Transformer based seq2seq models for Chemical Reaction Prediction: CS 4644

Terry Ma
terryma@gatech.edu

Naiji Jabin Gong
ngong6@gatech.edu

Siddharth Pillai
spillai42@gatech.edu

Abstract

Predicting chemical reactions has various scientific industrial, and practical applications. Current chemical reaction prediction is done by searching vast realms of scientific literature to find experimental results from a specific reaction. We implemented a natural language deep learning-based approach to predict chemical reactions. By examining several new tokenization and input encoding schemes for a dataset of chemical reactions in string form and training a transformer model, we were able to obtain reasonably accurate predictors of chemical reactions and demonstrate an improvement over simpler methods of tokenization.

1. Introduction

Chemical reactions are fundamental to the understanding of various fields, ranging from chemistry to material science, drug discovery, and even environmental studies. Deep learning methods used in recent years has led to significant progress in the prediction of chemical reactions, paving the way for the accelerated discovery of new molecules and materials.

Deep learning uses artificial neural networks to learn patterns and make predictions from large amounts of data. This approach has been remarkably successful in a variety of applications like computer vision and natural language processing. Discoveries like convolutional neural networks, recurrent neural networks, and transformers have been essential towards the breakthroughs that we have seen across many fields.

In the field of chemistry, deep learning has emerged as a powerful tool for predicting chemical reactions. Though machine learning has been used extensively to predict chemical reactions, deep learning has unlocked new approaches to the problem. By training on large sets of experimental reaction data, deep learning models can identify complex patterns to make highly accurate predictions and derive new insights about chemistry. This technology can significantly reduce the time and resources required for laboratory experimentation, accelerate the discovery of new reactions and materials, and generate new insights into chemical systems [4].

Our report aims to implement a Transformer model with new tokenization and input encoding schemes, inspired by *"Found in Translation": Predicting Outcomes of Complex Organic Chemistry Reactions using Neural Sequence-to-Sequence Models* [7]. We demonstrated the utility of a novel tokenization scheme for chemical reaction prediction.

1.1. State-of-the-art

Schwaller and Laino [8] conducted a review of recent approaches in data-driven chemical reaction prediction. Here we will summarize some of the points made.

Template-based methods is one of the most commonly used methods for chemical reaction prediction. These models leverage known reaction templates, which are generalized representations of reaction types, to predict the outcome of a given set of reactants. Machine learning models are trained to identify the most likely templates that can be applied to the reactants, which are then used to predict the products. Limitations of this method is that it has limited generalization. The performance of these methods depends on the quality and diversity of the templates in the database. If a reaction type is not well-represented in the template library, prediction accuracy can suffer. Additionally, there is an inability to predict novel reactions. These methods struggle to predict reactions that don't fit any existing template, limiting their usefulness for exploring new reaction pathways or discovering new chemistry.

Graph-based methods is also a common approach. In this approach, molecules are represented as graphs, where atoms are nodes and bonds are edges. Graph neural networks (GNNs) or graph convolutional networks (GCNs) are used to learn the features of molecules and predict reaction outcomes. These models are capable of capturing local and global chemical properties, making them well-suited for reaction prediction tasks. Some limitations with graph-based methods is the complexity. Graph-based methods often require more computational resources compared to template-based or sequence-based methods due to the complexity of graph manipulation and processing. Additionally, accu-

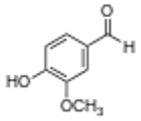
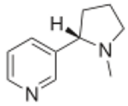
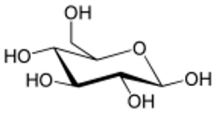
Structure	SMILES formula
$Cu^{2+}SO_4^{2-}$ Copper Sulfate	`[Cu+2].[O-]S(=O)(=O)[O-]`
 Vanillin	`O=Cc1ccc(O)c(OC)c1`
 Nicotine	`CCc(c1)ccc2[n+]1ccc3c2[nH]c4c3cccc4`
 Glucose	`OC[C@@H](O1)[C@@H](O)[C@H](O)[C@@H](O)[C@H](O)1`

Figure 1. Several examples of chemical structures with their corresponding SMILES formulas. Note that SMILES encodes information about elements, structure (rings), connectivity, and even stereochemistry. [C@@H] and [C@H] indicate the orientation of the tetrahedral carbons (either clockwise or counter-clockwise.)

rately representing and predicting stereochemistry can be challenging with graph-based methods, as it is not always straightforward to incorporate 3D information into graph structures.

1.2. Motivation

Reliably predicting chemical reactions can reduce the time and resources required for experimentation, making it a crucial tool for accelerating the discovery of new molecules and materials. Achieving accurate chemical reaction prediction would have wide-ranging consequences across disciplines. The following are some examples of where accurate chemical reaction prediction may be useful.

In drug discovery, the ability to predict chemical reactions can help identify new drug candidates and optimize existing drugs. By predicting the reaction pathways of molecules, scientists can design drugs with specific properties, such as improved efficacy or reduced toxicity. This approach can significantly reduce the time and cost associated with drug development, leading to more efficient and cost-effective treatments [2].

In materials science, chemical reaction prediction can help identify new materials with specific properties, such as strength, conductivity, or catalytic activity. By predicting the reaction pathways of different materials, scientists can design new materials with tailored properties [1].

In environmental studies, chemical reaction prediction can be helpful for predicting the toxicity of chemicals [10] or developing organic semi-conductors for solar cells [6]. By predicting the reactions of pollutants with different environmental components, scientists can identify potential hazards and design mitigation strategies to reduce the impact on the environment.

1.3. Dataset

We used a dataset that was prepared by the authors of the previously mentioned paper [8], a combination of chemical reaction strings from Lowe's dataset and the UPSTO dataset The dataset includes many single-product reactions in the SMILES format, which are text strings representing chemical reactions. SMILES (Simplified Molecular Input Line Entry System) is a notation for representing chemical compounds and reactions. SMILES is a compact format for chemical information composed of a series of characters that describe the connectivity and composition of the atoms in a molecule. Our dataset can be found here. Dataset

In the SMILES format, atoms are represented by their atomic symbols. Hydrogen atoms are usually omitted, as their presence can be inferred from the valence of the other atoms. Aromatic compounds are represented by enclos-

ing the aromatic ring with parentheses and using lowercase letters to indicate the aromatic atoms. Some examples of structures and their corresponding SMILES formulas can be found in Figure 1.

In addition to representing individual compounds, SMILES can also be used to describe chemical reactions. Typically, reactants and products are separated by an arrow (->), with the SMILES strings for the reactants on the left-hand side and the products on the right-hand side. In our dataset, reactants and products have already been separated and prepared for tokenization. The SMILES format has many advantages, including its simplicity, compactness, and machine-readability. Because of these qualities, there are large datasets readily available for use with machine learning. The dataset we used was derived from US patents grants and applications from 1976 to 2016, comprising about 1.8 million reactions in the SMILES format.

In our project, we used two different datasets from [7]: USPTO and OCR. These two datasets are available on Kaggle. The OCR dataset is a large dataset consisting of over 900,000 single-product chemical reactions. To ensure a practical training time, we sampled 20% of this dataset for use in training our models. The USPTO (United States Patent and Trademark Office) dataset is a subset of the OCR dataset published by Jin *et al*. [3], and consists of atom-mapped reactions with no stereochemical information. We also tested trained models on the UPSTO test set to evaluate model performance on a smaller, less noisy dataset.

2. Approach

2.1. Atom-wise tokenization

We first investigated an atom-wise tokenization scheme. With this scheme, individual atoms are considered unique tokens. This approach is simple, as tokens are extracted by splitting on whitespace in the datasets, which are already tokenized in this manner. Any arbitrary SMILES formula can be tokenized by using the regular expression provided in [7]. We found that there were 252 unique tokens across all reactant and product strings.

2.2. Atom-in-SMILES tokenization

We also investigated Atom-in-SMILES (AIS) tokenization, which was proposed by Ucak *et al*. [11]. AIS tokenization converts each atom in a compound to a chemical "word" that includes additional information about an atom's local environment. This approach attempts to resolve certain ambiguities present in atom-wise tokenization, such as the abundance of repeated tokens that are not identical with regard to their local environments. For example, a carbon atom belonging to a carboxylic acid functional group is not identical to a carbon residing within a carbon chain, yet both are tokenized the same way in the atom-wise tokenization scheme. In AIS, the two carbons are uniquely identified by their neighboring atoms. Each atom is represented as a token with three elements: central atom, ring information, and neighbor atoms. For example, an atom may be tokenized like `[C;!R;COO]`, which indicates that it is a carbon atom (`C`), it is not found in a ring (`!R`), and it is adjacent to one carbon and two oxygen atoms (`COO`). These three elements together provide more specific information about a particular atom's local environment.

2.3. Functional group information

Lastly, we attempted to supplement our model with information about the functional groups present in the reactant and product of each reaction. The RDKit and IFG python API's were used in conjunction to estimate the functional groups present in SMILES strings. These API's were used to create a supplementary dataset, about the functional groups present on the reactant and product sides. To simplify the data pipeline through our model, we allocated a single functional group vector to the reactant side, which represented the counts of 119 functional groups in all of the reactants. The same was done for the product side. The data now had two components: 1) Reactant-product smiles strings and 2) Reactant product functional group vectors.

2.4. Transformer-based seq2seq

We defined a sequence-to-sequence architecture using a transformer model. We expected the transformer to perform better than the LSTM-based encoder-decoder from [7] due to the self-attention mechanism of the transformer. This mechanism allows transformers to learn dependencies across entire SMILES strings, which is especially important as reactions can modify any part of a string. The architecture consists of three major components:

1. PositionalEncoding: PositionalEncoding is used to add positional encoding to the token embeddings. This is used to introduce order to the sequence since transformers don't inherently have any sequence order. This is generated using sine and cosine functions applied to the position of each token in the sequence. For a position p in the sequence and a dimension i, the positional encoding is defined as follows:

$$PE(p, 2i) = sin\left(\frac{p}{10000^{\frac{2i}{d}}}\right),$$

$$PE(p, 2i+1) = cos\left(\frac{p}{10000^{\frac{2i}{d}}}\right),$$

where d is the embedding dimension.

2. TokenEmbedding: TokenEmbedding converts our input indices into corresponding token embeddings. It employs an embedding layer and scales the output by the

square root of the embedding size. Let T be the token embedding matrix; then, the token embedding for a token t can be represented as:

$$E_t = T_t$$

Where T_t is the row in the matrix T corresponding to the token t.

3. Seq2Seq Transformer: The seq2seq transformer class defintes a seq2seq model using a transformer. The class contains sub-components, and the model can be mathematically represented as follows.

Given a source sequence X and the target sequence Y, the input embeddings are calculated as:

$$E_X = PE(X) + T_x, E_Y = PE(Y) + T_Y$$

where $PE(\cdot)$ is the positional encoding function, T_X and T_Y are the token embeddings for the source and target tokens, respectively.

The transformer processes the input embeddings E_X and E_Y through the encoder and decoder layers. The layers of the encoder and decoder consist of multi-head self-attention mechanisms, positional feed forward networks, and layer normalization. In the multi-head self-attention mechanism, the attention function is calculated as follows:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)$$

where Q, K, and V are the query, key, and value matrices, respectively, and d_k is the dimension of the key vectors. The multi-head attention is performed by concatenating the results of the attention function applied on different linear projections of the input embeddings:

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_h)W^O$$

where $head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$ and W^O is the output projection matrix.

The output of the transformer is then passed through the generator linear layer to obtain the predicted token probabilities:

$$P_Y = softmax(G(O))$$

where O is the output of the Transformer, G is the generator linear layer, and P_Y represents the predicted probabilities of the target tokens.

2.5. Seq2Seq Transformer trained with Functional Groups

We attempted to augment the aforementioned transformer architecture into a multi-input multi-output(MIMO) Transformer, which could use information about functional groups present in the reactants and products to learn better relationships between the reactant strings and product strings. The architecture remained the same, but included a separate input channels for the functional group vectors and reactant strings. The reactant strings were passed through the token and positional embeddings layers, and then into the transformers encoder. We defined a linear layer, in which the functional group vectors were lengthened to the transformers hidden dimension. These vectors were then combined with the encoder outputs, to be used as input to the transformers decoder, which would predict both the functional groups in the output and the product strings. Unfortunately, this architecture failed to provide a good result, and loss did not converge on the validation set. Potential issues will be outlined in the Challenges section. A link to the code can be found here: Google Drive

2.6. Training and Hyperparameter Tuning

We computed our loss by utilizing cross entropy loss, which is a commonly used loss function for multi-class classification problems. Given the probabilities P_Y and the true target indices Y_true, the cross entropy loss function is defined as:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{i,j}\log(p_{i,j})$$

where N is the number of samples, C is the number of classes, $y_{i,j}$ is the true label for class j in sample i, and $p_{i,j}$ is the predicted probability for class j in sample i.

Additionally, we used Adam *et al.* [5] as our optimizer for updating the model weights. Adam is an adaptive learning rate optimization algorithm that has worked well on a wide range of deep learning tasks. The Adam optimizer updates the model parameters θ based on the gradients of the loss function with respect to the parameters, and estimates of first and second moments of the gradients.

Though we performed no systematic hyperparameter tuning, we chose reasonable values for hyperparameters that were informed by research from Schwaller *et al.* [7, 9]. We adjusted hyperparameters to achieve reasonable compute requirements and training times. Hyperparameters including embedding size, number of attention heads, feed-forward size, batch size, # of encoder/decoder layers, dropout, and training epochs can be found in Table 1.

We also employed, we used the ReduceLROnPlateau learning rate scheduler which, starting from an initial learning rate of 0.001, reduced the learning rate by a factor of 2 whenever two epochs passed without a decrease in validation loss. This adaptive learning rate scheduler helps in converging faster and achieving better performance when we fine tune the model during training.

Hyperparameter	Value
Embedding size	200
Number of heads	5
Feed-forward size	1024
Batch size	128
# of encoder layers	3
# of decoder layers	3
Dropout	0.1
Epochs	50

Table 1. Chosen hyperparameters for Transformer model.

Our implementation is available here: Google Drive

2.7. Challenges

One of the problems we encountered in this project was the size of the dataset. We ended up training only on 20% dataset because of computing limitations present. Training the entire dataset would require much more memory and computation power than what was accessible to us. This was one of the problems we anticipated we tried our best to access additional computing power by utilizing google colab.

The Functional Group MIMO Transformer over fit the training set and did not converge(plateaued at 0.56 average loss) on the training set. One issue was defining a data pipeline for the supplementary functional group data through the transformer, to be combined with the input sequence before predicting output sequences. In our implementation, we broadcasted the functional group embedding's (created from a linear layer) to the size of the encoder output, and added the two tensors together. This was passed in to the decoder, but did not effectively combine the pieces of information. Another option was using tensor concatenation, and changing the decoder input size to account for the reshaped encoder output, but this was difficult to implement into our architecture.

3. Experiments and Results

Following tokenization, we obtained vocabularies of different sizes. For the standard atom-wise tokenization scheme, we found 252 unique tokens. For the Atom-in-SMILES (AIS) tokenization scheme, we found 3,008 unique tokens. The larger vocabulary size from the AIS scheme is because atoms are converted to tokens that include contextual information about neighboring atoms and structures, resulting in a far greater number of unique tokens.

We evaluated the Transformer model on several tokenization and input encoding schemes: atom-wise tokenization, AIS tokenization, and functional group encoding. To evaluate the prediction quality of our models, we performed a greedy search procedure to generate translations from reactant strings to product strings. To qualify as a correct prediction, predicted product strings needed to be exact matches to target strings.

Tokenization/encoding	UPSTO	OCR
Atom-wise	0.761	0.538
AIS	0.736	0.495

Table 2. Prediction accuracy on test sets. Correct product SMILES predictions are exact matches to the target SMILES formulas.

Our results show a maximum accuracy of 76.1% achieved using atom-wise tokenization on the UPSTO test set. Performance was worse on the OCR test set, with 53.8% accuracy. When using AIS tokenization, model performance was degraded, decreasing accuracy to 73.6%. AIS performance on the OCR test set was 49.5%. We observe similar loss curves (Figure 3) for both tokenization schemes and almost identical validation losses after 50 epochs. Training loss, however, is lower for the AIS model. It is likely that the AIS model has a greater degree of overfitting to the training data, resulting in a degraded model performance compared to the atom-wise model. We also noted that around 2.7% of predictions were invalid SMILES strings.

4. Discussion and Future work

AIS tokenization appears to be capable of retaining important information about chemical structures. We achieve a similar, though somewhat degraded, performance with AIS tokenization compared against atom-wise tokenization. Importantly, this demonstrates the ability of transformers to learn reactant-product mappings even with a much larger vocabulary consisting of thousands of unique atom tokens.

Investigating novel tokenization schemes for SMILES strings could help address some of the limitations of the SMILES format and improve the accuracy of deep learning models for chemical reaction prediction. However, this area of research presents several challenges, including the need for large, high-quality datasets and the difficulty of representing the complex structure and behavior of chemical reactions. Further research in this area is needed to explore the potential benefits and challenges of using novel tokenization schemes for chemical reaction prediction. Another approach, learning novel representations of molecules with neural networks, has been explored in previous work [12]. Novel representations of molecules may hold more promise for improving chemical reaction predictions, especially those representations that can capture relevant information about chemical structure, composition, and properties.

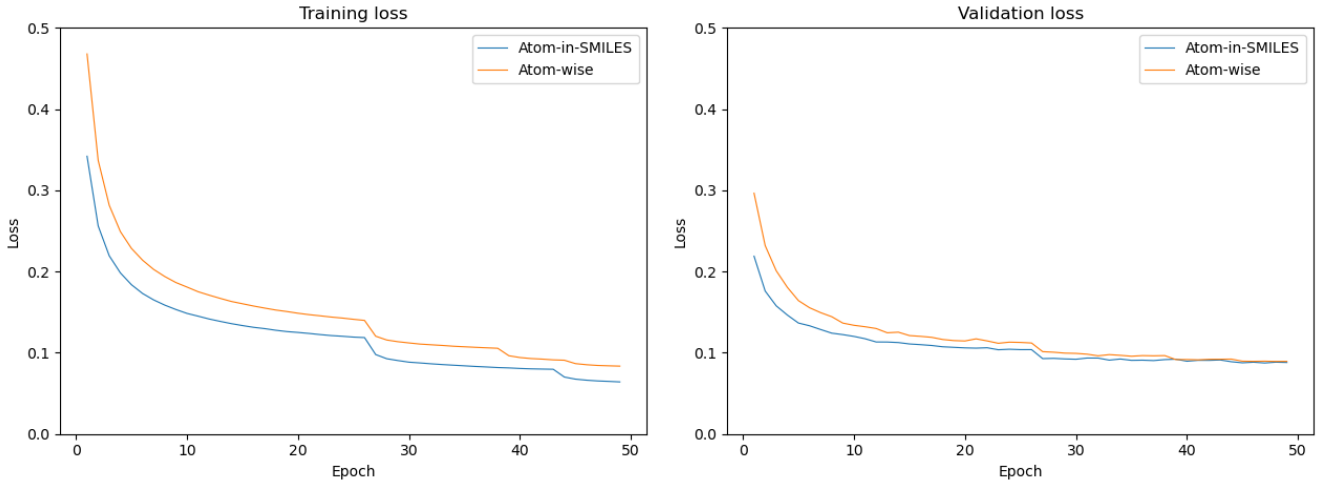


Figure 2. Training loss and validation loss curves for Atom-in-SMILES and atom-wise tokenization schemes.

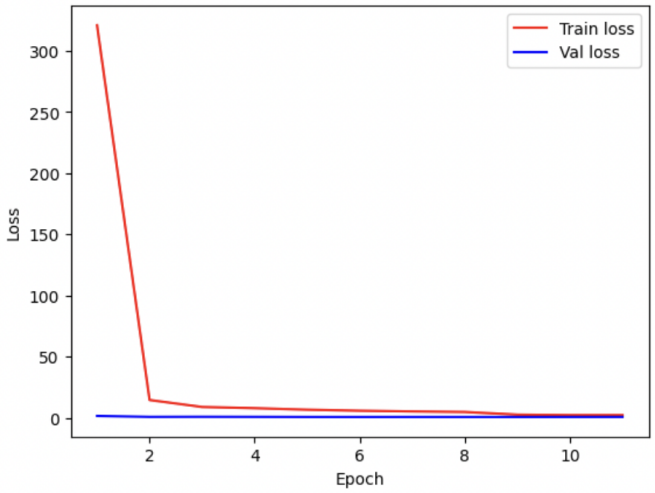


Figure 3. Training loss and validation loss curves for functional group transformer, which was unable to accurately capture the data.

References

[1] Keith T. Butler, Daniel W. Davies, Hugh Cartwright, Olexandr Isayev, and Aron Walsh. Machine learning for molecular and materials science. *Nature*, 559(7715):547–555, July 2018. Number: 7715 Publisher: Nature Publishing Group. 2

[2] José Jiménez-Luna, Francesca Grisoni, and Gisbert Schneider. Drug discovery with explainable artificial intelligence. *Nature Machine Intelligence*, 2(10):573–584, Oct. 2020. Number: 10 Publisher: Nature Publishing Group. 2

[3] Wengong Jin, Connor W. Coley, Regina Barzilay, and Tommi Jaakkola. Predicting Organic Reaction Outcomes with Weisfeiler-Lehman Network, Dec. 2017. arXiv:1709.04555 [cs, stat]. 3

[4] John A. Keith, Valentin Vassilev-Galindo, Bingqing Cheng, Stefan Chmiela, Michael Gastegger, Klaus-Robert Müller, and Alexandre Tkatchenko. Combining Machine Learning and Computational Chemistry for Predictive Insights Into Chemical Systems. *Chemical Reviews*, 121(16):9816–9872, Aug. 2021. Publisher: American Chemical Society. 1

[5] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, Jan. 2017. arXiv:1412.6980 [cs]. 4

[6] Asif Mahmood and Jin-Liang Wang. Machine learning for high performance organic solar cells: current scenario and future prospects. *Energy & Environmental Science*, 14(1):90–105, 2021. Publisher: Royal Society of Chemistry. 2

[7] Philippe Schwaller, Theophile Gaudin, David Lanyi, Costas Bekas, and Teodoro Laino. "Found in Translation": Predicting Outcomes of Complex Organic Chemistry Reactions using Neural Sequence-to-Sequence Models, Nov. 2017. arXiv:1711.04810 [cs, stat]. 1, 3, 4

[8] Philippe Schwaller and Teodoro Laino. Data-Driven Learning Systems for Chemical Reaction Prediction: An Analysis of Recent Approaches. In *Machine Learning in Chemistry: Data-Driven Algorithms, Learning Systems, and Predictions*, volume 1326 of *ACS Symposium Series*, pages 61–79. American Chemical Society, Jan. 2019. Section: 4. 1, 2

[9] Philippe Schwaller, Teodoro Laino, Théophile Gaudin, Peter Bolgar, Christopher A. Hunter, Costas Bekas, and Alpha A. Lee. Molecular Transformer: A Model for Uncertainty-Calibrated Chemical Reaction Prediction. *ACS Central Science*, 5(9):1572–1583, Sept. 2019. Publisher: American Chemical Society. 4

[10] Weihao Tang, Jingwen Chen, Zhongyu Wang, Hongbin Xie, and Huixiao Hong. Deep learning for predicting toxicity of chemicals: a mini review. *Journal of Environmental Science and Health, Part C*, 36(4):252–271, Oct. 2018. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/10590501.2018.1537563. 2

[11] Umit Volkan Ucak, Islambek Ashyrmamatov, and Juyong Lee. Atom-in-SMILES tokenization. preprint, Chemistry, Oct. 2022. 3

[12] Hongwei Wang, Weijiang Li, Xiaomeng Jin, Kyunghyun Cho, Heng Ji, Jiawei Han, and Martin Burke. Chemical-Reaction-Aware Molecule Representation Learning. Jan. 2022. 5

Student Name	**Contributed Aspects**	**Details**
Naiji J Gong	Data preparation and implementation	Implemented and trained Transformer and generated tokenization vocabularies.
Terry Ma	Prepared report and worked on code	Worked on training transformer as well as writing report
Siddharth Pillai	Prepared supplementary dataset of functional groups	Worked on training the Functional Groups transformer.

Table 3. Contributions of team members.