시스템 이상징후 탐지 연구 및 테스트베드 변경 계획

경북대학교 권영우

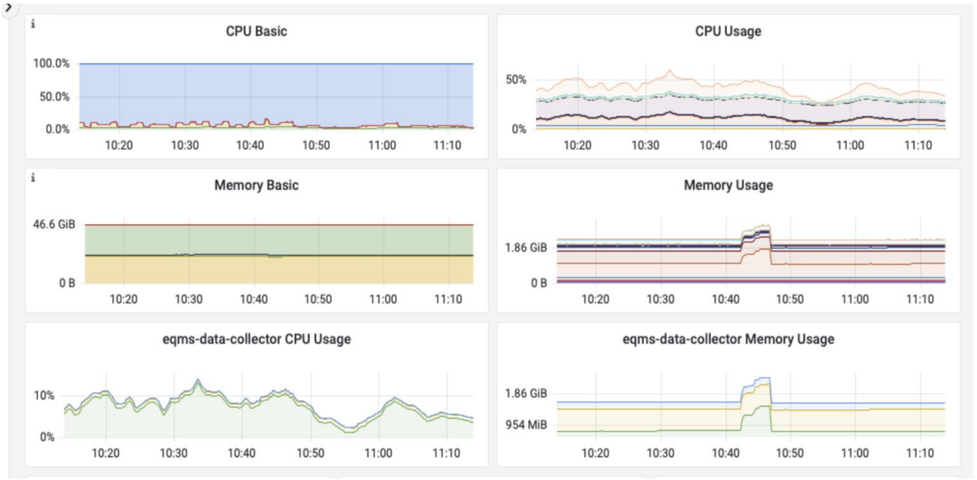
주요 연구 내용

- 주요 연구 내용 요약
  - 이상징후 탐지 연구
    - 로그기반 시스템 이상 징후 탐지
    - 딥러닝 기반 오류 탐지
  - 재난 감지 및 대응 연구
    - 지식 그래프 기반의 재난 징후/상황 인지
    - 지진조기경보 체계 개발
- 향후 연구 계획
  - CrowdQuake 구조 변경
  - Serverless 함수 추출 및 변환 연구

시스템 이상징후 탐지 연구

시스템 이상징후 탐지

- 시스템 이상징후
  - 비정상 CPU, 메모리, 네트워크 사용
  - 비정상 동작, 예외 발생, 시스템 크래쉬 등
- 로그 예




```
datacollector_1    | Message forwarded of size 713        :43:09.039 / 01227501105 / a1c602042212050143090000 / bytes: 7
datacollector_1    | Message forwarded of size 713        14
datacollector_1    | Message forwarded of size 713        eqms_sensor_sensor.1.ad8aqawa8dgz@hdc-node1    | 2022-12-05 01
datacollector_1    | Message forwarded of size 713        :43:10.039 / 01227501105 / a1c502042212050143100000 / bytes: 7
datacollector_1    | Message forwarded of size 714        13
datacollector_1    | Message forwarded of size 713        eqms_sensor_sensor.1.ad8aqawa8dgz@hdc-node1    | 2022-12-05 01
datacollector_1    | Message forwarded of size 713        :43:11.04 / 01227501105 / a1c602042212050143110000 / bytes: 71
datacollector_1    | Message forwarded of size 713        4
datacollector_1    | Message forwarded of size 713
datacollector_1    | Message forwarded of size 713        ──────────────────────────────────────────────────────────────
datacollector_1    | Message forwarded of size 713        01227501104,1670202374000,1670169975125,9
datacollector_1    | Message forwarded of size 713        01227501104,1670202376000,1670169977084,9
datacollector_1    | Message forwarded of size 714        01227501104,1670202379000,1670169980119,9
datacollector_1    | Message forwarded of size 713        01227501104,1670202380000,1670169981100,9
datacollector_1    | Message forwarded of size 714        01227501104,1670202384000,1670169985101,9
datacollector_1    | Message forwarded of size 713        01227501104,1670202385000,1670169986104,9
datacollector_1    | Message forwarded of size 713        01227501104,1670202387000,1670169988114,9
datacollector_1    | Message forwarded of size 714        01227501104,1670202389000,1670169990086,9
datacollector_1    | Message forwarded of size 714        01227501104,1670202394000,1670169995093,9
datacollector_1    | Message forwarded of size 713        01227501104,1670202397000,1670169998131,9
datacollector_1    | Message forwarded of size 713        01227501104,1670202411000,1670170012097,9
datacollector_1    | Message forwarded of size 714        01227501104,1670202414000,1670170015126,9
datacollector_1    | Message forwarded of size 714        01227501104,1670202421000,1670170022097,9
datacollector_1    | Message forwarded of size 713        01227501104,1670202424000,1670170025126,9
datacollector_1    | Message forwarded of size 714        01227501104,1670202425000,1670170026095,9
datacollector_1    | Message forwarded of size 713        01227501104,1670202447000,1670170048120,9
datacollector_1    | Message forwarded of size 714        01227501104,1670202460000,1670170061123,9
```

```
s=Hamburger,5_Cola,2_Chicken,10.5
s=Rice,1.2_Chicken Soup,2.5
s=Big Burger,1.2_Bone Soup,2.5
s=Big Burger,1.2_Bone Soup,2.5
s=Big Mac,2.2_McChicken,1.5
s=Big Burger,1.2_Bone Soup,2.5
s=French Fries,1.2_Vegetable Seafood Soup,2.5
s=Hot Chocolate,1.2_Pineapple Pie,2.5
s=Karubi Beef,1.2_ Low Fat Yogurt Blackberry,2.5
s=Pork Chop with rice,9.5_Egg Soup,3.2
s=Beef with rice,9.5_Soup,3.7_Pork pickled mustard green noodles,10
s=Seafood noodles,9.5_Glutinous rice,0.9_Dumplings,5.5
s=Spring rolls,1.5_Vegetable soup,0.8_Rice and vegetable roll,1
s=Spicy hot noodles,5_Soup,3.7_Oily bean curd,2
[SSO Service][Init Account] Before:4d2a46c7-71cb-4cf1-b5bb-b68406d9da6f
[SSO Service][Init Account] After:4d2a46c7-71cb-4cf1-b5bb-b68406d9da6f
[SSO Service][Init Account] Before:1d1a11c1-11cb-1cf1-b1bb-b11111d1da1f
[SSO Service][Init Account] After:1d1a11c1-11cb-1cf1-b1bb-b11111d1da1f
[Consign price service] [Init data operation]
[Consign Price Service][Create New Price Config]
[Route Service] Create And Modify Start:shanghai End:taiyuan
[Route Service] Create And Modify Start:nanjing End:beijing
[Route Service] Create And Modify Start:taiyuan End:shanghai
[Route Service] Create And Modify Start:shanghai End:taiyuan
```

로그 기반의 Anomaly 탐지

- **로그 삽입을 통한 시스템 상태 정보 수집 (실패)**
  소스코드 수정 없이 로그 생성 코드 추가
  - Java: Bytecode Instrumentation (BCI)
  - Python: Meta-Object protocol + Dynamic Typing
  - 함수 시작/종료 지점, 매개변수, 인수, 반환값, 실행시간, print 문 등
- **피드백 기반의 로그의 양 조절 (실패)**
  - 함수의 중요 수준, 시스템의 자원 사용량, 이상징후의 심각도에 따라 출력되는 로그의 양 조절
  - 시스템의 이상 정도를 점수로 계산
    - 이벤트 밀도: 일정 시간 동안 시스템의 상태 변화량

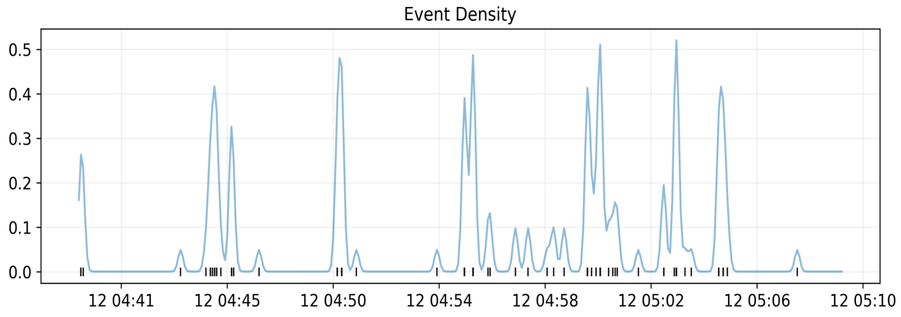


Fig. Processed Event stream using Kernel Density Estimation

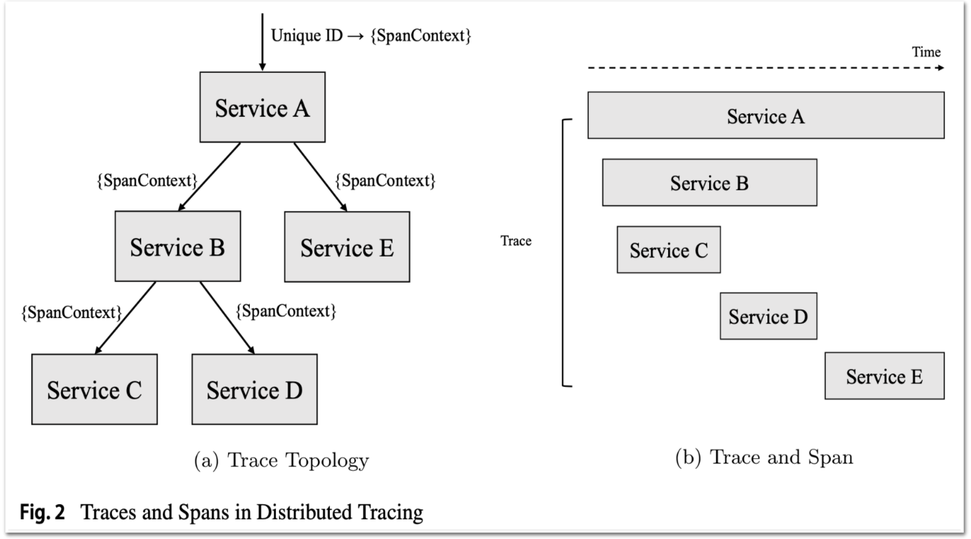


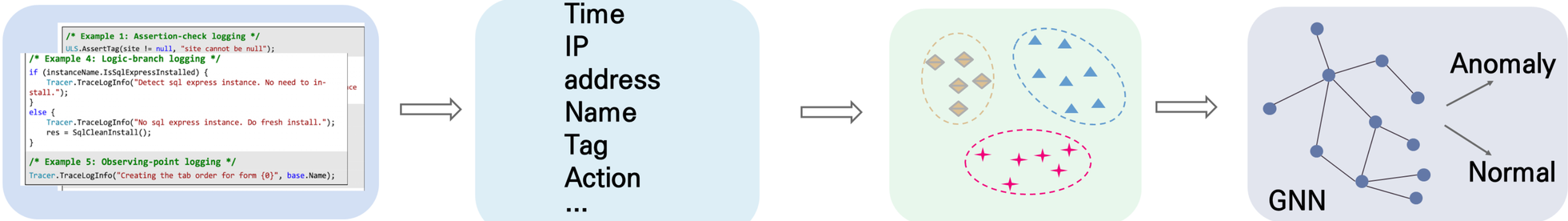
Fig. 2 Traces and Spans in Distributed Tracing

```
{'context': {'span_id': -8820367226001093835,
             'thread_id': '53',
             'trace_id': 2766762858404346241},
 'elapsed_time': datetime.timedelta(microseconds=139),
 'end_time': datetime.datetime(2022, 12, 13, 6, 9, 46, 494053, tzinfo=tzutc()),
 'events': [{'attributes': ['@parameter0: io.netty.channel.Channel [id: '
                            '0xb62100a7, L:/192.168.64.2:28000 - '
                            'R:/155.230.118.234:44812]'],
             'name': 'Method Start',
             'timestamp': datetime.datetime(2022, 12, 13, 6, 9, 46, 493914,
tzinfo=tzutc())},
            {'attributes': ['java.lang.Integer -1239351129'],
             'name': 'Method end',
             'timestamp': datetime.datetime(2022, 12, 13, 6, 9, 46, 494053,
tzinfo=tzutc())}],
 'name': '<com.finedigital.bean.SensorEventHandler: java.lang.Integer '
         'getChannelId(io.netty.channel.Channel)> ',
 'parent_id': 3969049755867096819,
 'start_time': datetime.datetime(2022, 12, 13, 6, 9, 46, 493914, tzinfo=tzutc())}
```

(시스템) 로그 기반의 Anomaly 탐지

- Anomaly의 종류
  - Execution path anomaly
  - Parameter value anomaly
  - Performance anomaly
- 로그 데이터의 그래프화

Figure 2: The example of raw log parsing and graph construction.



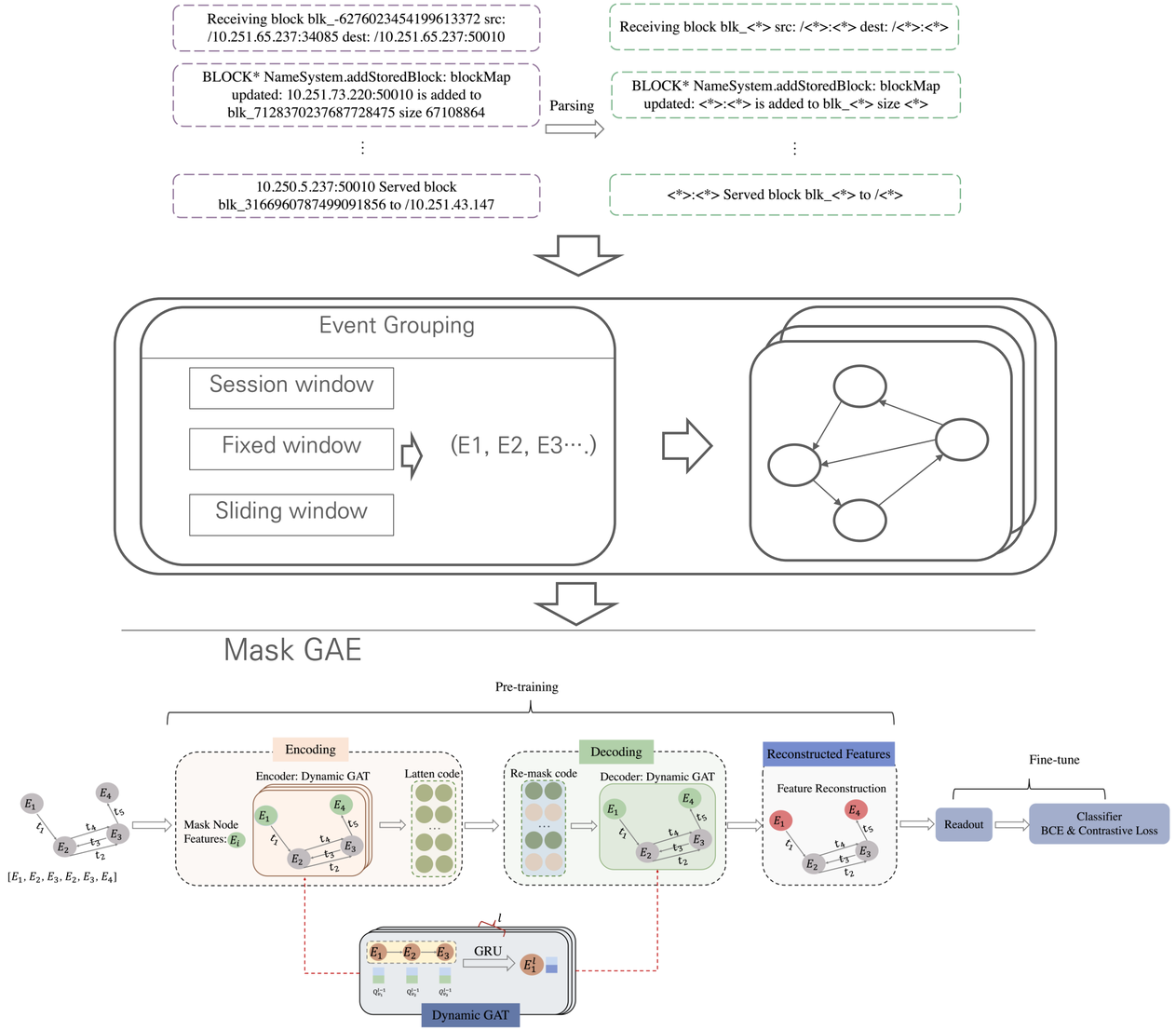
Log collection and preprocessing

Log parsing

Feature extraction

Anomaly detection

(시스템) 로그 기반의 Anomaly 탐지



Stage	Task	Implementation
Data collection	Traces	Log Hub
Data pre-process	Parsing and template extraction	Drain [1]
	Event grouping	Session/Fixed window
	Semantic embedding generation	
GNN	Masked GAE with GRU instead of the GraphConv layer	Randomly mask and re-mask input nodes for graph reconstruction
Evaluation	HDFS/BGL/Thunderbird data generation	
	Automate rebsult collection	

(시스템) 로그 기반의 Anomaly 탐지

- 오픈 데이터셋을 활용한 이상 실행 정보 탐지 테스트
  - 데이터셋: HDFS, BGL, Thunderbird
- 비교 모델
  - Statistical learning-based methods: PCA, IM
  - NLP-based methods: DeepLog, LogRobust
  - Graph-based methods: DeepTraLog

Dataset	#log Events	#Duration	#of Logs	#of Anomalies	#Anomaly Rate
HDFS	47	38.7 hours	1175629	16838	1.43%
BGL	377	7 months	4747963	348460	7.34%
Thunderbird	1758	244 days	2000000	6590	0.33%

Model	Datasets								
	HDFS			BGL			Thunderbird		
	Precision	Recall	F1_score	Precision	Recall	F1_score	Precision	Recall	F1_score
PCA	0.8583	0.6547	0.7766	0.2647	0.6337	0.3439	0.5337	0.4167	0.4973
IM	0.8185	0.7561	0.7923	0.2876	0.4734	0.4219	0.3367	0.4132	0.5431
DeepLog	0.9124	0.7841	0.8625	0.7452	0.9119	0.7612	0.4554	0.3786	0.6215
LogRobust	0.9335	0.9578	0.9623	0.9115	**0.9217**	0.8815	0.8118	0.8726	0.7967
DeepTraLog	0.8577	0.9126	0.8365	0.7742	0.8337	0.6953	0.6367	0.7331	0.6126
Pre-LogMGAE	**0.9798**	**0.9917**	**0.9815**	**0.9543**	0.8736	**0.9048**	**0.8367**	**0.9935**	**0.9042**

(시스템) 로그 기반의 Anomaly 탐지

Type	Experiment Setup	Model	Metrics	Precision/Recall/F1-score				
				HDFS	BGL		Thundermird	
				Session fixed	100logs Fixed	20logs fixed	100logs Fixed	20logs Fixed
Traditional Statistical Learning	• Training data =0.8 • Validation data = 0.2 • Epochs=20~60 • Batch size=128 • Semantic embedding=Bert • Average over 3 runs	PCA	Precision	0.8583	0.2647	0.4691	0.5337	0.5924
			Recall	0.6547	0.6337	0.6922	0.4167	0.6217
			F1	0.7766	0.3439	0.3398	0.4973	0.5549
			ROC					
		IM	Precision	0.8185	0.2876	0.4155	0.3367	0.3790
			Recall	0.7561	0.4734	0.6991	0.4132	0.3929
			F1	0.7923	0.4219	0.5334	0.5431	0.4859
			ROC					
Sequantial-Based		LogRobust	Precision	0.9335	0.9115	0.9104	0.8118	0.8213
			Recall	0.9578	**0.9217**	0.9147	0.8726	0.8813
			F1	0.9623	0.8815	0.8913	0.7967	0.7822
			ROC					
		DeepLog	Precision	0.9124	0.7612	0.7513	0.4554	0.4813
			Recall	0.7841	0.9119	0.8781	0.3786	0.5317
			F1	0.8625	0.7452	0.7645	0.6215	0.6771
			ROC					
Graph-Based		DeepTraLog	Precision	0.8577	0.7742	0.7617	0.6367	0.5974
			Recall	**0.9126**	0.8337	0.8198	0.7331	0.6801
			F1	0.8365	0.6953	0.7344	0.6162	0.7855
			ROC					
		Our Method	Precision	0.9745 ±0.0025	0.9523 ±0.0016	0.9257 ±0.0010	0.8337 ±0.0016	0.8122 ±0.0032
			Recall	0.9857 ±0.0001	0.8716 ±0.0028	0.8688 ±0.0006	0.9935 ±0.0002	0.9472 ±0.0021
			F1	0.9735 ±0.0012	0.9028 ±0.0011	0.9064 ±0.0046	0.9022 ±0.0013	0.9033 ±0.0016

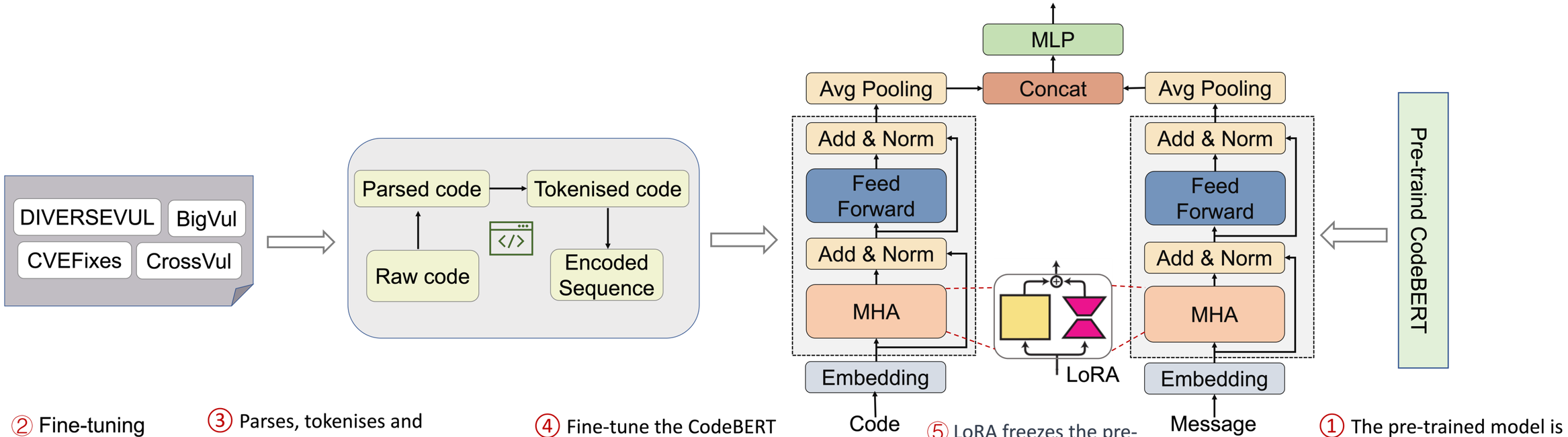
딥러닝 기반 코드 오류 탐지

- Language model 기반
- Graph 기반

Pre-trained Language Models for Code

- CodeBERT (EMNLP 2020)
- GraphCodeBERT (ICLR 2021)
- UniXcoder (ACL 2022)
- CodeReviewer (ESEC/FSE 2022)
- CodeExecutor (ACL 2023)
- LongCoder (ICML 2023)

Fine-tuning CodeBERT with LoRA



② Fine-tuning datasets

③ Parses, tokenises and encodes fine-tuning datasets based on the prebuilt vocabulary.

④ Fine-tune the CodeBERT model on a software vulnerability detection task across multiple vulnerability detection dataset.

⑤ LoRA freezes the pre-trained model's weights while adding trainable layers (rank decomposition matrix) to each Transformer block.

① The pre-trained model is ready for fine-tuning task.

LoRA [1] (Adaptation of Large Language Models) is a novel technology proposed by Microsoft to solve the problem of fine-tuning large language models. LLM with billions of parameters, such as GPT-3, require high-cost tuning to adapt to downstream tasks. LoRA proposes to freeze the weights of the pre-trained model and inject trainable layers (rank decomposition matrix) into each Transformer block. This greatly reduces the number of trainable parameters and GPU memory requirements.

[1] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., ... & Chen, W. (2021). Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

데이터 셋

Dataset Features:

- Real-world project without synthetic datasets, keep sample imbalance, large-size number, across-projects.
- Extracting code vulnerabilities in over 300 different open source C/C++ projects with over 150 different vulnerability types (including CVE numbers).

Table 1: Statistics about previous four datasets.

Dataset	Label Accuracy	Granularity	Projects	CWEs	Functions	Vul Func	Vul Func with CWE Information	Commits
DIVERSEVUL	High	Commit	809	150	379241	26635	21586	7861
CrossVul	High	Commit	489	107	134126	6884	6883	3009
CVEFixes	High	Commit	564	127	168089	8932	8343	3614
BigVul	High	Commit	348	91	264919	11823	8783	3754

Note: CrossVul and CVEFixes are multi-language datasets. here we just focus on C/C++ code. DiverseVul, a new C/C++ vulnerable source code dataset , it is 60% larger than the previous largest dataset for C/C++, and the most diverse compared to all previous datasets.

실험 및 평가

- Evaluate the models on the same test set for full-fine-tuning and fine-tuning with LoRA.

Datasets	Model	Trainable Parameters	Accuracy	Precision	Recall	F1_score	FPR
DIVERSEVUL	CodeBERT (Full fine-tuning)	125.0M	86.09%±3.5%	14.32%±4.7%	23.61%±2.8%	18.14%±2..6%	7.13%±2.3%
	CodeBERT (LoRA)	32.1M	91.14%±2.5%	23.72%±2.7%	30.14%±1.8%	**23.35%±3.2%**	10.13%±1.7%
CrossVul	CodeBERT (Full fine-tuning)	125.0M	90.47%±3.3%	16.74%±3.9%	19.34%±2.1%	12.18%±3.7%	8.05%±4.2%
	CodeBERT (LoRA)	32.1M	94.21%±2.7%	21.85%±2.1%	26.71%±1.8%	**19.84%±1.3%**	9.29%±2.2%

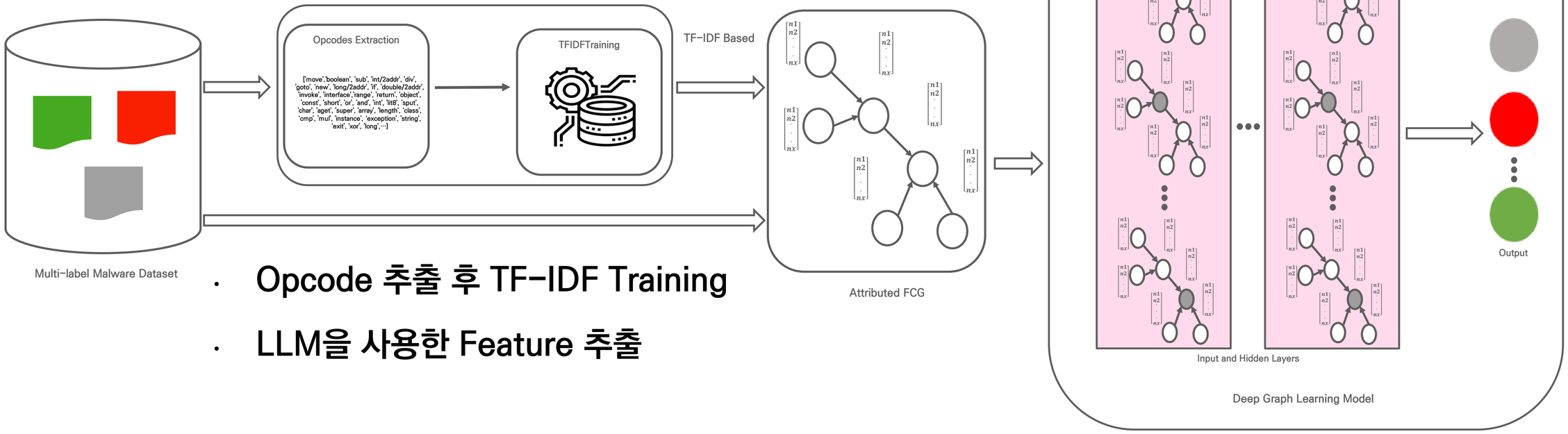
〈Table: Results with CodeBERT-base models on different datasets〉

- CodeBERT with LoRA presents better results on each dataset.
- The parameter quantity is significantly reduced compared to full fine-tuning, and the detection performance is better than full fine -tuning.

Model Family	Model Arch	Pretrain on Code	Pretrain on C/C++	Code-specific Pretrain task	Training Set	Test on Unseen Projects (%) Acc	Prec	Recall	F1	FPR
GNN	ReVeal				Previous	82.88	5.06	20.92	8.15	14.78
					Prev + DiverseVul	85.88	5.67	18.46	8.67	11.58
RoBERTa	RoBERTa				Previous	94.69	6.20	3.23	4.25	1.85
					Prev + DiverseVul	95.59	10.46	2.78	4.40	0.90
	CodeBERT	✔			Previous	94.94	9.53	4.57	6.17	1.64
					Prev + DiverseVul	**94.19**	**13.34**	**10.80**	**11.94**	**2.65**
	GraphCodeBERT	✔		✔	Previous	95.32	4.64	1.45	2.21	1.12
					Prev + DiverseVul	94.74	12.48	7.35	9.25	1.95

그래프 기반 악성코드 및 오류 탐지 연구

입력
- 소스코드
- 바이트코드

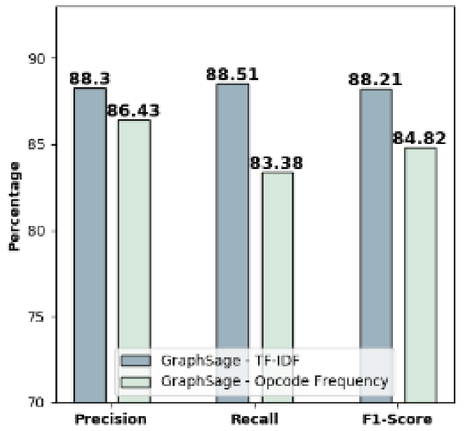


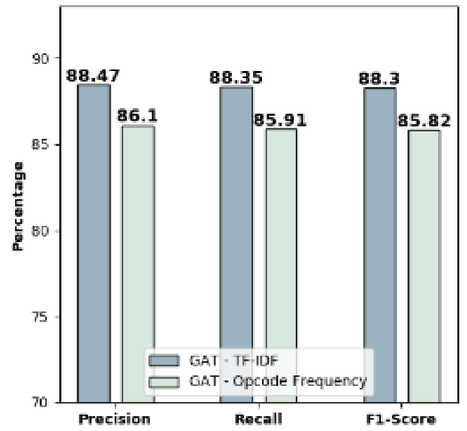
- 호출 그래프, PDG
- Opcode 추출 후 TF-IDF Training
- LLM을 사용한 Feature 추출
- 그래프 신경망 기반의 다중 분류
  - GCN, SageConv, GAT, GIN

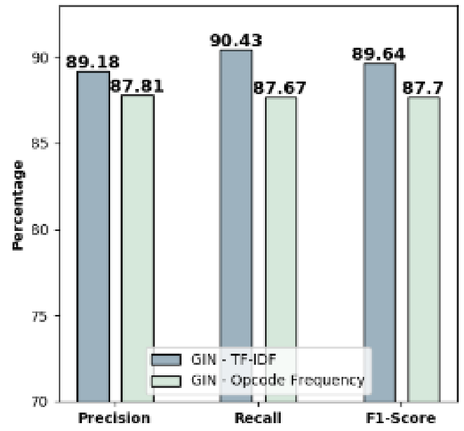
그래프 기반 악성코드 및 오류 탐지 연구

- 결과 요약
  - Precision, Recall, and F1 Score 측정
  - 성능 순위
    - GIN, GAT, SageConv, and GCN
- 향후 연구
  - 수집된 취약점 데이터셋 활용
  - 다른 언어에 적용
  - 모델 변경
    - Binary vs Muti-classification

Model	GraphConv			GraphSage			GATConv			GIN		
Class	Prec:	Rec :	F1:	Prec:	Rec :	F1:	Prec:	Rec :	F1:	Prec:	Rec :	F1:
Adware	62.20	93.95	74.85	87.22	91.58	89.34	91.01	86.51	88.70	87.54	94.47	90.87
Banking	86.66	95.79	91.00	91.58	96.40	93.92	88.05	90.36	89.19	94.15	92.03	93.08
SMS	64.92	60.66	62.72	86.63	86.63	86.63	84.85	88.53	86.65	82.96	91.74	87.13
Riskware	99.35	57.76	73.05	97.45	81.68	88.87	98.20	88.40	93.04	99.07	86.93	92.60
Benign	76.47	83.88	80.00	78.62	86.23	82.25	80.21	87.97	83.91	82.19	86.96	84.51
Overall	77.92	78.41	76.32	88.30	88.50	88.20	88.47	88.35	88.30	89.18	90.43	89.64



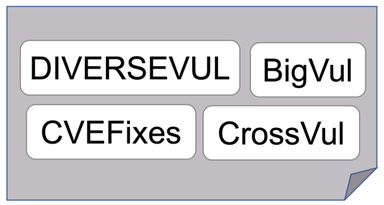




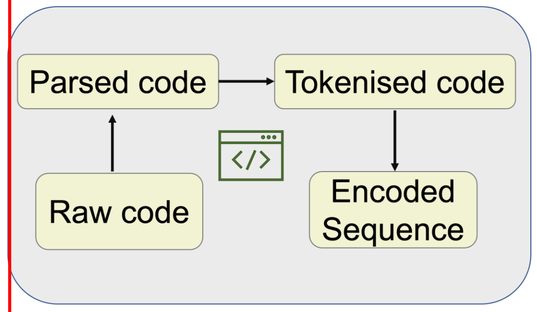
향후 계획

DL 기반 코드 오류 탐지 연구 계속

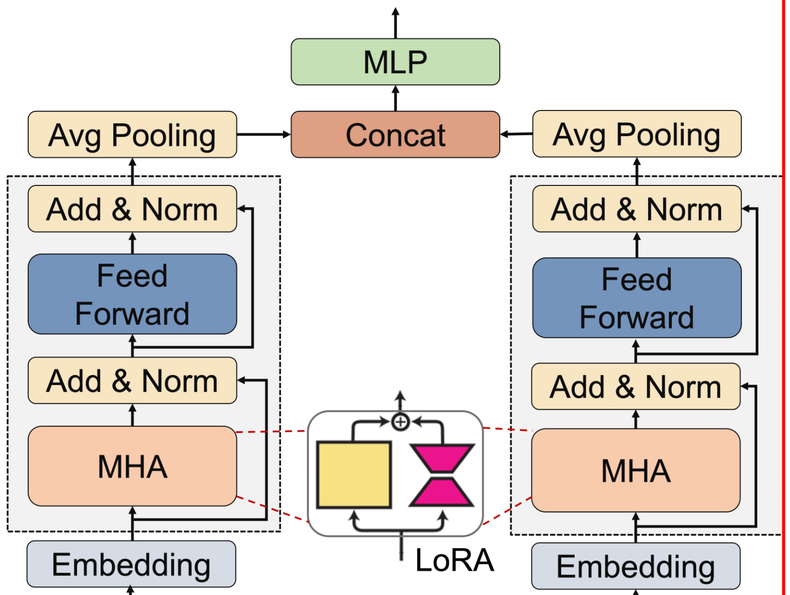
재난 DB로 대체



② Fine-tuning datasets

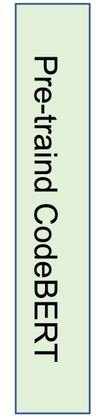


③ Parses, tokenises and encodes fine-tuning datasets based on the prebuilt vocabulary.



④ Fine-tune the CodeBERT model on a software vulnerability detection task across multiple vulnerability detection dataset.

⑤ LoRA freezes the pre-trained model's weights while adding trainable layers (rank decomposition matrix) to each Transformer block.

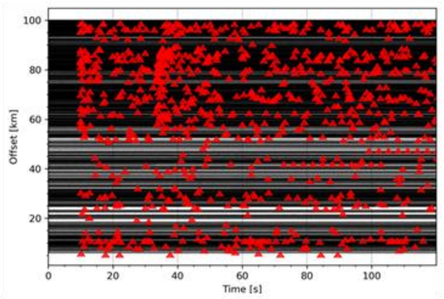


① The pre-trained model is ready for fine-tuning task.

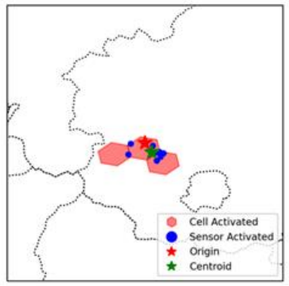
지진경보시스템 테스트베드의 현재

CrowdQuake (a.k.a 테스트베드)

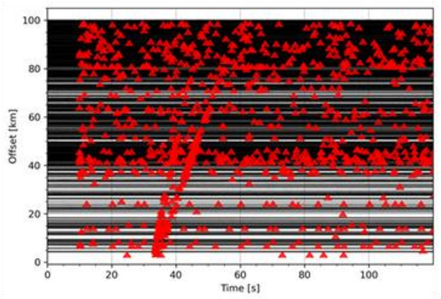
실시간 지진 감지 과정



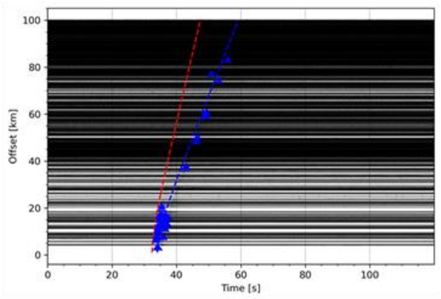
(1) Triggers by STA/LTA



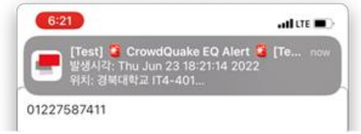
(2) Cell Activation



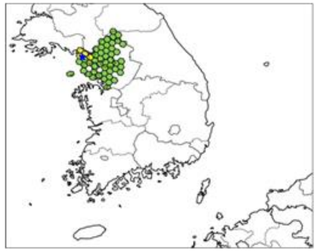
(3) Earthquake origination



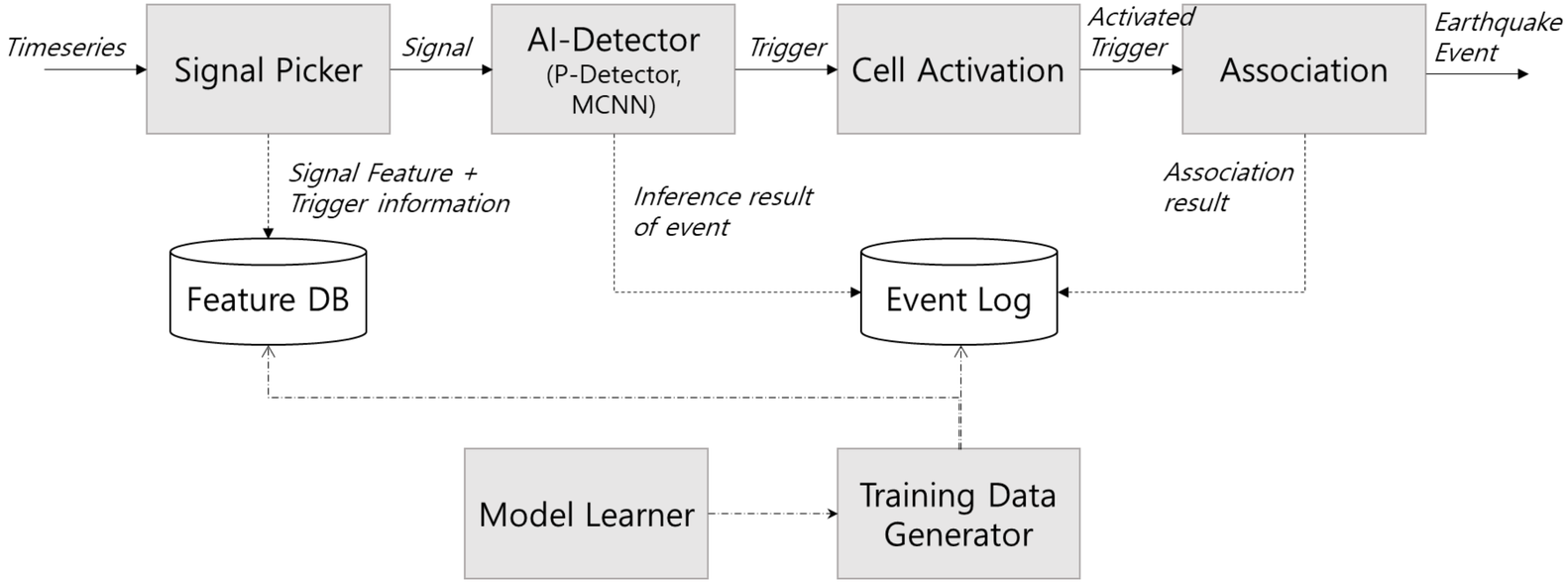
(4) Earthquake event association



(5) Alert generation



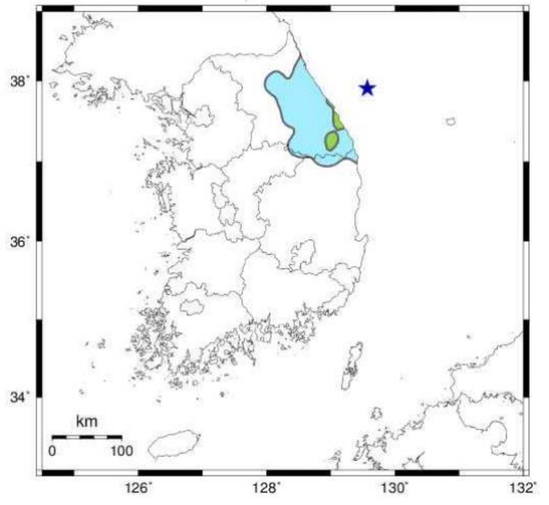
(6) Intensity map generation

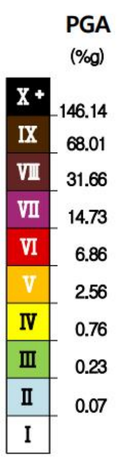


사례 연구: 2023년 5월 15일 동해 지진 M4.5

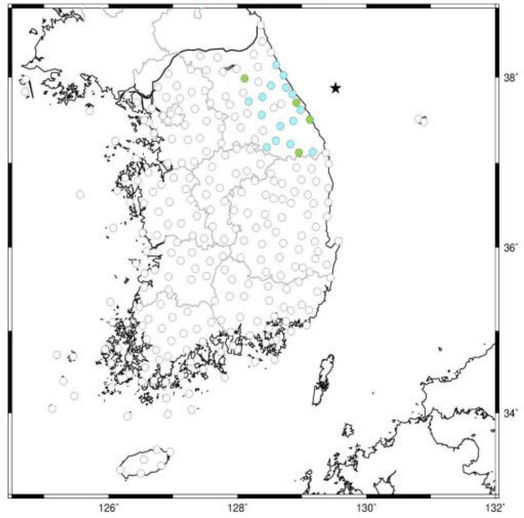
• 발생시각		2023년 5월 15일 06시 27분 37초			
• 위치(불확도)		강원 동해시 북동쪽 52km 해역 위도: 37.874° N, 경도: 129.522° E (±4.29 km)			
• 규모(불확도)		4.5 M_L (± 0.3)	깊이	31 km	
• 진도	최대계기진도	Ⅲ(강원, 경북), Ⅱ(충북)			
	최대지반가속도	관측소	동해(TOHA)	PGA(%g)	0.395

진도분포도

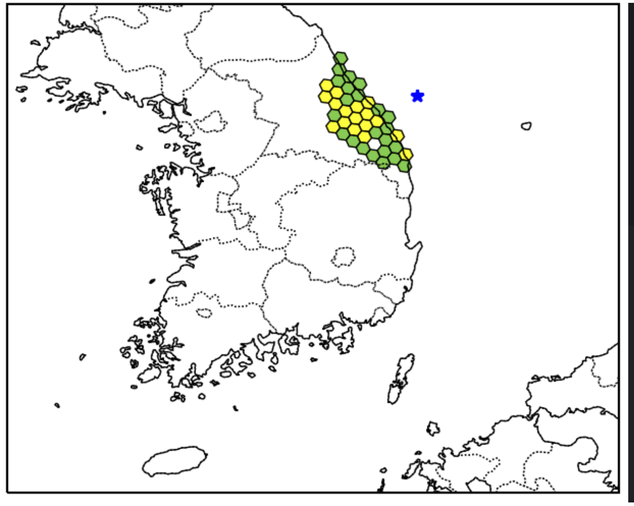


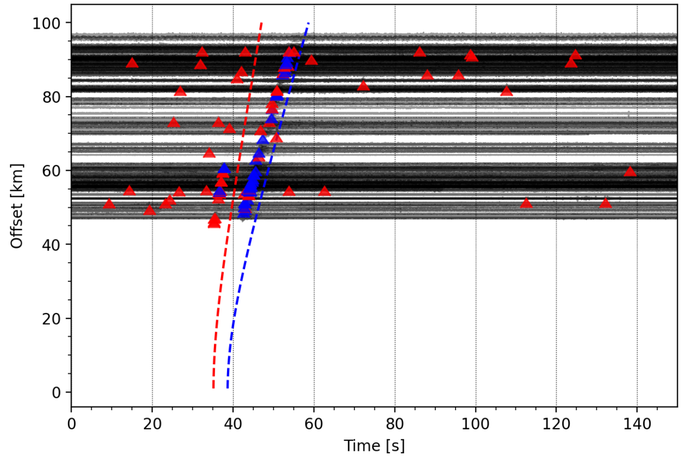


최대지반가속도 분포도



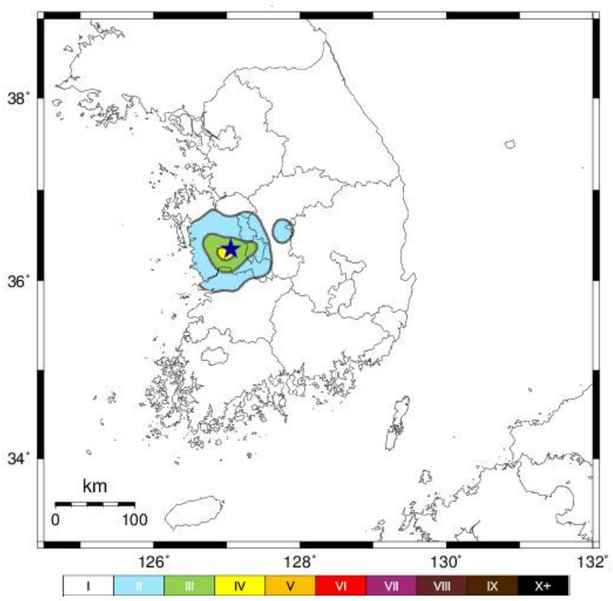
- 지진 발생 후 11초 후 최초 관측
- 최초 관측 2초 후 지진 경보 정보 생성



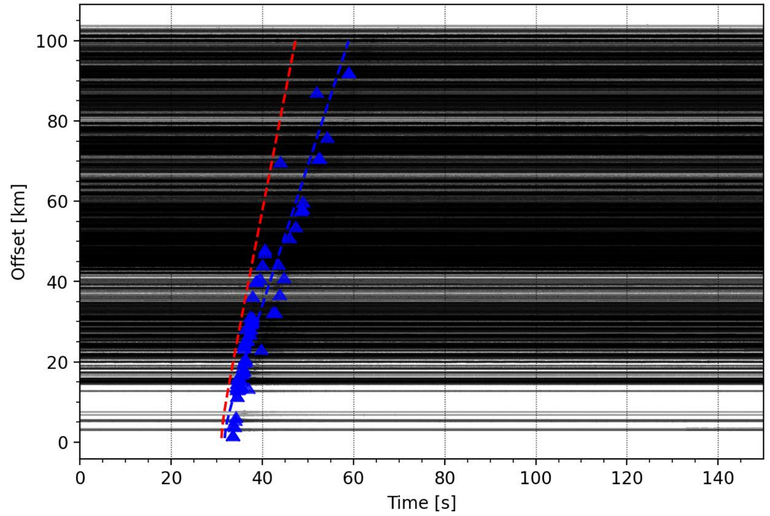


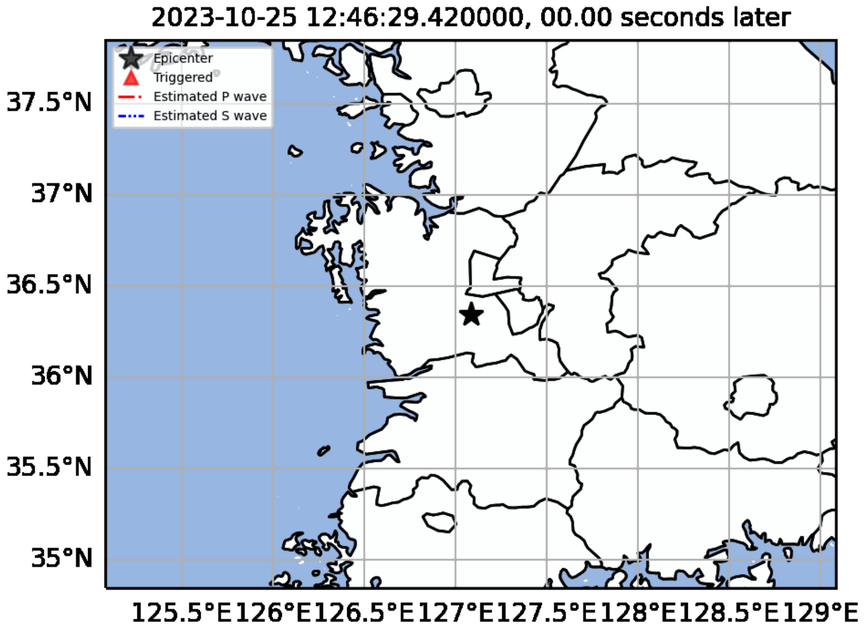
사례 연구: 2023년 10월 25일 공주 지진 M3.4

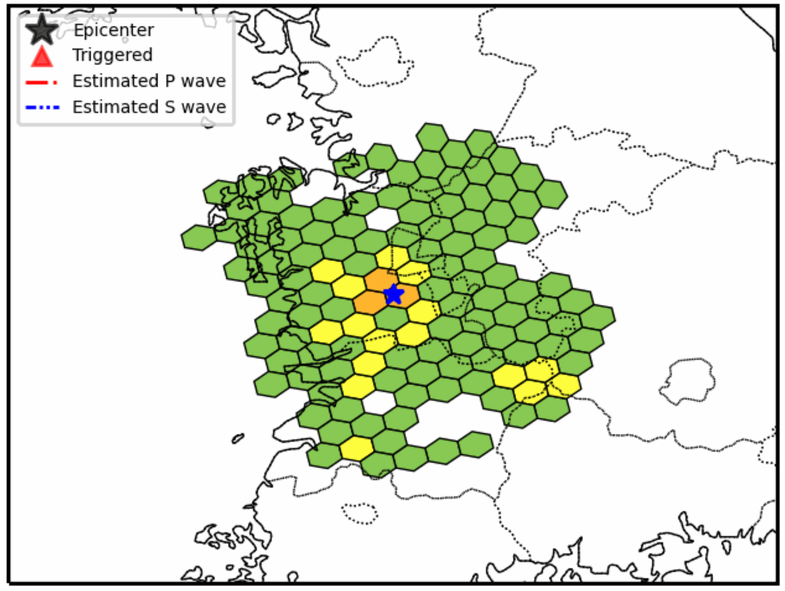
- 23년 7월 29일 19시 07분 59초
- 23년 7월 기준 최대 규모 (내륙기준)



- 지진 발생 후 1.87초 후 최초 관측
- 최초 관측 후 1.7초 후 지진 경보 정보 생성
- 대전 인근에서 발생하여 다수의 센서에서 감지

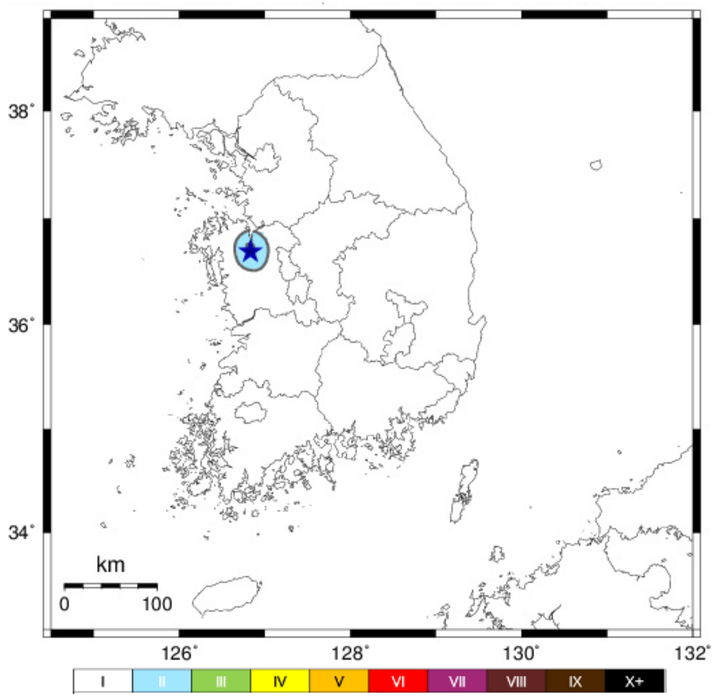




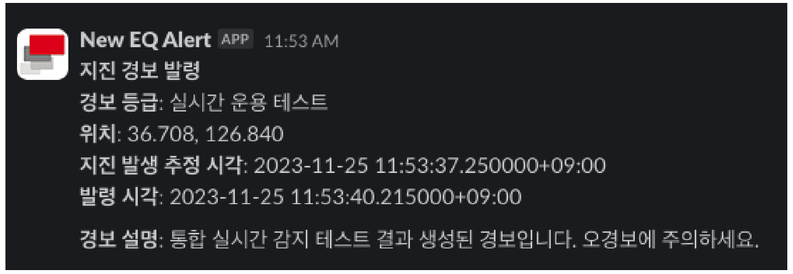


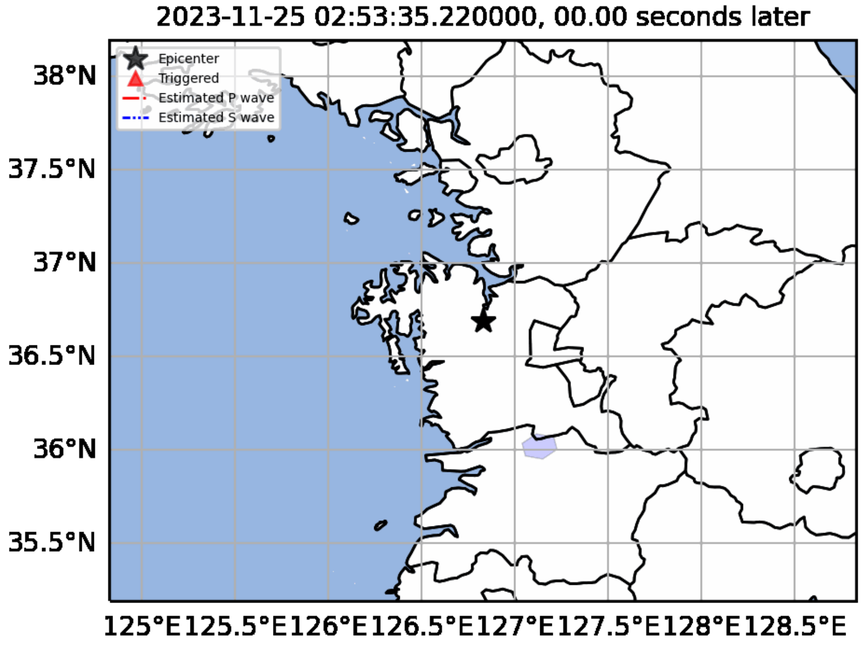
사례 연구: 2023년 11월 25일 예산 지진, M2.6

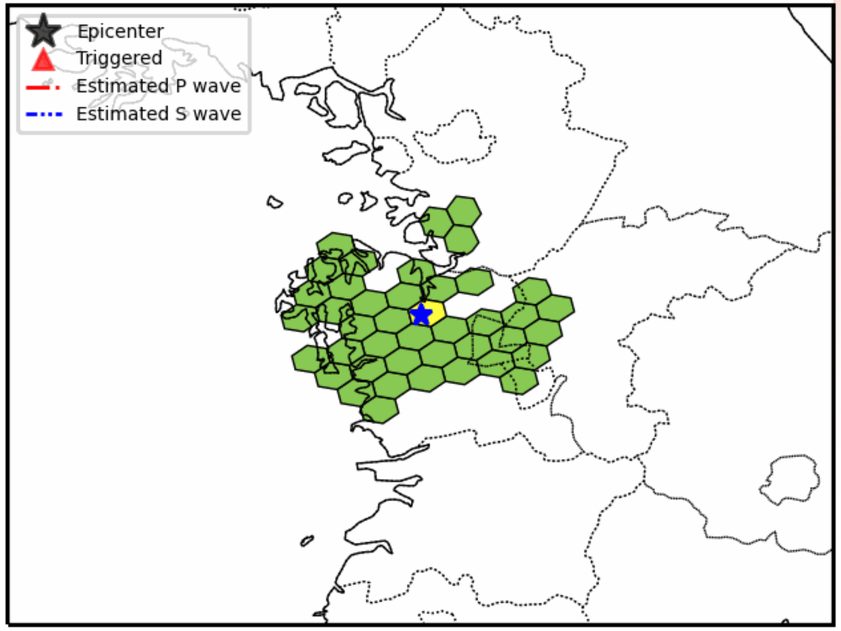
- 23년 11월 25일 11시 53분 35초



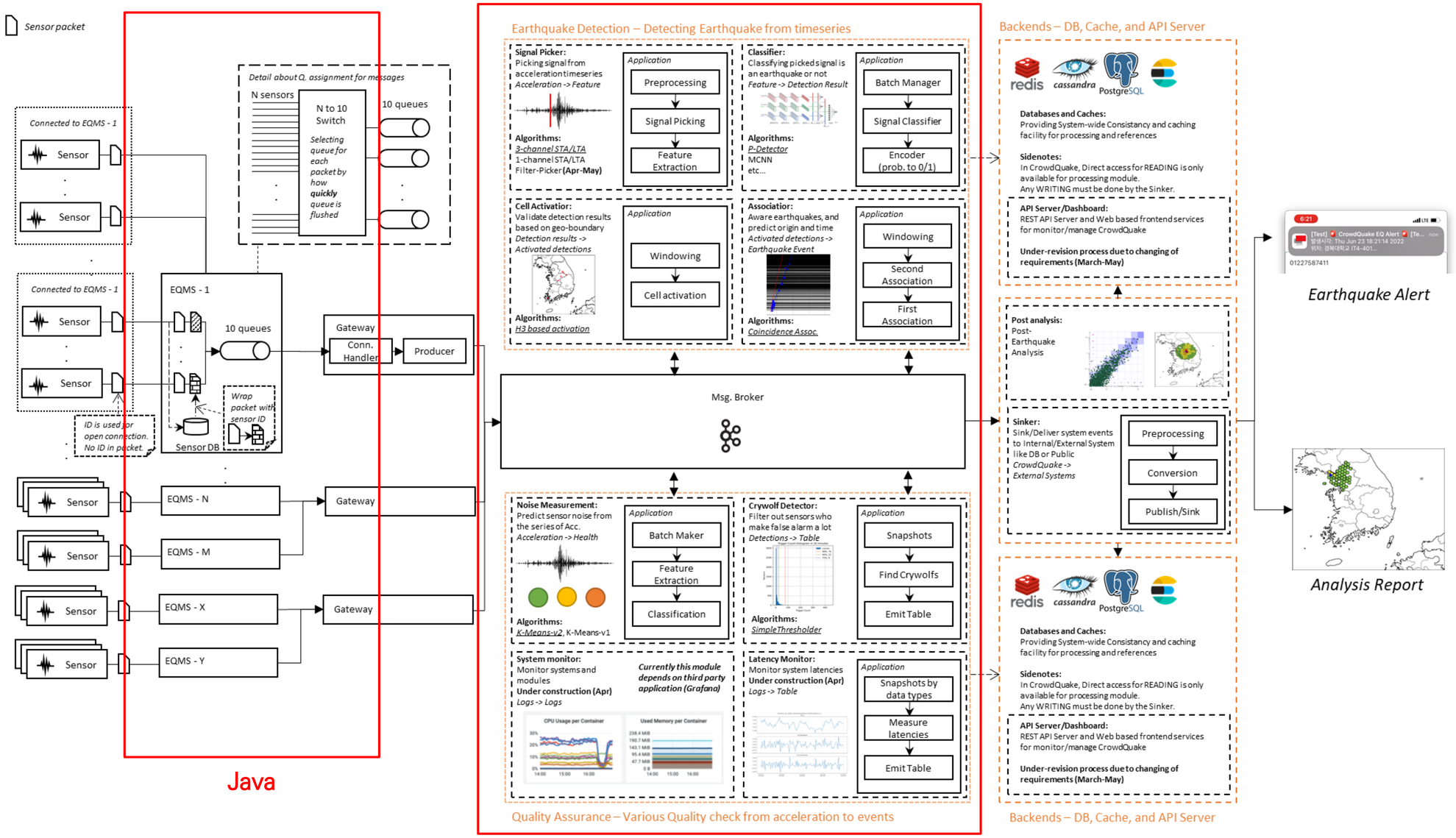
- 지진 발생 후 2초 후 최초 관측
- 최초 관측 후 3초 후 지진 경보 정보 생성
- 내부 채널로 경보 정보 자동 전송







시스템 개요

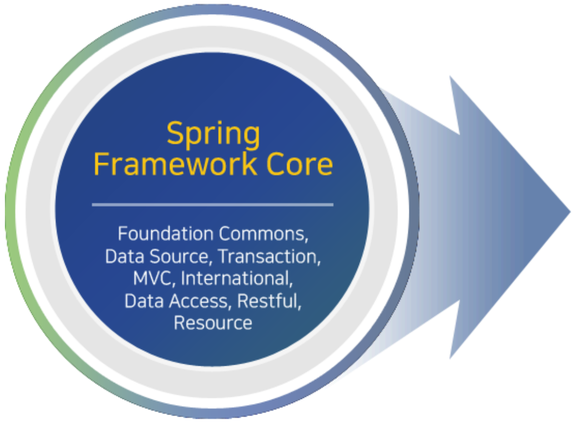

Sensor packet
Detail about Q. assignment for messages
N sensors
N to 10 Switch
Selecting queue for each packet by how quickly queue is flushed
10 queues
Connected to EQMS - 1
Sensor
Sensor
Connected to EQMS - 1
Sensor
Sensor
EQMS - 1
10 queues
Gateway
Conn. Handler
Producer
Wrap packet with sensor ID
Sensor DB
ID is used for open connection. No ID in packet.
Sensor
EQMS - N
Gateway
Sensor
EQMS - M
Sensor
EQMS - X
Gateway
Sensor
EQMS - Y
Java
Earthquake Detection – Detecting Earthquake from timeseries
Signal Picker:
Picking signal from acceleration timeseries
Acceleration -> Feature
Algorithms:
3-channel STA/LTA
1-channel STA/LTA
Filter-Picker (Apr-May)
Application
Preprocessing
Signal Picking
Feature Extraction
Classifier:
Classifying picked signal is an earthquake or not
Feature -> Detection Result
Algorithms:
P-Detector
MCNN
etc...
Application
Batch Manager
Signal Classifier
Encoder (prob. to 0/1)
Cell Activation:
Validate detection results based on geo-boundary
Detection results -> Activated detections
Algorithms:
H3 based activation
Application
Windowing
Cell activation
Association:
Aware earthquakes, and predict origin and time
Activated detections -> Earthquake Event
Algorithms:
Coincidence Assoc.
Application
Windowing
Second Association
First Association
Msg. Broker
Noise Measurement:
Predict sensor noise from the series of Acc.
Acceleration -> Health
Algorithms:
K-Means-v2, K-Means-v1
Application
Batch Maker
Feature Extraction
Classification
Crywolf Detector:
Filter out sensors who make false alarm a lot
Detections -> Table
Algorithms:
SimpleThresholder
Application
Snapshots
Find Crywolfs
Emit Table
System monitor:
Monitor systems and modules
Under construction (Apr)
Logs -> Logs
Currently this module depends on third party application (Grafana)
CPU Usage per Container
Used Memory per Container
Latency Monitor:
Monitor system latencies
Under construction (Apr)
Logs -> Table
Application
Snapshots by data types
Measure latencies
Emit Table
Quality Assurance – Various Quality check from acceleration to events
Python
Backends – DB, Cache, and API Server
redis
cassandra
PostgreSQL
Databases and Caches:
Providing System-wide Consistancy and caching facility for processing and references
Sidenotes:
In CrowdQuake, Direct access for READING is only available for processing module.
Any WRITING must be done by the Sinker.
API Server/Dashboard:
REST API Server and Web based frontend services for monitor/manage CrowdQuake
Under-revision process due to changing of requirements (March-May)
Post analysis:
Post-Earthquake Analysis
Sinker:
Sink/Deliver system events to Internal/External System like DB or Public
CrowdQuake -> External Systems
Preprocessing
Conversion
Publish/Sink
Backends – DB, Cache, and API Server
Earthquake Alert
Analysis Report

향후 계획: CrowdQuake의 구조 변경 및 테스트베드로의 활용

- MSA 구조로 변경 및 클라우드 환경에서 운영
  - Spring 프레임워크 기반으로 재설계 및 개발
  - 클라우드 환경 도입
- 서버리스 함수의 자동 추출 및 변환 연구
  - 자주 사용되는 혹은 변화가 잦은 기능에 대해서는 서버리스 함수로 자동 변환
    - Cloud refactoring: automated transitioning to cloud-based services, J. of ASE 2014
- 함수 단위의 시스템 이상동작 파악에 활용
  - 로그 기반의 시스템 이상징후 탐지 후 이상 (후보) 함수 파악
  - 딥러닝 기반의 오류 탐지 기법을 통하여 이상 함수 확정

활용 가능성

- 스프링 기반의 전자정부 프레임워크에의 활용



공통기반	AOP, Cache, Compress/Decompress, Encryption/Decryption, Mail, File Handling, File Upload/Download, ID Generation, Resource, Ioc Container, Logging, Marshalling/Unmarshalling, Property, Object Pooling, Scheduling, Server Security, String Util, FTP, Excel, XML Manipulation
화면처리	Ajax Support, Internationalization, MVC, Security, UI Adaptor
모바일 화면처리	jQuery, Jquery Mobile
데이터처리	Data Access, Data Source, ORM, Transaction
연계통합	Naming Service, Web Service, Intergration Service
업무처리	Process Control, Exception Handling
배치처리	Job Configuration, Step Configuration, ItemReader/ItemWriter, Job Execution, Step Execution, Tasklet, Job Repository, Scalability, Job Runner, Job Launcher, History Management, Skip/Repeat/Retry, Sync/Async Processing, Pre/Post Processing

‘디지털 정부’ 해외 홍보 중 ‘디지털 재난’ 터졌다

이상민 행안장관, 성과 내세우러 나갔다가 행정망 먹통에 귀국
대응 매뉴얼 없고 마비 원인도 사흘째 못 찾아… 국민들만 피해

정부공공IT

'디지털 정부 재난', 정부 IT시스템 근본적인 검토 필요해져

'행정망 마비' 대혼란 속 가까스로 복구...재발방지책 마련 시급

16일 밤 정보관리원 서버 보안 패치 업데이트 발단 17일부...
국민 대혼란에도 문자 공지조차 하나 없이 오후 5시 39분...
15년 된 노후화 시스템 대체할 '차세대 지방행정공통시스...
고기동 행안부 차관 "지방행정전산서비스 개편 T...

등록 2023-11-19 오후 5:23:42
수정 2023-11-19 오후 7:12:50

가 가

초유의 정부 행정전산망 마비, 디지털 재난 상황..."앞으로의 더 큰 사고들에 대한 예고"

KNU KYUNGPOOK NATIONAL UNIVERSITY

감사합니다