A Closed-Track System for Palestinian Arabic in the AMIYA Shared Task

Khaleel Hamad
Brigham Young University
khamad@byu.edu

Ahmad Al-Najjar
University Of Utah
u1386803@utah.edu

Abstract

We describe a closed track system for modeling Palestinian Arabic that is developed for the AMIYA shared task using a parameter efficient fine-tuning strategy. A 1.5B instruction-tuned language model was adapted with LoRA (Hu et al., 2021), updating only .28% of the model parameters, and trained on an aggregated set of conversations between Palestinians and resources covering both translation and generation. Model selection was guided by a comparative benchmark that prioritized performance efficiency and its tradeoffs. At the same time the paper focuses on targeting error analysis as well as structured instruction following. These findings illustrate both the viability and shed light on the current limitations of efficient adaptation methods for low-resource Arabic dialects.

1 Introduction

This paper presents our submission to the AMIYA Shared Task (Arabic Modeling In Your Accent), which aims to advance dialectal Arabic language modeling. The competition includes three tracks (1) closed data, (2) closed model, and (3) closed. Each imposed different constraints on the allowed training data and/or model choice. We participated in the closed track as it aligns with the task constraints while still enabling informed system design and evaluation. Following the shared task guidelines, which restrict the use of external resources in the closed track, we carefully selected and processed only the permitted datasets relevant to our target variety. We eventually selected the Palestinian dialect track, motivated by our prior experience developing a dialect-focused model and our familiarity with Palestinian Arabic, which supported dialect-aware data selection and error analysis. This paper makes five primary contributions, each addressing a distinct stage of the system development process: (1) a multi-source Palestinian Arabic dataset aggregation framework combining more than four dialectal corpora; (2) a parameter-efficient fine-tuning pipeline using LoRA adapters (rank=16, alpha=32), updating only 0.28% of model parameters; (3) a structured model comparison framework enabling systematic evaluation of multiple LLM architectures using standardized diagnostic metrics; (4) an instruction-tuning strategy targeting both generation and translation tasks through unified prompt formatting; and (5) a flexible data preparation pipeline supporting reproducible experimentation. Together, these contributions highlight both the scarcity of Palestinian dialect resources and the importance of shared-task initiatives such as AMIYA in advancing inclusive Arabic language technologies.

2 Database

The dataset is a Palestinian Arabic corpus prepared for the AMIYA shared task at VarDial 2026 and is used to train and evaluate LLMs for dialect translation and generation. Examples are combined from several existing dialect resources into a single, constantly formatted JSONL dataset with pairs of instruction-response.

Our data is constructed from four underlying Palestinian dialect sources while the Combined Dialect Dataset is a reference to their union.(Abdul-Mageed et al., 2024; Talafha et al., 2024).We extract the Palestinian subset, which adds conversational, spoken-style utterances to the corpus. The dataset is constructed from the following dialectal resources:

1. **Maknuune Corpus** (Dibas et al., 2022): A Palestinian Arabic lexicon containing dialect words and phrases paired with Modern Standard Arabic (MSA) and English glosses. The corpus provides lexical entries and example sentences, enabling PalestinianMSA and PalestinianEnglish translation tasks.

Proceedings of the Thirteenth Workshop on NLP for Similar Languages, Varieties and Dialects, pages 385–392

2. **Shami Dataset** (Kwaik et al., 2018): A Levantine Arabic corpus derived from Twitter data. We retain only samples explicitly annotated as Palestinian dialect to capture user-generated, informal language typical of social media.

3. **Casablanca Corpus** (Abdul-Mageed et al., 2024; Talafha et al., 2024): A speech transcription dataset annotated for dialect, gender, and code-switching. We extract the Palestinian subset to incorporate spoken and conversational utterances.

4. **MASC Corpus**: A Jordanian Arabic review dataset consisting of 1,414 sentiment-annotated customer reviews. This corpus is used exclusively for auxiliary diagnostic experiments and is not included in AMIYA training or evaluation.

5. **JODA Corpus**: A large Jordanian Arabic corpus comprising social media text, film transcripts, and aligned MSA translations. Similar to MASC, this dataset is used only for diagnostic analysis and is excluded from AMIYA system training.

6. **Combined Dialect Dataset (ours)**: A unified JSONL corpus created by aggregating Palestinian Arabic instances from Maknuune, Shami, and Casablanca. Each example preserves source metadata and is formatted into instructionresponse pairs for generation and translation tasks.

2.1 Data Statistics

The Combined Dialect Dataset (unified corpus) contains 155,299 entries drawn from the five Arabic dialect datasets, the average text length of about 50 characters and a median of 35 characters. At the dialect level Palestinian contributes 48,278 instances, Jordanian contributes 69,262, and Syrian 37,759, giving a reasonably balanced Levantine pool from which the Palestinian subset can be extracted and used for AMIYA. In terms of source datasets, JODA corpus is the largest component with 59,135 sentences, followed by the Shami twitter corpus (55,418), the Maknuune lexicon (36,302 entries), Casablanca speech transcriptions (3,030 segments), and the smaller MASC review corpus (1,414 documents) (Figure 1).

The Maknuune lexicon contains primarily of very short lexicon items with a mean length of

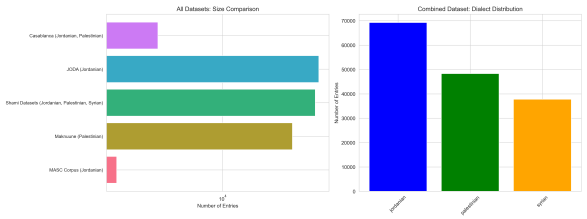

Figure 1: Overall comparison

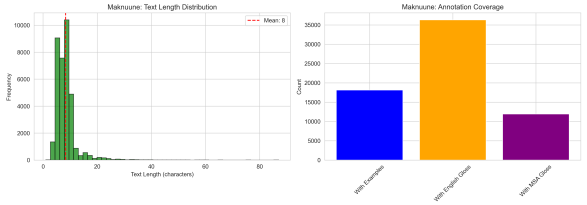

Figure 2: Maknuune analysis

roughly 8 characters (1.2 words), and entries offer additional supervision: 18,140 entries include example sentences, and all entries have their English glosses paired, and 11,905 (32.8%) provide MSA translations (Figure 2). The Shami datasets contain sentence-level conversational text in Palestinian(10,642), Jordanian(7,017), and Syrian dialects(37,759), with average lengths between 67 and 86 characters with noticeable long tail of longer tweets (Figure 3). JODA corpus adds 59,135 Jordanian sentences with a mean length of 49 characters (9.5 words); the split assigns about 91.5% of examples for training and 4.2% each to validation and tests as illustrated by the length histogram and split pie chart below (Figure 4).

The remaining resources contribute more domain-specific or spoken material. The MASC corpus provides 1,414 Jordanian customer reviews with an average length of 245 characters (46.2 words) annotated with positive or negative sentiment labels (Figure 5). Casablanca adds 3,030 speech transcripts including 1,334 Palestinian and 1,696 segments whose length distributions speak around 50-70 characters, with Palestinian utterance is slightly longer on average against the Jor-

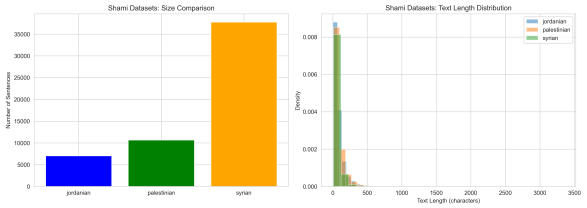

Figure 3: Shami Analysis

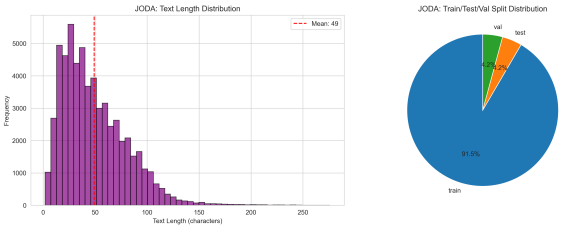

Figure 4: Joda Analysis

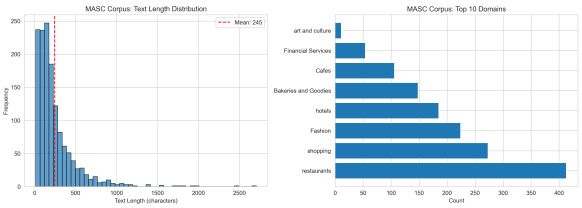

Figure 5: Masc Corpus

danian segment (Figure 6).

Overall, these statistics and their visualizations represent the pool from which we curate our Palestinian AMIYA dataset spans short lexical entities, sentence-level social media and translations pairs, and longer narrative or conversational passages, providing a diverse foundation for dialect generation and translation experiments.

Based on these statistics, we determined that sufficient Palestinian dialect data is available to support closed-track training without incorporating non-Palestinian dialects. Accordingly, only Palestinian-labeled instances from Maknuune, Shami, and Casablanca were used for AMIYA training and evaluation. Jordanian and Syrian data were retained solely for auxiliary diagnostic analyses and were excluded from all shared-task experiments.

2.2 Data Processing

For pre-processing the data we apply a uniform pre-processing pipeline that (1) loads Palestinian examples from all available sources, (2) filters and normalizes the raw text (3) converts examples into task specific input-output pairs, and (4) wrap them into an instruction format that is compatible with Llama-3 (Grattafiori et al., 2024). Dialect labels and encodings are inherited from the combined corpus pre-processing (section 3.b), where text has been cleaned, normalized, and dialect tags standardized to lowercase.

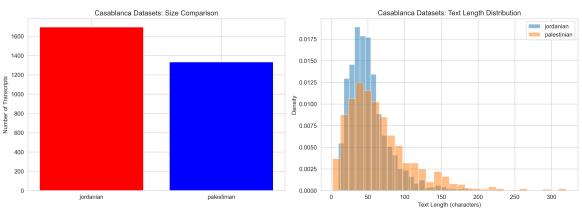

Figure 6: Casablanca

For generation tasks, a function is created to construct monolingual Palestinian examples by identifying the appropriate text column in each source (e.g. <Text>, < Sentences> , <example_usage>) and discarding entries with missing fields or length below a configured minimum threshold. Valid texts are used as outputs and conditions on the same content (for example, Write in Palestinian dialect: or Generate Palestinian dialect text about:), which makes the task instruction following rather than pure language modeling.

For translation tasks, a function is created that exclusively operates on the Maknuune lexicon, where it builds a Palestinian side by preferring examples sentences and falling back to the base dialect forms, splitting into multi-translations, it filters out very short or empty segments, and then creates aligned pairs for 4 directions (MSADA, DAMSA, ENDA, DAEN).

The resulting examples are converted into instruction style records, which puts each input-output pair into a task-specific template from <AMIYA_PROMPT_TEMPLATES> and stores the rendered prompt in an <instruction> field while preserving the raw input, output, task_type, and source metadata.

We partition the final instruction-formatted corpus into training, validation, and test sets using the <split_data> routine function from all examples to a pandas DataFrame, then applies a two-stage random split with 80/10/10 proportions; 80% for training, 10% for validation, and 10% for testing using <train_test_split> (Pedregosa et al., 2011) with a fixed random seed, ensuring that results are reproducible and that no example appears in more than one split.

The split is performed jointly over all task types so that each subset contains a representative mixture of generation and translation examples, while preserving the original task type and source metadata for downstream analysis.

Listing 1: Pseudocode: data preparation and model selection

```
DATASETS = [Maknuune, Shami, Casablanca]
```

```
ALL = []

for ds in DATASETS:
  rows = load(ds)
  rows = keep_label(rows, "Palestinian")
  rows = normalize_text(rows)
  rows = drop_missing(rows)

  gen = build_generation_pairs(rows)
  trn = build_translation_pairs(rows)

  ALL = ALL + gen + trn

TRAIN, DEV, TEST = split(ALL, 0.8, 0.1, 0.1,
    seed=42)

CANDIDATES = [TinyLlama_1p1B,
              Qwen2_1p5B,
              Qwen2_7B,
              Llama3p1_8B,
              Dallah_3B,
              Jais_13B]

SCORES = {}
for m in CANDIDATES:
  subset = sample(DEV, n=500, seed=42)
  y_true = get_labels(subset)
  y_pred = infer_labels(m, subset)

  acc = accuracy(y_true, y_pred)
  f1m = macro_f1(y_true, y_pred)
  tps = throughput(m, subset)

  SCORES[m] = (acc, f1m, tps)

BEST = argmax(SCORES, key="f1m_then_tps")
```

3 Systems

Based on these statistics, we determined that sufficient Palestinian dialect data is available to support closed-track training without incorporating non-Palestinian dialects. Accordingly, only Palestinian-labeled instances from Maknuune, Shami, and Casablanca were used for AMIYA training and evaluation. Jordanian and Syrian data were retained solely for auxiliary diagnostic analyses and were excluded from all shared-task experiments.

- TinyLlama-1.1B (Zhang et al., 2024)
- Qwen2-1.5B-Instruct (Yang et al., 2024)
- Qwen2-7B-Instruct (Yang et al., 2024)
- Meta-Llama-3.1-8B-Instruct (Grattafiori et al., 2024)
- Dallah-3B-v1
- Jais-13B-chat (Sengupta et al., 2023)

Metric	Value
TinyLlama-1.1B	
Accuracy	44.00%
Macro F1	20.37%
Weighted F1	26.89%
Avg Inference Time (s)	0.9161
Samples/sec	1.09
Qwen2-1.5B-Instruct	
Accuracy	44.00%
Macro F1	24.76%
Weighted F1	29.83%
Avg Inference Time (s)	0.0975
Samples/sec	10.25

Table 1: Diagnostic model performance and inference speed on a 500-sample subset. Detailed diagnostic scores were retained only for the two top-performing models, which were selected for further experimentation.

The first thing was to compare between them on a small dataset of around 500 rows, mainly due to limited resources. Next was to filter out all except the 2 highest: TinyLlama-1.1B and Qwen2-1.5B-Instruct (Table 1).

For completeness, Table **??** reports the same diagnostic metrics for the remaining candidate models evaluated on the same 500-sample subset. These results are included only to contextualize model selection and were not part of the AMIYA official evaluation.

During preliminary screening, all candidate models were evaluated using the same diagnostic setup. However, detailed per-model metrics were retained only for the two top-performing models that were selected for fine-tuning and downstream analysis. For the remaining candidates, evaluation was used solely to inform relative ranking and early filtering decisions, and detailed diagnostic outputs were not preserved.

Accuracy, predicted labels, and true labels reported in the confusion matrices correspond to an internal diagnostic dialect classification task used during model comparison. These metrics are not part of the AMIYA evaluation framework, and ADI2 or similar shared-task metrics are not applicable at this stage. The purpose of this analysis is to inspect error patterns rather than competitive task scores.

Although Qwen was the winner in most of these results, deeper analysis was required to ensure that it will be the best model to work with (Figures 7

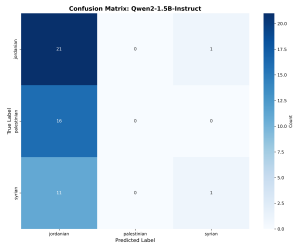

Figure 7: Confusion matrix: Qwen2-1.5B-Instruct

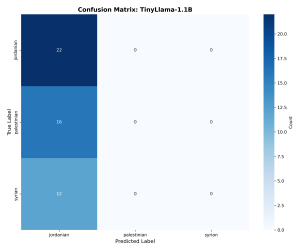

Figure 8: Confusion matrix: TinyLlama-1.1B

Metric	Value
Jordanian	
Precision	43.75%
Recall	95.45%
F1 score	60.0%
Support	22
Palestinian	
Precision	0.00%
Recall	0.00%
F1 score	0.00%
Support	16
Syrian	
Precision	50%
Recall	8.33%
F1 score	14.23%
Support	12

Table 2: Dialect classification metrics.

and 8). From these truth tables the final decision to choose Qwen was taken. Next, we examined the models understanding of the different dialects, where a new analysis was run for the models where Qwen came out with the highest scores of (Table 2). Dialect classification was used as an exclusively diagnostic tool during model selection and error analysis. It wasn't created as part of the AMIYA shared task evaluation and is not used at inference time for the submitted system. The purpose of this analysis was to assess the models' sensitivity to the varied dialects we are working on prior to fine-tuning, rather than using it to optimize performance for a standalone dialect identification task.

It is also important to note that dialect classification was conducted solely as a diagnostic tool for model selection and error analysis. It is not part of the AMIYA shared task evaluation nor used at inference time in the submitted system. In this case the results seemed to be too far apart to be used as the main decision making tool, which is why there was a restoration for other methods. So overall the decision to go with Qwen as the model to work on is a good decision because it has the better f1 score, has a much faster inference (10 samples per sec), as well as lower latency (.9161 per sample).

The final system that was built uses *Qwen/Qwen2.5-1.5B-Instruct* (Yang et al., 2024) as the base model, a 1.5 billion - parameter decoder-only transformer optimized for instruction following. The architecture follows a standard GPT style decoder following the multihead self attention techniques, feed-forward networks, layer normalization, and residual connections. The model is loaded with mixed precision for memory efficiency, as shown in the model initialization code. As a simple baseline, we evaluated the base Qwen2.5-1.5B-Instruct model in a zero-shot setting using the same instruction templates and prompts as our fine-tuned system. Relative to this zero-shot baseline, LoRA fine-tuning improved instruction adherence and dialectal consistency, with the clearest gains observed in generation outputs.

Listing 2: Model loading code

```
model = AutoModelForCausalLM.from_pretrained(
    FINETUNING_MODEL_ID,
    dtype=torch.bfloat16 if device == "cuda"
        else torch.float32,
    device_map="auto" if device == "cuda" else
        None,
    trust_remote_code=True,
    low_cpu_mem_usage=True
)
```

This base model includes around 1.5 billion parameters, all of which remain frozen during fine-tuning. Gradient checkpointing is enabled to reduce memory consumption during training, allowing the model to really trade computation for memory by basically recomputing activations during backpropagation rather than storing them.

The fine-tuning process was done using LoRA (Hu et al., 2021) to adapt to only attention projection layers. LoRA injects trainable low-rank met-

rics into the attention mechanism, enabling efficient adaptation with minimal parameter overhead. This would target the four projection layers in each transformer block:

Listing 3: LoRA configuration and model wrapping

```
lora_config = LoraConfig(
    r=LORA_R, # Rank: 16
    lora_alpha=LORA_ALPHA, # Alpha: 32
    target_modules=LORA_TARGET_MODULES, #
        ["q_proj", "k_proj", "v_proj", "o_proj"]
    lora_dropout=LORA_DROPOUT, # Dropout: 0.1
    bias="none",
    task_type=TaskType.CAUSAL_LM,
)

model = get_peft_model(model, lora_config)
```

Here the rank parameter (r = 16) is controlling the dimensionality of the low-rank decomposition, while the alpha parameter (alpha = 32) acts as a scaling factor. The configuration has resulted in 4.2 million trainable parameters, representing around only .28% of the base models total parameters. This allowed training in a much smaller time window and became much less resource depleting.

As for the training process it follows a structured pipeline beginning with data preparation , where Palestinian dialect examples are loaded from the JSON files and then converted to HuggingFace Dataset format (Lhoest et al., 2021). The training data here is just limited to 15,000 examples for computational efficiency, with task type distribution monitored to ensure balanced representation across the generation and translation tasks. For the final presentation model further training will be performed using the entire dataset. As for training, the **Trainer** class from huggingface transformers (Wolf et al., 2020) orchestrates the forward pass, loss computation and backpropagation as well as a couple of other tasks. The process generates comprehensive training statistics, including trainable parameter counts.

Training was conducted by using HuggingFace's Trainer framework with AdamW optimizer, a learning rate of 2e-4, batch size of 8 per device, and gradient accumulation steps of 4 for a total of 3 epochs. A fixed random seed was used for data splitting and training. Experiments were run on different GPUs with mixed precision enabled. These settings were kept constant across all compared models to ensure fair comparison.

4 Results

For the generation task, qualitative analysis of model output was conducted revealing both strengths and limitations. For example, when prompted to generate a greeting the model outputs natural Palestinian dialect phrases that are able to preserve regional linguistic features. Still, translation tasks are revealing limitations when translating from Modern Standard Arabic MSA to Palestinian Dialect. The model also seems to sometimes repeat the input MSA text rather than producing actual dialect output, indicating insufficient translation training data. Sample outputs have also demonstrated that the model is handling longer, contextually rich inputs better than short phrases or single words; this suggests that contextual information aids dialect identification.

Listing 4: LLM-as-judge prompt template used for qualitative audit

```
Given:
(1) Task instruction
(2) Model output
(3) Reference output

Decide one label: {Correct, PartiallyCorrect,
    Incorrect}
Criteria:
- For translation: meaning preservation and
    dialect appropriateness
- For generation: Palestinian dialect usage and
    instruction adherence
Return:
- label
- 1-2 sentence justification
```

4.1 LLM-based qualitative audit (error analysis)

Based on the 200-sample qualitative audit, we categorized model errors into a small number of recurring patterns. Table **??** summarizes the most frequent error types observed in the sampled outputs, along with their relative frequency and representative examples. These categories were derived inductively by inspecting incorrect and partially correct outputs and are intended to highlight systematic weaknesses rather than provide an exhaustive error taxonomy.

The most frequent errors are closely tied to data sparsity and prompt ambiguity, suggesting that richer contextual prompts and additional dialectal supervision may improve performance in future iterations.

In addition to the observed weaknesses, the model also demonstrates consistent strengths in

Error type	Freq.
Dialect drift to MSA	31%
Instruction violation	24%
Input copying	18%
Semantic shift	15%
Hallucination	12%

Table 3: Distribution of error types in the 200-sample qualitative audit.

Error type	Typical manifestation
Dialect drift	Output rendered in MSA
Instruction violation	Wrong task format
Input copying	Source text unchanged
Semantic shift	Meaning partially altered
Hallucination	Added content

Table 4: Qualitative descriptions of error types observed during audit.

Phenomenon	Success observed
Dialectal greetings	Natural informal salutations
Colloquial particles	Correct discourse markers
Verb morphology	Dialect-appropriate forms
Contextual coherence	Consistent dialect in context
Idiomatic usage	Region-appropriate expressions

Table 5: Representative strengths observed in Palestinian Arabic generation during the qualitative audit.

common dialectal constructions. Table 5 highlights representative cases where the model successfully produced natural Palestinian Arabic outputs.

To assess whether dialectal expressions were memorized or generalized, we inspected outputs corresponding to common idiomatic phrases. While some frequent expressions are likely present verbatim in the training data, the model also produces structurally similar variants that were not observed directly in the source corpora. This suggests that the model learns productive dialectal patterns rather than relying solely on memorization, although data sparsity limits the robustness of this generalization for less frequent constructions.

5 Discussion

Overall, parameter-efficient fine-tuning using LoRA proves effective for adapting a compact instruction-tuned model to Palestinian Arabic under constrained data and computational budgets. While the fine-tuned system demonstrates strengths in dialectal generation and idiomatic usage, limitations persist in translation robustness and instruction adherence. The qualitative error analysis highlights data sparsity and prompt ambiguity as primary sources of failure, indicating directions for future improvements.

6 Conclusion

This paper presented our submission to the AMIYA Shared Task closed track, focusing on Palestinian Arabic modeling through parameter-efficient fine-tuning. By aggregating multiple dialectal resources and applying LoRA adaptation to a 1.5B instruction-tuned model, we demonstrate the feasibility of efficient adaptation for low-resource Arabic dialects. At the same time, persistent challenges in translation accuracy and instruction adherence underscore the need for richer, task-aligned datasets and combined qualitativequantitative evaluation. We hope this work contributes practical insights toward more inclusive Arabic language technologies.

7 References

References

Muhammad Abdul-Mageed, Basel Talafha, Motaz Saad, and 1 others. 2024. Casablanca: Data and models for multidialectal arabic speech recognition. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

S. Dibas, C. Khairallah, N. Habash, O. F. Sadi, and 1 others. 2022. Maknuune: A large open palestinian arabic lexicon. In *Proceedings of the Seventh Arabic Natural Language Processing Workshop (WANLP)*.

A. Grattafiori, A. Dubey, Y. Jernite, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zhiqing Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

K. A. Kwaik, M. Saad, S. Chatzikyriakidis, and S. Dobnik. 2018. Shami: A corpus of levantine arabic dialects. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC)*.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, and 1 others. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*.

F. Pedregosa, G. Varoquaux, A. Gramfort, and 1 others. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830.

N. Sengupta, S. K. Sahu, B. Jia, and 1 others. 2023. Jais and jais-chat: Arabic-centric foundation and instruction-tuned open generative large language models. *arXiv preprint arXiv:2308.16149*.

B. Talafha, M. Abdul-Mageed, and 1 others. 2024. Casablanca: A multi-dialect arabic speech dataset with code-switching. In *Proceedings of EMNLP 2024*.

Thomas Wolf, Lysandre Debut, Victor Sanh, and 1 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*.

A. Yang, B. Chen, B. Zhang, and 1 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

P. Zhang, G. Zeng, T. Wang, and W. Lu. 2024. Tinyllama: An open-source small language model. *arXiv preprint arXiv:2401.02385*.