VANGUARD at eRisk 2026 Task 2: Contextualized Early Detection of Depression

Notebook for the eRisk Lab at CLEF 2026

Anamaria Vișan[1], Ciprian-Octavian Truică[1,2], Adrian Paschke[3] and Elena-Simona Apostol[1,*]

[1]National University of Science and Technology Politehnica Bucharest, Splaiul Independenței 313, București 060042, Romania
[2]Academy of Romanian Scientists, Ilfov 3, Bucharest, 050044, Romania
[3]Fraunhofer Institute for Open Communication Systems, Berlin, 10589, Germany

Abstract

In this paper, we describe the VANGUARD team in the eRisk 2026 Task 2 (Contextualized Early Detection of Depression). Systems read streaming multi-party conversations (submissions and comments) and submit timely alerts with risk scores. We implement three detectors: TF-IDF with handcrafted features and logistic regression, fine-tuned MentalBERT, and a weighted ensemble, combined with an ERDE-inspired early-decision policy and the official REST client. On a 100-subject offline simulation, the classical pipeline achieves $ERDE_5$ = 0.11, F1 = 0.71, and median alert latency of three writings. Partial live server participation spans more than 130 interaction rounds. The lab overview [1] reports best F1 = 0.74 (Run 0) on 129 processed threads; see Section 4.4.

Keywords

depression, early risk detection, social media, conversational context, eRisk, ERDE, MentalBERT

1. Introduction and task description

1.1. Motivation and task overview

Major depressive disorder remains under-diagnosed; language in online conversations may reveal early risk before formal assessment. The eRisk lab studies early risk prediction from user-generated text [1]. Task 2 (2026), i.e., Contextualized Early Detection of Depression, is the second edition of this track, first introduced in eRisk 2025 [2].

The task targets early signs of depression by analyzing full conversational contexts. Unlike earlier eRisk depression tasks that relied largely on isolated posts from a target user, participants must process multi-party interactions sequentially. Each round may include discussion titles, bodies, and comments from all participants in chronological order, including exchanges between the user to classify and others. Systems therefore monitor how dialogue evolves in settings such as blogs, forums, or social networks, rather than scoring static bags of documents.

1.2. Test collection and labels

The test collection follows the format established for eRisk depression research [3] and draws on the same family of social-media sources as prior editions. For each subject, the stream provides target-user writings, i.e., posts authored by the user whose risk must be estimated, together with full conversational context that includes the submission title and text plus threaded comments from all participants in chronological order. Subjects belong to either the depressed or the control class. Each sequence bundles

CLEF 2026 Working Notes, 21 – 24 September 2026, Jena, Germany
*Corresponding author.
anamaria.visan@stud.acs.upb.ro (A. Vișan); ciprian.truica@upb.ro (C. Truică); adrian.paschke@fokus.fraunhofer.de (A. Paschke); elena.apostol@upb.ro (E. Apostol)
https://sites.google.com/view/ciprian-octavian-truica (C. Truică); https://sites.google.com/view/elena-simona-apostol (E. Apostol)
0009-0002-7229-9649 (A. Vișan); 0000-0001-7292-4462 (C. Truică); 0000-0003-3156-9040 (A. Paschke); 0000-0001-6397-4951 (E. Apostol)

the target user's contributions with the surrounding conversation so that models can exploit interaction dynamics such as withdrawal, negative reciprocity, or shifts in tone after criticism that a single post might not reveal.

1.3. Training stage

Teams receive the contextualized dataset from eRisk 2025: isolated target-user writings plus full conversation contexts, with labels indicating whether the user has explicitly mentioned a depression diagnosis. The 2026 training release comprises 909 subjects (102 depressed, 807 control). Prior early-detection collections may be used jointly for development.

Figure 1 summarizes the training split. Most subjects contribute few submissions and few target-user writings; both classes include long-tail users with large thread histories. Class counts are 807 control and 102 depressed.

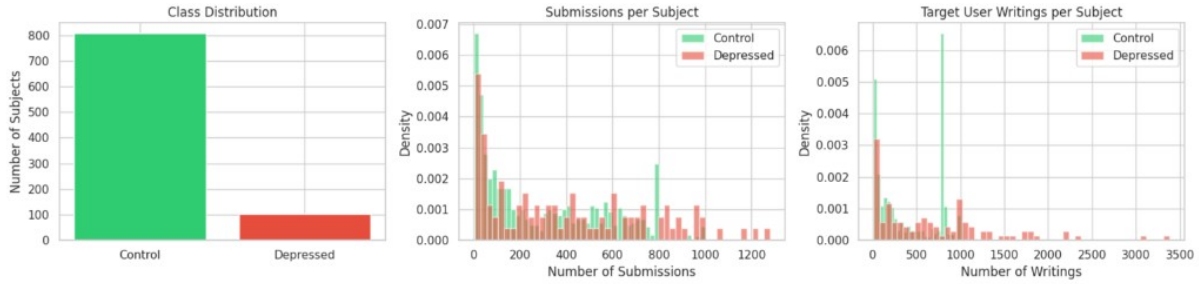

Figure 1: Training-set statistics: class counts (left), density of submissions per subject (center), and density of target-user writings per subject (right). Control subjects are shown in green and depressed subjects in red.

1.4. Live test protocol

Each registered team receives a token and up to five run indices (variants). Teams connect to the organizer REST API: a GET request retrieves the current round of discussions; after local processing, the team sends one POST per run with decisions for every user in the cohort. The server advances to the next round only when all runs have submitted complete responses for the current round, so runs must stay synchronized and a faster variant cannot observe writings before slower variants from the same team.

The first GET returns the first discussion for each user in the collection, and teams should persist the full user list because every later POST must include all users, even when a subject no longer receives new threads. Each subsequent round delivers one thread per active user in which the `targetSubject` participated; the target may author the submission, one or more comments, or both. An empty GET response signals that all writings have been released.

Each discussion combines submissions and comments. A submission is a primary post identified by `submissionId` and `author`, with an ISO-8601 `date`, `title`, `body`, a round `number` (0 denotes the first writing), the `targetSubject` to classify, and a `comments` list. A comment is a reply with its own `commentId`, `author`, `date`, `body`, and `parent` reference to either a submission or another comment, forming a tree under the submission.

Each POST body is a JSON array of objects with fields `nick`, `decision`, and `score` for every user in the cohort. The `decision` field is 0 when no alert is raised and may be revised in later rounds, or 1 when an alert is final; after an alert, later `decision` values for that user are ignored but must still be sent. The `score` field is a numeric risk estimate used for ranking-based evaluation as well as decision metrics. After an alert, teams continue processing new writings for that user and submit updated scores each round.

Teams that deploy fewer than five systems must still POST for runs 0 to 4, sending `decision=0` and nominal scores on unused runs to preserve synchronization. The server does not mark a user's last

writing explicitly; subjects disappear from GET responses once exhausted, so a user with 25 writings appears in rounds 1 to 25 only and latency is measured in rounds until alert. We recommend a retry with exponential backoff to avoid overloading the service.

1.5. Evaluation

Assessment considers both correctness in distinguishing depressed from control users and timeliness of alerts. The primary cost-sensitive measure is $ERDE_o$ (Early Risk Detection Error) [3, 4], which penalizes late correct alerts and false positives on healthy users. Organizers also report F1, precision, recall, latency measured as writings before alert, and latency-aware variants used in recent eRisk overviews. Our development replay optimizes an ERDE-inspired alert policy on held-out labeled subjects, while official ranking relies on organizer scripts applied to the blind live stream [1].

1.6. Core difficulties

Teams must fuse target-user text with contributions from interlocutors while scoring a growing stream under a separate early-stop policy and severe class imbalance. The REST protocol further requires run synchronization and full-cohort POSTs every round, which makes long-running sessions operationally demanding. As in all eRisk tasks, emitted alerts are research flags and must not be interpreted as clinical diagnoses.

1.7. Our participation (VANGUARD)

We address Task 2 with three complementary detectors: a classical machine learning model, a MentalBERT model, and a weighted ensemble. These correspond to live API run indices 0, 1, and 2, respectively. To comply with the five-run synchronization protocol, indices 3 and 4 post zero-decision padding. As detailed in Section 3, our live test phase also incorporates an ERDE-oriented early-decision rule and a logged REST client.

Our request log shows that the classical ML run (index 0) failed during live deployment and never issued an alert, so only the MentalBERT (index 1) and ensemble (index 2) runs contributed usable decisions; the organizer's official scoring reports only these two runs, and Section 4.4 works out which organizer run label corresponds to which of our detectors.

2. State of the art

We reviewed 51 papers on social-media depression detection, early risk, conversational context, and eRisk methodology. The subsections below summarize topics relevant to Task 2.

2.1. Social-media text and classical ML/DL

Cross-sectional studies show that depressive signal can be recovered from posts when reliable labels exist. Classical pipelines combine linguistic features with support vector machines, random forests, or ensembles [5, 6]. Systematic reviews describe a shift toward recurrent networks, convolutional networks, and BERT-family encoders, while BiLSTM with attention and hybrid stacks remain strong baselines [7, 8]. These studies repeatedly report difficulties with class imbalance [9], noisy self-labels, platform bias, and weak cross-domain generalization. Multi-level frameworks that map posts to severity bands using BERT, clustering, and support vector machines [10]. Task 2 adds a streaming evaluation protocol and requires scoring interlocutor text alongside the target user.

2.2. Ground truth, labels, and imbalance

Reviews note that ground truth is constructed inconsistently across studies, using clinician surveys, structured interviews, self-declaration, or community membership as proxies [10]. The 2026 training

release labels users who explicitly mentioned a depression diagnosis, which yields a heavily imbalanced cohort of 102 depressed versus 807 control subjects among 909 users. That imbalance favors high accuracy on controls but penalizes missed depressed users in cost-sensitive metrics. Our classifiers therefore use balanced losses and threshold tuning on development folds.

2.3. Transformers and mental-health language models

BERT-family fine-tuning is common in recent depression-detection work, and domain-adapted MentalBERT improves mental-health text classification over generic encoders [11, 8]. Fine-tuned GPT-3.5 and LLaMA-2 models report high accuracy on depressed versus control posts but raise concerns about cost, privacy, and reproducibility. Hybrid fusion of contextual embeddings with mental-health-specific representations, including Multi-MentalBERT-style cross-attention, helps in code-mixed and low-resource settings. Because Task 2 histories grow each round, we encode accumulated target text for Run 1 with head-and-tail truncation at 512 tokens.

2.4. Hybrid, lexical, and explainable pipelines

Handcrafted features such as sentiment, self-focus, absolutist language, and depression lexicons remain competitive with neural models on small or imbalanced corpora [5, 8]. Studies that combine TF-IDF or fastText with gradient-boosted or logistic classifiers achieve strong Reddit baselines, and explainable ML techniques such as SHAP can aid inspection of model inputs. Emotion-aware models report that polarity alone underperforms clinically aligned affect cues. Our Run 0 uses TF-IDF with scaled handcrafted vectors and logistic regression; Run 2 combines rule-based, ML, and transformer scores.

2.5. Early detection, ERDE, and streaming policies

Losada and Crestani introduced shared collections and the ERDE metric to balance accuracy and delay on sequential writings [3, 4]. Trotzek showed that convolutional networks on embeddings and ensembles of user-level linguistic metadata reach strong early-detection results. At the same time, they also critiqued ERDE's sensitivity to cohort size and proposed cost functions tied to the proportion of documents read rather than fixed latency offsets [12]. Neuro-symbolic streaming systems report competitive ERDE_5 and flatency on social timelines [13]. Task 2 evaluates ERDE_o, F1, precision, recall, and latency in rounds on the live REST stream [1]. Our ERDE-inspired policy relaxes alert thresholds over rounds, smooths scores with an exponentially weighted average and a running maximum, and locks alerts once fired.

2.6. Conversational and contextual modeling

Prior eRisk work on contextualized depression detection uses submissions and comments from all participants in chronological order [2]. Related dialogue studies use interview transcripts, counseling corpora, and spoken-language data with turn-level or topic-level features [14, 15]. We represent context through separate target and interlocutor text channels and conversational handcrafted features; we do not model reply graphs explicitly.

2.7. From eRisk timelines to contextualized Task 2

Prior eRisk depression tracks used target-user timelines on Reddit-like sources [3]. The 2025 to 2026 contextualized track adds multi-party threads, training on 909 labeled subjects and a synchronized REST protocol that requires five runs and full-cohort POSTs each round [2, 1]. We submitted a classical ML baseline, a MentalBERT run, and an ensemble to compare sparse, neural, and fused approaches.

2.8. Design implications for Task 2

We use three active detectors on runs 0 to 2: TF-IDF with linguistic, lexicon, and conversational features plus logistic regression; fine-tuned MentalBERT; and a weighted ensemble with a rule-based branch. Alert timing is handled by a separate early-stop policy tuned on held-out labeled streams. Runs 3 to 4 post-zero decisions to satisfy server synchronization.

2.9. Ethics and limitations

Prior work notes that automated flags are not diagnoses, that false positives can affect users, and that platform bias limits generalization [7, 8, 10]. Our submissions use research collections only.

3. Proposed solution

Team VANGUARD implemented Task 2 in a single notebook pipeline (`task2_contextualized_depression.ipynb`) with two coupled stages. The first stage covers offline training and simulation on the 909-subject release, and the second stage covers a logged REST client for the synchronized live test. The design deliberately separates the question of what risk score to emit, handled by three detector stacks on runs 0 to 2, from the question of when to alert, handled by `EarlyDecisionPolicy`, because ERDE-style metrics penalize both misclassification and delay.

3.1. Pipeline overview

Offline, we load labeled subjects from `all_combined/`, extract features, train a classical ML detector and a MentalBERT classifier under stratified five-fold cross-validation, and tune ensemble weights. We persist `classical_ml_detector.joblib` and a fine-tuned transformer directory for reuse on the server. During the live test phase, the `ERiskServerClient` polls discussions each round, rebuilds per-user `Subject` histories, scores the active detectors, applies the early-decision policy independently for each run, and POSTs a full cohort for all five run indices; runs 3 to 4 submit zero-padding responses only.

3.2. Data representation

Each training or test instance is represented as a `Subject` with a user identifier, a binary label, and a chronological list of `Submission` objects. Each submission stores its title, body, round index, authorship, and nested `Comment` objects linked through parent references. For scoring, we accumulate only writings observed so far. At live round k, the client appends new threads to per-user histories and builds a truncated subject containing submissions seen through the current round, and offline simulation follows the same rule. Target-user text for machine learning and transformers is the concatenation of the target's submission bodies and comments, whereas conversational statistics use the full thread from all authors to capture the interaction context described in Section 1.

3.3. Feature extraction

The `FeatureExtractor` module computes handcrafted vectors on the accumulated history, optionally augmented with trajectory deltas when enabled in configuration. Linguistic features are derived from target-user text and comprise VADER sentiment statistics, token and sentence counts, type-token ratio, and rates of absolutist words. Depression-lexicon features aggregate mention rates over ten hand-built categories that cover emotional distress, anhedonia, self-deprecation, suicidal ideation, sleep disturbance, social withdrawal, cognitive symptoms, guilt and shame, anxiety, and treatment mentions. Conversational features quantify how the target participates in each thread, including comment depth, mental-health subreddit indicators, and activity from interlocutors. Temporal features summarize

Table 1
Handcrafted feature families produced by `FeatureExtractor` (51 features).

Family	n	Column names (prefix or group)
Linguistic	17	`n_texts`, `n_words`, `n_sentences`, `n_chars`, length averages, `vocabulary_richness`, VADER statistics (`sentiment_compound_mean`, `sentiment_negative_mean`, ...), `first_person_rate`, punctuation rates
Lexicon	14	`lex_emotional_distress`, ..., `lex_treatment_mention` (10 categories), `lex_total_hits`, `lex_total_rate`, `lex_pattern_hits`, `lex_pattern_rate`
Conversational	10	`conv_n_target_texts`, `conv_n_context_texts`, `conv_participation_ratio`, thread counts, `conv_context_sentiment_mean`, `conv_mh_title_ratio`, ...
Temporal	5	`temp_posting_span_days`, `temp_avg_posts_per_day`, `temp_posting_regularity`, `temp_night_posting_ratio`, `temp_weekend_posting_ratio`
Trajectory	5	`traj_n_segments`, `traj_sentiment_slope`, `traj_lex_stress_slope`, `traj_sentiment_early_late_gap`, `traj_sentiment_volatility`
Metadata	2	`subject_id`, `label`

round-index statistics and track changes in sentiment or lexicon density as the stream grows. During sequential replay, features are recomputed up to the current round only, so each live step uses evidence available at decision time.

Offline, we build a feature matrix of shape 909×53 by calling `extract_all_features` on each labeled subject (51 numeric columns plus `subject_id` and `label`). Table 1 lists the feature families; Table 2 shows selected columns for one control subject.

We compare depressed and control subjects on the training matrix using Cohen's d with pooled standard deviation (positive d indicates a higher mean in the depressed class). Table 3 lists the top 20 features; Figure 2 plots the top 15. Lexicon totals, sentiment variability, first-person rate, and mental-health thread titles rank highest; vocabulary richness and conversation initiation are higher in controls.

3.4. Offline training

We train on the organizer release (909 subjects: 102 depressed, 807 control) with StratifiedKFold (n=5, seed 42). Ground-truth labels come from `shuffled_ground_truth_labels.txt`. The notebook's development table indexes offline runs 1 to 3 (classical ML, transformer, ensemble); the live server maps the same models to API run indices 0 to 2 (see below).

For the classical machine learning detector mapped to live Run 0, the `ClassicalMLDetector` fits TF-IDF on target-user corpora with up to 10,000 features, unigram and bigram ngrams, document-

Table 2
Selected handcrafted features for one control subject (`subject_01ZzrIT`, label 0). Full lexicon and conversational columns are omitted.

Column	Value
`n_texts`	248
`n_words`	8388
`n_sentences`	480
`vocabulary_richness`	0.234
`sentiment_compound_mean`	0.385
`sentiment_negative_mean`	0.067
`temp_posting_regularity`	565.4
`temp_night_posting_ratio`	0.215
`temp_weekend_posting_ratio`	0.040
`traj_n_segments`	248
`traj_sentiment_slope`	4.6×10^{-4}
`traj_lex_stress_slope`	0.014
`traj_sentiment_volatility`	0.462

Table 3
Top 20 handcrafted features by |Cohen's d| on the 909-subject training set (102 depressed, 807 control). Sign of d: higher in depressed ($d > 0$) or control ($d < 0$).

Feature	Cohen's d	Higher in
`lex_total_hits`	1.880	depressed
`sentiment_compound_std`	1.506	depressed
`traj_sentiment_volatility`	1.506	depressed
`first_person_rate`	1.186	depressed
`conv_mh_title_count`	1.117	depressed
`conv_mh_title_ratio`	1.041	depressed
`pct_negative_texts`	1.033	depressed
`sentiment_negative_mean`	0.941	depressed
`lex_pattern_rate`	0.917	depressed
`sentiment_compound_min`	−0.857	control
`vocabulary_richness`	−0.822	control
`lex_total_rate`	0.803	depressed
`avg_words_per_text`	0.795	depressed
`lex_treatment_mention`	0.788	depressed
`lex_pattern_hits`	0.750	depressed
`n_sentences`	0.727	depressed
`temp_posting_span_days`	0.676	depressed
`n_words`	0.661	depressed
`conv_initiation_ratio`	−0.626	control
`conv_context_negative_pct`	0.600	depressed

frequency cutoffs of 3 and 0.95, and sublinear term frequency. The sparse representation is concatenated with StandardScaler-normalized handcrafted vectors, yielding a $(909, 10{,}051)$ feature matrix (10,000 TF-IDF plus 51 handcrafted columns). We compare logistic regression (LR), linear SVM, random forest (RF), and gradient boosting (GB) under five-fold stratified cross-validation with balanced class weights. Table 4 and Figure 3 report mean F1, precision, and recall; linear SVM achieves the highest F1 (0.819) and is saved as `classical_ml_detector.joblib` for live Run 0.

For the transformer detector mapped to live Run 1, the `TransformerDetector` fine-tunes `mental/mental-bert-base-uncased` for three epochs with learning rate 2×10^{-5}, batch size 16, AdamW with 10% warmup, and gradient clipping. Training uses class-weighted loss to mitigate label imbalance, and long histories are encoded with head-and-tail truncation to 512 tokens. The resulting

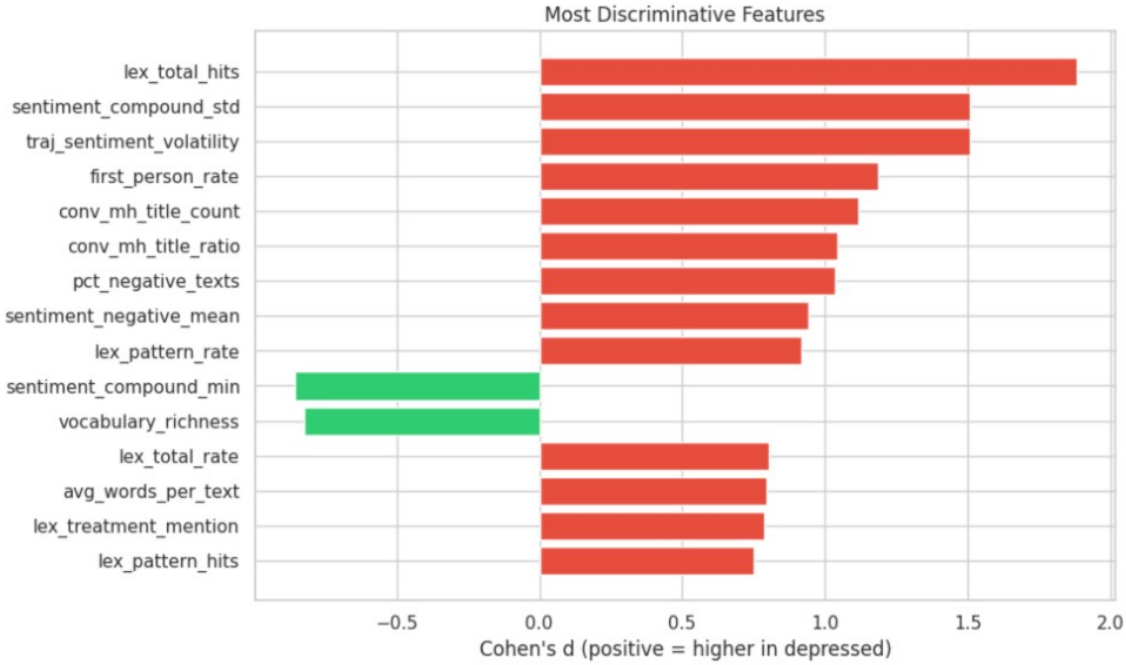

Figure 2: Top 15 features by |Cohen's d| (training set). Red bars: higher mean in depressed subjects; green bars: higher mean in controls.

Table 4
Five-fold cross-validation: classical ML on TF-IDF + handcrafted features ($909 \times 10{,}051$). Best F1 in bold.

Model	F1	Precision	Recall
LR	0.816	0.739	0.911
SVM	**0.819**	0.777	0.871
RF	0.648	0.893	0.511
GB	0.771	0.813	0.754

checkpoint is written under `run2_mentalbert_finetuned/` for Colab transfer and for live loading from disk.

For the ensemble mapped to live Run 2, the `EnsembleDetector` fuses a rule-based score derived from lexicon and conversational features with the classical ML probability and the transformer probability. Default weights are 0.2, 0.5, and 0.3 for the rule, ML, and transformer branches respectively; if the transformer is unavailable, weights fall back to 0.3, 0.7, and 0.0. The rule-based branch caps lexicon contributions and uses negative sentiment and mental-health thread indicators.

3.5. Early decision policy

Alert timing is handled by `EarlyDecisionPolicy` with an ERDE-inspired strategy as the default. The delay parameter is o=5, the base confidence threshold is τ=0.7, and no decision is emitted before three writings are observed. For round index k, time pressure is defined as $p = \min(1, k/(3o))$, which lowers the positive threshold to $\tau_{\text{pos}} = \tau(1 - 0.45p)$. The risk score combines an exponentially weighted average of the last five probabilities, with weights from 0.5 to 1.0, and the running maximum according to $r = 0.55\,\text{EWMA} + 0.45\,\max p$. An alert fires when $r \geq \tau_{\text{pos}}$. A negative decision is permitted when r falls below a relaxed negative threshold or after $k \geq 4o$ rounds with consistently low risk. Alternative strategies such as fixed threshold, patience, and adaptive rules are implemented for ablation but were not used in submitted live runs.

Once `decision=1` is emitted for a user on a given run, the client locks the alert in accordance with task rules but continues to POST updated score values each round. Users without new writings in a

Figure 3: Cross-validation F1, precision, and recall for LR, SVM, RF, and GB (TF-IDF + handcrafted features).

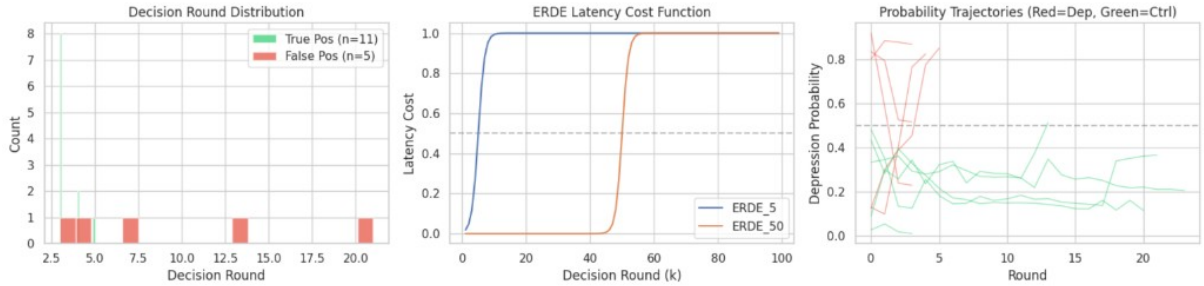

Figure 4: Offline early-detection replay (classical ML, 100 subjects): decision round histogram for true and false positives (left), ERDE latency cost vs. round k for o=5 and o=50 (center), and per-round depression probabilities for five depressed (red) and five control (green) subjects (right).

round reuse the last probability as input to the policy.

3.6. Offline simulation

The `EarlyDetectionSimulator` replays each subject round by round, calling `predict_proba` on truncated histories and applying the same early-decision policy used in the live client. The `ERDEEvaluator` then aggregates ERDE_o for $o \in \{5, 50\}$ together with F1, precision, recall, and median alert latency. The offline table in Section 4 reports the classical ML detector on 100 held-out subjects, while development runs for MentalBERT and the ensemble follow the same simulator. Figure 4 summarizes alert timing on that replay: true positives cluster at early rounds (median latency 3), the ERDE latency cost rises around $k = o$ for $o \in \{5, 50\}$, and sample probability traces separate most depressed from control subjects before round 20.

3.7. Live REST client

The `ERiskServerClient` implements the official protocol. A GET request to `getdiscussions/{token}` returns the current discussion batch, and an empty list terminates the stream. A POST to `submit/{token}/{run}` sends one JSON record per user for the entire cohort. HTTP calls use exponential backoff with up to five retries. On the first non-empty GET, the client stores the full user list and persists it to `dummy_users_target_subjects_t2.txt` so that later rounds still list users whose streams have ended. Each round saves raw GET JSON under

Table 5
Offline early-detection metrics (classical ML, live Run 0).

Metric	Value
$ERDE_5$	0.110
$ERDE_{50}$	0.090
F1	0.710
Precision	0.688
Recall	0.733
Median latency	3

`output_discussions_t2/` and POST payloads and responses under `output_submissions_t2/`. Optional client-state dumps write recoverable JSON snapshots with redacted tokens, accumulated histories, per-run probability traces, and alert state.

The live driver registers five detectors corresponding to classical machine learning, MentalBERT, the ensemble, and two padding slots that POST zero decisions and scores for every user. The team token is read from configuration or from the environment variable `ERISK_T2_TOKEN` and is never committed to the repository.

4. Results

4.1. Offline simulation (classical ML, 100 subjects)

We replayed a held-out set of 100 labeled subjects with the classical ML detector (live Run 0). Table 5 summarizes the early-detection metrics.

This export produced 16 alerts, and 15 of them correspond to true positives on depressed users according to the ground truth.

4.2. Live logs (partial)

Our archived server traffic includes discussion snapshots through round 129, POST payloads saved at rounds 0, 5, 53, and 130, and a full request/response log covering rounds 0 to 112. The request log shows that internal Run 0 (classical ML) returned a score of exactly 0.0 and `decision=0` for every one of the 523 users in all 113 logged rounds: this run never produced a working score during live deployment, unlike its offline cross-validation F1 of 0.816-0.819 (Table 4), which points to a live-side failure (e.g., a feature-building or model-loading fault specific to the REST client) rather than a genuinely low-risk cohort. Internal Run 1 (MentalBERT) behaved as expected: it accumulated persistent alerts from early rounds onward and had issued 109 alerts by round 53. Internal Run 2 (ensemble) also produced varying nonzero scores and accumulated thousands of `decision=1` records over rounds 0 to 112, but the last archived snapshot at round 130 (301 users, after the cohort had shrunk) shows every score reset to exactly 0.0 with no alerts, which is consistent with the run's accumulated alert state being lost across a session restart. Runs 3 and 4 never emitted a `decision=1` in the logged rounds, as expected for zero-decision padding.

4.3. Evaluation against released test labels

Organizers released user-level depression labels for 522 test subjects (control vs. depressed). We scored our saved live submissions by matching user identifiers to those labels.

At round 53, Run 1 issued 109 alerts, including 21 false positives on control users when scored against the released golden labels. Run 0 issued no alerts at any archived snapshot, consistent with the zero-score failure described above rather than with its offline F1 of approximately 0.71 on a 100-subject replay (Table 5). Run 2 and Run 4 also show zero alerts at their respective archived snapshots (round 130

Table 6
Live submissions vs. golden labels (latest archived round per run). Run indices are our internal API indices (Section 3), not the organizer's numbering used in Tables 7–8; see Section 4.4 for the crosswalk.

Run	Detector	Round	Users	F1
0	Classical ML	0	523	0.00
1	MentalBERT	53	523	0.74
2	Ensemble	130	301	0.00
4	Padding	53	523	0.00

Table 7
VANGUARD Task 2 decision-based scores (lab overview). Run indices are the organizer's own numbering: Run 0 is our internal Run 1 (MentalBERT), Run 1 is our internal Run 2 (ensemble); see Section 4.4.

Run	P	R	F1	ERDE_5	ERDE_{50}	lat. TP	F_{latency}
0	0.90	0.62	0.74	0.14	0.05	6.5	0.72
1	0.45	0.95	0.61	0.17	0.11	16.0	0.58

and round 53); for Run 4 this is expected zero-decision padding, but for Run 2 it reflects the round 130 state reset noted above rather than the run's behavior over rounds 0 to 112, when it was actively issuing alerts. Table 6 therefore misrepresents Run 0 and Run 2 relative to their live behavior: Run 0 was non-functional throughout, and Run 2's zero-alert row reflects only its last archived snapshot, not its mid-stream activity. Computing ERDE on the live stream requires organizer timing-aware evaluation, so Table 6's F1 values and Run 1's 109-alert, 21-false-positive count are standard classification metrics evaluated at the saved decision point.

4.4. Organizer evaluation

Task 2 results are reported in the eRisk 2026 lab overview [1]. The overview reports only two runs for VANGUARD, labeled Run 0 and Run 1. Our client, however, posts all five API indices (0–4) every round, so these two labels cover only part of what we submitted. The organizer's Run 0/Run 1 numbering is simply a sequential count over the runs it chose to score, and it is not guaranteed to match our internal indices from Section 3.

VANGUARD submitted two runs and processed 129 of 500 user threads within 12 hours and 42 minutes, which is partial coverage compared with ten teams that completed all 500 threads.

Run 0 balances high precision (0.90) with F1 of 0.74, achieves ERDE_5 of 0.14, and yields a median true-positive latency of 6.5 writings with $F_{\text{latency}} = 0.72$. Run 1 maximizes recall at 0.95 but reduces precision to 0.45 and slows alerts, with a median true-positive latency of 16.0 writings. We can resolve most of this mapping using our request/response log (Section 4.2). Internal Run 0 (classical ML) returned a zero score and `decision=0` for every user across all 113 logged rounds, so it never issued a real alert during live deployment; a run with zero recall cannot score $P = 0.90$, $R = 0.62$, F1$= 0.74$ or $P = 0.45$, $R = 0.95$, F1$= 0.61$, which rules out internal Run 0 as either organizer row. Internal runs 3 and 4 are zero-decision padding by design and are likewise excluded. That leaves internal Run 1 (MentalBERT) and internal Run 2 (ensemble) to account for the two organizer rows. The F1 score of 0.74 obtained by the organizer Run 0 matches our MentalBERT submission (internal Run 1) in round 53 in Table 6, so we identify the organizer Run 0 as our internal Run 1 (MentalBERT). By elimination, and because our log shows internal Run 2 (ensemble) actively accumulating alerts over rounds 0–112 before the round 130 state reset described in Section 4.2, we identify organizer Run 1 as our internal Run 2 (ensemble); its higher recall (0.95) and more aggressive alerting profile are consistent with the ensemble's mid-stream behavior. In summary: of our five API run indices, only internal Run 1 (MentalBERT) and internal Run 2 (ensemble) contributed to the official scores, reported as organizer Run 0 and Run 1 respectively; internal Run 0 (classical ML) failed live and contributed nothing; internal Runs 3 and 4 were zero-decision

Table 8
VANGUARD Task 2 ranking-based scores (nDCG@100; lab overview). Run indices follow the same organizer numbering as Table 7(Run 0 is internal Run 1/MentalBERT, Run 1 is internal Run 2/ensemble).

Run	@1 wrt.	@100 wrt.	@250 wrt.	@500 wrt.
0	0.71	0.75	0.75	0.75
1	0.67	0.80	0.80	0.80

padding and were never intended to be scored.

Ranking-based scores in the overview (Table 8) use nDCG@100 at 1, 100, 250, and 500 writings seen. Both submitted runs reach nDCG@100 between 0.75 and 0.80 once at least 100 writings are available.

5. Discussion and conclusion

We combine three detectors, an ERDE-oriented alert policy, and a logged REST client. Dialogue is modeled with handcrafted conversational features rather than reply-graph encoders, live logs are incomplete, and thresholds were not tuned separately for control users. MentalBERT training requires GPU resources. Two live-deployment failures limit our results independently of modeling choices: the classical ML run (internal Run 0) never produced a nonzero score over 113 logged rounds despite strong offline cross-validation performance, and the ensemble run (internal Run 2) appears to have lost its accumulated alert state around round 112, resetting to zero decisions before the stream ended (Section 4.2). Only the MentalBERT run was unaffected.

Future work may add thread-level encoders, calibrate thresholds on controls, and report ablations for all three detectors, harden the client against the state loss observed for the ensemble run, and add a startup self-check that would have caught the classical ML run's zero-score failure before deployment.

We presented VANGUARD's contextualized early depression pipeline for eRisk 2026 Task 2. On held-out labeled subjects, the classical run reaches offline $\mathrm{ERDE}_5 \approx 0.11$, while live MentalBERT submissions achieve F1 = 0.74 at round 53 on 523 users with golden labels. Our participation covered 129 of 500 user threads within the time budget reported by organizers, and the logged REST client documents partial but reproducible server interaction.

Acknowledgments

The research presented in this paper was supported in part by (1) The Academy of Romanian Scientists, through the funding of the project "NetGuardAI: Intelligent system for harmful content detection and immunization on social networks" (AOȘR-TEAMS-IV); and (2) the National University of Science and Technology, POLITEHNICA Bucharest, through the PubArt program.

Declaration on Generative AI

During the preparation of this work, the authors used GPT-4 to perform grammar and spelling checks, paraphrase, and reword. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] A. Perez, J. Parapar, X. Wang, F. Crestani, Overview of eRisk 2026 Early Risk Prediction on the Internet: Symptom Ranking and Conversational Approaches for Depression and ADHD (Extended Overview), in: Working Notes of the Conference and Labs of the Evaluation Forum (CLEF 2026),

Jena, Germany, 21–24 September, 2026, volume To be published of *CEUR Workshop Proceedings*, CEUR-WS.org, 2026.

[2] J. Parapar, A. Perez, X. Wang, F. Crestani, eRisk 2025: Contextual and Conversational Approaches for Early Detection of Depression, in: ECIR Workshops, 2025. Contextualized early detection pilot; multi-party threads.

[3] D. E. Losada, F. Crestani, A test collection for research on depression and language use, in: International conference of the cross-language evaluation forum for European languages, Springer, 2016, pp. 28–39.

[4] J. Parapar, P. Martín-Rodilla, D. E. Losada, F. Crestani, Overview of erisk 2023: Early risk prediction on the internet, in: International conference of the cross-language evaluation forum for European languages, Springer, 2023, pp. 294–315.

[5] R. Chiong, G. S. Budhi, S. Dhakal, F. Chiong, A textual-based featuring approach for depression detection using machine learning classifiers and social media texts, Computers in Biology and Medicine 135 (2021) 104499.

[6] M. De Choudhury, M. Gamon, S. Counts, E. Horvitz, Predicting depression via social media, in: Proceedings of the international AAAI conference on web and social media, volume 7, 2013, pp. 128–137.

[7] D. William, D. Suhartono, Text-based depression detection on social media posts: A systematic literature review, Procedia Computer Science 179 (2021) 582–589.

[8] W. A. Gadzama, D. Gabi, M. S. Argungu, H. U. Suru, The use of machine learning and deep learning models in detecting depression on social media: A systematic literature review, Personalized Medicine in Psychiatry 45 (2024) 100125.

[9] C.-O. Truică, C. A. Leordeanu, Classification of an imbalanced data set using decision tree algorithms, University Politehnica of Bucharest Scientific Bulletin - Series C Electrical Engineering and Computer Science 79 (2017) 69–84.

[10] A. Aldkheel, L. Zhou, Depression detection on social media: a classification framework and research challenges and opportunities, Journal of Healthcare Informatics Research 8 (2024) 88–120.

[11] A. Q. Zulfa, S. A. I. Alfarozi, S. Kusrohmaniah, S. Wibirama, MentalBERT with Contrastive Learning for Mental Disorder Classification from Social Media: A Comparative and Explainable Study, in: 2026 International Conference on Current Research in Artificial Intelligence and Data Science (ICCRAIDS), volume 1, IEEE, 2026, pp. 1–7.

[12] M. Trotzek, S. Koitka, C. M. Friedrich, Utilizing neural networks and linguistic metadata for early detection of depression indications in text sequences, IEEE transactions on knowledge and data engineering 32 (2018) 588–601.

[13] R. Dou, X. Kang, TAM-SenticNet: A Neuro-Symbolic AI approach for early depression detection via social media analysis, Computers and Electrical Engineering 114 (2024) 109071.

[14] T. Althoff, K. Clark, J. Leskovec, Large-scale analysis of counseling conversations: An application of natural language processing to mental health, Transactions of the Association for Computational Linguistics 4 (2016) 463–476.

[15] C. Howes, M. Purver, R. McCabe, Linguistic indicators of severity and progress in online text-based therapy for depression, in: Proceedings of the workshop on computational linguistics and clinical psychology: from linguistic signal to clinical reality, 2014, pp. 7–16.