Tokens with Meaning: A Hybrid Tokenization Approach for NLP

M. Ali Bayram[1], **Ali Arda Fincan**[2], **Ahmet Semih Gümüş**[2], **Sercan Karakaş**[3],
Banu Diri[1], Savaş Yıldırım[4], Demircan Çelik[2]
[1]Yıldız Technical University, [2]Yeditepe University, [3]University of Chicago,
[4]Istanbul Bilgi University
malibayram20@gmail.com

Abstract

Tokenization shapes how language models perceive morphology and meaning, yet widely used frequency-driven subword tokenizers (e.g., BPE/WordPiece) can fragment morphologically rich and agglutinative languages in ways that obscure morpheme boundaries. We introduce a linguistically informed hybrid tokenizer that combines (i) dictionary-driven morphological segmentation (roots and affixes), (ii) phonological normalization that maps allomorphic variants to shared identifiers, and (iii) a controlled subword fallback for out-of-vocabulary coverage. Concretely, our released Turkish vocabulary contains 20,000 root identifiers, 72 affix identifiers that cover 177 allomorphic surface forms, and 12,696 subword units; special tokens represent whitespace and orthographic case without inflating the vocabulary.

We evaluate tokenization quality on TR-MMLU using two linguistic alignment metrics: Turkish Token Percentage (TR %), the proportion of produced tokens that correspond to Turkish lexical/morphemic units under our lexical resources, and Pure Token Percentage (Pure %), the proportion of tokens aligning with unambiguous root/affix boundaries. The proposed tokenizer reaches 90.29% TR % and 85.80% Pure % on TR-MMLU, substantially exceeding several general-purpose tokenizers. We further validate practical utility with downstream sentence embedding benchmarks for Turkish: on STSb-TR, models using the proposed tokenizer consistently outperform a strong Turkish subword baseline under the same training budget, while remaining competitive across MTEB-TR task categories. Although our implementation is Turkish-specific, the framework is intended to transfer to other morphologically rich languages given comparable lexical resources and decoding rules.

Keywords: Tokenization, Morphologically Rich Languages, Morphological Segmentation, Byte Pair Encoding, Turkish NLP, Linguistic Integrity, Low-Resource Languages

1 Introduction

Tokenization—the mapping from text to discrete symbols—is a foundational design choice in modern NLP systems, influencing vocabulary size, sequence length, and how reliably models can represent morphology and meaning [1]. Subword tokenizers such as Byte Pair Encoding (BPE) [2] and WordPiece [3] have become the default for transformer-based models [4, 5] because they mitigate out-of-vocabulary issues and scale to large corpora. However, frequency-driven segmentation can conflict with linguistic structure, particularly in morphologically rich and agglutinative languages where words encode many grammatical features through productive affixation.

Turkish is a canonical case: a single lemma can generate many surface forms through suffix sequences, while morphophonological alternations (e.g., vowel harmony, consonant alternations) create additional

surface variability. In such settings, subword splits can fragment roots, merge partial morphemes, and treat allomorphic variants as unrelated symbols, reducing interpretability and potentially weakening generalization [6, 7].

This paper investigates a simple guiding hypothesis: *token boundaries that better align with morpheme boundaries provide cleaner input structure for learning*. Following [8], we quantify linguistic alignment using (i) Turkish Token Percentage (TR %), the proportion of tokens that correspond to Turkish lexical/morphemic units, and (ii) Pure Token Percentage (Pure %), the proportion of tokens that align with unambiguous root/affix boundaries. These metrics are not a substitute for downstream evaluation, but they provide an interpretable diagnostic for segmentation quality.

We propose a hybrid tokenizer that is *morphology-first*: it segments via curated root and affix dictionaries with phonological normalization (mapping surface allomorphs to shared identifiers), and applies a controlled subword fallback to handle out-of-vocabulary material. While our implementation is Turkish-specific, the framework is designed to transfer to other morphologically rich languages given comparable lexical resources.

Contributions. This work makes four contributions:

- A linguistically informed hybrid tokenization pipeline combining dictionary-based morphological segmentation, phonological normalization, and a subword fallback.
- A transparent Turkish implementation with open dictionaries (`kokler.json`, `ekler.json`) and decoding rules (`turkish_decoder.py`).
- An evaluation on TR-MMLU using TR % and Pure % to quantify token–morpheme alignment.
- A downstream sentence embedding evaluation on Turkish STS and MTEB-TR, showing that under the same training budget the proposed tokenizer yields stronger STS performance and remains competitive across MTEB-TR categories.

Scope and limitations. Our experiments are Turkish-focused; we therefore frame cross-linguistic generality as a *framework claim* (the pipeline structure) rather than an empirical claim across typologically diverse languages. We discuss transfer requirements and limitations in Section 5.

Paper organization. Section 2 reviews subword and morphology-aware tokenization approaches and Turkish-specific baselines. Section 3 details our vocabulary design, encoding/decoding procedures, and edge-case handling. Section 4 reports tokenization alignment on TR-MMLU and downstream sentence embedding results on STSb-TR and MTEB-TR.

2 Related Work

Subword tokenization. Subword methods such as BPE [2], WordPiece [3], and Unigram-based approaches [9] are widely used because they balance vocabulary size and coverage in neural models [4, 1]. For morphologically rich languages, however, segmentation granularity and vocabulary allocation become more consequential, often leading to higher token "fertility" and longer effective sequences [7].

Morphology-aware tokenization for Turkish and beyond. Prior work on Turkish shows that tokenizer choice can substantially affect model quality and efficiency. Toraman et al. [6] study tokenization impacts for Turkish language models, while Bayram et al. [8] propose linguistic alignment metrics (including TR % and Pure %) as diagnostics for Turkish tokenization quality. More broadly, morphology-aware segmentation has been explored for tasks such as machine translation [10, 11] and abstractive summarization [12].

Hybrid approaches combining morphology and subwords. Hybrid designs that retain linguistic structure while keeping statistical coverage are increasingly common. Jabbar [13] proposes MorphPiece, which applies morpheme segmentation prior to subword encoding. Asgari et al. [14] propose MorphBPE, a morphology-aware extension of BPE that constrains merges and introduces morphology-based evaluation metrics across multiple languages. Rahimov [15] introduces miLLi, a

dictionary+BPE hybrid for Azerbaijani with a phonological restoration layer that maps allomorphic variants back to canonical roots. Our work aligns with this line but adopts a morphology-first pipeline with explicit allomorph unification and reversible decoding rules.

Turkish-specific baselines and benchmarks. Recent Turkish-focused foundation models and benchmarks emphasize the importance of language-specific evaluation and documentation. Türker et al. [16] introduce TabiBERT and a unified Turkish benchmark, motivating fairer Turkish baselines beyond English-centric tokenizers. Complementary work on Turkish sentence embeddings highlights practical tokenizer adaptation and evaluation on STS/retrieval benchmarks; Bayram [17] describes token remapping and embedding-alignment techniques for adapting pretrained embedding models to Turkish.

Downstream semantic similarity evaluation. Correlation-based evaluation (Pearson/Spearman) is standard for semantic textual similarity, and for agglutinative languages semantic similarity metrics can capture quality beyond surface lexical overlap. Beken Fikri et al. [18] translate STS resources into Turkish and motivate semantic similarity as an evaluation signal for Turkish text generation and summarization.

Empirical variability. Across languages and training setups, the effect of morphology-aware tokenization on downstream model performance can vary. We therefore treat linguistic alignment metrics as interpretable diagnostics for segmentation quality rather than as proof of downstream gains, and we complement them with downstream evaluation (Section 4.3).

3 Methodology

Our tokenizer is designed for morphologically rich, predominantly concatenative systems (e.g., Turkish) where many grammatical functions are expressed through productive suffixation and morphophonological alternations. The key design choice is *morphology-first tokenization*: we prioritize roots and affixes as atomic units when possible, while using subwords only as a fallback for coverage.

3.1 Resources: Roots, Affixes, and Subwords

The tokenizer vocabulary is the union of three components:

- **Root lexicon** (`kokler.json`): a curated list of Turkish roots and high-frequency lexicalized forms (including selected compounds). Each root may have multiple surface variants mapped to a shared identifier to control vocabulary growth.
- **Affix lexicon** (`ekler.json`): a curated list of productive suffixes and function markers. Allomorphic variants (e.g., vowel harmony and consonant alternations) are grouped to share identifiers.
- **Subword fallback** (`bpe_tokenler.json`): a list of frequent subword units used only when dictionary segmentation fails. In our release, these units are derived from a Turkish subword training run and exported as a lookup table.[1]

We additionally include special tokens for whitespace and case handling, notably `<uppercase>` and an explicit space token.

Identifier ranges and allomorph sets. Our released vocabulary uses disjoint identifier ranges for clarity: (i) roots: ids 0–19999, (ii) affixes: ids 20000–20071, and (iii) subwords: ids 20072–32767. Importantly, the lexicons map multiple surface strings to shared identifiers to encode allomorph equivalence classes. In the current release, `kokler.json` contains 22,231 surface strings mapped to 20,000 root identifiers, and `ekler.json` contains 177 surface strings mapped to 72 affix identifiers (a direct measure of allomorph unification). The subword fallback consists of 12,696 distinct subword units.

[1]The training corpora include public Turkish corpora hosted on HuggingFace (e.g., [19, 20]).

Table 1: Vocabulary composition in the released Turkish tokenizer. "Strings" counts surface forms in JSON files; "IDs" counts unique identifiers (i.e., equivalence classes).

Component	Strings	IDs	ID range
Roots (`kokler.json`)	22,231	20,000	0–19,999
Affixes (`ekler.json`)	177	72	20,000–20,071
Subwords (`bpe_tokenler.json`)	12,696	12,696	20,072–32,767

3.2 Dictionary Construction and Curation

The root lexicon is designed to cover high-frequency lexical material while keeping identifiers stable across surface variants. In practice, this involves (i) collecting high-frequency Turkish word types from large corpora, (ii) filtering and normalizing candidates (e.g., handling Turkish dotted/dotless *i*), and (iii) optionally adding lexicalized multiword compounds as atomic entries when they behave as stable meaning units in the target corpus. This curation improves robustness and interpretability but introduces a key trade-off: the lexicon is language-specific and requires maintenance and domain adaptation.

The affix lexicon encodes productive suffixes and function markers. Each affix identifier can correspond to multiple surface allomorphs. For example, vowel harmony yields alternations over {a, e} and {ı, i, u, ü}, and consonant hardness can yield alternations such as *d/t* or *k/ğ* (e.g., *kitap* vs. *kitabı*; *çocuk* vs. *çocuğu*). Instead of treating each surface form as a distinct token, we group them into equivalence classes and select the appropriate surface form at decode time based on phonological context.

3.3 Normalization and Allomorph Unification

Turkish suffixes and some roots exhibit systematic surface variation driven by vowel harmony and consonant alternations. Our vocabulary therefore supports *many-to-one* mappings: multiple surface strings can map to the same token identifier. During decoding, the appropriate surface allomorph is selected based on phonological context (front/back and rounded/unrounded vowel harmony; consonant hardness).

We explicitly model common Turkish vowel harmony patterns for suffix selection, including alternations over {a, e} and {ı, i, u, ü}. We also treat *y* as a *vowel or semivowel* in relevant boundary contexts (e.g., buffer consonant behavior), following standard Turkish morphophonology.

3.4 Encoding Algorithm

The encoder operates by longest-prefix matching against the three vocabularies in a fixed order (roots → affixes → subwords). It also inserts explicit markers for whitespace and capitalization.

Algorithm 1: Morphology-first hybrid encoding

Input: text string x
Output: token id sequence y

1. Split x into whitespace-separated segments; represent spaces explicitly.
2. For each segment w: (a) Split camelCase/PascalCase boundaries (e.g., `HTTPServer` → `H T T P Server`).
(b) For each subsegment starting with an uppercase letter, emit `<uppercase>` and lowercase the subsegment.
(c) Scan left-to-right; at each position, emit the longest matching prefix from (roots, then affixes, then subwords).
(d) If no match exists, emit `<unknown>` and advance by one character.
3. Return the concatenated ids.

This procedure is deterministic and designed to be reproducible from the released lookup tables. We use a longest-match heuristic over the *surface form*. This choice improves robustness in practice, but it can diverge from lexical (underlying) analyses in ambiguous cases; we discuss this limitation in Section 5.

3.5 Decoding and Surface Form Reconstruction

Decoding maps token identifiers back to text using reverse vocabularies and morphophonological rules. If an id corresponds to an allomorph set (multiple surface strings), the decoder selects the surface form based on the preceding lexical material (vowel harmony; consonant hardness). For the `<uppercase>` marker, the decoder capitalizes the subsequent token.

This design enables near-lossless reconstruction for typical Turkish text where unknown tokens are rare and capitalization is mostly word-initial. However, full losslessness is not guaranteed: unknown tokens are rendered as placeholders, and sequences of all-caps letters (e.g., acronyms) may be imperfectly reconstructed under the current marker design. We treat this as an explicit edge case for future refinement.

Decoder rules (implementation summary). Our public decoder (`turkish_decoder.py`) implements rule-based selection for affix allomorphs based on the most recent vowel in the preceding token (front/back; rounded/unrounded) and the hardness of the final consonant (e.g., *d/t* alternations). In addition, a small set of root identifiers are associated with multiple surface root forms (e.g., variants triggered by vowel-initial suffixes), and the decoder selects among them based on whether the next token begins with a vowel. These rules are intentionally lightweight: they do not attempt full morphological disambiguation, and they operate on local context rather than syntactic structure.

Complexity. The encoder performs longest-prefix matching over bounded prefix lengths, making runtime approximately linear in input length with a constant factor determined by maximum token string lengths in each lexicon. In practice, this enables efficient tokenization while preserving the transparency of a rule-based pipeline.

3.6 Worked Example with Leipzig-Style Glossing

We illustrate morphology-first segmentation on a multi-morpheme Turkish word:

> **Example 1**: anla-yabil-dik-ler-imiz-den
> *understand*-ABIL-NMLZ-PL-1PL.POSS-ABL
> "'from what we were able to understand"'

This example highlights two goals: (i) preserve the root as a stable unit, and (ii) isolate productive suffixes so that grammatical meaning is represented compositionally rather than through arbitrary subword fragments. We follow standard Leipzig-style conventions for interlinear glossing [21].

4 Results and Analysis

4.1 Metrics: TR% and Pure%

We report two linguistic alignment metrics introduced in [8]. Let a tokenizer produce a token sequence $t_1, \ldots, t_N$ for a corpus. TR % measures how many produced tokens correspond to Turkish lexical or morphemic units under a Turkish lexical resource:

$$\mathrm{TR}\% = \frac{\sum_{i=1}^{N} \mathbb{1}[\mathrm{TurkishUnit}(t_i)]}{N} \times 100.$$

Pure % measures how many produced tokens align with unambiguous root/affix boundaries:

$$\mathrm{Pure}\% = \frac{\sum_{i=1}^{N} \mathbb{1}[\mathrm{PureUnit}(t_i)]}{N} \times 100.$$

In our implementation, `turkish_tokenizer` uses explicit root and affix vocabularies, so the set of "TurkishUnit" and "PureUnit" tokens is defined directly by these lexicons. For baseline tokenizers, these metrics are computed via the same evaluation pipeline from [8]. While TR % and Pure % are interpretable diagnostics for segmentation quality, we complement them with downstream task evaluation in Section 4.3.

4.2 Tokenization quality on TR-MMLU

The performance of the proposed morphological tokenizer was evaluated using the TR-MMLU benchmark dataset, which comprises over 1.6 million characters and approximately 200,000 words curated specifically for Turkish [22]. This dataset is designed to reflect the linguistic complexity of Turkish, including its rich morphology, agglutinative structures, and diverse syntactic constructions. As such, it provides a rigorous basis for assessing tokenization quality in morphologically complex languages.

The evaluation compared five different tokenizers: `google/gemma-2-9b`, `meta-llama/Llama-3.2-3B`, `Qwen/Qwen2.5-7B-Instruct`, `CohereForAI/aya-expanse-8b`, and the proposed `turkish_tokenizer`. Each tokenizer was assessed using a consistent set of linguistic and computational metrics introduced in [8]. These metrics include total token count, vocabulary size, number of unique tokens, Turkish Token Percentage (TR %), and Pure Token Percentage (Pure %). TR % quantifies the proportion of tokens that correspond to valid Turkish words or morphemes, while Pure % measures the proportion of tokens that fully align with unambiguous root or affix boundaries, thus reflecting morphological integrity.

Table 2: Performance of the proposed `turkish_tokenizer` on the TR-MMLU dataset.

Metric	**Value**
Vocabulary Size	32,768
Total Token Count	707,727
Processing Time (s)	0.6714
Unique Token Count	11,144
Turkish Token Count	10,062
Turkish Token Percentage (TR %)	90.29%
Pure Token Count	9,562
Pure Token Percentage (Pure %)	85.80%

Table 3: Linguistic alignment comparison on TR-MMLU. We report TR % and Pure % for tokenizers where the corresponding metric was computed in our evaluation pipeline; missing entries indicate the metric was not computed in this run.

Tokenizer	TR (%)	Pure (%)
turkish_tokenizer (ours)	**90.29**	**85.80**
google/gemma-2-9b	40.96	28.49
meta-llama/Llama-3.2-3B	45.77	31.45
Qwen/Qwen2.5-7B-Instruct	40.39	–
CohereForAI/aya-expanse-8b	53.48	–

The proposed `turkish_tokenizer` demonstrated the highest linguistic alignment across all evaluated metrics. It achieved a TR % of 90.29% and a Pure % of 85.80%, substantially outperforming all competing tokenizers. In comparison, `google/gemma-2-9b` reached a TR % of only 40.96% and a Pure % of 28.49%, indicating that the majority of its tokens do not represent full morphemes. Similarly, `meta-llama/Llama-3.2-3B` produced a TR % of 45.77% and a Pure % of 31.45%, while `Qwen2.5` and `aya-expanse` achieved TR % values of 40.39% and 53.48%, respectively.

Despite employing significantly smaller vocabulary sizes, the proposed tokenizer demonstrated better linguistic segmentation. With a vocabulary of 32,768 tokens and 11,144 unique tokens used during evaluation, it balanced generalization and expressiveness more effectively than models such as `gemma-2-9b` and `aya-expanse`, which rely on vocabularies of over 255,000 tokens. These large-vocabulary tokenizers, rooted in frequency-based subword segmentation, tend to fragment morphologically rich expressions and introduce ambiguity in downstream tasks. In contrast, the morphological awareness of the `turkish_tokenizer` enables semantically coherent token formation and more consistent syntactic parsing.

Although the total token count generated by the proposed tokenizer (707,727) exceeds those of the other models—for instance, `aya-expanse` produced 434,526 tokens—this increase is offset by gains in interpretability and linguistic fidelity. High TR % and Pure % scores suggest reduced reliance on

spurious subword splits and improved preservation of morphosyntactic structure. This is particularly beneficial for tasks such as syntactic parsing, translation, summarization, and question answering, where semantic consistency across tokens is essential.

These tokenization results are consistent with the diagnostic perspective of Bayram et al. [8]: linguistic alignment metrics such as TR % and Pure % capture whether tokens correspond to coherent Turkish morphemes and lexical units. However, these diagnostics do not substitute for downstream evaluation. We therefore complement them with downstream sentence embedding evaluation results later in this section.

To illustrate the linguistic fidelity of different tokenization strategies, we present a qualitative comparison using an example sentence sampled from the TR-MMLU evaluation corpus [22]:

"Atasözleri geçmişten günümüze kadar ulaşan anlamı bakımından mecazlı bir mana kazanan kalıplaşmış sözlerdir."
("Proverbs are fixed expressions passed down from the past to the present that acquire a metaphorical meaning in terms of their significance.")

For example, the word *atasözleri* can be glossed as:

> **Example 2**: atasöz-ler-i
> *proverb*-PL-3SG.POSS
> "'his/her proverbs" (surface form used in the sentence)'

This sentence contains a wide range of morphological features, including compound words, multiple derivational and inflectional suffixes, and root forms that undergo phonological alternations. These properties make it an ideal test case for evaluating the morphological sensitivity of different tokenizers.

Proposed Hybrid Tokenizer:
The hybrid morphological tokenizer segments the sentence into linguistically meaningful units with high fidelity. It produces:
`["<uppercase>", "atasöz", "ler", "i", "<space>", "geçmiş", "ten", "<space>", "gün", "üm", "üz", "e", "<space>", "kadar", "<space>", "ulaş", "an", "<space>", "anlam", "ı", "<space>", "bakım", "ın", "dan", "<space>", "mecaz", "lı", "<space>", "bir", "<space>", "mana", "<space>", "kazan", "an", "<space>", "kalıp", "laş", "mış", "<space>", "sözle", "r", "dir", "."]`
It correctly separates suffixes (`"ler"`, `"i"`, `"ın"`, `"dan"`, `"lı"`, `"an"`, `"mış"`, `"dir"`), extracts root forms such as `"atasöz"`, `"gün"`, `"mana"`, and employs special tokens like `"<uppercase>"` and `"<space>"` to preserve orthographic structure.

Gemma-3:
The tokenizer `google/gemma-3` segments the sentence as:
`["<bos>", "At", "as", "öz", "leri", " geçmiş", "ten", " gün", "ümü", "ze", " kadar", " ulaş", "an", " anlam", "ı", " bakım", "ından", " mec", "az", "lı", " bir", " mana", " kaz", "anan", " kal", "ı", "pla", "ş", "mış", " söz", "lerdir", "."]`
Although it captures some suffixes like `"ten"` and `"ından"`, it fragments common roots (`"At"`, `"as"`, `"öz"` instead of `"atasöz"`) and fails to isolate inner morphemes in forms such as `"lerdir"` and `"kazanan"`, limiting morphological interpretability.

LLaMA-3.2:
The tokenizer `meta-llama/Llama-3.2-3B` yields:
`["<|begin_of_text|>", "At", "as", "öz", "leri", " geçmiş", "ten", " gün", "ümü", "ze", " kadar", " ", "ula", "ş", "an", " anlam", "ı", " bakımından", " me", "ca", "z", "lı", " bir", " mana", " kaz", "anan", " kal", "ı", "pla", "ş", "mış", " söz", "lerdir", "."]`
This tokenizer combines morphologically valid segments like `"bakımından"` and `"kazanan"` with fragmented roots like `"At"`, `"as"`, `"öz"`, creating inconsistency in morpheme alignment.

Qwen2.5:
The tokenizer `Qwen/Qwen2.5` outputs:
`["At", "as", "öz", "leri", " geçmiş", "ten", " gün", "üm", "ü", "ze", " kadar", " ulaş", "an", " anlamı", " bakım", "ından", " me", "ca", "z", "lı", " bir", " mana", " kaz", "anan", " kal", "ı", "pla", "ş", "mış", " söz", "ler", "dir", "."]`
While suffixes such as `"ten"` and `"ından"` are recognized, the tokenizer introduces redundant splits like `"üm", "ü", "ze"`, reducing the linguistic coherence of the token stream.

Aya-Expanse:
The tokenizer `CohereForAI/aya-expanse` returns:
`["<BOS_TOKEN>", "At", "as", "öz", "leri", " geçmiş", "ten", " günümüze", " kadar", " ulaşan", " anlamı", " bakımından", " mec", "az", "lı", " bir", " mana", " kazanan", " kalı", "pl", "aş", "mış", " söz", "lerdir", "."]`
It retains some complete word forms such as `"günümüze"` and `"ulaşan"`, but still fragments compounds like `"kalıplaşmış"` and splits the root `"atasöz"`, reducing morphological traceability.

Phi-4:
The tokenizer `microsoft/phi-4` produces:
`["At", "as", "ö", "z", "leri", " geç", "mi", "ş", "ten", " gün", "üm", "ü", "ze", " kadar", " ", "ula", "ş", "an", " an", "lam", "ı", " bak", "ım", "ından", " me", "ca", "z", "lı", " bir", " mana", " kaz", "anan", " kal", "ı", "pla", "ş", "m", "ış", " sö", "z", "ler", "dir", "."]`
This tokenizer over-fragments even basic stems like `"geçmiş"` into `"geç", "mi", "ş"` and `"anlam"` into `"an", "lam"`, increasing token count and reducing interpretability.

YTU Turkish GPT-2:
The tokenizer `ytu-ce-cosmos/turkish-gpt2-large-750m-instruct-v0.1`, trained on Turkish corpora, yields:
`["At", "as", "öz", "leri", " geçmişten", " günümüze", " kadar", " ulaşan", " anlamı", " bakımından", " mec", "az", "lı", " bir", " mana", " kazanan", " kalıp", "laşmış", " söz", "lerdir", "."]`
Although it still segments `"atasözleri"` incorrectly, it performs well with forms like `"geçmişten"`, `"günümüze"`, and `"bakımından"`, showing the advantage of Turkish-specific pretraining.

GPT-4o:
The tokenizer `gpt-4o-o200k_base` generates:
`["At", "as", "öz", "leri", " geçmiş", "ten", " gün", "ümü", "ze", " kadar", " ulaş", "an", " anlam", "ı", " bakım", "ından", " mec", "az", "lı", " bir", " mana", " kaz", "anan", " kal", "ı", "pla", "ş", "mış", " söz", "ler", "dir", "."]`
Its segmentation strategy is similar to LLaMA and Qwen—partially aware of Turkish morphemes but limited by frequent over-segmentation of compound and derived forms.

Overall, the qualitative comparisons show a consistent pattern: general-purpose tokenizers often fragment frequent Turkish roots and conflate or split suffix material inconsistently, whereas the proposed tokenizer aims to preserve morpheme boundaries and treat common allomorphic variants as shared identifiers. We next evaluate whether these segmentation differences translate into measurable downstream quality on Turkish semantic similarity and retrieval benchmarks.

4.3 Downstream Sentence Embedding Evaluation (STS and MTEB-TR)

Benchmarks. We evaluate downstream quality using (i) Turkish Semantic Textual Similarity (STSb-TR) with Pearson/Spearman correlation, a standard evaluation for sentence embeddings [18], and (ii) MTEB-TR-style benchmarks spanning retrieval, classification, clustering, and STS tasks [23].

Table 4: Turkish STS benchmark (STSb-TR) results. Pearson/Spearman are computed on the test split (1,379 pairs).

Model	Pearson (%)	Spearman (%)
MFT-Magibu	**74.41**	**73.08**
MFT-Gemma	71.02	70.00
Tabi-Magibu	66.29	64.97
Tabi-Gemma	61.54	60.56
MFT-RandomInit	47.09	45.96
Tabi-RandomInit	40.53	38.60

Table 5: MTEB-TR summary (26 tasks). We report the overall average and category averages from the repository report.

Model	Overall Avg (%)	Retrieval Avg (%)	STS Avg (%)
MFT-Gemma	62.09	**65.90**	72.94
MFT-Magibu	61.83	64.39	**74.73**
Tabi-Gemma	62.36	65.73	71.47
Tabi-Magibu	**62.59**	65.46	72.41
MFT-RandomInit	38.99	28.94	49.36
Tabi-RandomInit	33.33	18.46	33.24

Training and comparison protocol. To compare tokenizers under a controlled budget, we train sentence embedding models using a teacher-guided embedding alignment objective: student models are trained to match fixed teacher embedding vectors for the same text. This setting isolates tokenizer effects in a downstream-relevant representation learning task while avoiding the cost of online teacher inference. Our released training protocol uses a maximum sequence length of 2,048 tokens, cosine embedding loss, mean pooling, and bf16 training with gradient checkpointing. We report the complete experimental configuration (hardware, hyperparameters, dataset schema, filtering rules) in the repository artifact `TRAINING_DETAILS.md`.

Tokenizer comparison and baselines. We compare models trained under the same training budget with two tokenization settings: the proposed morphology-first tokenizer (MFT) vs. a strong Turkish subword baseline (Tabi). To isolate tokenizer effects, we report results within the same training recipe and include random-initialized baselines as sanity checks.

Key findings. On STSb-TR, MFT yields a substantial improvement over the Tabi baseline: the best MFT model reaches 74.41% Pearson vs. 66.29% for the best Tabi model (+8.12 points), and the gap persists under the random-initialized sanity check (47.09% vs. 40.53%, +6.56 points) (Table 4, Figure 1). On MTEB-TR, results are closer: the best Tabi model has a slightly higher overall average (62.59% vs. 62.09%), while MFT has the best STS category average (74.73% vs. 72.41%) and a markedly stronger random-initialized baseline (38.99% vs. 33.33%) (Table 5). Taken together, these results support two points: (i) morphology-first tokenization is consistently beneficial for semantic similarity, and (ii) when learning must start from scratch, Tabi-style subword tokenization does not close the gap to MFT within the same training budget.

Efficiency Analysis. Evaluation time on the STSb-TR test set serves as a proxy for tokenizer efficiency. The pure-Python implementation of the MFT tokenizer incurs a slight overhead compared to the Rust-based Tabi baseline: average inference time for MFT models was $\approx$ 22.7s versus $\approx$ 21.1s for Tabi models (a $\sim$7% increase). This minor latency cost is balanced by the significant gains in semantic alignment, particularly for applications where representation quality outweighs sub-millisecond throughput.

Robustness across checkpoints. We additionally track performance across multiple saved model checkpoints and report version-history plots in the repository (`VERSION_BENCHMARK_RESULTS.md`). The best-performing checkpoint in our runs is `mft-downstream-task-embeddingmagibu` with

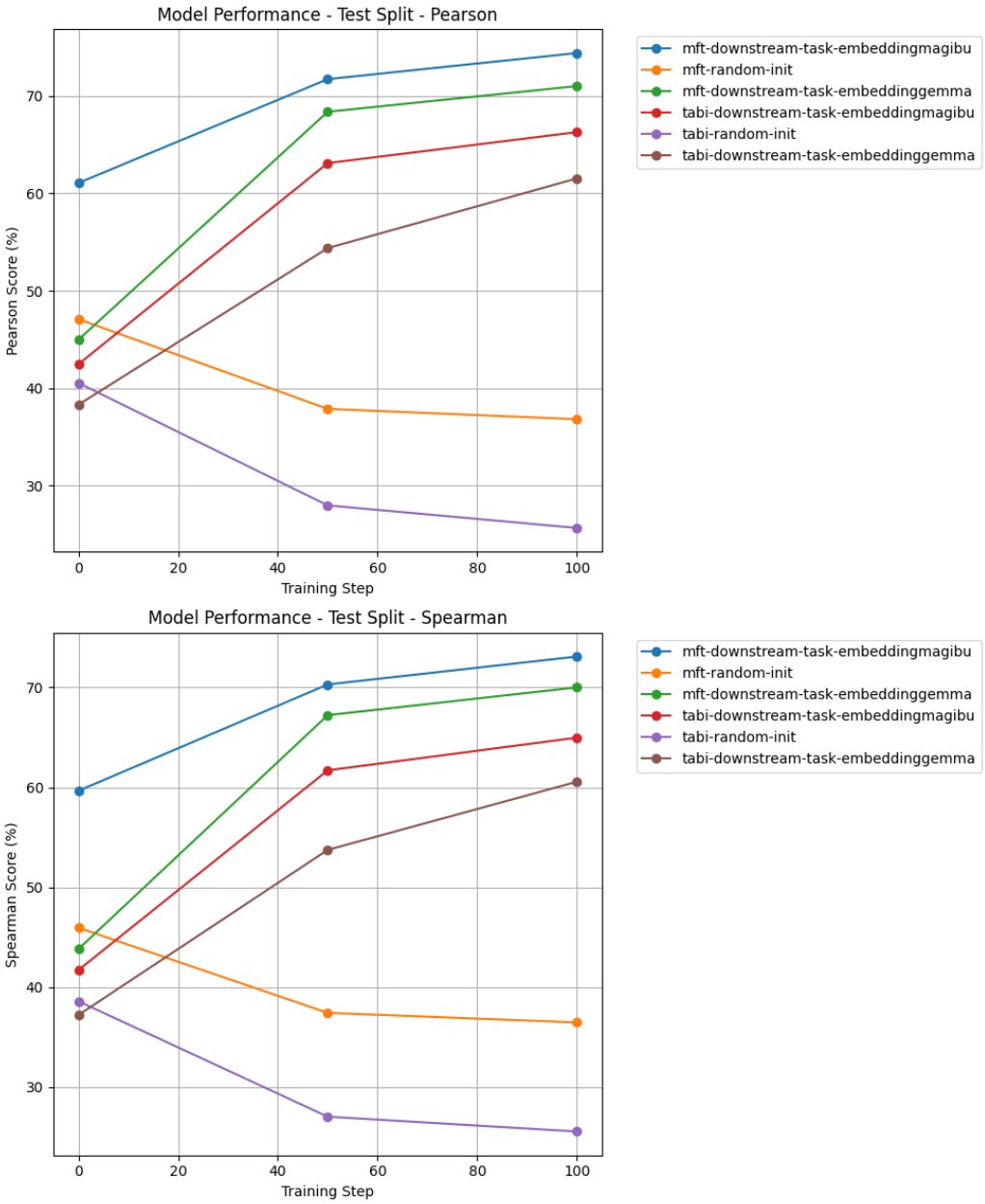

Figure 1: STS benchmark test split results across training steps (from repository artifact `STS_BENCHMARK_RESULTS.md`).

76.10% Pearson (Table 4 summarizes the main comparison point; see the repository report for full history).

5 Future Work

While we add downstream sentence embedding evaluation for Turkish (Section 4.3), several directions remain open.

Beyond Turkish. Our framework is language-agnostic in structure but requires language-specific resources (root and affix inventories, plus decoding rules). A critical next step is evaluating the same

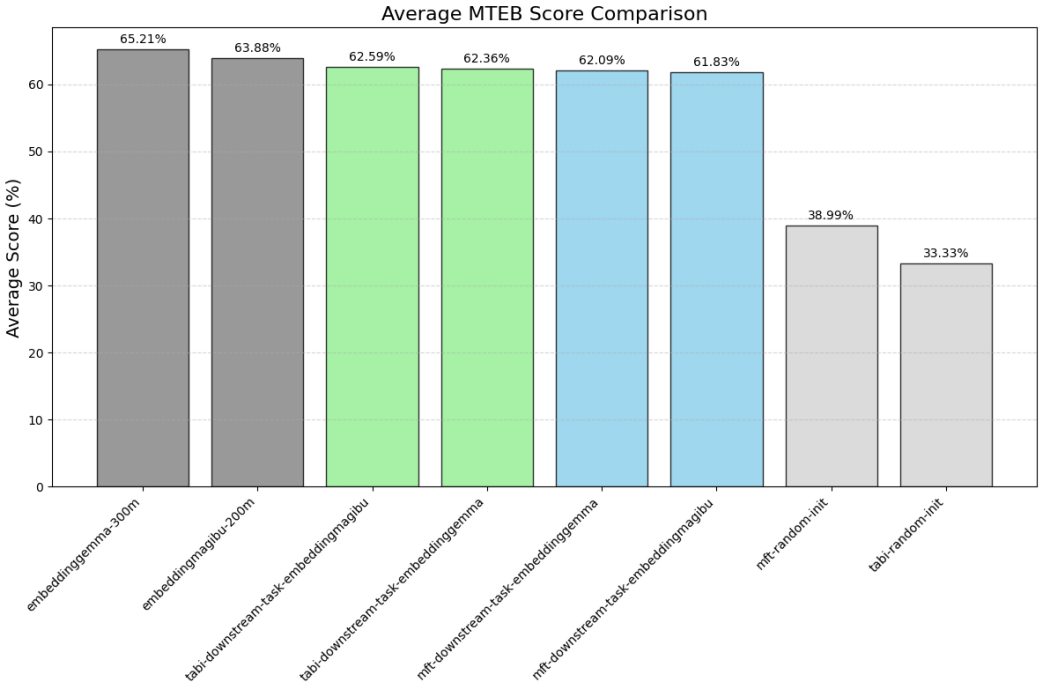

Figure 2: Overall average MTEB-TR scores across evaluated models (from repository artifact `MTEB_BENCHMARK_RESULTS.md`).

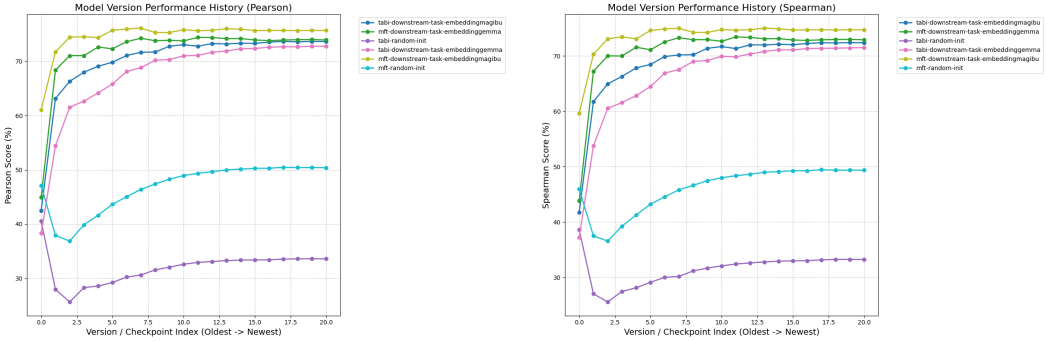

Figure 3: Version-history robustness on STSb-TR (Pearson/Spearman), from `VERSION_BENCHMARK_RESULTS.md`.

pipeline on additional morphologically rich languages (e.g., other Turkic languages, Finno-Ugric languages) to separate framework-level benefits from Turkish-specific choices.

Broader linguistic coverage. Turkish morphophonology includes alternations and exceptions beyond the subset covered by our current rules (e.g., additional consonant alternations such as k/ğ, gemination in loanwords, and harmony exceptions). Extending both the affix inventory and the decoder's restoration rules—and reporting ablations—would clarify which linguistic components drive improvements.

Edge cases and losslessness. Capitalization beyond simple word-initial case (e.g., acronyms and mixed-case identifiers such as `HTTPServer`) remains imperfect under the current `<uppercase>` marker design. Improving case preservation without vocabulary inflation is an important practical refinement.

Efficiency characterization. Our TR-MMLU table reports processing time for the proposed tokenizer, but we do not provide comparable speed measurements for all baselines. Future work should include standardized throughput/latency evaluation across tokenizers and analyze the trade-off between linguistic purity, token count, and downstream latency.

Additional benchmarks. To strengthen the linguistic motivation, future experiments should evaluate models (not only tokenizers) on targeted linguistic benchmarks where morphology matters, alongside broad downstream suites (e.g., Turkish-specific or multilingual BLiMP-style evaluations).

6 Conclusion

We presented a linguistically informed, morphology-first hybrid tokenizer designed for Turkish and similar agglutinative languages. The tokenizer combines curated root and affix lexicons with phonological normalization (mapping surface allomorphs to shared identifiers) and a controlled subword fallback for coverage. This design aims to produce token sequences that more closely align with morpheme boundaries while remaining practical for large-scale NLP pipelines.

On TR-MMLU, the proposed tokenizer achieves 90.29% Turkish Token Percentage (TR %) and 85.80% Pure Token Percentage (Pure %), indicating substantially stronger morpheme-level alignment than several general-purpose tokenizers. We additionally report downstream sentence embedding evaluation on Turkish STS and MTEB-TR. Under the same training budget, MFT-based models consistently outperform the Turkish subword baseline on STS and remain competitive across MTEB-TR task categories, with random-initialized baselines reinforcing that the observed gaps are not explained by training noise alone.

We emphasize that empirical claims in this paper are Turkish-focused. The framework structure is transferable, but cross-linguistic performance depends on the availability and quality of language-specific lexical resources and decoding rules. We outline concrete next steps—broader language coverage, improved morphophonological handling, better capitalization edge cases, and standardized efficiency measurements—in Section 5.

References

[1] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, July 2019. arXiv:1907.11692 [cs].

[2] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units, June 2016. arXiv:1508.07909 [cs].

[3] Mike Schuster and Kaisuke Nakajima. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152, March 2012.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, May 2019. arXiv:1810.04805 [cs].

[5] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[6] Cagri Toraman, Eyup Halit Yilmaz, Furkan Şahinuç, and Oguzhan Ozcelik. Impact of tokenization on language models: An analysis for turkish. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 22(4):1–21, April 2023. arXiv:2204.08832 [cs].

[7] Yiğit Bekir Kaya and A. Cüneyd Tantuğ. Effect of tokenization granularity for turkish large language models. *Intelligent Systems with Applications*, 21:200335, March 2024.

[8] M. Ali Bayram, Ali Arda Fincan, Ahmet Semih Gümüş, Sercan Karakaş, Banu Diri, and Savaş Yıldırım. Tokenization standards for linguistic integrity: Turkish as a benchmark, February 2025. arXiv:2502.07057 [cs].

[9] Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing, August 2018. arXiv:1808.06226 [cs].

[10] Yirong Pan, Xiao Li, Yating Yang, and Rui Dong. Morphological word segmentation on agglutinative languages for neural machine translation, January 2020. arXiv:2001.01589 [cs].

[11] Matthias Huck, Simon Riess, and Alexander Fraser. Target-side word segmentation strategies for neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 56–67, September 2017.

[12] Batuhan Baykara and Tunga Güngör. Abstractive text summarization and new large-scale datasets for agglutinative languages turkish and hungarian. *Language Resources and Evaluation*, 56(3):973–1007, September 2022.

[13] Haris Jabbar. Morphpiece: A linguistic tokenizer for large language models, February 2024. arXiv:2307.07262 [cs].

[14] Ehsaneddin Asgari, Yassine El Kheir, and Mohammad Ali Sadraei Javaheri. MorphBPE: A Morpho-Aware Tokenizer Bridging Linguistic Complexity for Efficient LLM Training Across Morphologies, February 2025. arXiv:2502.00894 [cs].

[15] Elshad Rahimov. miLLi: Model Integrating Local Linguistic Insights for Morphologically Robust Tokenization, December 2025.

[16] Melikşah Türker, A. Ebrar Kızıloğlu, Onur Güngör, and Susan Üsküdarlı. TabiBERT: A Large-Scale ModernBERT Foundation Model and A Unified Benchmark for Turkish, January 2026. arXiv:2512.23065 [cs].

[17] M. Ali Bayram. Adapting pretrained embedding models to turkish via token remapping and distillation. Manuscript, 2026. Included in repository under papers/.

[18] Figen Beken Fikri, Kemal Oflazer, and Berrin Yanikoglu. Semantic Similarity Based Evaluation for Abstractive News Summarization. In *Proceedings of the First Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 24–33, Online, August 2021. Association for Computational Linguistics.

[19] umarigan/turkish_corpus_small (dataset).

[20] kadirnar/combined-turkish-datasets-v4 (dataset).

[21] Leipzig glossing rules. Max Planck Institute for Evolutionary Anthropology.

[22] M. Ali Bayram, Ali Arda Fincan, Ahmet Semih Gümüş, Banu Diri, Savaş Yıldırım, and Öner Aytaş. Setting standards in turkish nlp: Tr-mmlu for large language model evaluation, January 2025. arXiv:2501.00593 [cs].

[23] Massive text embedding benchmark (mteb).