ISSN: 2588-2791
http://www.miej.ir

پژوهش در مدیریت و مهندسی صنایع (سال دهم)
شماره ۴۳ / پاییز ۱۴۰۴ / ص ۱۰- ۱۸

الگوریتم های فرا ابتکاری در سیستم‌های نرم افزاری

محسن نصیری [۱]
[۱] کارشناسی مهندسی کامپیوتر، دانشگاه خبر، ایران.

نام نویسنده مسئول:
محسن نصیری
Mohsen.nasiri23@gmail.com

تاریخ دریافت: ۱۴۰۴/۰۴/۰۱
تاریخ پذیرش: ۱۴۰۴/۰۵/۰۵

چکیده

هدف از انجام این مقاله الگوریتم‌های فرا ابتکاری در سیستم‌های نرم افزاری است. الگوریتم‌های فراابتکاری به عنوان رویکردهایی عمومی و انعطاف‌پذیر برای حل مسائل بهینه‌سازی پیچیده معرفی شده‌اند. این الگوریتم‌ها برای مسائل پیچیده‌ای طراحی شده‌اند که یافتن جواب بهینه با روش‌های دقیق و کلاسیک غیرعملی یا زمان‌بر است. هدف اصلی الگوریتم‌های فراابتکاری، جستجوی کارآمد در فضای حل و یافتن جواب‌های نزدیک به بهینه با هزینه محاسباتی معقول است. ضرورت تلاش برای پیش بینی و کاهش خطا در سیستم‌های نرم افزاری از اهمیت ویژه‌ای برخوردار می‌باشد. با توجه به رشد روز افزون سیستم‌های کامپیوتری در تمامی بخش‌های صنعتی و آموزشی یکی از چالش‌های پیش روی این صنعت قابلیت اطمینان به این سیستم‌ها در زمان‌های مختلف کاری می‌باشد.

واژگان کلیدی: رقابت استعماری، الگوریتم فراابتکاری، کاهش خطا، سیستم نرم افزاری.

پژوهش در مدیریت و مهندسی صنایع، شماره ۴۳، پاییز ۱۴۰۴، ص ۱۰- ۱۸
ISSN: 2588-2791
http://www.miej.ir

مقدمه

سیستم‌های نرم‌افزاری مجموعه‌ای از برنامه‌ها، فرآیندها، و ابزارهایی هستند که به‌صورت یکپارچه برای انجام وظایف مشخصی طراحی شده‌اند. این سیستم‌ها برای پردازش اطلاعات، خودکارسازی فرآیندها، و افزایش بهره‌وری در کسب‌وکارها و زندگی روزمره استفاده می‌شوند.

به‌طورکلی اینترنت، به‌سرعت ما را به سمت پخش‌شدگی یا محیط اطلاعاتی به‌هم پیوسته برده است. در این محیط، داده‌ها درجاهای جغرافیایی گوناگون از هم قرار دارند. این ویژگی انگیزه خواندن و نوشتن داده‌ها را به‌صورت محلی پدید آورده است. درنتیجه دسترس‌پذیری بر روی‌داده‌ها بالا رفته است و در پایان اجرایی سیستم به‌صورت جدی افزایش‌یافته است.از سوی دیگر کمبودهای طراحی و شکاف‌های امنیتی، خطاهایی را در نرم‌افزار به همراه داشته که در پایان این خطاها به شکست‌هایی در کل سیستم منجر خواهد شد. حال اگر خطایی در یکی از تکرارهای داده رخ دهد، به‌احتمال‌زیاد این خطا در تکرارهای دیگر نیز رخ خواهد داد.

در فرآیند تولید نرم افزار، مرحله ی آزمون از پراهمیت ترین و پرهزینه ترین مرحله های نرم افزاری است. یکی از اصلی ترین عملیات در این مرحله، بررسی قطعات نرم افزاری برای یافتن خطاها است. پیش بینی قطعات مستعد خطا، روشی است که برای تعیین قطعاتی که می بایست در مرحله ی آزمون برای یافتن خطاها بررسی شوند، به کار می رود. معیارهایی که معمولاً در این روش استفاده می شود، معیارهای عمومی نرم افزار همانند اندازه ی قطعه، عمق درخت توارث، تعداد دستورات شرطی و غیره هست که اکثراً ویژگی مشترکی که بین قطعات خطادار وجود داشته باشد را نشان نمی دهند. در این مقاله سعی شده بر روی ویژگی هایی از کد که بر روی خطادار شدن قطعات تأثیر می گذارد، تمرکز شود. از عوامل بروز خطا در قطعات، تغییراتی است که بر روی آن ها صورت گرفته است. [۸]

اما این تغییرات تأثیر خود را مستقیم بر روی خود قطعه نشان نمی دهند، بلکه از عوامل بروز خطا در قطعات وابسته هستند.لذا امروزه با توجه به رشد نرم‌افزارها از نظر اندازه و پیچیدگی، حفظ کیفیت بالای محصول نرم‌افزاری، یکی از مهم‌ترین مشکلات پیشروی صنعت نرم‌افزار است.پیش‌بینی‌کننده‌های خطا، ابزارهای مناسب و مقرون به صرفه‌ای برای رفع این مشکل هستند. [۱۴]

ازآنجایی‌که اکثر نقص‌های یک محصول نرم‌افزاری فقط در بخش کوچکی از ماژول‌های آن یافت می‌شوند، با تشخیص زودهنگام مستعد خطا بودن ماژول‌ها، توسعه‌دهندگان نرم‌افزار می‌توانند منابع محدود موجود را برای آزمون دقیق ماژول‌های مستعد خطا تخصیص دهند تا یک نرم‌افزار با کیفیت بالا در زمان مقرر، تولید گردد. به‌عنوان مثال، قبل از آزمون سیستم، شناسایی اجزایی که به احتمال زیاد در حین عملیات باعث بروز خطا می‌گردند، می‌تواند اثربخشی تلاش برای آزمون نرم‌افزار را بهبود بخشد. به همین دلیل روش‌های مختلف مدل‌سازی پیش‌بینی خطای نرم‌افزار توسعه داده‌شده‌اند و به‌طور واقعی برای پیش‌بینی کیفیت نرم‌افزار مورداستفاده قرار می‌گیرند[۲]؛ بنابراین مفهوم اتکاپذیری سیستم‌های نرم‌افزاری مطرح می‌شود. به‌طورکلی اتکاپذیری یعنی، چقدر به سرویسی که سیستم نرم‌افزاری ارائه می‌کند، می‌توان اتکا کرد.

ویژگی‌های اصلی الگوریتم‌های فراابتکاری

۱. انعطاف‌پذیری: الگوریتم‌های فراابتکاری می‌توانند در مسائل مختلف بهینه‌سازی با حداقل تغییرات مورد استفاده قرار گیرند.
۲. استفاده از مکانیزم‌های جستجوی تصادفی: این الگوریتم‌ها معمولاً از فرآیندهای تصادفی برای اکتشاف و بهره‌برداری از فضای جستجو استفاده می‌کنند.
۳. قابلیت ترکیب: الگوریتم‌های فراابتکاری می‌توانند با سایر روش‌های بهینه‌سازی ترکیب شوند تا نتایج بهتری به دست آورند.
۴. کاربردپذیری گسترده: ین الگوریتم‌ها در مسائل مختلفی از جمله مسیریابی، زمان‌بندی، تخصیص منابع و بهینه‌سازی شبکه استفاده می‌شوند.

پژوهش در مدیریت و مهندسی صنایع، شماره ۴۳، پاییز ۱۴۰۴، ص ۱۰- ۱۸
ISSN: 2588-2791
http://www.miej.ir

دسته‌بندی الگوریتم‌های فراابتکاری

۱. الگوریتم‌های مبتنی بر جستجوی محلی

این الگوریتم‌ها با بهبود تدریجی یک جواب اولیه از طریق جستجوی فضای همسایگی عمل می‌کنند. Tabu Search جستجوی ممنوع : با حفظ لیستی از جواب‌های ممنوع از بازدید دوباره نقاطی از فضای جستجو جلوگیری می‌کند. Simulated Annealingبازپخت شبیه‌سازی شده: با شبیه‌سازی فرآیند خنک‌سازی فلزات، احتمال پذیرش جواب‌های بدتر را به تدریج کاهش می‌دهد.

۲. لگوریتم‌های مبتنی بر جمعیت

این الگوریتم‌ها با استفاده از مجموعه‌ای از جواب‌ها (جمعیت) برای جستجو در فضای حل عمل می‌کنند. Genetic Algorithmsالگوریتم‌های ژنتیک(: با الهام از فرآیند تکامل زیستی و استفاده از عملگرهایی مانند تقاطع و جهش. Particle Swarm Optimization)بهینه‌سازی ازدحام ذرات: با الهام از رفتار اجتماعی گروه‌های حیوانی مانند پرندگان. Ant Colony Optimizationبهینه‌سازی کلونی مورچه: با شبیه‌سازی رفتار مورچه‌ها در یافتن کوتاه‌ترین مسیر.

۳. الگوریتم‌های مبتنی بر طبیعت

این الگوریتم‌ها از پدیده‌ها و فرآیندهای طبیعی الهام گرفته‌اند. Harmony Searchجستجوی هماهنگی با الهام از فرآیند تنظیم هماهنگی در موسیقی. Firefly Algorithmالگوریتم کرم شب‌تاب: با شبیه‌سازی رفتار نوری کرم‌های شب‌تاب. Grey Wolf Optimizerبهینه‌سازی گرگ خاکستری با الهام از رفتار شکار گروهی گرگ‌ها

۴. الگوریتم‌های ترکیبی(Hybrid Metaheuristics) این الگوریتم‌ها ترکیبی از روش‌های مختلف فراابتکاری یا کلاسیک هستند.مثال: ترکیب الگوریتم ژنتیک با جستجوی ممنوع برای بهبود بهره‌وری.

کاربردهای الگوریتم‌های فراابتکاری

۱. مسیریابی و لجستیک

- بهینه‌سازی مسیر در مسائل حمل‌ونقل و شبکه‌های توزیع.
- مثال: استفاده از Ant Colony Optimization برای یافتن کوتاه‌ترین مسیر.

۲. زمان‌بندی

- تخصیص منابع به فعالیت‌ها در بازه‌های زمانی معین.
- مثال: استفاده از Genetic Algorithms برای زمان‌بندی تولید.

۳.طراحی مهندسی

- بهینه‌سازی پارامترهای طراحی در سیستم‌های مهندسی.
- مثال: استفاده از Particle Swarm Optimization در طراحی آیرودینامیکی.

۴. یادگیری ماشین

- تنظیم ابرپارامترها و انتخاب ویژگی.
- مثال: استفاده از Simulated Annealing برای تنظیم مدل‌های یادگیری عمیق.

مزایا و معایب الگوریتم‌های فراابتکاری

مزایا:

۱. قابلیت حل مسائل پیچیده و غیرخطی.
۲. عدم نیاز به اطلاعات دقیق درباره ساختار مسئله.
۳. امکان جستجوی در فضای گسترده‌تر و جلوگیری از افتادن در بهینه‌های محلی.

ISSN: 2588-2791
http://www.miej.ir

معایب:

۱. هزینه محاسباتی بالا در برخی الگوریتم‌ها.
۲. وابستگی به تنظیم پارامترها برای عملکرد بهینه.
۳. عدم تضمین یافتن جواب بهینه مطلق.

اجزای اصلی سیستم‌های نرم‌افزاری

۱. **نرم‌افزارهای کاربردی(Application Software):**
   - این نرم‌افزارها برای کاربران طراحی شده‌اند تا وظایف مشخصی مانند مدیریت اسناد، ویرایش تصویر، یا مدیریت مالی را انجام دهند.
   - مثال:Microsoft Word، Google Chrome، Photoshop.
۲. **نرم‌افزارهای سیستمی(System Software):**
   - وظیفه مدیریت منابع سخت‌افزاری و ارتباط بین سخت‌افزار و نرم‌افزار کاربردی را دارند.
   - مثال: سیستم‌عامل‌ها) مانندWindows ، (Linux، درایورها.
۳. **میان‌افزارها(Middleware):**
   - به‌عنوان پل ارتباطی بین نرم‌افزارهای مختلف یا بین سیستم‌های نرم‌افزاری و سخت‌افزاری عمل می‌کنند.
   - مثال: نرم‌افزارهای مدیریت دیتابیس) مثل(Oracle Middleware .
۴. **پایگاه داده‌ها(Databases):**
   - سیستم‌هایی که برای ذخیره‌سازی، مدیریت و بازیابی داده‌ها طراحی شده‌اند.
   - مثال:MySQL، MongoDB، PostgreSQL.
۵. **رابط‌های کاربری(User Interfaces):**
   - بخشی از نرم‌افزار که کاربران با آن تعامل دارند.
   - رابط می‌تواند گرافیکی (GUI) یا متنی (CLI) باشد.

پیشینه های پژوهش

نعیم سلیا [1](۲۰۱۵) برای ارزیابی کیفیت نرم‌افزار و مؤلفه‌های نرم‌افزاری مستعد خطا یک الگوریتم نیمه نظارتی بیشینه‌سازی انتظار، ارائه کردند. پارامترهای الگوریتم نیمه نظارتی آن‌ها مبتنی بر داده‌های مفقود، میانگین و کوواریانس نمونه‌های نرمال بود. آن‌ها از دیتاست JM1 برای آموزش داده‌ها و از دیتاست KC2 برای تست داده‌ها استفاده کردند. دلیل استفاده از دیتاست تست KC2، شباهت داده‌های آن به داده‌های دیتاست آموزشی JM1 است. آن‌ها برای ارزیابی الگوریتم نیمه نظارتی بیشینه‌سازی انتظار، آن را با بیست‌وپنج عدد از الگوریتم‌های با نظارت مقایسه کردند.

پینگ جیو(۲۰۱۴) [2]یک الگوریتم نیمه نظارتی ماشین‌های بردار پشتیبانی هدایتی ارائه کرد که زیربنای پیاده‌سازی آن، الگوریتم‌های ماشین‌های بردار پشتیبانی است. صحت و انعطاف‌پذیری یا استحکام نرم‌افزار دو ویژگی مهم الگوریتم‌های ماشین‌های بردار پشتیبانی می‌باشد. این دو ویژگی، منجر شده که ماشین‌های بردار پشتیبانی، ابزاری قدرتمند برای ارزیابی کیفیت نرم‌افزار در حل مسائل دنیای واقعی باشد. آن‌ها مطرح کردند که ماشین‌های بردار پشتیبانی نسبت به سایر تکنیک‌ها، برای مدل کردن روابط توابع غیرخطی مناسب هستند و درصورتی‌که برای ساخت مدل آموزشی، تعداد نمونه‌های آموزشی کم باشد، ماشین‌های بردار پشتیبانی در ابعاد بالای آموزش داده‌ها، خوب عمل کرده و ساخت مدل با استفاده از ماشین‌های بردار پشتیبانی نسبت به سایر روش‌ها سریع‌تر انجام می‌شود. آن‌ها الگوریتم نیمه نظارتی ماشین‌های بردار پشتیبانی هدایتی را با

[1] Naeem Seliya
[2] PingGuo

ISSN: 2588-2791
http://www.miej.ir

الگوریتم‌های تقریب منظم توابع و الگوریتم‌های درخت تصمیم‌گیری مقایسه کردند. آن‌ها نوع دوم خطا را با استفاده از پارامتر (C) به نام خطای پنالتی ماشین‌های بردار پشتیبانی کنترل کردند و ادعا کردند که الگوریتم پیشنهادی آنها در بسیاری از حوزه‌های نرم‌افزاری قابلیت استفاده را دارد .

راجرز و همکاران(۲۰۱۳) در مقاله ای یک الگوریتم نیمه نظارتی بیشینه‌سازی انتظار ارائه کردند. آن‌ها از دیتاست JM1 برای آموزش داده‌ها و از دیتاست های مستقل دیگر برای تست داده‌های دیده نشده استفاده کردند. آن‌ها برچسب‌های کلاس‌ها را به‌عنوان داده‌های مفقودشده فرض کردند و با استفاده از الگوریتم نیمه نظارتی بیشینه‌سازی انتظار، کلاس هر نمونه دیده نشده را مشخص کردند.

کاتال [1] و دیگران (۲۰۱۰) برای ارزیابی کیفیت نرم‌افزار و شناسایی ماژول‌های نرم‌افزاری مستعد خطا، یک الگوریتم نیمه نظارتی ساده به نام YATSI را ارائه کردند. در ابتدا یک سیستم نرم‌افزاری با کاربرد مشخص معرفی شد. در مرحله دوم، آن‌ها روش به‌کارگیری تصادفی مجموعه‌ای از الگوریتم‌های با نظارت (دسته‌بندی) را معرفی کردند به‌طوری‌که الگوریتم‌های با نظارت به‌صورت مشترک بر روی چندین دیتاست اعمال شدند. در مرحله سوم، عملکرد سیستم نرم‌افزاری را بر مبنای الگوریتم نیمه نظارتی YATSI ارزیابی کردند. سپس روش به‌کارگیری تصادفی مجموعه‌ای از الگوریتم‌های با نظارت را مبتنی بر الگوریتم YATSIارائه کردند. نتایج ارزیابی نشان داد هنگامی الگوریتم YATSI کارایی سیستم نرم‌افزاری را بهبود می‌دهد که کارایی روش به‌کارگیری تصادفی مجموعه‌ای از الگوریتم‌های با نظارت بر روی دیتاست های بدون توازن کاهش یابد. آن‌ها نتیجه گرفتند که کارایی الگوریتم YATSI قابلیت مقایسه با روش به‌کارگیری تصادفی مجموعه‌ای از الگوریتم‌های با نظارت آن الگوریتم‌ها بر اساس تحقیقات زیادی انتخاب می‌شوند را دارد.

کازولا (۲۰۱۱) در تحقیقی یک الگوریتم نیمه نظارتی به نام MSNF برای ارزیابی کیفیت نرم‌افزار (قابلیت اطمینان نرم‌افزار و تست نرم‌افزار) ارائه کرده است. این روش نیازی به دانش قبلی و اوراکلی برای برچسب‌های کلاس ندارد و خطاهای نرم‌افزاری را بدون قاعده تعریف‌شده و به‌صورت پویا در حین اجرای نرم‌افزار، شناسایی می‌کند. این الگوریتم مبتنی بر الگوریتم ماشین‌های بردار پشتیبان می‌باشد.JOHN با استفاده از یک مثال ساده مفاهیم تئوری الگوریتم بیان کرده و در مرحله بعدی تعیین موقعیت خطاهای نرم‌افزاری و شناسایی مؤلفه‌های نرم‌افزاری مستعد خطا، پیچیدگی زمانی و پتانسیل عملیاتی بودن الگوریتم را بررسی کرده است. او در آزمایش‌های انجام‌شده با فایل داده‌های True Music مشخص کرده که دقت شناسایی خطاهای نرم‌افزاری افزایش و پیچیدگی زمانی تست نرم‌افزار کاهش‌یافته است و نتایج رضایت بخشی را در رگرسیون تست و تعیین محل خطاهای نرم‌افزار و شناسایی مؤلفه‌های نرم‌افزاری مستعد خطا به دست آورده است. میزان اطمینان قابلیت‌های عملیاتی الگوریتم MSNF در حوزه‌های عملیاتی گوناگون بررسی نشده‌است و حل مسائل دنیای واقعی مطالعات بیشتری را بر روی برنامه‌های عملیاتی در دامنه‌های دیگر، پیچیدگی و سایر برنامه‌ها نیاز دارد

موسوی (۱۳۹۴) به بررسی تشخیص نفوذ در شبکه‌های کامپیوتری به شیوه یادگیری نیمه نظارتی برخط پرداخته است. با گسترش سریع و روزافزون استفاده از اینترنت در دهه‌های اخیر مبحث امنیت شبکه به یکی از نگرانی‌های اصلی در حوزه فناوری اطلاعات تبدیل‌شده است تکنیک‌های یادگیری ماشین‌ابزار قدرتمندی در حوزه تشخیص نفوذ و حمله به شبکه‌های کامپیوتری به شمار می‌روند در سیستم‌های تحت شبکه داده‌ها به‌صورت تدریجی دریافت می‌شوند و امکان دسترسی به همه داده‌ها به‌طور همزمان وجود ندارد از طرف دیگر به علت کمبود داده‌های برچسب‌دار استفاده از داده‌های بدون برچسب که به تعداد زیاد موجود هستند می‌تواند به پیش‌بینی بهتر بر چسب‌ها کمک کند لذا در این مقاله با فرض قرارگیری داده‌ها بر روی منیفولد با بعد کمتر امکان دسته‌بندی هرچه بهتر داده‌ها با ترکیب دسته‌بندی برخط و یادگیری نیمه‌نظارتی صورت می‌گیرد مقایسه روش پیشنهادی منیفولد برخط با روش‌های پیشین بر روی حجم زیادی از داده NSL-KDD نشان می‌دهد که علاوه بر حفظ دقت نسبت به روش‌های برون‌خط زمان و حافظه موردنیاز برای دسته‌بندی داده ورودی جدید به شبکه کمتر می‌شود .

دشتی و رافع (۱۳۹۳) به بررسی رویداد گرایی در معماری سرویس‌گرا با استفاده از سیستم‌های انتقال گراف پرداخته‌اند.ازآنجاکه سیستم‌های نرم‌افزاری در بسیاری از برنامه‌های کاربردی حساس و بحرانی نقش دارند ممکن است یک

[1] Catal

ISSN: 2588-2791
http://www.miej.ir

خطای کوچک در آن‌ها منجر به بروز مشکلات جدی شود؛ بنابراین به نظر می‌رسد از چالش‌های طراحی سیستم‌هایی بر پایه معماری سرویس‌گرا ایجاد این سیستم‌ها به‌صورت عاری از خطاست که این امر بدون در نظر گرفتن جنبه‌های کیفی همچون قابلیت اطمینان و امنیت در هنگام طراحی و پیاده‌سازی سیستم محقق نخواهد شد. در این مقاله سعی شده با استفاده از یکی از ابزارهای صوری معماری سرویس‌گرا را نرمال کنیم. ازآنجاکه روش‌های صوری بر پایه ریاضی هستند چارچوبی را برای مدل‌سازی و راستی آزمایی سیستم‌های نرم‌افزاری مبتنی بر معماری سرویس‌گرا فراهم می‌کنند که بتوان آن‌ها را به‌طور صحیح مدل‌سازی، پیاده‌سازی و ارزیابی کرد.

دهقانی و همکاران (۱۳۹۳) به بررسی ارائه روشی جدید جهت ارزیابی قابلیت استفاده مجدد مؤلفه‌های نرم‌افزاری پرداخته‌اند. تقاضا برای برنامه‌های کاربردی نرم‌افزاری جدید رو به افزایش است ولی تعداد متخصصان باتجربه موردنیاز برای این کار در حال افزایش نیستند؛ استفاده مجدد به‌عنوان یک ابزار قدرتمند برای غلبه بر بحران نرم‌افزار است.لذا ما قصد داریم تا با استفاده از ترکیب این سه روش و استفاده از ابزار جدید، بتوانیم معایبی که در روش‌های قبل وجود داشت را برطرف نمائیم. چون استنباط قوانین و استخراج مجموعه اولیه این قوانین برای سیستم فازی کار دشواری است بنابراین پیشنهاد می‌شود الگوریتم درخت تصمیم‌گیری برای انتخاب مجموعه اولیه قوانین فازی مورداستفاده قرار گیرد. همچنین با کمک روش شبکه عصبی می‌توانیم قوانین استخراج‌شده فوق را بهبود بخشیده و یا اصلاح نمائیم که درنتیجه موجب ایجاد یک سیستم فازی کارآمد جهت ارزیابی قابلیت استفاده مجدد از مؤلفه‌های نرم‌افزاری می‌شود.

نتایج

سیستم‌های نرم‌افزاری به‌عنوان یک بخش اساسی از دنیای مدرن، نقش کلیدی در بهبود کیفیت زندگی و ارتقای کارایی در تمامی جنبه‌های زندگی انسان‌ها ایفا می‌کنند. این سیستم‌ها با ترکیب فناوری‌های پیشرفته، توانسته‌اند فرآیندهای پیچیده را ساده‌تر کرده و دسترسی به اطلاعات و منابع را سریع‌تر و دقیق‌تر کنند.

یکی از مهم‌ترین دستاوردهای سیستم‌های نرم‌افزاری، افزایش اتوماسیون و کاهش نیاز به مداخلات انسانی در وظایف تکراری است که به صرفه‌جویی در زمان و هزینه منجر می‌شود. همچنین، نرم‌افزارهای پیشرفته، امکان مدیریت بهتر داده‌ها و تصمیم‌گیری‌های مبتنی بر اطلاعات دقیق‌تر را فراهم می‌کنند.

با وجود این مزایا، توسعه و نگهداری سیستم‌های نرم‌افزاری با چالش‌هایی نظیر پیچیدگی، امنیت، و نیاز به پشتیبانی مداوم مواجه است. اطمینان از پایداری، مقیاس‌پذیری، و امنیت این سیستم‌ها برای موفقیت و پذیرش آن‌ها در بلندمدت ضروری است. طراحی دقیق و مدیریت هوشمند این سیستم‌ها می‌تواند از بروز مشکلات پیشگیری کند.همچنین، گسترش فناوری‌های نوین نظیر هوش مصنوعی، یادگیری ماشین، و اینترنت اشیا (IoT) فرصت‌های جدیدی برای توسعه سیستم‌های نرم‌افزاری ایجاد کرده است. این سیستم‌ها اکنون نقش بیشتری در اتصال افراد، کسب‌وکارها و دستگاه‌ها ایفا می‌کنند. در نهایت، سیستم‌های نرم‌افزاری می‌توانند ابزاری قوی برای پیشرفت جامعه باشند، به شرط آنکه طراحی آن‌ها مبتنی بر نیازهای واقعی کاربران و با رعایت اصول اخلاقی و امنیتی صورت گیرد. آینده این حوزه پر از فرصت‌ها برای بهبود زندگی انسان‌ها و ایجاد دنیایی هوشمندتر خواهد بود.سیستم‌های نرم‌افزاری جهت توسعه با چالش هایی مواجه است که در زیر به ان اشاره شده است. پیچیدگی: با بزرگ‌تر شدن سیستم، پیچیدگی نیز افزایش می‌یابد. مدیریت منابع: اطمینان از بهینه‌سازی منابع سخت‌افزاری و نرم‌افزاری. تضمین کیفیت: اطمینان از عملکرد صحیح سیستم در تمامی شرایط. امنیت: مقابله با تهدیدات سایبری.

ISSN: 2588-2791
http://www.miej.ir

منابع

۱) عسکری، محمدمهدی؛ خطیبی بردسیری، وحید.(۱۳۹۳). ارائه یک مدل ترکیبی هوشمند به‌منظور پیش‌بینی نقص نرم‌افزار. همایش ملی مهندسی و علوم کامپیوتر ، ۳ آذر.

۲) علی قار داشی، فاطمه(۱۳۹۳). پیش‌بینی خطا در نرم‌افزار مبتنی بر روش‌های یادگیری ماشین. سمینار کارشناسی ارشد .

۳) آرزو موسوی،" تشخیص نفوذ در شبکه‌های کامپیوتری به شیوه یادگیری نیمه نظارتی برخط "، کنفرانس بین‌المللی یافته‌های نوین پژوهشی در مهندسی برق و علوم کامپیوتر، ۱۳۹۴.

۴) رضا عزمی، بشری پیشگو، نرگس نوروزی و محمدرضا نیک پور، " ارائه یک چارچوب تجمیعی در یادگیری به روش نیمه نظارتی"، شانزدهمین کنفرانس بین‌المللی سالانه انجمن کامپیوتر ایران،۱۳۸۹

۵) انسیه دهقانی، مهدیه قزوینی و حجت بابایی، " ارائه روشی جدید جهت ارزیابی قابلیت استفاده مجدد مؤلفه‌های نرم‌افزاری"، همایش الکترونیکی پژوهش‌های نوین در علوم و فناوری،۱۳۹۳.

۶) رؤیا دشتی و وحید رافع، " بررسی رویداد گرایی در معماری سرویس گرا با استفاده از سیستم‌های انتقال گراف"، کنفرانس ملی ایده‌های نو علوم مهندسی، ۱۳۹۳.

7) Sommerville.Ian, "software Engineering," Wiley,6th Edition,2000.

8) Avizˇienis. Algirdas, Laprie. Jean-Claude. Randell. Brian," Fundamental Concepts of Dependability, " university of New Castel,2001.

9) Koren. Israel, Krishna. C.Mani, "fault tolerant system," Morgan Kaufmann Publishers in an imprint of Elsevier, 2007.

10) Eusgeld. Irene, Fraikin. Falk, Rohr.Matthias, Salfner Felix, " Software Reliability," Springer-Verlag Berlin Heidelberg, 2008, Dependability Metrics, LNCS 4909, pp. 104–125.

11) Oliver Chapell، Bernhard Schölkopf and Alexander Zien. Semi-Supervised Learning. Massachusetts Institute of Technology2006.

12) X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, Department of Computer Sciences, University of Wisconsin, Madison, 2005.

13) J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum, “YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia,” Artif. Intell., vol. 194, pp. 28–61, Jan. 2013.

14) Maengsik Choi, Harksoo Kim,"Social relation extraction from texts using a support- vector-machine-based dependency trigram kernel", Information Processing & Management, Vol.49, pp. 303-311, 2013.

15) Federica Bisio, Sergio Decherchi, Paolo Gastaldo, Rodolfo Zunino,"Inductive bias for semi-supervised extreme learning machine",Neurocomputing, Vol.174, pp. 154-167, 2016.

16) Andrew Carlson," Coupled Semi-Supervised Learning for Information Extraction", Computer Science Carnegie Mellon, 2010.

17) T. Khoshgoftaar and N. Seliya, “Comparative Assessment of Software Quality Classification Techniques : An Empirical Case Study,” Empir. Softw. Eng., vol. 9, no. 3, pp. 229–257, 2114.

18) K. Dejaeger, T. Verbraken, and B. Baesens, “Toward Comprehensible Software Fault Prediction Models Using Bayesian Network Classifiers,” IEEE Trans. Softw. Eng., vol. 39, no. 2, pp. 237–257, 2113.

ISSN: 2588-2791
http://www.miej.ir

19) Seliya. Naeem, Khoshgoftar. Taghi M,"Semi-supervised for software Quality Estimation," IEEE,2004.

20)]T. M. Khoshgoftaar ,S. Zhong, N. Seliya. "Analyzing software measurement data with clustering techniques," IEEE Intelligent Systems, vol. 19, no. 2, pp. 20-27, 2004.

21) Seliya. Naeem, Khoshgoftar. Taghi M,"Analyzing Software Quality with Limited Fault-Proneness Defect Data," IEEE,2005.

22) Xing. Fei, Guo. Ping, R. Lyu .Michael ," A Novel Method for Early Software Quality Prediction

23) Based on Support Vector Machine," Computer.ORG.2005.

24) Seliya, N. and Khoshgoftaar, T.M., "Software Quality Analysis of Unlabeled Program Modules with Semisupervised Clustering," IEEE Transactions on Systems, Man And Cybernetics-Part A: Systems And Humans, 37(2), 201-211, 2007.