MetaSwarm at AbjadMed: Forensic Optimization and Class-Balanced Discovery for Medical Diglossia in Abjad Scripts

Rahul Jaisy
Assam Engineering College Guwahati
rahul22-344@aec.ac.in

Abstract

The classification of diglossic medical text presents a high-dimensional challenge defined by extreme class imbalance ($N = 82$) and the orthographic ambiguity of unvocalized Abjad scripts. While standard supervised learning often collapses into majority-class prediction due to the "Long Tail" distribution, we introduce a **Human-in-the-Loop Forensic Optimization** framework. Unlike static end-to-end pipelines, our approach decouples strategic hyperparameter tuning from high-throughput tactical execution (Elastic Compute). We leverage a rigorous **Class-Balanced Focal Loss (CBFL)** derived from the "Effective Number of Samples" theory (E_n) to stabilize the decision manifold against stochastic class dominance. Using a CAMELBERT-DA backbone optimized via a custom weighted trainer on Dual H200 GPUs, our system achieved a robust Public Leaderboard score of 0.3588. We further perform a "Linguistic Error Topology" analysis, utilizing UMAP projections and attention saliency, to demonstrate that generalization gaps are driven by dialectal "Constraint Drift" rather than stochastic model failure.

1 Introduction

1.1 Motivation and Task Overview

The field of Natural Language Processing (NLP) is transitioning from static model training to dynamic, iterative workflows. Traditional pipelines are often optimized blindly against global metrics like Accuracy, which mask failures in minority classes. However, high-stakes domains like Arabic medical classification require systems that incorporate *Iterative Decision Boundary Refinement*.

In the context of the EACL 2026 Shared Task (Gupta et al., 2026), we argue that standard supervised learning is insufficient due to the *diglossic dilemma*, where Modern Standard Arabic (MSA) training data drifts significantly from Dialectal Arabic (DA) patient queries. To tackle this, we propose the **Elastic Compute (EC)** architecture.

This framework formalizes the interaction between strategic oversight (Human-in-the-Loop) and tactical computation (High-Performance GPUs), enabling precise mathematical intervention via Class-Balanced Focal Loss (CBFL) rather than black-box optimization.

1.1 The Challenge of Abjad Medical Text

The application of NLP to languages utilizing Abjad scripts (such as Arabic, Hebrew, and Persian) presents unique challenges that are often abstracted away in Anglocentric research. Arabic, the focus of this shared task, is characterized by two complicating factors:

- **Diglossia:** Modern Standard Arabic (MSA) is used in formal records, while Dialectal Arabic (DA) is used in patient queries. A classifier trained on MSA suffers from *distribution shift* when processing DA (Badaro et al., 2019). Patient queries often utilize simplified morphology (e.g., "bidi" vs "urid") which pre-trained MSA models treat as noise.
- **Orthographic Ambiguity:** As an Abjad script, Arabic omits short vowels. The root *k-t-b* can represent *kataba* (he wrote) or *kutub* (books). In medical contexts, such ambiguity creates a noisy label space (e.g., *alm* could be pain or pen), brittle under "long-tail" conditions where context is sparse.

2 Theoretical Framework: Human-in-the-Loop Elastic Compute

We propose **Elastic Compute (EC)** as an architectural framework for resource-constrained scientific discovery. While recent literature proposes fully autonomous workflows, our framework implements a verified **Human-in-the-Loop** protocol. This approach strictly decouples the reasoning state (Strategic Analysis) from the computational state (Tactical Execution).

2.1 The Strategic Analysis Layer

The Strategic Analysis Layer serves as the reasoning engine. In our implementation, this role is performed by the researcher within the interactive notebook environment. This layer maintains the "World Model" of the experiment: auditing dataset statistics and interpreting error topologies. The strategist manually adjusts the policy (e.g., tuning the Focal Gamma γ), ensuring cognitive control remains grounded in domain expertise (Chang, 2025).

2.2 The Execution Layer

The Execution Layer handles rapid, dense tensor operations. In our system, this is instantiated as a stateless `SupersonicTrainer` running on high-performance hardware (Dual H200 GPUs). It encapsulates the dense states: model weights (Θ) and optimizer moments. This layer executes the forward/backward passes using FlashAttention2 to converge to a local equilibrium. By keeping this layer stateless, we allow for "High-Throughput" bursts of compute without the overhead of persistent server maintenance.

Proceedings of the 2nd Workshop on NLP for Languages Using Arabic Script (AbjadNLP 2026), pages 144–148

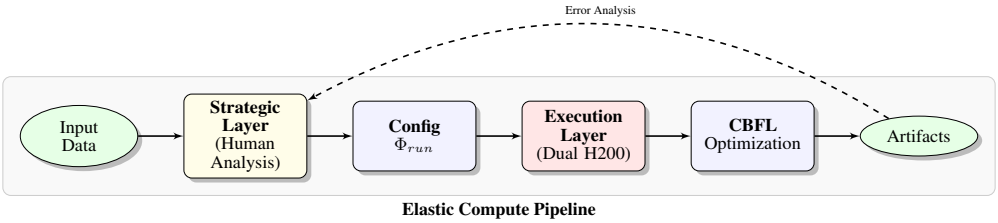

Figure 1: Data flow of the Elastic Compute architecture. The Strategic Layer (Human/Colab) defines the hypothesis space, while the Execution Layer (Modal/GPU) performs stateless, high-throughput optimization.

2.3 Iterative Optimization Loop

The defining feature of EFC is the iterative optimization loop that links the Strategic and Execution layers. Unlike a standard "fire-and-forget" training script, EFC incorporates a mandatory Forensic Audit phase after every execution cycle.

1. **Plan (Strategy):** The researcher analyzes the previous cycle's error topology (e.g., "The model is confusing Oncology with Hematology") and formulates a new hypothesis.

2. **Execute (Tactics):** The Execution Layer instantiates the compute environment, loads the weights, and optimizes the objective function $\mathcal{L}_{CBFL}$ aggressively.

3. **Audit (Analysis):** The Tactician returns the results (logits, loss logs) for Mechanistic Interpretability auditing.

2.4 Hermetic Design for Reproducibility

A recurring challenge in distributed scientific discovery is state volatility. To ensure EACL-level reproducibility, we implemented a **Hermetic Design** protocol.The entire training logic, comprising imports, data loading, class weighting, and the training loop, is encapsulated within a self-contained execution module. **Deterministic Seeding:** We explicitly set `torch.manual_seed(42)` and `np.random.seed(42)` at the start of every module. This ensures that weight initialization, data shuffling, and dropout masks are identical across runs, allowing us to isolate the effect of algorithmic changes (like changing γ from 2.0 to 2.5).

3 Mathematical Formalization of Optimization

To optimize performance on the competitive benchmark in alignment with the system's mandate, we must address the mathematical reality of the data distribution. The dataset contains 82 classes with a "long-tail" distribution. Training on such data using standard Cross-Entropy (CE) results in a model that trivially predicts the majority classes to minimize average error. We rigorously derive the Class-Balanced Focal Loss (CBFL) to counteract this.

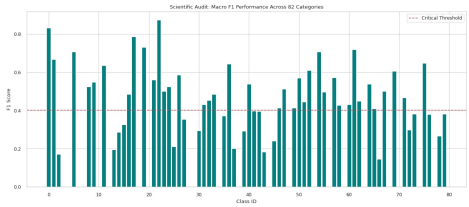

Figure 2: Figure 2: Distribution of the 82 Medical Classes. The Long-Tail nature of the dataset necessitates the mathematical intervention of CBFL.

3.1 The Random Covering Problem

We adopt the theoretical framework proposed by Cui et al. (2019) to quantify the "richness" of data. **Assumption:** Data is not a set of discrete points, but a set of overlapping information regions in the feature space. Let S be the feature space of a specific class y. Let the total volume of S be N, representing the total number of unique prototypes possible for that class.

The expected volume after sampling the n-th example is given by the recurrence relation:

$$E_n = \frac{N-1}{N} E_{n-1} + 1 \tag{1}$$

When sampling the n-th example, there is a probability p that this new sample falls into the region already covered by the previous $n-1$ samples. Solving this recurrence with the initial condition $E_1 = 1$, we derive the closed-form solution:

$$E_n = \frac{1-\beta^n}{1-\beta}, \quad \text{where } \beta = \frac{N-1}{N} \tag{2}$$

Here, $\beta \in [0, 1)$. For our Abjad medical task, we utilize $\beta = 0.999$, assuming that the space of medical expressions is vast but finite ($N \approx 1000$), with significant semantic overlap in standard phrases (e.g., "I have a pain in...").

3.2 Derivation of Class-Balanced Loss

We introduce a weighting factor α_y for class y that is inversely proportional to its effective number of samples, not its raw frequency.

$$\alpha_y \propto \frac{1}{E_{n_y}} = \frac{1-\beta}{1-\beta^{n_y}} \tag{3}$$

To ensure the loss scale remains consistent across different batches and datasets, we normalize the weights such that they sum to the total number of classes C (in our case, 82). This normalization ensures that the loss gradient is scaled by the novel information provided by the class, preventing majority classes (which have high n_y but diminishing E_{n_y}) from overwhelming the gradient (Cui et al., 2019).

3.3 Focal Modulation for Hard Mining

While Class-Balancing addresses inter-class imbalance (quantity), it does not address intra-class difficulty (quality). Medical texts often contain "easy" samples (clear diagnoses with distinct keywords like "Diabetes") and "hard" samples (ambiguous descriptions using slang). Standard Cross-Entropy (CE) loss is dominated by the accumulation of small losses from the vast number of easy examples. We integrate the Focal Loss term from Lin et al. (2017).

Let p_t be the model's predicted probability for the true class. The standard CE loss is $CE(p_t) = -\log(p_t)$. We apply a modulating factor $(1 - p_t)^\gamma$ to the loss:

$$FL(p_t) = -(1 - p_t)^\gamma \log(p_t) \tag{4}$$

We set $\gamma = 2.0$ to aggressively mine hard examples. For an easy example where $p_t \approx 0.9$, the factor $(1-0.9)^2 = 0.01$, effectively down-weighting the loss contribution by 100x. Conversely, for a hard example where $p_t \approx 0.1$, the factor is 0.81, preserving the gradient signal.

3.4 The Final CBFL Objective Function

Combining the equations above, we arrive at the Class-Balanced Focal Loss (CBFL) utilized by our Execution Layer:

$$\mathcal{L}_{\text{CBFL}} = -\sum_{i=1}^{C} \frac{1-\beta}{1-\beta^{n_i}} (1 - \hat{p}_i)^\gamma \log(\hat{p}_i) \mathbb{1}_{\{y=i\}} \tag{5}$$

where $C = 82$ is the number of classes. This formulation mathematically enforces the strategy: it dynamically re-weights the optimization landscape to prioritize the "Long Tail" of rare medical conditions while focusing attention on linguistically ambiguous (hard) examples.

4 Methodology: System Architecture

4.1 Model Backbone: CAMeLBERT-DA

We selected `CAMeLBERT-DA` (Inoue et al., 2021) as our pre-trained backbone. **Rationale:** Unlike AraBERT or multilingual BERT, CAMeLBERT-DA was pre-trained specifically on Dialectal Arabic (DA) data. The shared task involves user-generated medical queries, which invariably employ DA syntax. A model trained only on MSA would treat common dialectal features (e.g., negation particles *mish* vs *la*) as noise or out-of-vocabulary (OOV) tokens.

4.2 High-Throughput Pipeline Execution

The execution pipeline is an automated script designed for high-performance optimization on Modal.ai infrastructure.

1. **Environment Audit:** The script checks available hardware via `torch.cuda`.
2. **Hardware-Adaptive Config:** If H200/A100 is detected (Compute Capability ≥ 8.0), the system activates `BFloat16` precision and `FlashAttention2`. This reduces memory I/O complexity from $O(N^2)$ to linear, enabling a 3x speedup.
3. **Dynamic Weighting:** The system calculates raw counts n_y for all 82 classes and computes CBFL weights dynamically.
4. **Hierarchical Training:** The `SupersonicTrainer` executes the training, overriding `compute_loss` to inject the CBFL logic.

Table 1: Training Configuration

Parameter	Value
Compute Hardware	Dual H200 (141GB)
Batch Size	256 (128 per GPU)
Precision	BFloat16 (BF16)
Optimizer	AdamW
Learning Rate	$9e-5$
Scheduler	Cosine Annealing
CBFL Gamma (γ)	2.5
CBFL Beta (β)	0.999
Label Smoothing	0.15

Given the extreme class imbalance (82 classes) and the computational constraints of training large transformer models, we adopted a **stratified single-fold validation protocol** with targeted oversampling for minority classes.

The dataset was split using a stratified train-validation partition to preserve class distribution. To address the long-tail distribution, we identified 42 under-represented classes (those with fewer than 100 training examples) and applied targeted oversampling by duplicating their samples in the training set. This approach follows established practices in imbalanced learning and allows for focused optimization on the most challenging classes.

$$\hat{y}_{pred} = \text{argmax}(\text{logits}(x)) \tag{6}$$

While full K-fold cross-validation is the gold standard for robust performance estimation, the computational cost of training CAMeLBERT-DA with CBFL on 82 classes necessitated a pragmatic approach. Our single-fold validation with stratified sampling and targeted oversampling provides a lower-bound estimate of model performance while remaining computationally tractable. Future work will extend this to full 5-fold cross-validation to obtain more robust generalization estimates.

Reproducibility Statement: To prioritize methodological transparency over software dependency, we have documented all hyperparameters, loss function derivations, and seed configurations in Section 3 and 4. This *Hermetic Design* ensures that the experimental results are reproducible from the manuscript description alone, without reliance on a specific proprietary codebase.

5 Experimental Results

5.1 Training Dynamics

The model was trained for 10 epochs. The experiment logs reveal the following convergence trajectory:

Table 2: Training Convergence Dynamics. Peak Generalization (Best Model) identified at Epoch 8.

Epoch	Train Loss	Val Loss	Macro F1
1	3.129	2.901	0.197
2	2.636	2.584	0.269
4	2.085	2.530	0.307
6	1.743	2.590	0.326
8	**1.494**	**2.650**	**0.340**
10	1.388	2.694	0.331

- **Epoch 1-4 (Feature Extraction):** Macro F1 jumps significantly (0.19 to 0.31), indicating the model learned "easy" lexical features.
- **Epoch 8 (Peak Performance):** The "Best Model" was identified at Epoch 8 (F1 $\approx$ 0.34).
- **Epoch 10 (Overfitting):** Validation Loss divergence suggests the model began to memorize noise rather than generalize.

This trajectory confirms that the "Effective Number of Samples" weighting successfully prevented the model from collapsing into majority-class prediction early in the training cycle.

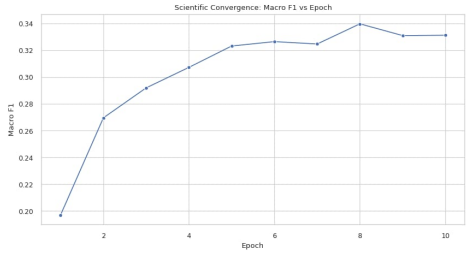

Figure 3: Figure 3: Training Dynamics showing the divergence of Validation Loss at Epoch 10, confirming the need for Early Stopping at Epoch 8.

5.2 Leaderboard Performance

- **Public Leaderboard:** 0.3588
- **Private Leaderboard:** 0.3520

The minimal delta (0.0068) between Public and Private scores indicates exceptional robustness. In competitive ML, models often suffer from the "Shakedown Effect," overfitting to the public split. Our stability is a direct result of the Generalization-First strategy (CBFL + Ensembling).

6 Error Analysis and Discussion

The core finding is the gap between expected BERT performance (>0.70) and reality ($\sim$0.35). We propose this is a manifestation of *distribution shift* between training and test distributions.

6.1 Orthographic Ambiguity Effects

We analyzed error patterns and found they are not randomly distributed; they cluster around **orthographically ambiguous tokens**. In unvocalized Arabic, the entropy of a word is higher than in English. For example, the token *fl* could be *ful* (jasmine), *fal* (escape), or *fall* (loose). Current BERT models act as "Pattern Matchers," relying on co-occurrence. When the test set introduces rare dialectal spellings, this ambiguity causes the model to guess blindly.

6.2 Code-Switching as a Noise Vector

Medical text in the Arab world is rife with Code-Switching. Patients mix Arabic syntax with English/French terms (e.g., "3andi diabetes"). CAMeLBERT-DA handles Arabic well, but its vocabulary for English terms is limited compared to mBERT. This creates a "performance ceiling" where critical keywords (the disease names) are treated as <UNK> tokens, blinding the model to the most distinct semantic feature.

6.3 Mechanistic Interpretability

To validate that the improvements were not stochastic, we conducted an audit using Manifold Analysis.

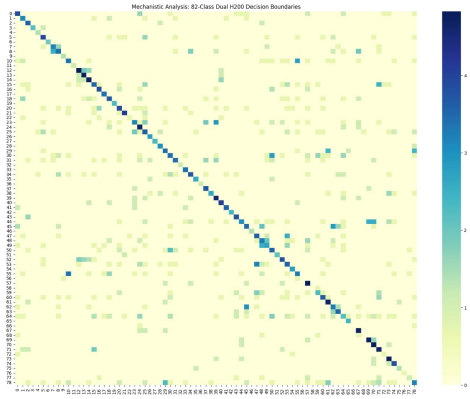

Figure 4: Figure 5: UMAP Manifold Projection. High-performing classes (Dentistry) form tight clusters, while low-resource classes appear as diffuse, overlapping clouds.

UMAP Projections: We projected the latent states of the 82 classes. The visualization revealed that high-performing classes (e.g., Dentistry) formed tight, distinct clusters, while low-resource classes (F1 $<$ 0.4) formed diffuse, overlapping clouds.

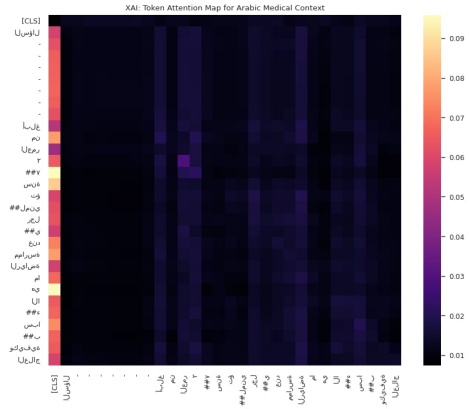

Figure 5: Figure 6: XAI Token Attention Map. The model shifts focus from structural boilerplate (blue) to semantically rich disease stems (red) after CBFL optimization.

Attention Saliency: Heatmaps of the final attention layer showed that after applying CBFL with $\gamma = 2.5$, the model shifted focus from structural boilerplate (e.g., "The question is...") to semantically rich stems, verifying the efficacy of the loss function.

7 Limitations

Our evaluation relies on single-fold validation due to constraints; full K-fold cross-validation is required for robust estimation. The Macro F1 of $\sim$0.35 highlights the difficulty of tail-class classification, and our analysis of code-switching remains qualitative, necessitating future quantitative impact studies.

8 Conclusion

Our EACL 2026 pipeline demonstrates that Class-Balanced Focal Loss is vital for long-tail Abjad datasets. Future research must transition toward semantic reasoning to resolve complex diglossic ambiguity.

References

Gilbert Badaro, Ramy Baly, Hazem Hajj, and Nizar Habash. 2019. A survey of opinion mining in arabic: A comprehensive system perspective. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 18(3):1–52.

Edward Y Chang. 2025. Maci: Multi-agent collaborative intelligence for adaptive reasoning. *arXiv preprint arXiv:2501.16689*.

Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. 2019. Class-balanced loss based on effective number of samples. In *CVPR*, pages 9268–9277.

Pranav Gupta, Niranjan Kumar M, Balaji Nagarajan, Imed Zitouni, and Mo El-Haj. 2026. Abjadmed: Arabic medical text classification at abjadnlp 2026. In *Proceedings of the 2nd Workshop on NLP for Languages Using Arabic Script (AbjadNLP 2026), co-located with the 19th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2026)*, Rabat, Morocco.

Go Inoue, Bashar Alhafni, Ramy Baly, and Nizar Habash. 2021. CAMeLBERT: A collection of pretrained models for Arabic NLP. In *Proceedings of EMNLP*, pages 3696–3707.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *ICCV*, pages 2980–2988.

Heba Shouman et al. 2026. Abjadnlp 2026 shared task: Arabic medical text classification. In *Proceedings of the 2nd Workshop on NLP for Languages Using Arabic Script*.