- Translates log messages in to log events
- Represents log events as vectors and detects anomalies using a Transformer-based classification model
- Events are grouped into a temporal order called log sequences
- Four main log parsers: Drain, AEL, IPLoM, and Spell
    - Drain uses fixed depth tree structure to extract common templates/events
    - Spell uses longest common subsequence to handle log streams
    - AEL groups log messages
    - IPLoM groups log messages according to length, positions, and mapping relationships
- Log parsing typically involves transforming log messages into instances of log templates, which are then evaluated as log sequences. Some sequences are innocuous while others are truly anomalous.
- Log parsing has three avenues of statistically significant error
    - Out Of Vocabulary (OOV) words
        - All typographical errors in log content are identified as anomalies
        - all new logs are identified as anomalous due to the rarity of those logs occurring, even if they're not
    - Semantic overfitting (misidentifying parameters as keywords)
        - Multiple log messages are applied to the same template and found to not match when they should, resulting in additional, incorrect log templates that then do not contribute to the model
    - Semantic underfitting (misidentifying keywords as parameters
        - Multiple log messages are applied to the same template, but too much is parsed out of the message, making
- In conclusion, log parsing is not a viable strategy as it is unacceptably inaccurate given that there are better alternatives.
- NeuralLog architecture
    - Preprocessing
        - Vocabulary is established on splitting log into words by delimiters and then moving delimiters
    - Neural representation
        - Words that seem to be OOV are then broken down into subwords and tokenized again based on likelihood
        - Uses Word2Vec to convert fully tokenized log messages into vectors
        - Uses BERT, deep learning model, to extract semantic meaning based on context
            - 12 layer of transformer encoder with 768 hidden units in each transformer.
            - Words get generated embeddings, and each log messages imbedding is calculated based on the average of each subword embedding
            - Uses self attention to measure importance of each word
    - Transformer based classification

        - Takes semantic vectors of log messages as input
        - Combines embeddings and semantic vectors, grouping similar semantic vectors closer together
    - Results
        - NeuralLog outperformed LogRobust and Log2Vec in terms of F1-Score on the datasets BGL, Thunderbird, and Spirit, but LogRobust did better on a HDFS set
        - NeuralLog also handled OOV words more accurately than if it had used Word2Vec or not used WordPiece on HDFS, BGL, Thunderbird, and Spirit datasets
        - Good performance is attributed to the use of raw log messages, resulting in no loss of information as logs are converted into vectors, meaning that NeuralLog can better understand semantic meaning of log messages