The Resistant Word at StanceNakba Shared Task: A Topic-Aware Model for Cross-Topic Stance Detection

Mohammed Bahgat[1], **Doaa Salah**[2], **Sarah Yassine**[3]
[1] SUT Egypt, Cairo, Egypt
[2] Cairo University, Cairo, Egypt
[3] Lebanese University, Beirut, Lebanon
mohamed.bahgat@sut.edu.eg
doaa.ibrahim00@eng-st.cu.edu.eg
sarah.yassine.2@st.ul.edu.lb

Abstract

Cross-topic stance detection in Arabic is the task of identifying whether a text expresses a *pro*, *against*, or *neutral* position toward a given issue, and it is particularly challenging under topic shifts and class imbalance. In Subtask B of the StanceNakba 2026 shared task on Arabic cross-topic stance detection, we are given a Levantine Arabic sentence and one of two topics: "Normalization with Israel" or "Refugee/Immigrant Presence in Jordan," and we must classify the expressed stance. A central difficulty is the systematic failure of standard fine-tuning to recognize the minority *neutral* class, driven by majority-class dominance in cross-entropy training and accuracy-based checkpoint selection. To address this, we combine random oversampling with class-weighted cross-entropy loss, and we build an ensemble of four Arabic pre-trained transformers MARBERT, AraBERT Large, XLM-RoBERTa Base, and CAMeL-BERT Mix each trained using Stratified 5-Fold cross-validation. Our final system achieves a macro-F1 of **0.9777** and an accuracy of **97.79%** on the evaluation set.

Keywords: Arabic NLP, stance detection, Levantine dialect, class imbalance, weighted loss, ensemble, transformer

1. Introduction

Stance detection identifies whether a text is *pro*, *against*, or *neutral/none* toward a target. For Arabic social media, this remains difficult because of dialectal variation and implicit or sarcastic language, which often blur stance and sentiment. Recent shared tasks such as StanceEval 2024 reflect growing interest in Arabic stance detection while highlighting remaining challenges for robust modeling (Alturayeif et al., 2024).

We present our submission to Subtask B of the StanceNakba shared task on Arabic cross topic stance detection (Aldous et al., 2026). Given a Levantine Arabic sentence and one of two targets, *Normalization with Israel* or *Refugee/Immigrant Presence in Jordan*, the system predicts *pro*, *against*, or *neutral*. The main challenge is reliable detection of the minority *neutral* class, which is often weakened by standard cross entropy training and accuracy based checkpointing. We address this with class weighted loss, random oversampling, and an ensemble of four Arabic pre trained transformers, MARBERT, AraBERT Large, XLM RoBERTa Base, and CAMeL BERT Mix, trained with Stratified 5 Fold cross validation and combined by probability averaging. Our official submission ranked fourth on the Subtask B leaderboard with 0.8562 macro F1 and 0.8564 accuracy. To support reproducibility, we release our code and experiment configuration in a public GitHub repository.

Our contributions are: (1) a Levantine Arabic cross topic stance system for two politically sensitive targets; (2) an imbalance aware training recipe that improves *neutral* detection; and (3) an ensemble framework combining model diversity, Stratified 5 Fold training, and consistent preprocessing.

2. Background & Related Work

Stance detection predicts whether a text expresses a *pro*, *against*, or *neutral* position toward a target. Recent work has moved from feature based models to neural and transformer approaches, while cross topic stance detection studies whether models trained on seen targets generalize to unseen ones (Khiabani and Zubiaga, 2024; Reuver et al., 2021). For Arabic, progress is still challenged by morphology, dialect variation, and limited annotated resources. Recent efforts such as MARASTA, ArabicStanceX, and StanceEval 2024 have expanded Arabic stance datasets and evaluation settings (Charfi et al., 2024; Alkhathlan et al., 2025; Alturayeif et al., 2024). Our work builds on this line by focusing on Levantine Arabic cross topic stance detection in a political setting.

Proceedings of the 2nd International Workshop on Nakba Narratives as Language Resources (Nakba-NLP 2026) @ LREC 2026, pages 182–186

3. System Overview

3.1. Data Description

Subtask B uses a topic specific subset of the MARASTA corpus, a multi dialectal Arabic cross domain stance corpus (Charfi et al., 2024). It contains 1,024 annotated instances split into training, evaluation, and test sets. Each instance is a Levantine Arabic sentence paired with one of two topics: *Normalization with Israel* (*al tatbi' ma' Isra'il*) and *Refugee/Immigrant Presence in Jordan* (*Wujud al Laji'in wal Muhajirin fi al Urdun*). The task is to predict one of three labels: *pro*, *against*, or *neutral*.

Example instances include a *pro* case for *Normalization with Israel* (*al tatbi' ma'a Isra'il sayakun khayran in sha' Allah*), an *against* case for the same topic (*ma fihi tatbi' ma'akum*), and a *neutral* case for *Refugee/Immigrant Presence in Jordan* describing limited work rights for Syrian refugees in Jordan since 2016.

3.1.1. Training Set

The training set contains 843 instances, with a relatively balanced label distribution: 298 *against*, 286 *pro*, and 259 *neutral*. It covers both topics, with 404 instances for *Normalization with Israel* and 439 for *Refugee/Immigrant Presence in Jordan*. For *Normalization with Israel*, the label distribution is 139 *against*, 145 *neutral*, and 120 *pro*; for *Refugee/Immigrant Presence in Jordan*, it is 159 *against*, 114 *neutral*, and 166 *pro*.

3.1.2. Test Set

The test set also contains 181 instances. It is released without labels and used for the final evaluation. The dataset is designed for topic-aware modeling: each sentence is provided with its topic, allowing models to condition predictions on the target.

3.1.3. Evaluation Set

The evaluation set contains 181 instances: 63 *against*, 62 *pro*, and 56 *neutral*. It includes 87 instances for *Normalization with Israel* and 94 for *Refugee/Immigrant Presence in Jordan*. The former is distributed as 29/32/26 (*against*/*neutral*/*pro*), and the latter as 34/24/36.

3.2. Evaluation Metrics

We report Macro-F1 (primary), Accuracy, Precision, and Recall. Macro-F1 averages per-class F1 scores, making it appropriate when performance varies across stance labels and minority classes should be weighted equally (Khiabani and Zubiaga, 2024; Alturayeif et al., 2024). Accuracy measures overall correctness but can be less informative under label imbalance (Alturayeif et al., 2024; Khiabani and Zubiaga, 2024). Precision and Recall quantify, respectively, false-positive control and true-positive coverage per class (Khiabani and Zubiaga, 2024). Here TP, FP, and FN denote true positives, false positives, and false negatives for each class in $\mathcal{C} = \{\text{pro}, \text{against}, \text{neutral}\}$, providing a balanced view of performance on dialectal Arabic stance detection (Alturayeif et al., 2024; Lichouri et al., 2024).

3.3. Methodology

Our system is built on a pipeline of four components applied in sequence: (1) Arabic text preprocessing, (2) class balancing via oversampling, (3) fine-tuning of multiple Arabic transformer models with weighted cross-entropy loss, and (4) probability-averaging ensemble with K-Fold cross-validation. Figure 1 illustrates the complete pipeline.

Given a topic sentence pair, the system first normalises both strings and encodes them as a single input sequence, with the topic as the first segment and the sentence as the second, separated by `[SEP]`. The encoded input is then passed to each of the four transformer models. For every model, Stratified 5-Fold training produces five probability vectors over {*pro*, *against*, *neutral*}. We average these probabilities across folds to obtain one prediction per model, then average again across the four models. The final stance label is the class with the highest averaged probability. For instance, if the final ensemble outputs $(0.08, 0.81, 0.11)$ for (*pro*, *against*, *neutral*), the predicted label is *against*.

3.4. Text Preprocessing

We apply an Arabic-specific normalisation pipeline to both sentences and topic strings prior to tokenisation to reduce social-media noise (inconsistent orthography, diacritics, and platform artefacts) (Charfi et al., 2024). The pipeline removes diacritics and tatweel, standardises common letter variants (alef forms, alef maqsura→ya', ta' marbuta→ha'), removes URLs and mentions, strips the # symbol while keeping hashtag text, and normalises whitespace.

After normalisation, each instance is encoded as a topic–sentence pair, with the topic as the first sequence and the sentence as the second, separated by `[SEP]`. This formatting enables topic-aware attention and supports cross-topic stance classification.

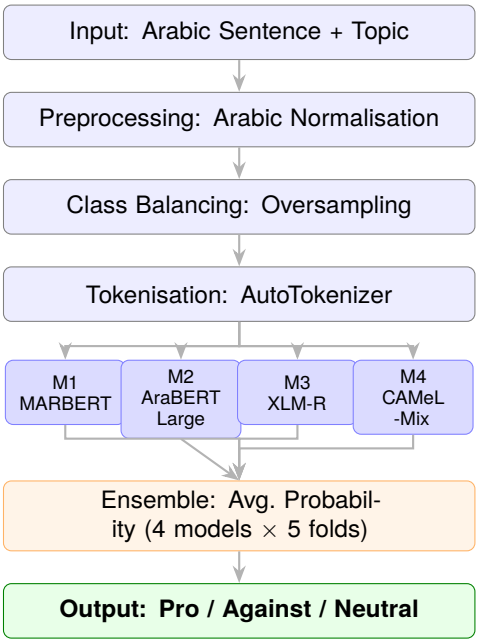

Figure 1: Full system pipeline from raw Arabic input to final stance label. All four models are fine-tuned with K-Fold and Weighted Cross-Entropy Loss; their softmax probabilities are averaged for the final prediction.

3.5. Class Balancing via Oversampling

Although the overall training labels are relatively balanced, we observe topic-level skew (Table **??**) that contributes to under-prediction of the minority *neutral* class. To counter this, we apply random oversampling on the training split using `sklearn.utils.resample`: we upsample each class with replacement (seed = 42) to match the largest class size, then shuffle the resulting balanced set. Oversampling is applied **only** to training; evaluation and test splits remain unchanged to preserve unbiased estimation.

3.6. Pre-Trained Models

We fine-tune four Arabic transformer models chosen for their complementary pre-training data and their demonstrated strength on Arabic NLP tasks (Antoun et al., 2020; Inoue et al., 2021):

- **M1 – MARBERT** (`UBC-NLP/MARBERT`): trained on one billion Arabic tweets spanning multiple dialects; particularly well-suited to dialectal and social media text (Abdul-Mageed et al., 2021).
- **M2 – AraBERT Large** (`aubmindlab/bert-large-arabertv02`): large variant of AraBERT v2, pre-trained on a diverse and large-scale Arabic corpus covering both MSA and web text (Antoun et al., 2020).
- **M3 – XLM-RoBERTa Base** (`FacebookAI/xlm-roberta-base`): a multilingual masked language model pre-trained on 100 languages, offering strong cross-lingual transfer and robustness to out-of-vocabulary Levantine terms.
- **M4 – CAMeL-BERT Mix** (`CAMeL-Lab/bert-base-arabic-camelbert-mix`): trained jointly on MSA and dialectal Arabic text, making it particularly suitable for the mixed-register nature of Levantine social media (Inoue et al., 2021).

The diversity in pre-training corpora of dialectal Twitter data, large-scale Arabic web text, multilingual data, and mixed-register Arabic is intentional. Each model captures different linguistic aspects of the Levantine dialect, and their ensemble corrects individual model weaknesses.

3.7. Weighted Cross-Entropy Loss

Motivation. Standard cross-entropy weights all samples equally, so under class imbalance majority labels dominate training; in our baseline this suppressed learning for the minority *neutral* class, causing systematic under-prediction.

Class-weighted loss. To counter this bias, we apply class-weighted cross-entropy, assigning each class c a weight inversely proportional to its (post-oversampling) frequency in the training set:

$$w_c = \frac{1/|D_c|}{\sum_{c'} 1/|D_{c'}|} \cdot |\mathcal{C}| \tag{1}$$

This normalisation ensures $\sum_c w_c = |\mathcal{C}| = 3$. The weighted loss is:

$$\mathcal{L}_w = -\sum_{c \in \mathcal{C}} w_c \cdot y_c \log p_c \tag{2}$$

where y_c is the ground-truth indicator and p_c is the predicted probability for class c.

Implementation. We implement a custom `WeightedLossTrainer` by subclassing the Hugging Face `Trainer` and overriding `compute_loss`. The weighted-loss implementation is shown in Appendix A.

3.8. K-Fold Cross-Validation Ensemble

To better exploit the limited training data (843 samples) and reduce prediction variance, we train each of the four models using Stratified 5-Fold cross-validation. In each fold, a model is fine-tuned on 80% of the training set and produces softmax probability predictions on the evaluation and test sets. We then average probabilities across folds for each

Hyperparameter	Value
Epochs per fold	5
Batch size	16
Learning rate	2×10^{-5}
Max sequence length	128 tokens
Warmup ratio	0.1
Weight decay	0.01
K-Fold splits	5 (stratified)
Precision	bfloat16
Loss function	Weighted CrossEntropy
Best checkpoint metric	macro-F1
Random seed	42

Table 1: Training configuration, identical for all four models.

model and across the $M = 4$ models to obtain the final prediction with $K = 5$. We use probability averaging rather than majority voting to leverage model confidence information and the full predictive distribution.

3.9. Training Configuration

Table 1 summarises the hyperparameters shared across all four models.

A key departure from the baseline is using **macro-F1** rather than accuracy as the criterion for saving the best checkpoint. Since accuracy rewards majority-class predictions, a model that rarely predicts *neutral* can still achieve high accuracy. Macro-F1 treats all classes equally and directly penalises failure to detect minority classes, producing checkpoints that generalise better across the full label set.

Experiments were run in a Kaggle GPU environment (https://www.kaggle.com/code) using PyTorch v2.8.0+cu126 (https://pytorch.org/), Transformers v4.57.1 (https://huggingface.co/docs/transformers/index), pandas v2.2.2 (https://pandas.pydata.org/), NumPy v2.0.2 (https://numpy.org/), and scikit-learn v1.6.1 (https://scikit-learn.org/stable/). Full scripts and configuration are available in our public repository.

4. Experimental Results

4.1. Model Comparison and Final System

Table 2 compares all evaluated systems on the 181-sample evaluation set. The upper block reports nine individual Arabic transformer models, each trained with Arabic preprocessing, class-weighted loss, random oversampling, and macro-F1 checkpoint selection. The highlighted row is our final ensemble, which combines four models with Stratified 5-Fold cross-validation and probability averaging.

Model / System	Acc.	Prec.	Rec.	F1
Individual fine-tuned models				
bert-base-arabic	0.7403	0.7403	0.7410	0.7401
camelbert-da	0.7624	0.7627	0.7629	0.7627
bert-base-arabertv02	0.7679	0.7733	0.7655	0.7677
MARBERT	0.7734	0.7737	0.7753	0.7735
roberta-eng-ara-128k	0.7790	0.7831	0.7810	0.7765
MARBERTv2	0.7900	0.7899	0.7900	0.7890
ARBERTv2	0.7950	0.7970	0.7950	0.7950
albert-xlarge-arabic	0.7955	0.7979	0.7968	0.7950
camelbert-msa	0.8011	0.8020	0.8034	0.7997
Ensemble system (this work)				
Ens. (4M×5F)	**0.9779**	**0.9778**	**0.9788**	**0.9777**

Table 2: Results on the evaluation set (181 samples), sorted by macro-F1. The highlighted row is the final submitted system.

Among single models, `camelbert-msa` performs best with a macro-F1 of 0.7997. Our ensemble improves substantially over the strongest single model, reaching 0.9777 macro-F1 on the evaluation set.

Official shared-task evaluation (test set). On the hidden official test set, our submission ranked **4th** on Subtask B with macro-F1 = 0.8562, accuracy = 0.8564, macro-precision = 0.8617, and macro-recall = 0.8566.

4.2. Generalization Gap and Error Analysis

The drop from local to official performance (0.9777 vs. 0.8562 macro-F1) suggests optimistic local estimates and some overfitting, likely because the evaluation split comes from the same collection as training, while the hidden test set contains harder lexical and discourse variation.

Detailed diagnostics are available only on the evaluation set. There, *pro* and *neutral* achieve perfect recall, and all four errors come from *against* examples, mostly confused with *neutral*. Thus, the main remaining difficulty is distinguishing implicit or ironic opposition from neutral content under distribution shift.

5. Conclusion

We presented our system for Subtask B of the StanceNakba 2026 shared task on Arabic cross-topic stance detection. To improve minority *neutral* detection, we combined class-weighted cross-entropy, macro-F1 checkpoint selection, random oversampling, and an ensemble of four Arabic transformers trained with Stratified 5-Fold cross-validation and probability averaging. The final system achieved 0.9777 macro-F1 on the evaluation set and 0.8562 macro-F1 on the official hidden test set.

6. Limitations

The gap between local and official performance suggests limited generalization and possible overfitting to the evaluation distribution. In addition, random oversampling does not add new linguistic diversity, the ensemble is computationally expensive, and the dataset covers only two politically sensitive topics in Levantine Arabic, which limits broader generalization. Future work will explore dialect-aware augmentation and instruction-tuned Arabic LLMs for more efficient and robust stance modeling (Alghaslan and Almutairy, 2024).

Acknowledgements

The authors thank the StanceNakba 2026 shared task organizers for providing the dataset and evaluation infrastructure.

7. Bibliographical References

Muhammad Abdul-Mageed, AbdelRahim Elmadany, and El Moatez Billah Nagoudi. 2021. ARBERT and MARBERT: Deep bidirectional transformers for Arabic. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP 2021)*, pages 7088–7105, Online. Association for Computational Linguistics.

Kholoud Khalil Aldous, Md Rafiul Biswas, Mabrouka Bessghaier, Shimaa Ibrahim, Kais Attia, and Wajdi Zaghouani. 2026. StanceNakba shared task: Actor and topic-aware stance detection in public discourse. In *Proceedings of the 15th International Conference on Language Resources and Evaluation (LREC'26)*, Palma, Spain.

Mamoun Alghaslan and Khaled Almutairy. 2024. MGKM at StanceEval2024: Fine-tuning large language models for Arabic stance detection. In *Proceedings of the Second Arabic Natural Language Processing Conference (ArabicNLP 2024)*, pages 816–822, Bangkok, Thailand. Association for Computational Linguistics.

A. Alkhathlan, F. Alahmadi, F. Kateb, and H. Al-Khalifa. 2025. Constructing and evaluating ArabicStanceX: A social media dataset for arabic stance detection. *Frontiers in Artificial Intelligence*, 8:1615800.

Nora Alturayeif, Hamzah Luqman, Zaid Alyafeai, and Asma Yamani. 2024. StanceEval 2024: The first Arabic stance detection shared task. In *Proceedings of the Second Arabic Natural Language Processing Conference (ArabicNLP 2024)*, Bangkok, Thailand. Association for Computational Linguistics.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools (OSACT4)*, pages 9–15, Marseille, France. European Language Resources Association (ELRA).

Anis Charfi, Marwa Bessghaier, Ali Atalla, Rand Akasheh, Sara Al-Emadi, and Wajdi Zaghouani. 2024. Stance detection in Arabic with a multi-dialectal cross-domain stance corpus. *Social Network Analysis and Mining*, 14(1).

Go Inoue, Bashar Alhafni, Nurpeiis Baimukan, Houda Bouamor, and Nizar Habash. 2021. The interplay of variant, size, and task type in Arabic pre-trained models. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop (WANLP 2021)*, pages 92–104, Kyiv, Ukraine (Online). Association for Computational Linguistics.

Parisa Jamadi Khiabani and Arkaitz Zubiaga. 2024. Cross-target stance detection: A survey of techniques, datasets, and challenges. *arXiv preprint arXiv:2409.13594*.

Mohamed Lichouri, Khaled Lounnas, Khelil Rafik Ouaras, Mohamed Abi, and Anis Guechtouli. 2024. dzStance at StanceEval2024: Arabic stance detection based on sentence transformers. In *Proceedings of the Second Arabic Natural Language Processing Conference (ArabicNLP 2024)*, Bangkok, Thailand. Association for Computational Linguistics.

Myrthe Reuver, Suzan Verberne, Roser Morante, and Antske Fokkens. 2021. Is stance detection topic independent and cross topic generalizable? a reproduction study. In *Proceedings of the 8th Workshop on Argument Mining*, pages 46–56, Punta Cana, Dominican Republic. Association for Computational Linguistics.

A. Weighted-Loss Implementation

```
weights = class_weights_tensor.to(device)
loss_fct = nn.CrossEntropyLoss(weight=weights)
loss = loss_fct(logits.view(-1,
num_labels), labels.view(-1))
```