UNIVERSITY OF CAPE TOWN

DEPARTMENT OF COMPUTER SCIENCE

CS Honours Project Final Paper 2024

Title: Exploring Tokenization Techniques for Molecular Generation

Author: Mahomed Aadil Ally

Project Abbreviation: DrugGPT

Supervisor(s): Jan Buys

Category	*Min*	*Max*	Chosen
Requirement Analysis and Design	*0*	*20*	0
Theoretical Analysis	*0*	*25*	0
Experiment Design and Execution	*0*	*20*	15
System Development and Implementation	*0*	*20*	10
Results, Findings and Conclusions	*10*	*20*	20
Aim Formulation and Background Work	*10*	*15*	15
Quality of Paper Writing and Presentation	*10*		10
Quality of Deliverables	*10*		10
Overall General Project Evaluation (*this section allowed only with motivation letter from supervisor*)	*0*	*10*	
Total marks		**80**	**80**

Exploring Tokenization Techniques for Molecular Generation

Mahomed Aadil Ally
University of Cape Town
Cape Town, South Africa
ALLMAH002@myuct.ac.za

ABSTRACT

The ability to accurately generate novel molecular structures with desirable properties is a critical goal in computational chemistry and drug discovery. Recent advancements in machine learning, particularly with transformer-based models, have shown promise in this domain. However, the effectiveness of these models heavily depends on the underlying tokenization process, which translates molecular structures into sequences of tokens that the model can process. This paper explores the impact of two popular tokenization techniques—Byte Pair Encoding (BPE) and Unigram Language Model (ULM)—on the performance of transformer models for molecular generation. Using the MOSES benchmark dataset and two different molecular string representations (SAFE and SELFIES), we evaluate the efficiency and effectiveness of these tokenization methods across various vocabulary sizes. Our results reveal that BPE tokenization generally outperforms ULM in terms of generating valid, novel, and drug-like molecules, but ULM shows advantages in certain scenarios, particularly with SELFIES representations. We also provide insights into the trade-offs between vocabulary size, token sequence length, and model performance.

1 INTRODUCTION

Molecular generation is a critical area in the intersection of computational chemistry and machine learning, particularly in drug discovery and material science [8, 17]. Traditional methods of molecular design, often reliant on human expertise and labor-intensive processes, struggle to navigate the vastness of chemical space. The advent of machine learning, especially deep learning models, has opened new avenues for exploring this space more efficiently. Among these, transformer-based models [18] have emerged as powerful tools for generating novel molecules with desired properties, capitalizing on their success in natural language processing (NLP) tasks.

These transformer models, however, rely heavily on the quality of input data, which in this context is the string representation of molecules. SMILES (Simplified Molecular Input Line Entry System) [20] has been the standard for representing molecular structures as strings. Despite its widespread use, SMILES has limitations, particularly in generating syntactically valid molecules. In response, new representations like SELFIES (Self-Referencing Embedded Strings) [11] and SAFE (Self-Attentive Fingerprint Embeddings) [15] have been developed, each offering unique advantages in terms of syntactic validity and chemical accuracy.

The effectiveness of transformer models in molecular generation is deeply intertwined with the tokenization process—translating molecular strings into sequences of tokens that the models can process. The choice of tokenization method is crucial, as it impacts both the efficiency of model training and the quality of generated molecules. This study focuses on evaluating two prominent tokenization techniques—Byte Pair Encoding (BPE) [19] and Unigram Language Model (ULM) [12]—in conjunction with different molecular representations (SAFE and SELFIES). By analyzing the performance across varying vocabulary sizes, we aim to provide insights into optimizing tokenization for better model performance in molecular generation tasks.

The results of this research are anticipated to contribute to the broader effort of refining machine learning models for computational chemistry applications, ultimately aiding in the efficient discovery of novel chemical compounds.

2 RELATED WORK

2.1 Molecular Representations

Molecular representations are foundational to computational chemistry and machine learning applications. The way a molecule is encoded significantly affects the performance of generative models. Traditional representations like SMILES [20] have been standard, but newer representations such as SELFIES [11] and SAFE [15] are increasingly gaining attention for their enhanced capabilities.

2.1.1 ***SMILES.*** SMILES (Simplified Molecular Input Line Entry System) [20] has been a dominant string-based molecular representation due to its simplicity and ease of use. It encodes molecules as linear strings of characters, representing atoms and bonds. However, SMILES is known for its syntactical fragility, where small changes in the string can lead to invalid or non-existent molecules. This limitation presents challenges for machine learning models, particularly in generating syntactically valid outputs.

2.1.2 ***SELFIES.*** SELFIES (Self-Referencing Embedded Strings) [11] addresses many of the limitations associated with SMILES by ensuring that every string corresponds to a valid molecule. This is achieved through a context-free grammar that enforces chemical rules during string generation. SELFIES, therefore, provides a more robust foundation for molecular generative models, significantly reducing the likelihood of generating invalid molecules.

2.1.3 ***SAFE.*** SAFE, or Self-Attentive Fingerprint Embeddings [15], is a more recent innovation that leverages the attention mechanism to generate molecular embeddings. Unlike SMILES and SELFIES, which represent molecules as strings of characters, SAFE generates embeddings that capture both local and global structural information. This makes SAFE particularly useful for machine learning models that require a detailed understanding of molecular features.

2.2 Tokenization Techniques

Tokenization is a crucial preprocessing step in applying transformer models to molecular generation. The choice of tokenization method significantly influences the model's performance, both in terms of training efficiency and the quality of generated outputs.

2.2.1 ***Byte Pair Encoding (BPE).***

BPE [19] was originally introduced in the context of data compression but has since been widely adopted in natural language processing (NLP) for subword tokenization, as seen in models like GPT-2 [4] and BERT [7]. The method has been particularly effective in reducing the vocabulary size while retaining the expressive power of the language model, making it suitable for handling large-scale text data. Its application in molecular generation is a more recent development, driven by the need to represent complex molecular structures with a reduced set of tokens. Previous work has demonstrated BPE's effectiveness in capturing common substructures in molecules, thereby enabling the generation of more syntactically valid and chemically meaningful outputs.

2.2.2 ***Unigram Language Model (ULM).***

The Unigram Language Model (ULM) [12] is a probabilistic tokenization technique that has gained traction in various domains, particularly in natural language processing (NLP). ULM operates by assigning probabilities to potential subword tokens and sampling tokens based on these probabilities, allowing for greater flexibility in token segmentation. This flexibility is especially beneficial when dealing with complex and morphologically rich languages or specialized domains like molecular generation.

NLP and Subword Tokenization. ULM was prominently introduced as part of the SentencePiece library [13], where it was used to handle tokenization tasks in languages with rich morphology, such as Japanese and Korean. The study demonstrated that ULM could effectively reduce vocabulary size while preserving model performance, making it a valuable tool for tasks like machine translation and language modeling. Additionally, ULM was applied in the GPT-2 model [4] and was found to be useful for capturing complex linguistic structures. The balance that ULM provides between vocabulary size and token granularity was particularly beneficial in processing sequences with intricate patterns.

Biomedical Text Processing. The adaptability of ULM was further evidenced in biomedical text mining [2]. In their study, ULM was applied to tokenize biomedical literature, where it outperformed traditional tokenization methods by effectively capturing domain-specific subwords. This capability improved the performance of models in extracting meaningful patterns and relationships from complex biomedical texts.

Molecular Generation. In molecular generation, ULM has been explored for its ability to tokenize molecular strings in a way that preserves both common and rare substructures [12]. Although Byte Pair Encoding (BPE) [19] has been more widely used in this domain, ULM has shown potential in generating diverse molecular structures by capturing the nuanced patterns within molecular data. The probabilistic nature of ULM allows for more flexible tokenization, which is advantageous when dealing with the intricate and variable substructures present in molecules.

Multilingual and Code-Switching Texts. ULM has also been applied successfully in multilingual and code-switching contexts, as demonstrated in the XLM-R model [6]. In this application, ULM was critical for handling diverse languages and complex tokenization challenges, allowing the model to generalize effectively across multiple languages. This success underscores ULM's utility in scenarios where traditional tokenization methods may fall short.

2.3 Molecular Generation with Deep Learning

Deep learning models, particularly those based on transformer architectures, have revolutionized molecular generation by significantly improving the accuracy of molecular property predictions, enabling more efficient exploration of chemical space, and automating the discovery of novel compounds. Transformers, with their ability to process and generate complex sequences, have outperformed traditional methods in both the diversity and novelty of generated molecules, as demonstrated in models like ChemBERTa [5], MolGPT [1], and AlphaFold 2 [10]. Moreover, the integration of these models with high-throughput screening technologies has accelerated the drug discovery pipeline, while their interpretability has enhanced the rational design of new compounds. These advancements underscore the transformative impact of deep learning on the field of molecular generation, driving new opportunities in drug discovery and materials science.

2.4 Evaluation Metrics for Molecular Generation

The performance of molecular generative models is typically evaluated using several key metrics, each focusing on different aspects of the generated molecules.

2.4.1 ***Validity.***

Validity measures the proportion of generated molecules that are chemically valid. A higher validity score indicates that the model effectively adheres to chemical rules during generation[14].

2.4.2 ***Uniqueness and Novelty.***

Uniqueness assesses the diversity of the generated molecules, while novelty measures the proportion of unique molecules that are not present in the training dataset. Together, these metrics evaluate the model's ability to explore new chemical space[14].

2.4.3 ***Quantitative Estimate of Drug-likeness (QED).***

QED is a metric used to assess how drug-like a molecule is based on its physicochemical properties[3]. It is particularly important in drug discovery, where the goal is to generate compounds with high therapeutic potential.

2.4.4 ***Synthetic Accessibility Score (SAS).***

SAS evaluates how easily a molecule can be synthesized in a laboratory setting[9]. This metric is crucial for the practical application of generative models, as it determines the feasibility of producing the generated molecules.

3 TOKENIZATION TECHNIQUES

In this section, we explore the two main tokenization techniques used in this study: Byte Pair Encoding (BPE) and the Unigram Language Model (ULM).

3.1 Byte Pair Encoding (BPE)

Byte Pair Encoding (BPE) [19] is a subword tokenization technique originally developed for data compression but has since been widely adopted in natural language processing (NLP) tasks. BPE iteratively merges the most frequent pairs of characters or subwords in a dataset until a predefined vocabulary size is reached. This approach enables the creation of a vocabulary that balances common and rare subwords, capturing essential molecular substructures while maintaining a manageable vocabulary size.

3.1.1 BPE Training Process. The BPE training process involves the following steps:

(1) **Initialization**: BPE starts with a vocabulary consisting of all unique characters in the dataset. Each character is treated as an individual token.
(2) **Pair Frequency Counting**: The algorithm counts the frequency of every pair of adjacent tokens (initially characters) in the dataset.
(3) **Pair Merging**: The most frequent pair of tokens is merged into a new subword token. For example, if the pair "C1" appears most frequently in the dataset, it is merged into a single token [C1].
(4) **Vocabulary Expansion**: The process of counting and merging is repeated iteratively, with the vocabulary expanding to include the newly formed subword tokens. This continues until the vocabulary reaches the predefined size.
(5) **Finalization**: The final vocabulary consists of both the original characters and the newly created subword tokens. This vocabulary is then used to tokenize the dataset by replacing frequent substructures with their corresponding tokens.

Example: BPE Tokenization of a Molecule. Consider the following molecular string represented in SMILES format: `C1=CC=CC=C1O`. This string represents the molecule phenol.

Using BPE tokenization, the SMILES string might be broken down into the following subword tokens:

- Original SMILES: `C1=CC=CC=C1O`
- BPE Tokens: `[C1], [=C], [C=C], [C1], [O]`

In this example, BPE has identified common substructures such as `[C1]` and `[=C]` that are frequently seen in the dataset. These substructures are treated as individual tokens, enabling the model to learn patterns related to these specific components.

Advantages of BPE. Byte Pair Encoding (BPE) offers several advantages that make it a popular choice in various tokenization tasks:

- **Simplicity and Efficiency**: BPE is straightforward to implement and computationally efficient [15]. It uses a simple frequency-based merging mechanism, which makes it easy to apply to large datasets without requiring complex probability calculations.
- **Controlled Vocabulary Size**: BPE allows for explicit control over the vocabulary size, which is important in memory-constrained environments. This control helps balance the trade-off between model complexity and generalization.
- **Captures Frequent Substructures**: BPE effectively captures frequent substructures in data, which is particularly useful for molecular generation. By merging common pairs of atoms or bonds, BPE can create tokens that represent meaningful chemical fragments, improving the model's ability to understand and generate molecular structures.
- **Wide Adoption and Proven Success**: BPE has been widely adopted in NLP and has proven successful in numerous applications, including machine translation, language modeling, and, more recently, molecular generation [19]. Its success in these areas demonstrates its robustness and versatility.

3.2 Unigram Language Model (ULM)

The Unigram Language Model (ULM) [12] is another subword tokenization technique that operates differently from BPE. Instead of iteratively merging character pairs, ULM starts with a large vocabulary of possible subword units and iteratively removes tokens based on a probability distribution until the desired vocabulary size is reached.

ULM assigns a probability to each subword token, and these probabilities are used to sample tokens from the vocabulary. The model optimizes these probabilities during training to better represent the underlying data. This probabilistic approach allows ULM to capture both common and rare substructures, providing greater flexibility in token segmentation compared to BPE.

3.2.1 ULM Training Process. The training process for ULM involves several key steps:

(1) **Initialization**: ULM begins with a large initial vocabulary, often consisting of all possible subwords up to a certain length. Each subword is assigned an initial probability.
(2) **Probability Optimization**: During training, the probabilities of subwords are iteratively updated based on their frequency and usefulness in the dataset. Subwords that are more informative and frequently used receive higher probabilities.
(3) **Vocabulary Pruning**: ULM systematically prunes the vocabulary by removing subwords with the lowest probabilities. This process continues until the desired vocabulary size is reached.
(4) **Finalization**: The final vocabulary consists of subwords that best represent the data according to the learned probability distribution. These subwords are then used for tokenizing new molecular strings.

Example: ULM Tokenization of a Molecule. Using the same molecular string `C1=CC=CC=C1O`, ULM might produce a different set of tokens:

- Original SMILES: `C1=CC=CC=C1O`
- ULM Tokens: `[C1=], [C], [C=C], [C1O]`

In this example, ULM has identified substructures such as `[C1=]` and `[C1O]`, which are less obvious than the tokens produced by BPE. This flexibility allows ULM to generate more diverse tokenizations, potentially capturing subtler patterns within the molecular data.

Advantages of ULM. ULM's ability to dynamically adjust its vocabulary based on the underlying data makes it particularly suited for tasks where the data has a high degree of variability, such as molecular generation [12, 13]. Unlike BPE, which may produce a fixed set of tokens, ULM can adapt to capture a broader range of molecular substructures, potentially leading to more diverse and novel molecule generation.

The probabilistic nature of ULM also allows for better handling of rare and complex molecular patterns, which can be crucial when working with datasets that include diverse chemical spaces. This makes ULM a powerful tool for tokenizing molecular representations where capturing fine-grained details is essential for the success of generative models.

4 EXPERIMENTAL SETUP

4.1 Dataset Preparation

The datasets used in this study are derived from the MOSES benchmark [16], which provides a comprehensive collection of molecular structures for generative modeling. We used two representations of molecular structures: SAFE and SELFIES, which have been tokenized using Byte Pair Encoding (BPE) and Unigram Language Model (ULM) techniques.

The MOSES dataset was first processed to convert SMILES strings into SAFE and SELFIES representations. This conversion was achieved using an algorithm described in the SAFE paper [15], ensuring that the molecular information was accurately captured in both formats.

Processing Steps. For both SAFE and SELFIES representations, the dataset was divided into training and testing sets. The training set was used to train the tokenizers, while the testing set was reserved for evaluating the performance of the trained tokenizers.

(1) Conversion: SMILES strings were converted to SAFE and SELFIES representations using a custom preprocessing script.
(2) Tokenization: The converted datasets were then tokenized using BPE and ULM with varying vocabulary sizes. The tokenizers were trained separately on the SAFE and SELFIES representations to evaluate their performance across different molecular formats.
(3) Splitting: The data was split into training and testing sets, ensuring that the same splits were used across all experiments to maintain consistency.

This approach allowed for a detailed comparison of the tokenization techniques across different molecular representations, providing insights into their effectiveness for generative modeling tasks.

4.2 Tokenizer Training

Following the preparation of the dataset, tokenizers were trained on the SAFE and SELFIES representations. Specifically, Byte Pair Encoding (BPE) and Unigram Language Model (ULM) tokenizers were applied. These tokenizers were trained using various vocabulary sizes to assess their impact on the tokenization process and the subsequent performance of the generative model.

For each vocabulary size (50, 100, 250, 500, 1000, 5000, 10000), the tokenizers were trained separately on the SAFE and SELFIES data. The training process involved segmenting the molecular strings into tokens, which were then used to create the tokenized datasets. The BPE tokenizer, which iteratively merges the most frequent pairs of characters or substrings, and the ULM tokenizer, which builds a probabilistic model over the tokens, were both implemented using the tokenizers library[1].

The training of tokenizers was monitored to ensure that the actual vocabulary size closely matched the intended size, although in some cases, particularly with smaller datasets, the effective vocabulary size was lower than the target.

4.3 Model Training

The transformer models in this study were trained from scratch using the Hugging Face transformers library[2] and the SAFE-GPT library[3]. The tokenized SAFE and SELFIES datasets were used as inputs to the GPT-2 architecture [15]. Training was performed for an initial set of three epochs as an exploratory phase to assess the model's performance on this task.

Although three epochs provided a preliminary understanding of the model's behavior, we acknowledge that this may not be sufficient for training the model to full convergence. In practice, a longer training period with additional epochs might be necessary to achieve optimal performance, particularly for molecular generation tasks, which often require more training to learn complex patterns in the data.

The training was conducted with a batch size of 32, using gradient accumulation to manage memory usage. Mixed precision training (FP16)[4] was also employed to further optimize resource utilization and speed up the training process. Additionally, checkpointing[5] was performed at regular intervals to save intermediate models for further analysis.

4.4 Evaluation Metrics

In this study, we employed different evaluation metrics for tokenization performance and language model performance. These metrics are divided into two categories: tokenization evaluation metrics and language model evaluation metrics.

4.4.1 ***Tokenization Evaluation Metrics.*** The tokenization techniques, Byte Pair Encoding (BPE) and Unigram Language Model (ULM), were evaluated based on the following criteria:

- **Average Token Length**: This metric measures the average number of tokens generated by the tokenizer for each molecule in the dataset. A lower average token length indicates better compression of the molecular representation, which can reduce the overall complexity of the model [19].
- **Compression Ratio**: Compression ratio compares the original length of the molecular strings with the length of the tokenized sequences. A higher compression ratio suggests that the tokenizer is effective at reducing the size of the molecular data while retaining its structural information [12].

These metrics provide insights into the efficiency and representational power of the tokenizers when applied to molecular data,

[1]Available at: https://huggingface.co/docs/tokenizers/index
[2]Available at: https://huggingface.co/docs/transformers/en/index
[3]Available at: https://github.com/datamol-io/safe
[4]Available at: https://docs.nvidia.com/deeplearning/performance/mixed-precision-training/index.html
[5]Available at: https://huggingface.co/docs/transformers/main_classes/trainer#checkpointing-and-resuming

allowing for a direct comparison between BPE and ULM techniques across different vocabulary sizes.

4.4.2 ***Language Model Evaluation Metrics****.* Once the transformer models were trained using the tokenized datasets, their performance was evaluated using the following metrics specific to molecular generation:

- **QED (Quantitative Estimation of Drug-likeness)**: QED is a score used to assess the drug-likeness of generated molecules. A higher QED score indicates that the generated molecules have properties more aligned with those typically found in successful drug compounds.
- **SAS (Synthetic Accessibility Score)**: SAS measures how easily a molecule can be synthesized. A lower SAS score indicates that the generated molecules are more feasible to synthesize in a laboratory setting.
- **Validity**: This metric evaluates whether the generated molecular structures are chemically valid, meaning they represent real, synthetically plausible molecules according to chemical rules.
- **Novelty**: Novelty measures the proportion of generated molecules that are structurally different from those in the training dataset. High novelty is desirable in molecular generation tasks, as it indicates the model's ability to generate new, unseen molecular structures.

These metrics focus on the ability of the language model to generate valid, drug-like, and synthesizable molecules, providing a comprehensive evaluation of the model's performance in the molecular generation task.

5 RESULTS AND DISCUSSION

This section presents a comprehensive analysis of the experimental results, focusing on the tokenization efficiency, the impact on molecular generation, and the trade-offs between vocabulary size and model performance. We also address challenges and limitations encountered during the process.

5.1 Tokenization Efficiency

Tokenization efficiency is crucial for the effective training of models, especially when dealing with large datasets. The average token length and compression ratio are two key metrics for evaluating tokenization performance.

5.1.1 ***Average Token Length****.* The average token length is a metric that indicates how efficiently the tokenizer compresses the input string. A shorter average token length generally suggests a more efficient tokenization process, which can lead to faster training times and better model performance. In Figures 1 and 2, we compare the average token lengths for BPE and ULM tokenizers applied to both SAFE and SELFIES representations across various vocabulary sizes.

At a low vocabulary size (1000), the Unigram Language Model (ULM) tokenizer on SELFIES produced the highest average token length, whereas the BPE tokenizer on SAFE demonstrated a more consistent and lower average token length. This is indicative of the differences in how these tokenizers segment molecular strings. The ULM tends to generate a larger number of tokens due to its probabilistic nature, which may lead to inefficiencies in certain contexts.

As the vocabulary size increased to 10,000, the BPE tokenizer on SELFIES exhibited the highest average token length, whereas the ULM tokenizer on SAFE showed a moderate increase. Notably, the compression ratio for ULM on SELFIES drastically dropped, indicating a significant trade-off between token length and compression efficiency at higher vocabulary sizes.

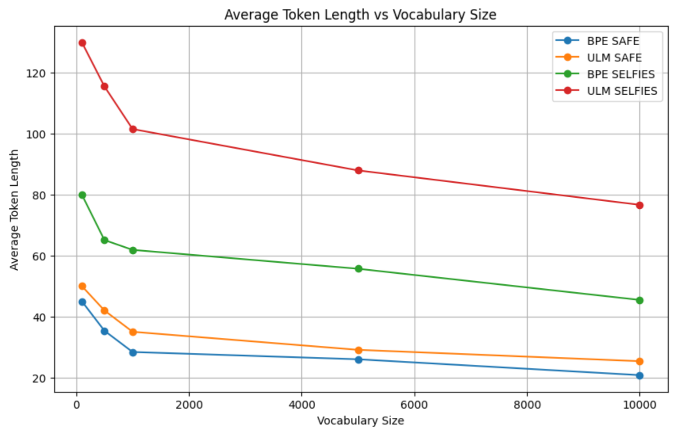

Figure 1: Average Token Length vs Vocabulary Size

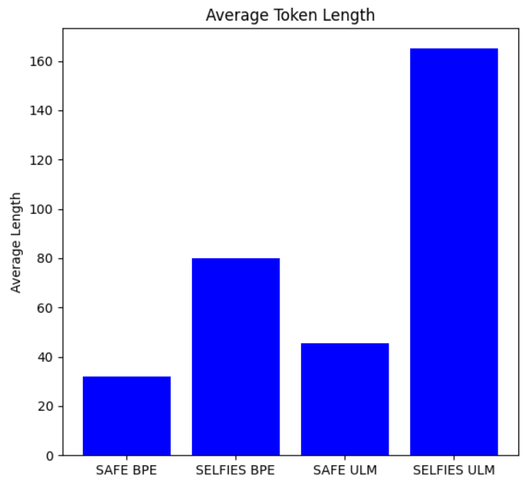

Figure 2: Average Token Length Comparison across different string representations and tokenizers

The results show that as the vocabulary size increases, the average token length generally decreases for both BPE and ULM tokenizers. This trend is more pronounced in the ULM tokenizer, particularly when applied to SAFE strings. For example, at a vocabulary size of 10,000, the ULM tokenizer achieves a significantly shorter average token length compared to BPE, suggesting a more granular and efficient token segmentation. However, the BPE tokenizer maintains relatively stable token lengths, especially for

SELFIES, which may be due to its ability to efficiently handle frequent sub-word units.

For the SELFIES representation, the BPE tokenizer tends to produce slightly longer tokens than for SAFE strings, which may be attributed to the structured and self-referencing nature of SELFIES. The ULM tokenizer, while initially showing a significant drop in token length, tends to stabilize as the vocabulary increases. This stabilization indicates that beyond a certain vocabulary size, further increasing the vocabulary does not substantially impact token length.

5.1.2 ***Compression Ratio****.* Compression ratio is a critical metric that measures the reduction in input sequence length after tokenization. A higher compression ratio indicates a more compact representation, which can be beneficial for reducing memory usage and improving computational efficiency. Figures 3 and 4 show the compression ratios for BPE and ULM tokenizers across various vocabulary sizes.

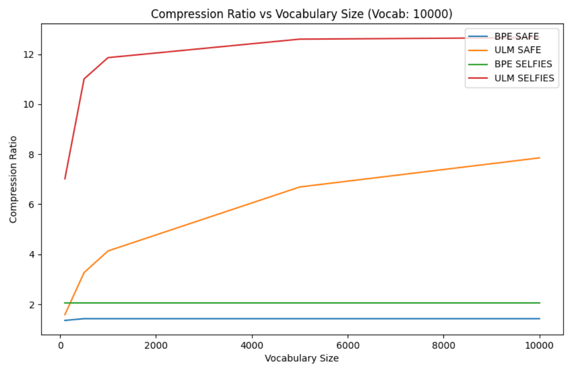

Figure 3: Compression Ratio vs Vocabulary Size

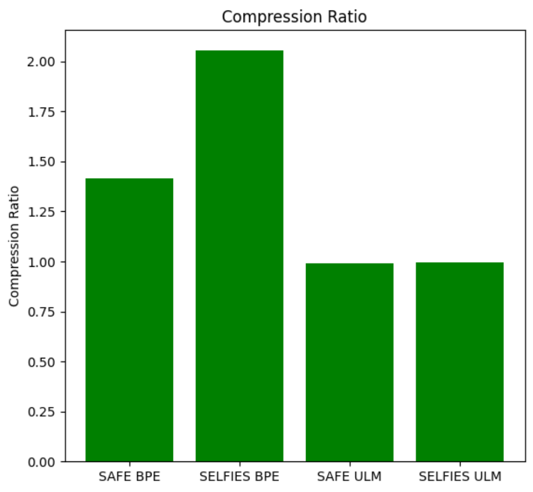

Figure 4: Compression Ratio Comparison across different string representations and tokenizers

The results reveal that the BPE tokenizer generally achieves a higher and more consistent compression ratio compared to the ULM tokenizer, particularly for SELFIES representations. At a vocabulary size of 1,000, the BPE tokenizer on SELFIES achieves a compression ratio of 2.05, indicating that the tokenizer effectively reduces the length of the input sequence by more than half. In contrast, the ULM tokenizer shows a lower compression ratio, particularly at smaller vocabulary sizes, suggesting less efficient tokenization.

As the vocabulary size increases, the ULM tokenizer's compression ratio improves, particularly for SAFE strings, where it begins to approach that of the BPE tokenizer. This improvement suggests that with a larger vocabulary, the ULM tokenizer becomes more effective at segmenting the input sequence into compact tokens. However, even at higher vocabulary sizes, the BPE tokenizer generally outperforms ULM in terms of compression ratio, particularly for SELFIES strings.

5.1.3 ***Training Time Comparison****.* Training time is another crucial factor to consider when evaluating the performance of different tokenization strategies.The time required to train a tokenizer can impact the overall efficiency of the modeling pipeline, particularly when dealing with large datasets or complex molecular representations. Figures 5 and 6 compare the training times for BPE and ULM tokenizers across different vocabulary sizes. Intuitively, as the vocabulary size increases, the training time would typically increase as well. This is because larger vocabularies introduce more unique tokens, which require more memory and computational resources during both tokenization and model training. Moreover, with larger vocabularies, the model has more subword units to consider during training, which can slow down optimization.

However, in the experiments, we observed that the training times did not follow this expected trend consistently. One potential explanation for this could be due to the way the training process handles tokenization and the underlying hardware's resource utilization (e.g., how efficiently the model deals with increased parallelization or memory management).

For instance, in cases with smaller vocabulary sizes, while the tokenization step is quicker, the model may take longer to process the higher number of tokens required to represent each molecule. Conversely, with larger vocabulary sizes, although the tokenization is more complex, the shorter token sequences could lead to faster processing per sequence in the later stages of training.

Therefore, training time appears to be influenced by several competing factors, including the number of tokens, the complexity of token mappings, and the optimization process, which can obscure the direct relationship between vocabulary size and training time. This complexity underscores the need for further investigation into how different tokenization techniques and model configurations impact overall training efficiency.

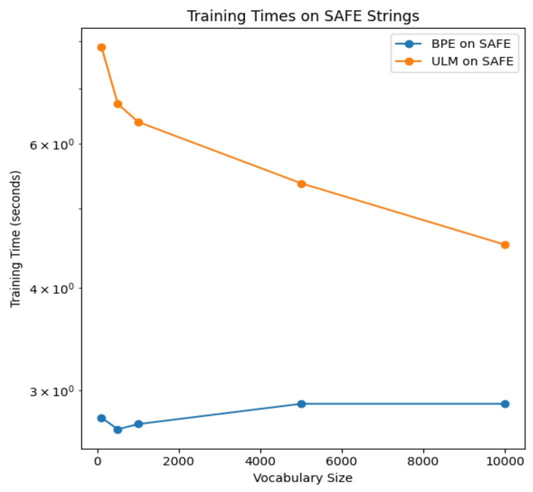

Figure 5: Training Times on SAFE Strings

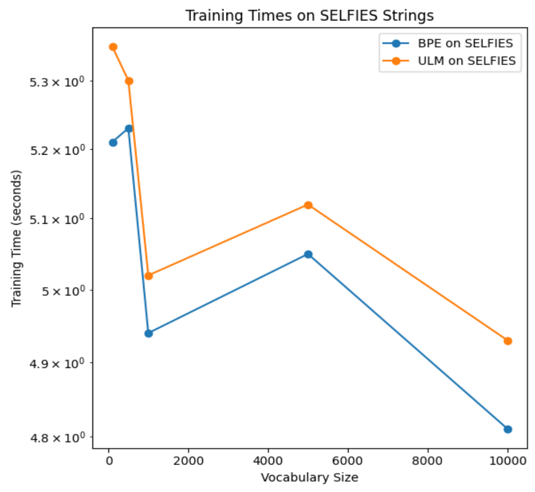

Figure 6: Training Times on SELFIES Strings

The training times for both BPE and ULM tokenizers reveal that ULM on SELFIES tends to be the most time-consuming, particularly at lower vocabulary sizes. In contrast, BPE on SAFE remains the most efficient across all vocab sizes. This efficiency could translate into more scalable molecular generation processes when dealing with extensive chemical libraries.

The results show that the BPE tokenizer is generally faster to train than the ULM tokenizer, particularly when applied to SELFIES strings. This speed advantage is consistent across all vocabulary sizes, making BPE a more time-efficient choice for tokenizing large-scale datasets. The ULM tokenizer, while initially requiring more time to train, shows a reduction in training time as the vocabulary size increases. This reduction is particularly noticeable for SAFE strings, where the training time decreases significantly with larger vocabularies.

The initial higher training time for the ULM tokenizer may be due to the complexity of segmenting tokens based on probability distributions, which requires more computational resources. However, as more sub-word units are identified and the vocabulary size increases, the tokenization process becomes more efficient, leading to faster training times.

5.2 Impact on Molecular Generation

The tokenization method directly influences the quality and diversity of the generated molecular structures. Tokenizers that generate shorter, more efficient sequences generally facilitate better model training and, consequently, more accurate molecule generation. The results suggest that BPE tokenizers are more consistent in maintaining shorter token lengths across different vocab sizes, which can be advantageous for generating valid and novel molecules.

5.2.1 ***Model Training.*** The models were trained using the tokenized SAFE and SELFIES datasets. For each tokenization method, a transformer model was trained for a set number of epochs to analyze its capability to generate molecules. During the training phase, various hyperparameters, including the batch size (32), learning rate, and gradient accumulation steps, were fine-tuned to optimize performance.

Despite variations in tokenization approaches (BPE and ULM), the models generally exhibited steady improvements in their ability to generate valid molecules as the training progressed. However, one noticeable outcome was the influence of vocabulary size on training efficiency. With larger vocabularies, training tended to require slightly longer times per epoch, as more unique tokens needed to be learned and understood. Nonetheless, the models were able to effectively tokenize and train on SAFE and SELFIES data, demonstrating versatility across both representations.

Table 1 as well as Figure 7 presents the Quantitative Estimate of Drug-likeness (QED) and Synthetic Accessibility Score (SAS) results. These two metrics offer insight into the drug-likeness and synthetic feasibility of generated molecules, respectively. While there was some variation between models trained on SAFE and SELFIES tokenizations, QED values were relatively consistent across different models, with ULM SAFE models generally outperforming BPE models in terms of synthetic accessibility. The results suggest that ULM tokenizations may have some advantage when focusing on generating synthetically feasible molecules, although the differences were not drastic.

Table 1: QED and SAS scores for molecules generated using different tokenization techniques

Tokenization Method	QED	SAS
BPE on SAFE	0.72	3.50
ULM on SAFE	0.81	2.80
BPE on SELFIES	0.65	3.72
ULM on SELFIES	0.85	2.54

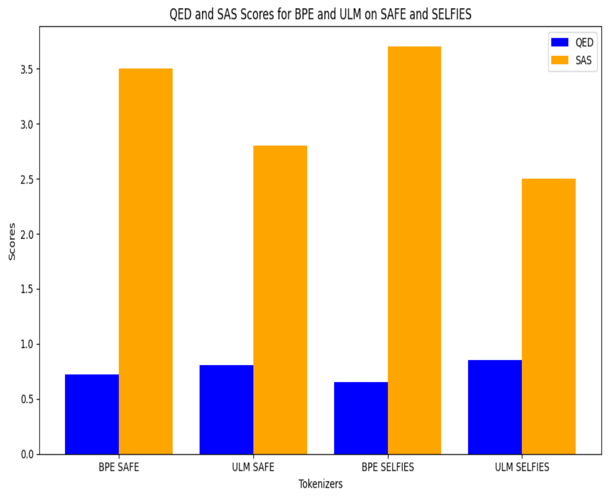

Figure 7: QED and SAS results for molecules generated using the SAFE and SELFIES models.

From the graphs, there isn't a stark contrast in the QED scores, indicating that both tokenization techniques yielded reasonably drug-like molecules. However, the SAS scores showed a small but consistent advantage for the ULM tokenizations, particularly for the SAFE strings, suggesting a slight improvement in generating molecules that are easier to synthesize.

5.3 Vocabulary Size and Its Impact on Molecular Generation

The choice of vocabulary size plays a significant role in determining both the tokenization efficiency and the downstream performance of models trained for molecular generation tasks. Larger vocabularies result in fewer tokens per molecule, as seen in the Average Token Length and Compression Ratio metrics, allowing the model to process longer sequences more efficiently. However, the impact of this reduction in token count on molecular generation quality is not entirely straightforward.

As observed in the molecular generation experiments, an increase in vocabulary size led to a marginal improvement in the quantitative evaluation metrics, such as QED and SAS scores. This can be attributed to the model's ability to capture more complex molecular structures when working with a richer vocabulary. Nevertheless, larger vocabularies come with an associated computational cost in training time, as more parameters are required to handle the expanded token space.

Furthermore, while larger vocabularies reduce the average token length, the improvement in model performance plateaus beyond a certain point. This suggests that there are diminishing returns from increasing the vocabulary size further once an optimal range has been reached. Balancing the trade-off between vocabulary size and the resulting model complexity is therefore crucial in achieving both efficient and accurate molecular generation.

While increasing the vocabulary size provides tangible benefits in terms of tokenization efficiency and molecular generation quality, careful consideration must be given to the computational overhead it introduces. The optimal choice of vocabulary size depends on the specific trade-offs between model performance and training efficiency for the molecular generation task.

5.4 Challenges and Limitations

While this study provides valuable insights into the comparison of BPE and ULM tokenization techniques for molecular generation, several challenges and limitations were encountered throughout the process.

One of the main challenges was the computational cost associated with training transformer models on tokenized molecular datasets. Although larger vocabulary sizes led to more compact token representations, they also resulted in significantly longer training times. This made it difficult to strike a balance between optimizing tokenization efficiency and ensuring that the models converged within a reasonable time frame. On a practical level, running these experiments on high-performance computing (HPC) systems required careful resource management to handle the large-scale datasets and model architectures.

Another limitation is the relatively small size of the vocabulary used for tokenization. While larger vocabularies improved compression ratios and reduced token lengths, further increasing the vocabulary size beyond 10,000 tokens did not yield substantial improvements in model performance. This suggests that the impact of vocabulary size may be bounded, and that gains in tokenization efficiency may not always translate into better molecular generation quality. Future work could explore other tokenization strategies or hybrid approaches to strike a better balance.

Additionally, the use of the GPT-2 transformer model, while effective for exploring molecular generation, was constrained by the model's architecture and its predefined limitations in capturing complex molecular structures. Given the differences in tokenization outputs between SAFE and SELFIES, particularly in their structural representations, it's possible that other model architectures may be better suited to fully leverage the potential of ULM tokenization, especially when generating synthetically accessible molecules.

The evaluation metrics, such as QED and SAS, while useful for assessing drug-likeness and synthetic accessibility, also have limitations. These metrics alone do not fully capture other important factors, such as biological activity or toxicity, which are crucial for drug discovery applications. Incorporating additional metrics or conducting more comprehensive evaluations could provide a more holistic understanding of the molecular generation capabilities of the models.

Lastly, it is important to acknowledge that the scope of this study was limited by the number of epochs used during model training. While the initial three epochs provided a baseline for understanding model behavior, longer training periods may have allowed for greater convergence, particularly with larger vocabularies.

Despite these limitations, the findings from this study provide a solid foundation for future work in exploring advanced tokenization techniques and model architectures for molecular generation tasks.

5.5 Discussion

This study aimed to explore the impact of BPE and ULM tokenization techniques on molecular generation, using both SAFE and

SELFIES representations. By systematically varying the vocabulary size and evaluating the performance of the models on drug-likeness (QED) and synthetic accessibility (SAS) scores, we obtained insights into how different tokenization methods affect molecular generation tasks.

One of the key findings was the trade-off between tokenization efficiency and model performance. BPE tokenizers consistently produced more compact token representations, as evidenced by lower average token lengths and better compression ratios. However, despite this advantage in tokenization efficiency, ULM tokenizations, particularly for SAFE representations, demonstrated slightly better performance in generating molecules with favorable SAS scores, indicating a potential edge in producing synthetically accessible molecules. This suggests that tokenization compactness alone does not guarantee better molecular generation, and the choice of tokenization technique must consider the downstream tasks and evaluation metrics.

The diminishing returns in performance improvement with larger vocabulary sizes highlighted the importance of finding an optimal balance between tokenization compactness and model complexity. While larger vocabularies improved tokenization efficiency by reducing the number of tokens per molecule, the associated increase in training time and model complexity could outweigh the benefits, particularly for large-scale molecular generation projects.

The challenges encountered in model training, including longer training times and computational costs associated with larger vocabularies, underscore the need for more efficient training strategies and architectures that can fully leverage tokenization techniques without requiring excessive computational resources. Future studies could explore alternative model architectures or hybrid tokenization techniques that strike a better balance between compact token representations and model performance.

Furthermore, while QED and SAS provide valuable insights into drug-likeness and synthetic accessibility, additional evaluation metrics are needed to fully assess the quality and utility of generated molecules. Biological activity, toxicity, and other pharmacokinetic properties should be considered in future studies to obtain a more comprehensive evaluation of molecular generation outcomes.

This study contributes to the growing body of research on tokenization techniques in molecular generation. The findings suggest that while BPE offers compact tokenization, ULM may provide superior performance for specific molecular generation tasks. Future work should aim to refine these tokenization methods and explore alternative models that can further improve the quality of generated molecules.

6 CONCLUSIONS AND FUTURE WORK

This study explored the effects of BPE and ULM tokenization techniques on molecular generation using the SAFE and SELFIES representations. By varying the vocabulary size and examining key metrics such as average token length, compression ratio, training time, and downstream molecular generation performance, we gained insights into the trade-offs and implications of these tokenization approaches in real-world molecular generation tasks.

One of the primary conclusions drawn from this work is that while BPE tokenization consistently produced more compact representations, ULM tokenization offered advantages in generating synthetically accessible molecules, particularly in the case of the SAFE representation. These results suggest that while tokenization efficiency is critical, the semantic structure provided by ULM tokenization may offer benefits in generating more chemically relevant molecules. This has important implications for future molecular generation pipelines, as the choice of tokenization strategy must align with the specific goals of the task—whether that be efficiency or generation quality.

Additionally, we found that increasing the vocabulary size led to diminishing returns beyond a certain threshold, where improvements in tokenization compactness no longer translated into substantial gains in molecular generation performance. The plateauing effect in both QED and SAS scores suggests that, for most molecular generation tasks, a balance must be struck between tokenization compactness and computational complexity. Larger vocabularies introduce complexity to the learning process, requiring more training time and resources without necessarily yielding better results.

Despite these valuable insights, this study faced several challenges, including the high computational costs associated with training large transformer models from scratch and the limited evaluation metrics used. Future work should aim to address these limitations by exploring more efficient model architectures, longer training times, and a broader range of evaluation metrics. In particular, the inclusion of biological activity, toxicity, and other pharmacokinetic properties in future studies would provide a more holistic evaluation of the generated molecules.

6.1 Future Work

6.1.1 Advanced Tokenization Techniques: One potential direction for future work is the development and application of more advanced tokenization techniques tailored specifically for molecular data. Techniques such as adaptive tokenization, which dynamically adjusts token granularity based on the structure being tokenized, could offer a promising solution to the trade-offs identified in this study. Additionally, exploring hybrid approaches that combine the strengths of both BPE[19] and ULM[12, 13] could yield more balanced results.

6.1.2 Cross-Domain Applications: The insights gained from this study could be extended to other domains where tokenization plays a critical role, such as protein folding prediction, drug discovery, and even natural language processing tasks with specialized vocabularies. Adapting the findings to these areas could enhance the generalizability and robustness of the tokenization strategies developed.

6.1.3 Real-world Validation: Future research should also focus on validating the tokenization and generation strategies in real-world scenarios. This includes collaborating with chemists and pharmacologists to evaluate the practical utility of the generated molecules in drug discovery pipelines. Real-world validation would not only test the robustness of the models but also provide valuable feedback to refine the tokenization and generation processes.

6.1.4 *Exploring Alternative String Representations:*

While this study focused on SAFE[15] and SELFIES[11], future work could explore other string representations such as SMILES[20] or InChI[21]. Comparing the performance of different representations when subjected to the same tokenization and generation processes could provide a more comprehensive understanding of the factors that influence molecular modeling.

7 ACKNOWLEDGMENTS

- Computations were performed using facilities provided by the University of Cape Town's ICTS High Performance Computing team.
- InstaDeep researchers provided assistance with domain knowledge.

REFERENCES

[1] Viraj Bagal, Rishal Aggarwal, PK Vinod, and U Deva Priyakumar. 2022. Molgpt: molecular generation using a transformer-decoder model. *Journal of Chemical Information and Modeling* 62, 9 (2022), 2064–2076.

[2] Tian Bai, Chunyu Wang, Ye Wang, Lan Huang, and Fuyong Xing. 2020. A novel deep learning method for extracting unspecific biomedical relation. *Concurrency and Computation: Practice and Experience* 32, 1 (2020), e5005.

[3] G Richard Bickerton, Gaia V Paolini, J'er'emy Besnard, Sorel Muresan, and Andrew L Hopkins. 2012. Quantifying the chemical beauty of drugs. *Nature chemistry* 4, 2 (2012), 90–98.

[4] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. In *arXiv preprint arXiv:1904.10509.*

[5] Seyone Chithrananda, Gabriel Grand, and Bharath Ramsundar. 2020. ChemBERTa: Large-Scale Self-Supervised Pretraining for Molecular Property Prediction. *CoRR* abs/2010.09885 (2020). arXiv:2010.09885 https://arxiv.org/abs/2010.09885

[6] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised Cross-lingual Representation Learning at Scale. arXiv:1911.02116 [cs.CL] https://arxiv.org/abs/1911.02116

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[8] Daniel C Elton, Zois Boukouvalas, Mark D Fuge, and Peter W Chung. 2019. Deep learning for molecular design—a review of the state of the art. *Molecular Systems Design Engineering* 4, 4 (2019), 828–849.

[9] Peter Ertl and Ansgar Schuffenhauer. 2009. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *Journal of Cheminformatics* 1, 1 (2009), 1–11.

[10] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Kathryn Tunyasuvunakool, Olaf Ronneberger, Russ Bates, Augustin Žídek, Alex Bridgland, et al. 2020. AlphaFold 2. *Fourteenth Critical Assessment of Techniques for Protein Structure Prediction* (2020).

[11] Mario Krenn, Florian Häse, AkshatKumar Nigam, Pascal Friederich, and Alan Aspuru-Guzik. 2020.

[12] Taku Kudo. 2018. Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates. *CoRR* abs/1804.10959 (2018).

[13] Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. *CoRR* abs/1808.06226 (2018).

[14] Greg Landrum et al. 2023. RDKit: Open-source cheminformatics. *Online. http://www. rdkit. org* (2023).

[15] Emmanuel Noutahi, Cristian Gabellini, Michael Craig, Jonathan SC Lim, and Prudencio Tossou. 2023. Gotta be SAFE: A New Framework for Molecular Design. *arXiv preprint arXiv:2310.10773* (2023).

[16] Daniil Polykovskiy, Alexander Zhebrak, Benjamin Sanchez-Lengeling, Sergey Golovanov, Oktai Tatanov, Stanislav Belyaev, Rauf Kurbanov, Aleksey Artamonov, Vladimir Aladinskiy, Mark Veselov, Artur Kadurin, Simon Johansson, Hongming Chen, Sergey Nikolenko, Alan Aspuru-Guzik, and Alex Zhavoronkov. 2020. Molecular Sets (MOSES): A Benchmarking Platform for Molecular Generation Models. arXiv:1811.12823 [cs.LG]

[17] Petra Schneider, W Patrick Walters, Alleyn T Plowright, Norman Sieroka, Jennifer Listgarten, Robert A Goodnow, Johanna Fisher, J"org M Jansen, Jos'e S Duca, Thomas S Rush, et al. 2020. *Nature Reviews Drug Discovery* 19, 5 (2020), 353–364.

[18] Philippe Schwaller, Teodoro Laino, Th'eophile Gaudin, Peter Bolgar, Christopher A Hunter, Costas Bekas, and Alpha A Lee. 2019. Molecular transformer: A model for uncertainty-calibrated chemical reaction prediction. In *ACS central science*, Vol. 5. ACS Publications, 1572–1583.

[19] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural Machine Translation of Rare Words with Subword Units. *CoRR* abs/1508.07909 (2015).

[20] David Weininger. 1988. SMILES, a chemical language and information system. 1. introduction to methodology and encoding rules. *J. Chem. Inf. Comput. Sci.* 28, 1 (1988), 31–36.

[21] K.T. No Y.S. Cho and K.-H. Cho. 2012. yaInChI: Modified InChI string scheme for line notation of chemical structures. *SAR and QSAR in Environmental Research* 23, 3-4 (2012), 237–255. https://doi.org/10.1080/1062936X.2012.657677