Fanar
An Arabic-Centric Multimodal Generative AI Platform

FANAR TEAM,[*] Ummar Abbas, Mohammad Shahmeer Ahmad, Firoj Alam, Enes Altinisik, Ehsannedin Asgari, Yazan Boshmaf, Sabri Boughorbel, Sanjay Chawla, Shammur Chowdhury, Fahim Dalvi, Kareem Darwish, Nadir Durrani, Mohamed Elfeky, Ahmed Elmagarmid, Mohamed Eltabakh,[†] Masoomali Fatehkia, Anastasios Fragkopoulos, Maram Hasanain, Mohammed Qusay Hashim, Majd Hawasly, Mus'ab Husaini, Soon-Gyo Jung, Ji Kim Lucas, Walid Magdy, Safa Messaoud, Abubakr Mohamed, Tasnim Mohiuddin, Basel Mousi, Hamdy Mubarak, Ahmad Musleh, Zan Naeem, Mourad Ouzzani, Dorde Popovic, Amin Sadeghi, Husrev Taha Sencar, Mohammed Shinoy, Omar Sinan, Yifan Zhang, Ahmed Ali,[‡] Yassine El Kheir[‡], Xiaosong Ma[‡], and Chaoyi Ruan[‡]

Qatar Computing Research Institute (QCRI),
Hamad Bin Khalifa University

---

[*]The author ordering is alphabetical by last name. See Section A for contribution details.
[†]Corresponding author: meltabakh@hbku.edu.qa
[‡]Members are not currently affiliated with QCRI. Their work in Fanar was entirely done while at QCRI.

Abstract

We present Fanar, a platform for Arabic-centric multimodal generative AI systems, that supports language, speech and image generation tasks. At the heart of Fanar are **Fanar Star** and **Fanar Prime**, two highly capable Arabic Large Language Models (LLMs) that are best in the class on well established benchmarks for similar sized models. **Fanar Star** is a 7B (billion) parameter model that was trained from scratch on nearly 1 trillion clean and deduplicated Arabic, English and Code tokens. **Fanar Prime** is a 9B parameter model continually trained on the Gemma-2 9B base model on the same 1 trillion token set. Both models are concurrently deployed and designed to address different types of prompts transparently routed through a custom-built orchestrator. The Fanar platform provides many other capabilities including a customized Islamic Retrieval Augmented Generation (RAG) system for handling religious prompts, a Recency RAG for summarizing information about current or recent events that have occurred after the pre-training data cut-off date. The platform provides additional cognitive capabilities including in-house bilingual speech recognition that supports multiple Arabic dialects, voice and image generation that is fine-tuned to better reflect regional characteristics. Finally, Fanar provides an attribution service that can be used to verify the authenticity of fact based generated content.

The design, development, and implementation of Fanar was entirely undertaken at Hamad Bin Khalifa University's Qatar Computing Research Institute (QCRI) and was sponsored by Qatar's Ministry of Communications and Information Technology to enable sovereign AI technology development.

Contents

1. Introduction

A case for Arabic-centric Large Language Models is made including the limitations of obtaining Arabic data and the distinctive characteristics of the language.

Large Language Models (LLMs) and Generative AI are becoming an integral part of day-to-day activities at the personal and enterprise level due to their ability to carry out multifaceted language and cognitive tasks. Diverse applications, including writing assistants, translation services, customer support, software development and image generation, are proliferating and being offered as human productivity enhancing tools. While failure cases of LLMs tend to become viral, continued growth in their adoption is an indicator of their practical utility. From a computer science and AI research perspective, LLMs and their multimodal extensions have opened up scientific challenges that will continue to drive innovation in the research community. An important practical challenge that is far from being overcome is the design and engineering of high-quality and effective LLMs for non-English languages. The major bottleneck for non-English languages is the limited availability of large datasets which are currently necessary to match the performance of English-centric models. As is well known, the most ubiquitous language on the Web, from where most data is harvested, is English. The latest statistics from Common Crawl, a non-profit organization that takes a complete snapshot of the Web approximately once a month, show that English documents constitute 46% of all textual web content, while other languages cap at around 6%. Arabic, the spoken language of more than 400 million people and the official language of over 20 countries, constitutes around 0.5% of web data[1] (Rana, 2010). Besides lack of data, another big hurdle is the cost of building an LLM, particularly from scratch, in terms of both the required hardware and technical expertise. The geopolitical environment is trending towards a polarized world where access to high-end GPUs is often constrained and even when hardware is available, the monetary cost of training even a moderate size LLM in a reasonable amount of time can be prohibitive and out of reach for many organizations. Finally, substantial scientific and engineering expertise is required to undertake an LLM building exercise. While many organizations that have built private and open source LLMs release technical reports about their experience, in many cases, technical details are often left out making it difficult to replicate the process of building an LLM from scratch. A significant amount of deep knowledge across the entire LLM stack, ranging from data collection and cleaning, to pre-training, post-training, and deep computer systems knowledge is therefore required.

In this work, we introduce Fanar (meaning lighthouse in Arabic), an Arabic-centric Generative AI platform that includes text-based LLMs, speech and image generation systems, specialized Retrieval Augmented Generation (RAG) modules, and an attribution service to authenticate and correct facts in generated text. At the center of Fanar are **Fanar Star** and **Fanar Prime** two 7B and 9B parameter LLMs respectively that are trained on nearly 1 trillion clean and deduplicated Arabic, English and Code tokens. **Fanar Star** was trained from scratch while **Fanar Prime** was continually pre-trained on the Gemma-2 9B model on the same 1 trillion data set. The two models work in concert and make up for the lack of Arabic data, especially in technical domains. Additionally, Fanar includes a new customized morphologically aligned Arabic tokenizer as well as benchmarks that evaluate cultural capabilities that will be of independent interest to the Generative AI research community.

The rest of this report is structured as follows. In Section 2, we give a brief overview of the Arabic language, its footprint, dialects and setup the socio-linguistic context. A high level overview of the major components of the Fanar platform and how they interact with each other is provided in Section 3. The efforts to collect, clean and integrate a large Arabic data set of nearly 1 trillion tokens are detailed in Section 4, followed by the design of our specialized Arabic tokenizer in Section 5. The model architectures of both **Fanar Star** and **Fanar Prime** and the pre-training pipeline are the subject of Section 6. A comprehensive overview of post-training steps and the necessary adaptations for Arabic are then provided in Section 7. The performance of both models on standardized and new culturally aware benchmarks is described in Section 8. Arabic speech services and the regionally representative image generation capabilities of Fanar are introduced in Section 9. The Retrieval Augmented Generation (RAG) systems for Islamic content, recent events and important biographies for fact-related response attribution are the focus of Section 10. Section 11 concludes with a reflection on building a large scale GenAI system and a discussion on a future road map for Fanar.

---

[1] https://commoncrawl.github.io/cc-crawl-statistics/plots/languages.html

2. Arabic Language

An overview of the Arabic language and its distinctive characteristics is provided, motivating the need for developing language technologies specifically tailored for Arabic.

The Arabic language, a member of the Semitic language family, is spoken by over 310 million people across the Middle East, North Africa, and the Arabian Peninsula (MENA region) (Horesh and Cotter, 2016), and by more than 467 million individuals across 60 countries worldwide (Gregory et al., 2021). Beyond its geographic and linguistic reach, Arabic carries immense spiritual significance as the liturgical language of over 2 billion Muslims who engage in daily prayers and religious practices in Arabic. It is the official or co-official language of 25 countries and holds a prominent place in linguistic and cultural studies due to its complexity and global significance (Gregory et al., 2021). Arabic exists as a spectrum of linguistic forms, ranging from Classical Arabic, which is used primarily in religious and classical literary texts, and Modern Standard Arabic (MSA), used in formal settings and communications, to a wide array of colloquial dialects spoken across the MENA region and widely used on social media platforms. The rich diversity and significance of Arabic, its complex grammar and structure, and the colloquial-nuances make it both a fascinating and challenging for language technology development (Farghaly and Shaalan, 2009; Habash, 2010).

At the core of Arabic's linguistic system lies a derivational morphology system where words are typically derived from roots that are fit into morphological templates and accept prefixes and suffixes. Roots, typically composed of three consonants (though occasionally four or five), encode fundamental semantic meanings. These roots combine with specific morphological patterns to produce words that convey nuanced meaning.

For example, the root *k-t-b* generates words such as *kitAb* ('book'), *maktabap* ('library'), *kutub* ('books'), *kAtib* ('writer'), *maktwb* ('written'), and many other nouns and adjectives, as well as verb conjugations that account for tense, gender, and person[2] (Watson, 2002). This non-linear system and Arabic's intricate inflectional paradigms, including the use of prefixes and suffixes that mark tense, mood, gender, and number and attaches determiners and pronouns, distinguishes it markedly from Indo-European languages. While Arabic derives a vast array of words from a limited set of roots, Indo-European languages such as English typically rely on concatenative morphology and a more extensive lexicon to achieve similar flexibility (Ferguson, 1959; Watson, 2002; Mustafa et al., 2017; Alolaywi, 2022). Arabic morphology allows for words such as وَبِكِتَابِهِمْ *wabikitAbihim* ('and with their book') and فَأَسْقَيْنَاكُمُوهُ *fa'asqaynākumūhu* ('And We gave it (water) to you to drink.').

The sociolinguistic phenomenon of diglossia further complicates the computational processing of Arabic. Modern Standard Arabic (MSA), derived from Classical Arabic, serves as the formal language for media, education, and political discourse. However, it is not the native language of any Arab speaker. Instead, native speakers use regional dialects, which differ significantly from MSA in phonology, syntax, and lexicon (Ferguson, 1959). These dialects are influenced by other languages, such as English, Berber, French, Persian, Turkish, and Aramaic, and can often be mutually unintelligible (Farghaly and Shaalan, 2009; Al-Wer and de Jong, 2017).

Another defining feature of Arabic is its script, which is derived from the Nabataean alphabet (Healey and Smith, 2012). The script, similar to other Semitic scripts, is written from right to left, and its adaptability allows it to represent unrelated languages such as Persian and Urdu. However, in most languages that use the Arabic script, short vowels are not explicitly represented, necessitating diacritization for accurate interpretation. This introduces challenges for NLP tasks, particularly those requiring diacritization or semantic disambiguation (Alnosairee and Sartini, 2021; Mubarak and Darwish, 2014b).

---

[2]Buckwalter encoding is used to represent Arabic text.

Although Arabic has many speakers and a profound cultural impact, the amount of Arabic content on the web accounts for only 0.5% of all online data, which presents significant challenges for technological development (Figure 17). The linguistic characteristics, cultural significance, and remarkable diversity of Arabic underscore the pressing need to develop language technologies specifically tailored to this language. Integrating linguistic expertise and anthropological insights into the development and evaluation processes is essential for effectively addressing the unique complexities inherent in Arabic. By doing so, we can ensure that these technologies not only meet the technical challenges of the language but also honor its rich cultural heritage and nuanced variations.

3. Fanar Platform Services

Fanar platform services are introduced including **Fanar Star** and **Fanar Prime**, speech and image generation capabilities and RAG features

The Fanar platform is organized into a set of services coordinated through an **Orchestrator** as shown in Figure 1. Requests from the Chat App or an API are either sent directly to speech and translation services or passed through a safety filter and then classified to be processed by other LLM-driven services. Prompts for image generation are also first passed through a safety filter. We briefly summarize the core services that make up the Fanar family. More details will be given in subsequent sections.
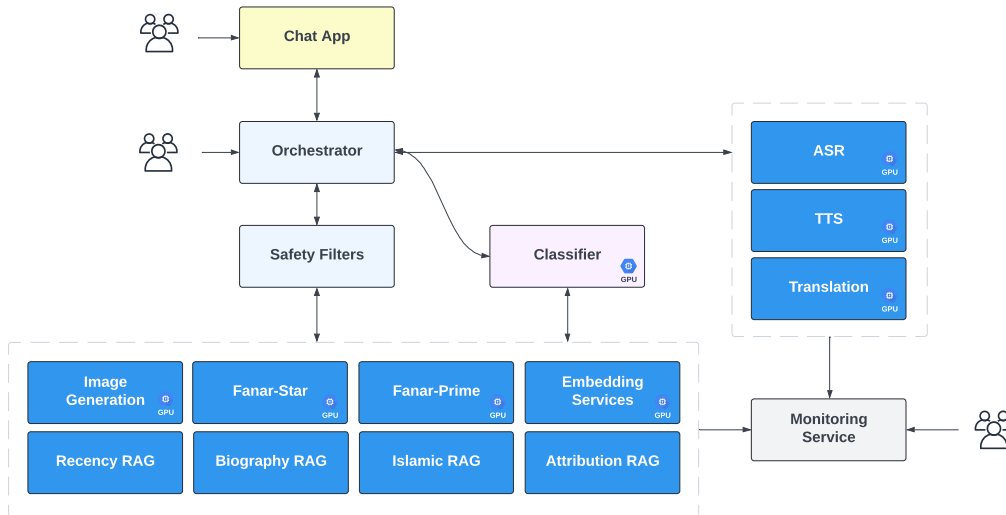
Figure 1: The GenAI services provided through the Fanar platform. The Orchestrator is responsible for routing prompts to the appropriate services depending upon the nature of the request. All responses are safety-checked to adhere to responsible AI guidelines and cultural alignment.

Fanar Star: This is the flagship 7 billion-parameter LLM trained entirely from scratch using a meticulously designed two-stage curriculum approach. This model leverages a refined implementation of the decoder-only Transformer architecture (Vaswani et al., 2017), inspired by the architectural principles of OLMo (Groeneveld et al., 2024) and the LLaMA family (Touvron et al., 2023). The pre-training process begins with a multi-epoch phase comprising two initial epochs over a diverse corpus of 1 trillion tokens, distributed across Arabic (40%), English (50%), and programming code (10%). In subsequent epochs, the token count is reduced to 0.8 trillion through targeted filtering using an education classifier, with an increased focus on Arabic content (50%), accompanied by proportionate adjustments for English (40%) and code (10%). The pre-training concludes with a carefully designed cool-down stage, incorporating an additional 100 billion tokens from high-quality datasets curated in-house and gradually diminishing the learning rate to zero. Detailed descriptions of the model architecture and pre-training recipe are provided in Section 6. **Fanar Star** undergoes a comprehensive post-training stage consisting of Supervised Fine-Tuning (SFT) and Direct Preference Optimization (DPO) (Rafailov et al., 2024) to achieve robust alignment with safety and ethical considerations, as detailed in Section 7. The deployment architecture

incorporates an orchestration mechanism whereby prompts other than Islamic or STEM domains, as determined by a specialized classifier, are routed to **Fanar Star**.

Fanar Prime: This model uses continual pre-training to build upon the Gemma-2-9B base model (Riviere et al., 2024), which was itself initially pre-trained on 8 trillion tokens using knowledge distillation from a larger model. Our approach begins with strategic vocabulary pruning, reducing the original 250,000-token vocabulary to 128,256 tokens to optimize compatibility with our training data. The model comprises 459 million embedding parameters and 8.32 billion non-embedding parameters, totaling 8.78 billion parameters. The continual pre-training process for **Fanar Prime** mirrors the two-stage curriculum strategy employed for **Fanar Star** but is limited to a single epoch, followed by a cool-down stage. The data mixture and filtering criteria are aligned with those of **Fanar Star** ensuring a balanced representation of Arabic, English, and code content. Post-training of **Fanar Prime** aligns closely with the methodology applied to **Fanar Star** including SFT and DPO for safety and value alignment. During orchestration, **Fanar Prime** is designated to handle STEM and reasoning-related prompts, routed through the specialized classifier.

Speech Recognition (SR): Fanar enables natural interaction through speech – the most effortless and natural form of human communication. The orchestrator integrates a state-of-the-art Arabic-English Automatic Speech Recognition (ASR) system. The advanced ASR system supports – *(i)* multiple Arabic dialects, e.g., Egyptian, Gulf and Levantine; *(ii)* non-native Arabic accents; and *(iii)* diverse code-switching scenarios, including both dialectal variations within Arabic (e.g., Modern Standard Arabic (MSA) $\leftrightarrow$ Egyptian dialect (EGY)) as well as seamless transitions between English and Arabic (Ar $\leftrightarrow$ En). These capabilities collectively empower Fanar to accommodate dialectal Arabic speakers and foster inclusivity for non-native Arabic speakers. Details of ASR are provided in Section 9.

Text-to-Speech (TTS): To enable better accessibility, the Platform integrates Arabic and English text-to-speech systems. The TTS systems leverage Diffusion Transformer with ConvNeXt V2 (Chen et al., 2024b) for better text-speech alignment during in-context learning, without the extra modules like grapheme/phoneme alignment, duration predictor, text encoder, or any aid of codec for semantic information infusion. For details, see Section 9.

Image Generation (IG): The Platform provides support for image generation that is aligned for reflecting Arab and Islamic preferences. The Stable Cascade model is used as it has a much smaller latent space compared to the well known Stable Diffusion model and its variants and is optimized for both faster fine tuning and inference. In the current landscape of state-of-the-art (SOTA) image models, such as Stable Cascade, biases are evident when generating images from neutral prompts. These models predominantly depict elements of Western cultures, including people, cuisine, and scenery. Additionally, there is a notable lack of accurate representation when generating images related to Middle Eastern topics. This includes details such as culturally appropriate attire, diverse skin tones, and iconic regional landmarks. Our approach for fine-tuning image generation to reflect local cultural values is provided in Section 9 along with concrete examples.

Retrieval Augmented Generation (RAG): A RAG system retrieves relevant information from external data sources for a given input prompt which can then be passed as contextual information to the LLM (Lewis et al., 2020). By grounding the generated response on the provided context, it can help improve the accuracy of generated responses (Ram et al., 2023). Fanar currently provides four RAG systems for controlled content generation in specific domains. These are: (i) Attribution RAG for providing supporting evidence (references) for fact-related queries. For example, if the prompt is "What is the length of the river Nile?" the response will be validated against Wikipedia and corrected if there is a mismatch; (ii) Recency RAG for information that is post the checkpoint date of the pre-training corpus. As an example, for the prompt "What is the latest weather in Doha?" the system will extract information from selected verified websites and summarize the information; (iii) Islamic RAG provides content from authoritative websites for Islam related prompts; and finally (iv) Biography RAG for ensuring that accurate information is generated for well known people in the region and beyond. More details about these four RAG systems are provided in Section 10.

Translation: Fanar provides a specialized service for translation from Modern Standard Arabic (MSA) to other Arabic dialects and directly from English to the dialects. As parallel data in this space is highly limited, we fine-tune an existing sequence-to-sequence transformer model for dialectal translations. Benchmarking details are provided in Section 4.5.1. The translation systems build upon 15 years of expertise within QCRI in MSA and dialectal Arabic translations.

4. Pre-training Data

Pre-training data composition for Arabic, English and code is presented. Data filtering pipeline, especially for Arabic is described. The role of machine translation to expand coverage of Arabic data is highlighted.

Data is a critical building block for modern AI systems in general and for LLMs in particular. We describe the pre-training data composition for both English and Arabic and the use of machine translation to augment both MSA and dialectal Arabic data. Given the scarcity of Arabic data, our syntactic and semantic filtering and cleaning approaches are more nuanced compared to English.

4.1. Composition of the Fanar Pre-training Data

To pre-train Fanar, we curated a dataset comprising of 1 trillion tokens spanning Arabic, English, and computer code. The tokens were sourced from diverse origins, including web documents, scientific articles, encyclopedic entries, mathematical problems, books, news articles, and source code from common programming languages. The diversity of data is instrumental in enabling the model to exhibit robust performance across a wide array of tasks. Detailed distribution of the data sources for each domain are presented in Figure 2, while Table 1 outlines the token counts for these sources. Recognizing that different corpora contribute uniquely to model training, we aimed to balance corpora that enhance language understanding, facilitate knowledge reasoning, and improve task-specific performance. We employed rigorous preprocessing, including data cleaning, filtering, and deduplication, to ensure the quality of the data. The final data mixture was determined based on extensive ablation studies and are described in Section 5.
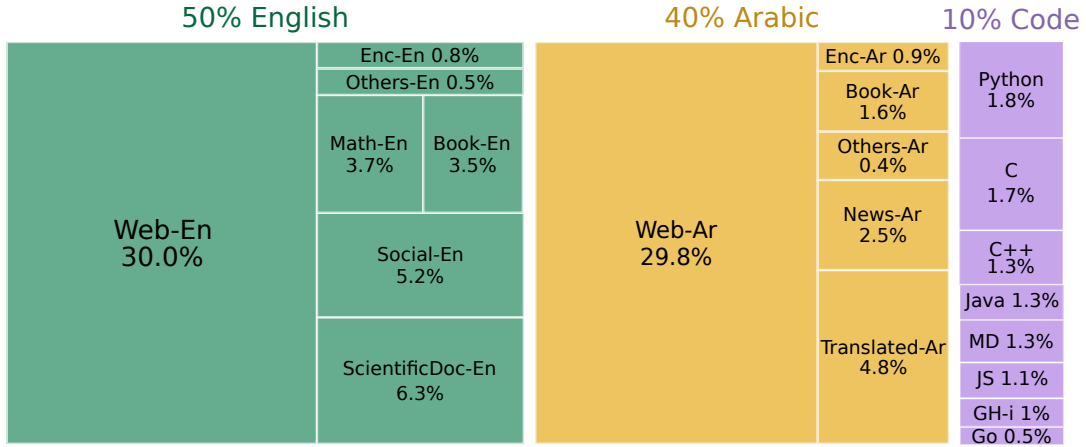
Figure 2: Composition of the Fanar pre-training data sources distributions.

4.1.1. English Data Composition

The English component of our pre-training dataset encompasses approximately 513 billion tokens, derived from a diverse range of sources many of which have been utilized in other representative LLMs. These include: (i) web documents from preprocessed Common Crawl sources, including C4 (Raffel et al., 2020), RefinedWeb (Penedo et al., 2023), DCLM (Li et al., 2024), Dolma (Soldaini et al., 2024) and RedPajama (Weber et al., 2024) ensuring broad web-based content representation, (ii) scientific documents derived from RedPajama-arXiv and PeS2o (Soldaini and Lo, 2023) datasets, (iii) social media data extracted from Pushshift Reddit dataset (Baumgartner et al., 2020) to capture conversational and informal language patterns, (iv) mathematical data from sources like Algebraic-Stack (Azerbayev et al., 2024) and Open-Web-Math (Paster et al., 2024) datasets to capture complex reasoning and mathematical language, (v) books from public domain book data from Project Gutenberg via Dolma corpus and (vi) encyclopedic content from Wikipedia dumps and MegaWika (Barham et al., 2023) for structured, factual information.

This rich and varied collection ensures a comprehensive representation of English-language data across multiple disciplines and styles.

Domain	Doc Source	Fanar Tokens (in Billions)
English	Web	307.8
	Scientific	64.6
	Social Media	53.4
	Math	37.9
	Books	35.9
	Encyclopedic	8.2
	Others	5.1
Arabic	Web	305.7
	Translated	49.3
	News	25.6
	Books	16.4
	Encyclopedic	4.1
	Others	9.2
Code	GitHub	102.6

Table 1: Fanar pre-training data includes ∼1T cleaned tokens in Arabic, English, and Code, sourced from diverse origins.

Figure 3: Code data composition in our pre-training data.

4.1.2. Arabic Data Composition

Recognizing the limited availability of high-quality Arabic pre-training data, we curated an extensive set of 410 billion Arabic tokens. The data spans multiple varieti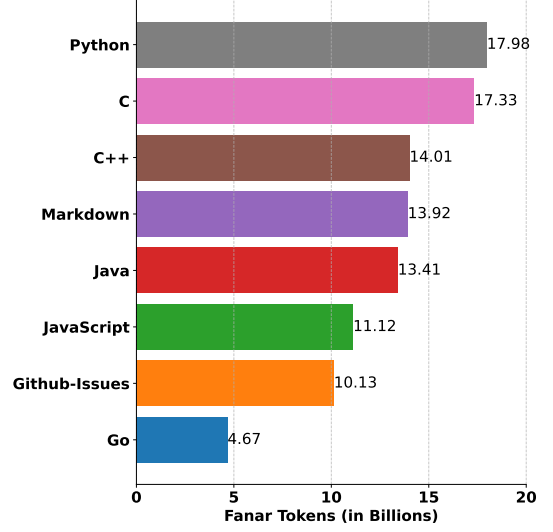es of Arabic, including MSA, Classical Arabic, and dialectal content encompassing a diverse range of sources: (i) web documents that were crawled in-house and preprocessed for quality, (ii) news articles and books spanning literature, religion, politics, culture and history, (iii) encyclopedic content from Arabic Wikipedia (iv) classical and contemporary Arabic poetry and (v) in-house machine translated books, STEM papers and encyclopedic documents to ensure English-Arabic language alignment.

4.1.3. Code Data Composition

Our code data subset comprises approximately 102 billion tokens, representing around 10% of the pre-training dataset. This subset was sourced primarily from The Stack (Kocetkov et al., 2023), a collection of permissively licensed GitHub projects across a large number of programming languages. We sub-selected data from common programming languages in The Stack, including Python, C, C++, Java, Go, and JavaScript. Additionally, we included Markdown and GitHub Issues data to provide contextual code understanding. Figure 3 presents the detailed code data composition.

4.2. Data Curation, Cleaning, and Standardization

The Arabic data (both web and curated) is collected from a multitude of sources, each having a different file format (e.g., txt, HTML, XML, JSON, zip) and text granularity (e.g., lines, paragraphs, articles, complete books, poems). We homogenize these varying formats into the format used by datasets such as Dolma (Soldaini et al., 2024). Files are collections of JSON records [3] with the following fields: **"id"** which serves as a unique identifier; **"text"** which contains the **core** text content at the granularity of the original data[4]; **"metadata"** which contains any additional information about the record such as creation

---

[3] For example, an article is considered as one record.

[4] Given a website, we automatically detected repeated headers, footers, and inner sections and links for each source, and removed them from all its articles to improve quality signals.

date and source URL, and **"quality_signals"** capturing a set of quality scores collected at the record level, which are described in more detail in Section 4.3. All text goes through a simple cleaning process where all HTML and JavaScript tags are removed and white spaces (including tabs, newlines, trailing escape characters) are normalized.

4.3. Data Quality and Filtering

As the data comes from a variety of sources, it contains a lot of noise and low quality content. We do not want to train the model on low quality data as it would adversely affect the quality of generation. As the data does not come with any pre-existing score of quality, we utilize a filtering pipeline that only retains high quality data based on their syntactic and semantic characteristics. This pipeline extends and tailors existing work on data filtering to the Arabic language.

4.3.1. Syntactic Filtering

Existing Arabic LLMs, such as **Jais** (Sengupta et al., 2023), apply hard-coded cutoffs based on heuristics to judge the quality of a given text and filter it out if it fails to pass certain thresholds (e.g, special symbols should not exceed 20% of the content). In our approach, we implemented 20 of the most widely used **quality signals** described in the **RedPajama** dataset[5]. These heuristic-based quality signals determine the quality of a given text. They cover a variety of measures such as the number of sentences, the number of words, the ratio of symbols and punctuation to words among others. We also removed records with insufficient amounts of Arabic.

We modified all quality signals to handle Arabic texts properly. Some examples include adding right-to-left punctuation marks and considering digits written in Arabic/Hindi alphabets, diacritics, ligatures, special symbols, Farsi and decorated characters. Other quality signals in RedPajama are handled as part of the deduplication and model-based filtering described in the next subsections.

Furthermore, rather than determining the cutoff thresholds for each of these quality signals in an ad-hoc manner, we utilize a systematic approach. For a given quality signal X, we divide X's score, which is typically between 0 and 1, into 10 histogram ranges with a fixed bin width of 0.1, and then distribute the dataset records based on their scores to one of the 10 buckets. Then, by manually investigating random hundred samples from each bucket, we make more informative decisions on the cutoffs to apply to all datasets. For instance, we observed that setting the threshold for the fraction of unique words in the content to 0.2 effectively identifies a significant portion of advertisement content due to its repetitive nature.

Existing web datasets, such as **C4** (Dodge et al., 2021), process CommonCrawl data using uniform filtering rules across *all* languages. We hypothesize that while these rules may be suitable for English, they may not be as effective for Arabic. For instance, one such filter excludes web pages containing fewer than three paragraphs, each with a minimum of 200 characters. Figure 4 illustrates a high-quality Arabic article that would be excluded by the aforementioned filter, as only one paragraph meets the criteria. To address this, we adjusted the filtering rules based on empirical observations from actual Arabic data, as previously described.

4.3.2. Semantic Filtering

To remove unwanted content from web data, we used the ASAD system (Hassan et al., 2021) to detect offensive and profane language, and adult content. We plan to increase the accuracy of hate speech detection in ASAD and use it to filter out hateful content. In addition, we sampled 20 articles from the most common domains from web data and gave them to annotators to estimate the quality and usefulness of each domain.

---

[5] https://www.together.ai/blog/redpajama-data-v2

صحة

وفاة جديدة بإنفلونزا الطيور في إندونيسيا

أعلنت مصادر طبية إندونيسية وفاة رجل من بلدة قريبة من العاصمة جاكارتا بعد أن ظهرت
عليه أعراض مرض إنفلونزا الطيور.

وقال متحدث باسم مستشفى سوليتني ساروسو في جاكارتا إن الرجل -وهو في التاسعة
والثلاثين من العمر- توفي بعد تلقيه العلاج ليوم واحد، مشيرا إلى أن فحوصات تجري حاليا
للتأكد من أن الوفاة كانت بسبب إنفلونزا الطيور أم لا.

وأضاف المتحدث أن هناك اشتباه بأن يكون المريض مصابا بالمرض بسبب تواجده المستمر
في مزارع الدواجن.

وفي حال تأكدت هذه الإصابة، يرتفع عدد الذين توفوا بإنفلونزا الطيور في إندونيسيا حسب
تأكيدات منظمة الصحة العالمية إلى 12 شخصا.

وقتلت سلالة "H5N1" من فيروس إنفلونزا الطيور أكثر من 70 شخصا في خمسة بلدان
آسيوية هي إندونيسيا وتايلند وفيتنام والصين وكمبوديا.

ومن المعروف أن الفيروس انتشر بين الدواجن في أجزاء من آسيا وأصاب الطيور في ثلثي
أقاليم إندونيسيا وهي أرخبيل مترامي الأطراف يتكون من حوالي 17 ألف جزيرة يسكنها 220
مليون نسمة.

Figure 4: High-quality Arabic article failed in a C4 filter. Only one paragraph (not three) has at least 200 characters.

4.3.3. Model-based Filtering

Recent progress on data preparation for LLM training (Li et al., 2024) has shown the advantage of data filtering using models that predict data quality. While our syntactic and semantic filters removed the majority of noisy data, there is some additional data such as Ads and SEO content that passed the first two filtering phases and require advanced filtering. We explored several approaches in this direction. We introduced two model-based filters that we used for the preparation of pre-training data:

- Perplexity filtering: We used KenLM models which are probabilistic n-gram language models for fast perplexity estimation (Heafield, 2011). These models are trained on Wikipedia content for several languages. For our perplexity filtering, we used a pre-trained model for Modern Standard Arabic (MSA)[6]. We assessed the perplexity distributions and defined a threshold per dataset to filter data with the highest 5% perplexity. In the last training epoch, an additional perplexity filtering is applied to remove low perplexity content. At that training stage, these documents are becoming too easy and do not give high learning opportunity for the LLM.

- Filtering using Education classifier: We build an education classifier following the approach introduced in FineWeb-Edu for English Web data (Lozhkov et al., 2024). In Fine-Web-edu-ar, a translation for FineWeb-Edu to Arabic is proposed. A small NLLB-500M model is used to translate FineWeb-Edu. The data quality in Fine-Web-edu-ar is poor due to the use of small Machine Translation model. Alternatively, we construct a native Web education Arabic dataset: First we sampled randomly 1M documents from our web corpus. We used an Qwen-2.5-73B-Instruct to annotate these documents for the classifier training (Qwen Team, 2024). The LLM is prompted to assign a score between 0 and 5 reflecting the richness of the document in educational content. We used the same annotation prompt as for FineWeb-Edu. Minor adjustment to the labeling prompt are made to accommodate the Arabic content. In order to train the classifier, we selected a multilingual embedding that is good for Arabic language (Chen et al., 2024a). We trained a classification head on top of the embedding to score the education level of web documents. The average accuracy of the classifier on a validation set is about 70% which is comparable to the results on English FineWeb-Edu classifier. We used the education classifier filter low education score with values 0 and 1. Inspection of the low education content revealed documents several unfiltered Ads and adult content. The filtered data correspond approximately to 20% of the data.

---

[6]https://huggingface.co/edugp/kenlm

4.4. Data Deduplication

Data deduplication is important in maintaining and controlling the high quality of the data. As such, the problem of scalable near duplicate detection and its benefit in training LLMs has been explored extensively. (Lee et al., 2022; Broder, 1997; Logasa Bogen et al., 2013; Elmagarmid et al., 2006) In Fanar, we apply two types of data deduplication, namely *exact-match dedup*, and *approximate-match dedup*. Additionally, we perform URL deduplication on all web-based data. We implemented a pipeline that can scale well to large datasets under limited compute and memory. We perform both inter-dataset and cross-dataset deduplication.

Exact-Match Deduplication: We implement a hashmap based approach to put similar objects in the same bucket. We control the number of buckets such that each one on average is around 10GB to 20GB and peak memory usage does not exceed 1TB. Then, in a parallel fashion, each bucket is processed by sorting its records and eliminating duplicates. We then reconstruct the files in the same order as the original ones using a merge-sort operation utilizing the unique id of each record.

Approximate-Match Deduplication: Exact-match deduplication, although computationally efficient, it has its own limitations. For the approximate-match deduplication (also known as fuzzy deduplication), we adopt the same approach used in other datasets (Shen et al., 2024; Tokpanov et al., 2024; Brown et al., 2020; Zeng et al., 2021), which is the min-wise locality sensitive hashing LSH technique (Broder, 1997). We experimented with various parameter configurations, and converged to the following parameters, *gram size* is set to 8, the *number of bands (b)* is set to 12, and *band length (r)* is set to 11 which results in the approximated *Jaccard similarity threshold* being 0.8 and *signature length* as 132. The algorithm is highly scalable and we used 350 CPU cores with a peak memory usage of 1TB and completed in about 12 hours.

4.5. Machine Translation

Leveraging over a decade of advancements in Arabic NLP, QCRI has consistently set benchmarks in machine translation and resource creation for Arabic and its underrepresented dialects. Our Shaheen machine translation system (Sajjad et al., 2017b) exemplifies this legacy, offering a state-of-the-art precision in translating between English and Modern Standard Arabic (MSA). This expertise enabled us to tackle the dual challenge of creating robust datasets and training state-of-the-art Fanar LLM addressing both linguistic and cultural nuances.

Large Language Models (LLMs) have demonstrated impressive capabilities, primarily due to their extensive size and the diversity of the data they are trained on. However, this advantage is heavily skewed towards high-resource languages like English, leaving low-resource languages, such as Arabic, at a significant disadvantage. To bridge this gap, researchers and practitioners have increasingly turned to synthetic data generation methods, with Machine Translation (MT) emerging as a prominent approach. By translating existing English datasets into low-resource languages, MT facilitates the creation of larger and more diverse datasets, thereby enhancing the ability and performance of LLMs for these underrepresented languages. This approach not only expands linguistic resources but also enables access to rich knowledge repositories, including scientific literature, encyclopedias, and other genres that are predominantly available in English, enriching the breadth and depth of training data for LLM.

4.5.1. English-to-MSA Translation

To initiate our exploration of MT systems, we benchmarked several open-source and commercial solutions, including state-of-the-art models such as **shaheen** (Sajjad et al., 2017b), **mbart** (Liu et al., 2020), **madlad400** (Kudugunta et al., 2023), **helsinki** (Östling et al., 2017), **nllb 3.3B** (Team et al., 2022), **gpt4** (OpenAI et al., 2024), and **S-T5** based on Ara-T5 (Nagoudi et al., 2022a).[7] These systems were evaluated using AraBench (Sajjad et al., 2020), which provides a diverse range of test sets spanning

---

[7]The Ara-T5 system does not perform translation out-of-the-box, we fine-tuned it a mixed domain of 515K sentences (Joty et al., 2015).

multiple domains. The evaluation revealed that different systems excelled in specific domains (results summarized in Table 2). Consequently, we selected three systems: shaheen, nllb, and S-T5 for translating specialized content across diverse domains.

Table 2 Bench-marking English-to-MSA Machine Translation Systems

Test Name	shaheen	mbart	madlad400	helsinki	nllb	gpt4	S-T5	Category
bible	4.6	3	**7.9**	4.3	6	6.3	5.1	Religious
iwslt.11	**14**	10.9	12.3	11.6	12.5	10.9	12.1	
iwslt.12	15.7	14.4	16.4	15.4	**16.9**	14.9	16	Spoken
iwslt.13	16.1	15.6	**19.5**	16.9	**19.5**	16.6	18.5	
iwslt.14	13.9	13.4	15.7	13.3	**16.5**	13.1	14.9	
news.04	24.2	9.7	22.1	19.8	21.6	19.1	**32.6**	
news.05	25.3	7.7	21.3	18.4	21.8	17.9	**36.6**	News
news.test06	15.5	5.7	12.8	10.1	12.5	12.4	**20.4**	
ldc_web_eg.test	**6.3**	4.7	5.5	2.2	5.6	5.6	5.4	
summa-AJ	20.9	10.9	19.2	18	20.8	20.8	**24.7**	General
summa-BBC	19.3	13.7	20.3	16.5	20.8	20.8	21.5	
qed_1	**24.7**	7.8	6.4	14.6	18.7	7.8	6.7	Education
qed_2	**19.1**	9.6	6.3	11.8	17.4	11.2	8.5	
travel	**21.1**	12.7	14	13.3	16.1	17.9	14.9	Travel
mayo	13.6	11.9	20.4	14.5	22	**20.7**	12.4	Health
Average	**17.0**	10.1	14.7	13.4	**16.6**	14.4	**16.7**	

Table 3 Human evaluation scores for English-to-MSA translation. Annotators were allowed to mark multiple best systems for each sample, or explicitly mark if none of the translations were good. The scores indicate the percentage of samples for which a system was in the best translations list.

	shaheen	nllb	S-T5	No Best System	Total Samples
Books	**53.36%**	50.59%	32.81%	11.86%	253
STEM Papers	28.05%	**58.54%**	14.63%	18.29%	82
Wiki Encyclopedia	28.26%	**68.48%**	31.52%	10.87%	92

We chose three specific genres, namely Books, STEM Papers, and Wiki Encyclopedia, for translation from English to MSA. To determine the most suitable system for each genre, we selected a small sample from each category and translated them using our chosen systems. Subsequently, we conducted a human evaluation on this subset to identify the best-performing system for each genre. In addition, we calculated COMET scores (Rei et al., 2020) as an additional evaluation metric. The evaluations are presented in Tables 3 and 4. Based on the outcomes, we opted for **Shaheen** for translating books[8] and the **Nllb 3.3B** model for translating STEM papers and Wiki Encyclopedia.

Table 4 COMET evaluation scores for English to MSA translation

	shaheen	nllb	S-T5
Books	0.7219	**0.7301**	0.6865
STEM Papers	0.6829	**0.7664**	0.6562
Wiki Encyclopedia	0.7143	**0.7735**	0.6654

4.5.2. MSA-to-Dialect Translation

Despite its rich diversity, dialectal Arabic remains significantly underrepresented in Large Language Models (LLMs). To address this gap during the development of Fanar, we developed English-to-Dialectal Machine Translation models, complemented by human post-editing, to create robust evaluation benchmarks. These benchmarks facilitate translation between Modern Standard Arabic (MSA) and two major

---

[8]Note that although the COMET scores favored the **Nllb** system for all genres, we ultimately selected **Shaheen** based on human evaluation.

dialects: Egyptian (Egy) and Levantine (Lev) Arabic. By extending MSA-based resources like ArabicMMLU to dialectal contexts, we provide valuable tools for assessing LLMs' comprehension of dialectal Arabic.

We fine-tuned two machine translation models: **AraT5** (Nagoudi et al., 2022b) and **NLLB** (Team et al., 2022) and experimented with several variants of these models, with sizes ranging from 600M to 3.3B parameters. In our preliminary experiments, we found the NLLB 3.3B model to surpass AraT5 and its smaller variants, post fine-tuning with dialectal data. We carried out ablation studies using different data mixtures on the NLLB 3.3B model. We shortlisted three systems per dialect using BLEU scores as our primary criterion and used human evaluation to select the best system for each dialect. Table 5 provides a summary of the evaluation results across various dialectal test sets within the community. More details on this can be found in (Mousi et al., 2024).

Table 5 SacreBLEU scores for MSA-to-LEV and MSA-to-EGY models. *S1* to *S4* represent different configurations: **MSA-to-LEV**: *S1* = UFAL, *S2* = +LDC, MADAR, PADIC, D2M, *S3* = +LDC, MADAR, PADIC, D2M, *S4* = GPT4 zero-shot. **MSA-to-EGY**: *S1* = MADAR + D2M + LDC, *S2* = MADAR + D2M + LDC + Arzen, *S3* = MADAR + D2M + LDC + Arzen + BOLT, *S4* = GPT4 zero-shot.

	MSA-to-LEV Models				MSA-to-EGY Models			
	OSACT	SADID	LDC	D2M	ARZEN	D2M	LDC	MADAR
S1	9.8	12.7	6.2	11.0	1.8	57.3	11.8	17.7
S2	9.8	11.8	6.3	11.7	17.3	57.2	12.3	17.6
S3	9.7	11.8	7.0	47.8	15.8	55.0	11.2	17.9
S4	5.92	8.42	3.46	4.89	1.88	7.53	2.83	6.02

---

5. Tokenization

> The importance of tokenization in LLMs is explained. The limitations of BPE type tokenizers for Arabic are explained. A new morphologically aware tokenizer algorithm is introduced.

Tokenization is a foundational step in any natural language processing (NLP) pipeline, segmenting text into tokens such as bytes, characters, subwords, words, or multi-word units. The quality of tokenization directly impacts downstream tasks, as errors can propagate through the pipeline, ultimately degrading the performance of downstream applications (Sajjad et al., 2017a; Adel et al., 2018). Tokenization has a rich history in NLP, with methods ranging from simple whitespace splitting to advanced statistical and neural approaches (Smit et al., 2014; Otani et al., 2020). Tokenization plays a critical role in Large Language Models (LLMs), influencing their efficiency, context length, and even their precisions (Dagan et al., 2024). While tokenization-free approaches also exist in LLM research as an alternative (Clark et al., 2022; Deiseroth et al., 2024), most successful models (e.g., Gemma, LLaMA, and the OpenAI GPTs) continue to rely on Byte Pair Encoding (BPE)-based tokenizers, along with its underlying assumptions and limitations.

5.1. Byte-Pair Encoding and its Limitations

BPE, originally introduced as a traditional text compression algorithm (Shibata et al., 1999), was first proposed for use in machine translation in 2016 as a text tokenizer (Sennrich et al., 2016). Since then, it has been widely adopted in NLP and LLMs due to its efficiency in managing vocabulary size, handling out-of-vocabulary words, prioritizing frequent patterns, and, to some extent, improving upon morphology-based tokenizers (Sennrich et al., 2016). Despite its widespread success, Vanilla BPE has notable limitations: (i) its greedy algorithm, (ii) inefficiencies in cross-lingual settings where similar words may use different character variations, and (iii) amount of character information is not equal in different languages. These shortcomings have spurred modifications, such as BPE dropout (Provilkov et al., 2020), sampling-based BPE (Asgari et al., 2019, 2020), byte-level extensions (Wang et al., 2020), multilingual BPEs (Liang et al., 2023).

5.2. Challenges of Byte-Pair Encoding and Arabic

Arabic Morphology and BPE: The additive nature of BPE makes it well-suited for suffixing languages like English. However, languages such as Arabic present unique challenges due to their root-and-pattern morphology, complex derivational systems, and their classification as primarily infixing languages (McOmber, 1995; Versteegh, 2014). Consequently, traditional BPE and byte-level approaches (Sengupta et al., 2023) often fail to effectively capture the intricate morphological structures of Arabic, underscoring the need for more advanced tokenization strategies tailored to the linguistic properties of such languages.

Analyzing the output of vanilla BPE on Arabic text, we observed that Arabic's morphological structure, characterized by its infixes and root-pattern system, is not well-suited to BPE. As a result, tokens are often segmented in a morphologically meaningless manner, introducing unnecessary ambiguity. For instance, the word الرحمن (Al-Rahman, "The Merciful") is segmented into من (min, "whom") + ال (al, "the") + رح (rah, an incomplete fragment). Here, من (min), a frequent token in Arabic, is semantically unrelated to the original word الرحمن. This segmentation forces the model to disambiguate these unrelated components, complicating the learning of meaningful embeddings. On the other hand, purely morphological segmentation in language models has also shown limitations, as it does not align with the frequent patterns present in natural language usage (Durrani et al., 2019).

Byte-level vs. Char-level Tokenizer: Since Arabic script characters typically require more than one byte, the use of byte-level BPE (BBPE) is inefficient and demands a larger number of merging steps. Moreover, byte-level patterns fail to preserve character similarity in many cases. Unlike English, where accented characters often share a byte with the base character, Arabic's encoding structure exacerbates this inefficiency. Similarly, adopting character-level tokenization approaches, such as those used by OpenAI, is also inefficient, particularly during text generation, and will limit the context-size.

5.3. Fanar Morphology-based Tokenizer

This observation motivated us to design a tokenization approach that combines the strengths of morphological segmentation and the statistical efficiency of byte-pair encoding (BPE), resulting in the Fanar morphologically aware tokenizer, MorphBPE [9]. The core idea is to align the BPE algorithm with the morphological structure of Arabic (or other morphologically rich languages) by modifying it to respect morpheme boundaries during the token merging process.

In the MorphBPE algorithm, we start by initializing the vocabulary with individual characters from the text. The training corpus is then segmented using morphological segmentation, ensuring that the structural morphemes are identified before applying any statistical operations. As shown in Algorithm 1, the algorithm iteratively computes byte-pair frequencies and selects the most frequent byte-pair merge candidate. However, unlike standard BPE, MorphBPE includes a modified step (line 5 in Algorithm 1), which ensures that merges do not cross morpheme boundaries. This modification preserves the morphological integrity of the text while enabling statistical efficiency during tokenization.

The iterative process continues until the desired vocabulary size is reached. At each step, the vocabulary is updated to include the newly merged tokens, balancing the morphological structure of the language with statistical considerations for improved representation in language models.

By respecting the morphological structure during tokenization, MorphBPE addresses the inefficiencies and ambiguities of standard BPE when applied to morphologically rich languages like Arabic. This approach enhances the ability of language models to learn meaningful embeddings that better capture linguistic nuances.

---

[9]U.S. provisional patent application number: 63/679,403.

Algorithm 1 Morphologically-aware BytePair Encoding (MorphBPE)

1: Initialize vocabulary with individual characters
2: Segment the training corpus using morphological segmentation
3: **while** number of merges < desired vocabulary size **do**
4:     Compute byte-pair frequencies
5:     **Modified Step:** Find the most frequent byte pair that does not cross morpheme boundaries
6:     Merge the most frequent byte pair into a new symbol
7:     Update the vocabulary with the merged symbol
8: **end while**

5.4. Tokenization Evaluation

Tokenization evaluation can be conducted using **intrinsic** or **extrinsic** metrics. Below, we discuss key intrinsic metrics and their applications:

(i) Fertility: Fertility measures the ratio of the number of tokens produced by a tokenizer compared to a baseline tokenizer, typically one that uses whitespace splitting. A lower fertility score is often interpreted as indicative of a better tokenizer, as it suggests more efficient representation. However, this argument can be disputed, particularly for agglutinative languages like Turkish, where meaningful representation for large language models (LLMs) requires more tokens to capture the underlying morphological structure and ensure sufficient context for each surface form.

(ii) Perplexity: Perplexity measures the likelihood of a held-out text for a trained model is another commonly used intrinsic evaluation metric. However, comparing perplexity across different tokenizers is valid only if their vocabulary sizes are the same. Otherwise, differences in vocabulary size render the comparisons non-equivalent.

(iii) Morphological Alignment Score (Proposed Metric): We propose a new intrinsic evaluation metric, the **Morphological Alignment Score**, which assesses how well the tokenization aligns with the underlying morphological segmentation of words. To calculate this, we use a pairwise alignment score based on dynamic programming, ensuring that the order of matching tokens with segmented morphemes is preserved. This method provides a quantitative measure of how effectively a tokenizer respects the morphological structure of the language.

5.5. Fanar Tokenizer Preprocessing and Training

Preprocessing is a crucial step in the BPE process to ensure accurate frequency counts of character patterns. We designed an extensive and peer-reviewed preprocessing pipeline for Arabic script-based languages, including Arabic, Persian, Urdu, and others, to normalize all related scripts into a standard form. As part of this process, we remove diacritics from Arabic text, as most data is not diacritized. However, diacritic characters are still included in the tokenizer vocabulary to allow enhancement during supervised fine-tuning (SFT). To maintain the integrity of the preprocessing, all data has been converted into the HuggingFace format.

The Fanar Tokenizer was trained on the complete Arabic dataset used for model development. By adhering to a vocabulary size that is a multiple of 1024, the tokenizer aligns with modern hardware architectures, such as GPUs and TPUs, which optimize processing in 1024-sized blocks (PS et al., 2024). This alignment improves throughput and reduces latency by enabling efficient token batch processing.

Drawing from LLaMA's use of a 32K vocabulary size for English and code (Lim and Lauw, 2023), we identified optimal morphological alignment at 45K tokens for Arabic. To ensure both efficiency and flexibility, we combined 32K English tokens with 45K Arabic tokens, pruned infrequent merges, and finalized a vocabulary size of $75 \times 1024 = 76,800$. This size accommodates reserved tokens for multimodality, diacritics, and other special cases.

5.6. Fanar Tokenizer Evaluation Results

The evaluation of the Fanar Tokenizer was conducted across the mentioned metrics, including fertility, morphological distance, and perplexity, comparing vanilla BPE with Morphological BPE. Figure 5-(i) shows that the Fanar Morph Tokenizer achieves a lower training loss compared to vanilla BPE, demonstrating its efficiency and faster convergence. Figure 5-(ii) highlights that the Fanar Morph Tokenizer attains the highest alignment with morphology while maintaining reasonable fertility. These results illustrate that Morphological BPE not only preserves morphological structure but also improves model performance, reducing perplexity loss and accelerating convergence across various model sizes.
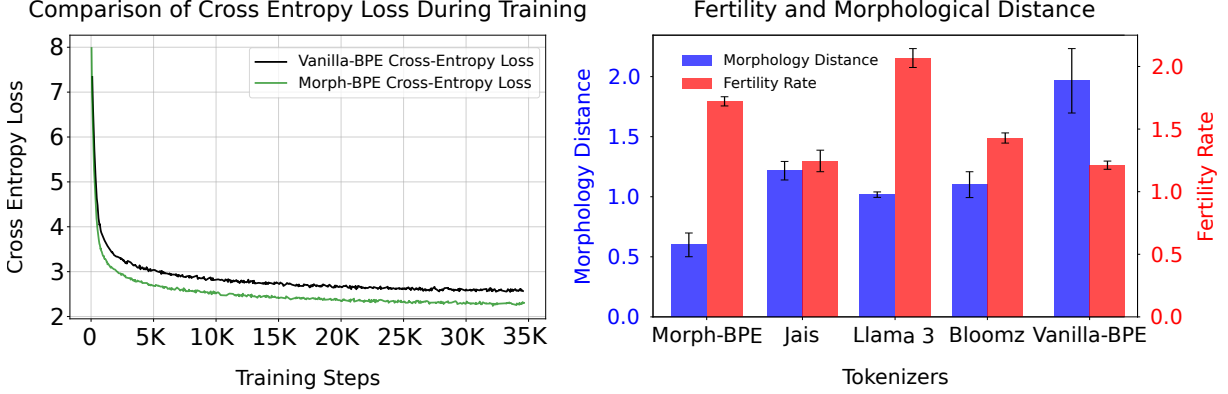

Figure 5: Intrinsic evaluation of Fanar Tokenizer (Morph-BPE). From left to right: **(i)** Cross-entropy loss comparison between Morph-BPE and vanilla-BPE, Morph-BPE demonstrates faster convergence and smaller loss than vanilla-BPE for the same vocabulary size. **(ii)** Comparison of fertility and morphological alignment among existing Arabic and Multilingual Tokenizers.

6. Modeling and Pre-training

> The pre-training recipe for both **Fanar Star** and **Fanar Prime** is detailed. The multi-epoch and phased approach for language data mixtures and curriculum training are highlighted.

We describe the model architectures of both **Fanar Star** and **Fanar Prime** our choice of training stages and our choice of data mixture. We also discuss our ablation studies that informed these choices.

6.1. Model Architecture

Both **Fanar Star** and **Fanar Prime** are refined versions of the classical decoder-only Transformer architecture. They respectively have 7.1B and 8.78B parameters (Vaswani et al., 2017). **Fanar Star** reuses the architecture of OLMo (Groeneveld et al., 2024) and the Llama family (Touvron et al., 2023) (trained from scratch) while **Fanar Prime** is built upon the Gemma-2-9B base model (Riviere et al., 2024) (continually trained). Table 6 compares the architectures. The vocabulary size of **Fanar Star** is 76,800 while that of **Fanar Prime** is 128,256. The vocabulary of **Fanar Prime** is pruned down from the original 250,000. It is also notable that the embedding dimensionality of **Fanar Star** is larger (4096) compared to that of **Fanar Prime** (3584).

6.2. Ablation Studies

We ablated key design choices related to data filtering and data mixture composition on 1B-parameter models. We used 50 to 100 Billion tokens to train a model for each configuration of interest. We

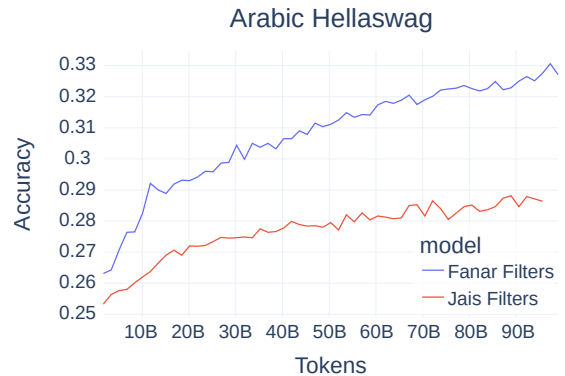**Table 6** Overview of the model configuration of Fanar Star and Fanar Prime.

	Fanar Star	Fanar Prime
Num Layers	32	42
Attention Heads	32	16
Model Dimension	4096	3584
Hidden Size	4096	3584
Intermediate Size	11008	14336
Pre-Normalization	RMSNorm	RMSNorm
Post-Normalization	None	RMSNorm
Positional Embeddings	RoPE	RoPE
Attention Variant	Full	GQA
Biases	None	None
Activation	SwiGLU	Approximated GeGLU
Context Length	4096	4096
Batch Size (samples)	1344	1071
Batch Size (tokens)	~5.5M	~4.4M
Vocab size	76,800	128,256
Weight Tying	False	False
Embedding Parameters	314M	459M
Non-embedding Parameters	6.79B	8.32B
Total Parameters	**7.1B**	**8.78B**

then benchmarked the resulting models on multiple Arabic-translated tasks, including HellaSwag, Open-BookQA, PIQA, and BoolQ. While the results were consistent across benchmarks, we report statistics on HellaSwag as a representative. We used the best performing design configurations to train our 7B-parameter model. The design choice of the tokenizer was already described in Section 4.

6.2.1. Comparing Data Filtering Strategies

We compared two data filtering approaches: (i) Our in-house Fanar filtering recipe (detailed in Section 4.3) and (ii) The data filtering methodology from the Jais model (Sengupta et al., 2023).

Figure 6 compares a few data filtering recipes. The Fanar filtering recipe achieves and improvement of about four points on the Arabic HellaSwag benchmark compared to models trained with the Jais data filtering recipe. Notably, models trained with the Jais-filtered data exhibited earlier performance plateauing, whereas those trained on Fanar-filtered data showed a consistent upward performance trajectory. The improvement in our filtering approach can be attributed to a multi-stage filtering process, including perplexity-based filtering and quality classification.

Figure 6: Ablation results on HellaSwag. Fanar and Jais filtering recipes are applied separately on the data and 1B-parameter models are trained on them.

6.2.2. Comparing Data Mixture Composition

We investigated different Arabic-to-English token ratios which we present in Figure 7. The 70:20 Arabic, English ratio (red line) improved Arabic HellaSwag performance by two points. However, this adjustment led to a 6-point degradation in English benchmark performance. On the other hand, 30:60 Arabic, English ratio (blue line) yielded higher English performance but at a cost to Arabic benchmarks. Given the disproportionate degradation in English performance compared to the Arabic gains, we adopted a dynamic data-mixing strategy: during the initial

phase of pre-training, we kept the English tokens ratio higher and we made progressive adjustments by increasing the Arabic tokens ratio towards the end of the training.

Figure 7: Ablation results on data mixture composition. 1B-parameter models were trained on different data mixtures.

6.3. Fanar Star Pre-training

6.3.1. Training Recipe

Fanar Star was pre-trained using a two-stage curriculum approach specifically designed to address the challenges of limited Arabic language data while maintaining robust multilingual capabilities. Our training recipe comprises a multi-epoch pre-training phase followed by a cool-down phase. This design leverages both breadth and depth in data utilization, ensuring optimal model performance across diverse linguistic and domain-specific tasks. The details of each phase are outlined below.

- **Stage-1: Multi-Epoch Pre-training Phase.** We trained Fanar using for four epochs. This is aligned with findings by (Muennighoff et al., 2023), which demonstrate that training data can stay fresh for four epochs. During the first two epochs, the training data comprised a consistent token composition of 40% Arabic, 50% English, and 10% Code. This initial training phase aims to establish a broad cross-lingual foundation. For the last two epochs, additional filtering was applied using an education classifier (see Section 4.3.3 for details), resulting in a 20% reduction in token volume. Concurrently, we adjusted the data mixture to prioritize Arabic language proficiency by increasing its proportion to 50%, while reducing the English token share to 40% and maintaining 10% for Code tokens. The results provided in this report for Fanar Star are based on completed three epochs of pre-training. The training of the fourth epoch is in progress.

- **Stage-2: Cool-Down Phase.** Recent studies suggest that training large language models on a subset of high-quality data in the final stages of pre-training significantly enhances their downstream task performance (Blakeney et al., 2024; Dubey et al., 2024). Inspired by this, the cool-down phase of our training recipe involved curating a high-quality data subset, comprising approximately 100 billion tokens from carefully selected Arabic and English sources. In this phase, the model continued training on the curated dataset, with the learning rate linearly annealed to zero, starting from the final learning rate of the multi-epoch pre-training phase.

Using this recipe, Fanar Star was pre-trained on a total of ~3 trillion tokens. The training loss trajectory of Fanar Star during multi-epoch pre-training stage is presented in Figure 8. The loss and perplexity curves decrease consistently which indicates the effective learning throughout the multi-epoch training stage. Furthermore, the model's performance on the Arabic MMLU benchmark, as shown in Figure 9, demonstrates progressive improvements across training phases, confirming the efficacy of our pre-training strategy.
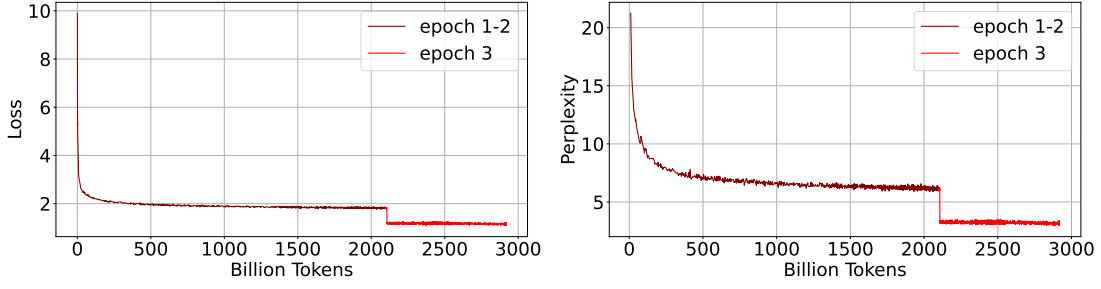
Figure 8: Training loss and perplexity Curves. The additional filtering applied in epoch 3 is reflected in the reduction of loss and perplexity.
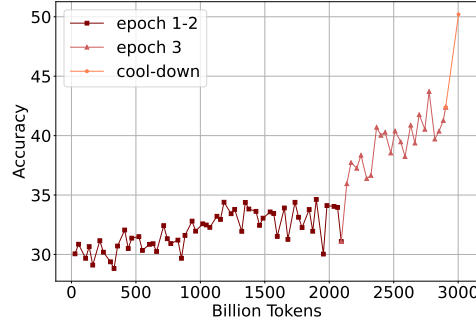

Figure 9: 3-shot accuracy of **Fanar Star** model during pre-training on Arabic MMLU (Koto et al., 2024). The rate of improvement is noticeable in epoch 3 after applying additional data filtering. Cool-down phase has also given a strong boost in downstream performance.

6.3.2. Optimization Configuration

In both stages, Fanar Star was trained using the standard auto-regressive language modeling objective, with a fixed context length of 4096 tokens. We used AdamW optimizer (Loshchilov and Hutter, 2019). The details of our training configuration is presented in Table 7. Our pre-training process utilized `bfloat16` mixed precision to enhance computational efficiency, while critical all-reduce gradient operations were performed in `fp32` to maintain numerical stability. The global batch size was set to 1344 samples, totaling ∼5.5 million tokens per optimization step. The learning rate was managed through a two-phase scheduling approach. Initially, we employed a warm-up stage spanning 2000 steps, during which the learning rate was linearly increased to a maximum value of 3×10^{-4}. Following the warm-up phase, a cosine annealing schedule was used to progressively reduce the learning rate to 3×10^{-5} over the course of Stage 1. In Stage 2, the cool-down phase, the learning rate was linearly reduced to zero, effectively training the model on the curated high-quality dataset.

Table 7 Overview of pre-training hyperparameters of Fanar Star and Fanar Prime.

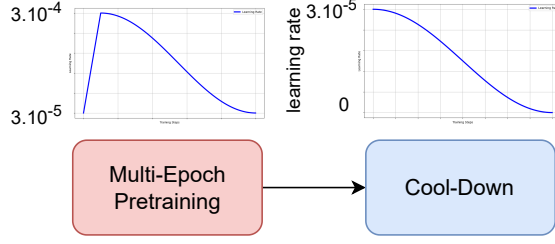|	Fanar Star	Fanar Prime
Warmup Steps	2000	100
Peak LR	3×10^{-4}	8×10^{-6}
Minimum LR	3×10^{-5}	1×10^{-6}
Optimizer	AdamW	AdamW
β_1	0.9	0.9
β_2	0.95	0.95
ϵ	1×10^{-5}	1×10^{-8}
Weight Decay	0.1	0.01
LR Schedule	cosine	cosine
Gradient Clipping	1.0	1.0
Gradient Reduce dtype	fp32	bfloat16
Optimizer State dtype	fp32	bfloat16

Figure 10: Pre-training phases: 1) multi-epoch training: training for 4 epochs over approximately 4T tokens. 2) cool-down phase: training with high-quality data with a decaying learning rate to zero.
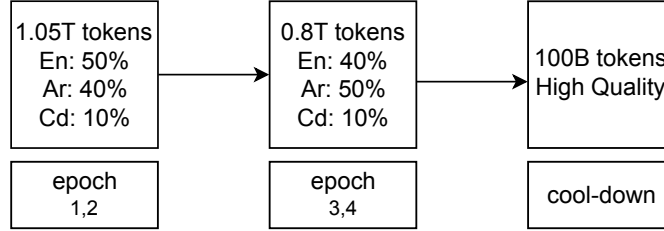

Figure 11: Curriculum learning in pre-training. phases. In the last epoch, additional model-based filtering are applied. The cool-down phase contains high-quality data such as books, encyclopedia, cultural and STEM related materials. Also conversational data and Wikipedia knowledge restructured in multiple-choice questions to boost benchmarking capability.

6.4. Continual Pre-training for Fanar Prime

6.4.1. Training Recipe

Fanar Prime is built through the continual pre-training of the Gemma-2-9B-Base model (Riviere et al., 2024), a robust multilingual foundation model initially pre-trained on eight trillion tokens through knowledge distillation from a larger, undisclosed model developed by Google. Although specific details regarding the training corpus of Gemma-2-9B are limited, its multilingual capabilities and extensive pre-training make it an ideal candidate for further adaptation.

As part of this adaptation, we performed vocabulary pruning on the original 250,000-token vocabulary of Gemma-2-9B, reducing it to 128,256 tokens for **Fanar Prime**. This reduced model size by decreasing the total parameters from 9.2 billion to 8.78 billion, enhancing computational efficiency without compromising performance.

Similar to the pre-training from scratch in **Fanar Star**, the continual pre-training process for **Fanar Prime** followed a two-stage curriculum strategy, utilizing a similar balanced data composition of 45% Arabic, 45% English, and 10% code. Since we started from a competent model, we chose to slightly alter the data composition as compared to **Fanar Star**. Unlike the training strategy employed for **Fanar Star**, **Fanar Prime** underwent a single-epoch pre-training phase, followed by a cool-down stage designed to enhance generalization. The results presented in this report are for a checkpoint around the 600B token mark, while the training is continuing to complete the full epoch. The cool-down stage utilized the same high-quality dataset curated for **Fanar Star**, ensuring consistency and alignment with the overarching training objectives. Through this training recipe, **Fanar Prime** was continually pre-trained on a total of 650 billion tokens.

6.4.2. Optimization Configuration

We reused the configuration of **Fanar Prime** to train **Fanar Star** with a few exceptions. The global batch size was set to 1071 samples, totaling ∼4.4 million tokens per optimization step. The learning rate

was first linearly increased to 8×10^{-6} for 100 steps, and the cosine annealed down to 1×10^{-6}. The hyperparameters for **Fanar Prime** are detailed in Table 7.

6.5. Pre-training Infrastructure and Frameworks

The pre-training of both **Fanar Star** and **Fanar Prime** was conducted on a 168 NVIDIA H100 80GB SXM5 GPUs, distributed across 21 nodes with 8 GPUs per node. These GPUs, powered by the NVIDIA Hopper architecture, are interconnected within each node via NVLink and NVSwitch, providing a bidirectional GPU-to-GPU bandwidth of 900 GB/s (450 GB/s in each direction). Inter-node communication was facilitated through high-speed InfiniBand connections, ensuring efficient data transfer across the cluster.

For the pre-training of Fanar Star, we utilized the OLMo framework[10], which is renowned for its advanced features and fine-grained control over training processes. This framework offers detailed artifact generation, including visualizations of layer weight dynamics, in-loop evaluation mechanisms, and back-tracing capabilities for training batches. These functionalities were invaluable for debugging, particularly in diagnosing and addressing loss spikes during our pre-training. For instance, the back-tracing feature allowed us to pinpoint problematic training batches and inspect their raw text inputs, helping us identify and resolve dataset issues such as improper language filtering, thereby ensuring cleaner and more balanced training data.

Additionally, we implemented custom in-loop evaluation pipelines tailored to specific downstream tasks. This enabled us to monitor performance trends during training, providing early insights into the model's generalization capabilities. To streamline the data preparation process, we leveraged the Dolma pipeline[11] —a flexible and scalable toolbox designed for large-scale language model data curation and preprocessing.

For the continual pre-training of **Fanar Prime**, we adopted the LitGPT framework[12]. This framework enabled seamless integration with the Gemma-2-9B-Base model, supporting efficient and scalable continual learning processes. Its modular design allowed us to focus on domain-specific optimizations, ensuring that the pre-trained model adapted effectively to new data distributions without compromising its foundational knowledge.

7. Post-Training

> Supervised Fine-Tuning (SFT) and preferential learning to accentuate Arabic cultural and safety alignment distinguish the post-training phase.

Our post-training objective was to develop a model that could effectively follow general instructions and engage users in meaningful conversations. Two additional key considerations guided our development process. First, the model needed to demonstrate strong Arabic proficiency to effectively serve our target user base. Second, it was essential to ensure that the model's responses were not only helpful and harmless but also culturally aligned. To accomplish this, Fanar's post-training process combined supervised fine-tuning with preference learning. The data needed to guide these training steps were acquired either through curating data from public sources and or generation by our post-training team over the course of training using a number of open models and also Fanar.

Our post-training strategy distinguishes itself from existing techniques through three key steps:

- **Sample-Level Data Quality Validation:** A capability-focused filtering process to curate a high-quality, bilingual supervised fine-tuning dataset from public sources.

---

[10]https://github.com/allenai/OLMo
[11]https://github.com/allenai/dolma
[12]https://github.com/Lightning-AI/litgpt

- **Multi-Stage Training Workflow:** A progressive training approach that incorporates high-quality, task-representative data samples at multiple stages. This process mirrors the cool-down and annealing stages commonly used in pre-training, to reinforce the model's ability to handle diverse tasks effectively.

- **Value-Aligned Synthetic Data Generation:** A data generation method tailored to align with Arabic cultural and religious values, addressing the challenge of developing a value-aligned preference/reward model.

The two key decisions we needed to address during training were how to generate Arabic instruction and preference data and how to balance the English and Arabic post-training datasets. While several recent models have demonstrated that synthetically generated data can be effective for creating instruction datasets (Adler et al., 2024; Dubey et al., 2024; Gunter et al., 2024; Lambert et al., 2024), the lack of open models with strong Arabic proficiency initially led us to rely on translating English data, particularly for core capability datasets. To enhance and evaluate Arabic proficiency, we tested several open models and identified instruction-tuned models that demonstrated significant reliability in Arabic. These included `Gemma-2-27B-it` from Google (Team et al., 2024), `Qwen2.5-72B-Instruct` from Alibaba (Qwen Team, 2024), `c4ai-command-r-plus` from Cohere (Cohere For AI, 2024), and `Llama-3.1-70B-Instruct` and `Llama-3.1-405B-Instruct` from Meta (Dubey et al., 2024). We leveraged these models for tasks such as filtering, judging, and data generation, ensuring compliance with the permissiveness of their respective licensing terms.

Balancing the language composition of post-training datasets presented additional challenges. Although previous studies suggest that language models generate a shared conceptual space across languages, the transferability of task-specific capabilities between languages – particularly dissimilar ones like English and Arabic – remains poorly understood (Csaki et al., 2024). To address this, we conducted post-training in both languages by generating data samples tailored to each language, ensuring that each version was contextually appropriate and aligned with cultural and value considerations.

Our training process follows a multi-stage training workflow, as illustrated in Fig. 12. In our earlier experiments, we found that combining general capability data with task-specific data requiring cultural and value awareness diminished the model's effectiveness in handling the latter. To address this, we implemented supervised fine-tuning (SFT) in multiple stages, performing additional training on what we identified as higher-quality data using a lower learning rate. Capabilities requiring value alignment were incorporated into the second stage of fine-tuning. The instruction-tuned model then underwent further training through a preference optimization step, utilizing preference data generated either by our SFT model (on-policy data) or by external models (off-policy data). Our development process was guided by a combination of benchmarks designed to evaluate the chat capabilities of our models and continuous feedback from external testers using our playground. Our training concludes with an annealing stage that utilizes a highly curated, small subsample of SFT and preference data. The same post-training stages were applied to both **Fanar Star** and **Fanar Prime** models. Both models were trained to support a system prompt as part of their chat template.
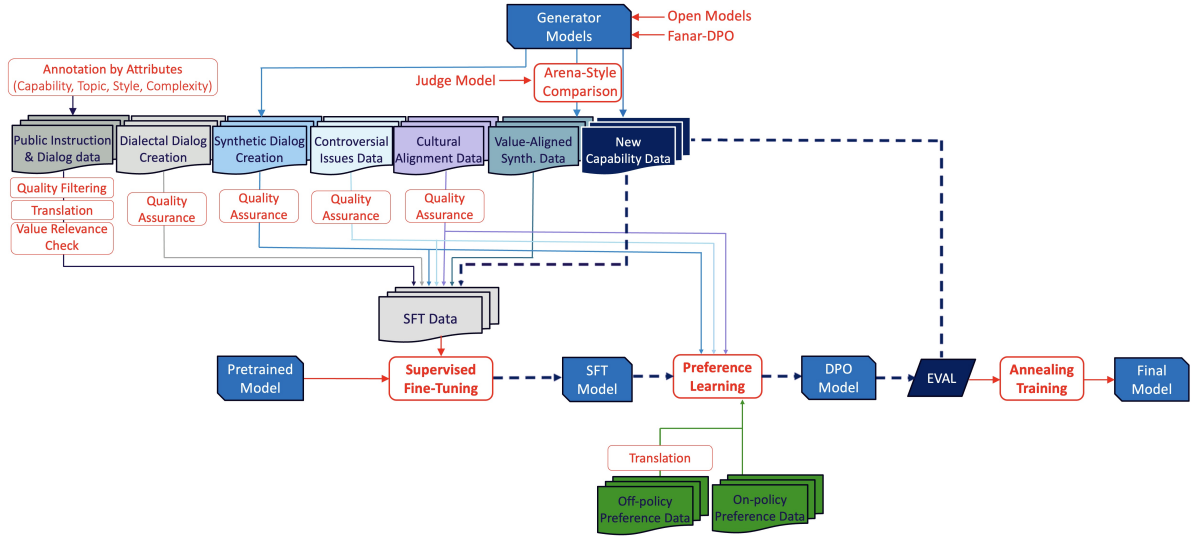
Figure 12: Post-training workflow for Fanar.

7.1. Supervised Fine-Tuning

The SFT data consists of several distinct splits, encapsulating a wide range of capabilities and behaviors, as illustrated in Fig. 12. This data is primarily obtained through three key processes: extensive filtering and annotation of publicly available SFT data, synthetic data generation, and a small yet critical amount of expert- and vendor-created data. To optimize performance, we implemented a two-stage SFT training approach rather than a single training round, drawing inspiration from the way pre-training is performed in multiple cooling-off rounds using progressively higher-quality data. Accordingly, the data relevant to behaviors that require a more nuanced interpretation and more challenging capabilities are introduced or repeated in the second round at a lower learning rate to reinforce learning. Our tests revealed that employing this two-stage training approach, rather than utilizing all the data in a single training round, results in a model that achieves higher performance in both automated benchmarks and user evaluations. To the best of our knowledge, this two-stage SFT process is novel, particularly in its application for creating a more value-aligned model.

7.1.1. Data Curation from Public Sources

Generating high-quality samples for behavior mimicking has been a significant area of research, with various criteria proposed to evaluate sample quality (Xia et al., 2024; Shen, 2024; Wang et al., 2023a; Zhao et al., 2024). Although a vast amount of public instruction and dialogue data is available, much of it is automatically generated en masse using other language models. Consequently, even widely-used and well-regarded datasets often contain a substantial proportion of low-quality samples. To address this, we began our curation process by annotating each instruction and dialogue sample along four key dimensions: topic, writing style, prompt complexity, and the capability reinforced by each data sample. For capability annotation, we expanded upon the capability-based test and evaluation taxonomy introduced in (Slack et al., 2023) by incorporating safety to develop a framework of 11 core capabilities. All classification tasks were conducted using the smallest Llama model, Meta-Llama-3.1-7B-Instruct.

Annotated data samples underwent a three-level filtering process to extract the highest-quality samples from public datasets. In the first level, each sample's quality was assessed. For this purpose, we designed specific rubrics for each capability category to quantify how effectively a data sample reinforced that capability. Separate rubrics were applied to single-turn and multi-turn data samples (Zhang et al., 2024). Llama-3.1 models were used to evaluate these rubrics, assigning a quality score to each sample. The quality score, along with prompt complexity and response length, was then used to identify accepted samples within each quality category. (Separate criteria were applied to category sample depending on the number of available samples.) Next, the selected samples were translated into Arabic using in-house

translation models as well as Google Translate. The translated samples were further filtered to eliminate incoherent outputs, particularly those involving tasks such as translation, grammar, word riddles, or sorting, which often yielded poor results. Finally, a value-relevance filter was applied to remove samples misaligned with cultural and religious values, ensuring the curated dataset met ethical and contextual standards.

This process resulted in approximately 2.5 million instructions and dialogues across 11 categories in both languages. Figure 13 presents the composition of our curated dataset, highlighting provenance characteristics and dataset size, sourced from public data after the three-level filtering process. Notably, our observations indicate that public datasets often consist of a mix of data samples with varying quality, with few being uniformly high-quality. This core capability data is utilized during the initial stage of supervised fine-tuning.
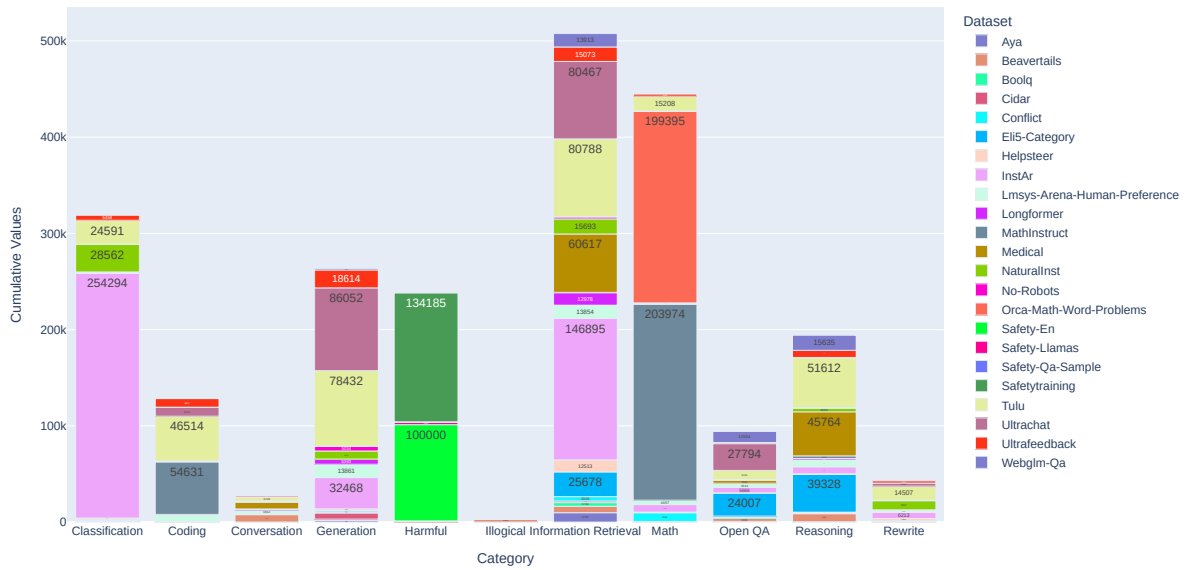

Figure 13: Composition and provenance characteristics of the core capability dataset curated from public sources.

7.1.2. Synthetic Data Generation

One notable drawback of curating data from public sources is the tendency of the translation process to introduce typical English-language contexts into Arabic. This includes culturally specific elements such as personal names, geographic references, traditions, social norms, and lifestyle practices, which often fail to capture local cultural nuances. To address this and ensure that our models are culturally and linguistically aligned with the preferences of Arabic-speaking users, we generated a substantial volume of synthetic data, including both single-turn and multi-turn dialogues.

Synthetic data generation, however, posed two key challenges. First, it required careful selection of models for generating completions. Rejection sampling was employed as a critical tool, generating multiple completions and selecting the best among them. However, our models trained on core capability data lacked the cultural awareness necessary to drive this generation effort effectively. To address this, we leveraged open, large-parameter public models, complemented by well-engineered system prompts specifically designed to instruct these models to create culturally and religiously appropriate samples for Arabic-speaking populations.

The second challenge was validating the quality of the model-generated outputs. This is typically achieved using a reward or preference model trained on extensive human preference data to assign a goodness score to each completion. However, due to the lack of sufficient user preference data at the start of our training process, we employed the quantized version of the largest Llama model (Meta-Llama-3.1-405B-Instruct-

FP8) as the judge model. The core steps of our assessment process for a given set of prompts are as follows:

- Generate output responses using multiple models for each prompt.

- For Arabic outputs, run a Spell Checker[13] (Mubarak and Darwish, 2014a) to automatically detect spelling errors. Set a threshold to filter out outputs that are deemed as having too many errors.

- Use the judge model to score responses on a scale of 1 to 10, considering scores of 8, 9, or 10 as good responses.

- Conduct an arena-style comparison to select the best response for each prompt by performing pairwise comparisons of high-scoring responses. Selection accounts for positional bias and response length.

- Create an SFT dataset using the selected prompts and responses.

To ensure meaningful cultural alignment, all prompts used at this stage focused on capabilities relating to creative writing, rewriting, in-context retrieval, and conversational queries, introducing a variety of contexts. These prompts were drawn from two main sources: prompts discarded during the quality filtering stage due to low response scores based on our rubrics, and existing prompts from the curated dataset.

Another issue we identified was the model's tendency to use English entity names and locations in creative-writing tasks, even when responding to Arabic queries. To address this, we regenerated responses from the curated dataset, specifically targeting queries containing the 100 most common English male and female names. Incorporating 30K contextually appropriate responses effectively mitigated this behavior. Additionally, we generated synthetic data to support a range of other capabilities, including precise instruction following, handling controversial topics, and uncertainty learning.

Our synthetic data generation process extensively utilized carefully crafted prompts to guide larger models with 27B+ parameters, ensuring the responses met our quality standards while aligning with Arabic cultural and religious values. To achieve this, we developed three types of prompts: system prompts to direct text generation by the models, a system prompt for the judge model to evaluate and score responses from selected generation models, and a system prompt for the judge model to perform arena-style comparisons of pairwise responses.

Overall, through the use of stronger models, we generated close to a million samples in both Arabic and English, mainly focusing on culturally contextualizing and aligning the model's responses. The synthetically generated data was utilized in both stages of SFT, with value-alignment-related splits included in their entirety, while other splits were only subsampled.

7.1.3. New Capability Data

Our tests and in-loop evaluations during the post-training phase revealed several gaps and weaknesses. To address these, we enhanced the existing dataset with additional targeted datasets, each designed to introduce specific behaviors. These included:

- A dataset created by rephrasing and expanding a core set of manually generated prompts and responses to cover a closed set of user questions.

- Datasets generated by transforming high-quality textual sources focusing on IslamQ&A, poetry, and humor into instructional formats.

- Datasets broadening the model's capabilities on language tasks, such as diacritization, question generation, and grammar correction.

- A vendor-generated dataset for dialectic dialogues.

---

[13]https://farasa-api.qcri.org/

- Expert-driven instructional and DPO data tailored to address nuanced topics, including controversial Islamic issues.

7.2. Preference Learning

To further enhance the model's capabilities, we implemented Direct Preference Optimization (DPO) (Rafailov et al., 2024), an offline reinforcement learning method that eliminates the need to explicitly build a reward model or sample from the model during training. Our binary preference data was obtained from two primary sources. Off-policy preference data was derived from three key public datasets: UltraFeedback (Cui et al., 2024), HelpSteer (Wang et al., 2023b, 2024), and Nectar (Zhu et al., 2024). This data underwent the same quality and value-relevance filtering process applied during the curation of the SFT data to ensure alignment with our standards. We also generated on-policy preference data by producing both accepted and rejected responses using our models. To distinguish between accepted and rejected responses, we employed the Reinforcement Learning with AI Feedback (RLAIF) approach, which primarily relied on the preferences provided by judge models. However, when generating culturally aligned data to address nuanced cultural issues, we relied on user annotations to determine preferences for model-generated responses.

To establish trust in the use of a large language model (LLM) for preference annotation in Arabic, we conducted a user study. We selected approximately 700 user-generated prompts where our model's responses were initially disliked and scored below 7 by the Llama judge model. Using our generator models, we created responses to these prompts, with the best response selected by the judge through scoring and arena-style comparison, as previously described. Users were presented with prompts paired with two anonymized responses: one being the rejected response from our model and the other the accepted response generated by other models. To ensure unbiased evaluations, the judge's selections were concealed, and each pair of responses was independently assessed by 5-6 users. We aggregated the user annotations by conservatively scoring *likes* and *greats* as +1 and *dislikes* as -1, calculating the total score for both the accepted and rejected responses. The score difference between the accepted and rejected responses was then computed for each prompt, where a negative score indicated disagreement of human annotators with the judge. Our findings showed that the Llama 405B model achieved approximately 87% agreement with human annotations, aligning with earlier studies on GPT-4's agreement with human evaluators (Zheng et al., 2023).

For generating on-policy data, prompts were selected from our curated dataset and user queries submitted through the playground with disliked responses. Accepted and rejected responses were produced by slightly varying the temperature settings to create pairs of outputs with noticeable deviations. The responses were evaluated using two judge models, Llama-405B and Gemma-27B. The average score from these models was used to classify the responses. A response was marked as accepted only if its average score was eight or higher, while rejected responses required scores between two and six, ensuring a minimum gap of two points between the paired responses. This process generated approximately 250K preference data samples, balanced across both languages, with around 20% comprising on-policy data.

During preference optimization, we explored several DPO variants as alternatives, including IPO (Azar et al., 2024), KTO (Ethayarajh et al., 2024), SimPO (Meng et al., 2024), and ORPO (Hong et al., 2024)[14]. However, these approaches led to a decline in performance on our automated benchmarks, so we decided not to adopt them. Additionally, we observed that applying preference data in batches, rather than processing the entire dataset in a single training run, led to a modest but consistent improvement in our benchmarks (+1-2%). Updating the reference model every 40K samples yielded the best results, even though it introduced minor off-policy characteristics to the initial accepted and rejected responses. During testing, we observed that the model occasionally failed to respond in the language of the user query. This issue was traced to an imbalance between the number of Arabic and English DPO samples, with the model defaulting to the majority language. To address this, we balanced the dataset between the two languages and incorporated some mismatched-language responses as rejected responses, which corrected the issue.

---

[14]LLaMA-Factory implementations are used.

7.3. Annealing Stage

Our training ends with an annealing phase, where the learning rates for the SFT and preference optimization steps are reduced to near zero while presenting the model with a high-quality subsample of SFT and DPO data. This final stage serves two main objectives. First, the training process involves a diverse range of tasks with varying levels of complexity, making it challenging to perfectly balance the data to reflect this variability. As a result, task interference becomes inevitable, especially at smaller parameter sizes. For example, as additional capability data was incorporated into the training, the model began to struggle with certain math problems and nuanced responses that earlier versions handled effectively. Thus, as the final set of training data presented to the model, this stage reinforces learning by helping the model retain the diverse range of tasks encountered in earlier phases.

In addition, this stage serves as a rapid response mechanism, allowing fast retraining of the model when harmful or culturally misaligned behaviors are identified. Rather than relying solely on such disliked responses to teach a new behavior or reinforce one, these samples are integrated with other capability data to create a balanced dataset, ensuring that previously learned capabilities are preserved. While this annealing stage led to a slight decrease in automated benchmark performance, it noticeably reduced the user dislike rate. In our playground, both models—**Fanar Star** and **Fanar Prime** —are deployed. Harder queries, such as those related to math, reasoning, and coding, are routed to **Fanar Prime** due to its superior performance in these areas. Since **Fanar Star** handles most user queries, the annealing stage was applied only to it, leaving **Fanar Prime** unchanged.

7.4. Infrastructure & Hyperparameters

All post-training activities were carried out on 2–3 nodes, each equipped with 8x H100 GPUs. Table 8 details the training parameters used during the supervised fine-tuning and preference optimization stages. Notably, the Gemma-based continually pre-trained model required a learning rate an order of magnitude smaller than the model trained from scratch.

Table 8 Post-Training Hyperparemeters

	Training Phase	Number of Sample	Fanar-Star Batch	Fanar-Star LR	Fanar-Prime Batch	Fanar-Prime LR
SFT	Stage-1	3.6M	256	5.0e-06	640	5.0e-07
SFT	Stage-2	834k	512	1.0e-06	640	1.0e-07
	DPO	250k	64	1.0e-07	128	1.0e-07
	Annealing-SFT	5k	128	6.0e-08	640	6.0e-08
	Annealing-DPO	4k	64	3.0e-08	64	3.0e-08

7.5. Collection of User Feedback

In the early stages of our model development, we implemented a playground equipped with evaluation functionality. This allowed users to compare responses from multiple models and provide feedback based on three attributes: *like, dislike*, and *great*. The first two attributes could be assigned to each model output, while *great* was reserved for a single model output that stood out. (Our menus supported both languages depending on the user preference.) When *dislike* was selected, users were prompted to choose from eight predefined dislike categories, including lack of factuality, length issues, instruction adherence failures, insufficient harmlessness, refusal, cultural misalignment, and grammatical errors. This feedback mechanism enabled continuous monitoring of model improvements and helped identify systematic gaps in capabilities.

Our system engaged a team of 40–250 annotators at different stages, primarily Arabic speakers from various locations, who, at peak capacity, contributed up to 10K prompts and provided feedback on model responses each day. Over time, our core user base grew to 130 individuals from diverse professional and academic backgrounds. On any given day, 60 to 90 of these users actively participated in model

evaluation. Additionally, students from several universities participated in the model testing efforts for varying durations. During the early testing phase, users were free to prompt the model with any queries they preferred. However, as development progressed, testing became more guided, focusing more on capability-oriented evaluations. Over 90% of the prompts were in Arabic, and when testing was unguided, most user prompts (∼70%) tended to focus on open-domain question-answering queries.

In addition to highlighting capability gaps and areas of cultural misalignment, user feedback enabled us to identify issues such as grammatical errors and presentation problems, to trace them back to their sources, and to refine our data filtering processes. This feedback also informed adjustments to how we utilized judge and generation models. We observed that our model, trained from scratch, had a dislike rate of approximately 13%, with the vast majority of dislikes related to factual accuracy. This can be safely attributed to the relatively small size of our pre-training corpus.

7.6. In-Loop Evaluations

The training progress was continuously monitored using a combination of benchmarks designed to evaluate the instruction-following and conversational capabilities of our models. For Arabic, we utilized a combination of five public and internal benchmarks. These included translated versions of MT-Bench (Zheng et al., 2023) and Alpaca-Eval (Li et al., 2023b), with modifications to humanities-related questions in MT-Bench to better suit the Arabic context (Boughorbel and Hawasly, 2023). To evaluate Arabic language proficiency, we used the development set of the BALSAM benchmark for tasks relevant to our use cases (Consortium, 2024). To further enhance evaluation, we developed a general capability benchmark called Almieyar, expanding upon the 10 capability categories used for annotating our SFT data. User chats were also incorporated as a crucial evaluation source, offering valuable insights into real-world performance. These chats typically consisted of multi-turn dialogues, sometimes extending up to 100 turns, enabling a comprehensive assessment of model capabilities. To ensure that English proficiency was maintained, we included the English counterparts of these benchmarks alongside IFEval (Zhou et al., 2023). For both languages, evaluations were primarily conducted using the LLM-as-a-Judge framework, with GPT-4o or GPT-4 scoring the model responses.

To validate model improvements, we also incorporated user preferences. Specifically, we identified 555 data samples focusing on various capabilities, primarily inspired by challenging user prompts. Users compared the responses of the baseline model to those of its updated version using these standardized queries. The data samples included both single-turn and multi-turn dialogs; for the latter, users were shown only the response from the final turn. Models that demonstrated improvements in both automated benchmarks and user comparisons were subsequently deployed in our playground for further testing. The following section presents a performance comparison of our two models against other models of similar parameter sizes across multiple-choice questions, conversational tasks, and instruction-following benchmarks used during post-training.

8. Evaluation

> **Fanar Star** and **Fanar Prime** are evaluated against Arabic-aware peer models on standard and proposed culturally aware benchmarks.

Evaluation of large language models (LLMs), especially in the context of Arabic, remains in its early stages, with no universally accepted framework for comprehensively assessing their capabilities. In this work, we adopt the common practice of benchmarking our models against comparable baselines across a diverse set of tasks and formats. This approach aims to provide a detailed understanding of the models' performance and highlight their unique strengths.

The Fanar model family is trained on diverse datasets encompassing standard Arabic, dialectal Arabic, English, and code. This multilingual and multi-domain training enables the model to excel in various tasks, including reading comprehension, logical reasoning, knowledge extraction, and standard NLP

applications. Our evaluation framework leverages widely recognized benchmarks for reasoning, reading comprehension, and question-answering in both Arabic and English. It also includes assessments for conversational abilities and instruction-following tasks. To address gaps in existing benchmarks, we have developed new datasets specifically designed to capture cultural and dialectal nuances. We plan to release these benchmarks for the benefit of the broader research community. The following sections provide detailed descriptions of the selected benchmarks and baseline LLMs, followed by an analysis of the evaluation results.

8.1. Benchmarks

We benchmark Fanar and baselines on three types of benchmarks: automatic evaluations, open-ended and conversational assessments, and human evaluations.

8.1.1. Automatic evaluation with multi-choice questions

We use LM-Evaluation-Harness[15] (Gao et al., 2024) version 0.4.3 as the backbone for most of the automatic evaluation tasks. We use the default task definitions for English tasks and create custom task configurations for Arabic tasks. For the Open Arabic LLM Leaderboard (OALL) benchmark, we use Hugginface's Lighteval[16] (Fourrier et al., 2023) that supports the live leaderboard (Elfilali et al., 2024).

English tasks we opt for standard English benchmarks that capture key capabilities: 1) **MMLU** (Hendrycks et al., 2020) tests for general world knowledge through 4-choice questions in 57 subcategories; 2) logical inference is tested through **PIQA** (Physical Interaction QA) (Bisk et al., 2020) comprising 3K binary commonsense questions, **Hellaswag** (Zellers et al., 2019) comprising 10K multi-choice continuation questions about commonsense events, **ARC Challenge** (Clark et al., 2018) comprising 2.5K difficult multiple-choice grade-school level science questions, and **Winogrande** (Sakaguchi et al., 2021) which presents 1.7K binary fill-in-the-blank type logical questions; 3) mathematical skills are tested with **GSM8K** which comprises 1K multi-step math word problems.

Standard Arabic tasks 1) The Arabic section of **MMMLU** (OpenAI, 2024): an OpenAI expert translation of the MMLU dataset; 2) **ArabicMMLU** (Koto et al., 2024) which covers 40 subjects in 14,575 multiple-choice questions in Modern Standard Arabic (MSA); 3) The MSA-translation subset of **PIQA** provided by the AraDiCE project (Mousi et al., 2024) to test commonsense reasoning; 3) the **Open Arabic LLM Leaderboard** suite (Elfilali et al., 2024) which is the average of 14 tasks.

Cultural benchmarks 1) **Arabic Cultural Value Alignment** (ACVA) (Huang et al., 2023) to measure culture understanding through true/false questions generated by GPT-3.5 Turbo; 2) **Arab Cultural MCQ** an in-house developed task to capture cultural nuances in Arabic communities, comprising 1K multi-choice questions extracted from relevant cultural guides and sources and curated thoroughly. More details about the benchmark could be found in Appendix C.2.

Dialectal benchmarks 1) the Arabic part of **Belebele** (Bandarkar et al., 2024): the average of six subsets of the dialectal reading comprehension task in MSA, Levantine, Gulf, Egyptian, Iraqi and Moroccan dialects; 2) **Almieyar**: an in-house task that measures competency in different aspects of Arabic language and its dialects. More details about the benchmark could be found in Appendix C.1; and 3) human-translated subset of **PIQA** and **ArabicMMLU** to Egyptian and Levantine dialects provided by the AraDiCE project (Mousi et al., 2024).

---

[15]https://github.com/EleutherAI/lm-evaluation-harness
[16]https://github.com/huggingface/lighteval

8.1.2. Conversation and instruction following evaluations

To benchmark the models' ability to follow instructions and engage in dialog, we utilize verifiable instructions and capable LLMs as judges in open-ended generation tasks.

English tasks we benchmark using 1) MT-Bench (Zheng et al., 2023), a set of 80 two-turn open-ended questions that test model's ability in writing, roleplay, reasoning, math, coding, extraction and STEM and humanities knowledge, judged by GPT-4; 2) AlpacaEval (Li et al., 2023b) a set of 805 instructions derived from a diverse set of instruction sources, judged by GPT-4; and IFEval (Zhou et al., 2023) which comprises 25 verifiable instructions and 541 prompts.

Arabic tasks we benchmark using 1) the Arabic MT-Bench (Boughorbel and Hawasly, 2023) which is a curated and culturally-appropriated translation of the original MT-Bench; 2) a translation of AlpacaEval to Arabic scored by GPT-4; and 4) our in-house conversational benchmark, user-chats, which includes 3.5K single- and multi-turn chats (averaging 2.7 turns per chat, with a maximum of 100 turns) from our testers, spanning a diverse range of topics and capabilities. The user prompts from each chat session are provided to the model, and GPT-4o is used to judge the quality of the answers.

8.1.3. Human evaluation

As instruction datasets might not be representative of real usage, we created a conversational test set of 555 single- and multi-turn dialogues, primarily drawn from our user messages. Model responses are generated for each prompt and presented to three human evaluators to express their preferences. To streamline the evaluation, users were shown only the final prompt and its corresponding model response rather than the full sequence of responses.

8.2. Baselines

Table 9 Fanar and baseline model sizes and training data statistics. The shaded rows indicate the models which have been continually-trained from some base model as compared to training from scratch. A '-' indicates that information is not available or not applicable.

	Parameters	Tokens	
		Pre-training	Continual training
Jais-family-6p7b	6.7B	0.48T	-
OLMo-7B-0724-hf	7B	2.46T	-
Fanar Star	7B	3T	-
ALLaM-Base (from scratch)	7B	5.2T	-
Qwen2.5-7B	7B	18T	-
ALLaM-Base	7B	2T (Llama 2)	1.2T
AceGPT-v2-8B	8B	15T (Llama 3)	0.11T
Aya-Expanse-8B	8B	-	-
Llama 3.1	8B	15T	-
Fanar Prime	8.7B	8T (Gemma-2-9b)	1.0T
Gemma-2-9b	9B	8T	-
Jais-13B	13B	0.4T	-
Jais-adapted-13b	13B	2T (Llama 2)	0.28T

We choose a number of capable LLMs that have been shown to exhibit excellent bilingual performance in Arabic and English in the same size range as Fanar models. These models are:

- Jais (Inception, 2024): a bilingual Arabic-English family of models. We use the 6.7B parameter `Jais-family-6p7b` model and `Jais-13B` (Sengupta et al., 2023), which are both trained from scratch, (plus their corresponding chat models `Jais-family-6p7b-chat` and `Jais-13B-chat`), and the newer `Jais-adapted-13B` that has been evolved from Llama 2 (and its corresponding `Jais-adapted-13B-chat`).

- OLMo (AllenAI, 2024): the family of models whose architecture and training framework were the starting point for `Fanar Star`. We benchmark the `OLMo-7B-0724-hf` base model and its corresponding `OLMo-7B-0724-Instruct-hf` chat variant.

- Allam (Bari et al., 2024): A family of Arabic-centric models. We report results for the 7B from-scratch `ALLaM-Base (from scratch)` and the 7B model evolved from Llama 2 `ALLaM-Base`. We also report results for the corresponding chat models `ALLaM-Instruct (from scratch)` and `ALLaM-Instruct`. Because Allam models are only released via endpoints, we reproduce all the results that require access to logits from the Allam report (Bari et al., 2024).

- Qwen 2.5 (Cloud, 2024): a series of multilingual models with strong performance in Arabic. We benchmark `Qwen2.5-7B` and its corresponding fine-tuned sibling `Qwen2.5-7B-Instruct`.

- AceGPT v2 (FreedomIntelligence, 2024): An Arabic-centric model built on top of Llama 2. We benchmark `AceGPT-v2-8B` and its chat version `AceGPT-v2-8B-Chat`.

- Aya Expanse (Cohere, 2024): a family of fine-tuned models with highly advanced multilingual capabilities. We select the `Aya-Expanse-8B` variant.

- Llama 3.1 (Meta, 2024): open models from Meta. We benchmark the similarly-sized `Llama-3.1-8B` and its chat version `Llama-3.1-8B-Instruct`.

- Gemma 2 (Google, 2024): a family of light-weight models from Google. We benchmark `Gemma-2-9B`, a student model distilled from a much larger teacher and the starting point of `Fanar Prime`, and its corresponding instruction-tuned version `Gemma-2-9B-it`.

Table 9 presents statistics about the chosen baselines.

8.3. Evaluation results

8.3.1. Base models

We present in Table 10 the benchmarking results of base Fanar and baselines on Arabic benchmarks, and in Table 11 the English benchmarks. All the results reported are in *accuracy* percentage for multi-choice benchmarks with character output (e.g. 'A', 'B', 'C' and 'D' as in `MMLU`), and *normalized accuracy* for multi-choice benchmarks with textual answer (e.g. `PIQA`). For the generation task `GSM8K` we report the results for *exact_match* with flexible extract.

The shaded rows in the tables below indicate the models which have been continually-trained from some base model as compared to training from scratch (refer to Table 9), and we order the rows by model size. The best score for a benchmark is shown in **bold**, while the second best is indicated by an underline.

`Fanar Prime` offers excellent performance in Arabic benchmarks, achieving the best result in most of the benchmarks including the culture ones, with the only exception being dialectal PIQA. For English, it consistently beats its starting point `Gemma-2-9b-it` in most benchmarks except in PIQA and ARC Challenge, and achieves best or second best score all over.

8.3.2. Instruction-tuned models

We present in Table 12 the benchmarking results of fine-tuned Fanar and baselines on Arabic benchmarks, and in Table 13 the English benchmarks. As before, we report results in *accuracy* for multi-choice benchmarks with character output, *normalized accuracy* for multi-choice benchmarks with textual answer, and *exact_match* with flexible extract for `GSM8K`. For LLM-as-a-judge benchmarks the results are out of 10.

As can be seen from the tables, `Fanar Prime Instruct` is a strong competitor in Arabic, producing top scores in many automated and generative tasks. In nearly all Arabic evaluations, our models ranked

Table 10 Arabic benchmarks for Fanar base models and baselines. All the reported results are in accuracy or normalized accuracy. The top half contains benchmarks of world knowledge/logical inference and cultural benchmarks, and the lower half presents dialectal Arabic benchmarks. **Fanar Prime** achieves the best accuracy in most of the benchmarks.

		MMMLU(Ar)	ArabicMMLU	PIQA(Ar)	OALL	ACVA	CulturalMCQ
		0-shot	0-shot	3-shot	0-shot	0-shot	5-shot	3-shot
Jais-family-6p7b	6.7B	25.34	29.12	34.81	65.18	37.55	60.13	34.00
OLMo-7B-0724-hf	7B	25.37	27.22	35.35	49.95	37.20	49.60	29.30
Fanar Star	7B	36.16	38.75	50.20	65.13	42.02	68.60	62.80
ALLaM-Base (from scratch)	7B	36.28*	44.45*	-	-	-	68.46*	-
Qwen2.5-7B	7B	51.77	60.15	65.08	59.68	48.66	80.37	65.70
ALLaM-Base	7B	34.42*	41.52*	-	-	-	66.18*	-
AceGPT-v2-8B	8B	41.71	45.96	58.55	63.17	43.58	78.36	67.50
Llama 3.1	8B	43.21	46.56	55.73	57.51	43.01	77.72	60.00
Fanar Prime	8.7B	**57.30**	**61.14**	**67.35**	**65.83**	**54.79**	**81.40**	**71.90**
Gemma-2-9b	9B	54.04	57.80	64.32	63.98	50.24	79.66	68.60
Jais-13B	13B	29.91	35.77	39.23	64.96	38.20	62.70	35.60
Jais-adapted-13b	13B	34.01	42.93	51.96	65.02	40.79	73.52	60.90

		Belebele(Ar)	Almieyar(Ar)	PIQA(Egy)	PIQA(Lev)	ArabicMMLU(Egy)	ArabicMMLU(Lev)
		3-shot	3-shot	0-shot	0-shot	0-shot	0-shot
Jais-family-6p7b	6.7B	34.54	32.17	60.23	58.38	28.50	29.46
OLMo-7B-0724-hf	7B	30.87	41.52	51.41	50.05	26.72	26.32
Fanar Star	7B	47.76	59.10	60.88	57.78	29.31	39.75
Qwen2.5-7B	7B	71.72	76.81	57.51	55.44	47.33	49.26
AceGPT-v2-8B	8B	60.61	66.83	61.48	56.75	43.40	40.96
Llama 3.1	8B	61.59	63.84	55.28	53.81	41.44	38.39
Fanar Prime	8.7B	**79.37**	**77.68**	61.97	57.78	**55.68**	**55.41**
Gemma-2-9b	9B	75.31	73.82	60.17	58.05	49.61	47.15
Jais-13B	13B	35.39	37.03	**62.73**	58.65	29.65	36.38
Jais-adapted-13b	13B	43.02	62.34	62.19	**59.25**	38.24	37.93

Table 11 English benchmarks for Fanar base models and baselines. **Fanar Prime** has the highest scores for Hellaswag and Winogrande, is the second best for the rest of the benchmarks after `Qwen2.5-7B` and `Gemma-2-9b-it`, and it also beats its starting point `Gemma-2-9b-it` in MMLU and GSM8K.

		MMLU		PIQA	Hellaswag	GSM8K	ARC Challenge	Winogrande
		0-shot	5-shot	0-shot	0-shot	5-shot	0-shot	0-shot
Jais-family-6p7b	6.7B	29.03	32.50	75.95	69.28	6.90	40.27	65.11
OLMo-7B-0724-hf	7B	49.01	54.11	80.25	77.83	29.42	43.69	67.56
Fanar Star	7B	48.27	48.43	76.12	71.88	38.89	39.76	61.01
Qwen2.5-7B	7B	**71.71**	**74.18**	79.71	78.95	**83.24**	51.37	73.01
ALLaM-Base	7B	40.71*	-	79.00*	76.17*	16.98*	45.65*	68.90*
ALLaM-Base (from scratch)	7B	42.91*	-	80.58*	76.26*	16.15*	43.52*	68.43*
AceGPT-v2-8B	8B	59.98	63.55	80.03	76.97	30.93	49.40	73.01
Llama 3.1	8B	63.10	65.10	81.01	78.95	51.63	53.41	73.72
Fanar Prime	8.7B	69.24	71.14	81.34	**80.56**	71.42	60.15	**75.37**
Gemma-2-9b	9B	68.50	70.60	**82.97**	79.82	67.10	**65.53**	74.19
Jais-13B	13B	33.19	34.72	77.91	71.77	11.14	41.89	68.43
Jais-adapted-13b	13B	45.18	50.42	78.94	78.02	18.95	48.55	71.67

among the top two performers, with the Gemma-based model particularly excelling. This trend was also observed in English benchmarks, where the strengths of the Gemma, Llama, and Qwen models became more pronounced.

Table 12 Arabic benchmarks for instruction-tuned Fanar and baselines. We show in the table the same benchmarks that were used for base models, in addition to the LLM-as-a-judge benchmarks and human evaluation in the bottom. **Fanar Prime** Instruct shows superior performance in most of the standard benchmarks, all the cultural assessments and dialectal tasks, and offers competitive performance in generative open-ended tasks judged by LLMs and humans.

* ALLaM results are reproduced from the technical report (Bari et al., 2024) and not validated with our setup
** The ALLaM model was accessed via the Microsoft Azure cloud platform.

		MMLU(Ar)	ArabicMMLU		PIQA(Ar)	OALL	ACVA	CulturalMCQ
		0-shot	0-shot	3-shot	0-shot	0-shot	5-shot	3-shot
Jais-family-6p7b-chat	6.7B	41.59	54.92	55.80	62.51	48.20	72.04	64.10
OLMo-7B-0724-instruct-hf	7B	27.14	34.75	35.88	50.22	38.43	50.94	34.40
Fanar Star Instruct	7B	40.67	51.32	53.24	**69.86**	48.76	74.55	61.40
ALLaM-Instruct (from scratch)*	7B	51.38*	**69.16***	-	-	-	79.59*	-
Qwen2.5-7B-Instruct	7B	55.63	61.55	63.96	60.55	54.19	78.09	68.10
ALLaM-Instruct*	7B	49.60*	66.90*	-	-	-	80.33*	-
AceGPT-v2-8B-Chat	8B	51.16	61.02	62.61	64.58	50.16	77.66	68.90
Aya-Expanse-8B	8B	47.14	58.82	60.10	63.00	53.18	77.11	67.30
Llama 3.1-Instruct	8B	47.58	56.50	59.05	58.11	47.81	76.67	66.10
Fanar Prime Instruct	8.7B	**58.74**	64.90	**67.82**	67.14	**63.81**	**80.85**	**70.60**
Gemma-2-9b-it	9B	57.93	63.43	64.16	61.26	56.11	80.67	68.60
Jais-13B-chat	13B	40.44	57.30	55.36	65.72	49.76	71.88	64.90
Jais-adapted-13b-chat	13B	44.45	57.81	58.97	61.10	46.41	75.24	65.30

		Belebele(Ar)	Almieyar(Ar)	PIQA(Egy)	PIQA(Lev)	ArabicMMLU(Egy)	ArabicMMLU(Lev)
		3-shot	3-shot	0-shot	0-shot	0-shot	0-shot
Jais-family-6p7b-chat	6.7B	65.11	61.72	60.12	57.24	49.11	47.49
OLMo-7B-0724-instruct-hf	7B	34.54	45.39	49.67	50.27	28.18	27.88
Fanar Star Instruct	7B	62.02	58.98	62.30	59.25	46.63	47.15
Qwen2.5-7B-Instruct	7B	74.61	75.69	58.65	56.04	48.74	53.42
AceGPT-v2-8B-Chat	8B	74.56	74.31	61.32	56.91	54.53	53.91
Aya-Expanse-8B	8B	70.78	70.20	59.41	56.53	53.52	53.71
Llama 3.1-Instruct	8B	66.72	69.70	55.39	54.24	46.85	47.52
Fanar Prime Instruct	8.7B	**82.48**	**78.30**	**63.60**	59.74	**59.22**	**60.14**
Gemma-2-9b-it	9B	78.31	73.69	59.96	57.24	57.95	59.25
Jais-13B-chat	13B	66.39	62.34	62.40	59.19	51.30	51.87
Jais-adapted-13b-chat	13B	67.52	65.46	58.05	55.77	52.87	53.59

		MT-Bench(Ar)	Alpaca LCW (Ar)	User-Chats
Jais-family-6p7b-chat	6.7B	6.47	25.60	8.43
OLMo-7B-0724-instruct-hf	7B	2.11	1.04	2.60
Fanar Star Instruct	7B	6.89	36.81	9.46
ALLaM-7B-Instruct**	7B	7.61	33.16	9.43
Qwen2.5-7B-Instruct	7B	8.63	33.61	8.65
AceGPT-v2-8B-Chat	8B	7.48	25.76	9.01
Aya-Expanse-8B	8B	8.51	**62.95**	9.52
Llama 3.1-Instruct	8B	6.95	14.92	7.59
Fanar Prime Instruct	8.7B	**8.93**	55.60	**9.66**
Gemma-2-9b-it	9B	8.88	49.60	9.27
Jais-13B-chat	13B	4.31	6.44	6.73
Jais-adapted-13b-chat	13B	6.61	26.62	8.97

Table 13 English benchmarks for instruction-tuned Fanar and baselines. We show the same benchmarks as the base model in the top, in addition to LLM-as-a-judge and verifiable generative tasks in the bottom. **Fanar Prime** Instruct offers competitive performance in English even though it was developed as an Arabic-centric model.

* ALLaM results are reproduced from the technical report (Bari et al., 2024) and not validated with our setup
** The ALLaM model was accessed via the Microsoft Azure cloud platform. MT-Bench and Alpaca were not evaluated during the time of access.

		MMLU		PIQA	Hellaswag	GSM8K		ARC Challenge	Winogrande
		0-shot	5-shot	0-shot	0-shot	0-shot	5-shot	0-shot	0-shot
Jais-family-6p7b-chat	6.7B	50.50	49.42	74.05	72.04	55.04		44.62	62.35
OLMo-7B-0724-instruct-hf	7B	51.69	52.29	79.00	79.99	18.42		47.01	65.11
Fanar Star Instruct	7B	47.38	46.83	79.11	73.34	42.84		45.90	64.48
ALLaM-Instruct (from scratch)*	7B	42.91*	-	80.58*	76.26*	53.60*		52.05*	69.93*
Qwen2.5-7B-Instruct	7B	**72.08**	**74.21**	79.92	80.44	<u>77.33</u>		55.03	71.35
ALLaM-Instruct*	7B	40.71*	-	79.00*	76.17*	49.28*		51.45	70.56*
AceGPT-v2-8B-Chat	8B	64.86	66.45	80.58	79.21	57.01		53.50	73.72
Aya-Expanse-8B	8B	60.55	62.85	<u>81.18</u>	78.54	76.12		56.40	64.80
Llama 3.1-Instruct	8B	67.92	68.04	80.74	79.22	74.91		55.29	73.88
Fanar Prime Instruct	8.7B	69.15	70.97	**81.99**	**83.06**	**83.02**		64.93	**78.69**
Gemma-2-9b-it	9B	<u>71.34</u>	<u>71.65</u>	79.38	79.06	60.42		<u>63.99</u>	<u>75.69</u>
Jais-13B-chat	13B	49.45	48.46	78.24	77.59	27.98		46.84	68.59
Jais-adapted-13b-chat	13B	55.63	56.64	80.47	<u>80.86</u>	68.23		54.27	69.77

		MT-Bench	IFEval	Alpaca LCW
Jais-family-6p7b-chat	6.7B	4.68	43.28	5.77
OLMo-7B-0724-instruct-hf	7B	5.64	51.07	8.53
Fanar Star Instruct	7B	5.16	67.98	10.40
ALLaM-7B-Instruct**	7B	-	47.36	-
Qwen2.5-7B-Instruct	7B	<u>8.30</u>	66.67	32.78
AceGPT-v2-8B-Chat	8B	7.14	59.83	11.96
Aya-Expanse-8B	8B	7.17	66.90	<u>44.68</u>
Llama 3.1-Instruct	8B	7.62	**82.25**	27.31
Fanar Prime Instruct	8.7B	7.48	74.82	27.71
Gemma-2-9b-it	9B	**8.38**	<u>78.65</u>	**49.65**
Jais-13B-chat	13B	5.59	59.71	1.87
Jais-adapted-13b-chat	13B	4.18	34.77	8.33

9. Integrating Multimodal Support

The training process for the speech and image modalities is explained, including efforts to ensure inclusive support for multi-dialectal Arabic speech and culturally aligned image generation.

Fanar provides support for both speech and image modalities. Users of Fanar can interact with the models using speech as input and can receive the model-generated responses in audio format, enhancing accessibility and user experience. We leverage in-house state-of-the-art bilingual speech technology that supports both English and Arabic, including a wide range of Arabic dialects. For image modality our objective was to generate imagery which is culturally aligned with and appropriate for the Arabic-speaking region.

9.1. Speech Modality

Speech is one of the most crucial and natural forms of interaction. This multifaceted modality does not just convey language but encapsulates human emotion, the intention behind their action, and personal identification while reflecting the speakers' surrounding condition. Understanding such rich information is paramount to achieving advanced AI capabilities. Therefore, with Fanar, we added support for processing spoken instruction and generating output with natural-sounding speech. An overview of Fanar's spoken interaction support is presented in Figure 14.

Inclusive Arabic Speech Recognition System Fanar integrates a state-of-the-art in-house Arabic-English automatic speech recognition (ASR) system that supports multiple Arabic dialects regardless of the speaker's nativeness, accentedness, or environmental context. Additionally, Fanar accommodates accented English and seamlessly handles code-switching between languages (Ar ↔ En) and within dialects (e.g., MSA ↔ EGY).

Model Design: To design such an inclusive model, we leveraged a multilingual self-supervised foundation model (FM) to extract rich contexual representation from the raw speech input. We then passed the extracted representation through the layers of conformer-based encoders, followed by transformer decoders. We then train the model utilizing multitask learning objectives – combining the decoder's cross-entropy loss and CTC loss with a weighting factor, α (Chowdhury et al., 2021). Finally, we average the weight of the best 10 model checkpoints based on the validation set.
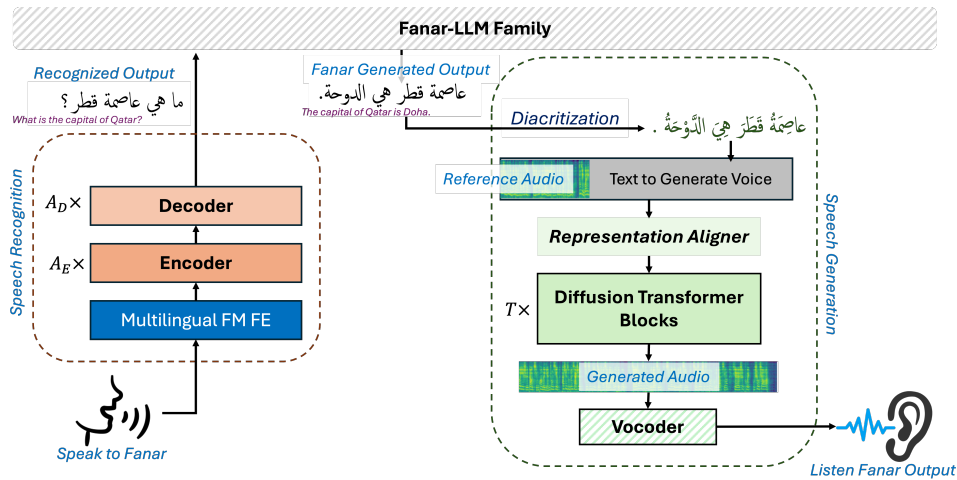

Figure 14: Overview of Fanar spoken interaction Capabilities enabled by automatic speech recognition (ASR) and text-to-speech generation (TTS) modules. FM: Foundation model, FE: feature extractor. A_E and A_D represent the numbers of encoder and decoder blocks respectively in the ASR, and T represents number diffusion blocks in the models.

Training Strategy and Data: We trained the speech model with a total of $\approx 15,000$ hours of data in English and Arabic. For Arabic, the dataset comprised both MSA and dialectal Arabic from diverse domains, such as broadcasts, podcasts, meetings, and academic conferences. Incorporating dialectal support was pivotal to Fanar for expanding accessibility - reaching people of varying education levels and age groups while upholding cultural inclusion (Ali et al., 2021). However, the scarcity of annotated dialectal speech data posed significant challenges. To address this, we employed data augmentation techniques and a customized curriculum-based batching strategy. The customized batching approach introduced dialects incrementally based on their phonetic and linguistic proximity to MSA, ensuring effective model generalization (Kheir et al., 2024). For English, the training data is selected from widely used publicly available speech datasets, capturing a wide range of speaking styles, domains, and accents. The model is highly Arabic-centric, with 75% of the vocabulary tokens (out of total 10,000 bpe tokens) representing the Arabic language, and only 25% tokens are reserved for English.
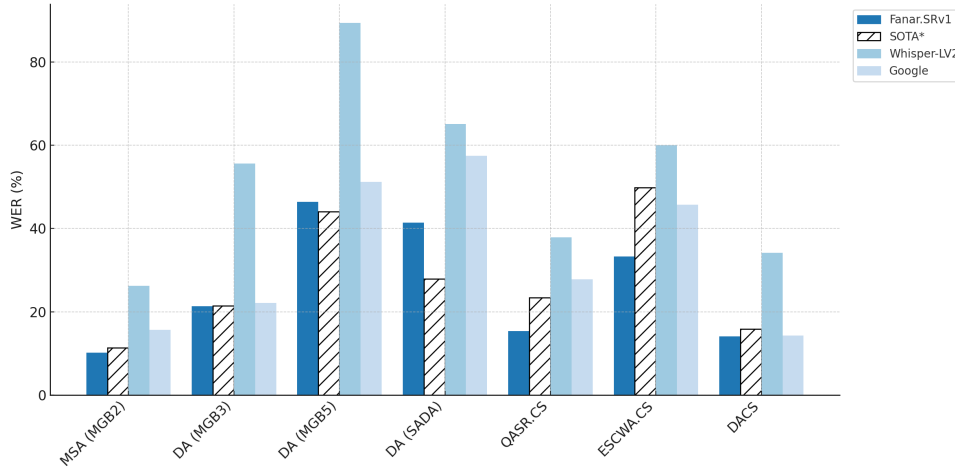
Figure 15: Reported word error rate (WER) on widely used Arabic testsets, presented in LAraBench (Abdelali et al., 2024) showcasing Fanar's ASR performance compared to Google and OpenAI's whisper models. All SOTA and third-party ASR results are from LAraBench, except the SADA results which are from the SADA-dataset paper (Alharbi et al., 2024). The SOTA results represent performance when the model is trained specifically on the training set portion of the test data. Note that SADA, ESCWA.CS test sets are completely unseen to the Fanar ASR model.

Model Performance: We evaluate the performance of Fanar SR using widely recognized test sets and compare the result with comprehensive benchmarks reported in LAraBench (Abdelali et al., 2024). Following LAraBench, for Arabic, we reported QASR-test (Mubarak et al., 2021) for MSA and MGB3 (Ali et al., 2017), MGB5 (Ali et al., 2019) for dialectal performance. In addition, we also included performance for the latest SADA (Alharbi et al., 2024) dataset for evaluating dialectal varieties, and showcasing the model's inclusivity across regional dialects. For English, we benchmarked with the popular LibriSpeech (Panayotov et al., 2015) dataset. For assessing the model's performance on code-switching tasks, we used the QASR.CS (Chowdhury et al., 2021) and ESCWA.CS (Chowdhury et al., 2021) for Arabic-English intra-utterance code-switching and DACS.CS (Chowdhury et al., 2020) for dialectal code-switching scenarios.

Our results, in Figure 15, show the efficacy of the Fanar ASR. The results indicate that the in-house model consistently outperforms the available large-scale speech models such as Google ASR and Whisper-Largev2 for the LAraBench (Abdelali et al., 2024) and SADA (Alharbi et al., 2024) datasets. Notably, Fanar ASR also achieve significantly better or comparable results, in most of these datasets, with respect to the current in-domain systems (SOTA) reported in LAraBench. Fanar ASR is primarily an Arabic-centric speech recognition model. However, it demonstrates a satisfactory performance on English benchmarks, achieving a WER of 7.5% on Libri-Clean, and 12% on Libri-Other. Remarkably, this performance is achieved while utilizing only 25% of the model's vocabulary tokens, highlighting the model's capability of transcribing English in this constrained setting.

Text to Speech Generation   To improve accessibility, Fanar integrates the state-of-the-art in-house Arabic text-to-speech (TTS) model (based on (Abdelali et al., 2022)) following recent flow-based diffusion model architechture supporting in-context learning. The model outputs synthesized speech that is natural, culturally adaptive, and intelligible to a diverse audience.

Model Design: To design the TTS, we opt for a fully non-autoregressive architecture that offers human-level naturalness and state-of-the-art speaker similarity and intelligibility with only small reference speaker audio. We adopted a flow matching strategy for generating mel spectrogram using diffusion transformer model (DiT) (Chen et al., 2024b; Eskimez et al., 2024).

To generate the spoken response from Fanar, we first diacritize the input prompt using Farasa diacretizer (Abdelali et al., 2016). Next, the diacritized text is converted into a sequence of characters and padded with filler tokens to match the length of the input speech. Given the reference mel spectrogram of the speaker (audio prompt) and the text representation, the model exploit ConvnextV2 (Woo et al., 2023) to address text-speech alignment within the in-context learning framework. The aligned representation is then passed through DiT blocks to generate the text and reference audio conditioned mel spectrogram. We then converted the mel spectrogram into the final raw speech output using the vocoder (Siuzdak, 2023). Unlike traditional approaches, our adapted architecture eliminated the need for duration models, text encoders, or phoneme alignment, making it well suited for phonetically rich and diverse languages like Arabic.

9.2. Image Modality

We developed the image generation model of Fanar by fine-tuning Stable Cascade (Pernias et al., 2023). Compared to other open-source projects, Stable Cascade has satisfactory characteristics, including generation quality, generation speed, configurability, documentation and license.

Stable Cascade operates at three steps. The first step is stage C where given a textual prompt, a small (typically 12×12) image is created with tens of channels. This image is given to stage B which upscales the image to 256×256 with 4 channels. Stage B is not conditioned on the prompt. Finally, the output of stage B is given to stage A which upscales the image to 1024×1024 with three channels. The models for the three stages C, B and A have respectively 3.6B, 700M and 20M parameters. Stages B and A primarily increasing image resolution. We focus on fine tuning stage C to achieve our desired results.

Stable Cascade works with English Prompts. We support Arabic and English. If we are given an Arabic prompt, we query the model with the English translation of the prompt. If the prompt is in English, we use the same prompt. We augment the text prompt with some textual descriptions to help increase the quality and the details of the result.

Problems with Stable Cascade: We observe two problems with publicly available open source models. We refer to these problems as the problems of knowledge and preferences. The problem of knowledge is similar to variance problem which refers to lack of knowledge about certain topics of interest. The problem of preference refers to the bias of the model and the training data.

Problem of Knowledge: The base Stable Cascade model is trained using LAION-5B (Schuhmann et al., 2022) dataset. In this dataset, many topics familiar in the Western culture are represented. However, some concepts that are familiar in Qatar and the broader Middle East are under represented in this dataset. For example, the model generates a more elaborate image for "American children playing basketball" than for "Qatari children playing basketball". In practice, the base Stable Cascade model generates higher quality and more accurate images in the American context, while they do poorly in the Qatari and Middle Eastern context. Since the knowledge about Arab culture is represented in only a very small fraction of training data, even if the size of training data is increased, the model cannot desirably learn the local topics. This is because the limited capacity of the model is allocated proportionally to the training data. Therefore, if we increase the size of data, still certain concepts will be inevitably under-represented in the training data.

Problem of Preference: Knowledge and preference are often correlated, however, this is not always the case. In human psychology, familiarity heuristic (Tversky and Kahneman, 1974) or "information bias" is a useful heuristic where concepts that are more familiar, are also generally preferred more. Even though

this heuristic is helpful, it can also lead to some adverse effects, including discrimination, archetypes, misalignment, and safety issues. For example, a person may be familiar with violence, but does not prefer it. We observe that no matter how we train our model, this information bias will persist unless we explicitly enforce preferences.

Solution to the Knowledge problem: No matter how we increase data size, certain important concepts will be under-represented in the training data. To make sure we exhaustively cover important cultural concepts we took a principled approach. We prepared a taxonomy of visual concepts specific to the Arab world and the Middle East. This taxonomy tree covers familiar visual concepts including clothing, architecture, natural scenes, familiar people, and everyday life from Arab and Middle Eastern countries. To make sure important concepts are not missed in the training data, our taxonomy has four layers. For example, we have: Food and Drink → Beverages → Pakistan → Falooda. For each concept within this taxonomy tree, we collected several related images. Using this process, we had more fine grained control over the topics that we wanted our model to be familiar with.

To create the taxonomy tree, we used Wikipedia articles from multiple languages and performed some manual adjustment. Our taxonomy tree includes more than 5000 visual concepts in four layers of abstraction. We used the elements of this taxonomy tree to obtain images which we use to create a dataset of about 200,000 images with captions. We then applied quality filters (e.g., ignore low resolution or grayscale images) and reduced the data to about 100,000 high-quality images. We fine-tuned several image generation models using several different sources of data. In addition, we collected images from cultural Wikipedia pages and obtained other images from Qatari sources. For each source of data we created a separate dataset. We then fine-tuned a separate model given each dataset.

Model Averaging: Knowing the ideal proportion of different datasets in the training data is difficult before training; therefore, we used a training process to have more principled control over the mix of data. We partitioned the training data into a few categories. Then, for each category we fine tuned a separate model. Finally, we averaged the models after they were trained. Then, a group of users reviewed hundreds of generated images and assigned a quality score to each fine-tuned model. We then picked weights to each model accordingly and averaged the best performing models. The final average performs better than each individual model. Figure 16 compares the outputs of our final averaged model vs the base model.

Solution to the Preference problem: After we adjusted the distribution of our training dataset and used it to train models, there are still unwanted biases in our model. To fix these biases, we first input a list of preferences in textual format in the form of "A should B" or "A should not B". (For example "Muslims should not drink alcohol"). Then we create three textual prompts for image generation: neutral prompt, negative prompt, and positive prompt. Neutral prompt contains A and its variations. Negative prompt contains A and not B. Positive prompt contains A and B. Then we generate three images using the three prompts using the same Stable Cascade model. Finally, we use DPO to train the model to prefer "A and B" over "A and not B" given prompt A.
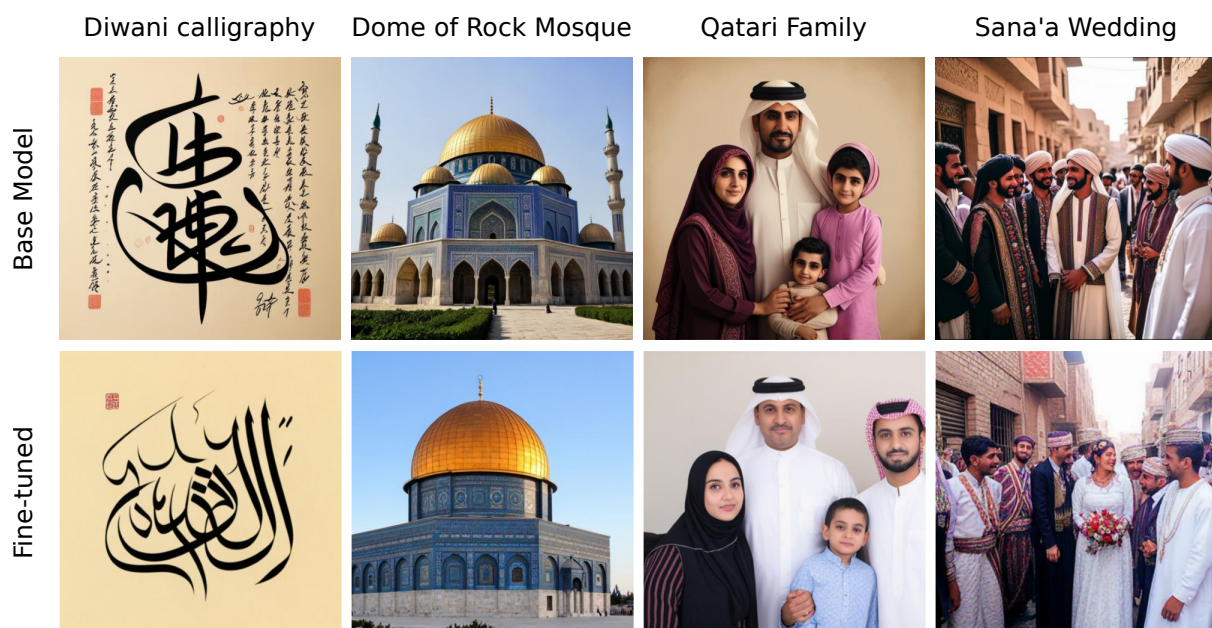
Figure 16: Comparison between images generated by the base Stable Cascade model versus our fined-tuned version. Column names indicate input prompt. The full prompt for the right-most column is: "A picture of a wedding party in the streets of old Sana'a and a group of people are happy and wearing traditional dress". Notice that the images from the fine-tuned model are more culturally nuanced compared to the original counterparts. The corresponding images are generated using identical initializations.

10. Retrieval Augmented Generation

> The various RAG systems used in Fanar are explained. These systems handle prompts related to specific domains in order to improve the accuracy of model-generated content.

Fanar extensively uses Retrieval Augmented Generation (RAG) in various forms. RAGs are used for generating Islamic content, biographical information, accessing recent information and for attributing fact-related prompts.

10.1. Islamic RAG

Islamic RAG consists of two main components. The first is a data pipeline designed to store curated texts from Islamic literature within a vector database (Milvus). This process involves text chunking, where each document is divided into segments of up to 2048 tokens, with a 50-token overlap between consecutive segments. Chunking is performed while ensuring that splits occur at semantic boundaries (e.g., paragraphs or lines) to preserve coherence. The hyperparameters of the RAG system, including the embeddings model, chunk size, re-ranking strategy, and others, were fine-tuned using carefully selected domain-specific benchmark datasets. The second component is the prompt engine. When a prompt is submitted, a similarity search is conducted within the vector database to retrieve the top twenty documents that match the prompt and exceed a predefined similarity threshold. This threshold was calibrated using the same benchmark datasets. The retrieved documents are subsequently processed by a re-ranking model, which selects the top four documents based on relevance. These documents are further filtered to fit within the context window of the language model. Finally, a prompt is constructed and sent to Fanar, instructing it to generate a response based on the provided documents. The prompt design was optimized to maximize context utilization and minimize out-of-context references.

10.2. Recency and Biography RAG

An orchestrator uses a classifier to determine whether a prompt is related to recent information, e.g., "what is the weather in Doha", or biographical information, e.g., "who is Mohamed Salah,". The prompt is routed to the Recency or the Biography RAG where it rewritten as search query by **Fanar Prime** and relevant information is accessed through a search engine. The results of the search are provided to **Fanar Prime** which summarizes the information and creates a response with the included source.

10.3. Attribution RAG

Attribution refers to the capacity of an LLM model to generate and provide evidence, often in the form of references or citations, that substantiates the claims or statements it produces (Bohnet et al., 2022). This evidence is derived from identifiable sources, such as webpages with unique URLs, ensuring that the claims can be logically inferred, comprehended, and verified by LLM users (e.g., chatbot users). LLM attribution is used to enable users to validate the claims made by the model, promote the generation of responses that align with the cited sources, and to establish a framework for evaluating the completeness and relevance of the supporting evidence in relation to the presented claims (Li et al., 2023a).

There are three main types of LLM attribution (Li et al., 2023a): (1) direct model-driven attribution (Sun et al., 2022), where the model itself provides the attribution for its generated responses, (2) post-retrieval answering (Xu et al., 2023), where information is explicitly retrieved from specific references and then the model is conditioned on this information to generate responses, and (3) post-generation attribution (Gao et al., 2022), where the generated response is used along with the user's query to search for supporting references from a large corpus, after which the response is modified and attributed using the references.

We decided to avoid making direct modifications to Fanar models, as that may affect the way they generate responses and other down-stream tasks (e.g., model evaluation, guardrails). As such, attribution in Fanar is an opt-in feature and uses post-generation attribution, making it a stand-alone, independent service. In particular, we view attribution as a post-generation RAG task that performs fact-checking and revision of Fanar responses. It takes in a user query and Fanar's response to the query, and then uses an LLM model to generate a revised response with references based on top-k most relevant documents from a corpus of facts-related documents (e.g., Wikipedia articles, Data Commons factoids). In concept, attribution in Fanar implements a research-then-revise methodology (Gao et al., 2022) using prompt engineering, where relevant evidence is retrieved from the corpus for attribution and the response is minimally modified to make it consistent with the evidence while preserving its properties, such as style and structure.

11. Discussion and Future Plans

In this work we have presented our first release of Fanar, an Arabic-centric multimodal generative AI platform. The core capabilities of Fanar are encapsulated in **Fanar Star** and **Fanar Prime**, two LLMs that operate concurrently, each responsible for answering different types of prompts. **Fanar Star** is a 7B parameter model trained from scratch on nearly 1 trillion tokens including 300 billion tokens in original Arabic and nearly 50 billion tokens in translated MSA and dialectal Arabic to enhance coverage. The **Fanar Prime** model is continually trained on the base Gemma-2 9B model with the same 1 trillion set. In addition, the Fanar platform provides multimodal capability in speech and image generation and four RAG systems for Islamic content, recent information, factual content attribution and select biographies. An overarching orchestrator coordinates all the Fanar services and enables safety filters to verify content at the prompt and response level. During the course of the project we have learned several lessons and identified several directions for future research and development.

1. Lack of Arabic data (both MSA and dialectal) that has both broad coverage and clean is the biggest bottleneck towards building a very large scale Arabic-centric LLM. Coupling **Fanar Star** with **Fanar Prime** is our pragmatic approach to take advantage of the coverage of English data while maintaining Arabic characteristics. On Arabic-centric benchmarks, results indeed show that our solution is robust. With human testers the Fanar "dislike" rate for prompts is around 15% and most of it due to factuality related errors which is an indicator that we need to scale to bigger

models. We have further attempted to reduce errors by extensively using RAGs and supporting attribution. However new approaches maybe required to increase the confidence of using LLMs for critical tasks.

2. While the current version of Fanar platform has an orchestrator module to coordinate the prompts, it is not agentic. Fanar is reactive rather than pro-active whereby it can reason and plan with minimal guidance and pursue goals autonomously. The next generation of Fanar will include forms of reasoning and verification capabilities either endogenously or through tool calling. An emerging area is to explore the use of enhanced test time computation for hard tasks (e.g., math reasoning) where multiple responses are generated and an external reward models selects the best answer (Snell et al., 2024).

3. The Fanar platform has multimodal capabilities but they are not intrinsic to the autoregressive model with image and speech tokens that can be combined to perform the appropriate multimodality task. For the next version, we plan to make speech, image and text generation as part of a unified generative model. We will also provide support for video generation that is both culturally appropriate and efficient.

4. The first release of the Fanar platform is accompanied by a small class of applications in education, news summarization and a chatbot to access government information. Our goal is to integrate Fanar at the enterprise level including in the government and the private sector. For Fanar to evolve and thrive, real use cases with visible productivity benefits will have to drive its evolution.

5. Finally, we return to the data issue. In the current landscape of GenAI, web publishers are increasingly grappling with how to maintain free and open access to their content while ensuring fair compensation. With the rise of LLM-driven tools such as ChatGPT, publishers often see diminished traffic to their websites, leading to reduced advertisement revenue. Moreover, legal questions are emerging regarding whether LLM providers should pay for access to copyrighted content that are used to train their models. As a result, over 5% of all Web data and 25% of high-quality Web sources in AI training are now restricted, mainly due to publishers putting their content behind paywalls and login screens (Longpre et al., 2024). QCRI has launched a parallel project, **TokenX**, to address the data issue in a principled, market-driven manner. In particular, TokenX provides a new service that leverages blockchain technology and LLM attribution (see §10.3) to incentivize publishers to keep their content openly-accessible by paying them whenever the content is used for attribution by subscribed LLM providers. We aim to use TokenX to encourage Arabic speakers to generate content both in MSA and dialects across a wide range of disciplines and possibly create an Arabic language renaissance.

A. Contributions

The Fanar project was divided into work packages and sub-projects and the specific contributions of the authors are listed below. The names within each sub-group are ordered alphabetically.

Data Collection and Quality: Hamdy Mubarak (lead), Mohammad Shahmeer Ahmad, Sabri Boughorbel, Abubakr Mohamed, Tasnim Mohiuddin, Ahmad Musleh, Zan Naeem, Omar Sinan, Yifan Zhang
Pre-Training (Fanar Star): Sabri Boughorbel (lead), Mohammad Shahmeer Ahmad, Fahim Dalvi, Tasnim Mohiuddin, Zan Naeem, Amin Sadeghi
Pre-Training (Fanar Prime): Fahim Dalvi (lead), Sabri Boughorbel, Tasnim Mohiuddin, Amin Sadeghi
Post-Training: Husrev Taha Sencar (lead), Enes Altinisik, Masoomali Fatehkia
Benchmarking and Evaluation: Majd Hawasly and Ehsannedin Asgari (leads), Sabri Boughorbel, Fahim Dalvi, Mus'ab Husaini, Hamdy Mubarak
Fanar Morphology-based Tokenizer: Ehsannedin Asgari (lead)
Machine Translation: Nadir Durrani (lead), Ummar Abbas, Fahim Dalvi, Kareem Darwish, Basel Mousi, Mohammed Shinoy
Image Generation: Amin Sadeghi (lead), Ehsannedin Asgari, Kareem Darwish, Abubakr Mohamed, Hamdy Mubarak, Zan Naeem
Speech: Shammur Chowdhury (lead), Mohammad Shahmeer Ahmad, Kareem Darwish, Abubakr Mohamed, Hamdy Mubarak
Applications: Kareem Darwish (lead), Mohammad Shahmeer Ahmad, Yifan Zhang
Engineering and Deployment: Mohamed Elfeky (lead), Ummar Abbas, Fahim Dalvi, Mus'ab Husaini, Soon-Gyo Jung, Ji Kim Lucas, Yifan Zhang
Islamic RAG: Ummar Abbas (lead), Kareem Darwish, Mohammed Qusay Hashim, Mus'ab Husaini, Walid Magdy, Hamdy Mubarak, Mourad Ouzzani, Omar Sinan, Mohammed Shinoy
Attribution: Yazan Boshmaf (lead), Soon-Gyo Jung, Ji Kim Lucas, Yifan Zhang
Infrastructure: Anastasios Fragkopoulos (lead)
External User Study and System testing: Hamdy Mubarak and Majd Hawasly (leads)
Scientific and Management Leadership: Mohamed Eltabakh (lead), Sanjay Chawla, Ahmed Elmagarmid, Mourad Ouzzani

A.1. Acknowledgments

B. Arabic-Speaking Map and Language Statistics

(a) Overview of Arabic-speaking Countries and Regions in the Middle East, North Africa, and the Arabian Peninsula (MENA Region).
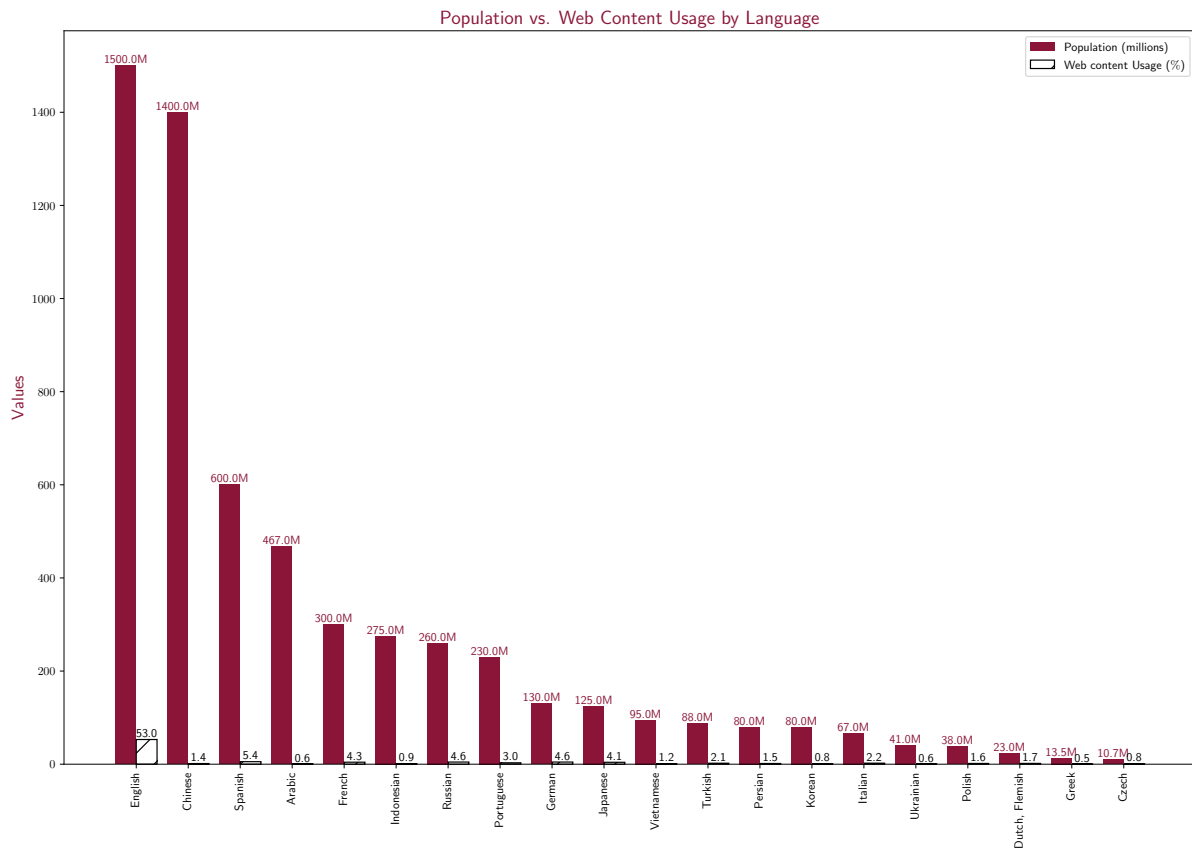
(b) Contrasting the language population and web-data availability of different languages.

Figure 17: (a) Overview of Arabic-speaking countries; (b) Population vs. web content statistics for the top 10 most popular languages in the world.

C. In-house Benchmarks

C.1. Almieyar: capability-based benchmarking

The evaluation of LLMs has emerged as a critical research focus, frequently discussed at recent ACL and ML conferences, particularly in 2023 and 2024. Answering benchmark questions in many existing evaluation datasets, such as MMLU (OpenAI, 2024), requires a mix of skills. For example, solving a logic puzzle often requires a combination of linguistic proficiency, logical reasoning, and domain expertise. However, this entanglement makes it challenging to provide clear feedback on the specific strengths and weaknesses of the models. In our view, a diverse set of essential capabilities should be benchmarked for LLMs. To this end, we introduce Al-Mieyar, a family of capability-focused benchmarks that target Arabic as primary languages/culture, while also providing a recipe that can be adapted to any language/culture of interest. Al-Mieyar targets a set of LLM-related capabilities, which will be introduced in a separate report. Here, we focus specifically on its language capabilities segment, Almieyar-Language.

Almieyar-Recipe: The creation of the Almieyar dataset involved three steps: (i) template-based question generation using an LLM guided by a refined taxonomy, (ii) human review by native Arabic speakers for linguistic and cultural accuracy, and (iii) linguistic validation by professionals to ensure quality.

Almieyar-Language: despite advances in benchmarks targeting common sense and logical reasoning, relatively few focus exclusively on linguistic understanding. To the best of our knowledge, *Holmes* is one of the few datasets in this area (Waldis et al., 2024), but it focuses solely on English and lacks detailed taxonomies for linguistic evaluation. Furthermore, it excludes important linguistic categories, such as phonology, and provides no resources for non-English languages. To address these limitations, we introduce **Almieyar-language**, a benchmark designed to evaluate linguistic understanding in Arabic, along with a recipe for extending it to other languages. *Almieyar-language* is grounded in linguistic capabilities and offers a structured approach to dataset generation and evaluation.

Linguistic theory classifies language knowledge into five core layers: phonology, morphology, syntax, semantics, and pragmatics. We developed a comprehensive taxonomy that encompasses these layers from a universal perspective, with a particular focus on the unique characteristics of Arabic. This focus is informed by the specific features of Arabic outlined in (Habash, 2010) for each of the five layers. The taxonomy of linguistic capabilities in Arabic is summarized in Figure 18.

By addressing Arabic's rich dialectal diversity, with contributions from speakers of 16 dialects, Almieyar-language emphasizes lexical semantics and pragmatics, including dialect-specific elements. The evaluation of the Fanar model on the "Almieyar-language" dataset, in comparison to other prominent models, provides detailed insights into its performance across the five core linguistic layers, assessed over a high-quality set of approximately one thousand questions, drawn from diverse linguistic categories.

Almieyar-Overall: This dataset contains high-quality and manually reviewed benchmarking questions on knowledge, reasoning, alignment, and robustness, in Arabic is another segment of Almieyar developed in-house to provide quick feedback on a diverse range of topics following SFT training. Figure 19 illustrates the hierarchical structure of the "**Almieyar-Overall**" segment. "**Almieyar-Culture**" and "**Almieyar-Multimodality**" are also among the segments that will be detailed in a separate report.

C.2. Arab Cultural MCQ

To help bridge the gap in high-quality benchmarks for Arabic culture, we created a new multi-choice question dataset that addresses cultural issues, values and nuances in Arab countries. The dataset comprises 1K questions that were curated carefully from a corpus of 7.5K generated automatically by GPT-4o. The LLM was presented with relevant web pages that contain high-quality content about Arabic culture[17], and instructed to create a number of multi-choice questions that revolve around the input text.

Then, the questions went through three stages of thorough validation and quality control:

---

[17]E.g., https://culturecrossing.net/, https://www.expatica.com/, Wikipedia culture pages.

1. An expert reviewed the language and the factuality of each question from the 7.5K corpus. In certain cases, the question body or some of the choices were changed by the expert.

2. Another round of review selected questions that are relevant to the Arabic culture, not particularly easy or obvious, and that have meaningful answer choices.

3. Lastly, semantic clustering was performed on the reviewed questions. This used `bge-multilingual-gemma2`[18], an LLM-based multilingual embedding model with good performance in Arabic (Chen et al., 2024a). The questions with the highest similarity were reviewed manually again to remove the redundant questions, resulting in 1K questions.

Below are a few examples from the culture questions development set.

1.

هذا سؤال عن الثقافة والعادات في السودان. ما اسم الشراب الذي يتم تحضيره في رمضان في السودان ويعتبر مميزًا؟

ـ العصير الطبيعي

ـ الكركديه

ـ التبلدي

ـ الحلو مر

2.

هذا سؤال عن الثقافة والعادات في قطر. في أي مناسبات ينشد النهام أغانيه؟

ـ أثناء الرحلات البحرية

ـ في الحفلات الموسيقية

ـ في الاجتماعات العائلية

ـ خلال فعاليات اجتماعية عامة

3.

هذا سؤال عن الثقافة والعادات في اليمن. ما هي طريقة الصباغة الأكثر شهرة في اليمن؟

ـ التلوين الحر
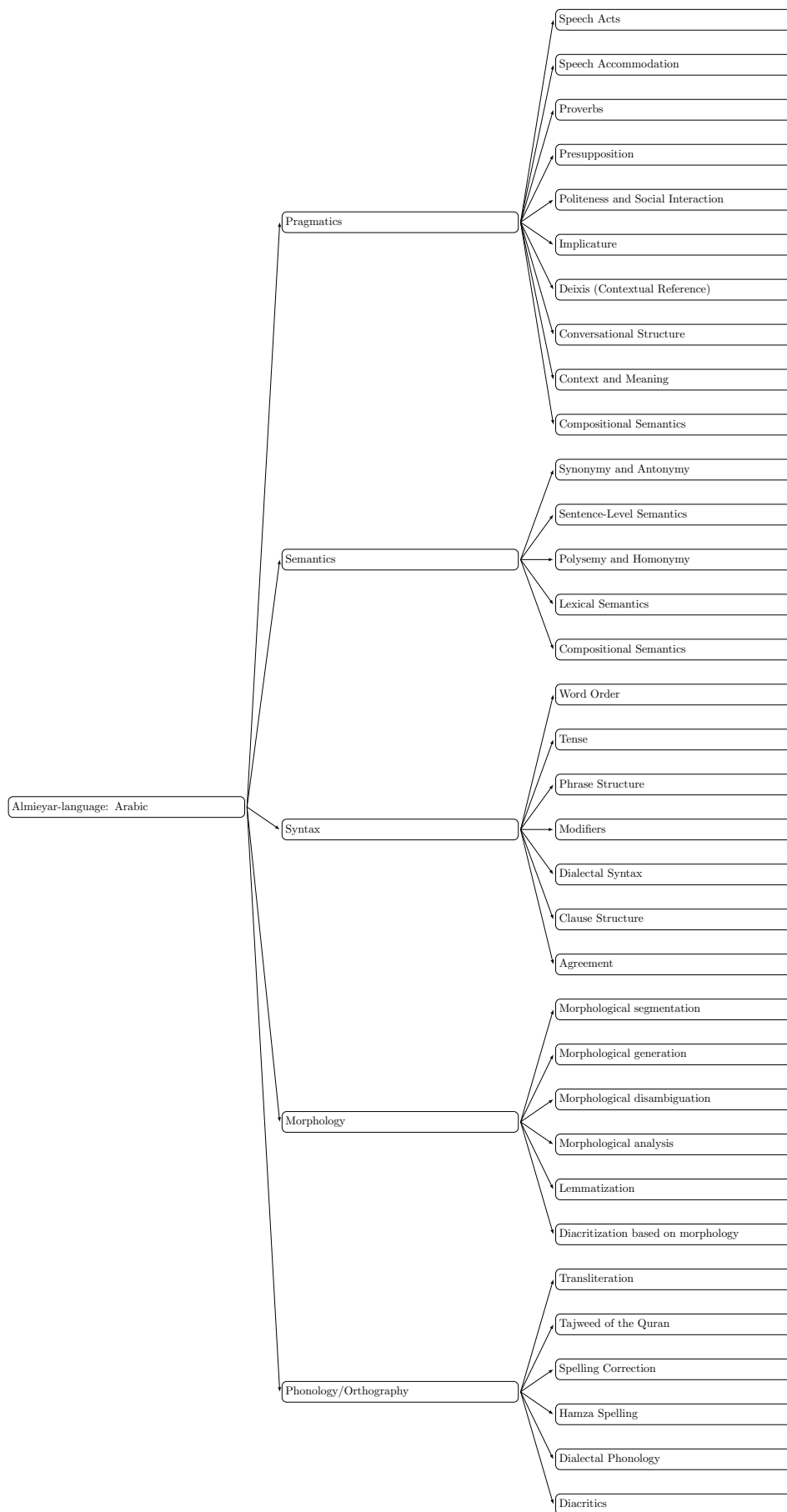
ـ الرش

ـ الوصايل

ـ الغمر

Figure 18: Hierarchical representation of Almieyar-language categories and sub-categories in Arabic: Phonology/Orthography, Morphology, Syntax, Semantics, and Pragmatics. Each category is further detailed with its sub-categories.
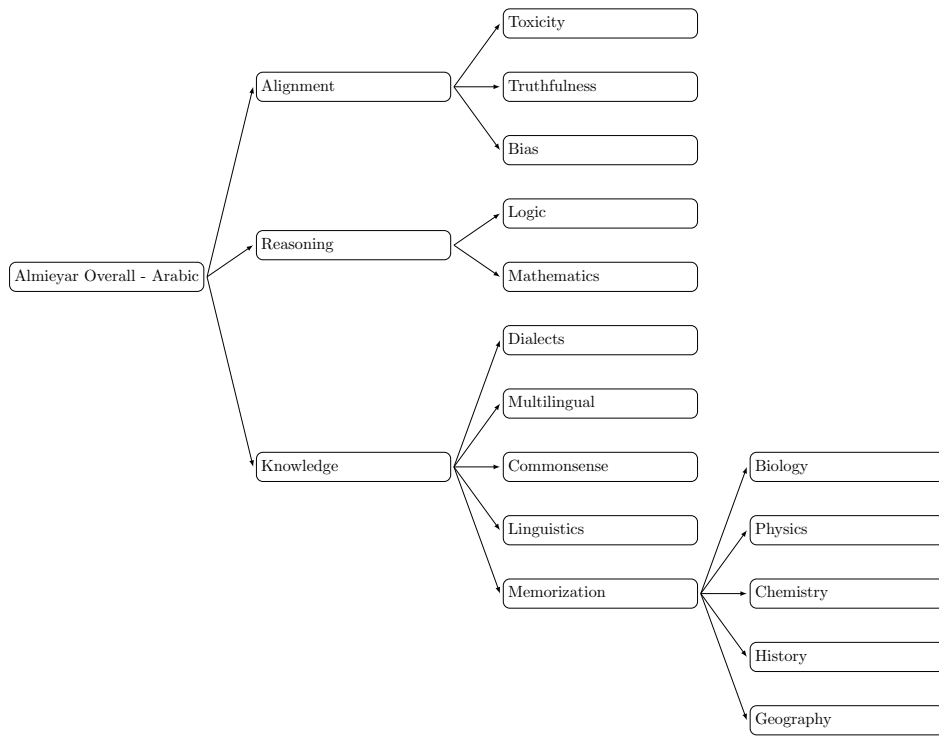
Figure 19: Hierarchical structure of the "Almieyar-Overall" segment, encompassing high-quality benchmarking questions on knowledge, reasoning, alignment, developed to provide quick feedback following SFT training.

D. LLM Security and Safety

LLMs, like any other software system, are vulnerable to attacks from malicious adversaries. At QCRI we have a parallel project that evaluates the safety of LLMs (Khalil et al., 2025). **aiXamine** includes over 30 tests that cover a range of security threats and safety vulnerabilities. The tests evaluate both data and model safety. For data vulnerabilities, these tests can detect issues such as poisoned samples that inject backdoors, biases or noise that compromise performance, and samples that may violate copyright laws. For models, tests are designed to evaluate robustness against adversarial and privacy attacks, detect backdoors and unfair or biased behavior, check the model's tendency to produce hallucinations and refuse innocuous requests, evaluate performance on out-of-distribution examples, verify watermarks, and assess content safety to prevent harmful or inappropriate outputs in the form of text or code. Table 14 shows the results of **aiXamine** on select open source models and how they compare against **Fanar Star**.

Table 14 Comparison of scores achieved by different models across the different aiXamine services.

Model Name	Adversarial Robustness	Code Security	Model & Data Privacy	OOD Robustness	Safety & Alignment	Average Score
Fanar-7B	58.24	**85.8**	81.14	89.6	97.68	**82.49**
LLaMA-3.2-3B-Instruct	43.38	49.85	**99.82**	86.07	95.14	74.85
LLaMA-3.1-8B-Instruct	57.48	49.1	77.05	87.63	97.07	73.67
Qwen-2-7B-Instruct	**61.78**	46.18	-	90.65	95.32	73.48
Llama-2-7b-chat-hf	51.24	48.87	79.07	85.93	**99.35**	72.89
Llama-3-8B-Instruct	49.97	48.37	74.53	**91.40**	97.52	72.36
Mistral-7B-Instruct-v0.1	45.00	46.00	-	88.47	71.28	62.69
Mistral-7B-Instruct-v0.2	46.02	45.59	-	70.56	84.44	61.65
gemma-2-9b-it	16.62	50.53	96.00	49.62	92.57	61.07

E. Arabic Summary

في هذا التقرير، نقدم التفاصيل الفنية لمشروع فنار، وهو منصة متخصصة في تطوير أنظمة الذكاء الاصطناعي التوليدي متعددة الوسائط التي تركز على اللغة العربية، وتدعم مهام توليد النصوص والصوت والصور. المنصة مبنية على نموذجين لغويين كبيرين هما «فنار من الصفر» و «فنار مع التدريب المستمر»، اللذان أثبتا كفاءتهما على مقاييس أداء قياسية مقارنة بنماذج مماثلة في الحجم.

ـ «فنار من الصفر»: نموذج يحتوي على ٧ مليارات مُعلَّمة، وتم تدريبه من البداية باستخدام بيانات منقحة عالية الجودة تضم ما يقارب تريليون رمز عربي وإنجليزي وبرمجي.

ـ «فنار مع التدريب المستمر»: نموذج يحتوي على ٩ مليارات مُعلَّمة من نموذج جيمّا ٢ الأساسي ثم تدريبه بعد ذلك على نفس مجموعة البيانات.

النموذجان يعملان بشكل متزامن عبر منظومة مخصصة لتوجيه مدخلات المستخدم إلى أي منهما.

توفر منصة فنار قدرات إضافية، مثل نظام استرجاع معزز مخصص للأسئلة الدينية، ونظام استرجاع للإجابة على الأسئلة المتعلقة بالأحداث الجارية التي وقعت بعد تاريخ انتهاء بيانات التدريب. كما تدعم المنصة تقنيات الوسائط المتعددة وتشمل التعرف على الصوت ثنائي اللغة (عربي وإنجليزي)، مع دعم للهجات عربية متعددة، بالإضافة إلى توليد الصوت والصور. تم مراعاة أن تكون الأجوبة والصور المولدة ملائمة للثقافة العربية وبما يعكس الهوية العربية بشكل دقيق. تقدم المنصة أيضا خدمة التحقق من صحة الأجوبة وإعطاء المراجع اللازمة لضمان دقة المعلومات.

تم تطوير فنار بالكامل في معهد قطر لبحوث الحوسبة التابع لجامعة حمد بن خليفة، بدعم من وزارة الاتصالات وتكنولوجيا المعلومات في قطر، بهدف تعزيز تقنيات الذكاء الاصطناعي وتطويعها لخدمة المستخدم العربي.

References

Abdelali, A., Darwish, K., Durrani, N. et al (2016). Farasa: A fast and furious segmenter for Arabic. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 11–16, San Diego, California. Association for Computational Linguistics.

Abdelali, A., Durrani, N., Demiroglu, C. et al (2022). Natiq: An end-to-end text-to-speech system for arabic. In *Proceedings of the Seventh Arabic Natural Language Processing Workshop*, pages 394–398, Abu Dhabi, UAE. Association for Computational Linguistics.

Abdelali, A., Mubarak, H., Chowdhury, S.A. et al (2024). LAraBench: Benchmarking Arabic AI with Large Language Models. In Graham, Y. and Purver, M., editors, *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 487–520, St. Julian's, Malta. Association for Computational Linguistics.

Adel, H., Asgari, E. and Schütze, H. (2018). Overview of character-based models for natural language processing. In Gelbukh, A., editor, *Computational Linguistics and Intelligent Text Processing: 18th International Conference, CICLing 2017, Budapest, Hungary, April 17–23, 2017, Revised Selected Papers, Part I 18*, pages 3–16, Cham. Springer International Publishing.

Adler, B., Agarwal, N., Aithal, A. et al (2024). Nemotron-4 340b technical report. *arXiv preprint arXiv:2406.11704*.

Al-Wer, E. and de Jong, R. (2017). Dialects of Arabic. *The handbook of dialectology*, pages 523–534.

Alharbi, S., Alowisheq, A., Tüske, Z. et al (2024). SADA: Saudi audio dataset for Arabic. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 10286–10290. IEEE.

Ali, A., Chowdhury, S., Hussein, A. et al (2021). Arabic Code-Switching Speech Recognition using Monolingual Data. In *Proc. of the 22nd Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 3475–3479, Brno, Czechia.

Ali, A., Shon, S., Samih, Y. et al (2019). The MGB-5 challenge: Recognition and dialect identification of dialectal Arabic speech. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 1026–1033. IEEE.

Ali, A., Vogel, S. and Renals, S. (2017). Speech recognition challenge in the wild: Arabic MGB-3. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 316–322. IEEE.

AllenAI (2024). OLMo Family Model Card. https://huggingface.co/collections/allenai/olmo-suite-65aeaae8fe5b6b2122b46778. Accessed on Dec 1, 2024.

Alnosairee, A. and Sartini, N.W. (2021). A sociolinguistics study in arabic dialects. *PRASASTI: Journal of Linguistics*, 6:1–17.

Alolaywi, Y. (2022). Derivation between english and arabic with reference to translation: a contrastive analysis study. *Arab World English Journal for Translation and Literary Studies*, 6:128–141.

Asgari, E., McHardy, A.C. and Mofrad, M.R. (2019). Probabilistic variable-length segmentation of protein sequences for discriminative motif discovery (DiMotif) and sequence embedding (ProtVecX). *Scientific reports*, 9(1):3577.

Asgari, E., Sabet, M.J., Dufter, P. et al (2020). Subword sampling for low resource word alignment. *arXiv preprint arXiv:2012.11657*.

Azar, M.G., Guo, Z.D., Piot, B. et al (2024). A general theoretical paradigm to understand learning from human preferences. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, pages 4447–4455. PMLR.

Azerbayev, Z., Schoelkopf, H., Paster, K. et al (2024). Llemma: An open language model for mathematics. In *The Twelfth International Conference on Learning Representations*.

Bandarkar, L., Liang, D., Muller, B. et al (2024). The belebele benchmark: a parallel reading comprehension

dataset in 122 language variants. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 749–775, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Barham, S., Weller, O., Yuan, M. et al (2023). Megawika: Millions of reports and their sources across 50 diverse languages. *arXiv preprint arXiv:2307.07049*.

Bari, M.S., Alnumay, Y., Alzahrani, N.A. et al (2024). ALLaM: Large Language Models for Arabic and English. *arXiv preprint arXiv:2407.15390*.

Baumgartner, J., Zannettou, S., Keegan, B. et al (2020). The pushshift Reddit dataset. *arXiv preprint arXiv:2001.08435*.

Bisk, Y., Zellers, R., Gao, J. et al (2020). PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Blakeney, C., Paul, M., Larsen, B.W. et al (2024). Does your data spark joy? performance gains from domain upsampling at the end of training. In *First Conference on Language Modeling*.

Bohnet, B., Tran, V.Q., Verga, P. et al (2022). Attributed Question Answering: Evaluation and Modeling for Attributed Large Language Models. *arXiv preprint arXiv:2212.08037*.

Boughorbel, S. and Hawasly, M. (2023). Analyzing Multilingual Competency of LLMs in Multi-Turn Instruction Following: A Case Study of Arabic. In Sawaf, H., El-Beltagy, S., Zaghouani, W. et al, editors, *Proceedings of ArabicNLP 2023*, pages 128–139, Singapore (Hybrid). Association for Computational Linguistics.

Broder, A. (1997). On the resemblance and containment of documents. In *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No.97TB100171)*, pages 21–29. IEEE.

Brown, T.B., Mann, B., Ryder, N. et al (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Chen, J., Xiao, S., Zhang, P. et al (2024a). BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. *arXiv preprint arXiv:2402.03216*.

Chen, Y., Niu, Z., Ma, Z. et al (2024b). F5-TTS: A fairytaler that fakes fluent and faithful speech with flow matching. *arXiv preprint arXiv:2410.06885*.

Chowdhury, S.A., Hussein, A., Abdelali, A. et al (2021). Towards One Model to Rule All: Multilingual Strategy for Dialectal Code-Switching Arabic ASR. In *Proc. of the 22nd Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 2466–2470, Brno, Czechia.

Chowdhury, S.A., Samih, Y., Eldesouki, M. et al (2020). Effects of dialectal code-switching on speech modules: A study using egyptian Arabic broadcast speech. In *Proc. Interspeech 2020*, pages 2382–2386, Shanghai, China.

Clark, J.H., Garrette, D., Turc, I. et al (2022). Canine: Pre-training an efficient tokenization-free encoder for language representation. *Transactions of the Association for Computational Linguistics*, 10:73–91.

Clark, P., Cowhey, I., Etzioni, O. et al (2018). Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Cloud, A. (2024). Qwen2.5 Family Model Card. https://huggingface.co/collections/Qwen/qwen25-66e81a666513e518adb90d9e. Accessed on Dec 1, 2024.

Cohere (2024). Aya Expanse Family Model Card. https://huggingface.co/collections/CohereForAI/c4ai-aya-expanse-671a83d6b2c07c692beab3c3. Accessed on Dec 1, 2024.

Cohere For AI (2024). c4ai-command-r-plus-08-2024. https://huggingface.co/CohereForAI/c4ai-command-r-plus-08-2024.

Consortium, B. (2024). Balsam. https://benchmarks.ksaa.gov.sa/b/balsam.

Csaki, Z., Li, B., Li, J.L. et al (2024). SambaLingo: Teaching large language models new languages. In Sälevä,

J. and Owodunni, A., editors, *Proceedings of the Fourth Workshop on Multilingual Representation Learning (MRL 2024)*, pages 1–21, Miami, Florida, USA. Association for Computational Linguistics.

Cui, G., Yuan, L., Ding, N. et al (2024). UltraFeedback: Boosting language models with scaled AI feedback. *arXiv preprint arXiv:2310.01377*.

Dagan, G., Synnaeve, G. and Rozière, B. (2024). Getting the most out of your tokenizer for pre-training and domain adaptation. *arXiv preprint arXiv:2402.01035*.

Deiseroth, B., Brack, M., Schramowski, P. et al (2024). T-FREE: Subword Tokenizer-Free generative LLMs via sparse representations for memory-efficient embeddings. In Al-Onaizan, Y., Bansal, M. and Chen, Y.N., editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21829–21851, Miami, Florida, USA. Association for Computational Linguistics.

Dodge, J., Sap, M., Marasović, A. et al (2021). Documenting large webtext corpora: A case study on the colossal clean crawled corpus. *arXiv preprint arXiv:2104.08758*.

Dubey, A., Jauhri, A., Pandey, A. et al (2024). The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Durrani, N., Dalvi, F., Sajjad, H. et al (2019). One size does not fit all: Comparing NMT representations of different granularities. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1504–1516, Minneapolis, Minnesota. Association for Computational Linguistics.

Elfilali, A., Alobeidli, H., Fourrier, C. et al (2024). Open arabic llm leaderboard. `https://huggingface.co/spaces/OALL/Open-Arabic-LLM-Leaderboard`.

Elmagarmid, A.K., Ipeirotis, P.G. and Verykios, V.S. (2006). Duplicate record detection: A survey. *IEEE Transactions on knowledge and data engineering*, 19(1):1–16.

Eskimez, S.E., Wang, X., Thakker, M. et al (2024). E2 tts: Embarrassingly easy fully non-autoregressive zero-shot tts. *arXiv preprint arXiv:2406.18009*.

Ethayarajh, K., Xu, W., Muennighoff, N. et al (2024). Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*.

Farghaly, A. and Shaalan, K. (2009). Arabic natural language processing: Challenges and solutions. *ACM Transactions on Asian Language Information Processing (TALIP)*, 8(4):1–22.

Ferguson, C.A. (1959). Diglossia. *WORD*, 15(2):325–340.

Fourrier, C., Habib, N., Wolf, T. et al (2023). LightEval: A lightweight framework for LLM evaluation. `https://github.com/huggingface/lighteval`.

FreedomIntelligence (2024). AceGPT-v2 Family Model Card. `https://huggingface.co/collections/FreedomIntelligence/acegpt-v2-670e00e5f9a7754078b8dcf2`. Accessed on Dec 1, 2024.

Gao, L., Dai, Z., Pasupat, P. et al (2022). RARR: Researching and Revising What Language Models Say, Using Language Models. *arXiv preprint arXiv:2210.08726*.

Gao, L., Tow, J., Abbasi, B. et al (2024). A framework for few-shot language model evaluation. `https://zenodo.org/records/12608602`.

Google (2024). Gemma 2 Family Model Card. `https://huggingface.co/collections/google/gemma-2-release-667d6600fd5220e7b967f315`. Accessed on Dec 1, 2024.

Gregory, L., Thomure, H.T., Kazem, A. et al (2021). Advancing arabic language teaching and learning. *World Bank*, 10:35917.

Groeneveld, D., Beltagy, I., Walsh, P. et al (2024). Olmo: Accelerating the science of language models. *arXiv preprint arXiv:2402.00838*.

Gunter, T., Wang, Z., Wang, C. et al (2024). Apple intelligence foundation language models. *arXiv preprint arXiv:2407.21075*.

Habash, N.Y. (2010). *Introduction to Arabic natural language processing.* Morgan & Claypool Publishers.

Hassan, S., Mubarak, H., Abdelali, A. et al (2021). ASAD: Arabic social media analytics and unDerstanding. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 113–118, Online. Association for Computational Linguistics.

Heafield, K. (2011). Kenlm: Faster and smaller language model queries. In *Proceedings of the sixth workshop on statistical machine translation*, pages 187–197.

Healey, J.F. and Smith, G.R. (2012). *A brief introduction to the Arabic alphabet*, volume 3. Saqi.

Hendrycks, D., Burns, C., Basart, S. et al (2020). Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Hong, J., Lee, N. and Thorne, J. (2024). Orpo: Monolithic preference optimization without reference model. In Al-Onaizan, Y., Bansal, M. and Chen, Y.N., editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11170–11189, Miami, Florida, USA. Association for Computational Linguistics.

Horesh, U. and Cotter, W.M. (2016). Current research on linguistic variation in the arabic-speaking world. *Language and Linguistics Compass*, 10:370–381.

Huang, H., Yu, F., Zhu, J. et al (2023). Acegpt, localizing large language models in arabic. *arXiv preprint arXiv:2309.12053*.

Inception (2024). Jais Family Model Card. https://huggingface.co/collections/inceptionai/jais-family-66add8bb9c381f5492ddb6f4. Accessed on Dec 1, 2024.

Joty, S., Sajjad, H., Durrani, N. et al (2015). How to avoid unwanted pregnancies: Domain adaptation using neural network models. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1259–1270, Lisbon, Portugal. Association for Computational Linguistics.

Khalil, I., Boshmaf, Y., Deniz, F. et al (2025). aiXamine: LLM Security and Privacy, Simplified. https://aixamine.qcri.org. Accessed: 2025-01-15.

Kheir, Y.E., Mubarak, H., Ali, A. et al (2024). Beyond orthography: Automatic recovery of short vowels and dialectal sounds in Arabic. In Ku, L.W., Martins, A. and Srikumar, V., editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13172–13184, Bangkok, Thailand. Association for Computational Linguistics.

Kocetkov, D., Li, R., allal, L.B. et al (2023). The stack: 3 TB of permissively licensed source code. *Transactions on Machine Learning Research*.

Koto, F., Li, H., Shatnawi, S. et al (2024). Arabicmmlu: Assessing massive multitask language understanding in arabic. *arXiv preprint arXiv:2402.12840*.

Kudugunta, S., Caswell, I., Zhang, B. et al (2023). MADLAD-400: A multilingual and document-level large audited dataset. *arXiv preprint arXiv:2309.04662*.

Lambert, N., Morrison, J., Pyatkin, V. et al (2024). T\" ULU 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*.

Lee, K., Ippolito, D., Nystrom, A. et al (2022). Deduplicating training data makes language models better. *arXiv preprint arXiv:2107.06499*.

Lewis, P., Perez, E., Piktus, A. et al (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Li, D., Sun, Z., Hu, X. et al (2023a). A Survey of Large Language Models Attribution. *arXiv preprint arXiv:2311.03731*.

Li, J., Fang, A., Smyrnis, G. et al (2024). DataComp-LM: In search of the next generation of training sets for language models. *arXiv preprint arXiv:2406.11794*.

Li, X., Zhang, T., Dubois, Y. et al (2023b). AlpacaEval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval.

Liang, D., Gonen, H., Mao, Y. et al (2023). XLM-V: Overcoming the vocabulary bottleneck in multilingual masked language models. In Bouamor, H., Pino, J. and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13142–13152, Singapore. Association for Computational Linguistics.

Lim, J.P. and Lauw, H. (2023). Disentangling transformer language models as superposed topic models. In Bouamor, H., Pino, J. and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8646–8666, Singapore. Association for Computational Linguistics.

Liu, Y., Gu, J., Goyal, N. et al (2020). Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Logasa Bogen, P., Symons, C.T., McKenzie, A. et al (2013). Massively scalable near duplicate detection in streams of documents using MDSH . In *2013 IEEE International Conference on Big Data*, pages 480–486, Los Alamitos, CA, USA. IEEE Computer Society.

Longpre, S., Mahari, R., Lee, A.N. et al (2024). Consent in crisis: The rapid decline of the ai data commons. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Loshchilov, I. and Hutter, F. (2019). Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Lozhkov, A., Ben Allal, L., von Werra, L. et al (2024). Fineweb-edu. https://huggingface.co/datasets/HuggingFaceFW/fineweb-edu.

McOmber, M.L. (1995). Morpheme edges and arabic infixation. *AMSTERDAM STUDIES IN THE THEORY AND HISTORY OF LINGUISTIC SCIENCE SERIES 4*, pages 173–173.

Meng, Y., Xia, M. and Chen, D. (2024). Simpo: Simple preference optimization with a reference-free reward. *arXiv preprint arXiv:2405.14734*.

Meta (2024). Llama 3.1 Family Model Card. https://huggingface.co/collections/meta-llama/llama-31-669fc079a0c406a149a5738f. Accessed on Dec 1, 2024.

Mousi, B., Durrani, N., Ahmad, F. et al (2024). AraDiCE: Benchmarks for dialectal and cultural capabilities in LLMs. In *COLING 2025*. arXiv:2409.11404.

Mubarak, H. and Darwish, K. (2014a). Automatic correction of Arabic text: A cascaded approach. In Habash, N. and Vogel, S., editors, *Proceedings of the EMNLP 2014 workshop on Arabic natural language processing (ANLP)*, pages 132–136, Doha, Qatar. Association for Computational Linguistics.

Mubarak, H. and Darwish, K. (2014b). Using Twitter to collect a multi-dialectal corpus of Arabic. In Habash, N. and Vogel, S., editors, *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*, pages 1–7, Doha, Qatar. Association for Computational Linguistics.

Mubarak, H., Hussein, A., Chowdhury, S.A. et al (2021). QASR: QCRI Aljazeera Speech Resource. A Large Scale Annotated Arabic Speech Corpus. In Zong, C., Xia, F., Li, W. et al, editors, *Proc. of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2274–2285, Online. Association for Computational Linguistics.

Muennighoff, N., Rush, A., Barak, B. et al (2023). Scaling data-constrained language models. *Advances in Neural Information Processing Systems*, 36:50358–50376.

Mustafa, M., Eldeen, A.S., Bani-Ahmad, S. et al (2017). A comparative survey on arabic stemming: approaches and challenges. *Intelligent Information Management*, 09:39–67.

Nagoudi, E.M.B., Elmadany, A. and Abdul-Mageed, M. (2022a). AraT5: Text-to-text transformers for Arabic language generation. *arXiv preprint arXiv:2109.12068*.

Nagoudi, E.M.B., Elmadany, A. and Abdul-Mageed, M. (2022b). AraT5: Text-to-text transformers for Arabic

language generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 628–647, Dublin, Ireland. Association for Computational Linguistics.

OpenAI (2024). Multilingual massive multitask language understanding. `https://huggingface.co/datasets/openai/MMMLU`.

OpenAI, Achiam, J., Adler, S. et al (2024). GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Östling, R., Scherrer, Y., Tiedemann, J. et al (2017). The Helsinki neural machine translation system. In *Proceedings of the Second Conference on Machine Translation*, pages 338–347, Copenhagen, Denmark. Association for Computational Linguistics.

Otani, N., Ozaki, S., Zhao, X. et al (2020). Pre-tokenization of multi-word expressions in cross-lingual word embeddings. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4451–4464, Online. Association for Computational Linguistics.

Panayotov, V., Chen, G., Povey, D. et al (2015). Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE.

Paster, K., Santos, M.D., Azerbayev, Z. et al (2024). OpenWebMath: An open dataset of high-quality mathematical web text. In *The Twelfth International Conference on Learning Representations*.

Penedo, G., Malartic, Q., Hesslow, D. et al (2023). The RefinedWeb dataset for Falcon LLM: Outperforming curated corpora with web data only. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Pernias, P., Rampas, D., Richter, M.L. et al (2023). Wuerstchen: An efficient architecture for large-scale text-to-image diffusion models. *arXiv preprint arXiv:2306.00637*.

Provilkov, I., Emelianenko, D. and Voita, E. (2020). BPE-dropout: Simple and effective subword regularization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1882–1892, Online. Association for Computational Linguistics.

PS, D., VG, J. and MS, A. (2024). Inference acceleration for large language models on CPUs. *arXiv preprint arXiv:2406.07553*.

Qwen Team (2024). Qwen2.5: A party of foundation models. `https://qwenlm.github.io/blog/qwen2.5/`.

Rafailov, R., Sharma, A., Mitchell, E. et al (2024). Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Raffel, C., Shazeer, N., Roberts, A. et al (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1).

Ram, O., Levine, Y., Dalmedigos, I. et al (2023). In-Context Retrieval-Augmented Language Models. arXiv:2302.00083 [cs].

Rana, A. (2010). Common crawl–building an open web-scale crawl using hadoop. `https://commoncrawl.org/blog/slideshare-building-a-scalable-web-crawler-with-hadoop`. Accessed on Dec 1, 2024.

Rei, R., Stewart, C., Farinha, A.C. et al (2020). COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Riviere, G.T.M., Pathak, S., Sessa, P.G. et al (2024). Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, abs/2408.00118.

Sajjad, H., Abdelali, A., Durrani, N. et al (2020). AraBench: Benchmarking dialectal Arabic-English machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5094–5107, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Sajjad, H., Dalvi, F., Durrani, N. et al (2017a). Challenging language-dependent segmentation for Arabic: An application to machine translation and part-of-speech tagging. In *Proceedings of the 55th Annual Meeting of*

the *Association for Computational Linguistics (Volume 2: Short Papers)*, pages 601–607, Vancouver, Canada. Association for Computational Linguistics.

Sajjad, H., Durrani, N., Dalvi, F. et al (2017b). Neural machine translation training in a multi-domain scenario. In *Proceedings of the 14th International Conference on Spoken Language Translation*, pages 66–73, Tokyo, Japan. International Workshop on Spoken Language Translation.

Sakaguchi, K., Bras, R.L., Bhagavatula, C. et al (2021). Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Schuhmann, C., Beaumont, R., Vencu, R. et al (2022). LAION-5B: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*.

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

Sennrich, R., Haddow, B. and Birch, A. (2016). Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Shen, M. (2024). Rethinking data selection for supervised fine-tuning. *arXiv preprint arXiv:2402.06094*.

Shen, Z., Tao, T., Ma, L. et al (2024). Slimpajama-dc: Understanding data combinations for llm training.

Shibata, Y., Kida, T., Fukamachi, S. et al (1999). Byte pair encoding: A text compression scheme that accelerates pattern matching. *Technical Report DOI-TR-161, Department of Informatics, Kyushu University*.

Siuzdak, H. (2023). Vocos: Closing the gap between time-domain and fourier-based neural vocoders for high-quality audio synthesis. *arXiv preprint arXiv:2306.00814*.

Slack, D., Wang, J., Semenenkov, D. et al (2023). A holistic approach for test and evaluation of large language models. https://static.scale.com/uploads/6019a18f03a4ae003acb1113/test-and-evaluation.pdf. Accessed on Dec 1, 2024.

Smit, P., Virpioja, S., Grönroos, S.A. et al (2014). Morfessor 2.0: Toolkit for statistical morphological segmentation. In Wintner, S., Tadić, M. and Babych, B., editors, *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 21–24, Gothenburg, Sweden. Aalto University, Association for Computational Linguistics.

Snell, C., Lee, J., Xu, K. et al (2024). Scaling llm test-time compute optimally can be more effective than scaling model parameters.

Soldaini, L., Kinney, R., Bhagia, A. et al (2024). Dolma: an open corpus of three trillion tokens for language model pretraining research. In Ku, L.W., Martins, A. and Srikumar, V., editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15725–15788, Bangkok, Thailand. Association for Computational Linguistics.

Soldaini, L. and Lo, K. (2023). peS2o (Pretraining Efficiently on S2ORC) Dataset. Technical report, Allen Institute for AI. ODC-By, https://github.com/allenai/pes2o.

Sun, Z., Wang, X., Tay, Y. et al (2022). Recitation-Augmented Language Models. *arXiv preprint arXiv:2210.01296*.

Team, G., Mesnard, T., Hardin, C. et al (2024). Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Team, N., Costa-jussà, M.R., Cross, J. et al (2022). No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

Tokpanov, Y., Millidge, B., Glorioso, P. et al (2024). Zyda: A 1.3t dataset for open language modeling. *arXiv preprint arXiv:2406.01981*.

Touvron, H., Martin, L., Stone, K. et al (2023). Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Tversky, A. and Kahneman, D. (1974). Judgment under uncertainty: Heuristics and biases: Biases in judgments reveal some heuristics of thinking under uncertainty. *science*, 185(4157):1124–1131.

Vaswani, A., Shazeer, N., Parmar, N. et al (2017). Attention is all you need. In Guyon, I., Luxburg, U.V., Bengio, S. et al, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Versteegh, K. (2014). *Arabic language*. Edinburgh University Press.

Waldis, A., Perlitz, Y., Choshen, L. et al (2024). Holmes: Benchmark the linguistic competence of language models. *arXiv preprint arXiv:2404.18923*.

Wang, C., Cho, K. and Gu, J. (2020). Neural machine translation with byte-level subwords. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 9154–9160.

Wang, Y., Ivison, H., Dasigi, P. et al (2023a). How far can camels go? exploring the state of instruction tuning on open resources. *Advances in Neural Information Processing Systems*, 36:74764–74786.

Wang, Z., Dong, Y., Delalleau, O. et al (2024). Helpsteer2: Open-source dataset for training top-performing reward models. *arXiv preprint arXiv:2406.08673*.

Wang, Z., Dong, Y., Zeng, J. et al (2023b). Helpsteer: Multi-attribute helpfulness dataset for steerlm. *arXiv preprint arXiv:2311.09528*.

Watson, J.C. (2002). *The phonology and morphology of Arabic*. Oxford University Press, USA.

Weber, M., Fu, D., Anthony, Q. et al (2024). RedPajama: an open dataset for training large language models. *arXiv preprint arXiv:2411.12372*.

Woo, S., Debnath, S., Hu, R. et al (2023). Convnext v2: Co-designing and scaling convnets with masked autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16133–16142.

Xia, M., Malladi, S., Gururangan, S. et al (2024). Less: Selecting influential data for targeted instruction tuning. *arXiv preprint arXiv:2402.04333*.

Xu, S., Pang, L., Shen, H. et al (2023). Search-in-the-Chain: Towards Accurate, Credible and Traceable Content Generation for Complex Knowledge-Intensive Tasks. *arXiv preprint arXiv:2304.14732*.

Zellers, R., Holtzman, A., Bisk, Y. et al (2019). HellaSwag: Can a machine really finish your sentence? In Korhonen, A., Traum, D. and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Zeng, W., Ren, X., Su, T. et al (2021). PanGu-α: Large-scale autoregressive pretrained Chinese language models with auto-parallel computation. *arXiv preprint arXiv:2104.12369*.

Zhang, C., D'Haro, L.F., Chen, Y. et al (2024). A comprehensive analysis of the effectiveness of large language models as automatic dialogue evaluators. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19515–19524.

Zhao, H., Andriushchenko, M., Croce, F. et al (2024). Long is more for alignment: A simple but tough-to-beat baseline for instruction fine-tuning. *arXiv preprint arXiv:2402.04833*.

Zheng, L., Chiang, W.L., Sheng, Y. et al (2023). Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Zhou, J., Lu, T., Mishra, S. et al (2023). Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

Zhu, B., Frick, E., Wu, T. et al (2024). Starling-7B: Improving helpfulness and harmlessness with RLAIF. In *First Conference on Language Modeling*.