerimizde ve kişilerdeki gelişimi, gelişim eksikliğini, bozulmayı ve gerilemeyi tanımlayabilir ve ölçebiliriz.
3. Teknik ve yönetimsel kararlar vermemizde yarar sağlar.
4. Trend'leri tanımlamamızı sağlar.
5. Mantıklı tahminler yapmakta kullanılabilinir.

Bu makalenin iki temel amacı vardır. Biri bir ölçev seti önermek diğeri bunu kullanarak bir yapay sinir ağı topolojisi önermektir.

Kaynakça

1 The Standish Group, CHAOS Chronicles, Standish Group Int. Report, 1995.

2 Stephen H.Kan: “Metrics and Models in Software Quality Engineering”, Ad.-Wesley, 2002

3 Gray, A.R. and S.G. MacDonell, A Comparison of Alternatives to Regression Analysis as Model Building Techniques to Develop Predictive Equations for SW Metrics. No. , University of Otago, 1996.

4 E.Fenton : “Software Metrics”,Chapman&Hall

5 “Bayesian Analysis of Software Cost and Qualıty models” ,Sunita devnani-chulani,University of Southern California, Doctor of philosophy Thesis,1999

6 Wittig, G. and G. Finnie, Estimating Software Development Effort with Connectionist Models. Information and Software Technology, vol 39pp469 - 476,1997.

7 Venkatachalam, A.R. Software Cost Estimation Using Artificial Neural Networks. in International Joint Conference on Neural Networks. Nagoya: IEEE, 1993.

8 Jorgensen, M., Experience with the Accuracy of Software Main. Task Effort Prediction Models. IEEE Transactions on Software Engineering, vol 21(8), pp674-681, 1995.

9 Serluca, C., An Investigation Into Software Effort Estimation using a Back-Propogation Neural Network. 1995, M.Sc. Thesis, Bournemouth University:

10 Karunanithi, N., D. Whitley, and Y.K. Malaiya, Using Neural Networks in Reliability Prediction. IEEE Software, vol 9(4), pp53 - 59, 1992.

11 Samson, B., D. Ellison, and P. Dugard. Software Cost Estimation using an Albus Perceptron(CMAC). in Proc. Eight International COCOMO Est. Meeting. Pittsburgh: 1993.

12 Srinivasan, K. and D. Fisher, Machine Learning Approaches to Estimating Software Dev. Effort. IEEE Transactions on Software Engineering, vol 21(2), pp126-136, 1995.

13 Hughes, R.T., An Evaluation of Machine Learning Techniques for Software Effort Estimation. University of Brighton, 1996.