سال ۱۴۰۰، شماره ۱، صفحات ۱۱

علوم داده در مهندسی نرم افزار

Course No. 40925

تشخیص ناهنجاری مبتنی بر گزارش‌های سیستمی

آرش لگزیان و حسن حمیدی [۱ و *]، یاسمن عابدینی [۲]، عباس حیدرنوری [۳]

[۱] دانشجوی کارشناسی ارشد هوش مصنوعی دانشگاه صنعتی شریف، [۲] دانشجوی دکتری نرم‌افزار دانشگاه صنعتی شریف

[۳] دانشیار دانشکده مهندسی کامپیوتر دانشگاه صنعتی شریف

*مسئول مکاتبات: lagzian@ce.sharif.edu

◄ واژگان کلیدی

تشخیص ناهنجاری
ارزیابی گزارش‌های سیستمی
یادگیری ماشین
یادگیری عمیق
یادگیری بانظارت
یادگیری بدون‌نظارت
یادگیری نیمه‌نظارتی

◄ تاریخچه مقاله

تاریخ پایان نگارش: ۱۴۰۰/۱۱/۱۹

◄ چکیده

امروزه تشخیص رویداد نامتعارف که از آن با نام ناهنجاری[a] یاد می‌شود در زمینه‌های مختلف مهندسی کامپیوتر مانند: نرم‌افزار، هوش مصنوعی و اینترنت اشیاء دارای اهمیت فراوان است و کاربردهایی از جمله تشخیص برنامه‌های ناهنجار یا غیرعادی، تشخیص دستورات غیرعادی، تشخیص ورودی‌های غیرعادی و مواردی که در آن یک سیستم هوشمند دچار اشتباه یا افت کارایی می‌شود را دارد و با تشخیص ناهنجاری[b] می‌توان از خسارات ناشی از ناهنجاری کاست. گزارش‌های سیستمی[c] به دلیل داشتن جزئیات اجرای برنامه‌ها، زمان اجرا، اطلاعات کاربر و طریقه اجرای هریک از برنامه‌ها در سیستم‌های کامپیوتری از سوی مهندسین نرم‌افزار و توسعه‌دهندگان سیستم مورد توجه فراوان است و با بررسی آن‌ها می‌توان به جزئیات کافی از اجرای برنامه‌ها پی‌برد. اما به دلیل حجم بالای این گزارش‌های سیستمی، بررسی آن‌ها توسط فرد دارای مهارت هزینه زمانی بالایی دارد و عملا غیرممکن است. در پژوهش‌های اخیر از سوی پژوهشگران از بین روش‌های موجود استفاده از یادگیری ماشین[d] و یادگیری عمیق[e] برای تشخیص ناهنجاری در رویکردهای مختلف بانظارت[f]، بدون‌نظارت[g] و نیمه‌نظارتی[h] مورد استقبال قرار گرفته‌اند. در این مقاله مروری به بررسی کامل پژوهش‌های انجام شده در هریک از این رویکردها پرداخته و نقاط قوت و ضعف هریک را مورد بررسی قرار می‌دهیم و در انتها روشی ارائه می‌دهیم که از روش‌های موجود کارایی بهتری دارد.

[a] anomaly [b] anomaly detection [c] logs [d] machine learning [e] deep learning [f] supervised [g] unsupervised [h] semi supervised

۱ مقدمه

گزارش‌های سیستمی به صورت گسترده جهت عیب‌یابی[۱] در سیستم‌های کامپیوتری از سوی مهندسین نرم‌افزار و توسعه‌دهنده‌ها مورد استفاده قرار می‌گیرند زیرا این گزارش‌های سیستمی دارای اطلاعات کامل و جامع از مسیر اجرای برنامه‌ها و جزئیات اجرای آن‌ها هستند. گزارش‌های سیستمی ساختاری مشابه رشته‌های متنی[۲] دارند و برای ثبت کردن وقایع[۳] و حالت‌های[۴] مهم و مورد توجه بکار می‌روند و مهندسین نرم‌افزار از طریق ثبت و بررسی آن‌ها می‌توانند وظیفه تشخیص ناهنجاری را انجام دهند و وضعیت سیستم را مورد بازبینی قرار دهند. در پژوهش‌های انجام شده در این جوزه، در بعضی پژوهش‌ها با فرض جهان بسته[۵]، ثابت بودن گزارش‌های سیستمی در زمان آموزش و ارزیابی را در نظر گرفته‌اند اما این مدل‌ها به دو دلیل کارایی زیادی در مسائل عملی ندارند که در قسمت چالش‌ها، شماره‌های ۱ و ۲ به بیان آن‌ها پرداخته می‌شود. همچنین بررسی این مسئله با چالش‌های دیگری هم همراه است که در از جمله آن‌ها به موارد زیر می‌توان اشاره کرد:

۱. توسعه و بروزرسانی گزارش‌های سیستمی موجب تغییر آن‌ها در طول زمان و باعث نقض فرض جهان بسته می‌شود [۱]، [۲]، [۳]، [۴].
۲. وجود نوفه پردازشی[۶] در گزارش‌های سیستمی باعث نقض فرض جهان بسته می‌شود [۵]، [۶]، [۷]، [۸]، [۹].
۳. نبود مجموعه دادگان غنی در بسیاری از موارد جهت آموزش مدل‌های هوشمند.
۴. تنظیم ابرپارامترها[۷] بعنوان مثال تعیین آستانه[۸] جهت تعیین مرز بین نمونه عادی و غیرعادی می‌تواند یک چالش باشد.

براي بررسی‌های بیشتر نیاز است تا ابتدا با مفاهیمی که در حوزه تشخیص ناهنجاري مورد استفاده قرار می‌گیرند آشنا شویم. در این گزارش پس از معرفی مفاهیم مورد نیاز، مسئله بطور دقیق تعریف می‌شود و به معیارهاي بررسی کیفیت نتیجه اشاره می شود. سپس به بررسی پژوهش‌هاي پیشین پرداخته می‌شود و در هر مورد ضعف هاي پژوهش مورد بررسی ذکر می شود و در انتها روش پیشنهادی و پیاده‌های انجام شده مطرح می‌گردند.

۲ تعریف مفاهیم و مسئله

در این بخش ابتدا مفاهیم مورد نیاز تعریف شده سپس مسئله شرح داده شده است.

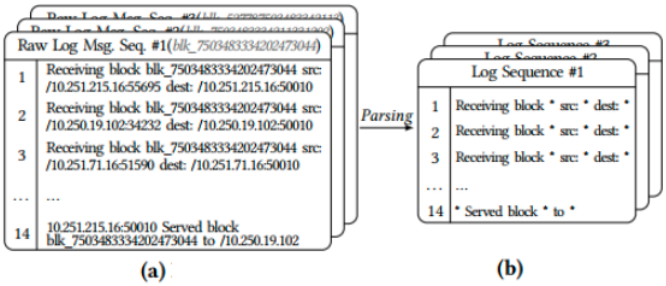
۱.۲ مفاهیم مورد نیاز

۱. **نمونه عادی:** *نمونه‌ای که از توزیع داده‌های عادی یا داخل دسته برداشته شده است.*

[1] troubleshooting [2] text string [3] events [4] states [5] closed-world [6] processing noise [7] hyper parametres [8] threshold

۲. **نمونه غیرعادی:** *نمونه‌ای که متعلق به توزیع داده‌های عادی نیست.*
۳. **تشخیص ناهنجاری:** *بطور کلی تمایز قائل شدن بین نمونه عادی و غیر عادی است. در این بررسی به دلیل تمرکز بر حوزه گزارش‌های سیستمی، به تمایز قائل شدن بین پیام‌هایی که گزارش حالت عادی کاری سیستم را گزارش می‌کنند و گزارش‌هایی که در آن‌ها حالت غیر عادی سیستم را گزارش می‌کنند گفته می‌شود.*
۴. **نمونه صحیح ـ مثبت[۹] یا TP:** *نمونه غیرعادی که به درستی تشخیص داده شده است.*
۵. **نمونه صحیح ـ منفی[۱۰] یا TN:** *نمونه عادی که به درستی تشخیص داده شده است.*
۶. **نمونه اشتباه ـ مثبت[۱۱] یا FP:** *نمونه عادی که به اشتباه غیرعادی تشخیص داده شده است.*
۷. **نمونه اشتباه ـ منفی[۱۲] یا FN:** *نمونه غیرعادی که به اشتباه عادی تشخیص داده شده است.*
۸. **اصطلاحات مرتبط به گزارش‌های سیستمی:** *در شکل ۱ نمونه داده از گزارش‌های سیستمی قابل مشاهده است. هر خط چاپ شده در قسمت a نشانگر یک پیام سیستمی[۱۳] می‌باشد. هر پیام سیستمی از دو بخش رویداد سیستمی[۱۴] و پارامتر سیستمی[۱۵] تشکیل شده است. پارامتر سیستمی اطلاعاتی مثل آدرس IP، نام فایل و URL را دارد که این اطلاعات متغیر هستند، اما رویدادهای سیستمی اکثرا ثابت درنظر گرفته می‌شوند و شبیه به رشته‌های متنی هستند که توسط برنامه‌نویسان و توسعه‌دهندگان سیستم نوشته می‌شوند. در فرایند تجزیه گزارش سیستمی[۱۶] رویدادهای سیستمی از گزارش‌های سیستمی جدا می‌شوند سپس یک توالی از رویدادهای سیستمی که Task_ID یکسان دارند بعنوان یک توالی سیستمی[۱۷] درنظر گرفته می‌شوند.*
۹. **تجزیه گزارش سیستمی:** *به فرآیند استخراج توالی سیستمی از پیام سیستمی گفته می‌شود.*

شکل ۱: مثالی از مجموعه دادگان HDFS که در آن پیام سیستمی، رویداد سیستمی و توالی سیستمی نمایش داده شده است. شکل a مجموعه‌ای از پیام‌های سیستمی و شکل b نمایی از یک توالی سیستمی با task_ID = blk_7503483334202473044 است[۱۰].



۲.۲ بیان مسئله

در این مقاله مروری قصد داریم تا با استفاده از بررسی پژوهش‌های انجام شده در هر یک از رویکردهای معرفی شده بر روی وظیفه تشخیص ناهنجاری، به تحلیل و بررسی و جمع‌بندی آنها بپردازیم و سپس روش جدید و مبتکرانه ارائه برای این مورد ارائه دهیم.

مجموعه دادگان: در این حوزه مجموعه دادگان‌های مختلفی مورد استفاده قرار می‌گیرند. جدول ۱ دسته‌بندی و جمع بندی با ارزش از تمامی آن‌ها را ارائه می‌دهد.

جدول ۱: جمع‌بندی مجموعه دادگان‌های موجود

Messages	Description	Dataset	Type logs
۱۱, ۱۷۵, ۶۲۹	Hadoop distributed file system log	HDFS	Distributed system logs
۳۹۴, ۳۰۸	Hadoop mapreduce job log	Hadoop	
۳۳, ۲۳۶, ۶۰۴	Spark job log	Spark	
۷۴, ۳۸۰	ZooKeeper service log	ZooKeeper	
۲۰۷, ۸۲۰	OpenStack software log	OpenStack	
۴, ۷۴۷, ۹۶۳	Blue Gene/L supercomputer log	BGL	Supercomputer logs
۴۳۳, ۴۸۹	High performance cluster log	HPC	
۲۱۱, ۲۱۲, ۱۹۲	Thunderbird supercomputer log	Thunderbird	
۱۱۴, ۶۰۸, ۳۸۸	Windows event log	Windows	Operating system logs
۲۵, ۵۶۷	Linux system log	Linux	
۱۱۷, ۲۸۳	Mac OS log	Mac	
۳۰, ۳۴۸, ۰۴۲	Android framework log	Android	Mobile system logs
۲۵۳, ۳۹۵	Health app log	HealthApp	
۵۶, ۴۸۱	Apache server error log	Apache	Server application logs
۶۵۵, ۱۴۶	OpenSSH server log	OpenSSH	
۲۱, ۳۲۹	Proxifier software log	Proxifier	Standalone software logs

از بین مجموعه دادگان‌های موجود دو مجموعه دادگان BGL[۱۱] و HDFS[۹] بیشتر مورد توجه هستند.

معیارهای ارزیابی: معیارهای ارزیابی در این حوزه موارد زیر می‌باشند:

$$Precision = \frac{TP}{TP+FP}$$
$$Recall = \frac{TP}{TP+FN}$$
$$F1-Score = \frac{2*Precision*Recall}{Precision*Recall}$$

۳ پژوهش‌های پیشین

۱.۳ تجزیه گزارش‌های سیستمی

تجزیه گزارش‌های سیستمی یک مرحله مهم است که در همه رویکردهای موجود که در ادامه بررسی می‌شوند مورد استفاده قرار می‌گیرد. با توجه به اینکه انتخاب یک تجزیه‌گر[۱۸] در مدل بسته به کاربردهای مختلف می‌تواند متفاوت باشد، در این بخش به بررسی این مهم پرداخته می‌شود. در جدول ۲ جمع‌بندی و خلاصه تمامی روش‌های موجود در این بخش که تا کنون ارائه شده‌اند گردآوری شده است که در ادامه به توضیح هر یک از ویژگی‌های بیان شده در این جدول می‌پردازیم.

۱. **روش فنی[۱۹]:** *تجزیه‌گرهای مختلف از رویکردهای متفاوتی جهت تجزیه پیام‌ها استفاده می‌کنند. در این ستون تجزیه‌گرهای موجود به ۷ دسته مختلف دسته‌بندی شده‌اند.*

(الف) **استخراج الگوهای تکرارشونده[۲۰]:** *الگوهای تکرارشونده به مجموعه‌ای از الگوها گفته می‌شود که تعداد تکرار زیادی در مجموعه دادگان دارند. تجزیه‌گرها با استفاده از یادگیری این*

[9] true positive [10] true negative [11] false positive [12] false negative [13] log message [14] log event [15] log parameter [16] log parsing [17] log sequence [18] parser [19] technique [20] frequent pattern mining [21] clustering

الگوهای تکرارشونده می‌توانند بصورت خودکار عمل تجزیه را انجام دهند

(ب) **گروه‌بندی**[۲۱]: در این روش‌ها مسئله تجزیه پیام‌ها به یک مسئله گروه‌بندی تبدیل می‌شود که در آن بر اساس الگوهایی که هر گروه از گزارش‌های سیستمی دارند گروه‌بندی می‌شوند و سپس بر اساس ویژگی‌های هر گروه طبق روش‌های از پیش تعریف شده فرآیند تجزیه صورت می‌گیرد. روش‌های گروه بندی مختلفی می‌تواند اعمال شود مانند گروه‌بندی سلسه مراتبی[۲۲].

(ج) **فرآیندهای کاوشی**[۲۳]: گزارش‌های سیستمی را باتوجه به مقایسه بین توکن‌های[۲۴] ثابت و متغیر که در آن‌ها وجود دارد انجام می‌دهد و بعد از تقسیم‌بندی پیام‌ها در هر گروه فرآیند تجزیه انجام می‌گیرد.

(د) **جزءبندی**[۲۵]: باتوجه به طول پیام، موقعیت هر توکن و رابطه بین آن‌ها پیام‌ها به دسته‌های مختلفی تقسیم می‌شوند و سپس فرآیند تجزیه انجام می‌گیرد.

(ه) **درخت تجزیه**[۲۶] از یک ساختار درختی با عمق ثابت برای تجزیه گزارش‌های سیستم استفاده می‌شود.

(و) **بزرگ‌ترین زیردنباله مشترک**[۲۷]: از بزرگترین زیردنباله‌های مشترک در بین گزارش‌های سیستمی جهت تجزیه استفاده می‌شود.

(ز) **الگوریتم‌های تکاملی**[۲۸]: تجزیه گزارش‌های سیستمی را به عنوان یک مسئله بهینه‌سازی چند هدفه مدل می‌کند و آن را با استفاده از الگوریتم‌های تکاملی حل می‌کند.

۲. **مد**[۲۹]: تجزیه‌گرهای مختلف را می‌توان به دو دسته کلی برخط[۳۰] و برون‌خط[۳۱] دسته بندی کرد.

(الف) **مد برون‌خط:** در این مد باید ابتدا تمامی پیام‌ها به تجزیه‌گر داده شود و سپس فرآیند تجزیه آغاز می‌گردد.

(ب) **مد برخط:** در این مد تجزیه‌گر می‌تواند بصورت تک به تک پیام‌ها را دریافت کند و فرآیند تجزیه را آغاز کند که در عمل استفاده از این مد کاربرد عملی بیشتری دارد.

۳. **میزان بهینگی**[۳۲]: با توجه به حجم زیاد گزارش‌های سیستمی، استفاده از یک تجزیه‌گر غیر بهینه علاوه بر مصرف زیاد انرژی می‌تواند کارایی سیستم را پایین آورد و تشخیص ناهنجاری را با مشکل و خطا روبرو کند. در مقابل تجزیه‌گرهای بهینه این مورد را بهبود می‌بخشند. در جدول ۲ تجزیه‌گرها به ۳ سطح بهینگی مختلف تقسیم شده‌اند: *Low, Medium, High*.

۴. **پوشش‌دهی**[۳۳]: تجزیه‌گرهای مختلف باتوجه به اینکه آیا می‌توانند تمامی پیام‌های ورودی را تجزیه کنند به دو دسته تقسیم می‌شوند. اگر یک تجزیه‌گر بتواند تمام پیام‌های ورودی را تجزیه کند در ستون مورد نظر پوشش‌دهی آن *yes* می‌شود و اگر فقط بتواند پیام‌های دارای ساختار را تجزیه کند پوشش‌دهی آن *no* است.

۵. **پیش پردازش**[۳۴]: عمل پیش پردازش به حذف موارد رایج و یا بدون کاربرد مانند آدرس IP یا اعداد از پیام ورودی گفته می‌شود که می‌تواند با استفاده از عبارت‌های منظم[۳۵] یا روش‌های دیگری صورت گیرد. اگر یک مرحله پیش پردازش به صراحت در روش تجزیه گزارش شده باشد در این ستون مقدار *yes* و در غیر اینصورت مقدار *no* قرار دارد و از این دیدگاه تجزیه‌گرها به دو دسته تقسیم شده اند.

۶. **متن‌باز**[۳۶]: تجزیه‌گرهای مختلف از نظر در دسترس‌پذیری برای محققین و پژوهش‌گرها به دو دسته کلی تقسیم می‌شوند. اگر تجزیه‌گر مورد نظر در دسترس محققین و پژوهش‌گران باشد مقدار *yes* و در غیر اینصورت *no* دریافت می‌کند.

جمع‌بندی: با توجه به ذکر اهمیت استفاده از تجزیه‌گر مناسب در فرآیند تشخیص ناهنجاری، در این بخش جمع‌بندی و خلاصه‌سازی کامل از تمامی تجزیه‌گرهای موجود ارائه شد و در جدول ۲ این اطلاعات قابل مشاهده است. با توجه به جمع‌بندی صورت گرفته، استفاده از تجزیه‌گر *Drain*[۱۲] با توجه به ویژگی‌های ذکر شده برای آن که مورد بررسی قرار گرفتند در اکثر روش‌های تشخیص ناهنجاری ترجیح داده شده است.

جدول ۲: جمع‌بندی روش‌های مختلف تجزیه گزارش‌های سیستمی

Log Parser	Technique	Mode	Efficiency	Coverage	Preprocessing	Open Source
SLCT[13]	Frequent pattern mining	Offline	High	no	no	yes
AEL[14]	Heuristics	Offline	High	yes	yes	no
IPLoM[15]	Iterative partitioning	Offline	High	yes	no	no
LKE[16]	Clustering	Offline	Low	yes	yes	no
LFA[17]	Frequent pattern mining	Offline	High	yes	no	no
LogSig[18]	Clustering	Offline	Medium	yes	no	no
SHISO[19]	Clustering	Online	High	yes	no	no
LogCluster[20]	Frequent pattern mining	Offline	High	no	no	yes
LenMa[21]	Clustering	Online	Medium	yes	no	yes
LogMine[22]	Clustering	Offline	Medium	yes	yes	no
Spell[23]	Longest common subsequence	Online	High	yes	no	no
Drain[12]	Parsing tree	Online	High	yes	yes	yes
MoLFI[24]	Evolutionary algorithms	Offline	Low	yes	yes	yes

۲.۳ رویکردهای بانظارت

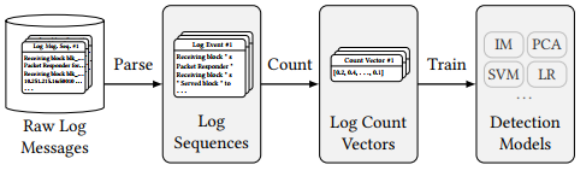
در این بخش به بررسی رویکردهای بانظارت جهت انجام وظیفه تشخیص ناهنجاری مبتنی بر گزارش‌های سیستمی پرداخته می‌شود و سپس دسته بندی و نتیجه‌گیری از تمامی روش‌های موجود ارائه می‌شود.

۱. ۲. **SVM**[۸]، **LR**[۲۵]: در این روش‌ها بصورت کلی بر روی داده‌های برچسب‌دار عمل تشخیص ناهنجاری صورت می‌گیرد و شباهت زیادی در پیاده سازی دارند و تفاوت آن‌ها در مدل‌های دسته‌بندی استفاده شده در هرکدام است. در این روش‌ها ابتدا از پیام‌های سیستمی ورودی توالی پیام‌ها توسط تجزیه گر استخراج می‌شوند. سپس از این توالی‌ها بردارهایی با طول ثابت استخراج می‌شود که این بردارها بر اساس شمارش رویدادهای هر پیام تشکیل شده‌اند و بعد از فرآیند بردارسازی آموزش مدل صورت می‌گیرد.

در شکل ۲ همانطور که قابل مشاهده است این دو روش از یک مسیر وظیفه تشخیص ناهنجاری را انجام می‌دهند و تفاوت آن‌ها در جزئیات مدل‌های استفاده شده می‌باشد. همچنین دو روش IM و PCA نیز که از روش‌های بدون‌نظارت هستند به دلیل تشابه مسیر اجرا در این شکل جهت دسته‌بندی آورده شده‌اند و در بخش رویکردهای بدون‌نظارت توضیح داده خواهند شد.

[22] hierarchical clustering [23] heuristics [24] token [25] partitioning [26] parsing tree [27] longest common subsequence [28] evolutionary algorithms [29] mode [30] online [31] offline [32] efficiency [33] coverage [34] preprocessing [35] regular expression [36] open source

شکل ۲: شمای کلی روش های [۲۶]IM، [۲۵]LR، [۸]SVM، [۲۷]PCA.



اما این روش‌ها دارای معایبی هستند از جمله:

- گزارش‌های سیستمی به دلیل بروزرسانی و تغییرات سیستم ماهیتی ناپایدار دارند که این روش‌ها با این تغییرات ناسازگار هستند و در صورت تغییرات نیاز به آموزش مجدد دارند.
- بردارسازی با توجه به شمارش رویدادهای سیستمی صورت می‌گیرد و نسبت به زمینه[۳۷] بی‌توجه هستند و در نتیجه نمی‌توانند بین رویدادهای مختلف تمایز قائل شوند و برای همه آن‌ها اهمیت یکسان در نظر می‌گیرند.
- مسئله بروزرسانی ابزار تشخیص ناهنجاری یک مسئله مهم است که در این روش‌ها قابلیت بروز رسانی وجود ندارد و مجددا باید آموزش ببینند که این عمل هزینه زمانی و پیچیدگی محاسباتی را به شدت افزایش می‌دهد.

با توجه به معایبی که دو پژوهش ارائه شده قبلی داشتند در ادامه بررسی رویکردهای بانظارت به بررسی پژوهش سوم می‌پردازیم.

۳. **LogRobust**[۱۰]: *در ادامه پژوهش‌ها در این حوزه، پژوهش LogRobust جهت رفع مشکل ناپایدار بودن گزارش‌های سیستمی ارائه شد. این مدل از سه بخش کلی تشکیل شده است:*

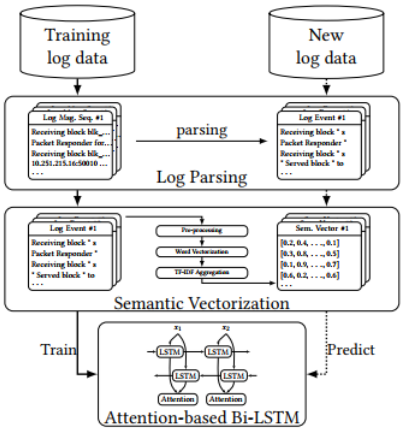
(الف) تجزیه گزارش‌های سیستمی

(ب) بردارسازی مفهومی[۳۸]

(ج) دسته‌بندی بر مبنای سازوکار توجه[۳۹]

در شکل ۳ شمای کلی مدل ارائه شده در این پژوهش و سه بخش معرفی شده قابل مشاهده هستند و به توضیح نحوه ارتباط این بخش‌ها می‌پردازیم.

شکل ۳: مدل ارائه شده در پژوهش LogRobust[۱۰]



در مرحله اول عمل تجزیه پیام سیستمی با استفاده از روش Drain صورت می‌گیرد سپس با استفاده از بردارسازی مفهومی برخلاف بردارسازی‌های سنتی که فقط به شمارش رویدادها می‌پردازند، یک بردار برای هر توالی سیستمی در نظر گرفته می‌شود از مزایای این روش بردارسازی می‌توان به انعطاف‌پذیری در برابر پیام‌های سیستمی ناپایدار و همچنین مقاوم بودن در برابر نوفه اشاره کرد زیرا در نهایت به مفهوم و شباهت بردارها توجه می‌کند.

بردار سازی مفهومی از سه بخش تشکیل شده است:

(الف) ابتدا یک عملیات پیش‌پردازش بر روی داده‌های ورودی انجام می‌شود و طی آن موارد زیر حذف می‌شوند:

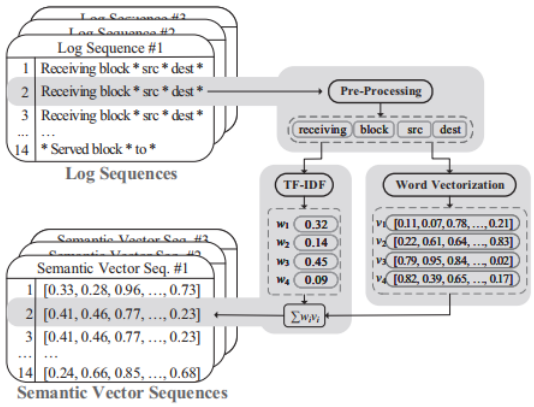
* جداکننده‌ها[۴۰]
* عملگرها[۴۱]
* نشانه‌ها[۴۲]
* اعداد
* ایست‌واژه‌ها[۴۳]

(ب) سپس با استفاده از الگوریتم FastText[۲۸] یک بردارسازی از داده‌های پیش‌پردازش شده بدست می‌آید.

(ج) و در انتها با اعمال TF-IDF[۲۹] اهمیت هر لغت در هر بردار مشخص می‌شود.

در شکل ۴ نحوه اعمال هریک از موارد فوق جهت بردارسازی نشان داده شده است.

شکل ۴: روش بردارسازی ارائه شده در پژوهش LogRobust[۱۰]



به ازای یک رویداد E در خروجی این مرحله بردار V را خواهیم داشت و بطور هم‌زمان برای رویدادهای دیگر نیز بردارسازی صورت می‌گیرد و در نهایت یک توالی برداری به صورت $[V1\ V2\ V3 \dots]$ را خواهیم داشت که به آن بردار توالی مفهومی[۴۴] گفته می‌شود.

سپس با بکارگیری شبکه عصبی attention-based Bi-LSTM[۳۰] رابطه بین رویدادهای سیستمی مختلف و همچنین اهمیت هر یک از آن‌ها توسط مدل آموزش دیده می‌شود و مجددا به دلیل توجه مدل به مفهوم پیام و رابطه‌ی بین اجزای آن، پایداری مدل نسبت تغییرات رویدادهای سیستمی بیشتر می‌شود و درنهایت مدل نسبت به ناپایدار بودن پیام‌های سیستمی مقاوم[۴۵] می‌شود.

در جدول ۳ جمع‌بندی نتایج روش‌های بانظارت ارائه شده است و همانطور که قابل مشاهده است روش LogRobust بعنوان بهترین مدل با رویکرد بانظارت

[37] context [38] semantic vectorization [39] attention based classification [40] delimiters [41] operators [42] punctuation [43] stop words [44] semantic vector sequence [45] robust

تا کنون در بین پژوهش‌های پیشین می‌باشد.

جدول ۳: نتایج پیاده‌سازی رویکردهای بانظارت بر روی مجموعه دادگان HDFS

Method	Precision	Recall	F1-Score
LR	۰.۹۹	۰.۹۲	۰.۹۶
SVM	۰.۹۹	۰.۹۴	۰.۹۶
LogRobust	۰.۹۸	۱.۰۰	۰.۹۹

با وجود نتایج خوب و دقت‌های بالای روش‌های موجود در رویکرد بانظارت، این روش‌ها دارای معایب اساسی هستند که عملا استفاده از آن‌ها را غیر ممکن می‌کند. مانند:

۱. این روش‌ها به داده‌ها زیادی جهت آموزش مدل‌ها هوشمند نیاز دارند. [۳۱]
۲. برچسب زدن داده‌ها هزینه زمانی زیادی دارد همچنین برای برچسب زدن داده‌ها نیاز به فرد متخصص است. [۳۱]

۳.۳ رویکردهای بدون‌نظارت

در این بخش به بررسی رویکردهای بدون‌نظارت جهت انجام وظیفه تشخیص ناهنجاری مبتنی بر گزارش‌های سیستمی پرداخته می‌شود و سپس دسته بندی و نتیجه‌گیری از تمامی روش‌های موجود ارائه می‌شود.

۱. **IM**[۲۶]: *در این پژوهش با فرض ثابت بودن گزارش‌های سیستمی برای برنامه‌های موجود در سیستم، یک مدل تشخیص ناهنجاری ارائه شده است و اساس این مدل توجه به تناقض‌ها در هر گروه از گزارش‌های سیستمی است. به این صورت که ابتدا از داده‌های بدون ساختار گزارش‌های سیستمی به وسیله یک تجزیه‌گر توالی‌هایی را که دارای ساختار هستند استخراج می‌کند، سپس با اعمال یک روش بردارسازی، بردارهای ثابتی از توالی‌ها را می‌سازد که بر مبنای تعداد تکرار رویدادها ساخته شده‌اند و بعد از آن با استفاده از این بردارها و تشابه پیام‌های هر گروه، کار گروه‌بندی را انجام دهد، حال این دسته‌بندی با توجه به ثابت بودن گزارش‌های سیستمی همیشه جواب ثابت و یکسانی دارد و تنها در صورتی دسته بندی جواب دیگری خواهد داشت که نمونه ناهنجار در گزارش‌های سیستمی رخ دهد، و این نمونه از طریق تجزیه ماتریس قابل شناسایی خواهد بود که از این طریق ابتدا وجود نمونه ناهنجار تشخیص داده می‌شود و سپس از طریق تجزیه ماتریس به نمونه ناهنجار دسترسی پیدا می‌کند.*

۲. **PCA**[۲۷]: *در این پژوهش با فرض برچسب نداشتن داده‌های آموزش، عمل تشخیص ناهنجاری انجام شده است. در مدل ارائه شده، ابتدا از گزارش‌های سیستمی که در ابتدا داده‌های بدون ساختار هستند توسط یک تجزیه‌گر داده‌ای ساختار یافته و توالی پیام‌ها استخراج می‌شود و برای این کار از الگوهایی که توسط برنامه نویسان طراحی شده است کمک گرفته می‌شود تا تجزیه گر بتواند طبق این الگوها رویدادها را استخراج کند. سپس با استفاده از بردارسازی بردارهایی از توالی پیام‌ها ساخته می‌شود. این بردار ها ثابت هستند و بر اساس شمارش رویدادها در هر توالی ساخته می‌شوند. سپس از این بردارها ویژگی‌ها استخراج می‌شود و بر اساس ویژگی‌های مشترک عمل دسته بندی و گروه‌بندی صورت می‌گیرد. سپس عمل تشخیص ناهنجاری با اعمال الگوریتم PCA بر روی ویژگی‌های استخراج شده از داده‌ها اعمال می‌شود که در نتیجه آن داده‌هایی که تشابه کمتری نسبت دیگر داده‌های دسته خود داشته باشند با در نظر گرفتن یک مقدار آستانه به عنوان داده ناهنجار انتخاب می‌شوند.*

۳. **LogCluster**[۳۲]: *در این روش با ذکر معایب روش‌های بانظارت، از اهمیت تشخیص ناهنجاری با استفاده از رویکردهای بدون‌نظارت سخن گفته است. در این روش ابتدا به هریک از پیام‌های سیستمی یک وزن خاص تخصیص داده می‌شود و سپس پیام‌های سیستمی گروه‌بندی می‌شوند. بعد از عمل گروه‌بندی بازنمایی[۴۶] برای هریک از گروه‌ها استخراج می‌شود. این روش در دوفاز اصلی ارائه شده است که در شکل ۵ قابل مشاهده است.*

(الف) **فاز بازسازی**[۴۷]: *توالی‌های سیستمی از محیط جمع‌آوری می‌شوند و بعد از بردارسازی، گروه‌بندی می‌شوند و برای هر گروه یک بردار بازنمایی استخراج می‌شود و سپس بعنوان دانش اولیه[۴۸] در پایگاه دانش[۴۹] در نظر گرفته می‌شوند.*

(ب) **فاز تولید**[۵۰]: *فرآیندی مشابه به فاز اول دارد بعد از جمع‌آوری داده از محیط، بردارسازی، گروه‌بندی و استخراج بازنمایی صورت می‌گیرد. سپس با مقایسه بازنمایی آن‌ها با بازنمایی‌های موجود در دانش اولیه و در نظر گرفتن یک مقدار آستانه، فرآیند تشخیص ناهنجاری صورت می‌گیرد. همچنین دانش موجود در پایگاه دانش با دانش بدست آمده از گروه‌های جدید در هر بار بروزرسانی می‌شود و برای گروه‌های تکراری این دانش کنارگذاشته می‌شود.*

با توجه به فازهایی که این الگوریتم دارد، تعداد داده‌هایی که نیاز به بررسی شدن دارند کم‌تر می‌شوند و نیاز است تا از هر گروه تعدادی از داده‌ها مورد ارزیابی قرار بگیرند تا ویژگی‌های داده‌های کل گروه با دقت خوبی مشخص شود. از این مورد بعنوان یکی دیگر از مزایای این روش می‌توان نام برد.

در این بخش به توضیح چهار فرآیند که در هر دو فاز مشترک هستند پرداخته می‌شود:

(الف) **بردارسازی:** *در این بخش ابتدا پیام‌ها توسط تجزیه‌گر LKE به توالی پیام‌ها تبدیل می‌شوند سپس عمل بردارسازی صورت می‌گیرد و با استفاده از TF-IDF مقادیر موجود در هر بردار وزن‌دهی می‌شوند و این بردارها جهت گروه‌بندی در فاز بعد مورد اسفاده قرار می‌گیرند.*

(ب) **گروه‌بندی:** *در این بخش با اعمال معیار شباهت کسینوسی[۵۱]، شباهت‌ها بین بردارهای موجود محاسبه می‌شوند. الگوریتم مورد استفاده جهت گروه‌بندی سلسله مراتبی تجمعی[۵۲] [۳۳] می‌باشد. این الگوریتم ابتدا تمامی داده‌ها را گروه‌های یک عضوی مجزا در نظر می‌گیرد و سپس با اعمال معیار شباهت*

[46] representation [47] reconstruction phase [48] prior knowledge [49] knowledge base [50] production phase [51] cosine similarity [52] agglomerative hierarchical clustering

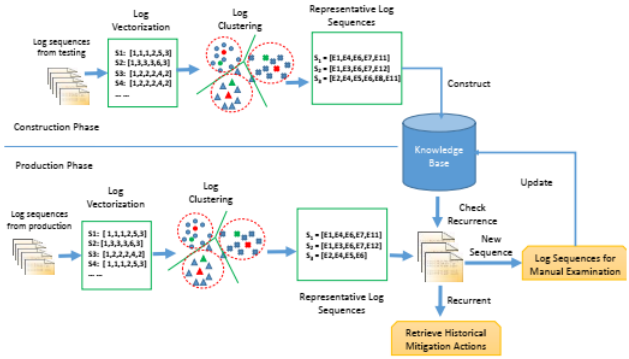
آن‌ها را در گروه‌های مشترک قرار می‌دهد و تا جایی این روند ادامه پیدا می‌کند که حداقل گروه‌های مجزا را داشته باشیم که به وسیله یک شرط توقف قابل تعریف که اغلب به صورت تجربی به‌دست می‌آید و اعمال می‌شود.

(ج) **استخراج بازنمایی از توالی پیام‌ها:** بعد از انجام گروه‌بندی و بدست آوردن گروه‌ها، برای هر گروه بازنمایی بدست می‌آوریم. به این صورت که به ازای همه توالی‌های سیستمی موجود در هر کلاستر، طبق رابطه (۱) مرکز هر کلاستر[۵۳] را می‌یابیم و از بازنمایی آن بعنوان نماینده هر کلاستر استفاده می کنیم.

(۱)
$$\text{Score}(i) = \frac{۱}{n-۱}\sum_{j=۱}^{n}(۱ - \text{Similarity}\,(S_i, S_j))$$

(د) **مقایسه بازنمایی‌ها:** در این مرحله بازنمایی‌های بدست آمده از مرحله قبل با بازنمایی‌های موجود در پایگاه دانش مقایسه می‌شوند و در صورت تکراری بودن کنار گذاشته می‌شوند زیرا تاریخچه آن‌ها را داریم و در صورت جدید بودن پایگاه دانش بروزرسانی می‌شود و دانش جدید را در خود ذخیره می‌کند.

شکل ۵: شمای کلی پژوهش LogCluster[۳۲]



۴. **DeepLog**[۵]: در این پژوهش با در نظر گرفتن شباهت پردازش گزارش‌های سیستمی و پردازش زبان طبیعی[۵۴] یک مدل شبکه عصبی به صورت بدون‌نظارت روی داده‌های عادی آموزش می‌بیند و سپس در فاز ارزیابی نمونه‌هایی که از توزیع داده‌های عادی منحرف می‌شوند را بعنوان نمونه ناهنجار شناسایی می‌کند. شبکه عصبی استفاده شده در این مدل LSTM[۳۴] است که روابط بین اجزای یک توالی را نیز در نظر بگیرید. مدل ارائه شده در این پژوهش دارای دو فاز اصلی است که در ادامه به تشریح هریک از آن‌ها می‌پردازیم.

فاز آموزش: در این مرحله با داشتن داده‌های عادی، آموزش مدل صورت می‌گیرد. به این صورت که ابتدا پیام‌های سیستمی به دو بخش کلید[۵۵] و بردار مقادیر پارامترها[۵۶] تجزیه می‌شوند و این کلید ها یکتا[۵۷] هستند. سپس با تعیین یک پنجره هربار تعدادی از داده‌ها به مدل داده می‌شوند و مدل این توالی داده‌ها را یاد می‌گیرد بعنوان مثال اگر طول پنجره برابر با $N = ۳$ باشد و توالی پیام‌های سیستمی به صورت زیر باشد $\{K_{۲۲}, K_{۵}, K_{۱۱}, K_{۹}, K_{۱۱}, K_{۲۶}\}$ داریم:

$$\{k_{۲۲}, k_{۵}, k_{۱۱} \rightarrow k_{۹}\}$$
$$\{k_{۵}, k_{۱۱}, k_{۹} \rightarrow k_{۱۱}\}$$
$$\{k_{۱۱}, k_{۹}, k_{۱۱} \rightarrow k_{۲۶}\}$$

که در هر بار سه کلید بعنوان ورودی به مدل داده می‌شوند و کلید بعدی بعنوان برچسب خروجی در نظر گرفته می‌شود و مدل آموزش می‌بیند.

فاز تشخیص ناهنجاری: بعد از آموزش مدل در هنگام تست بعد از استخراج کلیدها از داده‌ها، با در نظر گرفتن طول پنجره یکسان با زمان آموزش، توالی‌های تجزیه شده به مدل داده می‌شوند و مدل احتمال خروجی هر یک از کلیدهای پیش بینی شده را محاسبه می‌کند و اگر این احتمال از یک مقدار آستانه کمتر باشد این توالی بعنوان ناهنجاری در نظر گرفته می‌شود.

همچنین مدل DeepLog توانایی یادگیری بصورت برخط را در زمان تست دارد به این صورت که توالی‌های جدید را نیز آموزش می‌بیند اما به دلیل ضعف شبکه‌های عصبی در فراموشی داده‌های قبلی forgetting باعث می‌شود تا الگوهای قدیمی اهمیت کمتری پیدا کنند و از دست بروند که از نقاط ضعف این مدل می‌باشد.

۵. **LogAnomaly**[۳۵]: در این پژوهش اولین بار در رویکردهای بدون‌نظارت به مفهوم پیام‌های سیستمی توجه شد و از بردارسازی مفهومی استفاده شد. در ابتدا با توجه به الگوهای رایجی که پیام‌های سیستمی مختلف دارند، با استفاده از دسته‌بندی درختی پیام‌های مختلف گروه‌بندی می‌شوند. سپس با استفاده از روش template2vec که یک روش مشابه word2vec[۳۶] اما ابداعی خود مقاله است این بردارسازی مفهومی صورت می‌گیرد. در روش tempalte2vec سعی شده است علاوه بر علاوه بر word2vec عادی از هم‌معنی ها و متضادهایی که در WordNet[۳۷] آورده شده است استفاده کند تا بتواند بردارهای تولیدی برای پیام‌های سیستمی که کلمات هم معنی یا متضاد در آن‌ها آمده است را به خوبی مدل کند. بعنوان مثال پیام سیستمی که عبارت service-up در آن آمده است اغلب نشان‌دهنده یک حالت معمول سیستم و پیام سیستمی که عبارت service-down در آن آمده است نشان‌دهنده یک ناهنجاری است که اگر با word2vec این بردارسازی را انجام دهیم بردارهای حاصله این دو تفاوت زیادی ندارند اما با روش ارائه شده در این مقاله یعنی template2vec می‌توانیم حداکثر فاصله بین این دو بردار را داشته باشیم که از مزایای این پژوهش نسبت به پژوهش‌های قبلی است.

کدهایی که در برنامه اجرا می‌شوند اکثر اوقات توالی مشابهی از پیام‌های سیستمی را تولید می‌کنند و در این پژوهش مبنای تشخیص ناهنجاری همین اصل در نظر گرفته شده است و به صورت بدون‌نظارت فرآیند تشخیص ناهنجاری انجام می‌شود. و این فرآیند بعد از بردارسازی مفهومی توسط الگوریتم ذکر شده صورت می‌گیرد و از مدل یادگیری عمیق LSTM در این پژوهش استفاده می‌شود.

[53] centroid [54] Natural Language processing [55] key [56] parameter value vector [57] uniqe

بطور دقیق‌تر در زمان ارزیابی برخط این مدل ممکن است با پیام‌های سیستمی جدید برخورد کند که منطبق با هیچ یک از الگوهای پیشین نباشد، این پیام‌های سیستمی جدید در فرایند توسعه نرم‌افزار توسط برنامه نویس‌ها طی بروزرسانی‌هایی به نرم‌افزارها اضافه می‌شوند بسیار رایج هستند. اگر از روش‌های مبتنی بر شاخص به جای روش‌های مبتنی بر بردار ویژگی استفاده شود مدل نمی‌تواند این پیام‌های سیستمی جدید را در فرآیند بگنجاند و این پیام‌های سیستمی جدید را به عنوان ناهنجاری در نظر می‌گیرد. اما حال بردار ویژگی این پیام‌های جدید را استخراج می‌کند و با پیام‌های سیستمی قبلی مقایسه می‌کند و پیام سیستمی جدید را جزء یکی از دسته‌های قبلی دسته‌بندی می‌کند و عمومیت[58] مدل نیز افزایش می‌یابد در کنار افزایش دقت مدل و در محیط واقعی می‌تواند بکارگرفته شود.

در این بخش تمامی مدل‌های تاثیرگذار و مهم رویکرد بدون‌نظارت مورد بررسی قرار گرفتند و در جدول ۴ و ۵ به مقایسه نتایج این پژوهش‌ها پرداخته شده است.

۴.۳ رویکردهای نیمه‌نظارتی

با توجه به ضعف‌هایی که در روش‌هایی با رویکردهای بانظارت بیان شد و همچنین کارایی پایین روش‌های بدون نظارت، پژوهشگران در این حوزه به استفاده از رویکردهای نیمه‌نظارتی جهت انجام وظیفه تشخیص ناهنجاری روی آوردند. بعنوان بهترین پژوهش نیمه‌نظارتی که در ICSE-2021 ارائه شده است پژوهش PLELog[۳۱] را مورد بررسی قرار می‌دهیم و بعد از آن نتایج این پژوهش را با روش‌های بدون نظارت و روش LogRobust بعنوان بهترین روش موجود در پژوهش‌های بانظارت در جدول ۴ ، ۵ مورد بررسی قرار می‌دهیم.

پژوهش PLELog: *مدل ارائه شده در این پژوهش دارای سه بخش است که در ادامه به تشریح هر یک از این بخش‌ها می‌پردازیم:*

۱. **بازنمایی مفهومی**[59]: *ورودی این بخش پیام‌های سیستمی و خروجی آن بردارهای مفهومی هستند و در طی سه مرحله از ورودی به خروجی می‌رسیم طبق گام‌های زیر:*

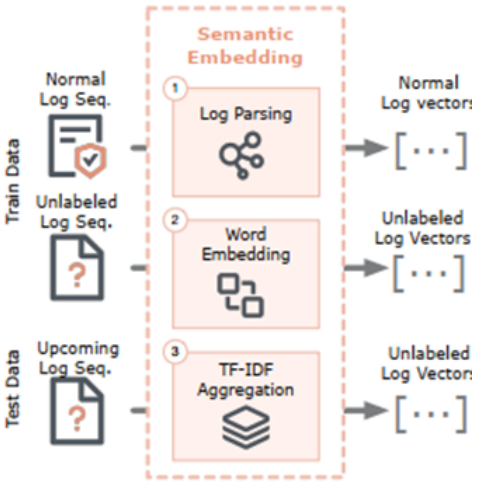
تجزیه پیام‌های سیستمی: *در این بخش با استفاده از تجزیه‌گر Drain رویدادهای سیستمی از پیام‌های سیستمی استخراج می‌شوند.*

بدست آوردن بازنمایی: *با توجه به شباهت توالی‌های سیستمی و جملات در بحث پردازش زبان طبیعی، PLELog به هر توالی سیستمی همانند یک جمله و به رویدادهای موجود در آن همانند لغات نگاه می‌کند. با این دیدگاه عملیات پیش‌پردازش بر روی لغات و جملات مثل حذف جداکننده‌ها، عملگرها، نشانه‌ها و ایست‌واژه‌ها انجام می‌شود. سپس با استفاده از الگوریتم FastText[۳۸] بردارسازی انجام می‌شود.*

وزن‌دهی: *در این بخش بر روی بردارهای حاصله از مرحله قبل TF-IDF اعمال می‌شود تا به وزن‌دهی مناسب از رویدادها در هر توالی برسیم.*

بعد از انجام این سه گام بردارهای مفهومی برای مرحله بعد آماده هستند. شکل ۶ این بخش و گام‌های آن را نشان می‌دهد.

شکل ۶: بخش بازنمایی مفهومی در پژوهش PLELog[۳۱]



۲. **پیش‌بینی برچسب احتمالی**[60]: *در این بخش در طی دوگام بردارهای مفهومی ورودی گروه‌بندی و برچسب‌دهی می‌شوند. اما این برچسب‌ها، برچسب‌های قطعی نیستند و به صورت احتمالی برچسب‌دهی می‌شوند که به توضیح این گام‌ها در ادامه می‌پردازیم.*

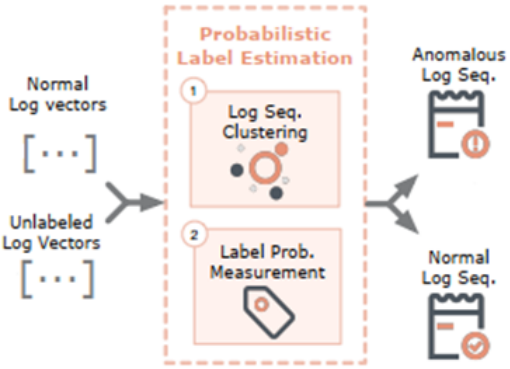
گروه‌بندی توالی‌های سیستمی: *باتوجه به اینکه پیام‌های سیستمی گروه‌بندی‌های متفاوتی دارند و این گروه‌بندی‌ها در طول زمان دستخوش تغییر می‌شوند، لذا استفاده از الگوریتم‌هایی مانند K-means که نیاز به مشخص کردن تعداد گروه‌ها در ابتدا دارند موثر نمی‌باشد. در این پژوهش از الگوریتم HDBSCAN[۳۹] استفاده شده است که یک الگوریتم مبتنی بر تراکم[61] است و نیاز به مشخص کردن تعداد گروه‌ها ندارد.*

تخصیص برچسب‌های احتمالی: *بعد از گروه‌بندی، نمونه‌ها در این بخش برچسب‌دهی می‌شوند. راه معمول برچسب‌دهی نمونه‌ها در پژوهش‌های پیشین تخصیص برچسب قطعی ۰ و ۱ به آن‌هاست. اما در PLELog برچسب‌دهی نمونه‌ها بصورت احتمالی است و این برچسب‌دهی کمک می‌کند تا نمونه‌هایی که به احتمال کمی به هریک از گروه‌ها تعلق دارند بیشتر مورد توجه قرار بگیرند و در فرآیند تشخیص ناهنجاری با تمرکز بر این نمونه‌های با احتمال پایین، دقت بیشتری در تشخیص ناهنجاری حاصل شود. همچنین باتوجه به نیمه‌نظارتی بودن این روش وجود تعدادی از داده‌ها با برچسب درست می‌تواند تاثیر بسیار مثبتی در تخصیص درست این برچسب‌های احتمالی داشته باشد.*

در شکل ۷ این بخش نمایش داده شده است.

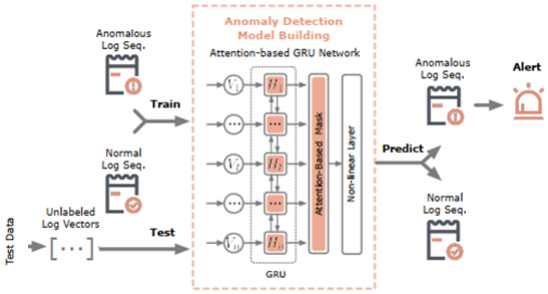
[58] generalization [59] semantic embedding [60] Probabilistic Label Estimation [61] density based

شکل ۷: بخش Probabilistic Label Estimation در پژوهش PLELog[۳۱]



۳. **مدل تشخیص ناهنجاری**[۶۲]: *در این بخش نمونه‌ها و برچسب‌های احتمالی آنها بعنوان ورودی در نظرگرفته می‌شوند. ابتدا با استفاده از واحد GRU[۴۰] روابط بین آنها استخراج می‌شود و سپس در واحد attention توسط سازوکار توجه*[۶۳] *با تمرکز بر روی نمونه‌هایی که برچسب احتمالی آنها به ازای هریک از گروه‌ها مقدار پایینی است، فرآیند تشخیص ناهنجاری صورت می‌گیرد.*
در شکل ۸ این بخش و اجزای آن قابل مشاهده هستند.

شکل ۸: بخش مدل تشخیص ناهنجاری در پژوهش PLELog[۳۱]



در جدول ۴ جمع‌بندی روش‌های موجود در رویکردهای بدون‌نظارت و نیمه‌نظارتی به همراه دقت‌های موجود و زمان مورد نیاز برای آموزش و ارزیابی آنها بر روی مجموعه دادگان HDFS قابل مشاهده هستند.

جدول ۴: نتایج پیاده‌سازی رویکردهای بدون‌نظارت بر روی مجموعه دادگان HDFS

Method	Precision	Recall	F1-Score	Training	Testing
LogRobust	۰.۹۸	۱.۰۰	۰.۹۹	۱h ۲۰min	۴۹s
IM	۱.۰۰	۰.۸۸	۰.۹۴	–	–
PCA	۰.۶۳	۰.۹۶	۰.۷۷	۱۸min	۱s
LogCluster	۱.۰۰	۰.۸۳	۰.۹۱	۱۹min	۲۳s
DeepLog	۰.۹۴	۰.۹۰	۰.۹۲	۱h ۵۰min	۲۰min
LogAnomaly	۰.۸۶	۰.۸۹	۰.۸۷	۴h ۴۰min	۴۷min
PLELog	۰.۹۵	۰.۹۶	۰.۹۵	۴۳min	۴۲s

جدول ۵: نتایج پیاده‌سازی بر روی مجموعه دادگان BGL

Method	Precision	Recall	F1-Score	Training	Testing
LogRobust	۰.۹۹۹	۰.۹۹۸	۰.۹۹۹	۳۰min	۱۵s
PCA	۰.۴۴۸	۰.۳۳۳	۰.۳۸۲	۸s	۱s
LogCluster	۰.۹۱۴	۰.۶۴۲	۰.۷۵۴	۱۹min	۲۳s
DeepLog	۰.۱۳۸	۰.۶۳۰	۰.۲۲۷	۴۱s	۴۰s
LogAnomaly	۰.۱۷۹	۰.۹۹۸	۰.۳۰۳	۴h ۲۰min	۳۹min
PLELog	۰.۹۶۵	۰.۹۹۹	۰.۹۸۲	۲۴min	۱۰s

۴ روش پیشنهادی

در بررسی انجام شده در پژوهش‌های پیشین موجود در تمامی رویکردها، مدل PLELog با کسب دقت‌های بالاتر نسبت به تمامی روش‌های بدون نظارت مقایسه شده در جدول ۴ و نداشتن مشکلات روش‌های بانظارت نظیر LogRobust بعنوان روش برگزیده از بین پژوهش‌های پیشین با مدل جدید پیشنهادی مورد مقایسه قرار می‌گیرد و در ادامه به تشریح مدل پیشنهادی می‌پردازیم و سپس آنرا با PLELog مقایسه می‌کنیم و در انتها تمهیدات علیه اعتبار روش پیشنهادی را بررسی می‌کنیم. فقط این نکته قابل ذکر است با توجه به نبود سخت‌افزار کافی جهت ارزیابی و پیاده سازی مدل پیشنهادی بر روی مجموعه دادگان HDFS از مجموعه دادگان BGL استفاده شده است و نتایج مدل با نتایج پژوهش‌های پیشین در این مجموعه نیز در جدول ۶ مقایسه شده است.

۱.۴ مدل پیشنهادی:

معرفی ترنسفورمرها: در سال‌های اخیر مدل های مبتنی بر ترنسفورمر[۶۴] در زمینه‌های مختلف نظیر پردازش زبان طبیعی بسیار مورد توجه قرار گرفته‌اند. از عمده دلایل توجه پژوهش‌گران به این مدل‌ها می‌توان موارد زیر را نام برد:

۱. بهره‌گیری از پردازش موازی[۶۵]
۲. استفاده از سازوکار توجه
۳. موازی سازی آسان بلوک‌های ترنسفورمر
۴. وجود مدل‌های از پیش آموزش دیده شده[۶۶]

بعنوان دو مدل از پیش آموزش دیده شده می‌توان به ALBERT[۴۱] و BERT[۴۲] اشاره کرد. مدل ALBERT به دلیل استفاده از تکنیک‌های توجه تنک[۶۷] و توجه بلوکی[۶۸] توانسته است حجم بسیار کمتری از سایر نمونه‌های ترنسفرمر داشته باشد که آنرا مناسب برای استفاده سخت‌افزارهای بیشتری می‌کند. مدل ALBERT بر روی مجموعه دادگان زبان انگلیسی آموزش دیده شده است و از همین جهت با توجه به تشابه گزارش‌های سیستمی به جملات انگلیسی می‌تواند انتخاب مناسبی برای استفاده در کارهای مرتبط به گزارش‌های سیستمی باشد.

معرفی روش پیشنهادی: در پژوهش PLELog بخش اول یعنی بخش بازنمایی مفهومی که در پژوهش LogRobust هم مشابه همین روش استفاده شده است، ابتدا پیام‌های سیستمی توسط تجزیه‌گر Drain تجزیه می‌شوند، سپس توسط الگوریتم FastText بازنمایی آن‌ها بدست می‌آید و بعد از آن توسط TF-IDF وزن‌دهی می‌شوند. اما در این بخش برای بدست آوردن بردارهای مفهومی می‌توانیم از ترنسفورمر استفاده کنیم زیرا مزایای زیر را به همراه دارد:

۱. استفاده از ترنسفورمر باعث می‌شود که ترتیب رخداد کلمات در یک الگو و کلمات موجود در آن در نظر گرفته شود بعنوان مثال دو الگوی پیام سیستمی زیر را در نظر بگیرید:

NodeCard VPD chip is not accessible

NodeCard VPD chip is accessible

[62] anomaly detection model [63] attention mechanism [64] transformer [65] parallel processing [66] pretrained [67] sparse attention [68] block attention

در این دو مثال مدل مبتنی بر ترنسفورمر قادر است تا مرجع not را تشخیص دهد ولی مدل مبتنی بر FastText و TF-IDF این توانایی را ندارد‌لذا در این موارد به دلیل توجه مدل‌های مبتنی بر ترنسفورمر به مفهوم جملات و نقش کلمات در جمله عملکرد بهتری دارند.

۲. مدل PLELog از مجموعه بردارهای glove.6B.300d استفاده می‌کند که حجم آن‌ها حدودا 989MB است ولی مدل ALBert حجمی برابر با 78MB دارد که این امکان را به ما می‌دهد تا بردارهای بیشتری را به محل اجرا آورده و نگرانی کمتری در فاز اجرا داشته باشیم.

لذا با توجه به دلایل بیان شده با جایگذاری ترنسفور به‌جای FastText و TF-IDF در بخش استخراج بردارهای مفهومی خروجی‌هایی به اندازه خروجی‌های مدل ALBERT به اندازه ۷۶۸ خواهیم داشت که با استفاده از الگوریتم convergence analusis[۴۳] آنها را به ۵۰ کاهش می‌دهیم تا بتوانیم ورودی‌های بخش بعدی و گروه‌بندی بر اساس الگوریتم HDBSCAN را انجام دهیم و این فرآیند سریع‌تر اجرا شود. از این مرحله به بعد مراحل مشابه روش ارائه شده در پژوهش PLELog طی می‌شوند.

پیاده‌سازی و نتایج: در مدل پیشنهادی به دلیل استفاده از داده‌های کمتری که در اختیار داریم و همچنین حجم زیاد مدل BERT از مدل ALBERT استفاده شده است و انتظار می‌رود نتایج بهتری داشته باشیم. نتایج پیاده‌سازی‌های روش پیشنهادی و پژوهش پایه در جدول ۶ قابل مشاهده هستند. این پیاده‌سازی بر دو میلیون از پیام‌های سیستمی مجموعه دادگان BGL صورت گرفته است که در نهایت افزایش دقتی در حدود ۲ درصد را برای مدل مبتنی بر ترنسفورمر ALBERT در بر داشته است.

جدول ۶: نتایج پیاده‌سازی بر روی مجموعه دادگان BGL

Method	Precision	Recall	F1-Score	Training	Testing
Our PLELog	۰.۹۶۵۳	۰.۹۶۵۹	۰.۹۶۵۶	۱۲min	۵s
Our BERT model	۰.۹۴۰۷	۰.۹۹۹۷	۰.۹۶۹۳	۱۵min	۱۶s
Our ALBERT model	۰.۹۸۱۷	۰.۹۷۸۶	۰.۹۸۰۱	۱۴min	۶s

۲.۴ پیشنهادات جهت کارهای آتی:

۱. با توجه به حجیم بودن مدل BERT شاهد بودیم که در این پیاده‌سازی دقت قابل توجهی ندارد، اما این به دلیل ناکارآمدی این مدل نیست بلکه به دلیل آموزش کم و مجموعه دادگان محدود برای این مدل است، توصیه می‌شود تا این مدل برای چندین مجموعه دادگان در این حوزه آموزش ببیند و سپس مورد استفاده قرار گیرد.

۲. استفاده از مدل‌های مبتنی بر ترنسفورمر در بخش‌های دیگر مدل پیشنهادی ازجمله بخش تجزیه‌گر و بخش تشخیص ناهنجاری. در بخش تجزیه‌گر می‌تواند جایگزین Drain باشد و در بخش تشخیص ناهنجاری می‌تواند جایگزین Bi-LSTM GRU Attention Based باشد که در کارهای مشابه در حوزه پردازش زبان طبیعی، مدل‌های مبتنی بر ترنسفورمر در این حوزه پیشتاز هستند.

۳. آموزش مدل ALBERT بر روی مجموعه دادگان‌های بیشتری می‌تواند باعث بهبود نتایج این مدل نیز شود. در حال حاضر در این روش پیشنهادی، از مدل از پیش آموزش دیده شده بر روی زبان انگلیسی استفاده شده است و قطعا با آموزش و تعدیل و تنظیم[۶۹] این مدل بر روی مجموعه دادگان پیام‌های سیستمی نتایج بهتری حاصل خواهد شد.

۳.۴ تمهیدات علیه اعتبار:

۱. **اعتبار درونی**[۷۰]: *با توجه به استفاده از یک مدل مبتنی بر ترنسفورمر، به دلیل اینکه این مدل‌ها نیاز دارند تا بر روی داده‌های زیادی آموزش ببینند یا اصطلاحا data hungry هستند، لذا با وجود دقت‌های مناسب این مدلها،این آموزش از ابتدا برای این مدل‌ها می‌تواند هزینه زمانی و محاسباتی زیادی داشته باشد و برای حل این مشکل از مدل‌های از پیش آموزش دیده شده بر روی زبان انگلیسی استفاده شد.*

۲. **اعتبار بیرونی**[۷۱]: *در پژوهش پیشنهادی از یک رویکرد نیمه‌نظارتی استفاده شده است و تعداد کمی از داده‌ها دارای برچسب هستند و به میزان قابل توجهی در جهت‌دهی مدل برای کسب معیارهای مناسب تاثیر دارند. حال اگر برچسب زدن این داده‌ها به‌دلیل عدم تخصص یا بی دقتی متخصص همراه با خطا باشد کارایی مدل افت می‌کند، لذا از مجموعه دادگان‌های آماده و مورد تایید در این حوزه استفاده شده است تا این مورد به‌وجود نیاید. همچنین برای استفاده مدل در دنیای واقعی توصیه می‌شود فرآیند برچسب‌دهی توسط چند متخصص صورت گیرد تا احتمال ایجاد خطا به حداقل برسد.*

۳. **اعتبار ساخت**[۷۲]: *جهت ارزیابی مدل پیشنهادی، از معیارهای رایج در پژوهش‌های پیشین استفاده شده است اما توصیه می‌شود از معیار ارزیابی AUROC*[۷۳] *نیز در کنار این معیارها به دلیل عدم تعادل در کلاس‌های*[۷۴] *داده‌های عادی و ناهنجار نیز استفاده شود.*

۵ نتیجه‌گیری

در طول سال‌های اخیر، پژوهش‌های ارزشمندی در زمینه تشخیص ناهنجاری مبتنی بر گزارش‌های سیستمی صورت گرفته است اما این روش‌ها اکثرا قابل استفاده در دنیای واقعی و محیط عملی نیستند. دلیل این امر همانطور که بررسی شد مشکلات مربوط به رویکردهای نظارتی نظیر زمان‌بر بودن برچسب‌زنی داده‌ها و نیاز به فرد متخصص برای انجام این‌کار و حجم زیاد داده‌های مورد نیاز برای آموزش مدل هوشمند در این رویکرد می‌باشد. در رویکردهای بدون‌نظارت نیز مشکلاتی نظیر کارایی پایین و فراموشی تاریخچه وجود دارد. بهترین نتیجه تا به اکنون در رویکردهای نیمه‌نظارتی و در روش PLELog ارائه شده است که به دلیل استفاده از تعداد کمی داده برچسب‌دار مشکلات روش‌های بانظارت را ندارد و همچنین به دلیل استفاده از بردارسازی مفهومی و سازوکار توجه مشکلات رویکردهای بدون‌نظارت را تاحد خوبی

[69] fine tune [70] internal validity [71] external validity [72] construct validity [73] Area Under the Receiver Operating Characteristic curve [74] class imblance

رفع کرده است. اما در بردارسازی مفهومی استفاده شده در این پژوهش هم ایراداتی نظیر توجه نکردن به نقش کلمات در جملات وجود دارد که استفاده از یک مدل مبتنی بر ترنسفورمر می‌تواند این ایرادات را بهبود بخش که در روش پیشنهادی با استفاده از همین ایده روشی مبتنی بر ترنسفور ALBERT جهت ساخت بردارهای مفهومی و درنهایت معیارهای ارزیابی بهتر نسبت به تمامی روش‌های پیشین از جمله PLELog حاصل شده است که در حوزه تشخیص ناهنجاری مبتنی بر گزارش‌های سیستمی بسیار ارزشمند است.

مراجع

[1] Chen, Lianping. Continuous delivery: Huge benefits, but challenges too. *IEEE software*, 32(2):50–54, 2015.

[2] Humble, Jez and Farley, David. *Continuous delivery: reliable software releases through build, test, and deployment automation*. Pearson Education, 2010.

[3] Kabinna, Suhas, Bezemer, Cor-Paul, Shang, Weiyi, Syer, Mark D, and Hassan, Ahmed E. Examining the stability of logging statements. *Empirical Software Engineering*, 23(1):290–333, 2018.

[4] Xu, Wei. *System problem detection by mining console logs*. University of California, Berkeley, 2010.

[5] Du, Min, Li, Feifei, Zheng, Guineng, and Srikumar, Vivek. Deeplog: Anomaly detection and diagnosis from system logs through deep learning. in *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pp. 1285–1298, 2017.

[6] He, Pinjia, Zhu, Jieming, He, Shilin, Li, Jian, and Lyu, Michael R. An evaluation study on log parsing and its use in log mining. in *2016 46th annual IEEE/IFIP international conference on dependable systems and networks (DSN)*, pp. 654–661. IEEE, 2016.

[7] He, Pinjia, Zhu, Jieming, He, Shilin, Li, Jian, and Lyu, Michael R. Towards automated log parsing for large-scale log data analysis. *IEEE Transactions on Dependable and Secure Computing*, 15(6):931–944, 2017.

[8] Liang, Yinglung, Zhang, Yanyong, Xiong, Hui, and Sahoo, Ramendra. Failure prediction in ibm bluegene/l event logs. in *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pp. 583–588. IEEE, 2007.

[9] Xu, Wei, Huang, Ling, Fox, Armando, Patterson, David, and Jordan, Michael I. Detecting large-scale system problems by mining console logs. in *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*, pp. 117–132, 2009.

[10] Zhang, Xu, Xu, Yong, Lin, Qingwei, Qiao, Bo, Zhang, Hongyu, Dang, Yingnong, Xie, Chunyu, Yang, Xinsheng, Cheng, Qian, Li, Ze, et al. Robust log-based anomaly detection on unstable log data. in *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pp. 807–817, 2019.

[11] Oliner, Adam and Stearley, Jon. What supercomputers say: A study of five system logs. in *37th annual IEEE/IFIP international conference on dependable systems and networks (DSN'07)*, pp. 575–584. IEEE, 2007.

[12] He, Pinjia, Zhu, Jieming, Zheng, Zibin, and Lyu, Michael R. Drain: An online log parsing approach with fixed depth tree. in *2017 IEEE international conference on web services (ICWS)*, pp. 33–40. IEEE, 2017.

[13] Vaarandi, Risto. A data clustering algorithm for mining patterns from event logs. in *Proceedings of the 3rd IEEE Workshop on IP Operations & Management (IPOM 2003)(IEEE Cat. No. 03EX764)*, pp. 119–126. Ieee, 2003.

[14] Jiang, Zhen Ming, Hassan, Ahmed E, Flora, Parminder, and Hamann, Gilbert. Abstracting execution logs to execution events for enterprise applications (short paper). in *2008 The Eighth International Conference on Quality Software*, pp. 181–186. IEEE, 2008.

[15] Makanju, Adetokunbo AO, Zincir-Heywood, A Nur, and Milios, Evangelos E. Clustering event logs using iterative partitioning. in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1255–1264, 2009.

[16] Fu, Qiang, Lou, Jian-Guang, Wang, Yi, and Li, Jiang. Execution anomaly detection in distributed systems through unstructured log analysis. in *2009 ninth IEEE international conference on data mining*, pp. 149–158. IEEE, 2009.

[17] Nagappan, Meiyappan and Vouk, Mladen A. Abstracting log lines to log event types for mining software system logs. in *2010 7th IEEE Working Conference on Mining Software Repositories (MSR 2010)*, pp. 114–117. IEEE, 2010.

[18] Tang, Liang, Li, Tao, and Perng, Chang-Shing. Logsig: Generating system events from raw textual logs. in *Proceedings of the 20th ACM international conference on Information and knowledge management*, pp. 785–794, 2011.

[19] Mizutani, Masayoshi. Incremental mining of system log format. in *2013 IEEE International Conference on Services Computing*, pp. 595–602. IEEE, 2013.

[20] Vaarandi, Risto and Pihelgas, Mauno. Logcluster-a data clustering and pattern mining algorithm for event logs. in *2015 11th International conference on network and service management (CNSM)*, pp. 1–7. IEEE, 2015.

[21] Shima, Keiichi. Length matters: Clustering system log messages using length of words. *arXiv preprint arXiv:1611.03213*, 2016.

[22] Hamooni, Hossein, Debnath, Biplob, Xu, Jianwu, Zhang, Hui, Jiang, Guofei, and Mueen, Abdullah. Logmine: Fast pattern recognition for log analytics. in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pp. 1573–1582, 2016.

[23] Du, Min and Li, Feifei. Spell: Streaming parsing of system event logs. in *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 859–864. IEEE, 2016.

[24] Messaoudi, Salma, Panichella, Annibale, Bianculli, Domenico, Briand, Lionel, and Sasnauskas, Raimondas. A search-based approach for accurate identification of log message formats. in *2018 IEEE/ACM 26th International Conference on Program Comprehension (ICPC)*, pp. 167–16710. IEEE, 2018.

[25] Breier, Jakub and Branišová, Jana. Anomaly detection from log files using data mining techniques. in *Information Science and Applications*, pp. 449–457. Springer, 2015.

[26] Lou, Jian-Guang, Fu, Qiang, Yang, Shenqi, Xu, Ye, and Li, Jiang. Mining invariants from console logs for system problem detection. in *2010 USENIX Annual Technical Conference (USENIX ATC 10)*, 2010.

[27] Xu, Wei, Huang, Ling, Fox, Armando, Patterson, David, and Jordan, Michael. Largescale system problem detection by mining console logs. *Proceedings of SOSP'09*, 2009.

[28] Joulin, Armand, Grave, Edouard, Bojanowski, Piotr, Douze, Matthijs, Jégou, Hérve, and Mikolov, Tomas. Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*, 2016.

[29] Salton, Gerard and Buckley, Christopher. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

[30] Huang, Zhiheng, Xu, Wei, and Yu, Kai. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

[31] Yang, Lin, Chen, Junjie, Wang, Zan, Wang, Weijing, Jiang, Jiajun, Dong, Xuyuan, and Zhang, Wenbin. Semi-supervised log-based anomaly detection via probabilistic label estimation. in *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pp. 1448–1460. IEEE, 2021.

[32] Lin, Qingwei, Zhang, Hongyu, Lou, Jian-Guang, Zhang, Yu, and Chen, Xuewei. Log clustering based problem identification for online service systems. in *Proceedings of the 38th International Conference on Software Engineering Companion*, pp. 102–111, 2016.

[33] Gower, John C and Ross, Gavin JS. Minimum spanning trees and single linkage cluster analysis. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 18(1):54–64, 1969.

[34] Hochreiter, Sepp and Schmidhuber, Jürgen. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[35] Meng, Weibin, Liu, Ying, Zhu, Yichen, Zhang, Shenglin, Pei, Dan, Liu, Yuqing, Chen, Yihao, Zhang, Ruizhi, Tao, Shimin, Sun, Pei, et al. Loganomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs. in *IJCAI*, vol. 19, pp. 4739–4745, 2019.

[36] Mikolov, Tomas, Chen, Kai, Corrado, Greg, and Dean, Jeffrey. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[37] Miller, George A. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[38] Pennington, Jeffrey, Socher, Richard, and Manning, Christopher D. Glove: Global vectors for word representation. in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[39] McInnes, Leland, Healy, John, and Astels, Steve. hdbscan: Hierarchical density based clustering. *J. Open Source Softw.*, 2(11):205, 2017.

[40] Cho, Kyunghyun, Van Merriënboer, Bart, Bahdanau, Dzmitry, and Bengio, Yoshua. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[41] Lan, Zhenzhong, Chen, Mingda, Goodman, Sebastian, Gimpel, Kevin, Sharma, Piyush, and Soricut, Radu. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

[42] Devlin, Jacob, Chang, Ming-Wei, Lee, Kenton, and Toutanova, Kristina. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[43] Oja, Erkki and Yuan, Zhijian. The fastica algorithm revisited: Convergence analysis. *IEEE transactions on Neural Networks*, 17(6):1370–1381, 2006.