2026
DESIGN AND VERIFICATION™
DVCON
CONFERENCE AND EXHIBITION
UNITED STATES
SANTA CLARA, CA, USA
MARCH 2 - 5, 2026

A novel ML-Driven Simulation Log Debugger

~Samsung Semiconductor India Research

Contents

Problem Statement

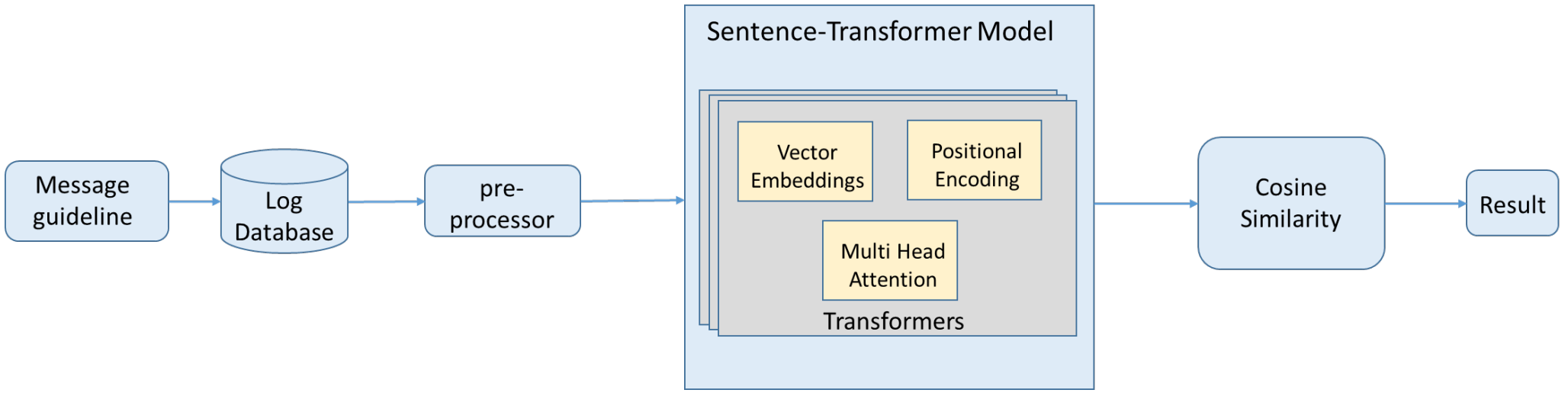
Hardships with traditional approach

- Hard to analyze Pretty Large sim file with many unwanted messages
- Lack of standardized format of messages in a log file
  - No streamlined logs
- Difficult to nail down to functional messages with ease
  - People follow wild GVIM techniques to delete un-wanted messages
- Time consuming setup to enable wave dump in a failing log
  - Requires wave analyzer setup, grouping of required signals etc.
- Log file comparisons when passing log's WS is deleted
- Log lines order check if any particular process is missed.

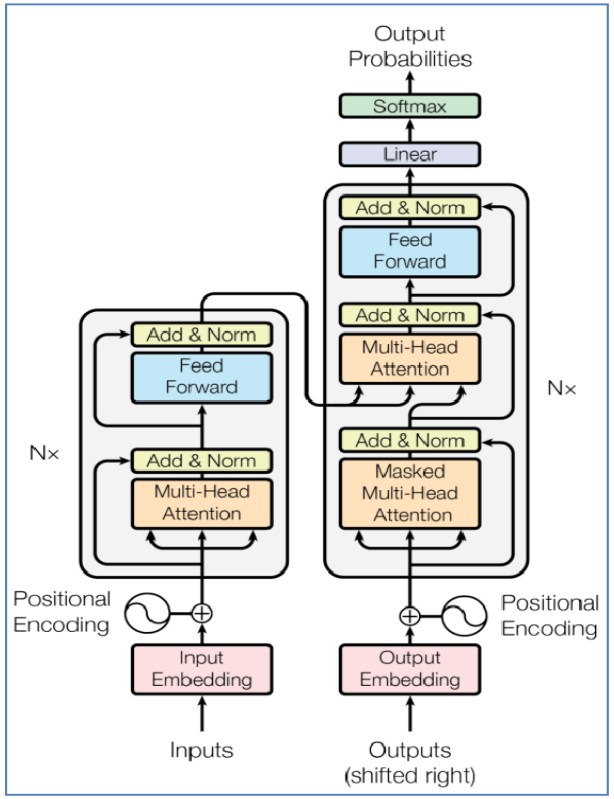
How do we address : Ourproposal

- **LogMiner** loads the large file and extracts all the required functional messages quickly.
- TAG wise messaging
  - Different stage/nature tags that separates the log messages based on the message patterns.
  - User can focus on the needed messages without any tedious methods.
- Anomaly detection
  - Compare the log file with passing one(Golden reference) using trained **ML model**
  - Compares the intended messages not just *words*.
  - Checks ordering of log lines
- Wave-less debug: Aid in RCA without wave dump
- Based on the UVM MESSAGE TAGs, it can analyze:
  - Register Read/Writes
  - Booting messages
  - APM messages
  - CORE messages
  - Common Task messages
  - Customized messages

LogMiner System Diagram



Multi -layer Transformer Architecture



"Attention Is All You Need", 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA

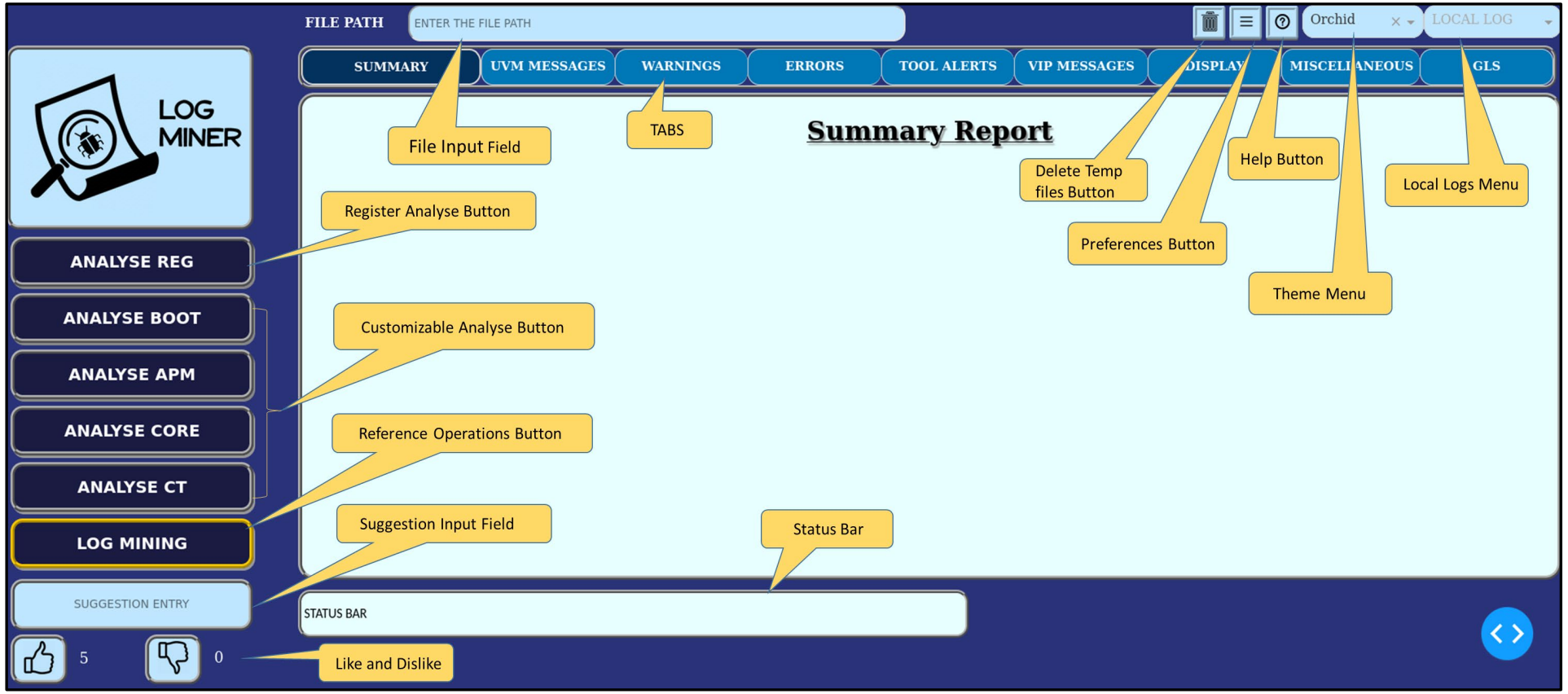
What is Sentence Transformers ?

- **BERT** (*Bidirectional Encoder Representations from Transformers*) understands context at the word level, making it great for classification, but slow for sentence similarity
- SBERT (*Sentence-BERT)* builds on BERT, using Siamese/Triplet networks to create fast, meaningful sentence embeddings, making it superior for semantic search, clustering, and paraphrase tasks by comparing sentence vectors directly
- Designed to overcome the computational overhead of BERT for semantic similarity search.
- Wide selection of over 10,000 pre-trained Sentence Transformers models are available for immediate use.
- Key Advantage:
  - Maps sentences to a dense vector space where similar sentences are close.
  - Enables efficient comparison using cosine similarity.
- It is easy to train or fine-tune our own embedding models using Sentence Transformers, which enables user to create custom models for domain specific use cases.

LogMiner Process

- Message guidelines is used to standardize the log formats, which is to add tags for different kinds of messages.
- The log files are **pre-processed** by removing unwanted messages, tool warnings, timestamps etc.
- **Sentence-Transformer model** is a Transformer model tailored to analyze sentences from HuggingFace.
- This model creates vector embeddings that convert text tokens into dense semantic representations.
- Positional encoding adds information about token order, enabling Transformers to understand sentence structure.
- Multi-head attention lets the model focus on different contextual relationships simultaneously.
- The model gives an overall score for a particular sequence of sentences which can be compared with reference sequence using **Cosine Similarity** with an ideal threshold value.
- The model can also detect the ordering of the lines as per trained order.

LogMiner GUI



Results

The simulation matrix shows the deviation of current log form the passing log

```
UVM_INFO (intr_test) test started
UVM_INFO (boot_vseq) boot_vseq is started
UVM_INFO (boot_vseq) boot_vseq is ended
UVM_INFO (intr_monitor) [ GIC_MON ]: acknowledge_intr is done
UVM_INFO (CORE_MISC_TMU) tmu interrupt test is done
UVM_INFO (CORE_MISC_common_task) ////[set_check_temperature] Done
UVM_INFO (intr_monitor) [ GIC_MON ] Interrupt clear checker .. finished
UVM_INFO (common_task_test) TEST PASSED
```

Reference

```
UVM_INFO (intr_test) test started
UVM_INFO (boot_vseq) boot_vseq is started
UVM_INFO (boot_vseq) boot_vseq is ended
UVM_INFO (intr_monitor) [ GIC_MON ]: acknowledge_intr is done
UVM_INFO (CORE_MISC) interrupt test is done
UVM_INFO (boot_vseq) boot_vseq at init_sys_vseq started ...
UVM_INFO (boot_vseq) init_lnpu_ncpu is started ..
UVM_INFO (CORE_MISC_common_task) ////[set_check_temperature] Done
UVM_INFO (intr_monitor) [ GIC_MON ] Interrupt clear checker .. finished
UVM_INFO (common_task_test) TEST PASSED
```

Log 2

```
tensor([[ 1.0000, -0.0215,  0.0151,  0.1432,  0.0434,  0.0366, -0.0275, -0.0790,  0.1942, -0.0461],
        [-0.0215,  1.0000,  0.5328,  0.0970,  0.0266,  0.7880,  0.3647,  0.1088,  0.1062,  0.1502],
        [ 0.0151,  0.5328,  1.0000,  0.1351,  0.1253,  0.4053,  0.0771,  0.1386,  0.2492,  0.1668],
        [ 0.1432,  0.0970,  0.1351,  1.0000,  0.2548,  0.0231,  0.0659,  0.2159,  0.5935,  0.1800],
        [ 0.0434,  0.0266,  0.1253,  0.2548,  1.0000,  0.0973,  0.0467,  0.3895,  0.4056,  0.6475],
        [-0.0790,  0.1088,  0.1386,  0.2159,  0.3895,  0.1496,  0.0533,  1.0000,  0.3047,  0.4086],
        [ 0.1942,  0.1062,  0.2492,  0.5935,  0.4056,  0.0612,  0.0494,  0.3047,  1.0000,  0.1329],
        [-0.0461,  0.1502,  0.1668,  0.1800,  0.6475,  0.1272,  0.0327,  0.4086,  0.1329,  1.0000]])
```

Simulation Matrix

Order Detection in Log

Figure-1 shows the plot of ordered log lines detection. Figure-2 shows the plot of log lines with 2 lines shuffled. As epochs increases, the accuracy of detection increases.

UVM_INFO (intr_test) test started
UVM_INFO (boot_vseq) boot_vseq has been started ...
UVM_INFO (boot_vseq) boot_vseq ended ...
UVM_INFO (intr_monitor) [GIC_MON]: acknowledge_intr is completed...
UVM_INFO (CORE_MISCU) interrupt test completed ...
.
.
UVM_INFO (gic600_init_seq_c) ITS :: ITT_BASE_ADDR = 80020000
UVM_INFO (gic600_init_seq_c) ITS :: ITS_CMD_Q_BASE_ADDR = 80030000
.
.
UVM_INFO (common_task_test) TEST SUCCESS

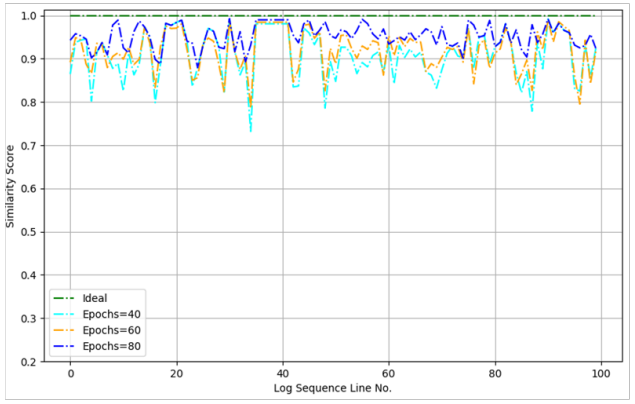


Figure 1

UVM_INFO (intr_test) test started
UVM_INFO (boot_vseq) boot_vseq has been started ...
UVM_INFO (boot_vseq) boot_vseq ended ...
UVM_INFO (intr_monitor) [GIC_MON]: acknowledge_intr is completed...
UVM_INFO (CORE_MISCU) interrupt test completed ...
.
.
UVM_INFO (gic600_init_seq_c) ITS :: ITS_CMD_Q_BASE_ADDR = 80030000
UVM_INFO (gic600_init_seq_c) ITS :: ITT_BASE_ADDR = 80020000
.
.
UVM_INFO (common_task_test) TEST SUCCESS

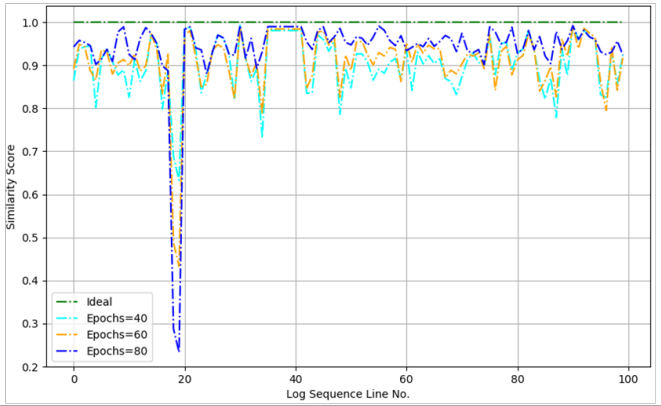


Figure 2

TAT Comparison

TAT(**T**urn **A**round **T**ime) for finding the actual difference in failing and passing log for 4 types of tests.

Sl No.	Test	Traditional Method		LogMiner	
		Log size (lines)	TAT	Log size (lines)	TAT
1.	A basic SFR test	10K	5 min	248	1 min
2.	Simple CORE test	100K	30 min	2.7k	5 min
3.	Complex CORE test	500K	1.5 hrs	13.5k	8 min
4.	Complex CORE Multi Boot test	1M	3 hrs	24.3k	12 min

Applications

- **LogMiner** can be customizable to SoC, IP, Subsystem level projects and other software testing logs as well
- Helps in Root cause analysis.
- Works as **Wave-Less** debug provided log message are fair enough.
- Smart comparator: Used to perform quick comparison of failed logs with a reference log created instead of traditional text editors which is difficult to compare with whole lot of information.

Future Scope

- Enhance to identify root cause deep in the design by providing structured messages those can be linked to find root cause.
- Extending the model to analyze different kind of files
- Enhance AI/ML techniques to suggest better reference models
- Integrating FAISS vector similarity search

Conclusion

- **E**ases the comparison of failing logs using ML model trained with passing signatures
- It **reduces the Turn Around Time** for a complete comparison which can be a main deciding difference for the failure.
- Transformer based design with cosine similarity is one of the best solution in anomaly Detection
- TAG based messaging guideline and implementation helps in reaching route cause **faster and cost effective**.
- Generalized and **effective solution to adopt at various stages of development cycle** (IP/Sub-System/SOC/Mulit-chip/Mulit-Die..)

References

1. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, “Attention Is All You Need”, 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA
2. L. Smith, “Fuzzy String Matching Procedure,” Open Bioinformatics Journal, vol. 13, pp. 50-62, 2020.
3. Reimers and Gurevych, “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks”, Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, pages 3982–3992, Hong Kong, China, November 3–7, 2019. c 2019 Association for Computational Linguistics
4. Cogita. “https://thevtool.com/vtool-eda”
5. Splunk. “https://www.splunk.com/”
6. Elasticsearch, Logstash, and Kibana. “https://www.elastic.co/elastic-stack”
7. Maria Teresa Colangelo, Marco Meleti, Stefano Guizzardi, Elena Calciolari and Carlo Galli, “A Comparative Analysis of Sentence Transformer Models for Automated Journal Recommendation Using PubMed Metadata”
8. “Sentence Transformers.” Hugging Face, https://huggingface.co/sentence-transformer


Thank You
Quicker
Easier
Waveless
Fast Debug