UP_NLP at Social Support Detection Shared Task, IberLEF 2026

Edgar Daniel Chacón Amaro[1,*], Fernando de la Rosa Flores[1]

[1]Universidad Panamericana, Ciudad de México, México

Abstract

This paper describes the UP_NLP team's submission to the Social Support Detection Shared Task at IberLEF 2026, a three-level hierarchical classification problem over English and Spanish YouTube comments: binary support detection (S1), individual-vs-group target identification (S2), and six-way group categorization (S3). We built two complementary systems: a classical machine-learning pipeline combining word- and character-level TF-IDF features with Logistic Regression, Linear SVM, and Multinomial Naive Bayes; and a transformer pipeline fine-tuning RoBERTa (English) and XLM-RoBERTa (Spanish) with class-weighted loss and three-seed ensemble averaging. Both systems follow the same hierarchical cascade and use class weighting to address the severe imbalance present in all three subtasks (up to 52:1 in S3). On the official test set, the transformer ensemble obtained the best overall macro-F1 (0.856), closely followed by the transformer single model (0.856) and Linear SVM (0.844); Naive Bayes lagged substantially (0.723). The small gap between the transformer and the best classical baseline indicates that well-engineered TF-IDF features remain highly competitive for this short-text, imbalanced, bilingual task.

Keywords

social support detection, transformers, RoBERTa, XLM-RoBERTa, TF-IDF, text classification, IberLEF 2026

1. Introduction

Understanding prosocial language in online communities is an increasingly relevant problem in Natural Language Processing. While sentiment analysis broadly separates content into positive, negative, or neutral, social support—expressions of encouragement, care, admiration, solidar-ity, or help—is a more specific and actionable form of positive language, with applications in community-health monitoring and crisis intervention [1, 7]. Beyond detecting whether a com-ment is supportive, identifying who or what group it targets enables richer downstream analysis of online communities, echoing the target/aspect identification problem studied in aspect-based sentiment analysis. The Social Support Detection Shared Task at IberLEF 2026 [6] poses this as a three-level hierarchical problem over English and Spanish YouTube comments:

- S1: Supportive vs. Not Supportive.
- S2 (on supportive comments): Individual vs. Group.
- S3 (on group-targeted comments): Nation, Other, LGBTQ, Black Community, Religion, Women.

This paper describes the UP_NLP team's submission, which centers on a simple research ques-tion: how do transformer-based models compare to classical ML baselines for this hierarchical, bilingual, imbalanced task? We submit two complementary systems—a TF-IDF+linear-classifier baseline and a fine-tuned RoBERTa/XLM-RoBERTa ensemble—sharing the same preprocessing

and hierarchical prediction cascade. The classical baseline serves both as a reproducible, computationally cheap reference and as a bridge to prior work on this dataset, while the transformer system targets maximum accuracy by leveraging transfer learning from large pre-trained lan-guage models. We evaluate both systems with stratified k-fold cross-validation on the training set and report official macro-F1 scores from the shared-task test phase. In summary, our contributions are: (1) a systematic

IberLEF 2026, León, Spain

*Corresponding author.

0213679@up.edu.mx (E. D. C. Amaro); 0278080@up.edu.mx (F. d. l. R. Flores)

comparison of nine combinations of TF-IDF representation and classifier across all three subtasks and both languages; (2) a class-weighted, multi-seed RoBERTa/XLM-RoBERTa ensemble tailored to the task's hierarchical structure and severe class imbalance; and (3) an analysis of official test-phase results explain-ing why the transformer advantage over the classical baseline, while consistent, is smaller than commonly expected for this short-text, social-media domain. The rest of the paper is organized as follows. Section 2 briefly reviews prior work on this task. Section 3 describes the dataset. Section 4 describes the submitted system. Section 6 reports results and analysis. Section 7 discusses limitations and future work, and Section 8 concludes.

2. Related Work

Ahani et al. [1] introduced social support detection as an NLP task with a foundational English dataset of 10,000 annotated YouTube comments, evaluating classical feature-based models and neural embeddings, and established the annotation scheme adopted by later work, including this shared task. Tash et al. [7] extended the task to Spanish with the first annotated Spanish dataset (3,189 comments), using GPT-4o paraphrasing to balance classes across the inherent imbalance of social-support labels, and found that transformer-based architectures outperform classical machine-learning and deep-learning baselines. Kolesnikova et al. [3] scaled the dataset to 42,695 comments (from an initial 66,272) and reported RoBERTa macro-F1 of 0.78 (S1), 0.84 (S2), and 0.80 (S3), substantially above CNN+GloVe and TF-IDF+LIWC (Linguistic Inquiry and Word Count) baselines. Complementing this line of work, Shahiki-Tash et al. [5] performed a dedicated LIWC-based linguistic analysis of different support types across Spanish and English online communities, characterizing the psycholinguistic markers—such as affective and social-process word categories—that distinguish supportive language across the two languages studied in this shared task. Kolesnikova et al.'s combined word+character TF-IDF representation and use of RoBERTa/XLM-RoBERTa as strong baselines directly motivate our choice of classical and transformer architectures in this work. The Social Support Detection Shared Task (SSD-2026) at IberLEF 2026 [6] builds directly on this line of work, adopting the same three-level hierarchical annotation scheme (S1/S2/S3) over a bilingual English-Spanish YouTube-comment corpus, and evaluating submissions with macro-F1 per subtask. SSD-2026 is one of several shared tasks at the broader IberLEF 2026 evaluation forum, co-located with the 42nd Conference of the Spanish Society for Natural Language Pro-cessing (SEPLN 2026) [2]. This shared-task setting differs from prior single-language studies in requiring a single system design that generalizes across two languages with substantially different training-set sizes and, as we show in Section 3, different S3 group distributions—which is one of the central challenges our system addresses.

2.1. Pre-trained transformers for short-text classification

Pre-trained transformer language models, fine-tuned with a lightweight classification head, are the de-facto standard for text classification tasks, including on short, informal social-media text: their self-attention mechanism lets each token's representation be conditioned on the full surrounding context, which is particularly useful for capturing negation and other context-dependent phenomena that static, bag-of-words representations such as TF-IDF cannot express. RoBERTa is a widely used, robustly-optimized encoder for English; XLM-RoBERTa extends the same architecture to 100 languages via a shared multilingual vocabulary, enabling effective trans-fer to lower-resource languages such as Spanish without language-specific pre-training. These properties motivate our choice of RoBERTa/XLM-RoBERTa as the transformer backbone for the bilingual, short-text setting of this shared task.

3. Dataset

The dataset consists of English and Spanish YouTube comments annotated at the three hier-archical levels described above. Table 1 summarizes the splits; English is roughly three times larger than Spanish.

Split	English	Spanish
Train	7998	2600
Test	2000	651

Table 1
Dataset statistics for English and Spanish.

3.1. Label distribution and class imbalance

S1. Roughly 22% of comments are supportive in both languages: 1,795 of 7,998 English com-ments (22.44%, a 3.46:1 non-supportive ratio) and 590 of 2,600 Spanish comments (22.69%, 3.41:1). This imbalance is intrinsic to the task—supportive comments are genuinely rarer than neutral or critical ones—so macro-F1, which weights both classes equally regardless of frequency, is essential for assessing true discriminative ability. Figure 1 visualizes this asymmetry for both languages.

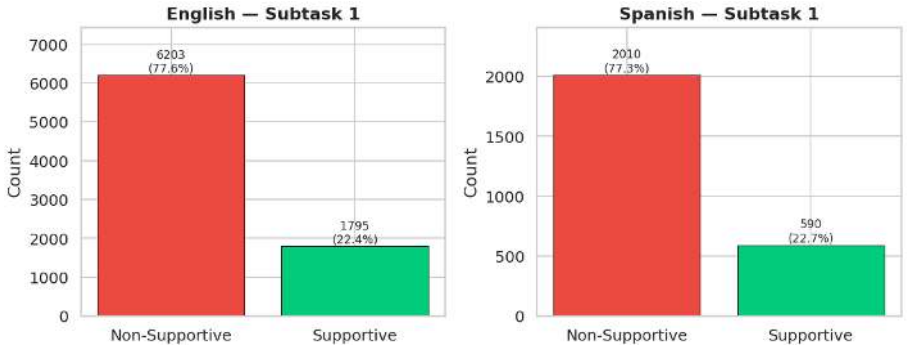

Figure 1: Class distribution for Subtask 1 in English (left) and Spanish (right). Both languages show a roughly 3.5:1 ratio of non-supportive to supportive comments.

S2. Among supportive comments, Group-targeted support dominates: 1,450 of 1,795 English comments (80.78%) and 456 of 590 Spanish comments (77.29%), a roughly 4:1 imbalance that particularly challenges probabilistic classifiers (Figure 2).

S3. S3 is the most extreme case. English is dominated by Nation (54.6%), followed by Other (28.9%), LGBTQ (8.5%), Black Community (6.3%), Women (1.3%), and Religion (1.0%, only 15 samples 52.4:1 ratio versus Nation). Spanish shows a markedly different distribution, dominated instead by LGBTQ (43.0%), reflecting cultural differences between the two data sources (Table 2 and Figure 3).

3.2. Text statistics and lexical analysis

English comments average 18.68 words (median 13.0,125.56 characters); Spanish comments are somewhat longer, averaging 25.36 words (median 13.0). Both distributions are right-skewed, with the bulk of comments under 50 words (Figure 4). Supportive comments tend to be slightly longer than non-supportive ones on S1, though the difference is modest and not a strong standalone signal for S2/S3 (Figure 5).

Figure 6 compares the top-20 most frequent terms overall, in supportive comments, and in non-supportive comments. Supportive comments are dominated by solidarity-oriented vocabulary, while non-supportive comments show a more diverse and critical vocabulary—this lexical separation supports the feasibility of both a lexical (TF-IDF) and a contextual (transformer) approach to the task.

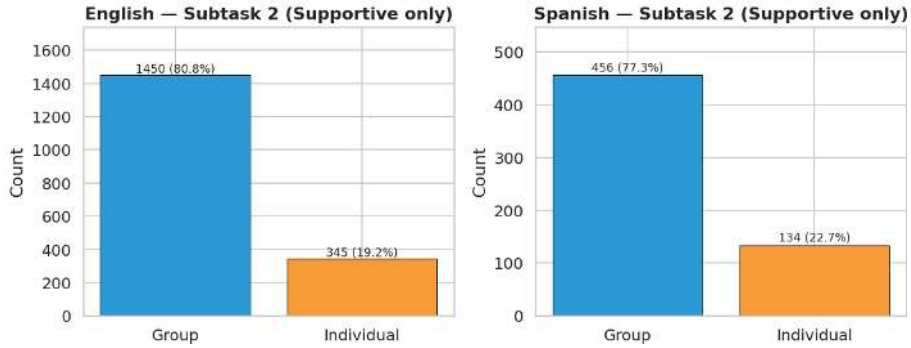

Figure 2: Class distribution for Subtask 2 in English (left) and Spanish (right). Group-targeted support dominates in both languages.

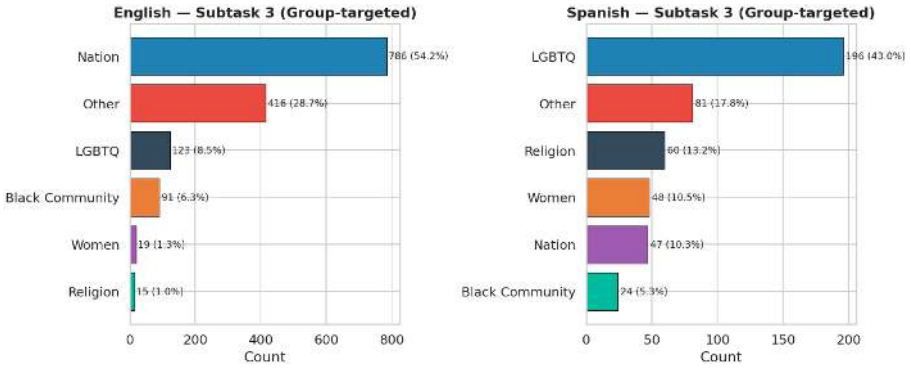

Figure 3: Group-level class distribution for Subtask 3 in English (left) and Spanish (right). English is dominated by Nation (54.6%), Spanish by LGBTQ (43.0%); minority classes have very few samples in both languages.

	English		Spanish	
Group	Count	%	Count	%
Nation	786	54.6	47	10.3
Other	416	28.9	81	17.8
LGBTQ	123	8.5	196	43.0
Black Community	91	6.3	24	5.3
Women	19	1.3	48	10.5
Religion	15	1.0	60	13.2
Total	1,450	100.0	456	100.0

Table 2
Group distribution for Subtask 3, English and Spanish training sets.

Figure 7 shows Jaccard vocabulary-overlap similarity between the six S3 group categories. Groups with high overlap (e.g., Nation and Other) are inherently harder to distinguish from bag-of-words features alone, foreshadowing part of the S3 error pattern discussed in Section 6.

3.3. Implications for system design

Three dataset properties directly shaped the system described in Section 4. First, the consistent ∼3–4:1 imbalance in S1/S2 and the much more severe imbalance in S3 (up to 52.4:1) motivate class weighting in both the classical (via classifier-level class weights, implicit in the margin-maximization objective of SVM) and transformer (via weighted cross-entropy) pipelines. Second, the short, noisy, informal nature of the comments (mean 13–25 words, frequent misspellings and slang) motivates including character-level n-grams alongside word-level ones in the classical representation, since character features are

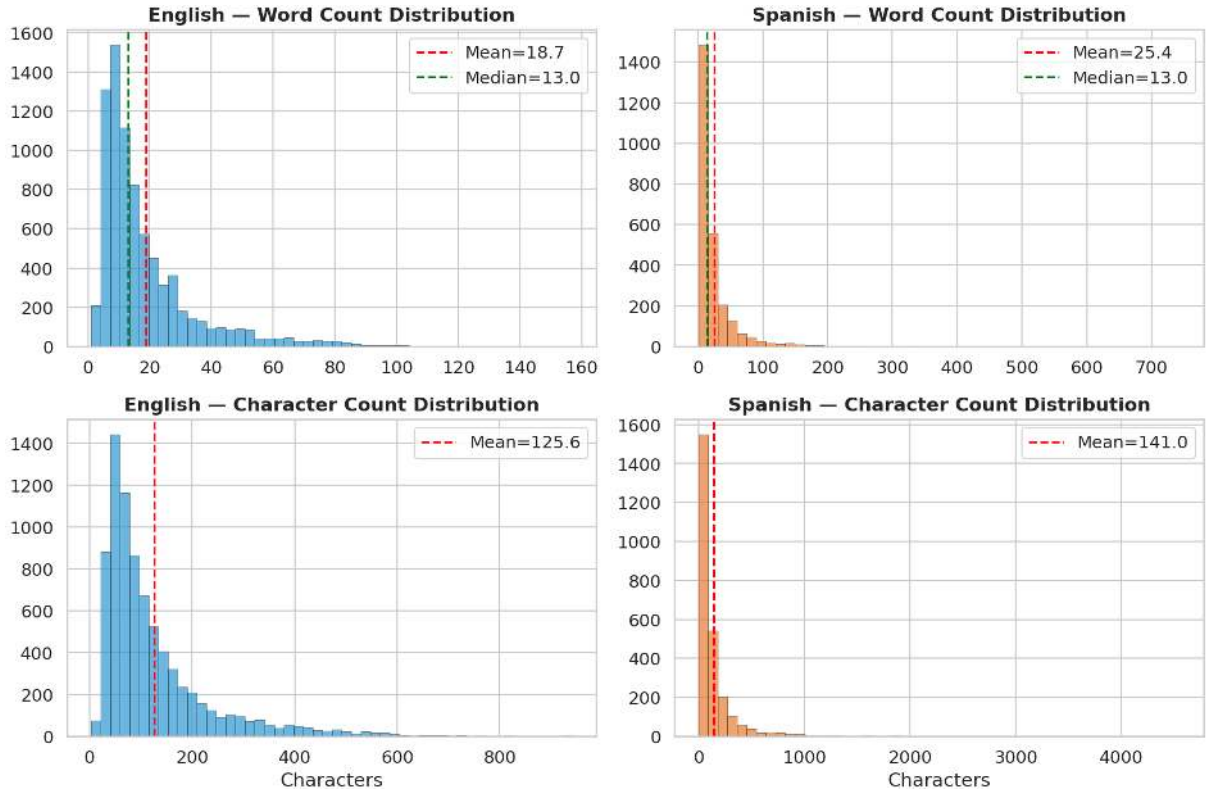

Figure 4: Word- and character-count distributions for English (left) and Spanish (right) comments. Both languages show right-skewed distributions typical of social-media text.

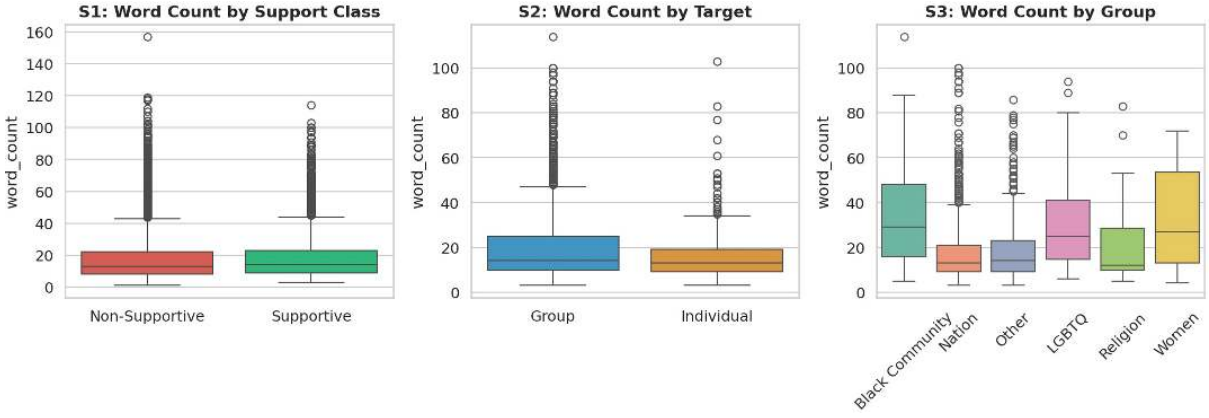

Figure 5: Word count by class for S1, S2, and S3 (English). Supportive comments are slightly longer on S1; no substantial difference is visible for S2/S3.

more robust to the morphological variation typical of user-generated content. Third, the divergent S3 group distributions between English (Nation-dominated) and Spanish (LGBTQ-dominated) mean that a single global class-weighting scheme is insufficient; weights must be computed independently per language and per cross-validation fold, as described in Section 4.1.

4. System Description

We submit two complementary systems that share the same input preprocessing and hierarchical cascade (Figure 8): raw comments are lowercased; URLs and @-mentions are stripped (they carry minimal semantic content for support detection); non-alphanumeric noise is removed while preserving punctuation that may carry emphasis (e.g., exclamation marks); and, for the classical pipeline only,

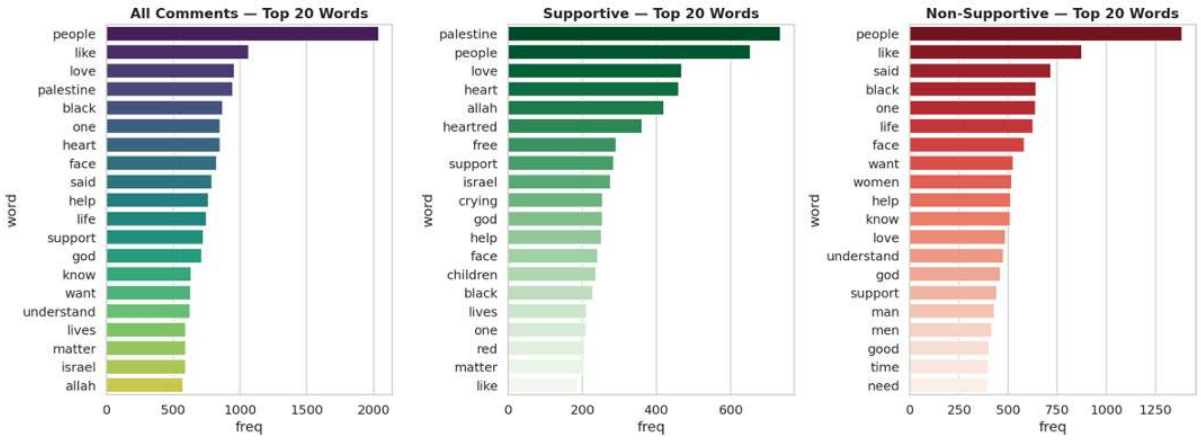

Figure 6: Top 20 most frequent words (after stopword removal) overall (left), in supportive comments (center), and in non-supportive comments (right).

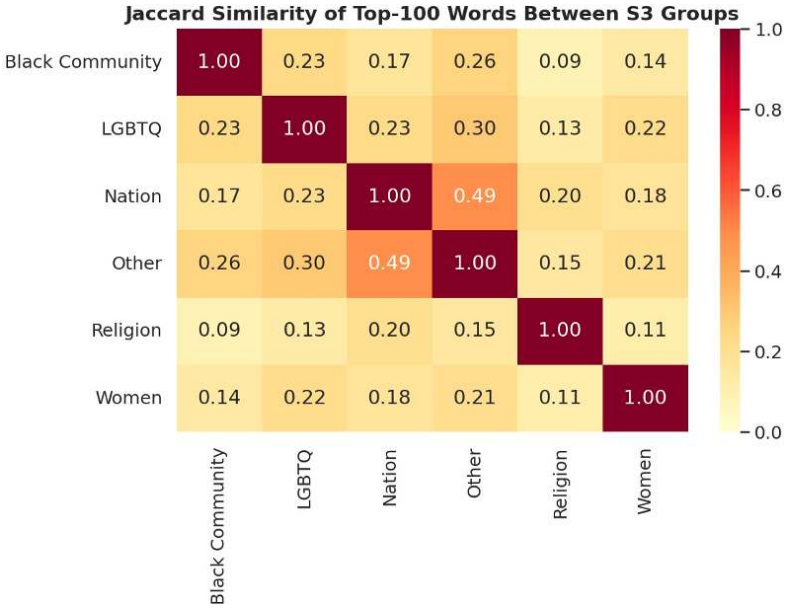

Figure 7: Jaccard similarity heatmap of the top-100 most frequent words per S3 group category (English). Higher values indicate greater vocabulary overlap.

common stopwords are additionally removed before TF-IDF vectorization (the transformer pipeline instead uses byte-pair-encoding subword tokenization, which handles out-of-vocabulary words and morphological variation directly, so stopword removal is unnecessary and would discard information the self-attention mechanism could otherwise use). Each system then predicts S1, applies S2 only to comments predicted supportive, and applies S3 only to comments predicted Group-targeted by S2. This cascade respects the task hierarchy but can propagate S1/S2 errors downstream.

Did preprocessing help? We did not run a formal ablation comparing model performance with and without each preprocessing step, so we cannot report a quantitative effect size, and we avoid overstating one. Qualitatively, we expect the effect to be modest but net positive for the classical pipeline: URL and @-mention removal eliminates tokens that are typically unique per comment (usernames, links) and would otherwise inflate the TF-IDF vocabulary without generalizing across comments, while stopword removal reduces dimensionality without discarding class-discriminative content, since function words carry little signal for support detection. For the transformer pipeline, lowercasing and URL/mention stripping likely have a smaller effect, since the BPE tokenizer and self-attention mechanism can already down-weight uninformative tokens; this is consistent with our decision not to apply stopword removal

there. We flag the absence of a preprocessing ablation as a limitation in Section 7.

Throughout this work, our primary metric is macro-F1, the mean of the per-class F1 score,

$$\text{Macro-F1} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} F_1(c), \qquad F_1(c) = \frac{2 \ \text{Precision}(c) \ \text{Recall}(c)}{\text{Precision}(c) + \text{Recall}(c)}. \tag{1}$$

which weights every class equally regardless of frequency and is therefore appropriate for the imbalanced label distributions described in Section 3; it is also the official ranking metric of the shared task.

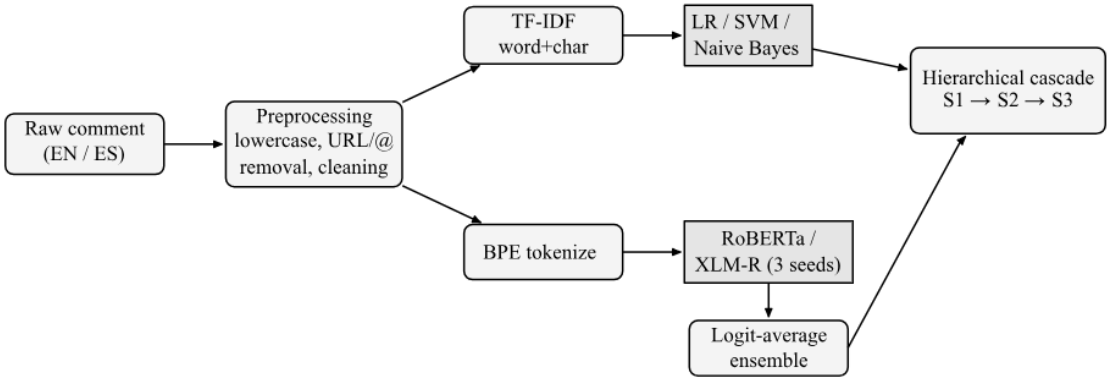

Figure 8: Shared preprocessing and prediction pipeline. The classical branch (top) uses TF-IDF features with a linear classifier; the transformer branch (bottom) fine-tunes RoBERTa/XLM-RoBERTa with three-seed logit-averaging. Both branches apply the same S1 → S2 → S3 hierarchical cascade at inference.

4.1. Classical baseline

Text representation. Each comment is represented with three TF-IDF (Term Frequency-Inverse Document Frequency) vector-space representations, computed as $\text{TF-IDF}(t, d) = \text{TF}(t, d) \cdot \log(N/n_t)$ for term t in document d, with N the corpus size and n_t the number of documents containing t: (1) word-level, unigrams+bigrams capped at 20k features; (2) character-level, 3–5 character n-grams capped at 30k features, which capture morphological patterns, misspellings, and domain-specific slang common in social media; and (3) their L_2-normalized concatenation (word+char, up to 50k features), implemented with scikit-learn's FeatureUnion [4], which fits the word- and character-level vectorizers independently and horizontally stacks their outputs into a single sparse feature matrix, following the combined-representation strategy of Kolesnikova et al. [3].

Classifiers. We train and compare three classifiers on top of each representation. Logistic Regression learns a linear decision boundary by optimizing log-likelihood; it is a strong, well-calibrated baseline but can be pulled toward the majority class under severe imbalance. Linear SVM instead maximizes the margin between classes,

$$\min_{\mathbf{w}, b} \ \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i. \tag{2}$$

which is comparatively more robust to imbalance since its objective depends on the margin rather than on class frequency; for multiclass S3 it uses a one-vs-rest decomposition, so each of the six binary subproblems can independently maximize its margin. Multinomial Naive Bayes is included as a lightweight, probabilistic lower-bound reference; its conditional-independence assumption is a poor fit for natural language and, combined with the very small minority classes in S3 (15–19 samples), leads to noisy probability estimates and, in the worst case, numerical underflow when multiplying thousands of sparse feature likelihoods.

Evaluation protocol. All nine (representation, classifier) combinations are evaluated per subtask and language via 5-fold stratified cross-validation with a fixed random seed, so that class proportions are preserved in every fold. Macro-F1—the mean of per-class F1 scores, giving equal weight to each class

regardless of frequency—is our primary metric throughout this work. Vectorizers are refit independently on each training fold (via factory functions) to avoid vocabulary/IDF leakage across folds, which would otherwise produce overly optimistic estimates.

4.2. Transformer approach

We fine-tune RoBERTa-base for English and XLM-RoBERTa-base for Spanish, using byte-pair-encoding tokenization (max. sequence length 128). Table 3 lists the training configuration, shared across subtasks and languages except for the number of epochs, which we scale with task difficulty (5 for S1, 8 for S2, 10 for S3, reflecting that harder tasks benefit from longer training). Gradient clipping (max-norm 1.0) bounds the L_2 norm of the gradient to prevent exploding gradients during fine-tuning; a linear warmup over the first 10% of steps avoids large, destabilizing updates while the classification head is still poorly calibrated.

To counter class imbalance we use a class-weighted cross-entropy loss,

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} w_c\, y_{i,c} \log \hat{p}_{i,c}, \qquad w_c = \frac{N}{C\, n_c}. \tag{3}$$

where n_c is the number of training samples in class c; this inversely-weights the loss so minority classes are penalized more heavily when misclassified, directly adjusting the optimization objective rather than relying on synthetic resampling. For each subtask we train three models with different random seeds (42,123,456) and average their output logits at inference,

$$\hat{\mathbf{z}}_{\text{ens}} = \frac{1}{3}\sum_{k=1}^{3} \mathbf{z}_k,$$

before applying softmax. This logit-averaging ensemble reduces variance from random initialization in the non-convex fine-tuning landscape at negligible extra implementation cost.

Hyperparameter	Value
Optimizer	AdamW
Learning rate	2×10^{-5}
Weight decay	0.01
Warmup ratio	0.1 (linear)
Batch size	32
Max. sequence length	128 tokens
Gradient clipping	max-norm 1.0
Epochs (S1 / S2 / S3)	5 / 8 / 10
Seeds (ensemble)	42, 123, 456

Table 3
Transformer training configuration.

At inference, models are applied hierarchically: S1 first; the S2 model only on comments predicted supportive; the S3 model only on comments predicted Group-targeted by S2. Three implementation details were important for correctness: (1) the S3 model must be trained only on the group-targeted supportive subset (1,450 English / 456 Spanish comments), not the full training set; (2) at inference, S3 is applied only where S2 predicts "Group"; and (3) submitted predictions use the numeric label format and column names required by the evaluation server (0/1 for S1/S2, 0–5 for S3).

Why contextualization matters here. Unlike TF-IDF, which assigns each n-gram a fixed weight regardless of context, a transformer's self-attention layers let every token's representation be updated by the other tokens around it. For a comment such as "I truly hope and stand with you in solidarity", attention lets the model combine "hope", "stand with", and "solidarity" into a single strongly-supportive

representation; for "I don't support this ideology", it lets the representation of "support" be modified by the nearby negation. TF-IDF records only that the trigram "support" occurs in both cases and weights it identically, which is one source of the transformer's advantage on S1/S2 observed in Section 6.

4.3. Implementation and submitted runs

Transformer fine-tuning uses the Hugging Face Transformers library; classical experiments use scikit-learn [4]. Training is done on a single GPU (NVIDIA T4/V100/A100; ∼20–30 minutes per subtask for transformer fine-tuning); classical experiments run on CPU in 1–2 hours per subtask/language. Based on cross-validation, we submitted five runs to the official evaluation: three classical configurations (Logistic Regression, Linear SVM, and Naive Bayes, all with the word+char representation) and two transformer configurations (single seed and three-seed ensemble).

Reproducibility. All classical experiments use a fixed random seed (42) for cross-validation splitting and model initialization, and vectorizers are rebuilt from factory functions on each fold to avoid vocabulary leakage (Section 4.1). Transformer experiments fix the three ensemble seeds (42,123,456) and use deterministic tokenization; remaining sources of run-to-run variance are primarily GPU-level (non-deterministic CUDA kernels), which is a known limitation of transformer fine-tuning noted in Section 7.

5. Experimental Setup

For classical ML, we perform an exhaustive grid search over all nine (representation, classifier) configurations per subtask per language, recording macro-F1 and per-class metrics across all five CV folds; the configuration with the highest mean macro-F1 is selected per subtask and language. For transformer models, exhaustive 5-fold cross-validation with three seeds is computationally prohibitive (∼45 GPU-hours per subtask); our intended protocol was a lighter 3-fold stratified cross-validation with a single seed per fold, run alongside final ensemble training. In practice, given our GPU-time budget before the submission deadline, we prioritized completing the three-seed ensemble and final training runs required for the official submission over this separate CV pass, so no transformer cross-validation estimate is reported in this paper (Section 6.3); the only transformer results reported are the official test-phase scores from the shared task's evaluation server (Section 6.4). Classical ML experiments run on CPU (1–2 hours per subtask/language); transformer experiments run on GPU (NVIDIA T4 16GB / V100 32GB / A100 80GB), with final ensemble training (three seeds, all subtasks, both languages) requiring roughly 1–2 GPU-hours in total—feasible on cloud services such as Google Colab Pro.

6. Results and Analysis

6.1. Cross-validation (classical baseline, English)

Tables 4–6 report the full 5-fold CV macro-F1 for all nine (representation, classifier) configurations on English. Across all three subtasks, the combined word+char representation ranks first regardless of classifier, confirming that word-level (semantic) and character-level (morphological, robust to misspellings) signals are complementary. Character-only features are notably strong on the harder subtasks, outperforming word-only with Linear SVM on both S2 (0.7657 vs. 0.7614) and S3 (0.6819 vs. 0.6661), suggesting that robustness to morphological variation matters more as the training signal weakens.

Why SVM overtakes Logistic Regression as difficulty grows. On S1, Logistic Regression marginally edges Linear SVM (0.7826 vs. 0.7789); on S2 the gap widens to 8.4 points in SVM's favor (0.7744 vs. 0.6904), and further to 13.1 points on S3 (0.6921 vs. 0.5615). Logistic Regression's log-likelihood objective is driven by the majority class under severe imbalance, whereas Linear SVM's margin-maximization objective is comparatively insensitive to class frequency; on S3, its one-vs-rest decomposition additionally lets each of the six binary subproblems maximize its own margin

Representation	Classifier	Macro-F1
word+char	LogReg	0.7826
word	LinearSVM	0.7824
word+char	LinearSVM	0.7789
char	LinearSVM	0.7784
char	LogReg	0.7666
word	LogReg	0.7540
word+char	NaiveBayes	0.7400
char	NaiveBayes	0.7330
word	NaiveBayes	0.6760

Table 4
CV results — S1: Support Detection (English).

independently, which is more stable than jointly optimizing a six-way softmax when some classes have only 15 samples.

Representation	Classifier	Macro-F1
word+char	LinearSVM	0.7744
char	LinearSVM	0.7657
word	LinearSVM	0.7614
word+char	LogReg	0.6904
char	LogReg	0.6468
word	LogReg	0.5445
word+char	NaiveBayes	0.4643
char	NaiveBayes	0.4497
word	NaiveBayes	0.4468

Table 5
CV results — S2: Target Identification (English).

Representation	Classifier	Macro-F1
word+char	LinearSVM	0.6921
char	LinearSVM	0.6819
word	LinearSVM	0.6661
word+char	LogReg	0.5615
char	LogReg	0.5216
word	LogReg	0.4993
word+char	NaiveBayes	0.2250
char	NaiveBayes	0.1838
word	NaiveBayes	0.1806

Table 6
CV results — S3: Group Categorization (English).

Why Naive Bayes fails on S3. Naive Bayes drops from a respectable 0.74 on S1 to just 0.18–0.23 on S3, far below SVM (0.6661–0.6921). Three compounding factors explain this: its independence assumption is violated by highly correlated words in natural language; minority classes with only 15–19 samples yield noisy class-conditional probability estimates that Laplace smoothing cannot fully correct; and multiplying thousands of sparse feature likelihoods with very few samples risks numerical underflow.

Performance drops sharply and monotonically across subtasks for the best classical configuration ($0.7826 \rightarrow 0.7744 \rightarrow 0.6921$), with a much steeper drop from S2 to S3 (10.6%) than from S1 to S2 (0.8%)—consistent with S3's smaller training set (1,450 vs. 7,998 samples for S1), higher imbalance

(52.4:1 vs. 3.46:1), and six-way complexity. Per-class inspection of the best S3 model shows the familiar imbalance signature: the majority class (Nation) has high recall but comparatively lower precision (over-prediction), while minority classes (Religion, Women) have very low recall, being frequently misclassified as Nation or Other.

Confusion matrices (cross-validation, training set). To inspect this error pattern directly, Figure 9 shows confusion matrices for the best classical configuration (Linear SVM, word+char), aggregated from out-of-fold predictions during 5-fold cross-validation on the English training set. These are not computed on the official test set: as noted in Section 7, the official test labels are not released to participants, so no test-set confusion matrix can be produced by us for any model in this paper. The class counts shown therefore match the training-set sizes in Tables 1 and 2 (e.g., 1,450 total S3 predictions, matching the S3 training set). For S1, the model correctly identifies most non-supportive comments (5,451 true negatives out of the training set) but misses 466 supportive comments as false negatives, consistent with the majority-class bias typical of imbalanced binary classification. For S2, predictions strongly favor the majority Group class (1,360 correct), with 122 Group comments confused as Individual. For S3, the diagonal shows that Nation and LGBTQ dominate correct predictions: the majority class (Nation, 786 samples) is over-predicted (high recall, lower precision); Other (416 samples) shows balanced precision/recall; LGBTQ (123 samples) has reasonable precision when confident but lower recall from limited data; and Religion/Women (15–19 samples) are almost never correctly predicted, being systematically absorbed into Nation or Other. This pattern—majority-class over-prediction paired with minority-class near-total miss rate—is the main source of residual error in cross-validation and, given the similar per-subtask trends in the official test results (Section 6.4), plausibly persists on the test set as well, though we cannot confirm this directly without access to test labels.

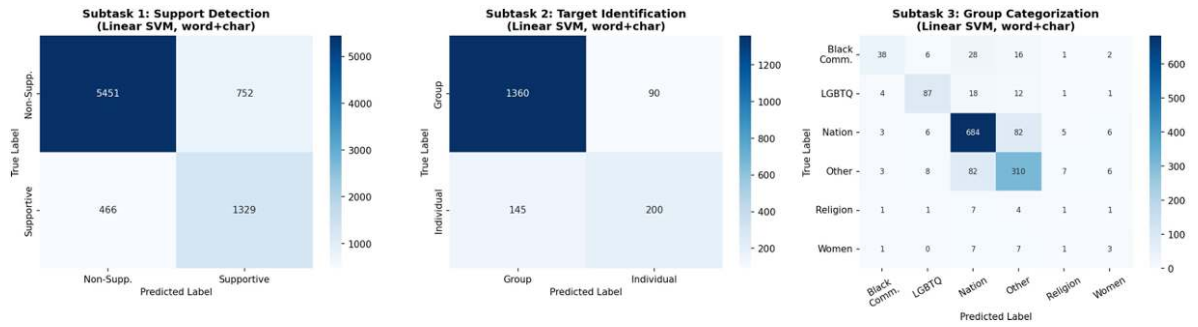

Figure 9: Cross-validation confusion matrices for S1, S2, and S3 (English, Linear SVM with word+char TF-IDF), aggregated from out-of-fold predictions on the training set—not the official test set, whose labels are not released to participants. Minority S3 classes (Religion, Women) are systematically misclassified as Nation or Other.

6.2. Cross-validation (classical baseline, Spanish)

The Spanish training set contains 2,600 comments for S1,590 supportive comments for S2, and 456 group-targeted comments for S3. Spanish models use the same representations, classifiers, and evaluation protocol as English. Table 7 reports the best configuration per subtask; the same overall ranking holds (word+char representation; SVM best on S2/S3, Logistic Regression competitive on S1), but absolute scores are consistently lower, reflecting the smaller training set (2,600 vs. 7,998 samples).

Classifier	S1	S2	S3
Logistic Regression	0.74	0.62	0.47
Linear SVM	0.73	0.71	0.58
Naive Bayes	0.69	0.40	0.18

Table 7
Best 5-fold CV macro-F1 per subtask, classical ML (Spanish, word+char TF-IDF).

6.3. Cross-validation summary and terminology

Table 8 summarizes, for reference, the best cross-validation macro-F1 achieved by the classical pipeline on each subtask. Here and elsewhere, "classical ML (best)" refers to the best-performing (TF-IDF representation, classifier) combination selected by cross-validation for that subtask—concretely, word+char TF-IDF with Linear SVM for S2 and S3, and word+char TF-IDF with Logistic Regression for S1 (Tables 4–6); it is not a single fixed classifier applied uniformly across subtasks.

We do not report a cross-validation estimate for the transformer models in this table. As described in Section 5, 3-fold cross-validation was our intended protocol for the transformer approach, but given the GPU-time budget available before the submission deadline, we prioritized completing full training and the three-seed ensemble for the official submission over a separate transformer cross-validation run. Consequently, the only transformer numbers reported anywhere in this paper are the official test-phase macro-F1 scores in Section 6.4, which come directly from the shared task's Codabench evaluation server rather than from an internal estimate; we do not include any projected or "expected" transformer score that was not obtained from an actual evaluation.

Approach	S1	S2	S3
Classical ML (best)	0.7826	0.7744	0.6921

Table 8
Best 5-fold CV macro-F1 per subtask, classical ML (English, word+char TF-IDF; see terminology note above).

6.4. Official test-phase results

Table 9 reports the per-subtask breakdown for the three classical submissions; Table 10 summarizes the overall (all-subtask) macro-F1 for all five submitted runs. Both tables report official results returned by the shared task's Codabench evaluation server on the held-out test set—as opposed to the cross-validation (CV) estimates of Table 8 and Tables 4–7, which are computed by us on the training set only. The transformer ensemble achieves the best overall test score (0.8562), marginally ahead of the single-seed transformer (0.8556) and Linear SVM (0.8440); Naive Bayes lags well behind (0.7225).

Submission	S1	S2	S3	Overall
Logistic Regression	0.8393	0.9018	0.8253	0.840
Linear SVM	0.8440	0.8849	0.8286	0.844
Naive Bayes	0.7225	0.4910	0.4504	0.722

Table 9
Official test-phase macro-F1 per subtask, classical ML submissions.

Why test-phase scores exceed the CV estimates. For every classical submission, the official per-subtask test macro-F1 (Table 9) is noticeably higher than the corresponding CV estimate (Tables 4–7): e.g., for Logistic Regression, S1 rises from 0.7826 (CV) to 0.8393 (test), S2 from 0.6904 to 0.9018, and S3 from 0.5615 to 0.8253. Two factors explain this gap, and we report them rather than adjust the numbers, since both estimates are computed correctly for what they measure. First, the CV estimate is obtained on models trained on only 4/5 of the training data per fold, whereas the submitted model is trained on the full training set; more training data directly improves a linear classifier's estimate of class boundaries, especially for S3, where the smallest classes have only 15–19 samples in total. Second, and more importantly for S2/S3, stratified 5-fold CV leaves very few minority-class examples per fold (e.g., $\sim$3 Religion samples per fold in S3); a handful of hard or noisy examples landing in the same fold can sharply depress that fold's macro-F1, and the 5-fold average inherits this variance. The official test set, being a single larger and fixed split, does not suffer from this per-fold sparsity, which is the most likely explanation for why S2/S3 show a larger CV-to-test gap than S1. We do not have access to the

official test labels, so we cannot directly verify this explanation with a per-class breakdown on the test set, and note it as an interpretation rather than a proven cause.

Figure 10 visualizes this comparison using only official, Codabench-reported numbers: per-subtask scores for the three classical submissions (left), and overall macro-F1 for all five submitted runs, including both transformer configurations (right)—we do not report a per-subtask breakdown for the transformer models, since the shared task's evaluation server only returned an overall score for those two runs. The gap between the transformer ensemble and the best classical baseline is small (1.2 points overall), suggesting that carefully engineered TF-IDF features already capture much of the lexical and morphological signal relevant to this short-text, social-media task, while the transformer's contextual representations provide a modest additional gain. The near-identical scores of the ensemble and single transformer indicate that multi-seed averaging mainly stabilizes predictions rather than adding raw accuracy.

Submitted run	Macro-F1
Transformer Ensemble	0.8562
Transformer (single)	0.8556
Linear SVM	0.8440
Logistic Regression	0.8393
Naive Bayes	0.7225

Table 10
Official test-phase macro-F1, overall (all subtasks/languages).

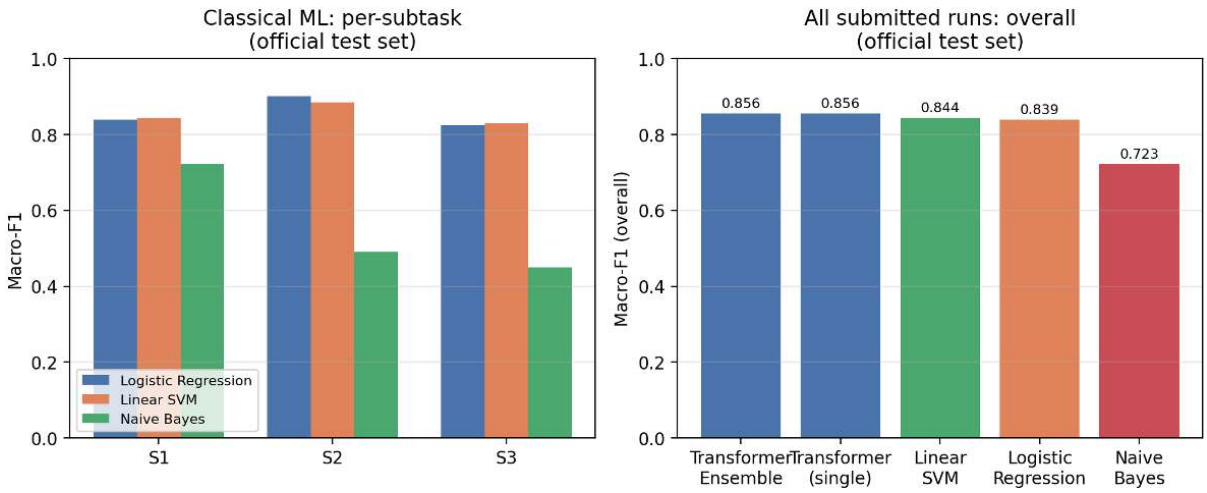

Figure 10: Left: official per-subtask test macro-F1 for the three classical submissions. Right: official overall test macro-F1 for all five submitted runs. Both panels use only values returned by the Codabench evaluation server—no cross-validation or projected values are included.

6.5. When each approach is preferable

Transformers are the better choice when comments contain subtle negation or context-dependent phrasing, when sufficient GPU budget is available, and when maximum accuracy is the priority—this matches our S1/S2 results, where contextual cues are more discriminative. Classical ML remains preferable when interpretability is required (TF-IDF weights directly show which terms drive a prediction), when only CPU resources are available, or on very small classes, where a linear model with few parameters is less prone to memorization than a fine-tuned transformer with millions of parameters—relevant for S3 categories with under 20 training examples.

Taken together, the test-phase results confirm that transformer-based fine-tuning is the strongest approach overall for this shared task, but the margin over a carefully engineered classical pipeline is

modest enough that the choice between them, in practice, should depend on the deployment constraints above rather than accuracy alone. In particular, the fact that Linear SVM outperforms Logistic Regression by an increasing margin as subtask difficulty grows (Section 6) suggests that, among classical options, the choice of classifier matters at least as much as the choice between classical and transformer architectures once a reasonable feature representation is fixed.

6.6. Why classical ML remains valuable

Despite the transformer ensemble's higher accuracy, the classical baseline retains four practical advantages that are easy to overlook when optimizing for macro-F1 alone. **Interpretability:** TF-IDF weights directly indicate which terms are most discriminative for a given prediction, which is straightforward to inspect and debug; the distributed attention weights of a 12-layer, 12-head transformer are comparatively harder to summarize into an actionable explanation. **Computational efficiency:** TF-IDF+SVM trains in minutes on CPU, whereas transformer fine-tuning requires GPU access, which raises both accessibility and environmental-cost concerns for teams without cloud budgets. **Reproducibility:** with fewer hyperparameters and less sensitivity to initialization, classical pipelines are simpler to reproduce exactly across runs and machines. **Data efficiency:** on very small classes—Religion and Women in S3 have only 15–19 training samples—a linear model with a modest number of parameters is less prone to memorizing idiosyncratic training examples than a transformer with over 100 million parameters, even under class-weighted loss.

7. Limitations and Future Work

7.1. Transformer-specific limitations

- Requires GPU hardware, limiting accessibility in resource-constrained settings.
- Hyperparameter tuning (learning rate, warmup, epochs) is more involved than for classical ML, and results carry higher run-to-run variance from random initialization.
- Fine-tuning on very small classes (e.g., Religion: 15 samples) risks overfitting/memorization rather than generalization.

7.2. Classical ML-specific limitations

- TF-IDF is a static, context-free representation: it cannot distinguish "I support this" from "I don't support this", since both share the same n-grams.
- Cannot resolve word-sense ambiguity or long-range dependencies.
- Naive Bayes' independence assumption fails catastrophically under multiclass imbalance (Section 6).

7.3. Shared limitations

The main limitation shared by both systems is class imbalance, especially on S3 (52.4:1), where minority classes remain poorly recognized despite class weighting and margin maximization. The hierarchical cascade also propagates errors: an S1 mistake cannot be recovered by S2/S3. Each comment is classified in isolation, without conversational context, and identical hyperparameters are used for both languages despite their morphological differences. We do not report a cross-validation estimate for the transformer models (Section 6.3), which limits direct like-for-like comparison with the classical CV results, and we did not run a formal ablation isolating the effect of individual preprocessing steps (Section 4). Finally, because official test labels are not released, our cross-validation estimates are computed only on the training set.

7.4. Future work

Promising directions include: (1) hybrid systems combining transformer embeddings with classical classifiers such as SVM via weighted voting or stacking, given the strong relative performance of classical methods observed here; (2) advanced imbalance handling for S3, such as SMOTE-based oversampling, focal loss, or adversarial debiasing; (3) hierarchical multi-task learning, jointly predicting S1/S2/S3 with task-aware loss weighting to avoid error propagation from the current cascade; (4) conversational context, incorporating preceding comments to disambiguate short, isolated text; (5) language-specific fine-tuning, optimizing hyperparameters separately for English and Spanish; and (6) applying interpretability tools (e.g., LIME, SHAP) to understand which words and phrases drive transformer predictions, complementing the more directly interpretable TF-IDF weights of the classical baseline; and (7) a dedicated cross-validation study for the transformer ensemble itself—as noted in Section 6.3, we were unable to complete a transformer CV run before the submission deadline, and rely solely on the official test-phase evaluation for the transformer models; a future 3-fold (or full 5-fold) CV run would let us compare CV-to-test gaps for the transformer approach directly against the classical results in Section 6; and (8) a controlled preprocessing ablation, systematically toggling URL/mention removal, lowercasing, and stopword removal on and off to quantify their individual contribution to macro-F1, rather than relying on the qualitative reasoning of Section 4.

8. Conclusions

We submitted two complementary systems to the Social Support Detection Shared Task at IberLEF 2026: a TF-IDF+linear-classifier baseline and a fine-tuned RoBERTa/XLM-RoBERTa ensemble, sharing the same hierarchical S1→S2→S3 cascade. Key findings:

1. The transformer ensemble achieved the best overall test macro-F1 (0.8562), closely followed by the single-seed transformer (0.8556) and Linear SVM (0.8440); Naive Bayes lagged well behind (0.7225).
2. The combined word+char TF-IDF representation consistently outperformed either representation alone, across all subtasks and classifiers.
3. Linear SVM's margin-based objective handled imbalance substantially better than Logistic Regression and Naive Bayes as task difficulty increased from S1 to S3, while Naive Bayes was not competitive on the multiclass subtask.
4. S3 was markedly harder than S1/S2 for every model, driven by extreme class imbalance (52.4:1 in English), multiclass complexity, and a much smaller training set.
5. The gap between the transformer ensemble and the best classical baseline was small (1.2 points overall), and the ensemble barely outperformed the single-seed transformer—suggesting that multi-seed averaging mainly stabilizes predictions, and that well-engineered TF-IDF already captures much of the relevant lexical and morphological signal for this short-text task.
6. Spanish results consistently lagged English across both approaches, reflecting its smaller training set (2,600 vs. 7,998 samples).

Overall, the results illustrate a contemporary picture of NLP practice: transformer fine-tuning delivers the best raw accuracy through contextualized representations and transfer learning, but well-engineered classical pipelines remain highly competitive, interpretable, and computationally cheap baselines for short-text, imbalanced, bilingual social-media classification—and a natural complement for future hybrid systems.

Finally, it is worth emphasizing what the results do not show: they do not indicate that classical TF-IDF pipelines are a universal substitute for transformers, nor that transformer ensembling is unnecessary. Rather, for this particular combination of short YouTube comments, a moderately sized bilingual training set, and macro-F1 as the ranking metric, the accuracy ceiling reachable by lexical and morphological features alone happens to sit close to what contextual fine-tuning achieves. On larger training sets, longer documents, or tasks that hinge more heavily on discourse-level or figurative language (sarcasm,

irony), the gap between the two families of methods would likely be considerably larger, as reported elsewhere in the social-media classification literature.

Acknowledgments

The authors thank the IberLEF 2026 organizers and the Universidad Panamericana for supporting this work. We acknowledge the authors of the Social Support Detection datasets for making this shared task possible.

Declaration on Generative AI

The authors have not employed any Generative AI tools in the design of experiments or analysis of results. All code development, model training, and result interpretation were performed by the authors.

References

[1] Zahra Ahani, Moein Shahiki Tash, Fazlourrahman Balouchzahi, Luis Ramos, Grigori Sidorov, and Alexander Gelbukh. Social support detection from social media texts. PLOS ONE, 2026.

[2] Alba Bonet-Jover, José Ángel González-Barba, and Luis Chiruzzo. Overview of iberlef 2026: Natural language processing challenges for spanish and other iberian languages. In Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2026), co-located with the 42nd Conference of the Spanish Society for Natural Language Processing (SEPLN 2026). CEUR-WS.org, 2026. Forthcoming.

[3] Olga Kolesnikova, Moein Shahiki Tash, Zahra Ahani, Aayush Agrawal, Raul Monroy, and Grigori Sidorov. Advanced machine learning techniques for social support detection on social media. Heliyon, 11(10), 2025.

[4] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12:2825–2830, 2011.

[5] Moein Shahiki-Tash, Zahra Ahani, Luis Ramos, Olga Kolesnikova, et al. Linguistic analysis in different types of support using liwc for spanish and english online communities. Research Square, 2026.

[6] Moein Shahiki Tash, Luis Ramos, Zahra Ahani, Ari Yair Barrera-Animas, Fazlourrahman Balouchzahi, Olga Kolesnikova, Grigori Sidorov, Alexander Gelbukh, Raul Monroy, and Francisco Vargas. Overview of ssd-2026 at iberlef 2026: Social support detection. In Proceedings ofthe Iberian Languages Evaluation Forum (IberLEF 2026), co-located with the 42nd Conference of the Spanish Society for Natural Language Processing (SEPLN 2026). CEUR-WS.org, 2026. Forthcoming.

[7] Moein Shahiki Tash, Luis Ramos, Zahra Ahani, Raul Monroy, Hiram Calvo, and Grigori Sidorov. Online social support detection in spanish social media texts. arXiv preprint arXiv:2502.09640, 2025.