یادگیری انتقالی با روش تلفیقی از انتقال نمونه و نمایش ویژگی برای پیش‌بینی نقص بین‌پروژه‌ای نرم‌افزار

سعاد شریفات‌زاده[۱]، دانشجوی کارشناسی ارشد؛ محمدعلی زارع چاهوکی[۲]، استادیار

۱- گروه مهندسی کامپیوتر - پردیس فنی و مهندسی - دانشگاه یزد - یزد - ایران — sharifatzadeh@stu.yazd.ac.ir
۲- گروه مهندسی کامپیوتر - پردیس فنی و مهندسی - دانشگاه یزد - یزد - ایران — chahooki@yazd.ac.ir

چکیده: پیش‌بینی نقص نرم‌افزار، نقش مهمی در بهبود کیفیت نرم‌افزار دارد. به‌طوری‌که منابع محدود آزمون نرم‌افزار، به‌جای کل ماژول‌های نرم‌افزار به ماژول‌های مستعد نقص اختصاص داده می‌شوند. در پیش‌بینی نقص درون‌پروژه‌ای، برای ساخت مدل پیش‌بینی، داده‌های برچسب‌دار محلی استفاده می‌شود. ولی ساخت این مدل در مورد پروژه‌هایی که فاقد داده‌های برچسب‌دار محلی هستند، تقریباً غیرممکن است. لذا، پیش‌بینی نقص بین‌پروژه‌ای مطرح می‌شود، که برای ساخت و آموزش مدل، از داده‌های سایر پروژه‌ها استفاده می‌کند. در این حوزه، توزیع داده‌ای بخش‌های آموزش و آزمون متفاوت است. ازاین‌رو، پژوهش‌های انجام‌شده روی کاهش اثر منفی تفاوت توزیع بخش‌های آموزش و آزمون تمرکز دارند. در این پژوهش، روش بازه تخمین دانش پیشنهاد شده است. در این روش نمونه‌هایی از بخش آموزش که از نظر توزیع داده‌ای، مشابه نمونه‌های بخش آزمون هستند، انتخاب می‌شود. سپس، نمونه‌های منتخب به مدل آموزشی داده می‌شود. برای افزایش اثربخشی، قبل از اعمال روش بازه تخمین دانش، تکنیک استخراج ویژگی روی بخش‌های آموزش و آزمون اعمال می‌شود. نتایج حاصل از ارزیابی روش پیشنهادی روی ۱۰ مجموعه داده از دادگان ناسا و SoftLab با معیار AUC بیانگر اثربخشی این روش در پیش‌بینی ماژول‌های مستعد نقص است. روش پیشنهادی به‌طور میانگین ۳۸/۱ درصد نسبت به پیش‌بینی نقص درون‌پروژه‌ای افزایش دقت دارد.

واژه‌های کلیدی: پیش‌بینی نقص نرم‌افزار، پیش‌بینی نقص بین‌پروژه‌ای، یادگیری ماشین، یادگیری انتقالی.

Compilation Instance Transfer and Feature-representation Transfer for Cross Project Defect Prediction

S. Sharifatzadeh[1], MSc Student; M. A. Z. Chahooki[2], Assistant Professor

1- Engineering Faculty, Computer Engineering Group, Yazd University, Yazd, Iran, Email: sharifatzadeh@stu.yazd.ac.ir
2- Engineering Faculty, Computer Engineering Group, Yazd University, Yazd, Iran, Email: chahooki@yazd.ac.ir

Abstract: Software defect prediction is critical for software quality improvement. So that, limited resources for software testing is allocated only to fault-prone instead of all software modules. In project defect prediction, to build a prediction model, usually local labeled dataset are used. But, building the predicting model for projects without local labeled data is almost impossible. Thus, cross project defect perdition is proposed for training the prediction model with data from other projects. In cross project defect perdition, training data and test data distribution are different. Therefore, researches in this area have focused for reduction the negative impact of different distribution between training and test data. In this research, the Knowledge Estimation Interval (KEI) method is proposed. In this method, instances of training data by similar distribution with test set are selected. Then, selected instances are given as training to the prediction model. To increase the effectiveness of the proposed approach, feature extraction techniques are applied on training and test set before KEI. The evaluation results of the proposed approach on 10 datasets from NASA and SoftLab with AUC indicate the effectiveness of this approach to predict the fault-prone modules. The proposed method has increased an average value of 38.1% in the accuracy compared to within project defect prediction models.

Keywords: Software defect prediction, cross project defect prediction, machine learning, transfer learning.

تاریخ ارسال مقاله: ۱۳۹۵/۰۸/۲۵
تاریخ اصلاح مقاله: ۱۳۹۵/۱۱/۱۸ و ۱۳۹۵/۱۲/۱۴
تاریخ پذیرش مقاله: ۱۳۹۵/۱۲/۲۰
نام نویسنده مسئول: محمدعلی زارع چاهوکی
نشانی نویسنده مسئول: ایران – یزد – صفاییه – بلوار دانشگاه – دانشگاه یزد – گروه مهندسی کامپیوتر.

۱- مقدمه

کیفیت، یکی از مسائل اساسی در فرآیند تولید و توسعه نرم‌افزار است. آزمون کدهای برنامه، یکی از فعالیت‌هایی است که برای تضمین کیفیت نرم‌افزار انجام می‌شود. باید توجه داشت که همواره محدودیت زمانی و هزینه‌ای در منابع انسانی برای آزمون کدهای برنامه وجود دارد. همچنین، از دیدگاهی دیگر بسیاری از خطاهای سیستم‌های نرم‌افزاری از تعداد اندکی از اجزای سیستمی ناشی می‌شود. قانون ۸۰:۲۰ بیان می‌کند، ۲۰ درصد از ماژول‌های نرم‌افزاری سبب ایجاد ۸۰ درصد از خطاها می‌شوند [۱]. بنابراین، برای رسیدن به کیفیت در نرم‌افزار با هزینه و زمان کم‌تر، بهتر است ابتدا ماژول‌های مستعد نقص[۱] شناسایی و سپس منابع محدود موجود، روی آزمون ماژول‌های شناسایی شده، متمرکز شوند. ازاین‌رو، مبحث پیش‌بینی ماژول‌های مستعد نقص در حوزه کاربرد یادگیری ماشین در مهندسی نرم‌افزار از سال ۱۹۹۰ مطرح شده است [۲]. این مبحث به اصطلاح "پیش‌بینی نقص نرم‌افزار[۲]" نام‌برده می‌شود. به بیان دیگر، پیش‌بینی نقص نرم‌افزار، مدیران پروژه‌های نرم‌افزاری را قادر می‌سازد که منابع محدود، جهت آزمون را روی قسمت‌های مورد نیاز متمرکز کنند.

پیش‌بینی نقص نرم‌افزار، روی طبقه‌بندی مؤلفه‌های نرم‌افزار به دو کلاس ماژول‌های مستعد نقص و ماژول‌های غیرمستعد نقص، تمرکز دارد [۳]. برای طبقه‌بندی مؤلفه‌های نرم‌افزار از روش‌های یادگیری ماشین و داده کاوی استفاده می‌شود [۴]. به این صورت که با توجه به سابقه ماژول ناقص، مدلی ایجاد می‌شود که بر اساس آن بتوان با دقتی مناسب پیش‌بینی نقص در ماژول جدید را انجام داد. در این روش از ویژگی‌های نرم‌افزاری نسخه‌های قبلی پروژه نرم‌افزاری استفاده شده و ماژول‌های مستعد نقص نسخه بعدی پروژه پیش‌بینی می‌شود. مدل‌های پیش‌بینی نقص، بر اساس ویژگی‌های کدهای نرم‌افزاری و وجود یا عدم وجود نقص در آن‌ها، ساخته می‌شود. ویژگی‌هایی که از کدهای نرم‌افزار استخراج شده است به‌عنوان متغیر مستقل و اطلاعات نقص بـه‌عنـوان متغیر وابسته در مدل‌های پیش‌بینی استفاده می‌شوند.

در یادگیری ماشین بانظارت، داده‌های برچسب‌دار نقش اصلی را ایفا می‌کنند. برچسب‌زدن به ماژول‌ها و فایل‌های دارای نقص در پروژه‌های نرم‌افزاری کار دشواری است و نیازمند سازمان‌دهی منظمی در مدیریت پیکربندی و کنترل تغییرات پروژه می‌باشد. همچنین، اطلاعات هر پروژه نرم‌افزاری در شرکت‌ها به‌صورت محلی جمع‌آوری می‌شود و برای بهبود کیفیت خود پروژه مورد استفاده قرار می‌گیرد. ازاین‌رو، ساخت مدل پیش‌بینی نقص، در مورد پروژه‌هایی با چرخه حیات کوتاه و پروژه‌هایی که فاقد داده‌های برچسب‌دار محلی هستند، دشوار می‌شود [۵]. بنابراین، روش‌های پیش‌بینی نقص نرم‌افزار را از این منظر می‌توان به دو دسته تقسیم کرد:

الف. روش‌های پیش‌بینی نقص درون‌پروژه‌ای

ب. روش‌های پیش‌بینی نقص بین‌پروژه‌ای

روش‌های پیش‌بینی نقص درون‌پروژه‌ای (WPDP[۳])، برای آموزش مدل پیش‌بینی از نمونه‌های برچسب‌دار محلی استفاده می‌کنند [۶]. از سوی دیگر در روش‌های پیش‌بینی نقص بین‌پروژه‌ای (CPDP[۴])، برای آموزش مدل از نمونه‌های برچسب‌دار سایر پروژه‌های نرم‌افزاری استفاده می‌شود [۵]. بسیاری از پژوهش‌های انجام‌شده، در دسته پیش‌بینی نقص درون‌پروژه‌ای قرار می‌گیرد. در این دسته، بخشی از نمونه‌های پروژه برای آموزش و مابقی نمونه‌های پروژه برای ارزیابی مدل استفاده می‌شود [۷]. بنابراین توزیع داده‌ای[۵] بخش‌های آموزش و آزمون یکسان است. ولی در دسته پیش‌بینی نقص بین‌پروژه‌ای، بخش‌های آموزش و آزمون از پروژه‌های متفاوت هستند. در این دسته، بخش‌های آموزش و آزمون به ترتیب بخش‌های منبع[۶] و هدف[۷] نامیده می‌شوند [۸]. یکی از راه‌حل‌ها، برای غلبه بر توزیع داده‌ای متفاوت بین بخش‌های منبع و هدف، تکنیک‌های یادگیری انتقالی می‌باشد.

هدف یادگیری انتقالی، استخراج دانش از یک یا چند کار منبع و اعمال این دانش به کار هدف است (منظور از کار، عملیات طبقه‌بندی است) [۸]. یادگیری انتقالی، توانایی سیستم برای تشخیص و اعمال دانش و مهارت‌های فراگرفته‌شده در کارهای قبلی به کارهای جدید است. مطابق با [۸] یادگیری انتقالی به‌صورت زیر تعریف می‌شود:

اگر دامنه منبع D_S، کار منبع T_S، دامنه هدف D_T و کار هدف T_T باشد، هدف یادگیری انتقالی بهبود یادگیری تابع پیش‌بینی هدف در D_T با استفاده از D_S و T_S است، که $D_S \neq D_T$ یا $T_S \neq T_T$ باشند.

CPDP، مطابق با یادگیری انتقالی با روش‌های ورارسایی[۸] می‌باشد، که در این یادگیری انتقالی، کارهای منبع و هدف مشابه هستند درحالی‌که دامنه‌های هدف و منبع متفاوت است (یعنی $D_S \neq D_T$ و $T_T = T_S$). فرضیه مهم در یادگیری انتقالی، نوع دانش انتقالی است. در یادگیری انتقالی ورارسایی انتقال دانش به دو صورت انتقال نمونه[۹] و انتقال نمایش ویژگی[۱۰] انجام می‌شود. در انتقال نمونه، هدف انتخاب یا وزن‌دهی نمونه‌های بخش منبع است، به‌طوری‌که توزیع داده‌ای مشابهی با بخش هدف داشته باشند. در انتقال نمایش ویژگی، هدف پیدا کردن بهترین فضای ویژگی بخش‌های منبع و هدف است، به‌طوری‌که تفاوت توزیع داده‌ای این بخش‌ها کم باشد.

در روش پیشنهادشده در این پژوهش، انتقال دانش به‌صورت انتقال نمونه است. در این روش نمونه‌های بخش منبع وزن‌دهی می‌شوند و نمونه‌هایی با بیش‌ترین وزن انتخاب می‌شوند. وزن‌دهی این نمونه‌ها مبتنی بر اطلاعات بخش هدف است. روش پیشنهادی، بازه تخمین دانش (KEI[۱۱]) نام نهاده شده است. نوآوری این روش، در استخراج اطلاعات بخش هدف است. برای افزایش اثربخشی روش KEI، قبل از اعمال این روش، تکنیک‌های استخراج ویژگی بی‌نظارت روی بخش‌های هدف و منبع اعمال می‌شود، که نوآوری جنبی در روش پیشنهادی می‌باشد. درواقع از استخراج ویژگی، برای انتقال نمایش ویژگی استفاده شده است.

در ادامه این پژوهش، در بخش ۲ به مرور پژوهش‌های انجام‌شده در حوزه پیش‌بینی نقص بین‌پروژه‌ای پرداخته شده است. در بخش ۳ روش

پیشنهادی بیان شده است. در بخش ۴ نتایج تجربی آورده شده است. در پایان، در بخش ۵ نتیجه‌گیری و پژوهش‌های پیش‌رو آورده شده است.

۲- پیشینه تحقیق

در این بخش، پژوهش‌های انجام‌شده در حوزه CPDP و تکنیک‌های یادگیری انتقالی انجام شده، مرور شده است. در پژوهش‌های [۵، ۷، ۹، ۱۰، ۱۱] انتقال دانش، به‌صورت انتقال نمونه و در پژوهش [۱۲] به‌صورت انتقال نمایش ویژگی است.

در پژوهش [۷]، در ابتدا تمام نمونه‌های برچسب‌دار بخش منبع به مدل آموزشی داده شده است. نتایج به‌دست‌آمده، در مقایسه با نتایج WPDP روی این دادگان، قابل قبول نبود. ازاین‌رو، برای بهبود عملکرد CPDP، روش نزدیک‌ترین همسایگی برای انتخاب نمونه‌های بخش منبع مورد استفاده قرار گرفته است. در این روش، هر نمونه بخش هدف، k نزدیک‌ترین نمونه، از بخش منبع را انتخاب می‌کند. این روش، فیلتر NN نام نهاده شده است. ولی با اعمال فیلتر NN نیز، نتایج CPDP از WPDP روی برخی از پروژه‌ها برابر و برخی دیگر ضعیف‌تر بود.

در پژوهش [۵]، از فیلتر پترس برای انتقال نمونه‌های بخش منبع استفاده شده است. فیلتر پترس فرض می‌کند داده‌های بخش منبع شامل اطلاعات مهمی در مورد نقص‌ها نسبت به داده‌های کوچک بخش هدف است. بنابراین، به‌جای اینکه هر نمونه بخش هدف، نمونه‌های نزدیک‌تر از بخش منبع را انتخاب کند. هر نمونه بخش منبع، k نزدیک‌ترین نمونه از بخش هدف به خود را انتخاب می‌کند. در این صورت، هر نمونه بخش هدف توسط چندین نمونه بخش منبع انتخاب شده است. در پایان، هر نمونه بخش هدف، نزدیک‌ترین نمونه‌های کاندیدشده از بخش منبع را انتخاب می‌کند و بقیه را رد می‌کند. فیلتر پترس در ابتدا و قبل از انجام همه این کارها، ترکیب نمونه‌های بخش‌های منبع و هدف را از طریق روش k-means خوشه‌بندی می‌کند. خوشه‌هایی که حداقل یک نمونه از داده‌های هدف را دارا باشند نگه‌می‌دارد و از بقیه خوشه‌ها صرف‌نظر می‌کند. در این پژوهش از طبقه‌بندهای نایوبیز، جنگل تصادفی و رگرسیون منطقی استفاده شده است. نتایج پژوهش، برتری CPDP را نسبت به WPDP نشان می‌دهد.

در پژوهش [۹]، الگوریتم نایوبیز انتقالی (TNB[۱۲]) پیشنهاد شده است. در این الگوریتم ابتدا، اطلاعات هر ویژگی از بخش هدف جمع‌آوری می‌شود. سپس، درجه شباهت هر نمونه از بخش منبع، از طریق مقایسه اطلاعات جمع‌آوری‌شده در مرحله اول، محاسبه می‌شود. مبتنی بر شباهت به‌دست‌آمده، به نمونه‌های بخش منبع، وزنی داده می‌شود. نایوبیز وزن‌دار، روی نمونه‌های وزن‌دار اعمال می‌شود. ایده اصلی از TNB، وزن‌دار کردن نمونه‌های بخش منبع مطابق با شباهت بین نمونه‌های بخش‌های منبع و هدف است. شباهت بین نمونه‌های بخش‌های منبع و هدف در سطح ویژگی انجام می‌شود. در این الگوریتم به‌جای جدا کردن و انتخاب بعضی از نمونه‌های بخش منبع، از همه نمونه‌های بخش منبع در مرحله آموزش، استفاده می‌شود. همچنین، وزن‌دهی این نمونه‌ها مبتنی بر توزیع بخش هدف است. ولی، این توزیع با محاسبه بیش‌ترین و کم‌ترین مقدار هر ویژگی تخمین زده می‌شود. درصورتی‌که این اطلاعات بازتاب دقیقی را از توزیع داده‌ای نمی‌دهند. روش TNB، ۱۰/۸۱ درصد افزایش دقت نسبت به روش فیلتر NN دارد.

در پژوهش [۱۰]، الگوریتم VCB-SVM پیشنهاد شده است. در این الگوریتم، نمونه‌های بخش منبع بر اساس توزیع داده‌ای تخمین‌زده‌شده از بخش هدف، وزن‌دهی می‌شود. درواقع، نمونه‌های بخش منبع در این پژوهش، همانند روش TNB، وزن‌دهی می‌شود. نمونه‌هایی که تشابه بیش‌تری با بخش هدف دارند، به الگوریتم بوستینگ داده می‌شود. آن نمونه‌هایی که در الگوریتم بوستینگ اشتباه تشخیص داده شده‌اند، با تکنیک‌های رایج نمونه‌گیری متوازن می‌شوند. طبقه‌بند الگوریتم بوستینگ، SVM می‌باشد. با استفاده از الگوریتم بوستینگ هزینه‌ای برای طبقه‌بندی نادرست، در نظر گرفته شده است. علاوه‌براین، مشکل عدم‌توازن دادگان، با تکنیک‌های نمونه‌گیری، حل شده است. ولی اطلاعات جمع‌آوری‌شده از بخش هدف، بازتاب دقیقی را از توزیع داده‌ای نمی‌دهند. VCB-SVM، ۶/۱۳ درصد افزایش دقت نسبت به TNB دارد.

در پژوهش [۱۱]، الگوریتم DTB پیشنهاد شده است. در این الگوریتم، دو سطح از، تکنیک‌های یادگیری انتقالی انجام می‌شوند. در سطح اول، با روش فیلتر NN، مناسب‌ترین نمونه‌های بخش منبع انتخاب می‌شوند. سپس، این نمونه‌ها با روش نمونه‌گیری SMOTE متوازن می‌شوند. در سطح دوم، نمونه‌های متوازن‌شده با توجه به مشابه بودن توزیع داده‌ای بخش‌های منبع و هدف، وزن‌دهی می‌شوند. درواقع، نمونه‌های بخش منبع در این پژوهش نیز، همانند روش TNB وزن‌دهی می‌شود. در آخر نمونه‌های وزن‌دهی‌شده بخش منبع و نمونه‌های برچسب‌دار بخش هدف به الگوریتم بوستینگ داده می‌شود. طبقه‌بند الگوریتم بوستینگ، نایوبیز است. داشتن دو سطح یادگیری انتقالی، حل مشکل عدم‌توازن دادگان با تکنیک SMOTE و استفاده از نمونه‌های برچسب‌دار بخش هدف در آموزش، به بهبود عملکرد DTB، کمک کرده است. ولی اطلاعات جمع‌آوری‌شده از بخش هدف، بازتاب دقیقی را از توزیع داده‌ای نمی‌دهند. DTB، ۳/۹ درصد افزایش دقت نسبت به TNB دارد.

در پژوهش [۱۲]، آنالیز مؤلفه انتقالی (TCA) روی پیش‌بینی نقص نرم‌افزار اعمال شده است. همچنین، رویکرد جدید TCA^+ پیشنهاد شده است. TCA، تکنیک استخراج ویژگی جدیدی است که هدف آن یافتن فضای ویژگی نهان برای بخش‌های منبع و هدف توسط کمینه کردن فاصله بین توزیع داده‌ها (درحالی‌که خصوصیت‌های داده‌های اصلی حفظ شود) است. درواقع، TCA برای کاهش تأثیر منفی تفاوت توزیع داده‌ای بخش‌های منبع و هدف، استفاده شده است. روش TCA^+ قبل از استخراج ویژگی، نرمال‌سازی مناسبی برای دادگان بخش‌های منبع و هدف انتخاب می‌کند. در این پژوهش از طبقه‌بند رگرسیون منطقی استفاده شده است. در TCA، هیچ انتقال نمونه‌ای انجام نمی‌شود. نتایج TCAو TCA^+ تقریباً برابر با نتایج WPDP است.

۳- روش‌های پیشنهادی

چالش اصلی در CPDP، تفاوت توزیع داده‌ای بخش‌های منبع و هدف است. در روش پیشنهادی، از تکنیک یادگیری انتقالی ورارسایی برای حل این چالش استفاده شده است. روش پیشنهادی، KEI و بهبودیافته این روش، FE-KEI[۱۳] نام نهاده شده‌اند.

هدف روش KEI، انتخاب کردن نمونه‌هایی از بخش منبع است، که درجه شباهت زیادی با نمونه‌های بخش هدف دارند. اندازه‌گیری این شباهت، توسط اطلاعات جمع‌آوری‌شده از بخش هدف، انجام می‌شود. این اطلاعات، بازتابی از توزیع داده‌ای بخش هدف است، که در سطح ویژگی جمع‌آوری می‌شود. نوآوری روش KEI، در جمع‌آوری این اطلاعات است. در روش‌هایی همچون TNB [۹]، اطلاعات جمع‌آوری‌شده از بخش هدف، بیش‌ترین و کم‌ترین مقدار هر ویژگی می‌باشد. اما این اطلاعات، بازتاب دقیقی را، از توزیع داده‌ای بخش هدف نمی‌دهند. بنابراین، در روش KEI اطلاعات جمع‌آوری‌شده، انحراف معیار و میانگین هر ویژگی است. انحراف معیار و میانگین اطلاعات دقیق‌تری از توزیع داده‌ای نسبت به بیش‌ترین و کم‌ترین مقدار، با خود به همراه دارند.

ازآنجایی‌که اطلاعات، در سطح ویژگی جمع‌آوری می‌شوند. لازم است، تعداد ویژگی‌ها برابر و مشترک باشد. روش KEI در سه گام انجام می‌شود:

گام اول: مجموعه $T_i = \{a_{i1}, a_{i2}, ..., a_{id}\}_{i=1}^{m}$ از بخش هدف، داده می‌شود. a_{ij}، ویژگی jام از نمونه T_i است. مقدار انحراف معیار و میانگین هر ویژگی j محاسبه می‌شود (رابطه (۱)):

$$\sigma = std\{a_{1j}, a_{2j}, ..., a_{mj}\} \quad (۱)$$

$$\delta = mean\{a_{1j}, a_{2j}, ..., a_{mj}\}$$

که در رابطه (۱)، $\boldsymbol{j} = \{1, 2, ..., d\}$ است که d، تعداد ویژگی‌ها و m، تعداد نمونه‌های بخش هدف هستند. مقادیر انحراف معیار و میانگین به رابطه (۲) داده می‌شود. رابطه (۲) بازه‌ای است، که توزیع داده‌ای هر ویژگی j در بخش هدف را تخمین می‌زند.

$$\mu_j \pm \alpha\sigma_j \equiv \left[\mu_j - \alpha\sigma_j, \mu_j + \alpha\sigma_j\right] \quad (۲)$$

$$0.5 \le \alpha \le 1 \qquad \text{که}$$

در رابطه (۲)، α ضریب پراکندگی بازه را نشان می‌دهد.

گام دوم: از طریق رابطه (۳)، شباهت هر نمونه از بخش منبع با بخش هدف، محاسبه می‌شود. به هر نمونه وزنی داده خواهد شد. مجموعه $S_i = \{b_{i1}, b_{i2}, ..., b_{id}\}_{i=1}^{n}$ از بخش منبع است. b_{ij}، ویژگی jام از نمونه S_i و d، تعداد ویژگی‌ها است.

$$p_i = \sum_{j=1}^{k} \frac{h(b_{ij})}{d} \quad (۳)$$

$$h(b_{ij}) = \begin{cases} 1 & b_{ij} \in \left[\mu_j - \alpha\sigma_j, \mu_j + \alpha\sigma_j\right] \quad \text{اگر} \\ 0 & \text{در غیر اینصورت} \end{cases}$$

در رابطه (۳)، p_i وزن هر نمونه از بخش منبع است. از رابطه (۳)، $P = \{p_1, p_2, ..., p_n\}$ به‌دست می‌آید، که n تعداد نمونه‌های بخش منبع است. مقادیر مجموعه P، بین صفر و یک نرمال می‌شود.

گام سوم: در این گام نمونه‌هایی از بخش منبع که بیش‌ترین وزن را دارند، انتخاب می‌شوند. درواقع، از مجموعه P، p_i هایی که از مقدار عددی β بیش‌تر هستند، انتخاب می‌شوند. در p_i، i اندیس نمونه بخش منبع است.

$$index = \{i \mid p_i \ge \beta\} \quad (۴)$$

$$0.5 \le \beta \le 1$$

$$training_{set} = \{\forall S_i \mid i \in index\}$$

در رابطه (۴)، S_i، نمونه iام از بخش منبع است. در الگوریتم ۱، شبه‌کد روش KEI آمده است.

پارامتر β، نشان می‌دهد حداقل چه درصدی از ویژگی‌های هر نمونه منتخب از بخش منبع، در بازه‌های تولیدشده از ویژگی‌های بخش هدف، قرار دارند. برای مثال، زمانی که β=۰/۵ باشد. فقط نمونه‌هایی که حداقل نیمی از ویژگی‌های آن‌ها، در بازه‌های مذکور باشد، انتخاب می‌شوند. بنابراین، هرچه مقدار β بیش‌تر باشد انتخاب نمونه‌های بخش منبع، سخت‌گیرانه‌تر می‌شود.

در روش KEI، مشترک بودن ویژگی‌های بخش‌های هدف و منبع ضروری است. اما، این امکان وجود دارد که، ویژگی‌های غیراشتراکی (ویژگی‌هایی که در مدل استفاده نمی‌شوند) بخش‌های منبع و هدف در بهبود دقت پیش‌بینی تأثیرگذار باشند. بنابراین، برای افزایش اثربخشی روش KEI، قبل اعمال این روش، تکنیک‌های استخراج ویژگی بی‌نظارت، روی بخش‌های منبع و هدف اعمال می‌شود. این روش FE-KEI نام نهاده شده است.

در روش FE-KEI، تکنیک استخراج ویژگی، به‌طور جداگانه روی بخش‌های منبع و هدف اعمال می‌شود. ولی با شرط، یکسان بودن تعداد ویژگی‌های استخراج‌شده و مشابه بودن روش استخراج ویژگی، این کار انجام می‌شود. با این کار، به‌جای استفاده از ویژگی‌های اشتراکی بخش‌های منبع و هدف، از همه ویژگی‌های این بخش‌ها استفاده می‌شود. سپس روش KEI، روی ویژگی‌های جدید اعمال می‌شود. ازآنجایی‌که هدف روش‌های استخراج ویژگی، نگاشت ویژگی‌ها با حفظ خصوصیت‌های ویژگی‌های اصلی است. می‌توان اطمینان داشت خصوصیت‌های ویژگی‌های اصلی دادگان بعد از اعمال روش‌های استخراج ویژگی حفظ می‌شود.

در پایان، نمونه‌های برچسب‌دار منتخب از بخش منبع (از اعمال روش‌های KEI یا FE-KEI به‌دست می‌آید) به مدل آموزشی داده می‌شود. انتقال دانش در روش KEI، به‌صورت انتقال نمونه است. ولی روش FE-KEI تلفیقی از انتقال نمونه و انتقال نمایش ویژگی است. در شکل ۱، روند کلی کار انجام شده در این پژوهش، نشان داده شده است.

Algorithm 1: KEI method (source dataset, target dataset, α, β)

Input:

$S_i = \{b_{i1}, b_{i2}, ..., b_{id}\}_{i=1}^{n}$ Where b_{ij} is *j*th attribute of *i*th sample of source dataset, *n* is the number of samples , and *d* is number of features.

$T_i = \{a_{i1}, a_{i2}, ..., a_{id}\}_{i=1}^{m}$ Where a_{ij} is *j*th attribute of *i*th sample of target dataset, *m* is the number of samples , and *d* is number of features.

Output:

newSource

Method:

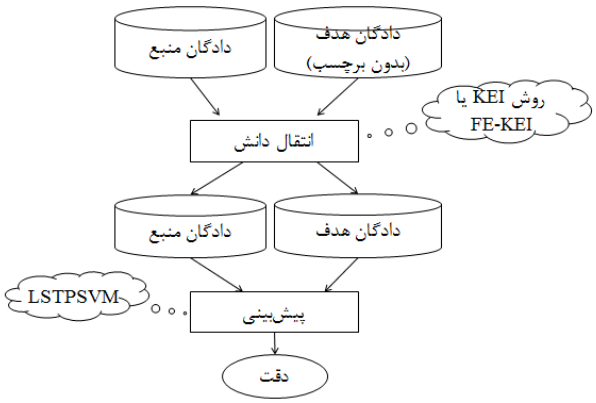
1. $P = \{p_1, p_2, ..., p_n\} = \{0^i\}_{i=1}^{n}$
2. μ=calculate average (mean) every features in target dataset
   ($\{\mu_1, \mu_2, ..., \mu_d\}$ d is number of feature)
3. σ=calculate standard deviation every features in target dataset
   ($\{\sigma_1, \sigma_2, ..., \sigma_d\}$ d is number of feature)
4. for i=1 to n // to number of samples source dataset
5. for j=1 to d
6. if ($b_{ij} \geq \mu_j - \alpha\sigma_j$ and $b_{ij} \leq \mu_j + \alpha\sigma_j$)
7. $p_i = p_i + 1$
8. end if
9. end for
10. $p_i = p_i / d$
11. end for
12. index $= \{i \mid p_i \geq \beta\}$
13. newSource $= \{\forall S_i \mid i \in index\}$

استفاده شده، پرداخته شده است. در بخش ۴-۳، نتایج تجربی از پیاده‌سازی روش‌های پیشنهادی و در بخش ۴-۴، مقایسه این نتایج با پژوهش‌های دیگر آورده شده است. در بخش ۴-۵ به تحلیل نتایج آزمایش‌ها پرداخته شده است.

۴-۱- دادگان ارزیابی

جهت بررسی مؤثر بودن روش‌های پیشنهادی در این پژوهش، از ۷ مجموعه داده از دادگان ناسا و ۳ مجموعه داده از دادگان SoftLab استفاده شده است. دادگان ارزیابی ناسا، از روی نرم‌افزارهای مربوط به پروژه‌های ناسا، گرفته شده است. سپس، متخصصان روی آن‌ها، ویژگی‌های مختلفی را تولید و یا حذف نموده‌اند، تا به شکل کنونی درآمده است. ویژگی‌های اصلی این دادگان، مرتبط با معیارهای پیچیدگی نرم‌افزار و تعداد اجزای ماژول نرم‌افزاری است. این نکته قابل ذکر است، که در اینجا ماژول کوچک‌ترین واحد عملیاتی است. ماژول، به معنای تابع است [۱۳].

دادگان SoftLab، مرتبط با کنترلرهای جاسازی‌شده از سازمان نرم‌افزاری ترکی هستند. SoftLab، آزمایشگاه تحقیقات نرم‌افزاری، در دانشگاه بوگازیسی استانبول در کشور ترکیه است. دادگان ناسا و



شکل ۱: روند کلی کار

۴- نتایج تجربی

در این بخش، به بررسی نتایج ارزیابی کارایی روش‌های پیشنهادی KEI و FE-KEI پرداخته شده است. برای پیاده‌سازی روش‌های ارائه‌شده در این پژوهش از متلب ۲۰۱۳ استفاده شده است. به این منظور، در بخش‌های ۴-۱ و ۴-۲ به ترتیب به معرفی دادگان و معیار ارزیابی

SoftLab به‌طور رایگان در مخزن‌های tera-PROMIS [۱۴] و TunedIT [۱۵] در دسترس هستند.

در جدول ۱، جزئیات دادگان ناسا و SofLab نشان داده شده است. ویژگی‌های استخراجی از این دادگان، سطح دانه‌بندی مشابهی دارند و در سطح ماژول هستند. تعداد نمونه‌ها در دادگان، نشان‌دهنده تعداد ماژول‌های نرم‌افزاری است. هر نمونه، بیانگر ماژولی است که ویژگی‌های مختلفی از آن استخراج شده است. این دادگان شامل ویژگی‌های McCabe، Halstead و تعداد خط کد استخراجی از کد منبع هستند. اندازه‌گیری McCabe و Halstead، مبتنی بر ماژول است. تمام دادگان SoftLab، شامل ۲۹ ویژگی هستند. اما دادگان ناسا، تعداد ویژگی‌های متفاوتی دارند، که کم‌ترین تعداد ۲۱ ویژگی و بیش‌ترین تعداد ۳۹ ویژگی است. در جدول ۲، ویژگی‌های دادگان ارزیابی‌شده در این پژوهش، آورده شده است. در این جدول، علامت × نشان‌دهنده وجود ویژگی و علامت ـ نشان‌دهنده عدم وجود ویژگی در دادگان است. ازآنجایی‌که، در تمام دادگان SoftLab، تعداد ویژگی‌ها یکسان است. فقط ویژگی‌های یکی از دادگان در جدول ۲ آورده شده است. در تمام دادگان، ویژگی آخر نشان‌دهنده کلاس نمونه است، که مستعد نقص یا غیرمستعد نقص بودن نمونه را نشان می‌دهد.

جدول ۱: اطلاعات دادگان پیش‌بینی نقص نرم‌افزار دادگان ناسا و SoftLab

دادگان		زبان	تعداد ویژگی	تعداد نمونه‌ها	تعداد نمونه‌های نقص‌دار
ناسا	CM1	C	۲۱	۴۹۷	۴۹
	KC1	C++	۲۱	۲۱۰۹	۳۲۶
	KC2	C++	۲۱	۵۲۲	۱۰۷
	KC3	JAVA	۳۹	۴۵۸	۴۳
	MC2	C++	۳۹	۱۶۱	۵۲
	MW1	C	۳۷	۴۰۳	۳۱
	PC1	C	۲۱	۱۱۰۹	۷۷
SoftLab	AR3	C	۲۹	۶۳	۶۳
	AR4	C	۲۹	۱۰۷	۲۰
	AR5	C	۲۹	۳۶	۸

۴-۲- معیار ارزیابی

جهت مقایسه روش پیشنهادی با دیگر روش‌ها، نیاز به معیارهای ارزیابی مناسبی است. در این پژوهش از معیار ارزیابی سطح زیرمنحنی مشخصه عملکرد سیستم (AUC) استفاده شده است. برای اندازه‌گیری این معیار ارزیابی، معیارهای ارزیابی مثبت واقعی (TP)، مثبت کاذب (FP)، منفی واقعی (TN)، منفی کاذب (FN)، نرخ مثبت واقعی (TP_r) و نرخ مثبت کاذب (FP_r) استفاده شده است.

معیار TP، بیانگر تعداد ماژول‌های مستعد نقص،که به‌درستی مستعد نقص پیش‌بینی شده‌اند. معیار FP، بیانگر تعداد ماژول‌های غیرمستعد نقص، که به اشتباه مستعد نقص پیش‌بینی شده‌اند. معیار FN، بیانگر تعداد ماژول‌های مستعد نقص، که به اشتباه غیرمستعد نقص پیش‌بینی شده‌اند. معیار TN، بیانگر تعداد ماژول‌های غیرمستعد نقص که به‌درستی غیرمستعد نقص پیش‌بینی شده‌اند. معیار TP_r، بیانگر نرخ تشخیص صحیح دسته مثبت توسط طبقه‌بند است. معیار FP_r، بیانگر نرخ تشخیص اشتباه دسته منفی توسط طبقه‌بند است. معیارهای TP_r و FP_r، به ترتیب توسط رابطه‌های (۵) و (۶) محاسبه می‌شوند. معیار AUC، توسط رابطه (۷) محاسبه می‌شود [۱، ۱۶].

$$TP_r = \frac{TP}{TP + FN} \quad (۵)$$

$$FP_r = \frac{FP}{FP + TN} \quad (۶)$$

$$AUC = \frac{1 + TP_r + FP_r}{2} \quad (۷)$$

۴-۳- نتایج آزمایش‌ها

در این پژوهش، برای تقسیم‌بندی دادگان به بخش‌های منبع و هدف دو نوع ارزیابی متفاوت وجود دارد. در ارزیابی اول، همه دادگان ناسا، به‌عنوان بخش منبع و هر مجموعه داده از دادگان SoftLab، به‌عنوان بخش هدف انتخاب شده‌اند. در ارزیابی دوم، هر مجموعه داده از دادگان ناسا، به‌عنوان بخش هدف و بقیه دادگان ناسا، به‌عنوان بخش منبع انتخاب شده‌اند (این دادگان در جدول ۱ معرفی شده‌اند). در تمام آزمایش‌های انجام‌شده، دادگان با روش استانداردسازی ـ z[۱۴]، نرمال شده‌اند.

تاکنون، از SVM در پژوهش‌های مختلفی در حوزه پیش‌بینی نقص نرم‌افزار استفاده شده است. همچنین در پژوهش [۱۷]، به بررسی اثربخشی نسخه‌های توسعه یافته SVM، در حوزه پیش‌بینی نقص نرم‌افزار، پرداخته شده است. در این پژوهش نیز، از ماشین بردار پشتیبان دوقلو با طرح حداقل مربعات بازگشتی (LSTPSVM[۱۵]) [۱۸] برای طبقه‌بندی ماژول‌های نرم‌افزاری استفاده شده است. در روش LSTPSVM، از هسته گوسی با ضریب تنظیم ۰/۱ استفاده شده است.

روش KEI

با توجه به اینکه، تعداد ویژگی‌های دادگان ناسا و SoftLab یکسان نیست، از ۱۷ ویژگی مشترک بین این دادگان استفاده شده است. در جدول ۳، انواع و نام ویژگی‌های مشترک آورده شده است.

در جدول ۴، نتایج پیاده‌سازی روش KEI آورده شده است. در روش KEI، دو پارامتر ورودی و نیاز به مقداردهی دارند. در این پژوهش مقدار ثابت و برابر با یک درنظر گرفته شده است (۱= β). مقدار بهینه ، از طریق روش ارزیابی چند بخشی ، به‌دست می‌آید. به این صورت که،

هر بار یک مجموعه داده از بخش منبع به‌عنوان بخش اعتبارسنجی و مابقی دادگان به‌عنوان بخش آموزش درنظر گرفته می‌شود (بخش منبع شامل دادگان چندین پروژه است). این‌کار تا زمانی‌که همه دادگان بخش منبع، به‌عنوان بخش اعتبارسنجی انتخاب شوند، ادامه پیدا می‌کند.

در جدول ۴، علاوه بر نتایج روش KEI، نتایج روش KEIminmax آورده شده است. روش KEIminmax، همان‌طور که از نامش مشخص است، مشابه روش KEI است. با این تفاوت که، به‌جای محاسبه انحراف معیار و میانگین، بیش‌ترین و کم‌ترین مقدار هر ویژگی محاسبه می‌شود. درواقع بازه رابطه (۲) و رابطه (۳)، به تغییر می‌کند. نتایج به‌دست‌آمده، برتری ۱۱/۱۷ درصدی روش KEI را در پیش‌بینی ماژول‌های مستعد نقص نسبت به روش KEIminmax نشان می‌دهد.

جدول ۲: لیست ویژگی‌های دادگان ناسا و SoftLab

	دادگان	ناسا							SoftLab
	نام ویژگی	CM1	KC1	KC2	KC3	MC2	MW1	PC1	AR
۱	Line of code	×	×	×	×	×	×	×	×
۲	Cyclomatic complexity	×	×	×	×	×	×	×	×
۳	Essential complexiy	×	×	×	×	×	×	×	–
۴	Design complexity	×	×	×	×	×	×	×	×
۵	Total number of operator	×	×	×	×	×	×	×	×
۶	Total number of operator	×	×	×	×	×	×	×	×
۷	Number of unique operators	×	×	×	×	×	×	×	×
۸	Number of unique operators	×	×	×	×	×	×	×	×
۹	Number of unique operators and operands	×	×	×	×	×	×	×	–
۱۰	Halstead Volume	×	×	×	×	×	×	×	×
۱۱	Halstead Difficult	×	×	×	×	×	×	×	×
۱۲	Halstead Length	×	×	×	×	×	×	×	×
۱۳	Halstead Content	×	×	×	×	×	×	×	–
۱۴	Halstead Effort	×	×	×	×	×	×	×	×
۱۵	Halstead Error estimate	×	×	×	×	×	×	×	×
۱۶	Halstead Programing time	×	×	×	×	×	×	×	×
۱۷	Number of blank lines	×	×	×	×	×	×	×	×
۱۸	Number of comment lines	×	×	×	×	×	×	×	×
۱۹	Number of code	×	×	×	–	–	–	×	–
۲۰	Number of lines both code & comments	×	×	×	×	×	×	×	×
۲۱	Branch count	×	×	×	×	×	×	×	×
۲۲	Number of condition	–	–	–	×	×	×	–	×
۲۳	Call pairs	–	–	–	×	×	×	–	×
۲۴	Cyclomatic density	–	–	–	×	×	×	–	×
۲۵	Number of decision	–	–	–	×	×	×	–	×
۲۶	Decision density	–	–	–	×	×	×	–	×
۲۷	Design density	–	–	–	×	×	×	–	–
۲۸	Number of edge	–	–	–	×	×	×	–	–
۲۹	Essential density	–	–	–	×	×	×	–	×
۳۰	LOC executable	–	–	–	×	×	×	–	×
۳۱	Number of parameter	–	–	–	×	×	×	–	–
۳۲	Global data complexity	–	–	–	×	×	–	–	–
۳۳	Global data density	–	–	–	×	×	–	–	×
۳۴	Halstead Level	–	–	–	×	×	×	–	×
۳۵	Maintenance severity	–	–	–	×	×	×	–	–
۳۶	Number of modified condition	–	–	–	×	×	×	–	–
۳۷	Number of multiple condition	–	–	–	×	×	×	–	×

۳۸	Number of node	-	-	×	×	×	-	-	-
۳۹	Normalized cyclomatic complexity	-	-	×	×	×	-	-	×
۴۰	Percent comments	-	-	×	×	×	-	-	-
۴۱	Halstead vacabulary	-	-	-	-	-	-	-	×

جدول ۳: ویژگی‌های مشترک دادگان ناسا و SoftLab

نوع ویژگی	تعداد ویژگی	نام ویژگی
McCabe	۲	Cyclomatic complexity, Design complexity
LOC	۴	lines both code & comments, comment lines, LOC executable, Line of code
Halstead	۱۰	Halstead Difficult, Halstead Effort, Halstead Error estimate, Halstead Length, Halstead Volume, Halstead Programing time, of unique operators, of unique operands, Total operands, Total operators
سایر	۱	Branch count

روش FE-KEI

در روش FE-KEI، از تکنیک‌های استخراج ویژگی بی‌نظارت تحلیل مؤلفه اصلی (PCA[۱۶])، مقیاس‌گذاری چندبعدی (MDS[۱۷])، نگاشت ویژگی ایزومتریک (ISOMAP[۱۸]) [۱۹] و تجزیه نامنفی ماتریس (NMF[۱۹]) [۲۰] استفاده شده است. سپس ویژگی‌های استخراجی به روش KEI، داده می‌شود. در روش KEI، ۱۷ ویژگی مشترک بین دادگان منبع و هدف استفاده شده است، ولی در روش FE-KEI، از کل ویژگی‌های دادگان منبع و هدف (ویژگی‌های مشترک و غیرمشترک دادگان) استفاده شده است.

تکنیک‌های استخراج ویژگی، تعداد ویژگی‌های که قرار است استخراج شود را به‌عنوان پارامتر ورودی می‌گیرند. تعداد ویژگی‌های ورودی ۱۰، ۱۵ و ۲۰ در نظر گرفته شده است. در پیاده‌سازی الگوریتم ISOMAP در مرحله یافتن نقاط همسایه از روش k نزدیک‌ترین همسایگی استفاده شده است، k =۵ در نظر گرفته شده است. در جدول ۵، نتایج پیاده‌سازی روش FE-KEI با ۱۰، ۱۵ و ۲۰ ویژگی استخراج‌شده، نشان داده شده است. نمودار شکل ۲ نیز، میانگین این نتایج را نشان می‌دهد. ازآنجایی‌که کم‌ترین تعداد ویژگی در دادگان استفاده شده، ۲۱ ویژگی است، حداکثر ویژگی استخراجی، ۲۰ ویژگی درنظر گرفته شده است.

نتایج جدول ۵ و نمودار شکل ۲، نشان می‌دهد روش FE-KEI با تکنیک‌های استخراج ویژگی PCA و NMF عملکرد بهتری داشته است. بنابراین، روش‌های PCA و NMF، تکنیک‌های استخراج ویژگی منتخب برای روش FE-KEI هستند.

هدف از روش FE-KEI، استخراج ویژگی‌های جدید، به‌طوری‌که توزیع داده‌ای دادگان حفظ شود، است. در PCA، ماتریس کواریانس محاسبه می‌شود. در MDS و ISOMAP، ماتریس فاصله محاسبه می‌شود. ماتریس کواریانس، بازتاب دقیق‌تری از توزیع داده‌ای دادگان، نسبت به ماتریس فاصله می‌دهد. بنابراین، این موضوع را می‌توان دلیل برتری تکنیک PCA نسبت به MDS و ISOMAP در روش FE-KEI دانست. NMF، روش مبتنی بر تکرار است، که به‌دنبال پیدا کردن عوامل نامنفی می‌باشد. در این روش، حفظ نامنفی بودن در تحلیل اطلاعات، خواص اصلی دادگان را حفظ می‌کند. ازآنجایی‌که روش NMF خواص دادگان را حفظ می‌کند، در روش FE-KEI عملکرد بهتری نسبت به بقیه تکنیک‌های استخراج ویژگی داشته است.

جدول ۴: نتایج پیاده‌سازی روش KEI و مقایسه آن با روش KEIminmax توسط AUC

α	KEIminmax	KEI	روش / هدف⟵منبع
۱	**۸۰/۲۳**	۷۹/۲۰	NASA⟶ar3
۱	۷۲/۳۳	**۸۴/۲۸**	NASA⟶ar4
۱	۸۶/۶۱	**۹۲/۸۶**	NASA⟶ar5
۱	۶۱/۵۰	**۷۹/۳۸**	NASA⟶cm1
۱	۶۵/۱۵	**۸۱/۶۹**	NASA⟶kc1
۱	۷۳/۸۲	**۸۶/۱۴**	NASA⟶kc2
۱	۶۸/۹۵	**۸۵/۲۶**	NASA⟶kc3
۱	۶۲/۳۰	**۷۶/۰۳**	NASA⟶mc2
۱	۶۸/۴۱	**۷۶/۶۱**	NASA⟶mw1
۱	۶۶/۱۸	**۷۵/۷۲**	NASA⟶pc1
-	۷۰/۵۵	**۸۱/۷۲**	میانگین

×NASA⟶ar3 به این معنی است که تمام دادگان ناسا به‌عنوان بخش منبع و دادگان ar3 بخش هدف است. NASA⟶cm1 نیز به این معنی است که دادگان cm1 به‌عنوان بخش هدف و بقیه دادگان ناسا به‌عنوان بخش منبع است.

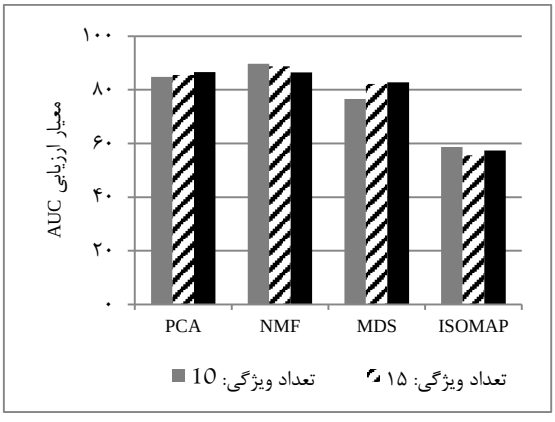
همان‌طور که در جدول ۵ و شکل ۲، مشاهده می‌شود، در FE-KEI با PCA، با افزایش تعداد ویژگی‌های استخراجی، دقت نیز افزایش می‌یابد. اما در FE-KEI با NMF، با کاهش تعداد ویژگی‌ها، دقت افزایش می‌یابد.

برای افزایش اثربخشی روش KEI، روش FE-KEI پیشنهاد شد. ازاین‌رو، نمودار میله‌ای شکل ۳، مقایسه‌ای از روش‌های KEI و FE-KEI را نشان می‌دهد. در این مقایسه، بهترین عملکرد از(NMF) FE-KEI و FEKEI (PCA) قرار داده شده است. در این نمودار، به‌وضوح، عملکرد بهتر روش FE-KEI نسبت به روش KEI نشان داده شده است. فقط روی

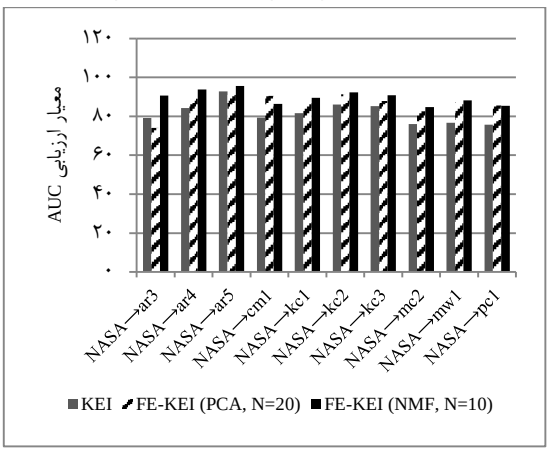
NASA→ar3 روش KEI افزایش دقت نسبت به روش FE-KEI داشته است. روش FE-KEI (NMF)، 01/8 درصد و روش FE-KEI (PCA)، 92/4 درصد نسبت به روش KEI، افزایش دقت داشته‌اند. می‌توان از این نمودار نتیجه گرفت، زمانی که از همه ویژگی‌های دادگان منبع به‌جای ویژگی‌های مشترک دادگان منبع و هدف برای آموزش مدل استفاده می‌شود، بهبود دقت بیش‌تری را به‌همراه خواهد داشت.

جدول ۵: نتایج پیاده‌سازی روش FE-KEI توسط AUC

روش / هدف→منبع	PCA α =۱			NMF α =۱			MDS α =۰/۹			ISOMAP α =۰/۸		
تعداد ویژگی استخراجی	۱۰	۱۵	۲۰	۱۰	۱۵	۲۰	۱۰	۱۵	۲۰	۱۰	۱۵	۲۰
NASA→ar3	۷۶/۷۰	۷۵/۷۹	۷۳/۹۸	**۹۰/۶۲**	۸۷/۷۴	۸۷/۲۵	۸۰/۳۴	۸۶/۵۹	۸۶/۵۹	۶۳/۴۵	–	۶۸/۷۵
NASA→ar4	۸۵/۴۰	۸۲/۳۳	۸۹/۲۵	**۹۳/۷۸**	۹۰/۹۳	۸۸/۶۷	۷۰/۰۰	۸۲/۵۰	۷۸/۲۷	۵۸/۳۰	۵۲/۵۰	۵۲/۵۰
NASA→ar5	۹۰/۱۸	۹۶/۴۳	۹۱/۰۷	**۹۵/۵۷**	۹۱/۷۹	۸۸/۰۶	۹۱/۹۶	۹۱/۹۶	۸۸/۳۹	۵۹/۳۷	۶۰/۷۱	–
NASA→cm1	۸۵/۲۳	۸۶/۵۱	**۹۰/۴۸**	۸۶/۴۱	۸۶/۷۳	۸۳/۷۳	۷۱/۱۳	۸۲/۳۷	۸۴/۵۲	۵۷/۰۴	۵۱/۹۳	۵۰/۶۹
NASA→kc1	۸۳/۴۱	۸۵/۸۲	۸۶/۹۱	۸۹/۴۹	**۸۹/۶۸**	۸۶/۲۶	۶۴/۷۶	۶۶/۱۶	۶۵/۳۹	۵۹/۴۳	۵۲/۴۲	۵۵/۳۱
NASA→kc2	۸۸/۷۹	۸۹/۲۴	۹۱/۳۴	**۹۲/۲۶**	۹۲/۱۸	۸۹/۶۹	۸۳/۰۲	۸۸/۶۱	۹۲/۴۳	۵۶/۲۶	۵۳/۶۲	۵۰/۴۷
NASA→kc3	**۹۱/۶۹**	۸۹/۲۰	۸۷/۸۸	۹۰/۸۶	۹۰/۳۲	۹۰/۳۹	۸۰/۴۷	۸۴/۰۳	۸۵/۹۶	۵۹/۳۱	–	۵۰/۴۷
NASA→mc2	۷۴/۸۱	۷۵/۴۰	۸۲/۶۷	۸۴/۷۱	**۸۵/۰۲**	۷۵/۲۳	۸۰/۳۱	۸۷/۵۴	۸۹/۴۷	۵۶/۲۵	۵۱/۹۲	۵۸/۲۸
NASA→mw1	۸۶/۶۹	۸۶/۵۶	۸۷/۲۳	۸۸/۲۵	۸۷/۳۵	**۸۸/۴۲**	۷۹/۳۹	۷۹/۰۳	۸۲/۵۳	۶۱/۳۷	۶۸/۵۵	۷۲/۱۸
NASA→pc1	۸۴/۵۳	**۸۸/۲۷**	۸۵/۶۰	۸۵/۳۷	۸۶/۱۴	۸۶/۵۲	۶۴/۷۲	۷۳/۴۴	۷۴/۵۵	۵۵/۵۱	۵۲/۸۶	–
میانگین	۸۴/۷۷	۸۵/۵۵	۸۶/۶۴	**۸۹/۷۳**	۸۸/۷۹	۸۶/۴۲	۷۶/۶۱	۸۲/۲۲	۸۲/۸۱	۵۸/۶۳	۵۵/۵۶	۵۷/۳۳



شکل ۲: نمودار میانگین نتایج پیاده‌سازی روش FE-KEI



شکل ۳: مقایسه روش‌های KEI و FE-KEI

۴-۴- مقایسه با دیگر پژوهش‌ها

نتایج حاصل از روش‌های پیشنهادی، فقط با نتایج دو پژوهش دیگر، مقایسه شده است. زیرا، فقط این پژوهش‌ها، آزمایش‌های خود را، روی دادگان ناسا و SoftLab انجام داده‌اند. علاوه‌براین، روش پیشنهادی، با روش‌های WPDP و فیلتر NN مقایسه شده است.

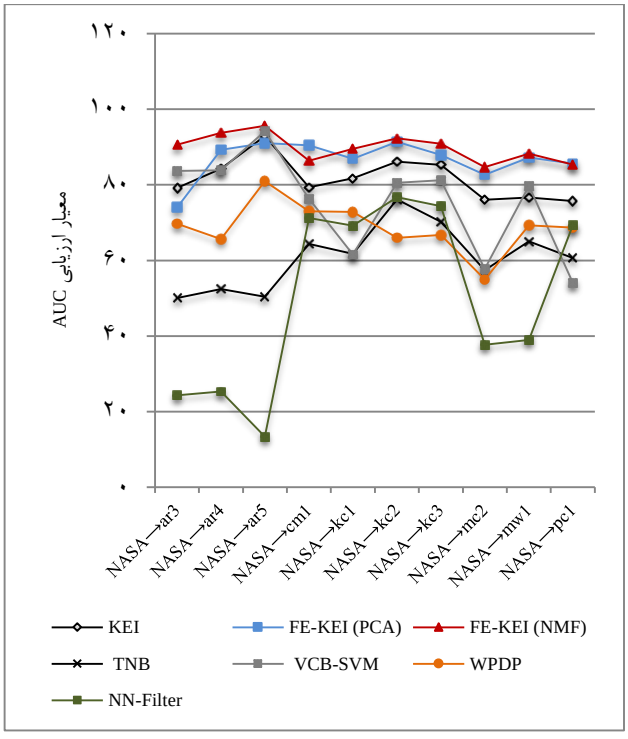
در WPDP آزمایش‌شده در این پژوهش، دادگان از طریق ارزیابی k بخشی (که k =۱۰ است)، به دو بخش آموزش و آزمون تقسیم می‌شوند. در ارزیابی k بخشی، داده به k بخش به‌صورت تصادفی تقسیم می‌شود. هر بار یک بخش از دادگان به‌عنوان داده آزمون و بقیه به‌عنوان داده آموزش انتخاب می‌شود. درواقع، ۱۰ بار ارزیابی تکرار می‌شود. در این روش نیز، از روش LSTPSVM برای طبقه‌بندی ماژول‌ها به مستعد نقص و غیرمستعد نقص استفاده شده است. میانگین نتایج به‌دست‌آمده از ۱۰ ارزیابی، نتیجه نهایی است.

در جدول ۶ و نمودار شکل ۴، به مقایسه‌های انجام‌شده، پرداخته شده است. در این جدول بهترین نتایج برای هر یک از دادگان، برجسته شده است. نتایج به‌دست‌آمده، عملکرد بهتر روش پیشنهادی، نسبت به پژوهش‌های مشابه را نشان می‌دهد. روش FE-KEI (NMF) در تمام ارزیابی‌های انجام‌شده، نسبت به سایر روش‌ها برتری داشته است. روش FE-KEI (PCA) در ۷ ارزیابی از ۱۰ ارزیابی انجام‌شده، نسبت به سایر روش‌ها برتری داشته است.

اگر میانگین نتایج روش پیشنهادی، با نتایج روش VCB-SVM (که عملکرد بهتری نسبت به سایر روش‌های مقایسه‌شده، دارد) مقایسه شود. ملاحظه می‌شود، روش FE-KEI (NMF)، ۱۴/۴۴ درصد، روش FE-KEI (PCA)، ۱۱/۳۵ درصد و روش KEI، ۶/۴۳ درصد، نسبت به VCB-SVM افزایش دقت داشته‌اند. همچنین روش KEI، ۳۳/۷۶ درصد، روش FE-KEI (PCA)، ۳۸/۶۸ درصد و روش FE-KEI (NMF)، ۴۱/۷۷ درصد نسبت به روش WPDP افزایش دقت داشته‌اند. نمودار شکل ۴، به‌وضوح برتری روش‌های پیشنهادی، نسبت به پژوهش‌های مشابه را نشان می‌دهد.

جدول ۶: مقایسه نتایج روش‌های پیشنهادی با دیگر پژوهش‌ها

روش / هدف←منبع	KEI	FE-KEI (PCA)	FE-KEI (NMF)	TNB[۹]	VCB-SVM [۱۰]	WPDP	NN-Filter
NASA→ar3	۷۹/۲۰	۷۳/۹۸	**۹۰/۶۲**	۵۰/۰۹	۸۳/۷	۸۵/۶۳	۲۴/۳۲
NASA→ar4	۸۴/۲۸	۸۹/۲۵	**۹۳/۷۸**	۵۲/۴۴	۸۳/۸	۶۹/۵۱	۲۵/۳۲
NASA→ar5	۹۲/۸۶	۹۱/۰۷	**۹۵/۵۷**	۵۰/۳۶	۹۴/۲	۶۳/۵۹	۱۳/۳۹
NASA→cm1	۷۹/۳۸	**۹۰/۴۸**	۸۶/۴۱	۶۴/۴۰	۷۶/۳	۳۵/۸۹	۷۱/۲۲
NASA→kc1	۸۱/۶۹	۸۶/۹۱	**۸۹/۴۹**	۶۱/۷۰	۶۱/۶	۴۶/۵۷	۶۹/۲۱
NASA→kc2	۸۶/۱۴	۹۱/۳۴	**۹۲/۲۶**	۷۶/۱۲	۸۰/۵	۴۷/۲۶	۷۶/۶۸
NASA→kc3	۸۵/۲۶	۸۷/۸۸	**۹۰/۸۶**	۷۰/۲۱	۸۱/۲	۳۱/۴۲	۷۴/۳۷
NASA→mc2	۷۶/۰۳	۸۲/۶۷	**۸۴/۷۱**	۵۷/۴۵	۵۷/۸	۳۸/۱۳	۳۷/۶۷
NASA→mw1	۷۶/۶۱	۸۷/۲۳	**۸۸/۲۵**	۶۵/۰۶	۷۹/۷	۳۱/۳۵	۳۸/۹۸
NASA→pc1	۷۵/۷۲	**۸۵/۶۰**	۸۵/۳۷	۶۰/۶۶	۵۴/۱	۳۰/۳۰	۶۹/۲۸
میانگین	۸۱/۷۲	۸۶/۶۴	**۸۹/۷۳**	۶۰/۸۵	۷۵/۲۹	۴۷/۹۶	۴۸/۶۱



شکل ۴: مقایسه نتایج روش‌های پیشنهادی با دیگر پژوهش‌ها

۴-۵- تحلیل نتایج

در روش پیشنهادشده در این پژوهش، از تکنیک یادگیری انتقالی استفاده شده است که انتقال دانش، به‌صورت انتقال نمونه می‌باشد. برای انتقال نمونه لازم است، اطلاعاتی از توزیع داده‌ای بخش هدف، جمع‌آوری شود. با کمک این اطلاعات، تشابه نمونه‌های بخش منبع و بخش هدف اندازه‌گیری می‌شود. در پایان نمونه‌های مناسب، انتقال داده می‌شود. در بخشی از پژوهش‌های CPDP پیشین، از بیش‌ترین و کم‌ترین مقدار هر ویژگی برای تخمین توزیع داده‌ای بخش هدف استفاده شده است. درصورتی‌که در روش پیشنهادی (KEI) از انحراف معیار و میانگین هر ویژگی، برای تخمین توزیع داده‌ای بخش هدف استفاده شده است. انحراف معیار و میانگین، بازتاب دقیق‌تری از توزیع داده‌ای دارند. نتایج به‌دست‌آمده از آزمایش‌ها، بازگوکننده این موضوع هستند.

علاوه‌براین، در پژوهش‌های پیشین، فقط از ویژگی‌های مشترک بین بخش‌های منبع و هدف استفاده شده است. درصورتی‌که، در بهبودیافته روش پیشنهادی (FE-KEI)، از همه ویژگی‌های دادگان بخش هدف و منبع استفاده می‌شود. نتایج به‌دست‌آمده معرف این است که ویژگی‌های غیرمشترک بخش‌های منبع و هدف، در افزایش دقت پیش‌بینی مؤثر است. در ساختار الگوریتم‌های PCA، MDS و ISOMAP، از بزرگ‌ترین مقادیر ویژه به‌دست‌آمده، ویژگی‌های جدید استخراج می‌شوند [۱۹]. در NMF، ویژگی‌های جدید، به‌صورت نزولی مرتب می‌شوند [۲۰]. بنابراین، می‌توان اطمینان داشت ویژگی‌های بخش منبع و هدف بعد از اعمال تکنیک استخراج ویژگی، برابر و مشترک هستند. روش FE-KEI، فقط روی دادگانی که ویژگی مشترک بیش‌تری نسبت به ویژگی‌های غیر مشترک دارند، عملکرد خوبی دارد.

به‌طور خلاصه با تحلیل نتایج به‌دست‌آمده، می‌توان دریافت:

- روش KEI در بهبود دقت پیش‌بینی مؤثر است، زیرا انحراف معیار و میانگین هر ویژگی، اطلاعات بهتری از توزیع داده‌ای نسبت به کم‌ترین و بیش‌ترین مقدار هر ویژگی می‌دهند.
- استخراج ویژگی‌های جدید، قبل از اعمال روش KEI، در بهبود دقت پیش‌بینی مؤثر است.
- روش‌های PCA و NMF، روش‌های برگزیده برای استخراج ویژگی در روش FE-KEI هستند.
- در روش FE-KEI، NMF نسبت به PCA اثربخشی بیش‌تری دارد.

۵- نتیجه‌گیری و پژوهش‌های پیش‌رو

کمبود دادگان محلی برچسب‌دار، به کار بردن روش‌های پیش‌بینی نقص نرم‌افزار درون‌پروژه‌ای را دشوار می‌سازد. ازاین‌رو، روش‌های پیش‌بینی نقص نرم‌افزار بین‌پروژه‌ای مطرح شده است، که چالش اصلی در این روش‌ها متفاوت بودن توزیع داده‌ای بخش‌های منبع و هدف است. در این پژوهش، پیش‌بینی نقص بین‌پروژه‌ای مورد مطالعه قرار گرفته است و برای رفع چالش اصلی این حوزه، روش‌های بازه تخمین دانش (KEI) و بهبودیافته این روش (FE-KEI) پیشنهاد شده است. در روش KEI انتقال دانش به‌صورت، انتقال نمونه است. ولی روش FE-KEI تلفیقی از انتقال نمونه و انتقال نمایش ویژگی است.

در روش KEI، انحراف معیار و میانگین هر ویژگی به‌عنوان اطلاعات بخش هدف استخراج می‌شود. با کمک اطلاعات استخراج شده، نمونه‌های مناسب بخش منبع انتخاب می‌شوند. در روش KEI از ویژگی‌های اشتراکی دادگان بخش‌های منبع و هدف استفاده می‌شود. لذا، روش FE-KEI برای استفاده از تمام ویژگی‌های این دادگان پیشنهاد شده است. در این روش ابتدا، تکنیک استخراج ویژگی روی دادگان بخش‌های منبع و هدف اعمال می‌شود. سپس، روش KEI روی ویژگی‌های جدید اعمال می‌شود. روش‌های پیشنهادی روی دادگان پیش‌بینی نقص نرم‌افزار ناسا و SoftLab مورد ارزیابی قرار داده شده است. نتایج حاکی از مؤثر بودن روش‌های پیشنهادی است. در آینده قصد بر این است، ترکیب همه خصوصیت‌های آماری ازجمله بیش‌ترین مقدار، کم‌ترین مقدار، انحراف معیار، میانه و میانگین به‌عنوان اطلاعات استخراج‌شده از بخش هدف استفاده کرد.

مراجع

[۱] فاطمه علیقارداشی، محمدعلی زارع چاهوکی، «تأثیر ترکیب روش‌های انتخاب ویژگی فیلتر و بسته‌بندی در بهبود اشکال نرم‌افزار»، مجله مهندسی برق دانشگاه تبریز، دوره ۴۷، شماره ۱، صفحات ۱۸۳ـ۱۹۵، بهار ۱۳۹۶.

[2] T. Menzies, J. Greenwald, and A. Frank, "Data mining static code attributes to learn defect predictors," IEEE Transactions on Software Engineering, vol. 33, no. 1, pp. 2–13, Sep. 2007.

[3] Q. Song, Z. Jia, M. Shepperd, S. Ying, and J. Liu, "A general software defect-proneness prediction framework," IEEE Transactions on Software Engineering, vol. 37, no. 3, pp. 356–370, May. 2011.

[4] R. Malhotra, and A. Jain, "Fault prediction using statistical and machine learning methods for improving software quality," Journal of Information Processing Systems, vol. 8, no. 2, pp. 241–262, Jun. 2012.

[5] F. Peters, T. Menzies, and A. Marcus, "Better cross company defect prediction," 10th IEEE Working Conference on Mining Software Repositories (MSR). IEEE, pp. 409–418, Oct. 2013.

[6] F. Rahman, D. Posnett, and P. Devanbu, "Recalling the imprecision of cross-project defect prediction," Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering. ACM, no. 61, Nov. 2012.

[7] B. Turhan, T. Menzies, A. B. Bener, and J. Di Stefan, "On the relative value of cross-company and within-company data for defect prediction," Empirical Software Engineering, vol. 14, no. 5, pp. 540-578, Oct. 2009.

[8] S. J. Pan, and Q. Yang, "A survey on transfer learning," IEEE Transactions on Knowledge and Data Engineering, vol. 22, no. 10, pp. 1345-1359, Oct. 2010.

[9] Y. Ma, G. Luo, X. Zeng, and A. Chen, "Transfer learning for cross-company software defect prediction," Information and Software Technology, vol. 54, no. 3, pp. 248–256, Oct. 2012.

[10] D. Ryu, O. Choi, and J. Baik, "Value-cognitive boosting with a support vector machine for cross-project defect prediction," Empirical Software Engineering, vol. 21, no. 1, pp. 43-71, Dec. 2016.

[11] L. Chen, B.Fang, Z. Shang, and Y. Tang, "Negative samples reduction in cross-company software defects prediction," Information and Software Technology, vol.62, pp. 67-77, Jun. 2015.

[12] J. Nam, S. J. Pan, and S. Kim, "Transfer defect learning," Proceedings of the 2013 International Conference on Software Engineering. IEEE Press, pp.382-391, May. 2013.

[13] M. Shepperd, Q. Song, Z. Sun, and C. Mair, "Data quality : some comments on the NASA software defect datasets," IEEE Transactions on Software Engineering, vol. 39, no. 9, pp. 1208–1215, Sept. 2013.

[14] T. Menzies, J. Sayyad, "tera-PROMISE HOME." Internet: http://openscience.us/repo/defect/, [May. 14, 2016].

[15] "TunedIT." Internet: http://tunedit.org/repo/PROMISE/DefectPrediction, [Jun. 3, 2016].

[16] D. Rodriguez, I. Herraiz, R. Harrison, J. Dolado, and J. C. Riquelme, "Preliminary comparison of techniques for dealing with imbalance in software defect prediction," Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering. ACM, no. 43, May. 2014.

[۱۷] علیقارداشی، فاطمه، *پیش‌بینی اشکال در ماژول‌های نرم‌افزاری با تلفیق روش‌های انتخاب ویژگی در مدل‌های توسعه‌یافته ماشین بردار پشتیبان*، پایان‌نامه کارشناسی ارشد، دانشکده مهندسی برق و کامپیوتر، دانشگاه یزد، یزد، ایران، ۱۳۹۴.

[18] Y.-H. Shao, N.-Y. Deng, and Z.-M. Yang, "Least squares recursive projection twin support vector machine for classification," Pattern Recognition, vol. 45, no. 6, pp. 2299–2307, Jun. 2012.

[19] E. Alpaydin, *Introduction To Machine Learning*, 2nd ed, MIT press, 2014.

[20] M. W. Berry, M. Browne, A.N. Langville, P. Pauca, and R. J. Plemmons, "Algorithms and applications for approximate nonnegative matrix factorization," Computational statistics & data analysis, vol. 52, no. 1, pp. 155-173, Sep. 2007.

زیرنویس‌ها

[۱] Fault-proneness module
[۲] Software defect/fault prediction
[۳] Within Project Defect Prediction
[۴] Cross Project Defect Prediction
[۵] Data distribution
[۶] Source data
[۷] Target data
[۸] Transductive transfer learning
[۹] Instance transfer
[۱۰] Feature-representation transfer
[۱۱] Knowledge Estimation Interval
[۱۲] Transfer Naïve Bayes
[۱۳] Feature Extraction-KEI
[۱۴] Z-score
[۱۵] Least Squares recursive Projection Twin SVM
[۱۶] Principal Component Analysis
[۱۷] MultiDimensial Scaling
[۱۸] ISOmetric feature MAPping
[۱۹] Non-negative Matrix Factorization