ششمین کنفرانس بین المللی مهندسی صنایع و سیستم ها
۱۹ و ۲۰ شهریور ۱۳۹۹ • دانشگاه فردوسی مشهد
6th International Conference on Industrial and Systems Engineering
• 9 & 10 Sep. 2020 • Ferdowsi University of Mashhad

پیش بینی هزینه نرم افزار به کمک مدل PRO-ABE

سید حمید ثمره موسوی[۱]، سید مصطفی موسوی[۲]، شکوه پور محی آبادی[۳]

[۱] مدیر مرکز نوآوری شایان، دانشگاه جامع علمی کاربردی، مرکز آموزش علمی کاربردی شایان؛ hamid.moosavi17@gmail.com

[۲] کارشناس مسئول اداره حمایت از فعالیت های پژوهش و فناوری، دانشگاه جامع علمی کاربردی استان کرمان؛ m.mousavi@uast.ac.ir

[۳] گروه کامپیوتر، دانشگاه آزاد اسلامی واحد بیرجند؛ shokoohmahyabadi@gmail.com

* نویسنده مسئول: سید حمید ثمره موسوی

چکیده

با توجه به رشد روز افزون فناوری در صنایع مختلف و بکارگیری سیستم های هوشمند در صنایع و سازمان ها، استفاده از نرم افزار مناسب و توسعه نرم افزارهای قبلی یکی از مهمترین ارکان هر صنعتی است. یکی از چالش برانگیزترین مسائل برای توسعه نرم افزار سیستم ها انتخاب روش مناسب برای پیش بینی هزینه توسعه نرم افزار است. روش پیش بینی براساس مقایسه اگر چه یکی از قوی ترین روش ها برای تخمین هزینه های توسعه نرم افزار است، اما این روش نیز در برخی موارد کاستی هایی دارد. مدل پیشنهادی ارائه شده در این مقاله ترکیب روش پیش بینی براساس مقایسه و الگوریتم بهینه سازی فقیر و ثروتمند می باشد. الگوریتم بهینه ساز با وزن دهی صحیح ویژگی های پروژه های نرم افزاری در تخمین دقیق تر هزینه به روش تخمین براساس مقایسه کمک می کند. برای ارزیابی عملکرد مدل پیشنهادی از دو مجموعه داده واقعی استفاده شده است. نتایج بدست آمده از معیارهای ارزیابی و آزمون آماری نشان می دهند که مدل پیشنهادی نسبت به سایر مدل های مورد مقایسه عملکرد مطلوب تری ارائه کرده است.

کلمات کلیدی: هزینه، پیش بینی براساس مقایسه، بهینه سازی فقیر و ثروتمند.

Software Cost Estimation by Using PRO-ABE

Seyyed hamid samareh moosavi[1], seyed mostafa moosavi[2],shokooh pour mahyabadi[3]

[1] Director of Shayan Innovation Center, University of Applied Science and Technology, Center of Shayan; hamid.moosavi17@gmail.com

[2] Expert in the Office of Research and Technology Support, Kerman University of Applied Science and Technology; m.mousavi@uast.ac.ir

[3]Depatment of Computer Engineering, Birjand Branch, Islamic Azad University; shokoohmahyabadi@gmail.com

* Corresponding author: seyyed hamid samareh moosavi

ABSTRACT

As a result of the increasing growth of technology and utilizing intelligent systems in numerous industries and organizations, the employment of appropriate software and the development of preceding software is certainly one of the most crucial bases of any industry. One of the most challenging issues for the system's software development is opting for the appropriate method to predict software development costs. Although the analogy based estimation method is one of the most effective approaches to estimate the software development costs, in some cases, this method has some disadvantages. The proposed model presented in this paper is the combination of the analogy based estimation method and the Poor and Rich optimization algorithm. The optimization algorithm enhances the cost estimation precise in the analogy based estimation method by the achievement of the accurate weighting for software project features. To evaluate the performance of the proposed model, two real data sets have been employed. The results obtained from the statistical evaluation and test criteria demonstrate that the proposed model has been provided better performance than other models compared.

Keywords: cost, comparison-based prediction, poor and rich optimization.

۱- مقدمه

به طور کلی برآورد هزینه به پیش بینی میزان تلاش، میزان نیروی انسانی، زمان و نیازها برای به انجام رساندن یک پروژه نرم افزاری اطلاق مـی گردد [۱]. مدل های برآورد هزینه می توانند به دو دسته مدل های الـگوریتمی و غیـر الـگوریتمی تقسیم شوند. مدل های الگوریتمی براساس یک یا چند فرمول ریاضی که به طور معمول از طریق تجزیه و تحلیل آماری بدست می آیند بنا شده اند [۲]. یکی از

عمده ترین معایب مدل های الگوریتمی عدم تطبیق با شرایط جدید است[۳]. در نتیجه مدل های غیر الگوریتمی ارائه شد که براساس تجزیه و تحلیل پروژه های نرم افزاری قبلی ساخته شده اند. رایج ترین مدل غیر الگوریتمی مدل تخمین براساس مقایسه[1] است در این روش روند برآورد تلاش لازم برای تولید یا توسعه نرم افزار رویـکردی است که شباهت ها بین پروژه جدید و پـروژه های گذشته را می یابد و براساس میزان تشابه تلاش را تخمین می زند.

بسیاری از محققان از مدل های مختلف و مجموعه داده های گوناگون برای پیش بینی هزینه نرم‌افزار استفاده می کنند که اغلب مبتنی بر تکنیک های هوش مصنوعی هستند [۷-۴]. اکثر این محققان از سه گروه ابزار پرکاربرد هوش مصنوعی برای برآورد هزینه نرم افزار استفاده کرده اند. الگوریتم های بهینه سازی، منطق فازی و شبکه های عصبی جزء پرکاربرد ترین ابزارها در زمینه هوش مصنوعی هستند. روشه ارائه شده در این مقاله ارائه مدلی جدید برپایه الگوریتم بهینه سازی فقیر و غنی و روش تخمین بر اساس مقایسه است. در این مدل در هر مرحله از تکرار، الگوریتم بهینه ساز مناسبترین وزن ها را برای ویژگی های پروژه های نرم افزاری برای تخمین هر چه دقیق تر پیشنهاد می دهد. این مقاله در ۶ بخش سازماندهی شده است. در بخش دوم الگوریتم بهینه سازی فقیر و ثروتمند ارائه شده است. در بخش سوم روش پیش بینی براساس مقایسه شرح داده شده است. در بخش چهارم مدل پیشنهادی به طور کامل شرح داده شده است. در بخش های پنجم و ششم نتایج شبیه سازی و نتیجه و جمع بندی به ترتیب ارائه شده است.

۲- الگوریتم بهینه سازی فقیر و ثروتمند

الگوریتم بهینه سازی فقیر و ثروتمند (PRO) در سال ۲۰۱۹ توسط موسوی و خطیبی برای حل مسائل بهینه سازی ارائه شده است [۷]. در این الگوریتم دو دسته جمعیت تعریف شده است. دسته اول شامل افراد ثروتمند می باشد که در این دسته افراد از لحاظ دارایی و ثروت متوسط به بالا هستند. دسته دوم شامل افراد فقیر می باشد که در این دسته افراد از لحاظ دارایی و ثروت متوسط به پایین هستند. در واقع ایده اصلی این الگوریتم در دو مورد زیر خلاصه می شود:

۱. تک تک اعضای جمعیت فقیر با الگو گرفتن از جمعیت ثروتمند سعی در بهبود اوضاع اقتصادی خود و کاهش شکاف طبقاتی را دارند.

۲. تک تک اعضای جمعیت ثروتمند با در نظر گرفتن اعضای جمعیت فقیر و کسب ثروت سعی در افزایش فاصله طبقاتی خود با جمعیت فقیر را دارند.

۲-۱- جمعیت اولیه

در الگوریتم PRO ابتدا یک جمعیت اولیه به صورت تصادفی و با توزیع یکنواخت بین پارامترهای حد بالا و حد پایین ایجاد می شود. سپس این جمعیت اولیه توسط تابع هدف مورد ارزیابی قرار گرفته و براساس مقادیر بدست آمده از تابع هدف به صورت صعودی مرتب می شود. در الگوریتم PRO جمعیت اصلی متشکل از دو زیر جمعیت است. زیر جمعیت اول مربوط به ثروتمندان و زیر جمعیت دوم مربوط به فقرا می باشد. اندازه این دو زیر جمعیت بسته به نوع مسئله می تواند تغییر کند. تعداد جمعیت اصلی در این الگوریتم از رابطه (۱) بدست می آید:

$$POP_{Nmain} = POP_{Npoors} + POP_{Nriches} \quad (۱)$$

که در این رابطه POPNmain تعداد جمعیت اصلی، POPNpoors تعداد جمعیت فقیر و POPNriches تعداد جمعیت ثروتمند است. در الگوریتم PRO چون جمعیت اصلی به صورت صعودی مرتب شده است بخش اول از جمعیت اصلی مربوط به جمعیت ثروتمندان و بخش دوم مربوط به جمعیت فقرا می باشد. در واقع در این الگوریتم موقعیت همه ی اعضای جمعیت ثروتمندان از موقعیت همه ی اعضای جمعیت فقرا بهتر است.

۲-۲- بروزرسانی موقعیت ها

در الگوریتم PRO جمعیت اصلی متشکل از دو زیر جمعیت فقیر و ثروتمند است. در هر تکرار از الگوریتم موقعیت تک تک اعضای هر جمعیت باید براساس مکانیز خاصی تغییر کند. که در ادامه این مکانیز برای هر جمعیت به طور جداگانه مورد بررسی قرار گرفته است.

✓تغییر در موقعیت هر عضو از جمعیت ثروتمند

در هر تکرار از الگوریتم PRO موقعیت هر یک از اعضای جمعیت ثروتمند براساس رابطه زیر تغییر می کند:

$$\overrightarrow{X^{new}_{rich,i}} = \overrightarrow{X^{old}_{rich,i}} + r \times \left[\overrightarrow{X^{old}_{rich,i}} - \overrightarrow{X^{old}_{poor,best}}\right] \quad (۲)$$

[1] Analogy Based Estimation (ABE)

که در این رابطه $\overrightarrow{X^{new}_{rich,i}}$ مقدار جدید موقعیت iام از جمعیت ثروتمند، $\overrightarrow{X^{old}_{rich,i}}$ مقدار کنونی موقعیت iام از جمعیت ثروتمند، r پارامتر میزان شکاف طبقاتی و $\overrightarrow{X^{old}_{poor,best}}$ موقعیت کنونی بهترین عضو از جمعیت فقرا می باشد. هر عضو از جمعیت ثروتمند با ثروتمند تر شدن خود از تک تک اعضای جمعیت فقیر فاصله می گیرد. از آنجا که $\overrightarrow{X^{old}_{poor,best}}$ بهترین فقیر در میان اعضای جمعیت فقیر است، وقتی یک عضو از جمعیت ثروتمند فاصله خود را از $\overrightarrow{X^{old}_{poor,best}}$ بیشتر می کند، در نتیجه از تمامی اعضای جمعیت فقیر فاصله گرفته است. در واقع اعضای جمعیت فقیر هر چه فقیرتر باشند فاصله آنها با اعضای جمعیت ثروتمند بیشتر است. پارامتر r مقداری تصادفی بین صفر و یک دارد و به صورت تصادفی مشخص می کند هر عضو از جمعیت ثروتمند باید به چه اندازه از جمعیت فقیر فاصله بگیرید. تصادفی بودن مقدار r باعث یک رقابت درون گروهی میان جمعیت ثروتمندان می شود. به عبارت دیگر وقتی دو عضو از جمعیت ثروتمند دارای موقعیت های نزدیک بهم هستند با توجه به ثابت بودن $\overrightarrow{X^{old}_{poor,best}}$ پارامتر r مشخص کننده میزان بهبود این دو عضو از جمعیت است. یعنی موقعیت های با هزینه بیشتر ممکن است با گرفتن مقادیر r بزرگتر میزان بهبود بیشتری نسبت به موقعیت های با هزینه کمتر داشته باشند.

✓**تغییر در موقعیت هر عضو از جمعیت فقرا**

در هر تکرار از الگوریتم PRO موقعیت هر یک از اعضای جمعیت فقیر براساس رابطه زیر تغییر می کند:

$$\overrightarrow{X^{new}_{poor,i}} = \overrightarrow{X^{old}_{poor,i}} + [r \times (Pattern) - \overrightarrow{X^{old}_{poor,i}}] \quad (۳)$$

که در این رابطه $\overrightarrow{X^{new}_{poor,i}}$ مقدار جدید موقعیت iام از جمعیت فقیر، $\overrightarrow{X^{old}_{poor,i}}$ مقدار کنونی موقعیت iام از جمعیت فقیر، r پارامتر میزان بهبود الگو و Pattern الگوی ثروتمند شدن است. مقدار Pattern از رابطه زیر بدست می آید:

$$Pattern = \frac{\overrightarrow{X^{old}_{rich,best}} + \overrightarrow{X^{old}_{rich,mean}} + \overrightarrow{X^{old}_{rich,worse}}}{3} \quad (۴)$$

که در این رابطه $\overrightarrow{X^{old}_{rich,best}}$ موقعیت کنونی بهترین عضو از جمعیت ثروتمند، $\overrightarrow{X^{old}_{rich,mean}}$ میانگین موقعیت کنونی اعضای جمعیت ثروتمند و $\overrightarrow{X^{old}_{rich,worse}}$ موقعیت بدترین عضو از جمعیت ثروتمند است. از آنجا که میزان ثروتمند شدن برای هر فرد متفاوت است در هر تکرار میانگین موقعیت سه نماینده از جمعیت ثروتمند بعنوان هدف در نظر گرفته شده است. با توجه به ثابت بودن مقدار Pattern در هر تکرار پارامتر تصادفی r تعیین کننده میزان بهبود الگو و بدنبال آن میزان بهبود $X^{old}_{poor,i}$ است. در واقع هر چه مقدار پارامتر r به صفر نزدیکتر باشد مقدار Pattern بهبود بیشتری خواهد داشت و با توجه به این موضوع که مقدار $X^{old}_{poor,i}$ از مقدار Pattern بزرگتر است پس میزان بهبود $X^{old}_{poor,i}$ بیشتر خواهد بود و برعکس. از آنجا که مقدار پارامتر r به صورت تصادفی محاسبه می شود، باعث یک رقابت درون گروهی میان جمعیت فقرا نیز می شود. یعنی هر چه مقدار پارامتر r کوچکتر باشد، میزان بهبود Pattern بیشتر خواهد بود و در مواردی که موقعیت ها در نزدیکی هم قرار دارند مقدار این پارامتر باعث جابجایی رتبه این موقعیت ها می شود.

۲-۳- جهش

در دنیای اقتصاد عواملی وجود دارند که می توانند باعث بهبود و یا عدم بهبود اوضاع اقتصادی شوند. از آنجا که پیش بینی این عوامل بسیار سخت و در بعضی موارد غیر ممکن است از این عامل در این الگوریتم بعنوان جهش استفاده شده است. در این الگوریتم از یک توزیع نرمال با میانگین صفر و واریانس یک و با احتمال معین برای جهش روی جمعیت فقرا و ثروتمندان به صورت جداگانه استفاده شده است.

۲-۴- ساخت جمعیت جدید بعد از هر تکرار

در پایان هر تکرار از الگوریتم PRO چهار جمعیت قدیمی فقیر، قدیمی ثروتمند، جدید فقیر و جدید ثروتمند وجود دارند. این چهار جمعیت در پایان هر تکرار از الگوریتم توسط تابع هدف مورد ارزیابی قرار می گیرند و سپس با هم ترکیب می شوند. جمعیت مرکب براساس مقادیر بدست آمده از تابع هدف به صورت صعودی مرتب شده و دو زیر جمعیت جدید ثروتمند و فقیر به تعداد تعریف شده از قبل از ابتدای این جمعیت مرکب جدا می شوند.

۳- پیش بینی براساس مقایسه

روش پیش بینی بر اساس مقایسه در سال ۱۹۹۷ برای پیش بینی هزینه نرم افزار توسط شفرد ارائه شد، و این روش به عنوان جایگزینی برای روش‌های الگوریتمی بود [۸]. در این روش، برآورد پارامترهای پروژه‌های نرم افزاری به وسیله مقایسه پروژه هدف با پروژه قبلی و

پیدا کردن مشابه ترین پروژه با پروژه هدف قابل اجرا می‌باشد. ABE دارای ۴ بخش اصلی مجموعه داده های قبلی، تابع شباهت، قوانین بازیابی و تابع راه حل می باشد. فرآیند برآورد ABE شامل مراحل زیر است:

- ✓جمع آوری داده‌ها از پروژه‌های قبلی و تولید مجموعه‌ای از اطلاعات اولیه.
- ✓انتخاب پارامترهای اندازه گیری مناسب مانند FP، LOC.
- ✓بازیابی پروژه‌های قبلی و محاسبه شباهت بین پروژه هدف و پروژه‌های قبلی.
- ✓برآورد تلاش پروژه هدف.

۳-۱- تابع شباهت[۲]

روش مبتنی بر مقایسه از یک تابع شباهت استفاده می‌کند، که ویژگی‌های دو پروژه را مقایسه و میزان شباهت بین آن دو را تعیین می‌کند. دو تابع شباهت رایج فاصله اقلیدسی[۳] و فاصله منهتن[۴] برای این کار استفاده می‌شوند [۸].

در ریاضیات فاصله اقلیدسی یا پارامتر اقلیدسی در اصل فاصله معمولی بین دو نقطه است. این روش در اغلب موارد در مسائلی استفاده می‌شود که در آن فقط فاصله باید مقایسه شود. فاصله منهتن یک شکل از هندسه است که در آن تابع فاصله یا اندازه هندسه اقلیدسی با مقیاس جدید عوض شده است و فاصله بین دو نقطه مجموع تفاوت مطلق مختصاتشان می‌باشد. هر دو تابع به طورگسترده برای اندازه‌گیری درجه شباهت بین پروژه‌های نرم افزاری استفاده می‌شوند. هیچ مبنایی برای انتخاب فاصله اقلیدسی یا فاصله منهتن برای مجموعه داده خاص وجود ندارد و این امر به وسیله آزمون و خطا انجام شده است. ماهیت پروژه‌ها در مجموعه داده و سطح نرمال می‌تواند اثر قابل توجهی بر روی عملکرد توابع شباهت داشته باشد. رابطه (۵) فاصله اقلیدسی را نشان می‌دهد:

$$\mathrm{Sim}(p,p') = \frac{1}{\left[\sqrt{\sum_{i=1}^{n} w_i \mathrm{Dis}(fi,fi')}+\delta\right]} \quad \delta = 0.0001 \qquad (۵)$$

$$\mathrm{Dis}(fi,fi') = \begin{Bmatrix} (fi-fi')^2 & \text{if fi and fi' are numerical or ordinal} \\ 0 & \text{if fi and fi' are nominal and } fi = fi' \\ 1 & \text{if fi and fi' are nominal and } fi \neq fi' \end{Bmatrix}$$

که در آن P' و P نشان دهنده پروژه‌ها، W_i وزن تعیین شده برای هر ویژگی، f_i' و f_i نشان دهنده ویژگی هر پروژه و n تعداد ویژگی می‌باشند. از δ نیز برای بدست آوردن نتایج غیر صفر استفاده می‌شود.

فرمول فاصله منهتن بسیار مشابه فرمول فاصله اقلیدسی است و برای محاسبه اختلاف مطلق ویژگی استفاده می‌شود. محاسبه فاصله منهتن به صورت رابطه (۶) است:

$$\mathrm{Sim}(p,p') = \frac{1}{\left[\sum_{i=1}^{n} \mathrm{WiDis}(fi,fi')+\delta\right]} \quad \delta = 0.0001 \qquad (۶)$$

$$\mathrm{Dis}(fi,fi') = \begin{Bmatrix} |fi-fi'| & \text{if fi and fi' are numerical or ordinal} \\ 0 & \text{if fi and fi' are nominal and } fi = fi' \\ 1 & \text{if fi and fi' are nominal and } fi \neq fi' \end{Bmatrix}$$

۳-۲- تابع راه حل[۵]

تابع راه حل به منظور برآورد تلاش توسعه نرم افزار با در نظر گرفتن پروژه‌های مشابه که به وسیله تابع شباهت یافت شده‌اند استفاده می‌شود. توابع راه حل معمول شامل مشابه ترین پروژه، میانگین مشابه ترین پروژه[۶]، میانه مشابه ترین پروژه[۷] و میانگین وزن دار فاصله معکوس[۸] می‌باشند. میانگین وزن دار فاصله معکوس، مقدار متوسط تلاش برای K پروژه مشابه که در آن $K>2$ می باشد را به صورت رابطه (۷) محاسبه می‌کند:

$$C_p = \sum_{k=1}^{K} \frac{\mathrm{Sim}(P,P_k)}{\sum_{i=1}^{k} \mathrm{Sim}(P,P_i)} \; C_{P_k} \qquad (۷)$$

به طوری که P پروژه جدید، P_k، k امین پروژه مشابه، C_{Pk} تلاش واقعی پروژه مشابه P_k، $\mathrm{sim}(p, p_k)$ شباهت بین پروژه P و P_k می باشد.

---

[2] Similarity Function
[3] Euclidean Distance
[4] Manhattan Distance
[5] Solution Function
[6] Average of Most Similar Projects
[7] Median of Most Similar Projects
[8] Inverse distance Weighted Mean

۳-۳- k نزدیکترین همسایه[9]

انتخاب بهترین مقدار K در دهه‌های گذشته موضوع مطالعات متعددی بوده است. زیرا مقدار K بر سطح دقت برآوردها تاثیر می‌گذارد و می‌تواند در مجموعه داده‌های مختلف متفاوت باشد. در بسیاری از مطالعات محققان مقدار ثابتی برای K در نظر گرفته اند. از سوی دیگر تعدادی از مطالعات محدوده ای برای K تعیین کرده اند و سپس برای پیدا کردن بهترین مقدار K تلاش کردند. اخیراً برخی از محققان از یک روش جستجو پویا برای پیدا کردن ارزش بهینه k استفاده می‌کنند. بنابراین، این ارزش ممکن است برای پروژه‌های مختلف متفاوت باشد [۹].

۴- مدل پیشنهادی

مدل پیشنهادی ارائه شده در این مقاله بر مبنای بهینه سازی ABE توسط الگوریتم PRO است. این مدل از دو مرحله اساسی آموزش و آزمایش تشکیل شده است که در ادامه شرح داده شده اند. برای ارزیابی عملکرد مدل پیشنهادی از ۴ معیار ارزیابی، خطای نسبی[10]، بزرگی خطای نسبی[11]، میانگین بزرگی خطای نسبی[12] و درصد پیش بینی[13] استفاده شده است. روابط ریاضی این ۴ معیار در ادامه به ترتیب آورده شده است.

$$RE = \frac{(estimate - actual)}{actual} \quad (۸)$$

$$MRE = \frac{|estimate - actual|}{actual} \quad (۹)$$

$$MMRE = \frac{\sum_{i=1}^{N} MRE}{N} \quad (۱۰)$$

$$PRED(X) = \frac{A}{N} \quad (۱۱)$$

دررابطه (۱۱)، A تعداد پروژه‌ها با MRE کمتر یا مساوی X و N تعداد پروژه های در نظر گرفته شده است [۱۰]. معمولا مقدار قابل قبول برای X در روش های برآورد هزینه نرم افزار ۰,۲۵ است. estimate نشان دهنده هزینه پیش بینی شده و actual نشان دهنده مقدار واقعی هزینه است.

۴-۱- مرحله آموزش

در ابتدای این مرحله تمام پروژه‌های نرم افزاری به سه دسته مبنا، آموزش و آزمایش تقسیم می‌شوند. پروژه‌های گروه مبنا و آموزش برای ساخت مدل پیش بینی و پروژه‌های گروه آزمایش برای بررسی عملکرد استفاده شده است. در واقع پروژه‌های آموزش با پروژه‌های مبنا برای پیدا کردن مطلوب ترین وزن‌ها، و پروژه‌های آزمایش با پروژه‌های مبنا برای ارزیابی عملکرد مدل مقایسه شده‌اند. یک پروژه از دسته آموزش انتخاب و به عنوان یک پروژه جدید به تابع شباهت اعمال می‌شود. الگوریتم PRO به ویژگی‌های مستقل در تابع شباهت وزن‌ها را اختصاص می‌دهد. پروژه برداشته شده با پروژه‌های مبنا توسط رابطه ۱ مقایسه شده است. تابع شباهت شبیه‌ترین پروژه‌ها را (از پروژه‌های پایه) برای پروژه برداشته شده جدا کرده و آنها را به تابع راه حل ارسال می‌کند. در نهایت هزینه در تابع راه حل برآورد شده و معیار عملکرد MRE محاسبه شده است. این فرآیند تا برآورد همه‌ی پروژه‌های آموزش ادامه دارد. در مرحله بعد MMRE و PRED(0.25) برای گروه آموزش براساس مقدار به دست آمده از MRE محاسبه و این مقدار به الگوریتم PRO ارسال شده است. الگوریتم PRO مقدار (MMRE-PRED(0.25)) را به عنوان تابع هدف مینیمم می‌کند. در صورت برقراری شرط خاتمه الگوریتم بهینه ساز (پایان تکرارها) مقدار بهینه وزن‌ها برای مرحله آزمایش ذخیره می‌شود.

۴-۲- مرحله آزمایش

هدف اصلی مرحله آزمایش ارزیابی دقت مدل پیش بینی پیشنهادی با اعمال پروژه‌های جدید است. در این مرحله پروژه‌های مبنا و آزمایش به عنوان ورودی تابع شباهت به کار گرفته شده‌اند. علاوه براین وزن‌های بهینه به دست آمده از مرحله آموزش نیز به این تابع اعمال شده است. مانند مرحله آموزش یک پروژه از پروژه‌های آزمایش برداشته و با پروژه‌های مبنا توسط تابع شباهت مقایسه شده است. پروژه‌های بسیار شبیه به پروژه برداشته شده تعیین و به تابع راه حل ارسال می‌شوند. هزینه پیش بینی و مقدار MRE نیز محاسبه شده است. این عمل برای همه‌ی پروژه‌های آزمایش تکرار، و معیارهای MMRE و PRED(0.25) برای مرحله آزمایش محاسبه شده است.

---

[9] K Nearest Neighborhood (KNN)
[10] Relative Error (RE)
[11] Magnitude Of Relative Error (MRE)
[12] Mean Magnitude Of Relative Error (MMRE)
[13] Percentage Of Prediction (PRED)

۵- نتایج شبیه سازی

نتایج بدست آمده از مدل پیشنهادی با دوسته از نتایج مورد مقایسه قرار گرفته اند. دسته اول نتایج مربوط به مطالعات گذشته می شود و دسته دوم نتایج خاصل از جایگزینی الگوریتم های بهینه سازی دیگر به جای الگوریتم PRO در مدل پیشنهادی است. درواقع با جایگرین کردن چند الگوریتم بهینه سازی پرکاربرد به جای الگوریتم PRO میزان کارایی الگوریتم PRO هرچه بهتر سنجیده می شود.

۵-۱- نتایج بدست آمده روی مجموعه داده ماکسول

مجموعه داده ماکسول مربوط به یکی از بزرگترین بانک های تجاری فنلاند است و شامل ۶۲ پروژه و ۲۶ ویژگی می باشد. نتایج بدست آمده از مدل پیشنهادی روی مجموعه داده ماکسول با روش های [۱۱] classification، [۱۲] OABE، MLR و SWR و همچنین با الگوریتم های [۱۳] PSO، [۱۴] TLBO و [۱۵] FA مورد مقایسه قرار گرفته است. در جدول (۱) نتایج شبیه سازی شده مدل پیشنهادی با سایر مدل ها مورد مقایسه قرار گرفته است.

جدول (۱): نتایج بدست آمده از مدل ها روی مجموعه داده ماکسول

Method	MMRE	PRED(0.25)
Classification	۰/۶۰	۰/۳۷
MLR	۱/۲۶	۰/۲۰
OABE	۰/۴۱	۰/۳۴
SWR	۰/۹۳	۰/۲۵
PRO-ABE	**۰/۳۸**	۰/۴۳
PSO-ABE	۰/۴۴	۰/۳۹
TLBO-ABE	۰/۴۱	۰/۳۷
FA-ABE	۰/۵۲	۰/۲۹

همانطور که مشخص است مدل پیشنهادی با الگوریتم PRO بهترین مقادیر را نسبت به مدل های دیگر روی هر دو معیار ارزیابی بدست آورده است. برای سنجش دقیق تر عملکرد مدل پیشنهادی نتایج بدست آمده به لحاظ آماری مورد سنجش قرار گرفته اند. در جدول (۲) نتایج بدست آمده از Wilcoxon test روی نتایج MRE نشان داده شده است.

جدول (۲): مقادیر بدست آمده آزمون آماری روی مجموعه داده ماکسول

Classification	MLR	OABE	SWR	PSO-ABE	TLBO-ABE	FA-ABE
۰/۰۱۹	۱/۵۹E-۴	۱/۷۴E-۴	۰/۰۲۵	۰/۰۴۱	۹/۲۰E-۴	۹/۲۰E-۴

همانطور که در جدول (۲) نشان داده شده است تمامی مقادیر بدست آمده از آزمون آماری کمتر از ۰،۰۵ هستند و نشان دهنده این است که روش پیشنهادی از نظر آماری نیز نسبت به سایر روش ها دارای برتری است.

۵-۲- نتایج بدست آمده روی مجموعه داده دشارنیز

مجموعه داده دشارنیز شامل ۸۱ پروژه و ۱۱ ویژگی می باشد. از آنجایی که ۴ پروژه این مجموعه داده شامل ویژگی های از دست رفته می باشند تنها ۷۷ پروژه این مجموعه داده برای ارزیابی مدل ها قابل استفاده می باشند. نتایج بدست آمده با سایر مدل ها روی مجموعه داده دشارنیز در جدول (۳) نشان داده شده است.

جدول (۳): نتایج بدست آمده از مدل ها روی مجموعه داده دشارنیز

Method	MMRE	PRED(0.25)
MLR	۰/۹۷	۰/۲۳
OABE	۰/۳۴	۰/۴۸
SWR	۰/۷۳	۰/۳۰
PRO-ABE	**۰/۳۱**	۰/۵۴
PSO-ABE	۰/۳۴	۰/۴۷
TLBO-ABE	۰/۴۱	۰/۴۸
FA-ABE	۰/۴۸	۰/۳۹

همانطور که از نتایج جدول (۳) مشخص است مدل پیشنهادی با الگوریتم PRO روی مجموعه داده دشارنیز هم بهترین نتایج را نسبت به سایر مدل ها بدست آورده است. روی معیار MMRE مدل پیشنهادی و PSO-ABE بسیار نتایج رقابتی و نزدیک بهم بدست آورده اند. روی معیار PRED(0.25) روش های OABE و TLBO-ABE نتایج شبیه بهم بدست آورده اند. همینطور در جدول (۴) نتایج آزمون آماری مدل های مختلف نسبت به مدل های پیشنهادی نشان داده شده است.

جدول (۴): مقادیر بدست آمده آزمون آماری روی مجموعه داده دشارنیز

MLR	OABE	SWR	PSO-ABE	TLBO-ABE	FA-ABE
۲/۰۲E-۴	۰/۰۴۸۹	۰/۰۰۴	۰/۰۴۷	۰/۰۶۱	۰/۰۹۴

همانطور که نتایج بدست آمده نشان می دهند تمامی مقادیر بدست آمده از آزمون آماری کمتر از ۰,۰۵ می باشند که با این نتایج برتری آماری مدل پیشنهادی تایید می شود.

۶- نتیجه و جمع بندی

پیش بینی هزینه نرم افزار برای برنامه ریزی صحیح و همچنین ارائه محصول به موقع به مشتری، موضوع مهمی برای شرکت توسعه نرم افزار است. اگرچه ABE مدل مناسبی برای تخمین مبتنی بر مقایسه است و در ارزیابی هزینه توسعه نرم افزار بسیار مورد استفاده قرار گرفته است، اما هنوز در بسیاری از مواقع قادر به تولید تخمین دقیق نیست، این کاستی با الگوریتم های بهینه سازی قابل جبران است. در این مقاله از الگوریتم PRO به عنوان یک الگوریتم جدید و مناسب برای وزن دهی دقیق می باشد استفاده شده است. مدل پیشنهادی ارائه شده در این مقاله دارای دو مرحله اصلی است. در مرحله اول بهترین وزن ها برای ویژگی های پروژه های نرم افزاری توسط الگوریتم PRO پیشنهاد می شوند و در مرحله دوم از این وزن ها برای تخمین دقیق تر هزینه نرم افزار استفاده شده است. در این کار از دو مجموعه داده واقعی ماکسول و دشارنیز استفاده شده است. نتایج بدست آمده از مدل پیشنهادی در مقایسه با سایر مدل ها نشان از برتری این مدل را دارد. به عنوان کار آینده از سایر الگوریتم های بهینه سازی و همین طور استفاده از روش های انتخاب ویژگی برای حذف ویژگی های کم اثر استفاده خواهد شد.

مراجع

[1]Kirsopp, C., SHepped, M., Hart, J., (2002), Search heuristics case-based reasoning and software project effort prediction in: genetic and evolutionary computing, Proceedings of the Genetic and Evolutionary Computation Conference Newyork, 1367-1374.

[2]Huang, S.J., Chiu, N.H., (2006), Optimization of analogy weights by genetic algorithm for software effort estimation, Information and Software Technology, 48(11):1034 -1045.

[3]Kad, S., Chopra, V., (2012), Software development effort estimation using soft computing, International Journal of Machine Learning and Computing, 2(5):437-439.

[4] Rao, T., Mall, R., (2018), DABE: Differential evolution in analogy-based software development effort estimation, Swarm and Evolutionary Computation, 38:158-172.

[5] Phannachitta, P., (2020), On an optimal analogy-based software effort estimation, Information and Software Technology, 125.

[6] Ezghari, S, Zahi, A, (2018), Uncertainty management in software effort estimation using a consistent fuzzy analogy-based method, Applied Soft Computing, 67:540-557.

[7] Samareh moosavi, S H, Khatibi bardsiri, V, (2019), Poor and rich optimization algorithm: A new human-based and multi populations algorithm, Engineering Applications of Artificial Intelligence, 86:165-181.

[8] Shepperd, M. Schofield. C. (1997).Estimating Software Project Effort Using Analogies. IEEE Transaction on software engineering, 23(11):736-743.

[9] Khatibi, V. Jawawi.A.N. Mohd Hashim .D. Khatibi. S.Z. (2012), A pso-based model to increase the accuracy of software development effort estimation, Software Qual J, 21:501-526.

[10] Basha, S.P.D. (2010), Analysis of empirical software effort estimation models, International Journal of Computer Science and Information Security (IJCSIS), 7(3):68-78.

[11] V. Khatibi, (2013). A flexible method to estimate the software development effort based on the classification of projects and localization of comparisons, Empir Software Eng, vol. 19(4): 857-884.

[12] M. Azzeh and Y. Elsheikh and M. Alseid. (2014). An Optimized Analogy-Based Project Effort Estimation. International Journal of Advanced Computer Science and Applications, Vol. 5(4):6-11.

[13] Kennedy J, Eberhart R . (1995). Particle swarm optimization. In: Proceedings of the IEEE international conference on neural networks.1942–8.

[14] Yang X-S . Firefly algorithm. (2010). stochastic test functions and design optimization. Int J Bio-Inspired Comput, 2:78–84.

[15] Rao RV., Savsani VJ., Vakharia DP. (2011), Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems, Computer-Aided Design, 43:303-315.