MUHAT Tokenization

Andy

May 4, 2025

1 Introduction

Research on the formal language expressivity of transformers has not considered the contribution of tokenization. Generally, studies have considered an embedding function that maps every single symbol to a unique word embedding Strobl et al. (2024). In practice, however, tokenization methods such as BPE (Sennrich et al., 2016) are used to group symbols together before applying a word embedding. Here we will investigate how the use of tokenization may affect the expressivity of a model, and use MUHATs (Yang et al., 2024) as a test bed.

2 Preliminaries

I take vocabulary and notation from Guo (1997).

Definition 2.1 (Dictionary)**.** Let Σ is an alphabet of characters. A *Dictionary* is a set of strings $\Delta \subseteq \Sigma^*$ containing at least ϵ. Each element of Δ is called a *token*. A string over Δ is one formed by concatenation of tokens from Δ.

For clarity, we will usually notate the tokenizations of strings in Σ^* with a box as tokens in Δ. That is, if $w \in \Sigma^*$ is a token of Δ we will write $\boxed{w} \in \Delta$. From each dictionary, we get a natural homomorphism from $\Delta^* \to \Sigma^*$ that we call a detokenization.

Definition 2.2 (Detokenziation)**.** The induced detokenization $D_\Delta \colon \Delta^* \to \Sigma^*$ is given by mapping a token $t \in \Delta$ to the corresponding string $t \in \Sigma^*$, so that

$$\begin{aligned} D_\Delta(\epsilon) &= \epsilon \\ D_\Delta(t_0 t) &= t_0 D_\Delta(t) \qquad\qquad \text{for } t_0 \in \Delta, t \in \Delta^* \end{aligned}$$

It is easy to check that D_Δ is a string homomorphism. Then we can define a tokenization, which can be thought of the inverse mapping of D_Δ

Definition 2.3 (Generalized Tokenization)**.** A *generalized tokenization* $T_\Delta \colon \Sigma^* \to \mathcal{P}(\Delta^*)$ is defined as the image under inverse homomorphism of the detokenization D_Δ. That is:

$$T_\Delta(w) = \{t \in \Delta^* \mid D_\Delta(t) = w\}$$

We say that every $t \in T_\Delta(w)$ is a *tokenization* of w. Unless specified otherwise, we will use *tokenization* to refer to a generalized tokenization.

Note that T_Δ may in fact map a given string w to many different tokenizations. We introduce some vocabulary for talking about how many tokenizations a given string has over Δ.

Definition 2.4. We introduce the following three terms. Given an alphabet Σ, let $w \in \Sigma^*$ and let Δ be a dictionary over Σ. Then we say

- w has *ambiguous* over Δ if $|T_\Delta(w)| > 1$
- w has *unambiguous* over Δ if $|T_\Delta(w)| = 1$
- w is *ill-formed* (not *well-formed*) over Δ if $|T_\Delta(w)| = 0$

Then, we say that the dictionary Δ is *complete* if every word $w \in \Sigma^*$ is well-formed over Δ. In this case, we will also say that T_Δ is complete. Note that a dictionary is complete iff every $\sigma \in \Sigma$ appears in Δ.

Let's look at some toy examples.

Example 2.5. Let $\Sigma = \{a, b\}$. Let $\Delta = \left\{\boxed{a}, \boxed{b}, \boxed{ab}, \boxed{ba}\right\}$. First, Δ is complete. The string *aaa* is unambiguous over Δ. The string *abab* is ambiguous over Δ. All strings in Σ^* are well-formed over Δ.

Example 2.6. Let $\Sigma = \{a, b\}$. Let $\Delta = \left\{\boxed{a}, \boxed{aa}, \boxed{ab}, \boxed{ba}\right\}$. First, Δ is not complete. The string *bbb* is not well-formed over Δ. The string *aaa* is ambiguous over Δ. The string *baba* is unambiguous over Δ.

It is important to note the following

Lemma 2.7. *If Δ is complete and contains any tokens t such that $|t| > 1$, then T_Δ is ambiguous.*

Proof. Let $\boxed{w} \in \Delta$ where $w = w_1 w_2 \cdots w_n$ for $n > 1$. Then w can be tokenized as either $\boxed{w}$ or as $\boxed{w_1}\boxed{w_2}\cdots\boxed{w_n}$. □

3 Tokenization for Star-Free Languages

The first observation we can make is that for every complete tokenization T_Δ, if L is recognized by some counter-free automata, then it is still possible to recognize L by a counter-free automata which only sees every word after the tokenization T_Δ is applied. More formally

Lemma 3.1. *Let Σ be an alphabet and let Δ be a complete dictionary of Σ. If $L \subseteq \Sigma^*$ is a star-free language, then the image of L under tokenization $T_\Delta(L)$ is also star-free.*

In particular, this is a simple corollary of the following closure property of star-free languages.

Lemma 3.2. *The star-free languages are closed under inverse homomorphism*.*

Proof. Let $L \subseteq \Sigma^*$ be a star-free language. Let M be a finite aperiodic monoid recognizing L. That is, there is a homomorphism $\eta\colon \Sigma^* \to M$ and $P \subseteq M$ such that $\eta^{-1}(P) = L$. Let $\varphi\colon \Gamma^* \to \Sigma^*$ be a string homomorphism. So $\eta \circ \varphi\colon \Gamma^* \to M$ is also a homomorphism. Furthermore we have that

*I couldn't easily locate a good citation for this other than this proof sketch by J.E. Pin on cs stackexchange `https://cs.stackexchange.com/questions/14785/are-regular-languages-closed-under-inverse-homomorphism`

$$\varphi^{-1}(L) = \varphi^{-1}(\eta^{-1}(P)) = (\eta \circ \varphi)^{-1}(P)$$

Thus, $\varphi^{-1}(L)$ is also recognized by M. Since a language is star-free iff it is recognized by a finite aperiodic monoid (Pin, 2020), we conclude $\varphi^{-1}(L)$ is also star-free. Thus, the star-free languages are closed under inverse homomorphism. □

Since tokenization is an inverse string homomorphism, we get Lemma 3.1 as an immediate corollary. For clarity, we can consider an example.

Example 3.3. Let $\Sigma = \{a, b\}$. Let $\Delta = \left\{\boxed{a}, \boxed{b}, \boxed{aa}, \boxed{ab}, \boxed{ba}, \boxed{bb}\right\}$. Δ is complete. The language $L = (ab)^*$ is star-free. First, we can consider some example strings and their tokenizations

$$\begin{aligned}
\epsilon &\mapsto \left\{\boxed{\epsilon}\right\} \\
ab &\mapsto \left\{\boxed{ab}, \boxed{a}\boxed{b}\right\} \\
abab &\mapsto \left\{\boxed{ab}\boxed{ab}, \boxed{ab}\boxed{a}\boxed{b}, \boxed{a}\boxed{b}\boxed{ab}, \boxed{a}\boxed{ba}\boxed{b}, \boxed{a}\boxed{b}\boxed{a}\boxed{b}\right\}
\end{aligned}$$

We see that the image of L under tokenization T_Δ can be written as the following star-free regular expression

$$T_\Delta(L) = \left(\left(\boxed{b}(\emptyset)^{\mathsf{C}} \cup (\emptyset)^{\mathsf{C}}\boxed{a} \cup (\emptyset)^{\mathsf{C}}\boxed{aa}(\emptyset)^{\mathsf{C}} \cup (\emptyset)^{\mathsf{C}}\boxed{bb}(\emptyset)^{\mathsf{C}}\right)\right)^{\mathsf{C}}$$

Thus, $T_\Delta(L)$ is also star-free. By the lemma this actually would have worked for any complete dictionary Δ, but this is the most straightforward example.

However, the other direction is not true. That is, the image of star-free languages under *detokenization* may not be star-free.

Lemma 3.4. *The star-free languages are not closed under homomorphism.*

This is best illustrated (and thus proven) with an example

Example 3.5. Let $\Sigma = \{a\}$ and let $\Delta = \{\boxed{a}, \boxed{aa}\}$ be a dictionary. Δ is complete. Consider $L = \boxed{aa}^* \subseteq \Delta^*$, which is star-free. The image of L under detokenization $D_\Delta(L) = (aa)^*$ is in fact not star-free. For instance $\mathbb{Z}_2$ as (the additive monoid) recognizes $(aa)^*$.

Note, however, this does not imply that that using the tokenization T_Δ can map $(aa)^*$ into a star-free language. This is because the image $T_\Delta((aa)^*) = \left\{t \in \Delta^* \mid \#\boxed{a} \text{ in } t \text{ is even}\right\}$, and this language is PARITY over the dictionary Δ. In general, tokenization in the sense defined above doesn't help.

Lemma 3.6. *The non-star-free languages are closed under complete tokenizations.*

Proof. First, complete tokenizations result from detokenizations which are surjective string homomorphisms. Since the regular languages are closed under homomorphism, if the image of a language L is to be star-free, L must at least be regular. This means we can follow Lemma 3.1 but let M be any periodic monoid. □

For now, we can state all of the above more formally as a result about MUHATs. First, we define how a MUHAT accepts words under generalized tokenizations.

Definition 3.7. Let Δ be a complete dictionary over alphabet Σ. Let $\mathcal{T}$ be a MUHAT over Σ considered as a string acceptor. We say $\mathcal{T}$ accepts $w \in \Sigma^*$ under tokenization T_Δ iff $\mathcal{T}$ accepts all $t \in T_\Delta(w)$. We say $\mathcal{T}$ recognizes a language $L \subseteq \Sigma^*$ when $\mathcal{T}$ accepts $w \in \Sigma^*$ iff $w \in L$.

Then, we can state and prove our main result.

Theorem 3.8. *MUHATs recognize exactly the star-free languages under generalized tokenizations.*

Proof. Let Δ be a complete dictionary over Σ. In one direction, let L be a star-free language. By Lemma 3.1, the image of L under any complete tokenization is still star-free. That is, there is some star-free language $K \subseteq \Delta^*$ such that for $w \in L, v \notin L$ we have that $T_\Delta(w) \subseteq K$ and $T_\Delta(v) \subseteq \Delta^* \setminus K$. Let $\mathcal{T}$ be a MUHAT which recognizes K (Yang et al., 2024), and then $\mathcal{T}$ will recognize L under generalized tokenization T_Δ. In the other direction, the proof is similar after using Lemma 3.6. □

4 Refinements of Tokenization

It is notable that Theorem 3.8 relied on a fairly general notion of acceptance, which required that in order accept w you had to accept all possible tokenizations of w. In practice, you typically only consider some of the feasible tokenizations of a word. In fact, it is possible that by reducing the size of $T_\Delta(w)$ we may be able to recognize non-star-free languages under tokenization using MUHATs.

Example 4.1. Let $\Sigma = \{a\}$ and let $\Delta = \{\boxed{a}, \boxed{aa}\}$. Define the refined tokenization $R_\Delta(w) = \left\{t \in T_\Delta(w) \mid t \in \left(\boxed{aa}\right) \cup \left(\boxed{aa}\right)^* \boxed{a}\right\}$. R_Δ is still complete, and then the image of $(aa)^*$ under R_Δ is actually a star-free language $R_\Delta((aa)^*) = \left(\boxed{aa}\right)^*$. Thus, there is a MUHAT which accepts $(aa)^*$ under tokenization R_Δ.

Given Lemma 3.6, we know that if image under the inverse of a surjective homomorphism of a language L is star-free, then L must be regular. Thus

Theorem 4.2. *MUHATs recognize only regular languages under refined tokenizations.*

It remains to see how tokenization methods used in practice, such as BPE, may be formalized as refined tokenizations in the sense above. The results above suggest it is possible to increase the expressivity of the model by just changing the tokenization method. This is interesting because it is a way to gain expressivity without changing any of the model internals - all tokenization does is modify how the inputs are pre-processed. Furthermore, a theoretical understanding of tokenization may also help us better understand the nuances of how models treat strings at different levels of granularity. Perhaps this could point towards better practices on how to tokenize natural language for different tasks.

References

Jin Guo. 1997. Critical tokenization and its properties. *Computational Linguistics* 23, 4 (1997), 569–596.

Jean-Éric Pin. 2020. How to prove that a language is regular or star-free?. In *International Conference on Language and Automata Theory and Applications*. Springer, 68–88.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 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)*, Katrin Erk and Noah A. Smith (Eds.). Association for Computational Linguistics, Berlin, Germany, 1715–1725. https://doi.org/10.18653/v1/P16-1162

Lena Strobl, William Merrill, Gail Weiss, David Chiang, and Dana Angluin. 2024. What formal languages can transformers express? a survey. *Transactions of the Association for Computational Linguistics* 12 (2024), 543–561.

Andy Yang, David Chiang, and Dana Angluin. 2024. Masked Hard-Attention Transformers Recognize Exactly the Star-Free Languages. In *Advances in Neural Information Processing Systems (NeurIPS)*. https://arxiv.org/abs/2310.13897 To appear.