GRAPHBTE: GRAPH BYTE TRIPLE ENCODING

Anonymous authors
Paper under double-blind review

ABSTRACT

Molecular property prediction commonly uses graph-based GNNs or SMILES-based Transformers, where SMILES tokenization captures molecular substructures beyond individual atoms. However, BPE-based SMILES tokenization has two problems regarding invariance under SMILES enumeration: SMILES-enumeration dependence and inconsistent subgraph tokenization. Direct tokenization of molecular graphs addresses these limitations but leads to two additional problems: connectivity uncertainty in pair-based merging and loss of atom-level positions. To address these limitations, we propose Graph Byte Triple Encoding (GraphBTE), which represents each merge rule using two adjacent tokens and the canonical representation of their merged subgraph, while incorporating atom-level positional information through spectral positional encoding (SPE). Across seven MoleculeNet classification datasets, GraphBTE outperforms GraphBPE with a GNN and achieves higher average ROC-AUC than SMILES BPE under the same Transformer architecture. Further analyses show that GraphBTE is invariant to SMILES enumeration, consistently tokenizes shared subgraphs in the analyzed examples, and distinguishes different merged subgraphs formed from the same token pair. SPE further improves prediction performance, and the best-performing vocabulary size is strongly associated with molecular size.

1 INTRODUCTION

Models for molecular property prediction commonly use graph-based representations with GNNs (Gilmer et al., 2017; Yang et al., 2019) or SMILES-based representations with Transformers (Chithrananda et al., 2020; Honda et al., 2019). In SMILES-based Transformers, molecules are represented as tokenized SMILES sequences (Weininger, 1988).

Tokenization has been widely studied in natural language processing (NLP), where character sequences are segmented into subword units that capture more meaningful linguistic patterns than individual characters (Mielke et al., 2021). A similar motivation applies to molecular representation, where molecular properties are influenced not only by individual atoms but also by molecular substructures (Zhang et al., 2021). For example, a single carbon atom alone may not be sufficient to characterize the properties of a molecule, whereas a group of carbon atoms forming a benzene ring exhibits distinct structural properties such as aromaticity (Breslow, 2014).

To capture such substructures in molecules, molecular fragmentation has been extensively studied in cheminformatics. Conventional fragmentation methods define such substructures using predefined chemical rules, which may reflect human bias and limit the diversity of substructures extracted during fragmentation (Bemis & Murcko, 1996b; Shao et al., 2024; López-López et al., 2026). To reduce dependence on these predefined criteria, an alternative is to learn frequent substructures directly from data. One such data-driven approach in NLP is byte-pair encoding (BPE), which iteratively merges frequent adjacent token pairs (Sennrich et al., 2016). BPE has also been applied to SMILES-based molecular models (Li & Fourches, 2021).

However, BPE-based SMILES tokenization has a limitation regarding invariance under SMILES enumeration, which leads to two problems.(1) SMILES-enumeration dependence. A single molecule can have multiple SMILES representations (Arús-Pous et al., 2019), which can result in different tokenizations, as illustrated on the left side of Figure 1(a). (2) Inconsistent subgraph tokenization. The same subgraph in different molecules can be represented by different SMILES sequences and consequently tokenized differently, as illustrated on the right side of Figure 1(a).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

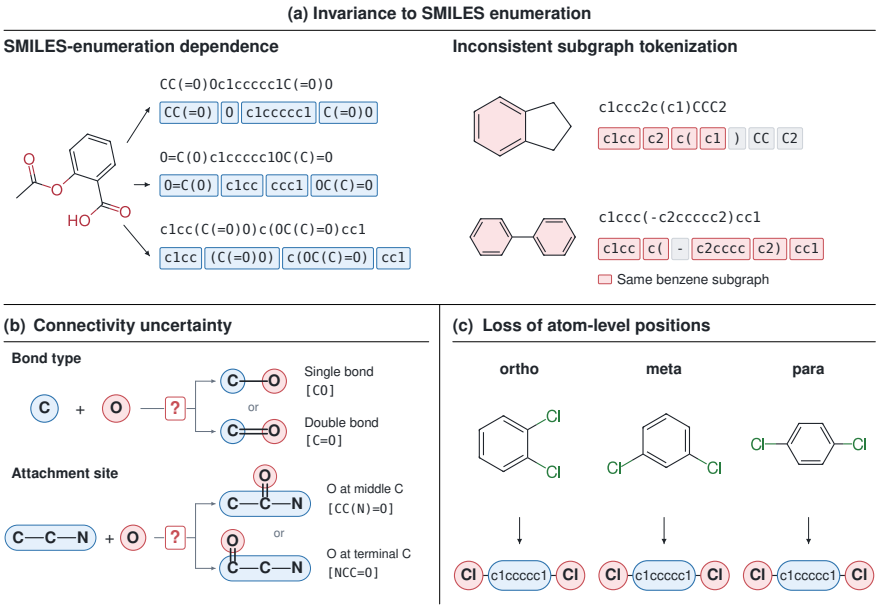


Figure 1: Motivating limitations of SMILES BPE and pair-based graph tokenization. (a) Invariance to SMILES enumeration. Different SMILES representations of the same molecule can result in different tokenizations, and the same subgraph in different molecules can be tokenized differently. (b) Connectivity uncertainty. The same token pair can form different merged subgraphs depending on the bond type or attachment site. (c) Loss of atom-level positions. Merging atoms into subgraph tokens can result in the loss of atom-level positional information, causing ortho, meta, and para isomers to result in the same tokenized graph.

These limitations motivate direct tokenization of molecular graphs. Applying pair-based BPE to molecular graphs leads to additional problems: (1) Connectivity uncertainty. In sequence BPE, a token pair is merged by concatenating the two tokens. In molecular graphs, however, the same token pair can form different subgraphs depending on the bond type and attachment site, as illustrated in Figure 1(b). Therefore, a token pair alone cannot correspond uniquely to a single subgraph. After tokenization, another problem can arise: (2) Loss of atom-level positions. As atoms are merged into a single token, the token graph retains only the connections between tokens, resulting in the loss of the original atom-level connectivity information. As shown in Figure 1(c), ortho, meta, and para isomers can result in the same tokenized graph.

To address these problems, we propose Graph Byte Triple Encoding (GraphBTE), which learns substructure tokens directly from molecular graphs. It extends pair-based encoding to a triple-based merge rule $\mathrm{Merge}(a, b, c_{ab})$, where c_{ab} represents the merged subgraph in canonical SMILES. By including c_{ab} in the merge rule, GraphBTE can distinguish different merged subgraphs formed from the same token pair. In addition, we incorporate SPE computed on the original atom graph to preserve atom-level positional information. Across seven MoleculeNet classification datasets, GraphBTE outperforms GraphBPE by 2.88 ROC-AUC points on average and achieves higher ROC-AUC on all seven datasets.

Our contributions are as follows:

- We identify four limitations in SMILES- and graph-based molecular tokenization and propose GraphBTE to address them.
- We propose GraphBTE, which learns substructure tokens directly from molecular graphs and extends pair-based encoding to a triple-based merge rule $\mathrm{Merge}(a, b, c_{ab})$.
- We incorporate SPE to preserve atom-level positional information during graph tokenization.

2 Related Work

2.1 Byte-Pair Encoding

BPE learns a vocabulary by iteratively merging frequent adjacent token pairs (Sennrich et al., 2016). Starting from initial tokens, BPE repeatedly identifies the most frequent adjacent pair, merges it into a new token, and adds the token to the vocabulary.

SMILES-based BPE is widely used to represent molecules as token sequences for Transformer-based models (Chithrananda et al., 2020). For example, CCO is first initialized as [C,C,O]. If [C,O] is selected as the most frequent pair, it is merged to produce [C,CO]. If [C,CO] is selected in a later iteration, the sequence is further merged into [CCO]. Through this process, frequently occurring SMILES patterns can be represented as larger tokens (Li & Fourches, 2021).

GraphBPE applies BPE directly to molecular graphs (Shen & Póczos, 2024). Starting from atom-level tokens, it repeatedly merges frequent adjacent token pairs to construct larger subgraph tokens and learn a graph-based vocabulary. GraphBPE provides a customizable contextualization framework but does not specify a concrete task-dependent contextualization for this downstream setting. Therefore, we did not apply additional task-dependent contextualization in our experiments.

2.2 Molecular Fragmentation Baselines

We compare GraphBTE with representative molecular fragmentation methods that decompose molecules into chemically meaningful substructures based on predefined rules (Lounkine et al., 2008; Jinsong et al., 2024). The methods are described below:

- **Atom-lvel tokenization** uses each atom as a token without fragmentation.
- **Motif fragmentation** groups rings that share atoms into ring-system motifs (Steinbeck et al., 2006), while non-ring regions are treated separately.
- **BRICS** fragments molecules by cleaving bonds between atom environments that match predefined BRICS rules (Degen et al., 2008).
- **RECAP** fragments molecules by cleaving non-ring bonds that match predefined RECAP reaction templates (Lewell et al., 1998).
- **Murcko scaffolds** retain the core scaffold consisting of ring systems and the linkers connecting them while removing peripheral side chains (Bemis & Murcko, 1996a).

3 GraphBTE Framework

Figure 2 provides an overview of GraphBTE. The framework consists of learning a substructure vocabulary from molecular graphs and tokenizing molecular graphs using the learned merge rules. SPE is applied to incorporate atom-level positional information into the token representations.

3.1 Learning Vocabulary and Merge Rules through Triple-Based Merging

We define each merge rule as $m = \mathrm{Merge}(a, b, c_{ab})$, where a and b are adjacent tokens and c_{ab} is the canonical SMILES representation of their merged subgraph (Weininger et al., 1989). The rule merges a matching token pair only when its combined subgraph has canonical representation c_{ab}. Thus, c_{ab} distinguishes different bond types or attachment sites for the same token pair, while providing a consistent representation for identical merged subgraphs.

Figure 2(a) illustrates the vocabulary learning process of GraphBTE. In **Iteration 1**, adjacent atoms are considered as token pairs. For each pair (a, b), the merged subgraph c_{ab} is constructed to form a candidate merge rule $\mathrm{Merge}(a, b, c_{ab})$. The occurrences of each triple are counted across the corpus, and the most frequent triple is selected as a merge rule. The selected triple is added to the merge-rule list, while its merged subgraph c_{ab} is added to the token vocabulary. The selected rule is then applied to the token graphs, producing the updated token graphs shown on the right. In **Iteration 2**, the same procedure is repeated on the merged token graphs obtained from Iteration 1, where newly adjacent tokens form the next candidate triples. This process is repeated for the predefined number

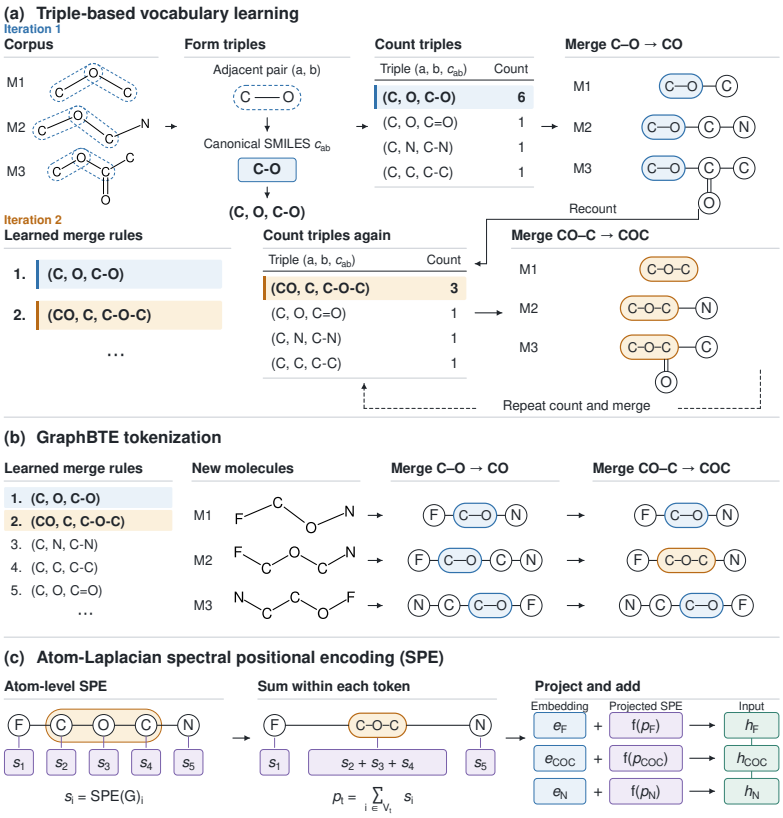


Figure 2: Overview of GraphBTE. (a) Vocabulary learning process of GraphBTE through iterative triple-based merging. (b) Molecular graph tokenization process using the learned merge rules. (c) Application of atom-level SPE to token representations.

of merge iterations. As a result, GraphBTE can distinguish different merged subgraphs even when they are formed from the same token pair. The overall procedure is presented in Algorithm 1.

3.2 Tokenizing Molecular Graphs with Learned Merge Rules

The learned merge-rule list is used to tokenize new molecules into substructure-level token graphs. Figure 2(b) illustrates the tokenization process of GraphBTE. The learned merge rules are applied sequentially to the input molecule. In the first step, the connected C and O tokens are merged into a single CO token using the merge rule $\mathrm{Merge}(\mathrm{C}, \mathrm{O}, \mathrm{CO})$. The next rule, $\mathrm{Merge}(\mathrm{CO}, \mathrm{C}, \mathrm{COC})$, is applied. In the original molecular graph of M2, the C atom is connected to the O atom of the CO subgraph, so the two tokens are merged into a single COC token. This process is repeated sequentially for the remaining merge rules.

In contrast, M3 also contains an adjacent CO and C token pair. However, in the original molecular graph, the C atom is connected to the C atom of the CO subgraph rather than to its O atom. Therefore, the pair does not match the merge rule $\mathrm{Merge}(\mathrm{CO}, \mathrm{C}, \mathrm{COC})$ and is not merged by this rule. If the merge rule were defined only by the token pair $(\mathrm{CO}, \mathrm{C})$, these two merged structures could not be distinguished. GraphBTE addresses this ambiguity by including the connectivity between a and b in c_{ab}, allowing different attachment positions to be distinguished. The overall procedure is presented in Algorithm 2 in the Appendix.

3.3 Preserving Atom-Level Positional Information in Subgraph Tokens

In GraphBTE, merging multiple atoms into a single token may not retain atom-level positional information. As illustrated in Figure 1(c), ortho, meta, and para isomers can produce the same token-

level graph even though their substitution positions are different. To preserve this information, we compute SPE from the original atom graph (Dwivedi et al., 2023). Specifically, we perform eigen-decomposition on the Laplacian matrix of the original atom graph to obtain a positional encoding for each atom. Since each token can contain multiple atoms, we aggregate the positional encodings of its constituent atoms to obtain a single positional encoding for the token. Let V_t denote the set of atoms contained in token t, and let PE_v denote the positional encoding of atom v. The positional encoding of token t is computed as:

$$PE_t = \sum_{v \in V_t} PE_v. \tag{1}$$

The aggregated positional encoding PE_t is then incorporated into the token embedding as:

$$h_t = e_t + f(PE_t), \tag{2}$$

where e_t denotes the embedding of token t from the vocabulary, and $f(\cdot)$ projects PE_t to the token embedding dimension. The projection layer is initialized with small parameter values to limit the initial influence of positional information on the token representation. The Laplacian construction, spectral encoding, and projection settings are detailed in Appendix C.1.

4 Experimental Setup

4.1 Datasets and Evaluation Protocol

We evaluate on seven molecular property classification datasets from MoleculeNet (Wu et al., 2018). BACE evaluates BACE-1 inhibition, BBBP evaluates blood brain barrier penetration, and SIDER contains drug side-effect prediction tasks. Tox21 and ToxCast contain toxicity-related prediction tasks, HIV evaluates anti-HIV activity, and MUV contains bioactivity prediction tasks. Detailed dataset statistics are provided in Appendix B.

Each dataset was split into training, validation, and test sets at a ratio of 8:1:1 using scaffold splitting (Wu et al., 2018). For evaluation, ROC-AUC was computed for each task and macro-averaged over valid tasks. Unless otherwise stated, ROC-AUC scores are reported on a 0-100 scale. All experiments were repeated with five random seeds, and the mean and standard deviation were reported.

4.2 Vocabulary Construction and Token Budgets

GraphBTE, GraphBPE, and the fragmentation-based methods use the same corpus of 250,000 molecules from ZINC15 (Sterling & Irwin, 2015) for vocabulary construction. We implemented GraphBPE and the fragmentation-based baselines for our experiments. GraphBPE provides a customizable contextualization framework but does not specify a concrete contextualization for this downstream setting; therefore, we used its graph-based pair merging without additional contextualization. For the fragmentation-based methods, each fragment is converted to canonical SMILES and treated as a vocabulary token. In our implementation, Motif treats each non-ring atom as an individual token. BRICS uses the fragments obtained after bond cleavage without retaining attachment-point dummy atoms. For RECAP, the connected components obtained after cleaving matched non-ring bonds are used as tokens rather than the recursive RecapDecompose output. For Murcko, the entire scaffold is used as one token and each remaining connected component as a separate token; molecules without a Murcko scaffold are represented at the atom level. The resulting vocabulary sizes are reported in Appendix Table 6.

For GraphBTE, the initial vocabulary consisted of 18 tokens, including atom-level tokens and three special tokens, `<pad>`, `<mask>`, and `<unk>`. We used vocabulary sizes of 38 and 68 for the main GNN and Transformer experiments, respectively, corresponding to 20 and 50 additional tokens learned through triple-based merging. For GraphBPE, we used a vocabulary size of 68. The vocabulary sizes for GraphBTE and GraphBPE were selected based on the highest average validation ROC-AUC across the seven datasets. For the SMILES BPE baseline, we used the 767-token vocabulary provided by ChemBERTa (Chithrananda et al., 2020).

Table 1: Molecular property prediction performance of GraphBTE and alternative molecular representations on seven MoleculeNet datasets. Graph-based methods use a one-layer GIN, while SMILES BPE uses a Transformer. Results are ROC-AUC scores (higher is better), reported as mean ± standard deviation over five random seeds with scaffold splitting. AVG denotes the average across the seven datasets. Bold and underlined values indicate the highest and second-highest mean scores in each column, respectively, based on unrounded scores.

Method	BACE	BBBP	SIDER	Tox21	ToxCast	HIV	MUV	AVG (↑)
GraphBTE (Ours)	<u>76.8</u> ±3.8	**70.4** ±0.9	**64.4** ±0.9	**73.4** ±0.3	**64.8** ±0.5	**76.2** ±1.3	**70.4** ±2.2	**70.9**
GraphBPE	69.9 ±2.5	67.0 ±0.5	62.4 ±0.7	72.3 ±0.8	<u>63.1</u> ±0.4	<u>74.6</u> ±1.4	66.9 ±2.6	<u>68.0</u>
Atom	71.1 ±2.3	64.3 ±1.5	<u>62.8</u> ±1.2	<u>72.3</u> ±0.2	62.5 ±0.3	74.4 ±1.3	66.8 ±1.5	67.7
Motif	72.7 ±2.0	<u>68.9</u> ±0.9	56.5 ±0.8	71.0 ±0.8	59.2 ±2.4	72.3 ±2.5	<u>70.0</u> ±1.5	67.2
BRICS	69.9 ±1.5	60.8 ±0.7	54.8 ±1.8	63.3 ±1.4	57.8 ±0.3	69.0 ±1.6	69.6 ±1.4	63.6
RECAP	63.9 ±1.0	62.5 ±2.2	56.3 ±1.5	59.7 ±1.1	52.0 ±0.8	62.3 ±0.8	64.1 ±2.4	60.1
Murcko	**77.0** ±1.3	66.4 ±0.5	58.9 ±1.0	63.3 ±0.5	55.2 ±0.4	66.4 ±1.3	51.7 ±1.3	62.7
SMILES BPE	68.3 ±3.5	68.2 ±1.6	56.4 ±1.9	70.9 ±0.8	59.4 ±1.1	69.3 ±2.2	62.7 ±6.4	65.0

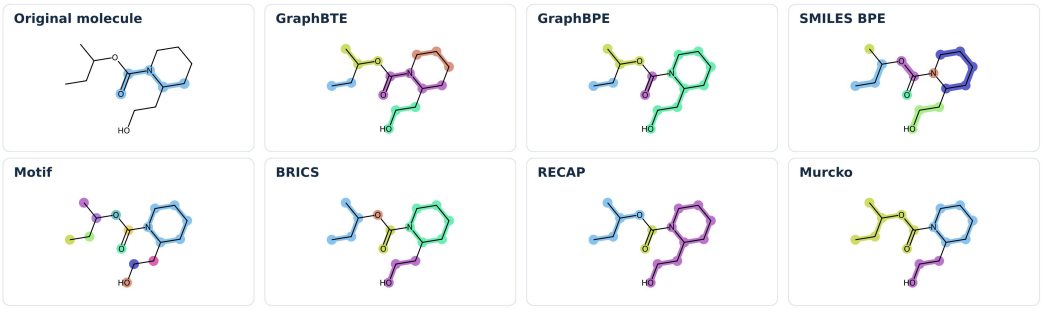


Figure 3: Tokenization examples for the same molecule using GraphBTE, GraphBPE, SMILES BPE, Motif, BRICS, RECAP, and Murcko. Colors indicate different tokens in each tokenized representation. Each method divides the same molecule into different substructures, showing the difference between learned tokenization methods and predefined fragmentation methods.

Table 2: Effect of SPE on GraphBTE in terms of ROC-AUC (↑) (mean ± standard deviation over five repetitions). The two settings differ only in the use of SPE.

Setting	BACE	BBBP	SIDER	Tox21	ToxCast	HIV	MUV	AVG (↑)
w SPE	**76.8** ±3.8	**70.4** ±0.9	**64.4** ±0.9	**73.4** ±0.3	**64.8** ±0.5	**76.2** ±1.3	**70.4** ±2.2	**70.9**
w/o SPE	71.7 ±1.6	69.6 ±1.9	62.2 ±0.8	72.4 ±0.8	63.1 ±0.3	75.0 ±0.7	70.3 ±2.5	69.2

4.3 GNN AND TRANSFORMER ARCHITECTURES

For the graph-based experiments, we used a Graph Isomorphism Network (GIN) (Xu et al., 2019) as the backbone. Since tokenized graphs contain fewer nodes than the original atom-level graphs, we evaluated GINs with one to four layers. One- and two-layer models generally showed the highest validation performance, and we selected a one-layer GIN for the main experiments. The results are provided in Appendix Table 8. The hidden dimension was set to 300. For GNN pooling, we used mean pooling (Gilmer et al., 2017). The resulting graph-level representation was passed through a two-layer MLP for final prediction. For GraphBTE, SPE was additionally applied.

For the Transformer-based experiments, we used a four-layer Transformer (Vaswani et al., 2017) based on HuggingFace RoBERTa (Liu et al., 2019; Wolf et al., 2020). The hidden dimension was set to 256, with four attention heads and an intermediate dimension of 512. GraphBTE used SPE, whereas the SMILES BPE baseline used the absolute positional encoding of ChemBERTa.

5 Experimental Results

5.1 Prediction Performance across Molecular Tokenization Methods

Table 1 presents the prediction results for GraphBTE and the baseline methods. GraphBTE achieves the best performance on six of the seven datasets, with the highest average ROC-AUC of 70.90, outperforming Atom (67.73) by 3.17 points on average. GraphBTE outperforms GraphBPE by 2.88 ROC-AUC points on average and achieves higher ROC-AUC on all seven datasets. Motif performs well on BBBP and MUV but shows lower performance on SIDER, whereas Murcko performs well on BACE but poorly on MUV. These results indicate that predefined fragmentation methods show dataset-specific strengths, suggesting that a fixed fragmentation method may not perform consistently across different molecular property tasks. In contrast, GraphBTE achieves consistently strong performance across datasets by learning substructure tokens directly from molecular graphs.

Figure 3 shows an example of how different methods tokenize the same molecule. The highlighted substructure is commonly observed among molecules in the positive class of the corresponding ToxCast task. Fragmentation-based methods generally preserve ring systems as tokens according to predefined structural rules. The highlighted substructure in the original molecule does not match these ring-based boundaries and includes only part of the ring together with neighboring atoms. GraphBPE and SMILES BPE also learn tokens from the data but do not represent this substructure as a single token. However, GraphBTE represents the highlighted substructure as a single token.

5.2 Invariance to SMILES Enumeration

Figure 4 compares prediction performance under canonical and randomized SMILES. This experiment examines invariance to SMILES enumeration by evaluating SMILES-enumeration dependence. To examine this, we keep the trained model and test molecules fixed and vary only the SMILES enumeration at test time. Canonical SMILES provide a fixed SMILES representation for each molecule, whereas randomized SMILES represent the same molecule using randomly generated SMILES sequences (Weininger et al., 1989; Arús-Pous et al., 2019).

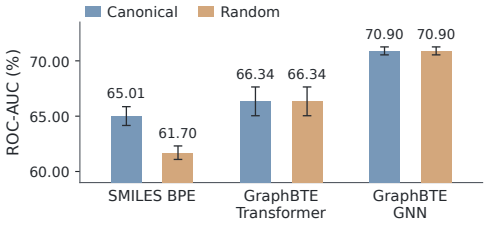


Figure 4: Prediction performance under canonical and randomized SMILES. Bars show the average ROC-AUC across seven datasets over five repetitions, and error bars represent standard deviations. Blue and orange indicate canonical and randomized SMILES, respectively.

For SMILES BPE, the average ROC-AUC decreases by 3.31 points with randomized SMILES compared with canonical SMILES. SMILES BPE shows lower ROC-AUC on six of the seven datasets with randomized SMILES, with the largest decreases on BACE and BBBP at 6.80 and 4.77 points, respectively. For the same molecule, changing only its SMILES representation results in different prediction performance with SMILES BPE. Regardless of whether performance improves or declines, this variation demonstrates a fundamental limitation of SMILES BPE.

In contrast, GraphBTE shows the same performance with canonical and randomized SMILES because tokenization is performed on the molecular graph rather than directly on the SMILES representation. For the Transformer, this invariance is further preserved by using graph-based SPE instead of sequence-based absolute positional encoding, avoiding dependence on token order. These results show that GraphBTE preserves prediction performance under the evaluated SMILES enumerations.

5.3 Effect of Spectral Positional Encoding

To evaluate the effect of SPE, we compare GraphBTE with and without SPE, with the vocabulary size and model architecture fixed. The results are shown in Table 2. The average ROC-AUC increases from 69.18 to 70.90 when SPE is applied. Even without SPE, GraphBTE achieves a higher average ROC-AUC than GraphBPE, indicating that its performance gain is not solely attributable to positional encoding. Performance improves on all seven datasets, with the largest gains observed on

Table 3: Prediction performance of GraphBTE and SMILES BPE using the same Transformer architecture in terms of ROC-AUC (↑) (mean ± standard deviation over five repetitions).

Method	BACE	BBBP	SIDER	Tox21	ToxCast	HIV	MUV	AVG (↑)
GraphBTE (Ours)	**74.8** ±3.8	64.6 ±1.7	**60.3** ±3.1	**71.7** ±0.8	**61.7** ±1.1	68.7 ±4.5	62.6 ±2.4	**66.3**
SMILES BPE	68.3 ±3.5	**68.2** ±1.6	56.4 ±1.9	70.9 ±0.8	59.4 ±1.1	**69.3** ±2.2	**62.7** ±6.4	65.0

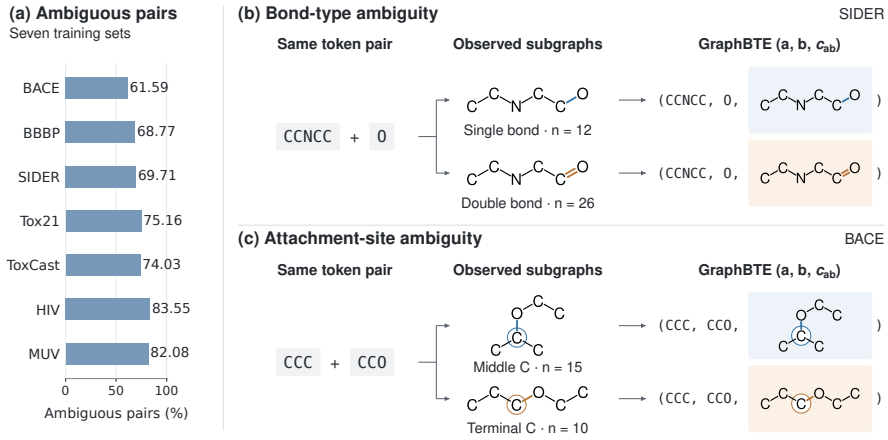


Figure 5: Structural ambiguity in pair-based merging across seven MoleculeNet training sets. Bars show the percentage of distinct adjacent token pairs in the final token graphs produced by GraphBTE with a vocabulary size of 38 that correspond to multiple merged subgraphs. Examples from SIDER and BACE show differences in (b) bond type and (c) attachment site, respectively. GraphBTE distinguishes these structures through different merge rules $\mathrm{Merge}(a, b, c_{ab})$.

BACE and SIDER. SPE is designed to address the loss of atom-level positions by incorporating positional information from the original molecular graph into the token representations. The ablation results show that this additional information improves prediction performance. Additional results for the fragmentation-based methods with SPE are reported in Appendix Table 9.

5.4 Analysis of Subgraph Consistency and Connectivity Uncertainty

To examine inconsistent subgraph tokenization under different molecular contexts, we qualitatively analyze identical subgraphs across molecules. As shown in the right panel of Figure 1(a), SMILES BPE can divide the same benzene subgraph into different tokens across molecules. In contrast, GraphBTE tokenizes the same benzene subgraph consistently, as shown in Appendix Figure 7.

We further quantify the connectivity uncertainty of pair-based graph tokenization, as shown in Figure 5. Across the seven training sets, 61.59–83.55% of adjacent token-pair types are ambiguous. Detailed statistics are provided in Appendix Table 10. GraphBTE distinguishes all observed merged subgraphs formed from the same token pair by including c_{ab} in the merge rule $\mathrm{Merge}(a, b, c_{ab})$. Representative examples from SIDER and BACE are shown in Figure 5, with additional examples provided in Appendix Table 11.

5.5 GraphBTE and SMILES BPE with a Transformer

In the GNN experiments, we evaluated GraphBTE on tokenized molecular graphs. Since SMILES BPE does not directly construct a token graph, we evaluate both GraphBTE and SMILES BPE using the same Transformer architecture to examine whether GraphBTE is also effective with Transformer-based models. The two methods differ only in their tokenization and positional encoding: GraphBTE uses SPE, whereas SMILES BPE uses the absolute positional encoding of ChemBERTa. The results are shown in Table 3.

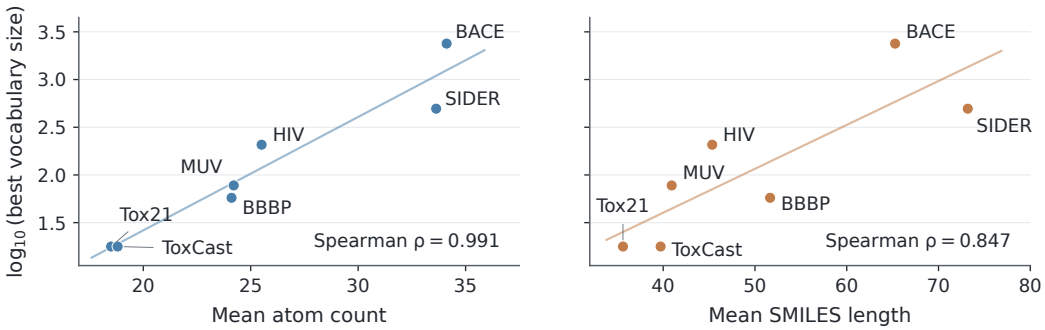


Figure 6: Molecular size is measured by mean atom count (left) and mean SMILES length (right). Each point represents a dataset, and the lines indicate linear fits. Spearman correlations are shown in each panel.

GraphBTE outperforms SMILES BPE on four of the seven datasets and improves the average ROC-AUC from 65.01 to 66.34. The largest improvements occur on BACE and SIDER, at 6.51 and 3.91, respectively. Since both methods use the same Transformer architecture, these results suggest that the advantage of GraphBTE is not limited to GNNs and also extends to Transformer-based models.

5.6 Post Hoc Analysis of Vocabulary Size and Molecular Size

Based on the vocabulary-size sweep in Appendix Figure 8, we conduct a post hoc analysis of the relationship between molecular size and vocabulary size. For each dataset, the best-performing vocabulary size is defined as the one with the highest observed test ROC-AUC. The dataset-specific results are summarized in Table 7 in the appendix. Although a vocabulary size of 68 achieves the highest average ROC-AUC across the seven datasets, the best-performing vocabulary size varies across datasets.

To examine whether this variation is related to molecular size, we analyze the correlation between the best-performing vocabulary size and the average number of atoms and SMILES length of each dataset, as shown in Figure 6. The best-performing vocabulary size shows a very strong positive correlation with the average number of atoms (Spearman $\rho = 0.991$) and a strong positive correlation with the average SMILES length (Spearman $\rho = 0.847$). Within the evaluated range, datasets with larger molecules tend to achieve their highest observed test performance with larger vocabularies. These results suggest that molecular size may be a useful factor for vocabulary-size selection.

6 Conclusion

In this study, we propose GraphBTE, a data-driven graph tokenization method for molecular property prediction. Our analyses show that SMILES-enumeration dependence can affect prediction performance and that the same token pair frequently forms different subgraphs depending on the bond type or attachment site. We also show that atom-level positional information can be lost when atoms are merged into subgraph tokens. GraphBTE addresses these problems by representing each merge rule using two adjacent tokens and the canonical representation of their merged subgraph, while incorporating atom-level positional information through SPE. By addressing these limitations, GraphBTE improves prediction performance over GraphBPE with a GNN and over SMILES BPE under the same Transformer architecture.

The post hoc analysis further shows that the best-performing vocabulary size varies across datasets and is associated with molecular size. Future work should explore methods for automatically selecting the vocabulary size based on molecular size.

REFERENCES

Josep Arús-Pous, Simon Viet Johansson, Oleksii Prykhodko, Esben Jannik Bjerrum, Christian Tyrchan, Jean-Louis Reymond, Hongming Chen, and Ola Engkvist. Randomized SMILES strings improve the quality of molecular generative models. *Journal of Cheminformatics*, 11(1):71, 2019. doi: 10.1186/s13321-019-0393-0.

Guy W. Bemis and Mark A. Murcko. The properties of known drugs. 1. molecular frameworks. *Journal of Medicinal Chemistry*, 39(15):2887–2893, 1996a. doi: 10.1021/jm9602928.

Guy W. Bemis and Mark A. Murcko. The properties of known drugs. 1. molecular frameworks. *Journal of Medicinal Chemistry*, 39(15):2887–2893, 1996b. doi: 10.1021/jm9602928.

Ronald Breslow. Novel aromatic and antiaromatic systems. *The Chemical Record*, 14(6):1174–1182, 2014. doi: 10.1002/tcr.201402070.

Seyone Chithrananda, Gabriel Grand, and Bharath Ramsundar. ChemBERTa: Large-scale self-supervised pretraining for molecular property prediction. *arXiv preprint arXiv:2010.09885*, 2020. URL https://arxiv.org/abs/2010.09885.

Jörg Degen, Christof Wegscheid-Gerlach, Andrea Zaliani, and Matthias Rarey. On the art of compiling and using 'drug-like' chemical fragment spaces. *ChemMedChem*, 3(10):1503–1507, 2008. doi: 10.1002/cmdc.200800178.

Vijay Prakash Dwivedi, Chaitanya K. Joshi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *Journal of Machine Learning Research*, 24(43):1–48, 2023. URL https://jmlr.org/papers/v24/22-0567.html.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pp. 1263–1272, 2017. URL https://proceedings.mlr.press/v70/gilmer17a.html.

Shion Honda, Shoi Shi, and Hiroki R. Ueda. SMILES Transformer: Pre-trained molecular fingerprint for low data drug discovery. *arXiv preprint arXiv:1911.04738*, 2019. URL https://arxiv.org/abs/1911.04738.

Shao Jinsong, Jia Qifeng, Chen Xing, Yajie Hao, and Li Wang. Molecular fragmentation as a crucial step in the ai-based drug development pathway. *Communications Chemistry*, 7(1):20, 2024. doi: 10.1038/s42004-024-01109-2.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. URL https://arxiv.org/abs/1412.6980.

Xiao Qing Lewell, Duncan B. Judd, Stephen P. Watson, and Michael M. Hann. RECAP—retrosynthetic combinatorial analysis procedure: A powerful new technique for identifying privileged molecular fragments with useful applications in combinatorial chemistry. *Journal of Chemical Information and Computer Sciences*, 38(3):511–522, 1998. doi: 10.1021/ci970429i.

Xinhao Li and Denis Fourches. SMILES pair encoding: A data-driven substructure tokenization algorithm for deep learning. *Journal of Chemical Information and Modeling*, 61(4):1560–1569, 2021. doi: 10.1021/acs.jcim.0c01127.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. URL https://arxiv.org/abs/1907.11692.

Edgar López-López, José L. Medina-Franco, and Filip Miljković. Crush—cleavage rules using smirks heuristics: an enhanced molecular fragmentation algorithm. *Journal of Cheminformatics*, 18(1):78, 2026. doi: 10.1186/s13321-026-01239-w.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1711.05101.

Eugen Lounkine, José Batista, and Jürgen Bajorath. Random molecular fragment methods in computational medicinal chemistry. *Current Medicinal Chemistry*, 15(21):2108–2121, 2008. doi: 10.2174/092986708785747607.

Sabrina J. Mielke, Zaid Alyafeai, Elizabeth Salesky, Colin Raffel, Manan Dey, Matthias Gallé, Arun Raja, Chenglei Si, Wilson Y. Lee, Benoît Sagot, and Samson Tan. Between words and characters: A brief history of open-vocabulary modeling and tokenization in nlp. *arXiv preprint arXiv:2112.10508*, 2021.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 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)*, pp. 1715–1725, 2016. doi: 10.18653/v1/P16-1162. URL https://aclanthology.org/P16-1162/.

Jinsong Shao, Qifeng Jia, Xing Chen, Yajie Hao, and Li Wang. Molecular fragmentation as a crucial step in the AI-based drug development pathway. *Communications Chemistry*, 7:20, 2024. doi: 10.1038/s42004-024-01109-2.

Yuchen Shen and Barnabás Póczos. GraphBPE: Molecular graphs meet byte-pair encoding. *arXiv preprint arXiv:2407.19039*, 2024. URL https://arxiv.org/abs/2407.19039.

Christoph Steinbeck, Christian Hoppe, Stefan Kuhn, Matteo Floris, Rajarshi Guha, and Egon L. Willighagen. Recent developments of the chemistry development kit (cdk) – an open-source java library for chemo- and bioinformatics. *Current Pharmaceutical Design*, 12(17):2111–2120, 2006. doi: 10.2174/138161206777585274.

Teague Sterling and John J. Irwin. ZINC 15—ligand discovery for everyone. *Journal of Chemical Information and Modeling*, 55(11):2324–2337, 2015. doi: 10.1021/acs.jcim.5b00559.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://arxiv.org/abs/1706.03762.

D. Weininger, A. Weininger, and J. L. Weininger. SMILES. 2. algorithm for generation of unique SMILES notation. *Journal of Chemical Information and Computer Sciences*, 29(2):97–101, 1989. URL https://www.daylight.com/meetings/summerschool02/course/basics/ref/smiles2.html.

David Weininger. SMILES, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28(1):31–36, 1988. doi: 10.1021/ci00057a005.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In Qun Liu and David Schlangen (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6. URL https://aclanthology.org/2020.emnlp-demos.6/.

Zhenqin Wu, Bharath Ramsundar, Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, and Vijay Pande. MoleculeNet: A benchmark for molecular machine learning. *Chemical Science*, 9(2):513–530, 2018. doi: 10.1039/C7SC02664A.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1810.00826.

Kevin Yang, Kyle Swanson, Wengong Jin, Connor Coley, Philipp Eiden, Hua Gao, Angel Guzman-Perez, Timothy Hopper, Brian Kelley, Miriam Mathea, Andrew Palmer, Volker Settels, Tommi Jaakkola, Klavs Jensen, and Regina Barzilay. Analyzing learned molecular representations for property prediction. *arXiv preprint arXiv:1904.01561*, 2019. URL https://arxiv.org/abs/1904.01561.

Zaixi Zhang, Qi Liu, Hao Wang, Chengqiang Lu, and Chee-Kong Lee. Motif-based graph self-supervised learning for molecular property prediction. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

A Vocabulary Learning and Tokenization Algorithms

We distinguish the token vocabulary $\mathcal{V}$, which contains atom, special, and learned subgraph tokens, from the ordered list $\mathcal{R}$ of merge rules. Here, $\mathcal{V}_{\text{atom}}$ denotes the initial atom-token vocabulary. Algorithm 1 returns both objects, and Algorithm 2 applies the rules in $\mathcal{R}$. When multiple valid occurrences overlap, matches are resolved using a canonical graph-based ordering independent of the input SMILES enumeration.

Algorithm 1 GraphBTE vocabulary learning through iterative triple-based merging.

Require: Training molecules $\mathcal{D}$; number of merge rules K
Ensure: Token vocabulary $\mathcal{V}$ and ordered merge-rule list $\mathcal{R}$
Initialize every molecule in $\mathcal{D}$ with atom-level tokens
$\mathcal{V} \leftarrow \mathcal{V}_{\text{atom}} \cup \{\texttt{<pad>}, \texttt{<mask>}, \texttt{<unk>}\}$
$\mathcal{R} \leftarrow [\,]$
for $k = 1, \ldots, K$ **do**
    Initialize an empty frequency map F
    for all molecules $G \in \mathcal{D}$ **do**
        for all adjacent token pairs (a, b) in G **do**
            Merge (a, b) and encode the result c_{ab} as canonical SMILES
            $F[\text{Merge}(a, b, c_{ab})] \leftarrow F[\text{Merge}(a, b, c_{ab})] + 1$
        end for
    end for
    $m^* = \text{Merge}(a^*, b^*, c^*_{ab}) \leftarrow \arg\max_m F[m]$
    Append m^* to $\mathcal{R}$
    $\mathcal{V} \leftarrow \mathcal{V} \cup \{c^*_{ab}\}$
    Merge valid non-overlapping occurrences of m^* in $\mathcal{D}$
end for
return $(\mathcal{V}, \mathcal{R})$

Algorithm 2 Tokenization of a molecular graph using the ordered GraphBTE merge rules.

Require: Molecular graph G; ordered merge-rule list $\mathcal{R}$
Ensure: Token graph G_{token}
Initialize every atom in G as an atom-token node
$G_{\text{token}} \leftarrow G$
for all merge rules $m = \text{Merge}(a, b, c_{ab})$ in $\mathcal{R}$, in learned order **do**
    Find adjacent tokens in G_{token} that match m
    for all valid non-overlapping matches **do**
        Merge the matched tokens into substructure token c_{ab}
        Retain edges connecting c_{ab} to the remaining token nodes
    end for
end for
return G_{token}

B Dataset Statistics

Table 4: Statistics of the seven MoleculeNet datasets used in the experiments. Tasks and Molecules indicate the numbers of prediction tasks and molecules, respectively. Avg. atoms and Avg. bonds indicate mean atom and bond counts per molecule.

Dataset	Type	Tasks	Molecules	Avg. atoms	Avg. bonds
BACE	Classification	1	1,513	34.1	36.9
BBBP	Classification	1	2,039	24.1	26.0
SIDER	Classification	27	1,427	33.6	35.4
Tox21	Classification	12	7,831	18.6	19.3
ToxCast	Classification	617	8,576	18.8	19.3
HIV	Classification	1	41,127	25.5	27.5
MUV	Classification	17	93,087	24.2	26.3

C Training and Implementation Details

All models are trained from scratch on downstream tasks without self-supervised pretraining. The GNN uses Adam (Kingma & Ba, 2015) with learning rate 10^{-3}, and the Transformer uses AdamW (Loshchilov & Hutter, 2019) with learning rate 5×10^{-4}. Both use weight decay 10^{-5}, batch size 64, and 50 epochs. ReduceLROnPlateau with patience 8 and factor 0.5 is used for both; the Transformer additionally uses gradient clipping with maximum norm 5.0. The checkpoint with the highest validation ROC-AUC is selected for final evaluation.

Table 5: Training settings for the GNN and Transformer experiments. Both models are trained from scratch for 50 epochs using ReduceLROnPlateau. Gradient clipping is applied only to the Transformer.

Setting	GNN	Transformer
Optimizer	Adam	AdamW
Learning rate	10^{-3}	5×10^{-4}
Weight decay	10^{-5}	10^{-5}
Batch size	64	64
Epochs	50	50
Scheduler	ReduceLROnPlateau (patience 8, factor 0.5)	
Gradient clipping	–	Maximum norm 5.0

C.1 SPE Implementation Details

For SPE, we use the symmetric normalized graph Laplacian,

$$L = I - D^{-1/2} A D^{-1/2},$$

computed from the unweighted and undirected atom graph. We compute the eigenpairs in ascending eigenvalue order, exclude the first eigenpair, and retain up to the next 10 eigenpairs. When fewer than 10 are available, zero padding is applied.

The retained eigenvalues are passed through an MLP with dimensions $1 \to 32 \to 10$ and a ReLU activation. The positional encoding of atom v is computed as

$$PE_v = \sum_j u_j(v)^2 \phi(\lambda_j),$$

where λ_j and u_j denote the retained eigenvalue and eigenvector, respectively, and ϕ denotes the eigenvalue MLP.

For each token, the positional encodings of its constituent atoms are summed to obtain PE_t. The resulting 10-dimensional token positional encoding is projected to the model embedding dimension using a linear layer, with dimensions $10 \to 300$ for the GNN and $10 \to 256$ for the Transformer. The projection weights are initialized from $\mathcal{N}(0, 0.02^2)$ with zero bias.

D Qualitative Comparisons of Molecular Tokenization

D.1 Consistent Tokenization across Molecular Contexts

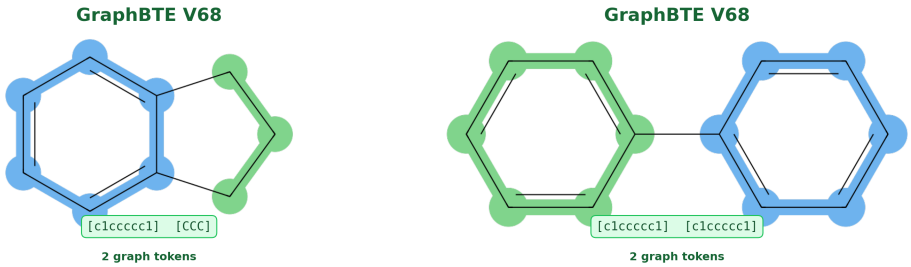


Figure 7: Consistent subgraph tokenization across molecular contexts using GraphBTE with a vocabulary of 68 tokens. The same benzene subgraph is represented by the same token in both molecules. Colors indicate individual graph tokens, and bracketed strings show their SMILES representations.

E Vocabulary Sizes and Additional Performance Analysis

E.1 Baseline Vocabulary Sizes

Table 6: Comparison of vocabulary sizes for atom-level tokenization and fragmentation-based methods. Fragmentation vocabularies are constructed from 250,000 ZINC15 molecules.

Atom	Motif	BRICS	RECAP	Murcko scaffold
18	253	664	879	269

E.2 Prediction Performance across Vocabulary Sizes

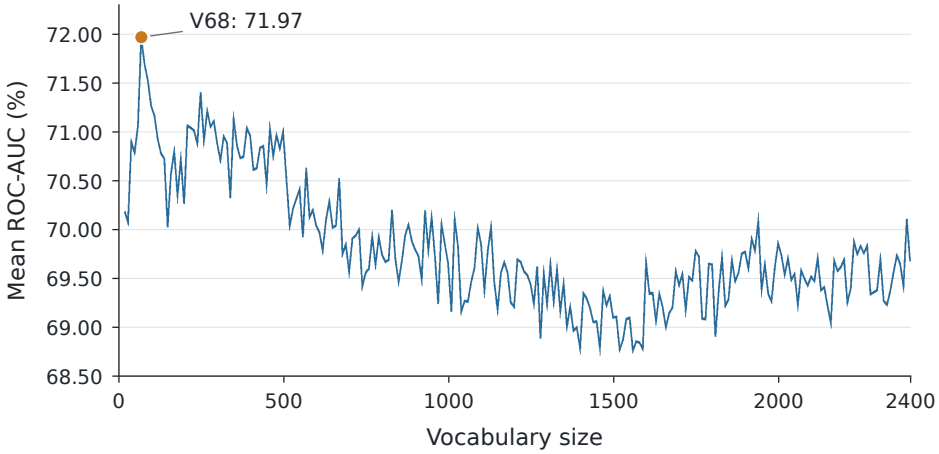


Figure 8: Post hoc analysis of vocabulary size in GraphBTE with a one-layer GIN. The curve shows average test ROC-AUC across seven datasets, and the highlighted point indicates the highest observed score at a vocabulary size of 68. The main-experiment vocabulary size of 38 was selected using validation ROC-AUC.

As a post hoc analysis, we evaluate GraphBTE across vocabulary sizes from V18 to V2398. Figure 8 shows the average test ROC-AUC across the seven datasets. The highest observed score is 71.97 at V68, while the main experiments use V38 selected based on validation ROC-AUC.

E.3 Dataset-Specific Best Vocabulary Sizes

Table 7: Post hoc analysis of vocabulary size in GraphBTE with a GNN. Best vocabulary denotes the vocabulary size with the highest mean test ROC-AUC in the sweep. Results are reported as mean ± standard deviation over five repetitions.

Dataset	Best vocabulary	ROC-AUC	Mean atoms	Mean SMILES length
BACE	V2388	82.2 ± 2.9	34.1	65.2
BBBP	V58	70.9 ± 1.3	24.1	51.6
SIDER	V498	67.7 ± 0.9	33.6	73.1
Tox21	V18	75.4 ± 0.2	18.6	35.6
ToxCast	V18	65.8 ± 0.3	18.8	39.7
HIV	V208	77.7 ± 0.8	25.5	45.3
MUV	V78	73.3 ± 0.8	24.2	40.9

F Prediction Performance across GNN Depths

Table 8: Effect of GNN depth on prediction performance in terms of average ROC-AUC (↑) across seven MoleculeNet datasets (mean ± standard deviation). GraphBTE uses SPE, and its vocabulary size is selected using average validation ROC-AUC for each depth. Atom and the fragmentation baselines use no positional encoding. Bold and underlined values indicate the highest and second-highest mean scores within each method, respectively; rankings use unrounded scores.

Method	GNN 1 layer	GNN 2 layers	GNN 3 layers	GNN 4 layers
GraphBTE (Ours)	**70.9 ±0.4** (V38)	<u>70.7</u> ±1.2 (V58)	69.6 ±1.0 (V48)	69.1 ±1.1 (V48)
Motif	67.2 ±0.8	**67.6 ±0.6**	<u>67.6</u> ±0.7	67.3 ±0.6
Atom	<u>67.7</u> ±0.6	**68.1 ±1.1**	66.6 ±1.0	67.7 ±0.4
RECAP	<u>60.1</u> ±0.5	60.0 ±0.3	60.1 ±1.0	**60.1 ±0.7**
BRICS	**63.6 ±0.9**	63.4 ±0.9	<u>63.4</u> ±0.9	63.3 ±0.4
Murcko scaffold	**62.7 ±0.2**	62.1 ±0.4	<u>62.4</u> ±0.8	61.7 ±0.9

Table 9: Effect of GNN depth on fragmentation-based representations with SPE in terms of average ROC-AUC (↑) across seven MoleculeNet datasets (mean ± standard deviation). All four methods use SPE. Bold and underlined values indicate the highest and second-highest mean scores within each method, respectively; rankings use unrounded scores.

Method	GNN 1 layer	GNN 2 layers	GNN 3 layers	GNN 4 layers
Motif	68.7 ±0.6	**70.0 ±0.6**	<u>69.0</u> ±0.4	<u>69.0</u> ±0.8
BRICS	**66.7 ±0.5**	66.0 ±0.5	<u>66.2</u> ±0.5	65.7 ±0.3
RECAP	65.1 ±0.4	**65.2 ±1.0**	64.9 ±0.5	<u>65.2</u> ±0.6
Murcko scaffold	**64.6 ±0.5**	64.1 ±0.8	<u>64.5</u> ±0.7	63.5 ±0.7

G Structural Ambiguity of Token Pairs

G.1 Ambiguity Rates across Datasets

We compute structural ambiguity separately for each dataset using the final token graphs produced by the V38 tokenizer on its training set. Let $\mathcal{P}$ denote the set of distinct unordered pairs of adjacent token IDs, written as $p = \{a, b\}$. Pairs with identical token IDs are also included. For each pair p, let $\mathcal{S}_p$ be the set of distinct merged subgraphs observed for that pair across the entire training set. Merged subgraphs are distinguished by their canonical SMILES representations.

Pair-level ambiguity. A token pair is ambiguous if it corresponds to at least two distinct merged subgraphs:

$$\mathcal{P}_{\text{amb}} = \{p \in \mathcal{P} : |\mathcal{S}_p| \geq 2\}. \tag{3}$$

The ambiguous pair rate is the percentage of unique token pairs that are ambiguous:

$$\text{Ambiguous pair rate} = \frac{|\mathcal{P}_{\text{amb}}|}{|\mathcal{P}|} \times 100. \tag{4}$$

Pair ambiguity is determined across the entire dataset training set, rather than within an individual molecule.

The requirement of at least 10 molecules per merged-subgraph variant is applied only when selecting the examples in Table 11, not when computing the ambiguous pair rates in Table 10. Table 10 reports the pair-level ambiguity rates.

Table 10: Structural ambiguity of token pairs across seven MoleculeNet training sets. Unique pairs indicate distinct adjacent token pairs after tokenization, and Ambiguous pairs indicate token pairs that correspond to multiple merged subgraphs. Rate indicates the percentage of unique pairs that are ambiguous.

Dataset	Training molecules	Unique pairs	Ambiguous pairs	Rate (%)
BACE	1,210	289	178	61.6
BBBP	1,567	381	262	68.8
SIDER	1,141	383	267	69.7
Tox21	6,264	455	342	75.2
ToxCast	6,860	462	342	74.0
HIV	32,901	547	457	83.6
MUV	74,469	413	339	82.1

G.2 Examples of Structurally Ambiguous Token Pairs

Table 11: Examples of structurally ambiguous token pairs across seven MoleculeNet datasets. Each token pair corresponds to multiple merged subgraphs. Labeled mols. indicates the number of labeled training molecules containing each merged subgraph, and Positive rate indicates the percentage with a positive label for the selected task.

Dataset	Canonical merged subgraph	Labeled mols.	Positive rate (%)
BACE	`CCOC(C)C`	15	0.0
	`C1CCOCC1`	16	31.3
	`CCCOCC`	13	84.6
	`CCCC(C)O`	19	94.7
BBBP	`CN[C@@H](C)C(C)C`	11	9.1
	`CCCN(C)CC`	13	100.0
	`CN1CCCCC1`	24	100.0
SIDER	`CCNC(C)=O`	18	33.3
	`CCNCCO`	12	58.3
	`CCNCC=O`	26	73.1
	`CCNCC[O-]`	10	100.0
Tox21	`CC1=CCCCC1`	11	0.0
	`CCC(C)C(C)C`	15	0.0
	`CC/C=C\CCC`	16	6.3
	`CCCC(C)CC`	19	15.8
	`CCCCCCC`	138	19.6
	`CC1CCCCC1`	79	34.2
	`CCCCC(C)C`	43	53.5
	`CCC[C@@H](C)CC`	42	61.9
	`CC[C@@H](C)C(C)C`	11	63.6
	`CCC[C@H](C)CC`	20	65.0
	`CC1C=CCCC1`	30	90.0
	`CC1CC[C@@H](C)C1`	12	100.0
	`CC1CC[C@H](C)C1`	10	100.0
ToxCast	`C1=CCC=CC1`	12	0.0
	`C1CCCCC1`	12	0.0
	`CC(C)C(C)C`	13	0.0
	`CC/C=C\CC`	12	0.0
	`CCC=C(C)C`	17	0.0
	`CCCC(C)C`	121	13.2
	`CCC=CCC`	11	100.0
HIV	`ccccNC`	11	0.0
	`ccc(c)NC`	35	2.9
	`ccccCN`	28	14.3
	`ccc(c)CN`	35	62.9
MUV	`cc(n)C=O`	152	0.0
	`nccC=O`	188	1.6
	`ccnC=O`	20	35.0