روشی برای بهبود آزمون جهش پیش‌گویانه با در نظر گرفتن اثر داده‌های از دست رفته

طه رستمی[۱]، سعید جلیلی[۲]
[۱]دانشگاه تربیت مدرس، taha.rostami@modares.ac.ir
[۲]دانشگاه تربیت مدرس، sjalili@modares.ac.ir

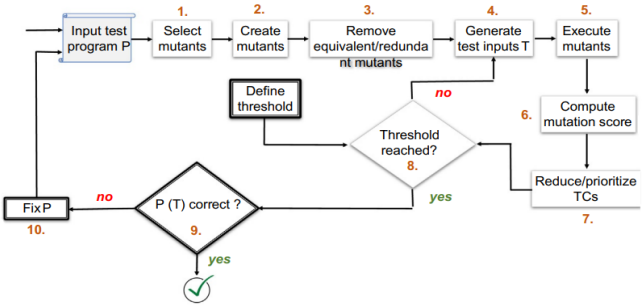
چکیده – آزمون جهش روشی قدرتمند است که در آزمون نرم‌افزار برای فعالیت‌های گوناگون از جمله راهنمایی برای تولید آزمون و ارزیابی کیفیت مجموعه آزمون استفاده می‌شود. با این وجود، هزینه زیاد آزمون جهش مقیاس‌پذیری آن را به طور جدی تهدید می‌کند. در همین راستا، آزمون جهش پیش‌گویانه به عنوان روشی برای کاهش هزینه‌های آزمون جهش پیشنهاد شده است که در آن هدف پیش‌بینی کردن کشف شدن یا کشف نشدن یک برنامه جهش‌یافته توسط مدل‌های یادگیری ماشین است. اخیراً نشان داده شده است که کارهای قبلی آزمون جهش‌پیش‌گویانه تاثیر برنامه‌های جهش‌یافته کشف نشده را در نظر نگرفتند و وقتی پیش‌بینی مدل‌های یادگیری ماشین قبلی محدود به چنین برنامه‌های جهش‌یافته‌ای شود AUC به ٪۶۱ کاهش پیدا می‌کند. در این پژوهش، علاوه بر تاثیر برنامه‌های جهش‌یافته کشف نشده، تاثیر داده‌های از دست رفته نیز در نظر گرفته شده است در حالی که کارهای گذشته آن را نادیده گرفته بودند و روشی پیشنهاد شده است که دقت AUC را از ٪۶۱ به ٪۷۲ بهبود داده است.

کلید واژه- آزمون جهش، آزمون نرم‌افزار، امتیاز جهش، یادگیری ماشین

۱- مقدمه

آزمون جهش روشی است که در آزمون نرم‌افزار برای فعالیت‌های گوناگون از جمله راهنمایی برای تولید آزمون و ارزیابی کیفیت مجموعه آزمون استفاده می‌شود و تقریبا تمام معیارهای دیگر کیفیت مجموعه آزمون را شامل می‌شود [۱].

شکل ۱ فرآیند آزمون جهش مدرن را نشان می‌دهد. مطابق این شکل فرآیند کلی این آزمون به این صورت است که ابتدا یک برنامه به عنوان ورودی داده می‌شود. سپس به کمک یک ابزار آزمون جهش، تعداد زیادی برنامه جهش‌یافته(یعنی برنامه‌هایی که از نظر گرامری تفاوتی جزئی با برنامه اصلی دارند) تولید می‌شود. در مرحله بعدی آزمون‌گر سعی می‌کند مجموعه آزمونی طراحی کند که بتواند تمام



شکل ۱: فرآیند آزمون جهش مدرن[۱]

برنامه‌های جهش‌یافته را کشف کند. به عبارت دیگر، اگر رفتار برنامه اصلی و برنامه‌جهش یافته وقتی که آن‌ها را علیه یک مورد آزمون اجرا می‌کنیم متفات باشد، می‌گوییم برنامه‌جهش یافته کشف شده است، در غیر این صورت می‌گوییم برنامه جهش‌یافته نجات پیدا کرده است. وقتی که مجموعه آزمون طراحی شد، برنامه اصلی و تمام برنامه‌های جهش‌یافته علیه یک به یک موارد آزمون موجود در مجموعه آزمون اجرا می‌شوند. سپس با ردگیری نتایج اجرای بدست آمده، امتیاز جهش(تعداد برنامه‌های جهش‌یافته کشف شده تقسیم بر تعداد کل برنامه‌های جهش‌یافته) محاسبه می‌شود. اگر امتیاز جهش به آستانه از پیش‌تعیین شده برسد، می‌توان مجموعه آزمون را مجموعه آزمون باکیفیت مناسبی در نظر گرفت. اما در غیر این صورت نمی‌توان مجموعه آزمون را مجموعه با کیفیتی دانست و آزمون‌گر باید مجموعه آزمون را بهبود دهد.

هزینه زیاد آزمون جهش را می‌توان مهم‌ترین چالشی دانست که مقیاس پذیری آن را تهدید می‌کند. به همین دلیل روش‌های مختلفی برای کاهش هزینه آزمون جهش پیشنهاد شده است. یکی از روش‌هایی که اخیراً برای کاهش هزینه زمان اجرای این آزمون پیشنهاد شده است، آزمون جهش پیش‌گویانه است [۲و۳]. در آزمون جهش پیش‌گویانه، یک برنامه جهش‌یافته و یک مجموعه آزمون داده می‌شود. سپس به جای اینکه واقعاً برنامه‌های جهش‌یافته و موارد آزمون برعلیه یکدیگر اجرا شوند، تعدادی ویژگی استخراج می‌شود و به کمک مدل‌های یادگیری ماشین نظارت شده کشف شدن یا کشف نشدن یک برنامه جهش‌یافته پیش‌بینی می‌گردد.

بعد از کار ژانگ و همکاران [۲و۳] که آزمون جهش پیش‌گویانه را پیشنهاد کردند، توجه ویژه‌ای به این روش شده است. مائو و همکاران [۴]، چندین معیار نرم‌افزاری را به مجموعه ویژگی‌های قبلی اضافه کردند و با انتخاب زیرمجموعه‌ای از این ویژگی‌ها و به کمک الگوریتم جنگل تصادفی کار قبلی را بهبود دادند.

اخیراً، آقا محمدی و میریان-حسین آبادی [۵] برنامه‌های جهش‌یافته‌ای که هیچ آزمونی آن‌ها را اجرا نکرده بود از دیتاست تهیه شده در کار مائو و همکاران [۴] حذف کردند و بعد از نمونه برداری افزایشی و انتخاب زیرمجموعه‌ای از ویژگی‌ها توانستند کارهای قبلی را بهبود دهند.

با بررسی مجموعه داده تهیه شده توسط مائو و همکاران [۴]، مشاهده شد که این مجموعه داده حاوی تعداد زیادی داده‌های از دست رفته است که کارهای قبلی آن‌ها را نادیده گرفتند و سطر های شامل داده‌های از دست رفته را هم از مجموعه آموزش و هم از مجموعه آزمون حذف کردند. این درحالی است که نباید سطرهای شامل مقادیر از دست رفته را از مجموعه آزمون حذف کرد و باید همانند سایر سطرها، بتوان این سطرها را نیز پیش‌بینی کرد.

هدف اصلی در این مقاله، پیشنهاد روشی است که با در نظر گرفتن فاکتورهای نادیده گرفته شده در کارهای گذشته از جمله داده‌های از دست رفته، دقت کارهای گذشته را بهبود دهد.

سازمان‌دهی مقاله به این صورت است: در بخش ۲، کارهای مرتبط بررسی شده‌است. در بخش ۳، روش پیشنهادی معرفی می‌شود. سپس در بخش ۴، نتایج ارزیابی‌ها و مقایسه روش پیشنهادی با بهترین کارهای گذشته ارائه می‌شود. در بخش ۵، تهدیدهایی معرفی شدند که اعتبار روش پیشنهادی را به چالش می‌کشند. در نهایت در بخش ۶، جمع‌بندی و نتیجه گیری آورده شده است.

۲- تاریخچه پژوهش

در این بخش مروری بر کارهای گذشته انجام شده‌است.

ژانگ و همکاران [۲و۳]، آزمون جهش پیش‌گویانه را پیشنهاد کردند که در آن تعدادی ویژگی ایستا و پویا برای هر برنامه جهش‌یافته استخراج می‌شود، سپس به کمک الگوریتم جنگل تصادفی یک مدل یادگیری ماشین ساخته می‌شود که برای پیش‌بینی کشف شدن یا نجات یافتن یک برنامه‌جهش یافته به‌کار می‌رود. آن‌ها AUC برابر ٪۸۰ را در مقاله خود گزارش کردند.

بعد از آن، مائو و همکاران [۴] چندین معیار نرم‌افزاری را به مجموعه ویژگی‌های قبلی اضافه کردند. سپس با انتخاب زیرمجموعه‌ای از ویژگی‌ها و ساخت مدلی به کمک الگوریتم جنگل تصادفی کار قبلی را بهبود دادند. برای این کار آن‌ها یک مجموعه داده با تعداد پروژه‌های بیشتر از کارهای قبلی شامل ۶۵۴ پروژه متن باز جاوا را گردآوری کردند و توانستند AUC برابر ٪۸۹ و f1 برابر ٪۷۱ به‌دست آورند.

در همان سال، نعیم و همکاران [۶]، ویژگی‌هایی مرتبط با نظریه آزمون مبتنی بر محدودیت و همچنین شبکه‌های پیچیده را برای کار خود به همراه مدل‌های یادگیری عمیق استفاده کردند و f1 برابر ٪۸۰ را روی ۵ پروژه جاوا مورد بررسی خود گزارش کردند. ویژگی‌های استفاده شده در کار نعیم و همکاران [۶] به نظر ویژگی‌های متمایز کننده‌ای می‌آید، با این حال نقطه ضعف جدی کار آن‌ها این است که تحلیلی برای زمان اجرا مورد نیاز برای محاسبه ویژگی‌های استفاده شده در کار خود انجام ندادند. به عبارت دیگر، آزمون جهش پیش‌گویانه فقط در صورتی توجیه پذیر است که ما بتوانیم استخراج ویژگی و انجام پیش‌بینی‌ها را در زمان بسیار کوتاه‌تر در مقایسه با آزمون جهش معمولی انجام دهیم. به طور خلاصه، در پایان سال ۲۰۱۹ میلادی بیشترین دقت را نعیم و همکاران [۶] گزارش کردند اما شاید به دلیل مشکلاتی که در این کار بود در کارهای بعدی توجه بیشتر و بیشتری به کار ژانگ و همکاران [۲و۳] و همچنین کار مائو و همکاران [۴] شده است.

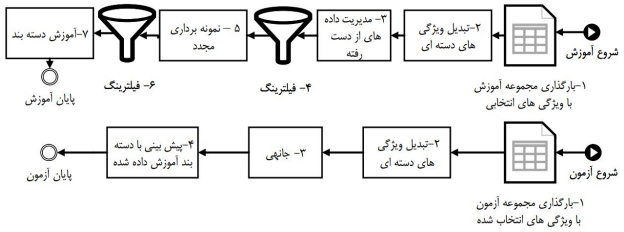
در سال ۲۰۲۰ میلادی، ژانگ و همکاران [۷] یک مدل احتمالاتی بدون ناظر به نام CBUA پیشنهاد کردند که می‌توان از آن نیز برای پیش‌بینی کشف شدن یا کشف نشدن یک برنامه جهش‌یافته استفاده کرد. در واقع، آن‌ها با در نظر گرفتن نتایج حاصل از مطالعات حوزه نرم‌افزار (به عنوان مثال اثر رانش نرم‌افزار) یک ویژگی جدید طراحی کردند که می‌توان از آن برای پیش‌بینی کیفیت مجموعه آزمون استفاده کرد. به طور خلاصه دقت نتایج گزارش شده در کار آن‌ها از دقت بهترین نتایج گزارش شده در کارهای قبلی بهتر نیست، اما می‌توان روش پیشنهادی آن‌ها را به عنوان مکملی برای روش‌های دیگر به‌کار برد.

اخیراً، آقا محمدی و میریان-حسین آبادی [۵]، برنامه‌های جهش‌یافته‌ای که هیچ آزمونی آن‌ها را اجرا نکرده بود از مجموعه داده مائو و همکاران [۴]، حذف کردند. سپس به کمک ADASYN که روشی برای نمونه‌برداری مجدد افزایشی است داده‌های آموزشی را متوازن کردند. در گام بعد به طور بازگشتی زیرمجموعه‌ای از ویژگی‌ها را مبتنی بر اهمیت جایگشت انتخاب کردند و در نهایت با میانگین گرفتن از خروجی یک مدل جنگل تصادفی و کیسه‌ای از درخت‌های تقویت گرادیان نتیجه کارهای قبل از خود را از نظر AUC به ٪۶۱ بهبود دادند. توجه داشته باشید که آن‌ها عملکرد دسته‌بند پیشنهادیشان را فقط برای برنامه‌های جهش‌یافته پوشش داده شده گزارش کردند و در این حالت AUC کارهای گذشته به شدت کاهش

پیدا کرده و برابر ٪۵۱ می‌شود.

۳- روش پیشنهادی

در شکل ۲ فرآیند آموزش و فرآیند آزمون روش پیشنهادی نشان داده شده است.



شکل ۲: فرآیند آموزش و فرآیند آزمون روش پیشنهادی

مطابق شکل ۲، ابتدا داده آموزش در حافظه اصلی بارگذاری می‌شود. سپس ویژگی‌های دسته‌ای به عددی تبدیل می‌شود. در گام بعد در رابطه با چگونه مدیریت کردن داده‌های از دست رفته تصمیم گیری می‌شود. بیشتر مجموعه داده‌ها شامل نویز و داده‌هایی هستند که بهتر است در مرحله آموزش از آن‌ها استفاده نشود. در گام چهارم قواعدی تعریف شده است که چنین داده‌هایی را در صورت وجود از مجموعه داده حذف می‌کند. سپس در صورت نیاز در گام پنجم نمونه‌برداری مجدد انجام می‌شود. از آن‌جایی که روش‌های موجود نمونه برداری مجدد از محدودیت‌های مساله اطلاع ندارند، ممکن است داده‌هایی تولید کنند که محدودیت‌های مساله را رعایت نکند. در گام ششم، چنین داده‌هایی از مجموعه داده آموزش حذف می‌شود و در گام هفتم، یک دسته‌بند برای پیش‌بینی برنامه‌های جهش‌یافته آموزش داده می‌شود.

در زمان آزمون بعد از بارگذاری داده‌ها در حافظه اصلی و تبدیل ویژگی‌های دسته‌ای به عددی، به کمک جانهی، داده‌های از دست رفته مدیریت می‌شود و سپس پیش‌بینی برنامه‌های جهش‌یافته صورت می‌گیرد. در ادامه مراحل آموزش و آزمون با جزییات بیشتر معرفی می‌شوند.

۳-۱- بارگذاری زیرمجموعه‌ای از داده‌های آموزش

اخیراً آقا محمدی و میریان-حسین آبادی [۵]، انتخاب زیرمجموعه‌ای از ویژگی‌ها را به صورت روش‌مند و موثری انجام دادند. آن‌ها به صورت بازگشتی، ویژگی‌ها را براساس اهمیت جایگشت رتبه‌بندی کردند و ویژگی‌های با اهمیت کمتر را در هر تکرار حذف کردند. در ضمن اگر دو ویژگی همبستگی بیشتر از ٪۹۰ داشتند ویژگی با اهمیت کمتر را حذف کردند. در نهایت ۳۰ ویژگی از ۹۵ ویژگی موجود در مجموعه داده تهیه شده توسط مائو و همکاران [۴] را انتخاب کردند. ویژگی‌های انتخاب شده توسط آن‌ها شامل ۱۳ ویژگی ایستا در سطح پکیج، ۷ ویژگی ایستا در سطح کلاس، ۶ ویژگی ایستا در سطح متد و ۴ ویژگی پویا می‌باشد که در این پژوهش نیز از همان ویژگی‌ها استفاده شده‌است. بنابراین در گام نخست آموزش، تمام سطرهای مجوعه داده و فقط ۳۰ ستون انتخابی بارگذاری می‌شود.

بسیاری از ابزارهای یادگیری ماشین ویژگی‌های دسته‌ای را پشتیبانی نمی‌کنند. علاوه بر این در بسیاری از مواقع که الگوریتم پیاده‌سازی شده این ویژگی‌ها را پشتیبانی می‌کند نیز تبدیل آن‌ها به داده‌های عددی می‌تواند موثر باشد.

در این پژوهش تعدادی از روش‌های معروف تبدیل ویژگی‌های دسته‌ای به عددی امتحان گردید و درنهایت همانند کار آقا محمدی و میریان-حسین آبادی [۵]، روش تبدیل نرخی انتخاب شد که در آن هر ویژگی دسته‌ای با درصد مشاهدات آن دسته جایگزین می‌شود.

۳-۲- مدیریت کردن داده‌های از دست رفته

کارهای گذشته آزمون جهش پیش‌گویانه در این قسمت دچار اشتباه شده‌اند. آن‌ها به اشتباه سطرهای شامل مقادیر از دست رفته را هم از داده‌های آموزش و هم از داده‌های آزمون حذف کردند. این درحالی است که نباید سطرهای شامل مقادیر از دست رفته را از مجموعه آزمون حذف کرد و باید همانند سایر سطرها، بتوان این سطرها را نیز پیش‌بینی کرد.

مطابق کار لین و سای [۸]، زمانی که مجموعه داده شامل درصد کمی مقادیر از دست رفته است (مثلاً در محدوده ۱۰ تا ۱۵ درصد) می‌توان آن‌ها را برای فاز آموزش نادیده گرفت بدون اینکه تاثیر چندانی در دقت نهایی داشته باشند. با در نظر گرفتن این نکته سطرهای شامل مقادیر از دست رفته را از مجموعه داده آموزش حذف می‌کنیم. جدول ۱ اطلاعاتی آماری در رابطه با سطرهای شامل مقادیر از دست رفته در برنامه‌های جهش‌یافته‌ی پوشش داده شده و ۳۰ ویژگی انتخاب شده در مجموعه داده مائو و همکاران [۴] را نشان می‌دهد.

جدول ۱: اطلاعات آماری در رابطه با تعداد سطرهای شامل مقادیر از دست رفته در مجموعه داده مائو و همکاران [۴]

	Train	Validation	Test
The number of mutants	1,198,052	109,676	71,190
The number of mutants with missing data	186,688	42,180	12,714
Percentage of mutants with missing data	15.5%	38.4%	17.8%

در این مرحله علاوه بر تصمیم گیری در رابطه با سطرهای شامل مقادیر از دست رفته در مجموعه داده آموزش، در صورتی که دسته‌بند نهایی چنین داده‌هایی را پشتیبانی نکند، باید در رابطه با چگونگی مدیریت مقادیر از دست رفته برای مجموعه داده آزمون نیز تصمیم گیری کرد. برای این کار تعدادی از روش‌های معروف جانهی بررسی شد و در آخر جایگزین کردن مقادیر از دست رفته با مقدار ثابت صفر روش مناسبی تشخیص داده شد. بنابراین در زمان آزمون، مقادیر از دست رفته با مقدار صفر جایگزین می‌گردد.

۳-۳- فیلتر کردن اول

مجموعه داده‌های جهان واقعی معمولاً شامل نویز و داده‌های پرت می‌شوند. علاوه بر نویز و داده‌های پرت ممکن است انواع خاصی از داده‌ها باشند که حذف آن‌ها به بهبود دقت کمک کند. در این مرحله سعی شده است با مشخص کردن قواعدی ساده بعضی از این نوع داده‌ها در صورت وجود در مجموعه داده آموزش شناسایی و حذف شوند.

آقا محمدی و میریان-حسین آبادی [۵] نشان دادند حذف برنامه‌های جهش‌یافته پوشش داده نشده می‌تواند دقت نهایی را بهبود دهد. بنابراین به عنوان قاعده اول، سطرهایی که مقدار ویژگی numExecuted یا مقدار ویژگی numTestCover آن‌ها کوچک‌تر مساوی صفر باشد از مجموعه داده آموزش حذف می‌شوند.

به عنوان قاعده دوم مطابق پیشنهاد ژانگ و همکاران [۷] سطرهایی که مقدار ویژگی numExecuted آن‌ها کمتر از مقدار ویژگی numTestCover آن‌ها باشد به عنوان نویز حذف می‌شوند. علت این کار این است که اگر برنامه جهش‌یافته‌ای توسط یک مورد آزمون پوشش داده شده باشد، جمله جهش یافته در برنامه نیز حداقل باید یک بار اجرا شده باشد. در نتیجه همواره تعداد دفعات اجرای جمله جهش یافته یک برنامه جهش‌یافته باید بزرگ‌تر مساوی تعداد موارد آزمونی باشد که آن را پوشش داده‌اند.

۳-۴- نمونه برداری مجدد

در این بخش سه روش برای نمونه برداری مجدد به صورت جداگانه بررسی و درنهایت بهترین روش انتخاب شده است .

به عنوان تلاش اول، هیچ روشی برای نمونه‌برداری مجدد در نظر گرفته نشد. به عنوان تلاش دوم مشابه روش پیشنهاد شده در کار ژانگ و همکاران [۲و۳] عمل شده است. فرض کنید تعداد کل برنامه‌های جهش‌یافته مجموعه داده آموزش برابر s باشد. در این روش به تک تک برنامه‌های جهش‌یافته یک پروژه دلخواه با تعداد برنامه جهش‌یافته a، وزن s/a اختصاص داده می‌شود. به عنوان تلاش سوم، از ADASYN که یک روش نمونه برداری مجدد افزایشی است استفاده می‌شود. در نهایت روش ADASYN به عنوان موثرترین روش برای بهبود دقت دسته بند شناسایی و انتخاب شد.

۳-۵- فیلتر کردن دوم

از آن‌جایی که روش‌های نمونه برداری از محدودیت‌های مساله با خبر نیستند و ممکن است داده‌هایی تولید کنند که مطابق نیاز مساله نباشد، در این مرحله مجداداً قواعدی که در گام فیلتر کردن اول بیان شدند، اعمال می‌گردد.

۳-۶- آموزش دسته‌بند

در آخر مطابق کار آقا محمدی و میریان-حسین آبادی [۵]، یک مدل جنگل تصادفی و کیسه‌ای از درختان تقویت گرادیان آموزش داده می‌شود و در زمان پیش‌بینی از میانگین خروجی این دو دسته بند به عنوان خروجی نهایی استفاده می‌گردد.

۳-۷- آزمون

در زمان آزمون، پس از بارگذاری داده های آزمون در حافظه اصلی، ویژگی‌های دسته‌ای با روش تبدیل نرخی که در زمان آموزش برازش شده بود به داده‌های عددی تبدیل می‌گردد. سپس مقادیر از دست رفته در مجموعه آزمون با مقدار ثابت صفر جایگزین می‌شود و در نهایت داده پیش‌پردازش شده به مدل آموزش دیده شده برای پیش‌بینی برنامه‌های جهش‌یافته داده می‌شود.

۴- ارزیابی روش پیشنهادی

در این بخش ابتدا عملکرد روش پیشنهادی با سایر روش‌ها مورد مقایسه قرار می‌گیرد. سپس تاثیر مولفه‌های روش پیشنهادی بررسی می‌شود.

۴-۱- معرفی مجموعه داده

برای انجام آزمایش از مجموعه داده تهیه شده توسط مائو و همکاران [۴] استفاده شده است که شامل ۵۲۲ پروژه برای آموزش، ۶۶ پروژه برای اعتبارسنجی و ۶۶ پروژه برای آزمون می‌باشد. این مجموعه داده به این دلیل انتخاب شده است که اولاً مجموعه داده با کیفیتی است [۷]، ثانیاً اکثر کارهای گذشته نیز از آن استفاده کرده‌اند [۴][۵][۷]. از این رو می‌تواند مقایسه با سایر روش‌ها را ساده تر نماید.

۴-۲- معیارهای ارزیابی

از آن‌جایی که مجموعه داده تهیه شده توسط مائو و همکاران

[۴] نامتوازن است، نیاز است که معیارهایی انتخاب کنیم که نسبت به این عدم توازن جانبداری نداشته باشند [۵]. به همین دلیل در این پژوهش از معیارهای استفاده شده در کار آقا محمدی و میریان-حسین آبادی [۵] یعنی ROC-AUC، Balanced Accuracy(BA) و Matthews Correlation Coefficient(MCC) استفاده شده است.

ROC-AUCمعیاری است که در آن با حرکت دادن آستانه تصمیم گیری سطح زیر نمودار true positive rate و false positive rate محاسبه می‌شود و می‌تواند به طور کلی توانایی دسته بند را نشان دهد. BA تغییر مقیاس داده شده که عددی بین ۱- تا ۱+ را به عنوان خروجی برمی‌گرداند معیاری است که می‌تواند نشان دهد دسته بند چه میزان در یادگیری کلاس مثبت و منفی تعادل ایجاد می‌کند و مطابق رابطه (۱) محاسبه می‌شود که در آن T مخفف True، F مخفف False، P مخفف Positive و N مخفف Negative است:

$$\mathrm{BA} = \frac{TP}{TP+FN} + \frac{TN}{TN+FP} - 1 \qquad (۱)$$

MCC حالت خاصی از Pearson correlation coefficient است که سعی می‌کند تمام اطلاعات موجود در ماتریس درهم ریختگی را در یک عدد بین ۱- تا ۱+ خلاصه کند، به صورتی که هرچه مقدار آن به ۱+ نزدیک تر باشد یعنی توانایی دسته‌بند در پیش‌بینی بهتر بوده است، مقدار ۰ یعنی دسته بند در حد دسته بند تصادفی عمل کرده است و ۱- نشان دهنده اختلاف زیاد بین پیش‌بینی انجام شده از مقدار واقعی مشاهدات است. رابطه (۲) نحوه محاسبه MCC را نشان می‌دهد:

$$\mathrm{MCC} = \frac{(\mathrm{TP}\times\mathrm{TN})+(FP\times FN)}{\sqrt{(\mathrm{TP}+\mathrm{FP})(\mathrm{TP}+\mathrm{FN})(\mathrm{TN}+\mathrm{FP})(\mathrm{TN}+\mathrm{FN})}} \qquad (۲)$$

۴-۳- نتایج مقایسه روش پیشنهادی و سایر روش‌ها

در این بخش دسته‌بندهای روش پیشنهادی آموزش داده شدند، سپس یک به یک پروژه‌های مجموعه آزمون برای پیش بینی به دسته‌بند نهایی داده شد و پیش‌بینی‌های آن ثبت گردید. در نهایت برای هر پروژه، مقدار معیارهای ROC-AUC، BA و MCC محاسبه گردید. جدول ۳، میانگین هر یک از این معیارها را به همراه انحراف از معیار برای تمامی پروژه های مجموعه آزمون گزارش می‌کند.

جدول ۲: ارزیابی کارایی روش پیشنهادی و سایر روش‌ها روی مجموعه داده مائو و همکاران [۴]

Method	Balanced accuracy	Matthews Correlation	ROC-AUC
PMT-95	0.122	0.138	0.540
PMT-12	0.113	0.126	0.539
EPMT	0.230	0.239	0.613
EPMT with missing data handling	0.291±0.02	0.160±0.02	0.697±0.01
Proposed	0.298±0.02	0.186±0.02	0.720±0.01

در جدول ۲، PMT-95 روش مائو و همکاران [۴] را وقتی که انتخاب زیر مجموعه‌ای از ویژگی‌ها انجام نشود، نشان می‌دهد. PMT-12 روش مائو و همکاران [۴] را وقتی از ۱۲ ویژگی پیشنهادی آن‌ها استفاده شود، نشان می‌دهد. EPMT روش پیشنهادی آقا محمدی و میریان-حسین آبادی [۵] را نشان می‌دهد. توجه کنید که سه سطر اول جدول ۲ از نتایج گزارش شده در کار آقا محمدی و میریان-حسین آبادی [۵] نقل شده است و فقط برای داده‌های آزمونی است که شامل مقادیر از دست رفته نمی‌شوند. به عبارت دیگر روش‌های پیشنهادی قبلی توانایی مدیریت مقادیر از دست رفته را نداشتند و چنین برنامه‌های جهش‌یافته‌ای را پیش‌بینی نمی کردند. بنابراین بهتر است سه سطر اول جدول ۲ را حد بالای نتایج کارهای گذشته در نظر گرفت. سطر چهارم زمانی است که روش پیشنهادی آقا محمدی و میریان-حسین آبادی [۵] طوری تغییر داده شده است که بتواند داده‌های شامل مقادیر از دست رفته را مدیریت کند. سطر آخر نتایج روش پیشنهادی این مقاله را نشان می‌دهد.

۴-۴- نتایج تجربی بررسی مولفه‌های روش پیشنهادی

در این بخش مولفه‌های روش پیشنهادی مورد بررسی قرار گرفته‌اند. نتایج این مقایسه ها در جدول ۳ نمایش داده شده‌اند.

جدول ۳: ارزیابی کارایی مولفه‌های روش پیشنهادی

Method	Balanced accuracy	Matthews Correlation	ROC-AUC
Proposed – missing data handling	0.269±0.02	0.156±0.02	0.676±0.01
Proposed – missing data handling, filtering 1	0.291±0.02	0.160±0.02	0.697±0.01
Proposed	0.298±0.02	0.186±0.02	0.720±0.01

در این جدول سطر اول مربوط به روش پیشنهادی است وقتی که دو گام فیلترینگ در آن اعمال نشود و فقط مدیریت کردن داده-های از دست رفته را پشتیبانی کند. در سطر دوم، علاوه بر مدیریت کردن داده‌های از دست رفته، فیلترینگ اول اعمال شده است. توجه داشته باشید از آن‌جایی که در داده‌های آزمون، برنامه جهش‌یافته‌ای وجود نداشت که قاعده دوم ذکر شده در بخش فیلتر کردن اول را نقض کند، سطر دوم جدول ۳ و سطر چهارم جدول ۲ معادل یکدیگرند. در نهایت در سطر سوم جدول ۳ که معادل روش پیشنهادی است، مدیریت کردن داده‌های از دست رفته، فیلترینگ اول و دوم در نظر گرفته شده است.

۵- تهدیدهای اعتبار ارزیابی ها

در این پژوهش از مجموعه داده تهیه شده توسط مائو و همکاران [۴] برای انجام آزمایش‌ها استفاده شد. بنابراین چگونگی تولید شدن برنامه‌های جهش‌یافته، مجموعه‌های آزمون و نحوه محاسبه ویژگی‌ها و ابزارهای استفاده شده توسط آن‌ها می‌تواند در این کار تاثیر گذار باشد. با این حال حداقل سه مقاله دیگر از این کار استفاده کردند [۴][۵][۷] و ژانگ و همکاران [۷] شواهدی مبنی بر با کیفیت بودن این مجموعه داده ارائه دادند.

۶- نتیجه گیری

کارهای گذشته آزمون جهش پیش‌گویانه به اشتباه داده‌های شامل مقادیر از دست رفته را از مجموعه داده آزمون حذف کردند. با در نظر گرفتن این نکته در این پژوهش تاثیر داده‌های از دست رفته در صحت نتایج گزارش شده قبلی و همچنین دقت دسته‌بندی، بررسی و روشی پیشنهاد شد که با مدیریت کردن داده‌های از دست رفته دقت AUC را از ٪۶۱ به ٪۷۲ بهبود داد. علاوه بر این در روش پیشنهادی تاکید ویژه‌ای بر ایده فیلترینگ قبل و بعد از نمونه برداری مجدد صورت گرفت که این نیز یکی از نوآوری‌های این پژوهش است.

برای کارهای آینده می‌توان قواعد فیلترینگ جامع‌تری ایجاد کرد. همچنین می‌توان یک روش جانهی که قواعد فیلترینگ را نقض نکند توسعه داد. به علاوه، می‌توان تاثیر داده‌های تکراری را در آزمون جهش پیش‌گویانه بررسی کرد و روشی برای رفتار مناسب با این نوع داده ها توسعه داد.

مراجع

[1] M. Papadakis, M. Kintis, J. Zhang, Y. Jia, Y. L. Traon, and M. Harman, “Mutation testing advances: An analysis and survey,” in Advances in Computers, pp. 275–378, 2019.

[2] J. Zhang, L. Zhang, M. Harman, D. Hao, Y. Jia, and L. Zhang, “Predictive Mutation Testing,” IEEE trans. softw. eng., vol. 45, no. 9, pp. 898–918, 2019.

[3] J. Zhang, Z. Wang, L. Zhang, D. Hao, L. Zang, S. Cheng, L. Zhang. “Predictive Mutation Testing,” In Proceedings of the 25th International Symposium on Software Testing and Analysis - ISSTA 2016. ACM Press, 2016.

[4] A. : D. Mao, L. Chen, and L. Zhang, “An extensive study on cross-project predictive mutation testing,” in 2019 12th IEEE Conference on Software Testing, Validation and Verification (ICST), 2019.

[5] A. Aghamohammadi and S.-H. Mirian-Hosseinabadi, “An ensemble-based predictive mutation testing approach that considers impact of unreached mutants,” Softw. Test. Verif. Reliab., 2021.

[6] M. R. Naeem, T. Lin, H. Naeem, F. Ullah, and S. Saeed, “Scalable mutation testing using predictive analysis of deep learning model,” IEEE Access, vol. 7, pp. 158264–158283, 2019.

[7] P. Zhang et al., “CBUA: A probabilistic, predictive, and practical approach for evaluating test suite effectiveness,” IEEE trans. softw. eng., pp. 1–1, 2020.

[8] W.-C. Lin and C.-F. Tsai, “Missing value imputation: a review and analysis of the literature (2006–2017),” Artif. Intell. Rev., vol. 53, no. 2, pp. 1487–1509, 2020.