دانشگاه پیام نور

مرکز نجف‌آباد

مقاله تحقیق درس سمینار

رشته مهندسی کامپیوتر

گرایش هوش مصنوعی و رباتیکز

عنوان

کاربرد هوش مصنوعی در آزمون نرم‌افزار: تولید آزمون واحد با یادگیری ماشین

استاد

دکتر مجید ایران‌پور مبارکه

دانشجو

صدرا صمدی طاهرگورابی

پاییز ۱۴۰۲

چکیده

در این مقاله می‌خواهیم کاربرد هوش مصنوعی در آزمون نرم‌افزار را بررسی کنیم، با تمرکز بر تولید آزمون واحد توسط تکنیک‌های یادگیری ماشین. با تکیه بر یادگیری sequence-to-sequence در شبکه‌های عصبی، یک مدل seq2seq را برای تولید خودکار توابع آزمون برای توابع ورودی داده شده ارائه می‌کنیم [٤]. این مدل (که با استفاده از TensorFlow پیاده شده) درنظر دارد تا راه‌حلی برای چالش‌های شناسایی شده در شیوه‌های فعلی آزمون نرم‌افزار ارائه دهد (به ویژه شیوه‌های دستی نوشتن آزمون برای نرم‌افزار) [٢].

رویکرد ما از شبکه‌های LSTM (حافظه کوتاه‌مدت بلندمدت) بهره می‌برد، با الهام از اثربخشی LSTMها در حل مسائل پیچیده‌ی یادگیری دنباله‌ها، مانند ترجمه ماشینی [٥]. این روش شامل یک اسکریپت برای تولید توابع ریاضی و توابع آزمون واحد نظیر آن‌ها، پیش‌پردازش داده‌ها، ایجاد یک واژگان از کاراکترها در مجموعه داده، ایجاد یک لایه جاسازی کلمه (Word Embedding)، و آموزش مدل seq2seq شامل LSTM کدگذار (Encoder) و LSTM کدگشا (Decoder) است.

این مقاله عملکرد مدل را ارزیابی می‌کند و پیامدهای آن را برای حوزه آزمون نرم‌افزار مورد بحث قرار می‌دهد. هدف ما این است که شکاف بین ابزارهای فعلی آزمون خودکار و رفتار هوشمندانه و ظریف آزمایشگرهای انسانی را پر کرده و در نهایت به فرآیندهای تست نرم‌افزار کارآمدتر و دقیق‌تر کمک کنیم.

کلیدواژه‌ها: هوش مصنوعی در آزمون نرم‌افزار، یادگیری ماشین، تولید تست واحد، شبکه‌های LSTM، مدل seq2seq، فریمورک TensorFlow، ابزارهای آزمون خودکار.

مقدمه

در زمینه‌ی روبه‌رشد آزمون نرم‌افزار، اطمینان از قابلیت اعتماد و کارایی کد به یک امر مهم تبدیل شده است. پیاده‌سازی آزمون واحد، متدی که در آن اجزاء یک نرم‌افزار به منظور تشخیص صحت عملکرد بررسی می‌شوند، نقش مهمی را در این پروسه ایفا می‌کند. با این حال، روش‌های مرسوم آزمون واحد اغلب نیازمند اجرای دستی است، که به افزایش زمان توسعه و خطای انسانی منجر می‌شود. با ظهور هوش مصنوعی (AI) و یادگیری ماشین (ML)، علاقه شدیدی به خودکارسازی این فرآیند جهت افزایش کارایی و دقت به وجود آمده است.

این مقاله به کاربرد هوش مصنوعی در آزمون نرم‌افزار می‌پردازد، به‌خصوص با تمرکز بر تولید توابع آزمون واحد توسط روش‌های یادگیری ماشین. انگیزه این کار از نیاز به ساده‌سازی فرآیند آزمون نرم‌افزار و کاهش حجم فعالیت‌های دستی توسعه دهندگان و آزمایش‌کنندگان نشأت می‌گیرد. با استفاده از قدرت هوش مصنوعی به ویژه مدل‌های یادگیری دنباله–به–دنباله (sequence-to-sequence)، هدف ما خودکارسازی تولید آزمون واحد برای توابع مختلف نرم‌افزاری است.

تحقیق ما بر اساس مبانی تولید متن و یادگیری دنباله‌ای مبتنی بر هوش مصنوعی است. پیشرفت در معماری شبکه‌های عصبی، مثل شبکه حافظه کوتاه‌مدت بلندمدت (LSTM)، فناوری زیربنایی را برای رویکرد ما فراهم می‌کند [٤].

علاوه بر آن، ایده‌ی یادگیری با شبکه‌های عصبی بازگشتی کدگذار-کدگشا [۵]، بر توانایی این شبکه در مدیریت دنباله‌های پیچیده تأکید می‌کند، که یک ویژگی مهم برای تولید آزمون‌های واحد منسجم و مرتبط به شمار می‌رود.
جایگاه فعلی هوش مصنوعی در آزمون نرم‌افزار، دارای شکافی بین قابلیت‌های آزمون‌های ماشینی و درک دقیق آزمایش کننده‌های انسانی است. هدف از انجام این تحقیق پر کردن شکاف ذکر شده توسط ایجاد یک مدل seq2seq است که نه‌تنها نحو (syntax) موارد آزمایشی را می‌آموزد، بلکه معنای (semantic) پشت آن‌ها را نیز درک می‌کند.
در این مقاله، روش پیاده‌سازی یک مدل seq2seq که جهت تولید خودکار توابع آزمون واحد طراحی شده را با استفاده از TensorFlow شرح خواهیم داد. ما چالش‌های موجود را بررسی کرده، کارایی مدل را با سناریوهای مختلف ارزیابی می‌کنیم و تأثیر بالقوه آن بر آینده آزمون نرم‌افزار را مورد بحث قرار می‌دهیم.
هدف ما مشارکت در زمینه آزمون نرم‌افزار با معرفی یک رویکرد مبتنی بر هوش مصنوعی است که دقت و کارایی تولید آزمون واحد را بهبود می‌بخشد و راه را برای فرآیندهای قابل اعتمادتر و راحت‌تر توسعه نرم‌افزار هموار می‌کند.

روش‌ها

نمای کلی مدل

روش ما بر توسعه یک مدل seq2seq به منظور تولید خودکار آزمون‌های واحد برای توابع ورودی داده شده متمرکز است. این مدل بر پایه شبکه های LSTM (حافظه کوتاه‌مدت بلندمدت) ساخته شده، که به دلیل کارایی آن‌ها در مدیریت داده‌های دنباله‌ای و یادگیری ارتباطات بلندمدت انتخاب شده‌اند [٤] [۵].

تولید و پیش‌پردازش داده‌ها

مبنای رویکرد ما یک اسکریپت برای تولید داده‌های سفارشی است که مجموعه داده‌های متنوعی از توابع ساده ریاضی و توابع آزمون واحد نظیر آن‌ها را ایجاد می‌کند. این اسکریپت طیف وسیعی از اسامی توابع و آزمون آن‌ها را پوشش می‌دهد، که شامل عملیات حسابی مختلف مثل جمع، تفریق، ضرب و تقسیم می‌شود. تنوع در مجموعه داده‌ها به وسیله انتخاب‌های تصادفی اسامی از پیش تعریف شده برای توابع، آزمون‌ها و جفت پارامتر ورودی آن‌ها به دست می‌آید، که دامنه گسترده‌ای از سناریوها را تضمین می‌کند.
مجموعه داده تحت یک مرحله پیش‌پردازش قرار می‌گیرد تا آن را به قالبی مناسب برای یادگیری تبدیل کند. این مرحله شامل جداسازی واژگان (Tokenization) کدهای تابع و آزمون و تبدیل آن‌ها به دنباله‌ای از اعداد صحیح است. ما همچنین از لایه‌گذاری (Padding)، برای اطمینان از یکنواخت بودن طول دنباله‌ها در سرتاسر مجموعه داده استفاده می‌کنیم، که برای آموزش موثر مدل ما ضروری است.

معماری مدل

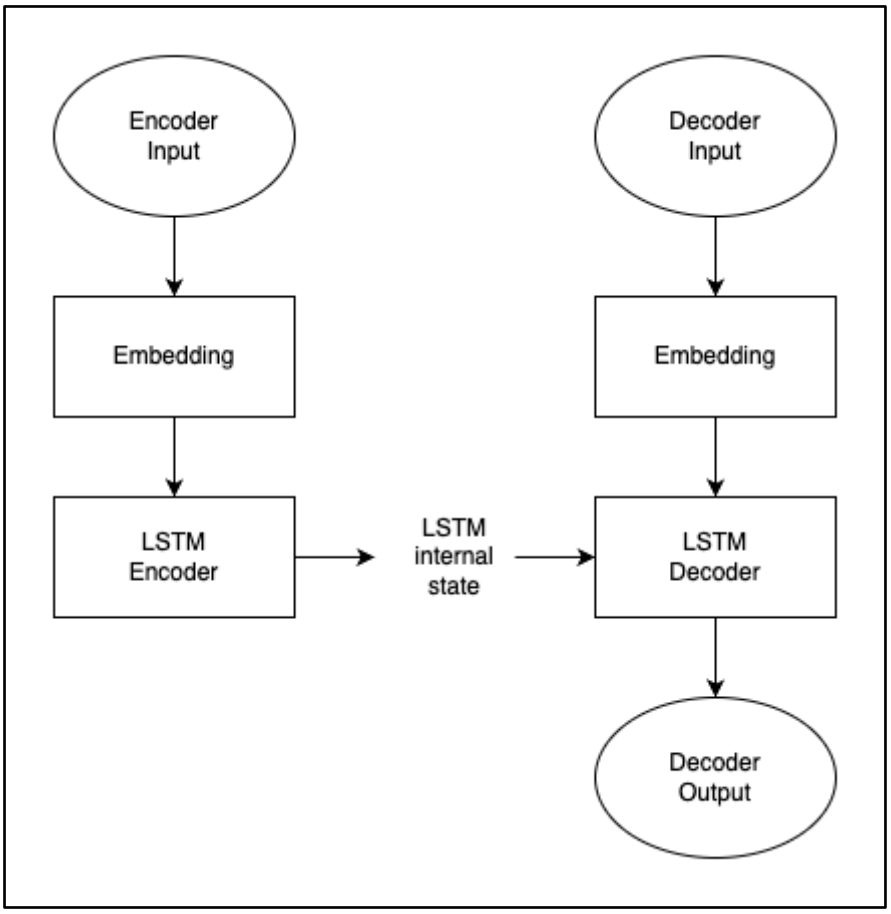
مدل seq2seq ما یک معماری شبکه عصبی را با استفاده از TensorFlow به کار می‌برد و بر اساس شبکه‌های LSTM (حافظه کوتاه‌مدت بلندمدت) ساخته شده است. این مدل شامل دو جزء اصلی است:

۱. **کدگذار (Encoder)**: کدگذار LSTM دنباله ورودی (توابع ریاضی) را پردازش کرده و آن را در یک بردار زمینه (Context Vector) کدگذاری می‌کند. این بردار اطلاعات مهم و زمینه را از دنباله ورودی دریافت می‌کند. LSTM به گونه‌ای پیکربندی شده که وضعیت‌های داخلی خود را بازگرداند تا اطلاعات دنباله به طور موثر دریافت شود.

۲. **کدگشا (Decoder)**: کدگشا LSTM بردار زمینه از کدگذار را به عنوان وضعیت داخلی اولیه خود استفاده می‌کند، که تولید دنباله خروجی را ممکن می‌سازد (یعنی تابع آزمون واحد متناظر). این بخش به گونه‌ای طراحی شده تا دنباله خروجی را گام‌به‌گام تولید کرده و هربار یک توکن را پیش‌بینی کند.

یک لایه جاسازی (Embedding) در ورودی کدگذار و کدگشا نیز گنجانده شده است. این لایه توکن‌های ورودی را به بردارهای متراکم با طول ثابت تبدیل می‌کند که توانایی مدل برای دریافت ارتباط معنایی بین داده‌ها را بسیار بهبود می‌بخشد.

تصویر زیر معماری مدل را به شکل ساده نشان می‌دهد:



فرآیند آموزش

مدل با استفاده از مجموعه داده‌هایی از جفت تابع و آزمون آموزش داده می‌شود. ما مدل را با استفاده از بهینه‌ساز RMSProp و تابع زیان Sparse Categorical Cross-entropy تنظیم می‌کنیم، که در مسائل مربوط به پیش‌بینی دنباله‌ها استفاده شده و مناسب‌تر هستند.

مرحله آموزش شامل خوراندن داده‌های ورودی کدگذار و داده‌های ورودی کدگشا به مدل و آموزش آن برای پیش‌بینی داده‌های خروجی کدگشا است. ما بخشی از مجموعه داده‌ها را به عنوان مجموعه اعتبارسنجی برای نظارت بر عملکرد و جلوگیری از بیش‌برازش مدل استفاده می‌کنیم.

مدل استنتاج

برای تولید آزمون‌های واحد جدید، ما از مدل‌های استنتاج جداگانه‌ای برای کدگذار و کدگشا استفاده می‌کنیم. مدل کدگذار یک تابع ورودی را در یک بردار زمینه کدگذاری کرده، و مدل کدگشا آزمون واحد متناظر را با استفاده از بردار زمینه برای حفظ اطلاعات دنباله تولید می‌کند.

ارزیابی

ما عملکرد مدل را براساس توانایی آن در تولید توابع آزمون واحد که از نظر نحوی و معنایی صحیح و همسو با توابع ریاضی ورودی هستند ارزیابی می‌کنیم. این ارزیابی شامل اندازه‌گیری دقت (Accuracy) و زیان مدل بر روی مجموعه داده‌های آزمایشی است که یک معیار کمی از عملکرد آن را ارائه می‌دهد.

به طور خلاصه، این روش یک رویکرد جامع به سمت خودکارسازی تولید آزمون واحد توسط یک مدل seq2seq را ترسیم می‌کند. این فرآیند از تولید داده‌های اولیه و پیش‌پردازش تا ساخت و آموزش دقیق مدل مبتنی بر LSTM را در بر می گیرد. مدل‌های استنتاج که برای پیش‌بینی استفاده شد، بر کاربرد عملی این تحقیق در تولید آزمون‌های واحد تأکید می‌کند. این بخش مراحل و پیکربندی‌های اساسی را نشان می‌دهد که توانایی مدل ما را برای یادگیری و تولید آزمون‌های واحد مرتبط ایجاد می‌نماید.

نتایج

در این بخش، عملکرد و نتایج مدل seq2seq که به منظور تولید خودکار توابع آزمون واحد برای عملیات ریاضی طراحی شده است را مورد بحث قرار می‌دهیم. ارزیابی این مدل هم نقاط قوت و هم زمینه‌هایی که نیاز به بهبود دارند را نشان داده و چشم‌انداز ارزشمندی را در مورد کاربرد هوش مصنوعی در آزمون نرم‌افزار ارائه می‌دهد.

معیارهای عملکرد مدل

مدل seq2seq ما به سطح بالایی از دقت در تولید توابع آزمون واحد دست یافته است. معیار اولیه برای ارزیابی کارایی مدل، قابلیت آن در ساخت آزمون‌های واحدی است که از نظر نحوی صحیح و از نظر منطقی مرتبط هستند. اگرچه مدل دقت بالایی را در اکثر موارد نشان می‌دهد، اما موارد قابل توجهی از عدم دقت در محاسبات ریاضی نیز در توابع آزمون تولید شده وجود دارد.

صحت نحوی و ارتباط معنایی

مدل در تولید آزمون‌های واحد از نظر صحت نحوی عملکرد خوبی دارد. هر خروجی به ساختار استاندارد آزمون واحد که شامل تعریف مناسب توابع و عبارات ادعا (Assertion) می‌شود پایبند است. از نظر ارتباط معنایی، مدل

به طور کلی موفق به تطبیق عمل ریاضی در تابع ورودی با یک تابع آزمون واحد مناسب می‌شود. این قابلیت نشان‌دهنده‌ی درک قوی از ارتباط بین توابع ورودی و نیازمندی‌های آزمون متناظر با آن‌ها است.

مسائل مربوط به محاسبات

یکی از مشاهدات مهم، عدم دقت گاه‌به‌گاه محاسبات ریاضی در آزمون‌های تولید شده است. به طور مثال، مدل ممکن است ادعا کند (Assert) که حاصل تفریق عدد ٦٤ و ٥٠ برابر با ١٢ است، که بسیار نزدیک اما از نظر حسابی نادرست است. این خطاها یک محدودیت در قابلیت استدلال عددی فعلی مدل را برجسته می‌کند. اگرچه آزمون‌های تولید شده از نظر نحو و ساختار معتبر هستند، اما صحت منطقی از نظر محاسبات عددی گاهی دچار کاستی می‌شود.

تنوع در خروجی

مدل توانایی قابل‌توجهی در مدیریت انواع مختلف عملیات حسابی نظیر جمع، تفریق، ضرب و تقسیم از خود نشان می‌دهد. این تنوع در خروجی نشان‌دهنده‌ی قابلیت تعمیم و سازگاری مدل در میان انواع مختلف توابع ریاضی است.

ارزیابی آزمون‌های تولید شده

هنگام ارزیابی آزمون‌های تولید شده، ما هم ساختار نحوی و هم صحت منطقی ادعاها (Assertion) را درنظر گرفته‌ایم. این ارزیابی‌ها نشان می‌دهد، درحالی که مدل در فهم و اعمال ساختار آزمون‌های واحد توانا است، اما برای اطمینان از دقت محاسبات ریاضی در این آزمون‌ها به اصلاح بیشتری نیاز دارد.

یک نمونه از خروجی برنامه را می‌توان در ادامه مشاهده کرد:

```
[Evaluation] Loss: 0.187 | Accuracy: %92.86
--------------------------------------------------------------------------------
[Input ] def subtract_numbers(number_a, number_b): return number_a - number_b
[Output] def test_subtraction(): assert subtract_numbers(64, 50) == 12
--------------------------------------------------------------------------------
[Input ] def calculate_sum(a, b): return a + b
[Output] def test_addition(): assert accumulate(82, 64) == 140
--------------------------------------------------------------------------------
[Input ] def plus(num1, num2): return num1 + num2
[Output] def test_addition(): assert plus(21, 100) == 119
--------------------------------------------------------------------------------
[Input ] def append(first, second): return first + second
[Output] def test_addition(): assert append(22, 23) == 45
--------------------------------------------------------------------------------
[Input ] def amplify(value_1, value_2): return value_1 * value_2
[Output] def test_multiplying_two_numbers(): assert amplify(64, 55) == 3060
--------------------------------------------------------------------------------
[Input ] def divide_numbers(number_a, number_b): return number_a / number_b
[Output] def test_divide_operation(): assert divide_numbers(8, 93) == 0
--------------------------------------------------------------------------------
[Input ] def calculate_quotient(x, y): return x / y
[Output] def test_quotient(): assert calculate_quotient(16, 50) == 0
--------------------------------------------------------------------------------
[Input ] def take_away(a, b): return a - b
[Output] def test_subtracting_two_numbers(): assert take_away(66, 83) == -17
--------------------------------------------------------------------------------
[Input ] def calculate_quotient(x, y): return x / y
[Output] def test_divide_operation(): assert partition(57, 63) == 0
--------------------------------------------------------------------------------
[Input ] def replicate(a, b): return a * b
[Output] def test_multiplying_two_numbers(): assert replicate(10, 50) == 400
--------------------------------------------------------------------------------
```

به طور خلاصه، این بخش یک تجزیه و تحلیل دقیقی از عملکرد مدل در تولید توابع آزمون واحد ارائه داده است. درحالی که مدل کارایی خود در ساخت آزمون‌هایی با صحت نحوی و ارتباط معنایی را نشان می‌دهد، محدودیت‌هایی در محاسبه دقیق مقادیر عددی را نیز نمایان می‌کند. این یافته‌ها هم پتانسیل و هم چالش‌ها در استفاده از هوش مصنوعی برای تولید خودکار آزمون واحد را برجسته می‌کنند، و یک مسیر مشخص جهت بهبود در دقت و قابلیت‌های استدلالی مدل در آینده ارائه می‌دهند.

بحث و بررسی

این بخش به تفسیر، پیامدها و زمینه‌های گسترده‌تر یافته‌های ارائه شده در بخش "نتایج" می‌پردازد، و اهمیت و تأثیر بالقوه مدل seq2seq در زمینه تولید خودکار آزمون واحد را مورد بحث قرار می‌دهد.

تفسیر یافته‌ها

عملکرد مدل seq2seq ما، به ویژه در تولید آزمون‌های واحدی با نحو صحیح و معنای مرتبط، امکان استفاده از هوش مصنوعی در آزمون نرم‌افزار را نشان می‌دهد. توانایی مدل در نگاشت دقیق توابع ریاضی به آزمون‌های واحد نظیر، درک قابل توجهی از روابط بین توابع ورودی و نیازمندی‌های آزمون آن‌ها را مشخص می‌کند. با این حال، عدم دقت گاه‌به‌گاه در منطق محاسبات آزمون‌ها به محدودیت در قابلیت استدلال عددی مدل اشاره می‌کند.

قیاس با رویکردهای موجود

در مقایسه با روش‌های مرسوم تولید آزمون واحد، که اغلب شامل پیاده‌سازی‌های دستی و زمان‌بر می‌شود، رویکرد ما نشان‌دهنده‌ی پیشرفت قابل توجهی در خودکارسازی این فرآیند است. اگرچه مدل فعلی به طور کامل نیاز به نظارت انسانی را از بین نمی‌برد، به خصوص در اعتبارسنجی صحت منطقی آزمون‌ها، اما تلاش اولیه موردنیاز در ایجاد موارد آزمون را به طور قابل توجهی کاهش می‌دهد.

پیامدها در آزمون نرم‌افزار

دقت بالای مدل در تولید آزمون‌های واحد برای عملیات حسابی پایه، کاربرد بالقوه آن در سناریوهای پیچیده‌ی آزمون را نشان می‌دهد. توانایی آن در مدیریت عملیات متنوع و تولید طیف گسترده‌ای از موارد آزمون می‌تواند به طور قابل توجهی فرآیند آزمودن در توسعه نرم‌افزار را ساده کند، و به طور بالقوه منجر به محصولات نرم‌افزاری کارآمدتر و قابل اطمینان‌تر شود.

چالش‌ها و محدودیت‌ها

یکی از چالش‌های اصلی که توسط تحقیقات ما برجسته شده است، تقلای گاه‌به‌گاه مدل در محاسبات عددی دقیق در موارد آزمون است. این محدودیت بر نیاز به اصلاح بیشتر در معماری مدل (شاید با ادغام تکنیک‌های پیشرفته‌تر برای استدلال و درک عددی) تأکید می‌کند. علاوه بر آن، تمرکز فعلی مدل بر توابع ساده حسابی ممکن است کاربرد آن را در سناریوهای پیچیده‌تر دنیای واقعی محدود کند.

دستورالعمل‌های آینده

با تکیه بر یافته‌های این مطالعه، تحقیقات آینده می‌تواند چندین مسیر را بررسی کند:

۱. **تقویت استدلال عددی**: لحاظ نمودن اجزاء یا تکنیک‌هایی که قابلیت مدل را در فهم و محاسبه دقیق مقادیر عددی بهبود می‌بخشد.
۲. **گسترش دامنه**: تطبیق مدل برای مدیریت انواع پیچیده‌تر و متنوع‌تر توابع، که فراتر از عملیات حسابی پایه هستند.
۳. **یکپارچه سازی با محیط‌های توسعه**: توسعه ابزار یا افزونه‌هایی که این فرآیند تولید آزمون مبتنی بر هوش مصنوعی را با محیط‌های توسعه نرم‌افزار برای کمک به آزمودن در موارد واقعی یکپارچه می‌کند.

بینش‌های به دست آمده از مطالعات ما، مسیر را برای تحقیقات آینده در حوزه آزمون نرم‌افزار مبتنی بر هوش مصنوعی هموار خواهد کرد. کارآیی مدل به عنوان یک معیار برای پیشرفت در این زمینه عمل می‌کند. مطالعات آینده می‌تواند بر افزایش دقت محاسباتی مدل و گسترش کاربرد آن در طیف وسیع‌تری از سناریوهای برنامه‌نویسی تمرکز کند. این تحقیق گامی به سوی عصر جدیدی در آزمون نرم‌افزار است، جایی که در آن هوش مصنوعی نه تنها فرآیند را تقویت بلکه در آن نوآوری کرده، و مرزهای آنچه در حال حاضر در روش‌های آزمون خودکار قابل دسترس است را جابه‌جا می‌کند.

نتیجه‌گیری

در پایان، تلاش ما در حوزه تولید خودکار آزمون واحد توسط یک مدل LSTM دنباله‌به‌دنباله (seq2seq) پیشرفت قابل توجهی را در ادغام هوش مصنوعی و آزمون نرم‌افزار نشان می‌دهد. تحقیق ارائه شده در این مقاله نه تنها پتانسیل مدل‌های یادگیری ماشین در انجام وظایف مرسومی که به صورت دستی توسط توسعه دهندگان انجام می‌شود را نشان می‌دهد، بلکه چالش‌ها و پیچیدگی‌های موجود در چنین تلاش‌هایی را نیز برجسته می‌کند.

توانایی مدل ما در تولید آزمون‌های واحد مناسب از نظر نحوی و معنایی برای عملیات حسابی، بر اثرگذاری کاربردهای هوش مصنوعی در زمینه‌های عملی توسعه نرم‌افزار تأکید می‌کند. با این حال، محدودیت‌های مواجه شده در انجام دقیق محاسبات عددی در آزمون‌ها، چالش‌های ظریف پیش رو را آشکار می‌سازند.

مسیر آغاز شده برای اصلاح رویکردهای مبتنی بر هوش مصنوعی در آزمون نرم‌افزار درحال ادامه است. بینش‌های به دست آمده از این مطالعه مبنایی ارزشمند را برای تحقیقات آینده فراهم می‌کند، که افزایش دقت و قابلیت اجراء چنین مدل‌هایی را هدف قرار می‌دهد. هدف نهایی دست‌یابی به یک هم‌افزایی بین قابلیت‌های هوش مصنوعی و مهارت‌های انسانی است که منجر به فرآیندهای توسعه نرم‌افزار کارآمدتر، قابل اعتمادتر و نوآورانه‌تر می‌شود.

بنابراین، این تحقیق نه تنها بر گفتمان آکادمیک کاربرد هوش مصنوعی در مهندسی نرم‌افزار کمک می‌کند، بلکه به عنوان پله‌ای به سمت تحقق ابزارهای پیشرفته‌تر و کاربردی‌تر هوش مصنوعی در آزمون نرم‌افزار عمل خواهد نمود.

منابع

[1] Wang, S., Shrestha, N., Subburaman, A. K., Wang, J., Wei, M., & Nagappan, N. (2021, May). Automatic unit test generation for machine learning libraries: How far are we?. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)* (pp. 1548-1560). IEEE. [URL]

[2] Santiago, D., King, T. M., & Clarke, P. (2018). AI-Driven test generation: machines learning from human testers. In *Proceedings of the 36th Pacific NW Software Quality Conference* (pp. 1-14). [URL]

[3] Santhanam, S. (2020). Context based text-generation using lstm networks. *arXiv preprint arXiv:2005.00048*. [URL]

[4] Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *Advances in neural information processing systems, 27*. [URL]

[5] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*. [URL]