CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Developing a one-to-many generation LLM for diverse, accurate and efficient retrosynthesis

Master's thesis in Computer science and engineering

Junyong Li

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2024

Master's thesis 2024

Developing a one-to-many generation LLM for diverse, accurate and efficient retrosynthesis

Junyong Li

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2024

Junyong Li

Supervisor: Richard Johansson, Computer Science and Engineering
Advisor: Annie Westerlund and Alessandro Tibo, AstraZeneca
Examiner: Ola Engkvist, Computer Science and Engineering

Master's Thesis 2024
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2024

Junyong Li
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

One of the most common applications of deep learning for cheminformatics is retrosynthesis, which is a task of predicting reactants given a chemical product. After transformer was invented, it has been widely used for retrosynthesis. Chemformer is a transformer-based model, which was pre-trained using SMILES of chemical molecules first and can be fine-tuned for retrosynthesis. The model achieves state-of-the-art performance on this task. Retrosynthesis task expects multiple predictions of reactants. Chemformer uses beam search or multinomial search to get multiple predictions, which results in a lack of diversity, accuracy and efficiency of the model. In this project, the sphere projection strategy, which is a one-to-many generation strategy, was applied to Chemformer to enable it to generate multiple predictions. The sphere projection achieves one-to-many generation by introducing variations of source embedding of encoder and combining those variations with a single-prediction sampler, such as greedy search and multinomial search (multinomial size = 1). By comparing the modified Chemformer with sphere projection strategy to the baseline Chemformer, it was shown that the strategy can improve diversity, accuracy and efficiency by 197%, 7% and 4% respectively for beam search, and 101%, 2% and 17% respectively for multinomial search.

Keywords: Retrosynthesis, LLM, large-language model, one-to-many generation, machine learning, deep learning, transformer, diversity, accuracy, efficiency.

Acknowledgements

I am sincerely thankful to my company advisors, Annie Westerlund and Alessandro Tibo, who gave me the oppotunity to work on this project that I learn a lot from. During this project, they helped me understand related theory and source code of Chemformer, which are the foundation of doing my project. And, they also gave me a lot of very detailed feedbacks on my thesis. Additionaly, I would like to thank Richard Johansson, my supervisor at Chalmers, for helping me promote the project with Chalmers, decide the direction of the project and give me many suggestions on my thesis. Finally, I want to thank Ola Engkvist, my examiner, who evaluated my thesis and gave me the comments.

Junyong Li, Gothenburg, 2024-06-05

Contents

List of Figures

List of Tables

1

Introduction

In recent years, the application of deep learning technologies has extended to various domains, including cheminformatics. Within cheminformatics, deep learning has been successfully applied to retrosynthesis, a task that takes target molecules and break them into smaller fragments called reactants. One category of deep learning methods employed for retrosynthesis are sequence-based methods that learns the chemical language encoded in the SMILES notation [1], which is a string representation (text) of a molecule. Language models such as transformers [2] are usually applied as generative methods to generate reactants based on target molecules. More recently, the emergence of large language models (LLMs) based on transformer has also seen their adoption in sequence-based retrosynthesis, exemplified by Chemformer [3] among others [4]. Chemformer was pre-trained to reproduce SMILES based on masked inputs prior to being fine-tuned on the retrosynthesis task. This self-supervised pre-training model can significantly improve performance and speed up convergence on downstream tasks. It achieves state-of-the-art results for top-1 accuracy on retrosynthesis after fine-tuning. And it is also proved that Chemformer performs well on multi-step retrosynthesis [5].

In retrosynthesis, it is expected that the model can generate multiple predictions for one target molecule, which are reactant sets that can be used to synthesize the target molecule in alternative ways. Like most of sequence-based methods with transformer, Chemformer generates multiple predictions by applying sampling algorithms like beam search [6] or multinomial search [7] on output of transformer decoder. The problem of using the original Chemformer for retrosynthesis is that predictions of Chemformer lack diversity. The fraction of unique and accurate predictions among all the predictions generated by the original Chemformer is low. For instance, in 50 predictions, the fraction is 2.7% using beam search and 13.7% using multinomial search. This means that most of accurate predicted molecules are repeated. Besides, although the model achieves state-of-the-art results for top-1 accuracy on retrosynthesis, its top-5 and top-10 accuracy is lower than Aug transformer [4] and a few other models, according to the paper of Chemformer [3]. Therefore, there is still room for improvement in the accuracy of Chemformer. Moreover, using these two sampling algorithms to sample multiple predictions is computationally inefficient, which means the algorithm usually takes a long time to generate multiple predictions. This is because of the high time complexity of the algorithms that is explained in Section 2.4. This problem poses challenges for practical deployment of the Chemformer model.

This thesis addresses problems of Chemformer mentioned above by applying sphere projection strategy to enable it to generate more diverse and accurate predictions efficiently. The method was applied to Chemformer and evaluated in terms of metrics, including fraction of unique and accurate predicted SMILES, round-trip top-k accuracy, and efficiency. The results show that it can significantly improve diversity, accuracy and efficiency of Chemformer by 197%, 7% and 4% respectively for beam search, and 101%, 2% and 17% respectively for multinomial search. Although we focus on retrosynthesis prediction, the method proposed in this project can be transferred to other domains for e.g. text, music and video.

1.1 Thesis outlook

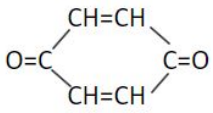
Chapter 2 prepares enough background knowledge for readers to be able to understand content of this thesis. It explains most of technologies used in this project, containing SMILES, retrosynthesis, transformer, sampling methods, Chemformer and evaluation metrics. Then, Chapter 3 states the theory of sphere projection strategy, how it is applied to Chemformer, dataset used for fine-tuning, the ways of inference with the modified model, and how the modified Chemformer was evaluated according to different metrics. Next, Chapter 4 shows results of evaluations based on each metric. Last, discussions and conclusions about the results are shown in Chapter 5.

2

Theory

2.1 SMILES

A SMILES is a sequence of characters used to represent a chemical molecule, which denotes "Simplified Molecular Input Line Entry System". Commonly, a molecule structure is represented as a graph, but it is hard to store a large dataset of graph-based molecules. SMILES is designed to be storage-friendly. Atoms are represented by their atomic symbols in a SMILES. For example, carbon is denoted as 'C' and oxygen is denoted as 'O'. Except atoms, structures in a molecule also need to be represented in a SMILES. Single, double, triple, and aromatic bonds are represented by the symbols "-", "=", "#", and ":". And "." represents "no bond", ie. separating two molecules. Branches are specified by enclosures in parentheses. And the way to represent cyclic structures is to enclose the atoms in the ring within a pair of numbers. Figure 2.1 shows an example of SMILES of quinone that contains atoms, bonds, branches and rings. Double bonds between carbon and oxygen in the left and right of the molecule are denoted as '='. The right oxygen is treated as a branch, so it is enclosed in parentheses. The structure in the middle of the molecule is a ring and it is labeled with a pair of digit '1's in the SMILES. SMILES can be also used to represent more complex structure of molecules like disconnected structures and aromaticity, as shown in the paper of SMILES [1].



O=C1CCC(=O)CC1

Figure 2.1: An example of SMILES: quinone.

2.2 Retrosynthesis

Retrosynthesis is a powerful technique used in chemistry to find a way to synthesize complex molecules. It is essentially a backward approach, starting from the target molecule that the chemist wants to produce and working backward to identify the starting reactants required for the synthesis. In drug discovery, retrosynthesis al-

lows chemists to plan the best route for synthesizing a target molecule, reducing costs and increasing efficiency. By understanding synthetic pathways, chemists can modify drug molecules to improve their efficacy, reduce side effects, and enhance pharmacokinetic properties, ultimately improving healthcare and quality of life. Recently, deep learning is applied to retrosynthesis to automate and accelerate the process, utilizing neural networks to predict the most efficient synthetic routes. It aims to save time and resources, offering quicker and more efficient ways to design complex molecules.

2.2.1 Deep learning in retrosynthesis

There are single-step retrosynthesis prediction and multi-step retrosynthesis search. Single-step retrosynthesis prediction is a kind of algorithms that can predict reactants given a target molecule in one step. While, multi-step retrosynthesis [8] [9] [10] is usually based on single-step retrosynthesis and builds a search tree or a directed acyclic graph to do retrosynthesis in multiple steps to convert target molecule to commercially available building blocks. Multi-step retrosynthesis predictions are of utmost importance because synthesis of a target molecule is usually done in multiple steps in the industrial pharmaceutical domain. However, multi-step predictions require an accurate and efficient single-step model. This project is related to single-step retrosynthesis.

2.2.2 Single-step retrosynthesis

Template-based and template-free methods are two common types of single-step retrosynthesis methods. Template-based methods do not directly generate reactants and they select (predict) appropriate reaction templates which are applied to the product to obtain the reactants, exemplified by [11]. Reaction templates represent local changes around the reaction center, and are therefore more general than reaction SMILES. Template-free methods do not rely on any reaction rules or templates and generate reactants directly. They can either be sequence-based methods, or graph-based methods. The difference between them is how they represent a molecule. Sequence-based methods represent molecules as sequences (SMILES), which will be introduced in next section, while graph-based methods represent molecules as graph and GNNs for retrosynthesis [12] [13] [14] are used to do generation. This project focuses on sequence-based methods, which are SMILES-based methods.

2.2.3 SMILES-based methods

SMILES-based methods formulate retrosynthesis as a sequence-to-sequence translation problem. Input and output molecules are commonly represented as sequences of SMILES tokens. Liu et al. were the first to propose a SMILES-based method for retrosynthesis [15], and the model they use is LSTM [16]. Later, Karpov et al. replaced LSTM with transformer and achieved a better result [17]. Since Karpov's method based on transformer was proposed, most of SMILES-based methods for retrosynthesis have been based on transformer and they were proposed to improve performance on retrosynthesis. For example, Zheng et al. proposed a self-corrected retrosynthesis

predictor(SCROP) to solve invalid output molecules generated by transformer [18], and this approach largely reduced the invalid rate of predictions. Some research performed data augmentation on SMILES representations to improve the performance of SMILES-based methods like [4], and their experiments illustrated that as they applied more intricate data augmentation strategies, the model's generalization ability improved. Moreover, some research tried to also use graph characteristics of molecules by adding a graph encoder [19], and the empirical results indicate a significant improvement in test accuracy for their model compared to the baseline vanilla Transformer model. These methods aim to achieve better accuracy on retrosynthesis in different ways. Another way to improve accuracy of retrosynthesis is to pre-train a large language model like Chemformer used in this project or others [4].

2.3 Transformer

Transformer [2] is a deep learning model architecture that can be applied to many areas like computer vision and natural language processing. And as said above, transformer has been widely used in retrosynthesis. The Chemformer model used in this thesis is based on the transformer architecture. Common transformers have an encoder and a decoder. Both of them consist of embedding layer, attention layer, feed-forward layer and normalization layer. First, the layers are explained, then how the layers form encoder and decoder is shown.

2.3.1 Embedding layer

Usually, texts are input of encoder and decoder of transformer, like SMILES in this project. Texts need to be vectorized before being input to transformer and this is done by embedding layer. Texts need to be tokenized first, which means to convert a character (or one word) into a token that is a number. In this project, a vocabulary for SMILES is used to map characters in SMILES to digits and all single characters are treated as tokens unless specified otherwise. After tokenizing, a sequence of texts is converted to a sequence of tokens. The embedding layer converts each token to a vector according to Equation 2.1. In the equation, W_x and W_t are two learnable matrices in the embedding layer, whose shape is (d, v). d is the number of dimensions of the model and v is the size of vocabulary. x_i is the one-hot vector of the i-th token in the sequence. t_i is the one-hot vector of position of the i-th token in the sequence. The length of the two vectors is v. h_i is the converted vector of the token, whose length is d. W_x and W_t are shared among all the tokens and positions.

$$h_i = W_x x_i + W_t t_i \tag{2.1}$$

2.3.2 Attention layers

Three kinds of attention layers are used in transformers. Multi-head self-attention layer is used in encoder. Masked multi-head self-attention layer and encoder-decoder layer are used in decoder. They are explained below.

2.3.2.1 Self-attention layer

Multi-head self-attention layer is based on self-attention layer, which is used to capture the correlation between the elements within the sequence. When computing output for one vector in the sequence, all vectors in the sequence are used with different weights. Two related vectors in the sequence should have higher weight. The mathematic representation of self-attention is shown in Equation 2.2, 2.3, 2.4. X is the input to the self-attention layer. W_k, W_q and W_v are three learnable matrices in this layer. They are multiplied with X to get K, Q and V, which mean "key", query", "value". K and Q are used to calculate weights among the vectors and they are stored in matrix W. d is the length of vectors and it is used to ignore the length while calculating the weights. The output of the layer is achieved by multiplying W and V.

$$K = W_k X,\ Q = W_q X,\ V = W_v X \tag{2.2}$$

$$W = softmax(K^T Q/\sqrt{d}) \tag{2.3}$$

$$Y = VW \tag{2.4}$$

2.3.2.2 Multi-head self-attention layer

The idea of multi-head self-attention layer is that the matrix with shape (L, d), which is the matrix representation of the sequence input to the self-attention layer, is divided into n sub-matrices with shape $(L, d/n)$. n is the number of heads in the multi-head self-attention layer. The sub-matrices are processed in the same way of self-attention layer by each head. The results from different heads are concatenated back to shape (L, d) after processing. The idea to process the sequence with multi-heads enables the model to extract diverse features and benefits from parallel processing.

2.3.2.3 Masked multi-head self-attention layer

Different from attention layer used in encoder, the multi-head self-attention layer used in decoder is masked. It introduces an attention mask to multi-head self-attention layer. Attention mask is used because decoder is expected to predict next token only based on previous tokens. It is a mask applied to weight matrix W shown in Equation 2.4, so that while computing output of a vector in the input sequence, only vectors before the vector are used. Weights among this vector and vectors after it are masked to zero.

2.3.2.4 Encoder-decoder layer

Decoder not only needs to process the input to decoder, but also needs to utilize information extracted by encoder. The layer that is responsible for this is encoder-decoder attention layer, since the layer uses both input of this layer and source embedding from encoder to compute the output. The computation is shown in Equations 2.6, 2.6, 2.7. The difference from self-attention is the way to compute K,

Q and V matrix in Equation 2.5. K and V are computed with source embedding that is output of encoder. Q is computed with input of this decoder layer. Following computations are the same with self-attention.

$$K = W_k X_e,\ Q = W_q X_d,\ V = W_v X_e \tag{2.5}$$

$$W = softmax(K^T Q/\sqrt{d}) \tag{2.6}$$

$$Y = VW \tag{2.7}$$

2.3.3 Feed-forward layer

Feed-forward layer is the most fundamental component in deep neural network. It has two learnable parameter matrices W and b. By optimizing parameters in these two matrices, feed-forward layer can learn the mapping between input and output. The formula of feed-forward layer is shown in Equation 2.8. X and Y are the input and output. σ is an activation function used to add nonlinearity to the layer, which is usually ReLU [20].

$$Y = \sigma(WX + b) \tag{2.8}$$

2.3.4 Layer normalization

Layer normalization [21] is used in normalization layer of transformer. The input to this layer is a sequence of vectors. The shape of matrix of one sequence is (L, d). L is the sequence length and d is the number of dimensions of the model. Normalization is applied to all the values in the matrix of the sequence according to Equation 2.9, 2.10, 2.11 and 2.12. μ is the mean of all values in the matrix and σ^2 is the variance of the matrix. Normalization is done with Equation 2.11. Each value in the matrix is modified with mean and variance. ϵ is a hyperparameter, but it usually set to a small value and never optimized. Equation 2.12 is used to improve flexibility of the layer by introducing two learnable parameters. Layer normalization helps address internal covariate shift, make training faster and improve performance of models.

$$\mu = \sum_{i=1}^{L}\sum_{j=1}^{d} X_i^j \tag{2.9}$$

$$\sigma^2 = \sum_{i=1}^{L}\sum_{j=1}^{d} (X_i^j - \mu)^2 \tag{2.10}$$

$$\hat{X} = \frac{X - \mu}{\sqrt{\sigma^2 + \epsilon}} \tag{2.11}$$

$$Y = \gamma\hat{X} + \beta \tag{2.12}$$

2.3.5 Encoder

The architecture of encoder is shown in Figure 2.2. Except embedding layer in the bottom and feed-forward layer in the top, the block in the middle is an encoder block, which consists of a multi-head self-attention layer, a feed-forward layer and two normalization layers. The input to a multi-head self-attention layer (or a feed-forward layer) is added to the output of the layer before normalization. This forms a residual architecture [22] that makes training easier. Encoder can contain multiple encoder blocks. The function of encoder is to extract features from the input of transformer.

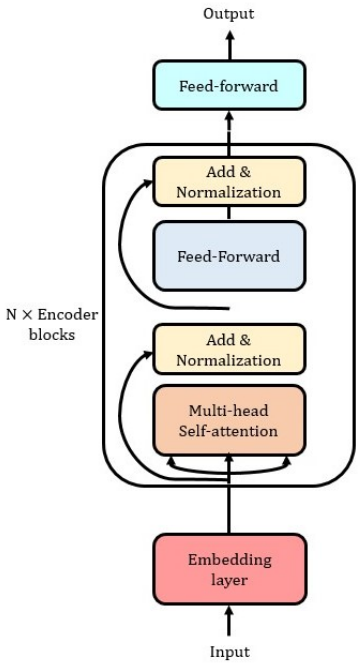


Figure 2.2: Architecture of encoder in a transformer.

2.3.6 Decoder

The architecture of decoder is similar to encoder and it is shown in Figure 2.3. Similar to encoder, decoder blocks also uses residual architecture and a decoder can have multiple decoder blocks. But attention layers used in decoder are different. They are designed for the function of decoder that is to generate sequences auto-regressively based on output of encoder.

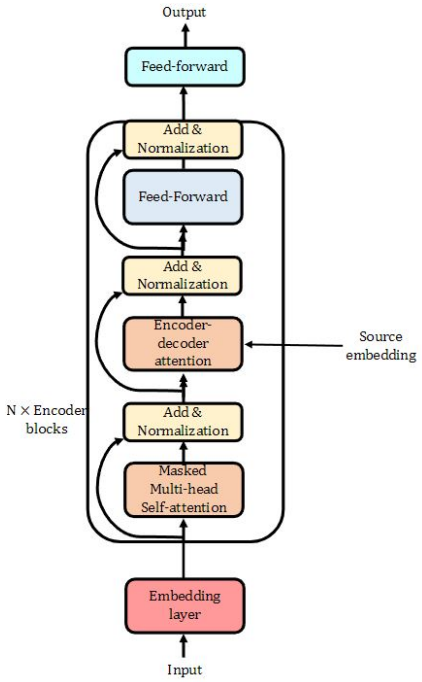


Figure 2.3: Architecture of decoder in a transformer.

2.4 Sampling

Transformers need to be trained first, and then, they can be used to predict results for different tasks. The process of predicting results is called inference. Sampling technologies are used during inference of transformer. During inference, decoder of transformer is responsible for generating a result sequence of tokens auto-regressively, which means to generate one token in one iteration given the source embedding and the previously decoded sequence of tokens. In each iteration, which token to select to be the next token in the sequence is determined by sampling method based on probability distribution of tokens in the vocabulary generated by decoder. Three sampling methods were used for this thesis, greedy search, multinomial search and beam search, as explained below. Multinomial search and beam search can be used by Chemformer to generate multiple predictions for one product. Because the multiple predictions are generated based on the same source embedding of the product as shown in Figure 2.7, the multiple predictions lack diversity. Less diverse predictions reduce the possibility to predict the right reactants to synthesize the product, which limits accuracy of a model. Moreover, multinomial search and beam search are not efficient, which is explained below.

2.4.1 Greedy search

Greedy search is a simple sampling method. In each auto-regressive inference step, the decoder generates probability distribution of tokens. Greedy search selects the token in vocabulary with highest probability as the next generated token. Hence, only one sequence can be generated using greedy search. The time complexity of

greedy search is $O(T \times V)$, where T is the length of generated sequences and V is the size of vocabulary.

2.4.2 Multinomial search

Similar to greedy search, multinomial search is also used to select the next token for the generated sequence based on the probability distribution of tokens in vocabulary. The way to sample one token is shown in Listing 1. The sum of probabilities of tokens in the vocabulary to be the next token is 1. A random number between 0 and 1 is generated. A cumulative probability is summed from the first probability to the last one. In this process, if the cumulative probability is larger than the random number after adding probability of a token, this token is selected as the next token. In this way, the token with higher probability has more chance to be selected. The time complexity of this algorithm is the same with greedy search, which is $O(T \times V)$, where T is the length of generated sequences and V is the size of vocabulary. Multinomial search can be repeated several times to select multiple tokens. The common way of using multinomial search to generate multiple sequences is to sample multiple tokens as started tokens of multiple sequence in the first step of the sequences' generation process. And, in later steps, one next token is sampled for each sequence. Time complexity of the first step of multinomial search for generating N sequences is $O(N \times V)$. The time complexity of the first step is not very high. However, the sampling is based on one probability distribution, so the N sampling processes can not be parallelized. If the sampling is based on N probability distributions and one token is sampled from each of them, the process will be parallelized and faster. That is why using multinomial search to sample multiple sequences is not efficient though its time complexity is not high.

```
# P is probabilities of tokens in vocabulary.
# Vocabulary size is pv.
# The sum of probabilities is 1.
P = [p1, p2, ... , pv]

# Generate a random number between 0 and 1.
random_p = random(0, 1)
sum_p = 0
i = 1
while i <= v and sum_p < random_p:
    sum_p += P[i]

return i-1
```

Listing 1: Pseudo code of multinomial search.

2.4.3 Beam search

Beam search algorithm is another sampling method, which can be used to generate multiple sequences by setting beam size greater than 1. If beam size is set as 1, beam search is equal to greedy search. Figure 2.4 shows how the decoder and beam search algorithm work to generate multiple predictions. Beam search is an iterative process and it depends on a parameter called beam size, B. The larger the beam size is, the better the results are, but this comes at a computational cost. At a time step t, beam search stores the B most probable subsequences (which are just the tokens at t). Then, the decoder predicts probabilities of possible tokens to be the next token for each subsequence. Next, beam search combined the B subsequences with all the new predicted possible tokens and computes probabilities of the B x V new subsequences, where V is the number of possible tokens (usually is number of tokens in vocabulary). Beam search keeps the B most probable ones for next time step t+1. The process ends when t=T (the max sequence length) or when all the subsequences have an EOS token, i.e. a special token that denotes the end of the string. The time complexity of selecting B most probable ones from B x V subsequences is $O(B \times V \times log(B))$, which leads time complexity of beam search to be $O(T \times B \times V \times log(B))$. The action of selecting top B subsequences in beam search causes its inefficiency. If decoder can generate k predictions by processing k sequences in parallel without beam search, the inference process could be much more efficient.

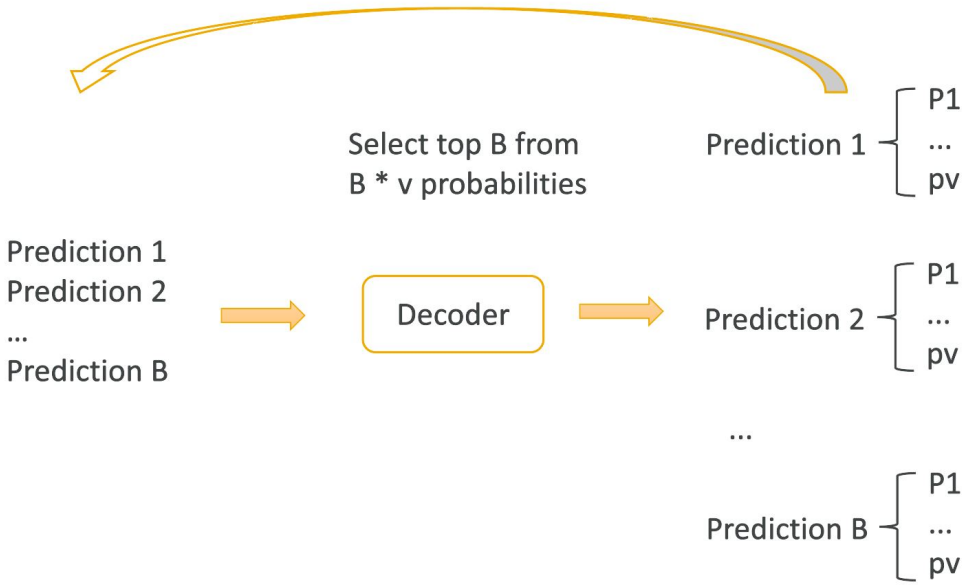


Figure 2.4: Beam search.

2.5 Chemformer

Chemformer [3] is a large language model used for chemical informatics. Its original base model is BART [23] model, i.e. Bidirectional and Auto-Regressive Transformers.

BART is a transformer-based model and has both encoder and decoder explained above. The model was first pre-trained with 100M SMILES, which were randomly selected from roughly 1.5 billion molecules in ZINC-15 dataset [24]. The pre-training process is shown in Figure 2.5. The SMILES is masked before input to the encoder and the decoder is trained to predict the same SMILES given the original SMILES.

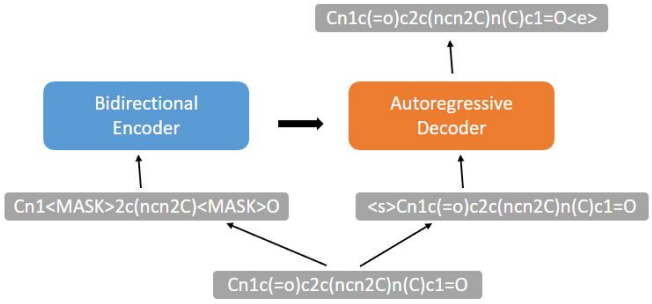


Figure 2.5: Pre-training process of Chemformer.

After pre-training, Chemformer can be fine-tuned to do different tasks, including retrosynthesis, reaction prediction, molecular optimization and property prediction. The way to fine-tune Chemformer's pre-trained model for retrosynthesis is shown in Figure 2.6. Products are the input to encoder, and decoder is trained to predict next token of reactants based on output of encoder.

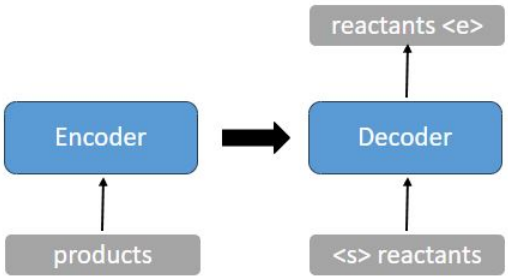


Figure 2.6: Fine-tuning of Chemformer for retrosynthesis.

How Chemformer is used for retrosynthesis during inference is shown in Figure 2.7. The encoder converts SMILES notation of target molecule to source embedding. The decoder works with multinomial search or beam search to generate different predictions. Each prediction is a SMILES, which contains a set of reactants separated by ".". Chemformer can use two sizes of BART models. The smaller model has 45M parameters and the bigger one has 230M parameters. For easier evaluation, the smaller BART model was used in this project. The setting of the small BART model is shown in Table 2.1.

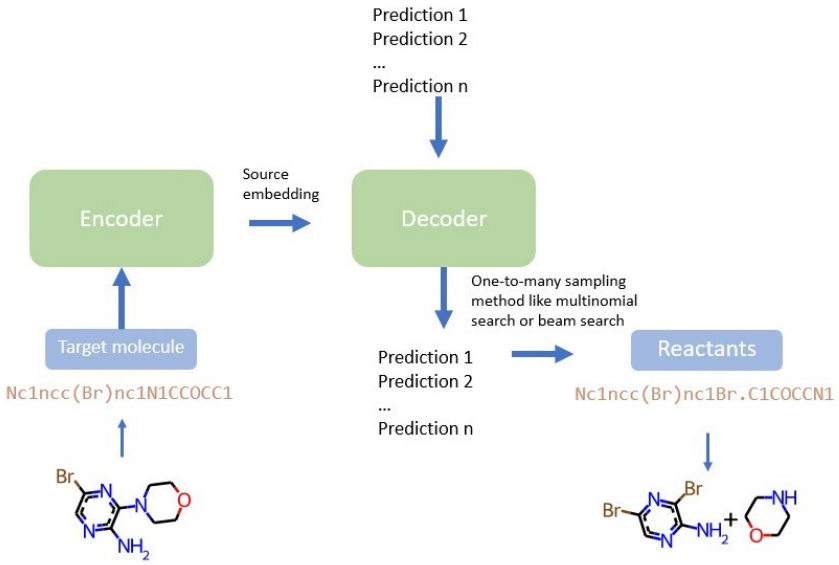


Figure 2.7: Chemformer for retrosynthesis.

Table 2.1: BART model settings.

	Chemformer
Model dimension	512
Feed-forward	2048
Layers	6
Attention heads	8
Parameters	45M

2.6 Evaluation metrics

The models in this project were evaluated in terms of diversity, accuracy and efficiency. Diversity was measured by using fraction of unique and accurate sampled SMILES. Accuracy was measured by round-trip top-k accuracy. And, efficiency was measured by batch-inference-time. Explanation of each metric is shown below.

2.6.1 Diversity

The way to measure diversity used in this project is the fraction of unique and accurate predictions. 'Accurate' means the prediction of reactants can be synthesized to the target product, and 'unique' means a prediction of reactants is not overlap with other predictions. The metric is the fraction of this kind of predictions that meet the two conditions in all predictions. To measure this metric, round-trip evaluation

is used as shown in Figure 2.8. Chemformer generates k predictions of reactants. Then, the predictions of reactants are synthesized back to products. This is done by using a forward synthesis model [5], which is Chemformer fine-tuned for synthesis prediction. Number of accurate predictions is achieved by comparing the synthesized products with the target product, which means that a prediction of reactants is accurate if the corresponding synthesized product's SMILES is the same with the one of the target product. While comparing, two SMILES are canonicalized using RDKit [25] first to avoid the situation that two different SMILES refer to the same molecule. From accurate predictions, the number of unique predictions is counted as n. The fraction of unique and accurate predictions is n/k.

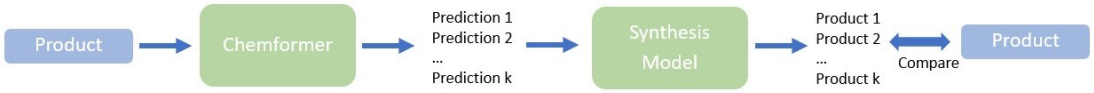


Figure 2.8: Round-trip metric.

2.6.2 Accuracy

A common way to evaluate accuracy of retrosynthesis is using top-k accuracy metric. The idea of top-k accuracy is to check if the target is in the first k predictions. In this project, k is set as the number of predictions of the model, so it is to check if the target is in the predictions. For this thesis, round-trip top-k accuracy was used. So, the way to measure this metric is also shown in Figure 2.8, and the idea is to check if the target product is in the synthesized products. Round-trip top-k accuracy is calculated based on a batch of data. If the number of instances in the batch that the target product is in the synthesized products during evaluation is counted as n, round-trip top-k accuracy is calculated by n / $batch\_size$.

2.6.3 Efficiency

Efficiency was measured during inference of models. Models process a batch of input sequences and make predictions together. The time for a model to generate predictions for a batch of input sequences is batch inference time. Batch inference time was used as the metric of efficiency in this project.

3

Methods

As mentioned in Chapter 1, the problem that this project aims to solve is that original Chemformer, which is BART model with a one-to-many sampling method like beam search or multinomial search, lacks diversity, accuracy and efficiency. It is explained in Section 2.4 that the reason of this problem is the use of one-to-many sampling methods. The idea to solve this problem is to give the model the ability of one-to-many generation so that the model can generate multiple predictions for one product without one-to-many sampling methods. The model still needs sampling methods for auto-regressive decoding, but only a one-to-one sampling method is needed like greedy search or multinomial search (multinomial size = 1). The method used in this project for one-to-many generation is sphere projection strategy. It is a strategy proposed by my adviser, Alessandro Tibo, and it was proved to be effective on improving diversity and efficiency of molecule optimization with the original Transformer model in master thesis of Oscar Almström and Anton Söfting last year. The method is explained in detail first in this chapter. The strategy was applied to BART model of Chemformer and a sphere projection model was achieved. How the sphere projection model was trained and how to use it for inference are also shown in this chapter.

3.1 Sphere projection Strategy

The original Chemformer generates predictions based on only products. The idea of sphere projection strategy is to train a sphere projection model that generates a prediction based on both a product and a unit vector. The hypothesis is that the model will generate diverse predictions if diverse unit vectors are used. The diverse vectors are created with singular value decomposition. More details on this strategy are shown below.

3.1.1 Singular value decomposition

Sphere projection strategy is based on orthogonal vectors. The mathematical way to generate orthogonal vectors is singular value decomposition (SVD) [26], which allows to decompose a matrix into a product of three structured matrix as shown in Equation 3.1. Σ is diagonal matrix. If A is a real matrix, U and V will be unitary matrices, which means that columns (or rows) of them have lengths of 1 and they are orthogonal to one another. In this project, a random real matrix

sampled from a normal distribution is used as A. After SVD, the unitary matrix V is used in sphere projection strategy. The columns of V are called orthogonal vectors or sphere projection vectors.

$$A = U\Sigma V^T, \quad where \quad A \in \mathbb{R}^{m \times n}, \quad U \in \mathbb{R}^{m \times m}, \quad V \in \mathbb{R}^{n \times n} \tag{3.1}$$

3.1.2 One-to-one generation model

The original Chemformer uses BART model that is a one-to-one generation model. The function of the model is shown in Equation 3.2, where x is the input product, Θ is parameters in the model, and y is a possible SMILES that represents reactants. Given a product, BART can compute probabilities of all possible SMILES of reactants that they can be synthesized to the target product. Based on the probabilities, the SMILES with the highest probability is chosen as the generated reactants. Therefore, the model needs to be trained to predict the correct reactants with highest probability. Assume $\hat{x}$ and $\hat{y}$ are a pair of product and corresponding reactants in training data. Parameters in the model, Θ, should be optimized to maximize the probability that $\hat{y}$ is predicted, as shown in Equation 3.3. Since there are always many samples in training dataset, the average of all probabilities for all samples should be maximized, as shown in Equation 3.4, where $\hat{x}^{(i)}$ and $\hat{y}^{(i)}$ are product and reactants of one sample in the dataset, and m is the number of samples in the dataset.

$$P(y|x;\Theta) \tag{3.2}$$

$$argmax P(y = \hat{y}|\hat{x};\Theta) \tag{3.3}$$

$$argmax \frac{1}{m} \sum_{i=1}^{m} P(y = \hat{y}^{(i)}|\hat{x}^{(i)};\Theta) \tag{3.4}$$

3.1.3 Sphere projection model

Unlike one-to-one generation model, sphere projection model receives an additional input z, except the input product x, as shown in Equation 3.5. z is one of sphere projection vectors created by singular value decomposition. Based on diverse sphere projection vectors that are orthogonal to one another, it is expected that the model can generate diverse probability distributions of y, and so different reactants can be predicted.

$$P(y|x,z;\Theta) \tag{3.5}$$

Sphere projection model is a one-to-many generation model, so one-to-many dataset should be used to train the model. A product should have multiple sets of reactants

in the dataset. The dataset used in this project is shown in Section 3.3. The model is trained to predict multiple sets of reactants based on sphere projection vectors. Given a sphere projection vector z_j, the model is expected to predict one of correct sets of reactants, $\hat{y}_j^{(i)}$. Parameters in the model should be optimized to maximize the sum of probabilities that the reactants are predicted by the model, which is shown in Equation 3.6, where p is the expected number of predictions.

$$argmax \frac{1}{m} \sum_{i=1}^{m} \sum_{j=1}^{p} P(y = \hat{y}_j^{(i)} | \hat{x}^{(i)}, z_j; \Theta) \tag{3.6}$$

3.1.4 Implementation

Sphere projection strategy does not change architectures of encoder and decoder. Figure 3.1 shows how to use sphere projection strategy while fine-tuning, while Figure 3.2 shows the process of one-to-many generation with this strategy during inference. However, sphere projection strategy works in the same way. It converts source embedding generated by encoder to multiple source embeddings, so different predictions can be generated by decoder based on different source embeddings. A target molecule is input to encoder of Chemformer and a source embedding with a shape of (L, d) is output by the encoder. L is the length of the source embedding and d is the dimension of the model. In the SVD step, the shape of A is (p, d), where p is number of predictions that is expected. So, according to Equation 3.1, the shape of V is (d, d). The first p columns of V are selected as sphere projection vectors. Therefore, there are p sphere projection vectors and each of them has shape (d). After adding each sphere projection vector to the source embedding, p source embeddings with shape (L, d) are obtained. During fine-tuning, p reactants are input to decoder and the decoder learns to predict next token of the reactants based on corresponding source embeddings. Parameters in encoder and decoder are optimized to adapt to one-to-many generation based on sphere projection vectors in this process. During inference, different predictions are generated by decoder based on different source embeddings with greedy search or multinomial search (multinomial size = 1). Each prediction is a set of reactants that may be able to be used to synthesize the target molecule.

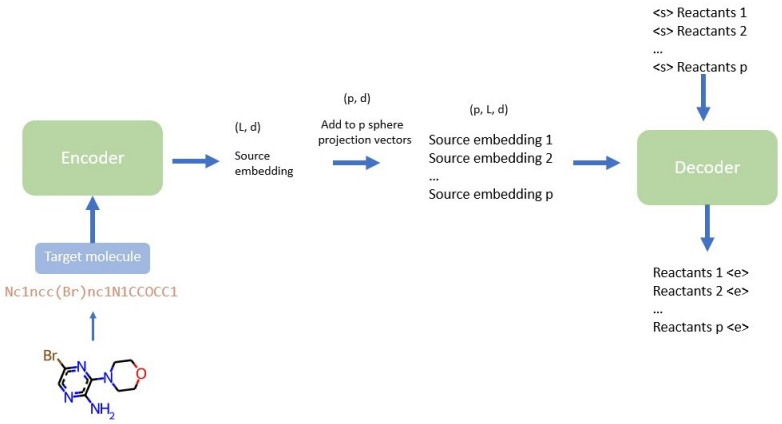


Figure 3.1: Sphere projection strategy while fine-tuning.

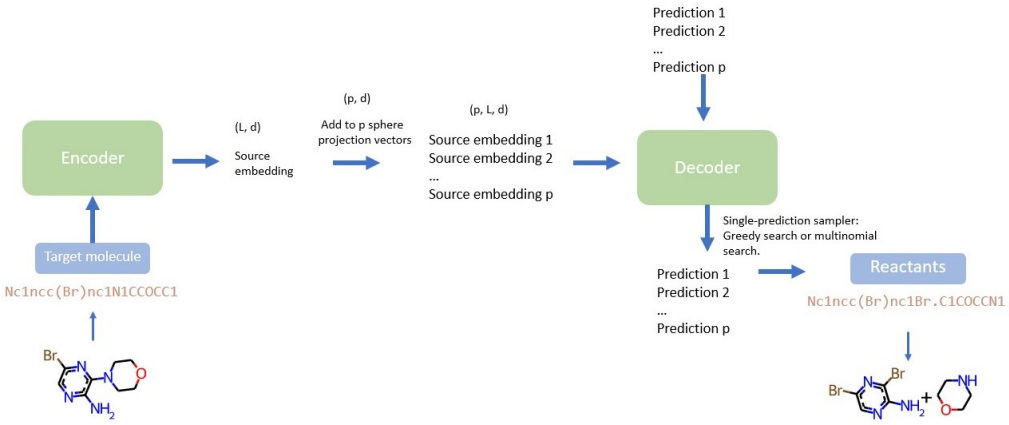


Figure 3.2: Sphere projection strategy while inference.

3.2 Advantages

Previously, one-to-many generation was achieved by using beam search or multinomial sampling by setting beam size or multinomial size greater than 1. Although multiple predictions can be generated given one input, the predictions are generated based on the same source embedding and this results in them being similar to one another. With sphere projection strategy, this does not make sense. Since the sphere projection forms an orthogonal basis (independent vectors), then source embeddings are different from one another. The hypothesis is that this leads to improved diversity and top-k accuracy of predictions.

As explained in Section 2.4, using beam search and multinomial search to sample multiple sequences is inefficient. By applying sphere projection strategy, sampling methods are not responsible for one-to-many generation. Therefore, sizes of beam search and multinomial search are set as 1. In this case, beam search is equal to greedy search. Since greedy search is used for sampling, time complexity of sphere projection strategy is smaller than beam search. As shown in Chapter 2, time

complexity of beam search in one step is $O(B \times V \times log(B))$ and it is used to generate B predictions. In order to generate B predictions with sphere projection strategy, p needs to be set as B. So, the input to decoder is B sequences. For each sequence, V next tokens are predicted. According to greedy search, the most probable token is selected and the time complexity of this action for B sequence is $O(B \times V)$. Therefore, the time complexity is reduced by a factor of $O(log(B))$. In the case of multinomial search, the hypothesis is that the first step of multinomial search for one-to-many generation, which is to sample multiple first tokens of sequences, can not be parallelized, because it works on the same probability distribution generated by decoder based on the only source embedding. With sphere projection strategy, multiple probability distributions are generated and one token is sampled from each distribution with multinomial search in parallel. So, sphere projection strategy can also improve efficiency of the model using multinomial search.

3.3 Dataset

The dataset used to fine-tune Chemformer for retrosynthesis is USPTO-50K [27]. This dataset contains approximately 50k reactions. One reaction is a pair of product and reactants, which can be synthesized to the product. After fine-tuning, Chemformer can predict one set of reactants given a product. With sphere projection strategy, the modified Chemformer generates multiple predictions of reactants. So, this dataset can not be used to fine-tune the one-to-many generation model. To solve this problem, a set-based dataset based on USPTO-50K was created to train the modified Chemformer model. In the new dataset, a product usually has multiple corresponding sets of reactants. These different sets of reactants were created using the reaction templates of the AiZynthFinder template-based model [28]. AiZynthFinder included 179k templates, but not all of them can be used to retrosynthesize a product, so different products get different number of sets of reactants. The sets of reactants for one product were ranked based on log-likelihoods between the product and the sets of reactants computed by using the Chemformer model for retrosynthesis. The product SMILES was input to encoder of Chemformer and the reactants SMILES was input to decoder of the model. The log-likelihood between the product and the reactants is the sum of log-probabilities of each token in the reactants SMILES output by decoder. The value of log-likelihood shows the probability that the reactants can be synthesized to the product, evaluated by Chemformer for retrosynthesis. After ranking, the top 100 sets of reactants were saved in the set-based dataset. There are some products that have less than 100 sets of reactants and they should be removed during fine-tuning if the number of sets of reactants is less than number of predictions. The new dataset is called SET-BASED-USPTO-50K.

3.4 Fine-tuning

Chemformer is based on BART model, which is the original model of Chemformer. Sphere projection strategy was used to modify Chemformer's BART model. The modified model is a sphere projection model, but it has the same encoder and decoder

as the original BART model. Therefore, the sphere projection model can be fine-tuned based on pre-trained parameters of Chemformer. Pre-trained parameters of Chemformer are loaded to sphere projection model's encoder and decoder, and it was fine-tuned with SET-BASED-USPTO-50K dataset. The fine-tuning process is descripted in Section 3.1.4. The loss function used to calculate the loss between output of decoder and ground-truth reactants is cross-entropy. Because multiple sequences are generated, the final loss is the sum of loss of each sequence. Based on the loss, parameters in both encoder and decoder are optimized. Hyper-parameters used to fine-tune the modified model are shown in Table 3.1.

Table 3.1: Fine-tuning hyper-parameters.

Learn rate	0.001
Activation	GeLU
Max sequence length	512
Warm up steps	8000
Dropout	0.1
Batch size	16
Epochs	20

Except these hyper-parameters, sphere projection model has one more hyper-parameter called number-predictions corresponding to how many predictions are expected for one product in sphere projection strategy. In this project, five sphere projection models were fine-tuned by setting number-predictions to be 10, 20, 30, 40 and 50. Before fine-tuning, the corresponding number of sphere projection vectors are generated with SVD. The same vectors are used for all training samples, and after fine-tuning, these sphere projection vectors are saved in checkpoints.

3.5 Inference

The reasonable way to do inference is to use the same sphere projection vectors used for fine-tuning, because parameters in the model are optimized for these vectors. It is not certain that the model can work well with new randomly generated sphere projection vectors. In this way, during inference, the specific checkpoint is used for different number of predictions. For example, if 10 predictions are expected, the checkpoint of the model fine-tuned for 10 predictions should be used. In the checkpoint, there are 10 sphere projection vectors that enable one-to-ten generation.

The drawback of the above way for inference is that the model requires special fine-tuning for different number of predictions. Moreover, when the number of predictions is large, there will be a lot of source embeddings for one input and it needs much GPU memory to do the computation of decoder based on the source embeddings. Therefore, batch size has to be reduced and this leads to much longer fine-tuning time. A solution to this problem is to use newly generated sphere projection vectors instead of ones in the checkpoint during inference. For instance, the

sphere projection model fine-tuned for 10 predictions is used for inference, but sphere projection vectors are newly generated randomly. In this way, the number of generated sequences depends on the number of newly generated sphere projection vectors instead of number of sphere projection vectors in the checkpoint. Because of unfixed sphere projection vectors, the performance of the model needs to be evaluated.

Another way is to let the model with sphere projection strategy work with one-to-many generation sampling methods (like beam search or multinomial search) together. In this way, the model can be fine-tuned once and be used to generate different number of predictions. Sphere projection model generates multiple source embeddings in order to generate multiple prediction. In the above two inference ways, one-to-one sampling methods are used to sample just one prediction for one source embedding. If one-to-many sampling methods are used, multiple predictions can be sampled from one source embedding. Therefore, after fine-tuning, with fixed sphere projection vectors, although fixed number of source embeddings are generated, different numbers of predictions can be achieve by setting beam size or multinomial size. For example, the model is fine-tuned for 10 predictions. So, with the sphere projection vectors in the checkpoint, the model can generate ten source embeddings. If the model works with beam search with beam size (1, 2, 3, 4 and 5), (10, 20, 30, 40 and 50) predictions can be achieved.

Above are the three ways to do inference for the model with sphere projection strategy. The original model used by Chemformer is BART. In order to refer to the three sphere projection models with different inference ways easily, they were given abbreviations as shown below.

- BART: The BART model used by original Chemformer.
- BART-SP: The BART model modified by sphere projection strategy. Fixed sphere projection vectors are used by fine-tuning and inference.
- BART-SP-unfix: The BART model modified by sphere projection strategy and it was fine-tuned for 10 predictions. During inference, new sphere projection vectors generated randomly are used.
- BART-SP-10: The BART model modified by sphere projection strategy and it was fine-tuned for 10 predictions. During inference, the same sphere projection vectors are used and it works with a one-to-many sampler to generate different number of predictions.

3.6 Experiments

The idea of experiments of this project is to compare the four models listed above: BART, BART-SP, BART-SP-unfix and BART-SP-10. From comparisons of them, the effectiveness of sphere projection strategy can be verified and the advantages and disadvantages of various inference methods can be drawn. For a comparison, models used the same sampling method, which can be either beam search or multinomial search. Because the models were tested for multiple predictions (10, 20, 30, 40 and 50), BART and BART-SP-10 do this by setting corresponding beam size and

multinomial size, while BART-SP and BART-SP-unfix use single-prediction sampler by setting beam size and multinomial size as 1. In this case, beam search will be greedy search. Metrics used for comparisons are shown in Section 2.6.

The figures used to show comparison are box and whisker plots (also called box plots), which are used to show distributions of a set of data. In this project, models were evaluated by using test dataset of UPSTO-50K. By setting batch size as 128, there are 39 batches for the test. A value of the metric is computed for each batch. Therefore, there are 39 values of the metric for one specific number of predictions. A box plot was plotted to show the distribution of the 39 values.

Figure 3.3 is an example of box plot, which shows round-trip top-k accuracy of BART model with beam search. A box plot is based on three numbers, which are explained below:

- First Quartile (Q1): The median of the lower half of the dataset, representing the 25th percentile.
- Median (Q2): The middle value of the dataset, representing the 50th percentile.
- Third Quartile (Q3): The median of the upper half of the dataset, representing the 75th percentile.

The box is plotted between Q1 and Q3. Q2 is also plotted as a line in the box. The mean of the data is plotted as a triangle in the figures. Interquartile range (IQR) equals to Q3 minus Q1. The lower whisker extends from Q1 to the minimum value within IQR of Q1, and the upper whisker extends from Q3 to the maximum value within IQR of Q3. Other data points that are not within the range of whisker are plotted as outliers. The distribution shown by a box plot can be used to compare a metric between two models. If Q1 of a model is higher than Q3 of another model, it can be said the performance of the first model is significantly better than the second model. If Q1 of a model is higher than Q2 of another model, it can be still said the performance of the first model is better than the second model. Otherwise, the performance of two models is similar.

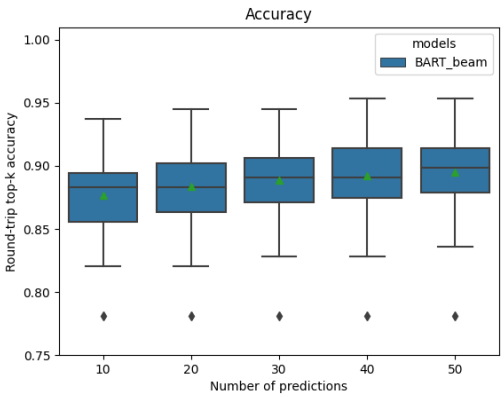


Figure 3.3: Example of box and whisker plot: Round-trip top-k accuracy of BART model with beam search.

4

Results

As explained in Chapter 3, the new BART model modified by sphere projection strategy was fine-tuned for different number of predictions. The result of fine-tuning is shown in this chapter. Results of comparing models are shown in three sections. Section 4.2 is the comparison between BART and BART-SP, which shows that sphere projection strategy performs very well with fixed sphere projection vectors. Section 4.3 is comparison between BART and BART-SP-unfix, which shows the problem of sphere projection strategy with unfixed sphere projection vectors. Section 4.4 is comparison between BART-SP and BART-SP-10, which shows sphere projection strategy works very well cooperating with beam search, but it has problem when it cooperates with multinomial search. According to the comparisons, BART-SP is the best sphere projection model for multinomial search and BART-SP-10 is the best one for beam search. They were compared with BART to get performance improvement achieved by sphere projection strategy and the results are shown in the Section 4.5 of this chapter.

4.1 Fine-tuning

The sphere projection model was fine-tuned by setting number-predictions as 10, 20, 30, 40 and 50. The training loss and validation loss during fine-tuning process of 10 predictions are shown in Figure 4.1. Fine-tuning processes of other numbers of predictions are similar. Training loss continues to decrease in these 20 epochs. Validation loss starts to increase around 10th epoch and this shows that overfitting starts to appear at that time. The best model of each number of predictions is selected based on validation loss.

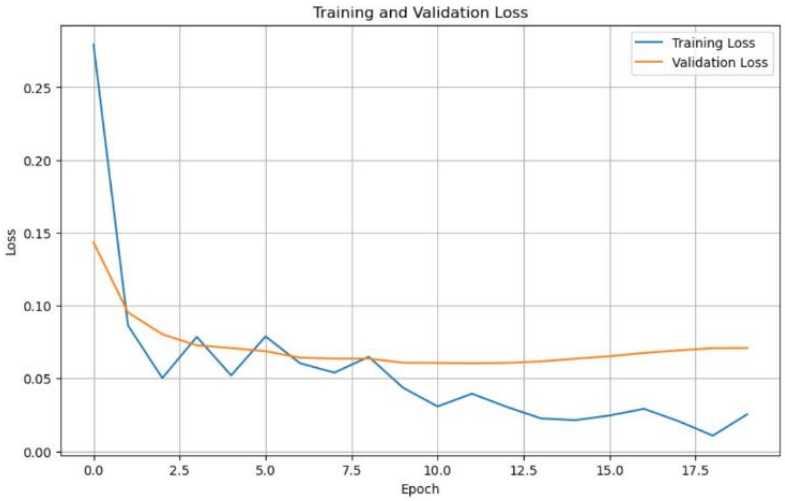


Figure 4.1: Fine-tuning of sphere projection model.

4.2 Comparison of BART and BART-SP

These two models were tested to generate multiple predictions: 10, 20, 30, 40 and 50. BART is a one-to-one generation model, so it needs to use beam search or multinomial search to generate multiple predictions by setting corresponding beam size and multinomial size. BART-SP is a one-to-many generation model applying sphere projection strategy, so it can use single prediction sampler like greedy search, which is beam search (beam size = 1), and multinomial search (multinomial size = 1). For fair comparison, the two models were compared with the same sampling method, and their diversity, accuracy and efficiency were evaluated with fraction of unique and accurate predictions, round-trip top-k accuracy and batch inference time.

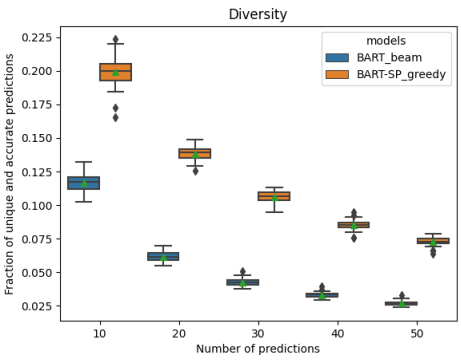


Figure 4.2: Fraction of unique and accurate predictions between BART and BART-SP with beam search.

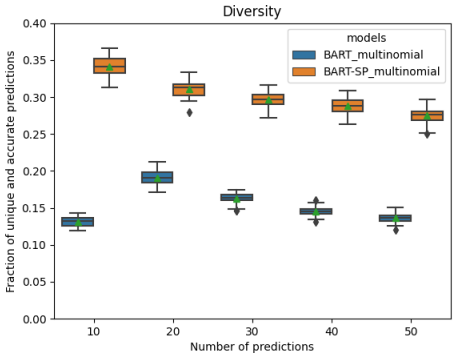


Figure 4.3: Fraction of unique and accurate predictions between BART and BART-SP with multinomial search.

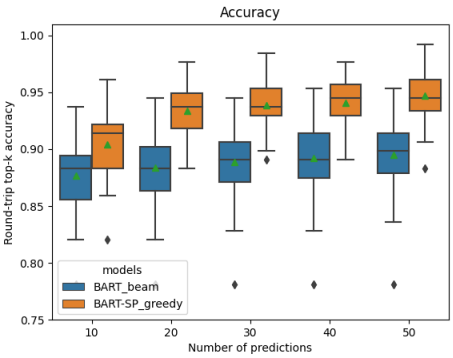


Figure 4.4: Round-trip top-k accuracy between BART and BART-SP with beam search.

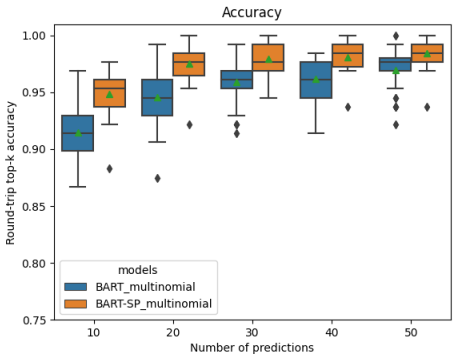


Figure 4.5: Round-trip top-k accuracy between BART and BART-SP with multinomial search.

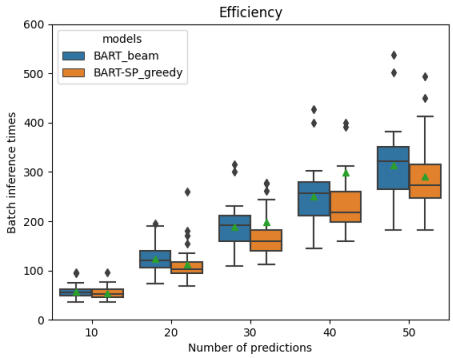


Figure 4.6: Batch inference time between BART and BART-SP with beam search.

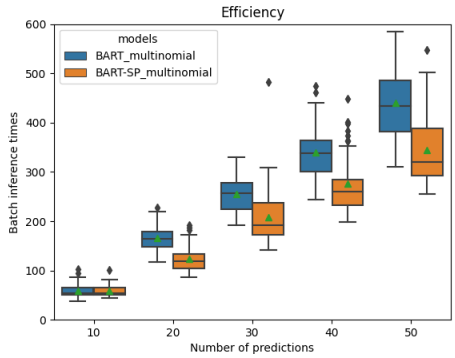


Figure 4.7: Batch inference time between BART and BART-SP with multinomial search.

Figures 4.2 and 4.3 show the results of diversity. Figures 4.4 and 4.5 show the results of accuracy. Figures 4.2 and 4.3 show the results of efficiency. For all the figures, the left ones are results of beam search and right ones are results of multinomial search. According to the results of diversity, it can be seen that not only Q1 of BART-SP is much higher than Q3 of BART model, even lower whisker of BART-SP is much higher than higher whisker of BART model. This proves that sphere projection strategy can improve diversity significantly, no matter which sampling method is used. The improvement on accuracy of sphere projection strategy is also significant, according to results of round-trip top-k accuracy. Although the improvement is not as large as the improvement on diversity, most of Q1 values of BART-SP are higher than Q3 values of BART model except 10 prediction of beam search and 50 predictions of multinomial search. However, for these two cases, Q1 values of BART-SP are still higher than Q2 values of BART model. The improvement of sphere projection strategy on efficiency is not very significant, but most of Q3 values of BART-SP are lower than Q2 values of BART. This shows the effectiveness of sphere projections strategy on improving efficiency. And, the

improvement is not bad, because it is hard to reduce inference time of deep learning models dramatically. Besides, the improvement of sphere projection strategy on efficiency is better for multinomial search than beam search. What is especially good is that even though the efficiency is improved, the accuracy is not affected negatively, rather the opposite when using sphere projection.

4.3 Comparison of BART and BART-SP-unfix

BART-SP performs very well according to results in Section 4.2, but it has a problem. BART-SP has to be fine-tuned for different numbers of predictions, because it needs to use the same sphere projection vectors used by fine-tuning during inference. In this case, for more predictions, it needs more computing resource to fine-tune. One way to solve this problem is to fine-tune the sphere projection model once with a number of predictions and use corresponding numbers of newly generated sphere projection vectors to generate different numbers of predictions during inference. The model using sphere projection strategy in this way is called BART-SP-unfix. In this project, BART-SP-unfix model was fine-tuned for 10 predictions and it was tested to generate 10, 20, 30, 40, and 50 predictions with new sphere projection vectors. Its performance was compared with BART model to check if it works. Unfortunately, it reduces diversity and accuracy in some cases.

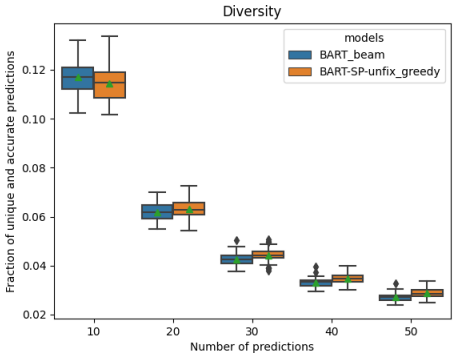


Figure 4.8: Fraction of unique and accurate predictions between BART and BART-SP-unfix with beam search.

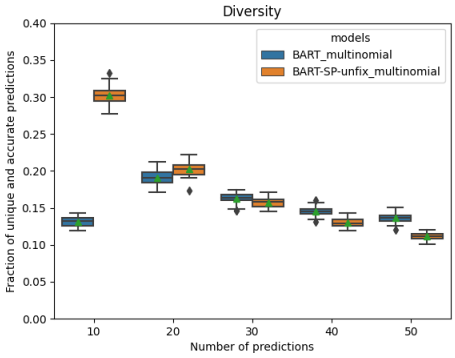


Figure 4.9: Fraction of unique and accurate predictions between BART and BART-SP-unfix with multinomial search.

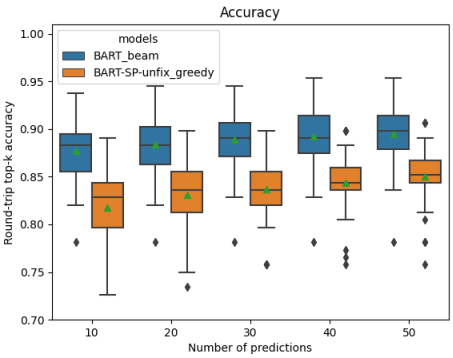


Figure 4.10: Round-trip top-k accuracy between BART and BART-SP-unfix with beam search.

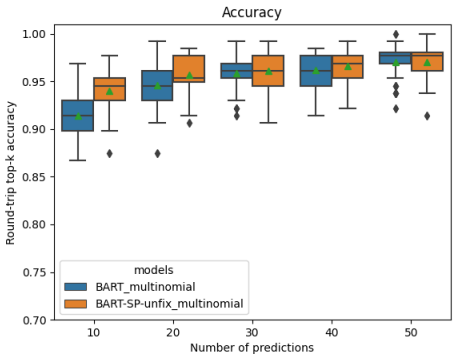


Figure 4.11: Round-trip top-k accuracy between BART and BART-SP-unfix with multinomial search.

Figures 4.8 and 4.10 show the comparison of diversity and accuracy between BART and BART-SP-unfix using beam search. It can be seen that sphere projection strategy using new sphere projection vectors can still achieve good diversity, but it reduces accuracy a lot. The comparison between two models using multinomial search is shown in Figures 4.9 and 4.11. It performs better with multinomial search. BART-SP-unfix can achieve similar accuracy with BART, but its diversity starts to be lower than BART model when number of predictions is larger than 20. In a word, BART-SP-unfix performs differently with beam search and multinomial search. However, with both sampling methods, the performance of BART-SP-unfix is not very good. It can not beat the original BART model, so it is absolutely much worse than BART-SP, too.

4.4 Comparison of BART-SP and BART-SP-10

Another way to solve the problem of BART-SP mentioned in Section 4.3 is to combine sphere projection model with one-to-many generation sampling methods. In this project, BART-SP-10 was tested, which is a model that can generate 10 predictions with sphere projection strategy. It was combined with beam search and multinomial search to generate 10, 20, 30, 40 and 50 predictions by setting appropriate beam size and multinomial size. The results show that the model can achieve better performance than BART-SP when beam search is used, but worse performance when multinomial search is used.

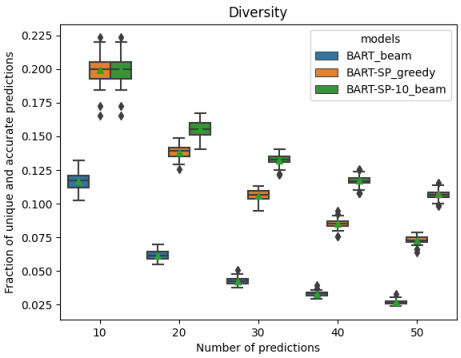


Figure 4.12: Fraction of unique and accurate predictions between BART-SP and BART-SP-10 with beam search.

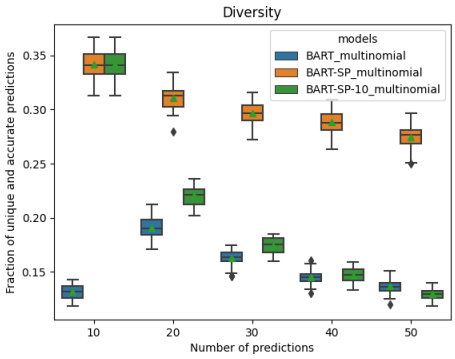


Figure 4.13: Fraction of unique and accurate predictions between BART-SP and BART-SP-10 with multinomial search.

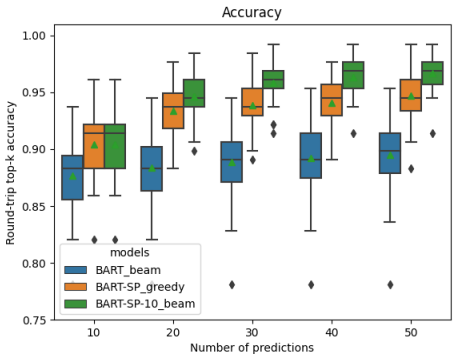


Figure 4.14: Round-trip top-k accuracy between BART-SP and BART-SP-10 with beam search.

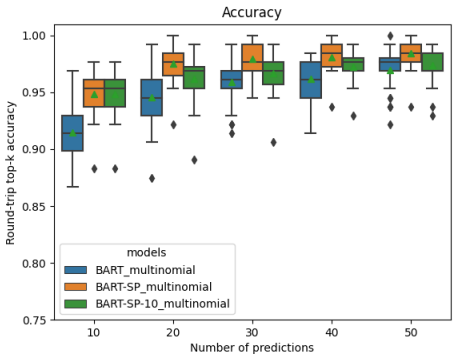


Figure 4.15: Round-trip top-k accuracy between BART-SP and BART-SP-10 with multinomial search.

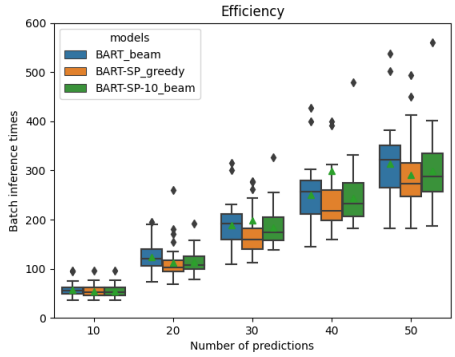


Figure 4.16: Batch inference time between BART-SP and BART-SP-10 with beam search.

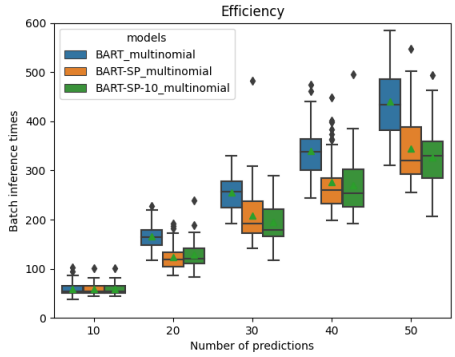


Figure 4.17: Batch inference time between BART-SP and BART-SP-10 with multinomial search.

Figures 4.12, 4.13, 4.14, 4.15, 4.15, 4.16 and 4.17 show the results of comparison between these two models, where orange boxes represent BART-SP and green boxes represent BART-SP-10. Three figures in the left show the comparison of models with beam search, while three figures in the right show the comparison of models with multinomial search. The figures indicate that BART-SP-10 is better than BART-SP when beam search is used. Its diversity and accuracy are much higher than BART-SP. On the contrary, BART-SP is much better on diversity and accuracy than BART-SP-10 when multinomial search is used. However, in this case, BART-SP-10 is still better than BART model, which is represented by blue boxes. Efficiency of the two models is similar. Although efficiency boxes of the two models have different height, Q2 of each model is not higher than Q3 or lower than Q1 of another model. And, their efficiencies are both better than BART model.

4.5 Performance improvement

The above results show sphere projection strategy can improve diversity, accuracy and efficiency substantially. The models, BART-SP and BART-SP-10, that use sphere projection strategy in two ways, are both better than original BART model. The difference between the two models is that BART-SP works better with multinomial search (multinomial size = 1), while BART-SP-10 is better with beam search. To evaluate the performance improvement achieved by sphere projection strategy, the best sphere projection model for each sampling method is compared with BART model, which means to compare BART and BART-SP-10 for beam search and compare BART and BART-SP for multinomial search. Instead of showing the comparisons with box plots like above sections, comparisons of mean values of each metric are shown in this section.

During evaluation, a model was tested for different number of predictions. Metrics were calculated based on each batch. So, for a specific number of predictions and a specific metric, many results of different batches (39 in this project) for this metric were computed. In order to compute the rate of improvement on a model achieved by sphere projection model, mean value of the results of different batches were calculated. And, the rate of improvement is calculated according to Equation 4.1. In the equation, *model_metric* is the metric value of the original BART model and *model_sp_metric* is the metric value of the new sphere projection model.

$$Improvement\_rate = \frac{|model\_sp\_metric - model\_metric|}{model\_metric} \tag{4.1}$$

4.5.1 Improvement of BART-SP-10 compared with BART (beam search)

This sub-section shows the improvement of models with beam search achieved by sphere projection strategy. The average results of the two models of diversity for different numbers of predictions are shown in Table 4.1. Diversity of the sphere projection model, which is BART-SP-10, is much higher than BART model. Although

for both models, the diversity becomes lower as number of predictions increases, diversity of BART decreases faster, which leads to higher improvement of diversity when the number of predictions is larger. The improvement on diversity is striking. When number of predictions is 50, the fraction of unique and accurate predictions can be improved by 296%.

Table 4.2 shows improvement of accuracy achieved by sphere projection strategy. From 10 predictions to 30 predictions, the improvement increases, but it becomes stable after 30 predictions. The improvement of accuracy is from 3% to 8%, which is also great results for round-trip top-k accuracy. Table 4.3 shows the improvement of efficiency. The results of improvement on efficiency do not show a pattern. It does improve the efficiency of the original model, although the improvement is not very big, which is about 4% on average.

Fraction of unique and accurate predictions			
predictions	BART (beam)	BART-SP-10 (beam)	Improvement
10	0.116927083	0.199519231	71%
20	0.061698718	0.155298478	152%
30	0.042654915	0.132071314	210%
40	0.032967748	0.117022236	255%
50	0.027015224	0.106899038	296%

Table 4.1: Improvement of diversity on beam search achieved by sphere projection strategy.

Round-trip top-k accuracy			
predictions	BART (beam)	BART-SP-10 (beam)	Improvement
10	0.876802885	0.904246795	3%
20	0.884014423	0.946113782	7%
30	0.888621795	0.958533654	8%
40	0.892027244	0.963141026	8%
50	0.895032051	0.966947115	8%

Table 4.2: Improvement of accuracy on beam search achieved by sphere projection strategy.

Batch inference time (seconds)			
predictions	BART (beam)	BART-SP-10 (beam)	Improvement
10	56.94988172	54.37116109	5%
20	123.7305162	113.649431	8%
30	187.9783645	184.4848653	2%
40	250.1690653	244.3542343	2%
50	314.1912636	299.2281778	5%

Table 4.3: Improvement of efficiency on beam search achieved by sphere projection strategy.

4.5.2 Improvement of BART-SP compared with BART (multinomial search)

This sub-section shows the improvement of models with multinomial search achieved by sphere projection strategy. Table 4.4 shows the result of diversity. From 20 predictions to 50 predictions, it also shows the same pattern with the result of beam search, which is that improvement on diversity achieved by sphere projection strategy increases as number of predictions increases. The improvement on diversity is also amazing for multinomial search, which is 159% for 10 predictions and 101% for 50 predictions.

Table 4.5 shows the result of improvement on accuracy. The improvement is not very high, although the BART-SP can achieve quite high accuracy. This may be because BART model with multinomial search can already achieve high accuracy. But sphere projection strategy can still improve the accuracy by 2% on average. Table 4.6 shows the efficiency improvement, which is great. Sphere projection strategy can reduce more inference time when the number of predictions is larger, and the improvement on efficiency reaches 22% for 50 predictions.

Fraction of unique and accurate predictions			
predictions	BART (multinomial)	BART-SP (multinomial)	Improvement
10	0.131650641	0.341366186	159%
20	0.191005609	0.310346554	62%
30	0.162807158	0.296254006	82%
40	0.145352564	0.288095954	98%
50	0.136670673	0.274719551	101%

Table 4.4: Improvement of diversity on multinomial search achieved by sphere projection strategy.

Round-trip top-k accuracy			
predictions	BART (multinomial)	BART-SP (multinomial)	Improvement
10	0.914463141	0.948317308	4%
20	0.945913462	0.974959936	3%
30	0.958733974	0.979366987	2%
40	0.961738782	0.981169872	2%
50	0.969951923	0.984174679	1%

Table 4.5: Improvement of accuracy on multinomial search achieved by sphere projection strategy.

Batch inference time (Seconds)			
predictions	BART (multinomial)	BART-SP (multinomial)	Improvement
10	58.48170372	58.46212319	0%
20	166.4940955	123.8274617	26%
30	255.161085	208.0740086	18%
40	339.4160405	276.4899812	19%
50	439.6269857	345.0413615	22%

Table 4.6: Improvement of efficiency on multinomial search achieved by sphere projection strategy.

4.5.3 Average improvement of different numbers of predictions

The average improvements of diversity, accuracy and efficiency are shown in Table 4.7, which are averages of improvements for 10, 20, 30, 40 and 50 predictions. It can not be concluded that improvements by sphere projection strategy on these three metrics are the percentages shown in the table, because the improvement from sphere projection strategy depends on number of predictions. However, the percentages in the table can give readers a more intuitive feeling about the effect of sphere projection strategy.

	Beam	Multinomial
Fraction of unique and accurate predictions	197%	101%
Round-trip top-k accuracy	7%	2%
Batch inference time	4%	17%

Table 4.7: Average improvement by sphere projection strategy on beam search and multinomial search.

5

Conclusion

Chemformer is a BART-based model that can be used to do retrosynthesis. In this project, sphere projection strategy was applied to modify Chemformer for one-to-many generation. The original model of Chemformer is BART, while the model of modifed Chemformer is a sphere projection model. There are three ways for the sphere projection model to carry out inference, which form three models: BART-SP, BART-SP-unfix and BART-SP-10, as explained in Chapter 3. BART-SP-unfix and BART-SP-10 were proposed to solve the problem that BART-SP has to fine-tuned for different numbers of predictions.

Experiments of this project revolved around these four models: BART, BART-SP, BART-SP-unfix and BART-SP-10. By comparing BART and BART-SP, it was shown that sphere projection strategy with fixed sphere projection vectors can improve diversity, accuracy and efficiency of the base model significantly. This experiment proves the effectiveness of this method very well, but the problem with fixed sphere projection vectors needs to be solved.

The experiment on BART-SP-unfix indicates that, compared with BART, it reduces both diversity and accuracy when beam search is used and it reduces diversity when multinomial search is used and number of predictions is big. Therefore, sphere projection strategy with unfixed sphere projection vectors is not a good idea, although it can solve the problem of BART-SP. Based on results of BART-SP-10, this model can achieve better performance than BART-SP when beam search is used, but its results with multinomial search are not good enough. BART-SP-10 not only solves the problem of BART-SP when beam search is used, but also improves the model performance further. However, the problem of BART-SP for multinomial search is still not solved.

According to results of the experiments, BART-SP-10 is the best sphere projection model for beam search, while BART-SP is the best sphere projection model for multinomial search. They were compared with BART to compute rate of improvement on diversity, accuracy and efficiency achieved by sphere projection strategy. The average improvements of the three metrics for beam search are 197%, 7% and 4% respectively, and the ones for multinomial search are 101%, 2% and 17% respectively. The results indicate that sphere projection strategy can improve diversity of predictions generated by the model substantially. Although the original Chemformer achieves state-of-the-art accuracy on retrosynthesis, its accuracy can still be further improved by using sphere projection strategy. And, the improvements are

also very significant, because they are achieved when accuracies of original Chemformer are already very high, which are around 88% with beam search and around 95% with multinomial search. And, the strategy does reduce inference time of the model. Its effect on efficiency of beam search is not big, but it can improve efficiency of multinomial search a lot.

5.1 Future work

As said above, for multinomial search, the sphere projection model needs to be fine-tuned for different numbers of prediction, because the problem of fixed sphere projection vectors is not solved for this sampling method. A modification on sphere projection strategy needs to be made to solve the problem. The dataset used in this project is based on USPTO-50k, which is a small dataset. Models in this project need to be trained using a bigger dataset to see if its performance can be improved further. The thesis proves that sphere projection strategy is a very effective method by comparing modified model with the strategy with original base model. To prove sphere projection strategy is a state-of-the-art method, it needs to be compared with other one-to-many generation methods like [29] and [30] , or other methods that can be used to enhance diversity of retrosynthesis, such as [31] and [32]. The sphere projection strategy is not limited to retrosynthesis prediction, but can be applied to other domains, including text, music and video.

Bibliography

[1] D. Weininger, "Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules," *Journal of chemical information and computer sciences*, vol. 28, no. 1, pp. 31–36, 1988.

[2] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[3] R. Irwin, S. Dimitriadis, J. He, and E. J. Bjerrum, "Chemformer: A pre-trained transformer for computational chemistry," *Machine Learning: Science and Technology*, vol. 3, no. 1, p. 015 022, 2022.

[4] I. V. Tetko, P. Karpov, R. Van Deursen, and G. Godin, "State-of-the-art augmented nlp transformer models for direct and single-step retrosynthesis," *Nature communications*, vol. 11, no. 1, p. 5575, 2020.

[5] A. M. Westerlund, S. M. Koki, S. Kancharla, *et al.*, "Do chemformers dream of organic matter? evaluating a transformer model for multi-step retrosynthesis," 2023.

[6] C. Tillmann and H. Ney, "Word reordering and a dynamic programming beam search algorithm for statistical machine translation," *Computational linguistics*, vol. 29, no. 1, pp. 97–133, 2003.

[7] D. A. Team. "From top-k to beam search: Everything you need to know about llm decoding strategies." Accessed: 2024-05-20. (2024), [Online]. Available: `https://deci.ai/blog/from-top-k-to-beam-search-llm-decoding-strategies/`.

[8] A. Heifets and I. Jurisica, "Construction of new medicines via game proof search," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 26, 2012, pp. 1564–1570.

[9] M. H. Segler, M. Preuss, and M. P. Waller, "Planning chemical syntheses with deep neural networks and symbolic ai," *Nature*, vol. 555, no. 7698, pp. 604–610, 2018.

[10] B. Chen, C. Li, H. Dai, and L. Song, "Retro*: Learning retrosynthetic planning with neural guided a* search," in *International Conference on Machine Learning*, PMLR, 2020, pp. 1608–1616.

[11] C. W. Coley, L. Rogers, W. H. Green, and K. F. Jensen, "Computer-assisted retrosynthesis based on molecular similarity," *ACS central science*, vol. 3, no. 12, pp. 1237–1245, 2017.

[12] C. Shi, M. Xu, H. Guo, M. Zhang, and J. Tang, "A graph to graphs framework for retrosynthesis prediction," in *International conference on machine learning*, PMLR, 2020, pp. 8818–8827.

[13] V. R. Somnath, C. Bunne, C. W. Coley, A. Krause, and R. Barzilay, "Learning graph models for template-free retrosynthesis," *arXiv preprint arXiv:2006.07038*, 2020.

[14] C. Yan, Q. Ding, P. Zhao, *et al.*, "Retroxpert: Decompose retrosynthesis prediction like a chemist," *Advances in Neural Information Processing Systems*, vol. 33, pp. 11 248–11 258, 2020.

[15] B. Liu, B. Ramsundar, P. Kawthekar, *et al.*, "Retrosynthetic reaction prediction using neural sequence-to-sequence models," *ACS central science*, vol. 3, no. 10, pp. 1103–1113, 2017.

[16] G. Van Houdt, C. Mosquera, and G. Nápoles, "A review on the long short-term memory model," *Artificial Intelligence Review*, vol. 53, pp. 5929–5955, 2020.

[17] P. Karpov, G. Godin, and I. V. Tetko, "A transformer model for retrosynthesis," in *International Conference on Artificial Neural Networks*, Springer, 2019, pp. 817–830.

[18] S. Zheng, J. Rao, Z. Zhang, J. Xu, and Y. Yang, "Predicting retrosynthetic reactions using self-corrected transformer neural networks," *Journal of chemical information and modeling*, vol. 60, no. 1, pp. 47–55, 2019.

[19] K. Mao, X. Xiao, T. Xu, Y. Rong, J. Huang, and P. Zhao, "Molecular graph enhanced transformer for retrosynthesis prediction," *Neurocomputing*, vol. 457, pp. 193–202, 2021.

[20] K. Fukushima, "Self-organizing multilayered neural network," *The Transactions of Electronics and communication Engineers D*, vol. 58, no. 9, p. 530, 1975.

[21] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[23] M. Lewis, Y. Liu, N. Goyal, *et al.*, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *arXiv preprint arXiv:1910.13461*, 2019.

[24] T. Sterling and J. J. Irwin, "Zinc 15–ligand discovery for everyone," *Journal of chemical information and modeling*, vol. 55, no. 11, pp. 2324–2337, 2015.

[25] "The rdkit 2024.03.1 documentation." Accessed: 2024-05-21. (2024), [Online]. Available: `https://rdkit.org/docs/source/rdkit.Chem.html`.

[26] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU press, 2013, pp. 69–75.

[27] N. Schneider, N. Stiefl, and G. A. Landrum, "Whats what: The (nearly) definitive guide to reaction role assignment," *Journal of chemical information and modeling*, vol. 56, no. 12, pp. 2336–2346, 2016.

[28] S. Genheden, A. Thakkar, V. Chadimová, J.-L. Reymond, O. Engkvist, and E. Bjerrum, "Aizynthfinder: A fast, robust and flexible open-source software for retrosynthetic planning," *Journal of cheminformatics*, vol. 12, no. 1, p. 70, 2020.

[29] T. Shen, M. Ott, M. Auli, and M. Ranzato, "Mixture models for diverse machine translation: Tricks of the trade," in *International conference on machine learning*, PMLR, 2019, pp. 5719–5728.

[30] M.-A. Lachaux, A. Joulin, and G. Lample, "Target conditioning for one-to-many generation," *arXiv preprint arXiv:2009.09758*, 2020.

[31] A. Toniato, A. C. Vaucher, P. Schwaller, and T. Laino, "Enhancing diversity in language based models for single-step retrosynthesis," *Digital Discovery*, vol. 2, no. 2, pp. 489–501, 2023.

[32] E. Kim, D. Lee, Y. Kwon, M. S. Park, and Y.-S. Choi, "Valid, plausible, and diverse retrosynthesis using tied two-way transformers with latent variables," *Journal of Chemical Information and Modeling*, vol. 61, no. 1, pp. 123–133, 2021.

A

Appendix 1