Tokens with Meaning: A Hybrid Tokenization Approach for NLP

Anonymous ACL submission

Abstract

Tokenization plays a pivotal role in natural language processing (NLP), shaping how textual data is segmented, interpreted, and processed by language models. Despite the success of subword-based tokenization techniques such as Byte Pair Encoding (BPE) and WordPiece, these methods often fall short in morphologically rich and agglutinative languages due to their reliance on statistical frequency rather than linguistic structure. This paper introduces a linguistically informed hybrid tokenization framework that integrates rule-based morphological analysis with statistical subword segmentation to address these limitations. The proposed approach leverages phonological normalization, root-affix dictionaries, and a novel tokenization algorithm that balances morpheme preservation with vocabulary efficiency. The framework also incorporates special tokens for whitespace and orthographic case, including an `<uppercase>` token to prevent vocabulary inflation from capitalization. Byte Pair Encoding is integrated to support out-of-vocabulary coverage without compromising morphological coherence. Evaluation on the TR-MMLU benchmark—a large-scale, Turkish-specific NLP benchmark—demonstrates that the proposed tokenizer achieves the highest Turkish Token Percentage (90.29%) and Pure Token Percentage (85.8%) among all tested models. Comparative analysis against widely used tokenizers from models such as LLaMA, Gemma, and OpenAI's GPT reveals that the proposed method yields more linguistically meaningful and semantically coherent tokens. A qualitative case study further illustrates improved morpheme segmentation and interpretability in complex Turkish sentences. This work contributes to ongoing efforts to improve tokenizer design through linguistic alignment, offering a practical and extensible solution for enhancing both interpretability and performance in multilingual NLP systems.

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

1 Introduction

Tokenization is a foundational step in Natural Language Processing (NLP), directly impacting vocabulary construction, model efficiency, and downstream task performance (Liu et al., 2019). While subword-based methods like Byte Pair Encoding (BPE) (Sennrich et al., 2016) and WordPiece (Schuster and Nakajima, 2012) effectively handle out-of-vocabulary (OOV) words in high-resource languages, they often disregard the linguistic structure of morphologically rich languages. In agglutinative languages like Turkish, Finnish, and Hungarian, words are formed by appending multiple affixes to a root, resulting in complex surface forms. Frequency-based subword models frequently violate morphemic boundaries in these languages, reducing semantic coherence and interpretability (Toraman et al., 2023).

Turkish poses specific challenges due to its agglutinative nature and phonological processes such as vowel harmony and consonant alternation. Words are formed by appending multiple affixes to a root, producing an expansive set of surface forms. For instance, the single word *Avrupalılaştıramadıklarımızdanmışsınızcasına* ("as if you were one of those whom we could not make resemble a European") conveys a meaning that requires an entire sentence in English. Standard tokenizers often treat these variants as distinct or fragment them inconsistently, leading to vocabulary redundancy and poor alignment with linguistic units (Bayram et al., 2025b). Recent benchmarks, such as TR-MMLU (Bayram et al., 2025a), indicate that "token purity"—the alignment of tokens with morphemes—correlates strongly with model performance. Token purity fundamentally determines the clarity of the statistical patterns LLMs learn. Unlike impure subwords that introduce ambiguity, tokens aligned with complete morphemes provide consistent semantic and syntactic signals, facilitating better generalization across complex word forms (Hofmann et al., 2021). Empirical evidence supports this: morphologically informed models have been shown to outperform standard BPE baselines, achieving superior efficiency even with fewer training iterations (Jabbar, 2024). This principle mirrors object-centric learning in computer vision, where decomposing inputs into meaningful entities—rather than undifferentiated features—enhances recognition.(capsule networks (Sabour et al., 2017) and object-centric architectures like Slot Attention (Locatello et al., 2020)) Consequently, token purity is not merely a linguistic preference but a structural necessity for semantic awareness, motivating its use as a primary evaluation metric in this work.

To address these limitations, we introduce a linguistically informed hybrid tokenization framework. Our approach integrates rule-based morphological segmentation with BPE to ensure both linguistic fidelity and broad coverage. First, phonological processes unify surface variants into shared identifiers. Second, a dedicated <uppercase> token decouples capitalization from lexical identity, preventing vocabulary inflation. Third, explicit formatting tokens are employed to preserve the structural integrity of the input for layout-sensitive tasks. Finally, a hybrid architecture integrates dictionary-based morphological segmentation with BPE, balancing the need for linguistic purity with robust coverage for out-of-vocabulary terms.

2 Related Work

Tokenization significantly impacts model performance, especially in morphologically rich languages (Toraman et al., 2023). While subword methods like BPE (Sennrich et al., 2016) and WordPiece (Schuster and Nakajima, 2012) are standard, they often fail to capture the agglutinative structure of languages like Turkish, Finnish, and Hungarian (Baykara and Güngör, 2022).

Early Turkish NLP relied on rule-based morphological analyzers like Zemberek (Akın and Akın, 2007), which offered precise segmentation but lacked the scalability required for modern LLMs. Recent research has sought to bridge this gap. Toraman et al. (2023) showed that morphological tokenization could recover 97% of BERT's performance with a fraction of the model size. Similarly, Pan et al. (2020) and Huck et al. (2017) demonstrated that morphology-aware segmentation improves Neural Machine Translation (NMT) by reducing data sparsity. Most recently, (Asgari et al., 2025) introduced MorphBPE, a hybrid method that constrains BPE merges to respect morpheme boundaries. Their experiments demonstrated that this linguistic alignment yields tangible computational benefits, resulting in significantly lower training loss and faster convergence rates compared to standard subword models.

Hybrid approaches combining rule-based and statistical methods have shown promise. Kayalı and Omurca (2024) used a hybrid tokenizer for Turkish NER and summarization, finding benefits in preserving linguistic structure. Jabbar (2024) introduced MorphPiece for English, achieving superior performance with smaller vocabularies. However, challenges remain in balancing vocabulary size, sequence length, and computational efficiency (Henderson et al., 2022; Kaya and Tantuğ, 2024). Our work extends these efforts by introducing a fully hybrid pipeline that integrates phonological normalization directly into the tokenization process.

Despite the theoretical appeal of morphological segmentation, some studies argue that strict linguistic boundaries may not always be optimal for neural models. Kudo (2018) introduced "subword regularization," suggesting that exposing models to multiple segmentations of the same word (including non-canonical ones) can improve robustness and generalization. Similarly, recent large-scale multilingual models often favor larger, data-driven vocabularies that maximize compression rates over linguistic purity. This "byte-premium" hypothesis suggests that minimizing the number of tokens per word is the primary driver of cross-linguistic performance gaps (Martins et al., 2024).

Furthermore, the rise of massive multilingual LLMs presents a dilemma for language-specific tokenization. While a dedicated Turkish tokenizer offers superior alignment for Turkish text, it may not be easily integrated into a model trained on 100+ languages without significantly increasing the combined vocabulary size or complicating the embedding space. Our work acknowledges this tension but argues that for high-stakes or specialized applications in morphologically rich languages, the benefits of semantic coherence and interpretability outweigh the costs of language-specific engineering.

3 Methodology

Traditional subword tokenization methods like BPE (Sennrich et al., 2016) and WordPiece (Schuster and Nakajima, 2012) often fail to capture the rich internal structure of agglutinative languages. For example, the Turkish word *anlayabildiklerimizden* is composed of multiple morphemes (*anla-yabil-dik-ler-imiz-den*). Standard tokenizers fragment such words arbitrarily, obscuring grammatical function. To address this, we propose a hybrid tokenization framework that prioritizes morphological segmentation while retaining BPE as a fallback for robustness.

3.1 Dictionary Construction

The foundation of our hybrid tokenizer is a curated set of morphological dictionaries derived from open-source linguistic resources, primarily the Zemberek NLP framework (Akın and Akın, 2007) and the Turkish Language Association (TDK) data.

1. **Root Dictionary :** We extracted approximately 22,000 high-frequency Turkish roots (nouns and verbs) from a large-scale corpus of Turkish web text. These roots were filtered to exclude rare or archaic terms that would unnecessarily inflate the vocabulary. Special control tokens (<uppercase>, <unknown>, <pad>, <eos>) were added to support model training.

2. **Suffix Dictionary :** This dictionary contains 230 distinct derivational and inflectional suffixes. Crucially, we applied phonological abstraction: suffixes that are phonetically distinct but functionally identical (allomorphs) are mapped to a single canonical ID. For example, the plural suffix forms *-lar* and *-ler* share the same token ID, as do the four variants of the accusative case (*-ı, -i, -u, -ü*). Similarly, the future tense suffixes *-acak* and *-ecek* are unified under a single canonical ID, reflecting their shared grammatical function despite the consonant change.

3. **BPE Vocabulary :** To ensure full coverage for foreign words, proper names, and rare scientific terms, we trained a Byte Pair Encoding (BPE) model on the same corpus with a vocabulary limit of 10,000 subwords. This serves as a fallback mechanism for any segment not found in the root or suffix dictionaries.

3.2 Encoding Process

Encoding process consists of three main stages when converting text into sequences of IDs. First, the input text is separated into base words by splitting it based on space characters. In this stage, a space prefix is added to the beginning of each word (Example: "elma" → " elma"), due to the fact that all tokens in the tokenizer's roots dictionary include a leading space. Second, these separated words are processed according to their capitalization status. If a word starts with a capital letter, it is represented by a special <uppercase> token followed by the lowercase version of the word (Example: "Ankara" → <uppercase> + " ankara"). If capitalization is not at the beginning (like "iPhone"), the word is fragmented and the capitalized parts are parsed with special tokens. The final stage is converting the pieces into numerical IDs. All obtained pieces (roots, suffixes, and BPE tokens) are searched within the root, suffix, and BPE dictionaries. The search algorithm proceeds by iteratively removing letters backward from the end of the piece (using the longest prefix match logic). For instance, in the word "kitaplar," attempts are made sequentially with "kitaplar," "kitapla," etc.; once the root "kitap" is found in the root list, its ID is added, and then the remaining part, "lar," is searched in the suffix list and its token ID is added. Upon completion of these operations, the final numerical ID list of the text is generated.

```
Algorithm: Hybrid Encoding
Input: text string T
Output: list of token IDs

1. Split T into words by whitespace.
2. For each word W:
    a. Identify uppercase positions.
    b. Split W into segments (handle camelCase).
    c. For each segment S:
        i.   Check ROOTS for longest prefix match.
             If match: add ID, continue.
        ii.  Check SUFFIXES for longest prefix match
             If match: add ID, continue.
        iii. Check BPE for longest prefix match.
             If match: add ID, continue.
        iv.  Else: add <unknown> ID.
    d. Insert <uppercase> tokens based on (a).
3. Return list of IDs.
```

Figure 1: Pseudocode for the hybrid encoding process.

3.3 Decoding and Phonological Resolution

The **Decoding** phase reconstructs text from token IDs using a "Single ID, Multiple Views" principle. Since multiple surface forms (allomorphs) map to a single ID, the decoder must dynamically resolve the correct form based on context. This process is critical for generating natural-sounding Turkish text.

3.3.1 Root Resolution (Lookahead)

In this system, a special root selection mechanism is employed in the Decoder stage, following tokenization, to accurately model morphological changes in natural language. Due to the morphological structure of Turkish, surface forms of the same root that have undergone sound events (phonological changes) are represented by the same ID in the system, even though they look different.For instance, kitap (book) and kitab- (the form resulting from consonant softening), ağız (mouth) and ağz- (the form resulting from vowel deletion), and küçük (small) and küçüğ- (the form resulting from consonant deletion/softening) all correspond to the same root ID . As seen in these examples, even though the forms affected by sound events are different on the surface, the model unifies these variants under a single root identity. While the Tokenizer stores these potential variations as a list associated with the ID, the Decoder's main task is to select the correct surface form from this list that is most appropriate for the context. To make this selection, the Decoder calls the specially defined function for every root ID. This function determines the most correct morphological form by looking at the information of the suffix that will follow the root. The main factors the function considers are: whether the next token is a suffix, if it is a suffix, whether it starts with a vowel (like "yor," "acak," or "ı"), whether it is one of certain specific special suffixes, and finally, whether the root's ID falls within a range subject to special rules.

A concrete example of this root selection mechanism is as follows: Let's assume the tokens corresponding to Root ID 100 are → ["sıcak", "sıcağ", "sıca"]. If the ID sequence arrives as [100, 2034] (where ID 2034 represents the suffix "ı"), the Decoder immediately activates the `_select_correct_root` function. This function detects that ID 2034 is a vowel-initial suffix. Based on this contextual information, the decision is made that consonant softening must occur, and the softened form of the root, "sıcağ," is selected. When the suffix is subsequently added ("sıcağ" + "ı"), the correct morphological form, "sıcağı" (the accusative case of hot/warm), is obtained. This ensures the root is selected and prepared in the correct form based on the type of suffix that will follow it. Additionally, the Decoder manages the Uppercase Marker, which is a special grammatical token. When the token ID is 0, the Decoder understands that this marker must convert the first letter of the immediately following root into a capital letter.

3.3.2 Suffix Resolution (Lookbehind)

In this system, if a token ID is 20,000 or greater, it represents a suffix, and just like roots, any given suffix ID may correspond to multiple surface forms. This is due to the fact that Turkish suffixes follow phonological harmony rules such as major/minor vowel harmony and consonant alternation (e.g., the plural suffix 20000 → ["lar", "ler"], or the future tense suffix 20030 → ["acak", "ecek", "acağ", "eceğ", "yacak", "yecek", "yacağ", "yeceğ"]). The primary responsibility of the Decoder is to call a specialized function that selects the correct surface form from these alternative suffix lists based on context. This function evaluates several factors when choosing the appropriate suffix: the last vowel of the preceding word to ensure vowel harmony, whether the final letter of the preceding word is a voiceless consonant for consonant alternation, whether the next token begins with a vowel to determine if a buffer consonant (such as y) is required, the position of the suffix within the word, and whether the suffix belongs to a special category (e.g., “la/le,” “da/de–ta/te,” “cık/cik,” “mak/mek,” “acak/ecek,” etc.). After retrieving the list of surface variants associated with the suffix ID, the function routes the selection process to the appropriate specialized sub-function depending on the suffix category, ensuring that the correct surface form is chosen.

- **Example 1 — Vowel Harmony:** If the preceding word is “ev” (its last vowel “e” is a front vowel), the plural suffix (20000) selects the form “ler” to satisfy front-vowel harmony. Result: [“ev”, “20000”] → “evler”.

- **Example 2 — Consonant Assimilation (Fortition):** If the preceding word is “kitap” (ending with the voiceless consonant “p”) and the suffix is -da/-de, the Decoder applies consonant assimilation and hardens the suffix to “ta/te”. Since the last vowel of the word is “a” (a back vowel), the suffix is further matched to the back-vowel form. Result: “kitapta”.

- **Example 3 — Future Tense (Vowel Harmony):** If the preceding word is “bak” (its last vowel “a” is a back vowel) and the suffix is -acak/-ecek, the Decoder selects the form “acak” to satisfy back-vowel harmony. Result: “bakacak”.

4 Results and Analysis

We evaluated the proposed tokenizer on the TR-MMLU benchmark (Bayram et al., 2025a), comparing it against five state-of-the-art tokenizers: `google/gemma-2-9b`, `meta-llama/Llama-3.2-3B`, `Qwen/Qwen2.5-7B-Instruct`, `CohereForAI/aya-expanse-8b`, and `microsoft/phi-4`. Metrics include Vocabulary Size, Total Tokens, Unique Tokens, Turkish Token Percentage (TR %), and Pure Token Percentage (Pure %).

As shown in Table 1, our tokenizer achieves the highest linguistic alignment (90.29% TR %, 85.80% Pure %) while being 8× smaller than Gemma’s 256k tokenizer. This size efficiency is crucial, as it significantly reduces the embedding layer parameters, thereby minimizing the model’s memory footprint.This linguistic precision leads to an anticipated trade-off: an increased sequence length, with our tokenizer generating 63% more tokens `aya-expanse` (707k vs. 434k). While a higher token count means longer sequences and a higher computational cost for attention mechanisms, it provides a crucial benefit for agglutinative languages. It furnishes the model with a more explicit and regular representation of the language, allowing it to learn the compositional rules of the grammar rather than being forced to memorize millions of distinct surface forms (e.g., *geldim, geldin, geldi...*) as individual vocabulary items.

Table 1: Performance comparison on the TR-MMLU dataset.

Tokenizer	Vocab	Tokens	Unique	TR %	Pure %
`turkish_tokenizer` (Ours)	32k	707k	11.1k	**90.29**	**85.80**
`google/gemma-2-9b`	256k	497k	6.3k	40.96	28.49
`meta-llama/Llama-3.1`	128k	488k	6.8k	45.77	31.45
`Qwen/Qwen2.5-7B-Instruct`	152k	561k	5.7k	40.39	30.15
`CohereForAI/aya-expanse-8b`	256k	434k	8.5k	53.48	32.96

These results confirm that frequency-based tokenizers, even with large vocabularies, struggle with the agglutinative structure of Turkish. Our hybrid approach, by contrast, yields tokens that are linguistically meaningful and consistent, a success that underscores its value for managing this characteristic complexity. By ensuring our tokens are linguistically meaningful and respect morpheme boundaries , we not only achieve higher alignment metrics but also provide the model with a superior, more regular representation of the language. This allows the model to efficiently learn the compositional nature of Turkish grammar, demonstrating that prioritizing morphological integrity can be more effective for complex linguistic structures than simply relying on large, frequency-driven vocabularies.

5 Discussion

The current practice in multilingual models (e.g., LLaMA-3, Gemma-2) is to use large, shared vocabularies (128k–256k), which, while efficient for high-resource languages like English, often treats low-resource, agglutinative languages as "second-class citizens" by forcing them into fragmented subword sequences.

Our findings propose a viable alternative: a modular tokenization approach or a "mixture-of-tokenizers" architecture, where language-specific morphological adapters are swapped in during training . This method allows models to process each language in its most natural structural form. Crucially, the principles of our hybrid framework are highly portable; they are directly applicable to other agglutinative languages such as Finnish, Hungarian, and Estonian, which share features like rich suffixation and compounding. The underlying "Longest Prefix Match" algorithm with phonological abstraction is language-agnostic, requiring only the replacement of language-specific dictionaries and the definition of phonological rules (like vowel harmony groups). This inherent portability is a significant advantage over purely statistical methods, which demand massive corpus retraining to implicitly discover these linguistic rules.

6 Conclusion

In this study, we introduced a linguistically informed hybrid tokenization framework designed to address the challenges of morphologically rich languages. By integrating rule-based morphological analysis with BPE, our approach preserves morpheme boundaries and minimizes vocabulary redundancy. Empirical evaluations on the TR-MMLU dataset demonstrate that our tokenizer achieves significantly higher linguistic alignment (90.29% TR %, 85.80% Pure %) compared to state-of-the-art multilingual models like LLaMA, Gemma, and Qwen. These results validate that incorporating linguistic structure into tokenization yields more semantically coherent representations.

6.1 Limitations

While our results are promising, this study has several limitations. First, our evaluation relies primarily on intrinsic metrics (TR % and Pure %). Due to computational constraints, we did not pretrain a language model from scratch to empirically verify the impact of our tokenizer on downstream task performance (e.g., perplexity, classification accuracy). Establishing a causal link between token purity and model performance remains a critical next step. Second, our current implementation is in Python, which may not match the inference speed of highly optimized Rust-based tokenizers used in production LLMs. We have not yet conducted rigorous benchmarking of processing time or memory usage. Third, the approach relies on manually curated dictionaries, which may require maintenance to cover evolving language use and neologisms.

6.2 Future Work

Future research will focus on three key areas: (1) **Downstream Evaluation:** Training a small-scale language model (e.g., 100M parameters) using our tokenizer to measure improvements in perplexity and task-specific accuracy compared to standard BPE; (2) **Optimization:** Re-implementing the tokenizer in Rust to ensure it meets the latency requirements of real-time applications; and (3) **Generalization:** Extending the framework to other agglutinative languages such as Finnish and Hungarian to test the cross-linguistic validity of our hybrid approach.

References

Mehmet Dündar Akın and Ahmet Afşin Akın. 2007. Türk Dilleri İçin Açık Kaynaklı Doğal Dil İşleme Kütüphanesi: ZEMBEREK. *elektrik mühendisliği*, 431:38–44.

Ehsaneddin Asgari, Yassine El Kheir, and Mohammad Ali Sadraei Javaheri. 2025. Morphbpe: A morpho-aware tokenizer bridging linguistic complexity for efficient llm training across morphologies. *arXiv preprint*. ArXiv:2502.00894 [cs.CL].

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

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

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

Peter Henderson, Jieru Hu, Joshua Romoff, Emma Brunskill, Dan Jurafsky, and Joelle Pineau. 2022. Towards the systematic reporting of the energy and carbon footprints of machine learning. *arXiv preprint*. ArXiv:2002.05651 [cs].

Valentin Hofmann, Janet Pierrehumbert, and Hinrich Schütze. 2021. Superbizarre is not superb: Derivational morphology improves bert's interpretation of complex words. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, pages 3594–3608.

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

Haris Jabbar. 2024. Morphpiece: A linguistic tokenizer for large language models. *arXiv preprint*. ArXiv:2307.07262 [cs].

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

Nihal Zuhal Kayalı and Sevinç İlhan Omurca. 2024. Hybrid tokenization strategy for turkish abstractive text summarization. In *2024 8th International Artificial Intelligence and Data Processing Symposium (IDAP)*, pages 1–6.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. *arXiv preprint*. ArXiv:1804.10959 [cs].

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

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. 2020. Object-centric learning with slot attention. In *Advances in Neural Information Processing Systems*, volume 33, pages 11525–11538.

Pedro Henrique Martins, Patrick Fernandes, João Alves, Nuno M. Guerreiro, Ricardo Rei, Duarte M. Alves, José Pombal, Amin Farajian, Manuel Faysse, Mateusz Klimaszewski, Pierre Colombo, Barry Haddow, José G. C. de Souza, Alexandra Birch, and André F. T. Martins. 2024. Eurollm: Multilingual language models for europe. *arXiv preprint*. ArXiv:2409.16235 [cs].

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

Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. 2017. Dynamic routing between capsules. *arXiv preprint*. ArXiv:1710.09829 [cs].

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

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

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