We thank all the reviewers for the helpful feedback. We have provided responses here for important questions that reviewers may commonly be interested in.

Important questions

Q1 How does DRGAT compare generally to well-tuned, off-the-shelf GNNs?

A1 The performance ordering of the mainstream GNNs is: DRGAT $>$ RGCN $>$ GAT $>$ GCN $>$ ChebNet $>$ GIN.

For center identification, we compare different GNNs in a same code framework, the results are shown in Table.1. Experiments were conducted with known classes because our reproduced results of baselines are more consistent with the reported ones under this setting. All these models use 6 layers of GNN with the hidden dimension of 256 and are trained up to 100 epochs. The results of RGCN are slightly different from G2G (Shi et al., 2020), probably because we removed the weighting hyper-parameter λ of the cross entropy loss and set the threshold for binary classification as 0.5. It should be pointed out that the reproduced RGCN results under our framework are slightly better than the reported ones in terms of top-2, top-3, and top-5 accuracy.

	top-1	top-2	top-3	top-5
GCN	84.5%	94.2%	97.0%	98.9%
GAT	85.3%	94.8%	97.1%	98.7%
ChebNet	74.3%	86.6%	90.6%	93.7%
GIN	71.7%	84.1%	88.3%	92.3%
RGCN	87.4%	96.4%	98.4%	99.4%
DRGAT	91.7%	97.5%	98.6%	99.5%

Table 1: GNN for center identification (class known).

Q2 Ablation study of the DRGAT. Where the performance gain come from?

A2 The performance gain mostly comes from the attention mechanism.

As shown in Table.2, DRGAT obtains similar accuracy as RGCN while the attention mechanism is disabled. Otherwise, DRGAT will significantly outperform RGCN. We also find that directional node embedding and edge features do not bring remarkable performance gains, inspiring us to further simplify the model.

	top-1	top-2	top-3	top-5
RGCN	87.4%	96.4%	98.4%	99.4%
w/o attention	88.0%	96.1%	98.0%	99.2%
w/o directed embedding	91.4%	97.3%	98.7%	99.5%
w/o edge features	91.5%	97.5%	98.8%	99.5%
DRGAT	91.7%	97.5%	98.8%	99.5%

Table 2: Ablation study of DRGAT for center identification (class known).

1 REVIEWER G14H

We would like to thank you for the constructive comments and careful reading. Below we address the concerns/questions mentioned in the review:

Q1 It could be instructive to try out the same model with the same architecture that G2G, GLN, or NeuralSym uses. Or maybe with DRGAT, G2G, GLN, or NeuralSym would also perform better?

A1 Both DRGAT and semi-template bring considerable performance gains. In terms of top-1 accuracy, their contributions seem similar, but in terms of top-k accuracy, DRGAT contributes more. In Table.3, we provide the detailed results.

However, we argue that SemiRetro have only exploited limited potential of semi-template as the extracted templates may not be perfect, as shown in Figure 1. The template extraction algorithm is not our key contribution, and we will improve it in the future.

	top-1	top-3	top-5	top-10
RGCN + G2G	61.0	81.3	86.0	88.7
RGCN + semi-template	63.7	82.3	86.1	88.7
DRGAT + semi-template	66.6	83.8	87.3	89.9

Table 3: Contribution of DRGAT and semi-templates (class known).

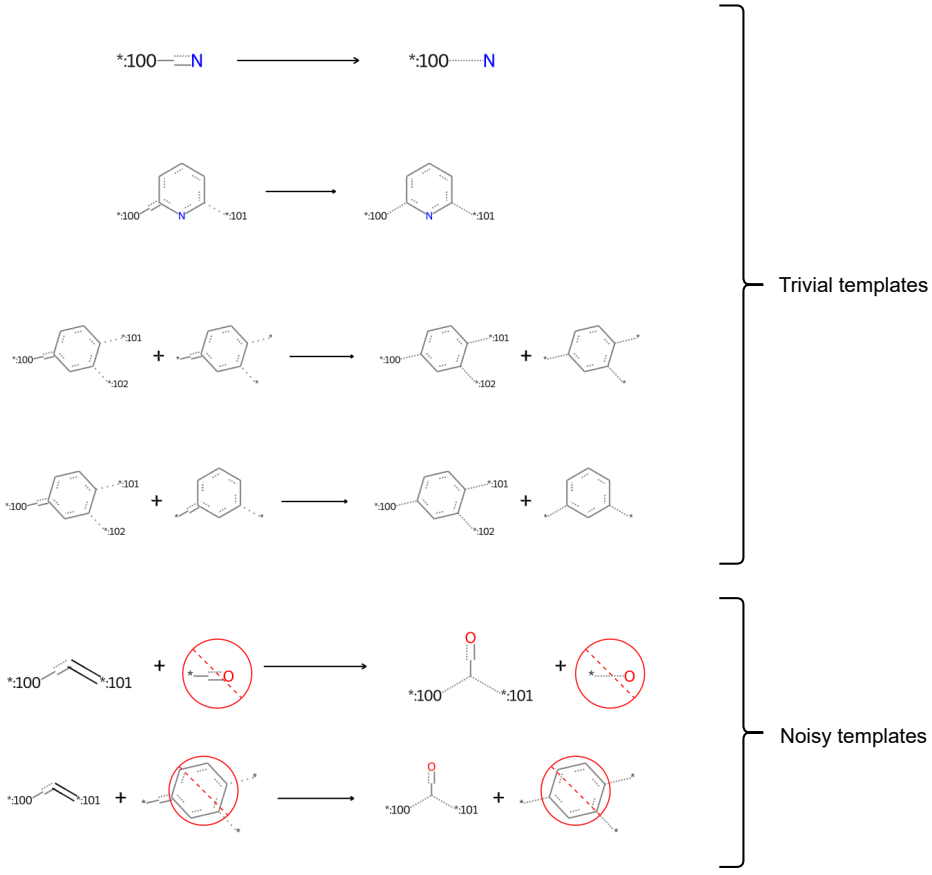

Figure 1: Bad templates. Trivial templates do not contain any meaninful structural changes. Noise templates include some substructural noise. Both the trivial and noisy templates increase the template diversity and confuse the model predictions.

2 REVIEWER XCBM

Thanks for your constructive suggestions and careful reading. Below we address the concerns/questions mentioned in the review:

Q1: Can you compare to other synthon completion methods in Table 3?

A1: We cannot provide synthon completion results of (Sun et al., 2020), becuase this work:

1. does not report the accuracy of synthon completion.
2. has no open-source code.
3. focuses on suggesting a new training strategy, instead of proposing a novel synthon completion model.

As G2G provides open-source code, we can conviently implement the same chemical features and code framework to ensure the fairness of experimental settings. Thus, we regard G2G as an important baseline. Other methods (Somnath et al., 2020; Sun et al., 2020; Yan et al., 2020) either have no open-source code or have quite difficult implementations, and we cannot guarantee the fairness of comparative experiments.

Q2: How do you make use of the reaction class when it is known? Do you just use it to mask out inapplicable templates or do you also provide the information to the reaction center network?

A2: We use the reaction class as additional embedding feature to enhance the prediction. We do not mask out inapplicable templates. The python code is:

```
    if 'reaction' in self.feature:
        reaction_type = synthon.reaction
        reaction_feature = self.type_embedding(reaction_type)
        embedding += reaction_feature
```

where

```
    self.type_embedding = nn.Embedding(num_reaction, output_dim)
```

Q3: In the right hand subplot of Figure 5, the dotted green light appears to end above 1. Is this correct?

A3: Thanks for your careful suggestion. We are sorry for this careless fault, and we have corrected it in the revised version.

Q4: I'm a little confused by equation 5, in particular the second readout term. Are the node embeddings taken from message passing run on (a) the template graph or from (b) the synthon graph?

A4: In equation 5, all the node embeddings comes from (b) the synthon graph by running message passing neural network. All the readout operations output the average pooling results of input feature vectors.

Q5: On p.6 you state: "Once reaction centers, synthons, and corresponding semi-templates are known, we can deduce reactants with almost 100% accuracy." Why "almost" 100%? When does it fail?

A5: We examine the falied samples and argue that the low quality templates lead to failsures. In Figure 2, we show two typical failed cases. These failures can be remedied by improving the template extraction algorithm. Although the template extraction algorithm is not our key contribution, we will improve it in future work.

Q6: I would be interested to have more details about why you think SemiRetro has better "interpretability" than previous approaches? (Previous approaches have also used templates, and the reaction center prediction here still seems to be done in a black box manner?)

A6: Our paper emphasizes the interpretability of synthon completion, notably the deterministic residual attaching algorithm.

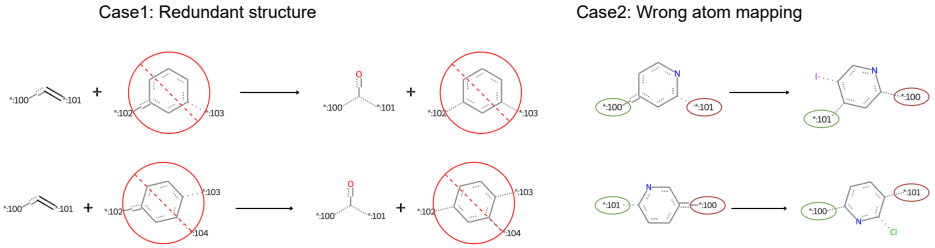

Figure 2: Failed cases. In Case1, there are some redundant trivial structures that confusing the residual attaching algorithm. In Case2, the atom mapping of the semi-template is wrong, resulting in local structure flipping.

The previous work G2G recursively predicts new atoms and new bonds without considering the integrity of functional groups, and RetroXpert applies a language model to generate reactants from synthons ignoring the graph structure. Our proposed SemiRetro considers both the graph structure and the integrity of functional groups to make synthon completion more interpretable.

W1 A large part of the performance of this approach seems to stem from DRGAT's superior reaction center prediction (compared to e.g. the R-GCN used by Shi et al., 2020) – in particular see Table 2. However, it is not clear exactly where this performance comes from: as well as introducing an edge based attention mechanism the authors also use more layers than the R-GCN. It would be helpful to have an ablation study. How does it compare generally to well-tuned, off-the-shelf GNNs...?

Replay to W1 In the section of common questions, we compare the center identification performance of various GNNs in Table. 1. We have also performed ablation experiments on DRGAT, and the results are shown in Table. 2, where all GNNs use the same number of layers and embedding size. We conclude that the performance gain comes from the attention mechanism of DRGAT.

W2 I have concerns about the novelty of the paper. SemiRetro seems quite similar to Somnath et al. (2020)'s GraphRetro (that method also has a two step process that first predicts the reaction center to create synthons and then adds "leaving groups" to create reactants). I would like to have seen a more detailed description of how the work here differs? For instance, is predicting semi-templates distinctly different to predicting leaving groups?

Replay to W2 There are three differences:

1. Semi-templates are more expressive than leaving groups. Semi-templates encode the structure transformation while leaving groups encode the structure, which is part of the transformation. Some common structural changes cannot be represented by residues, whereas semi-templates can, seeing Figure. 3.
2. Predicting semi-templates is different. We use the features of 1-neighborhood supporting atoms to help semi-template prediction, whereas leaving groups do not contain neighborhood information, seeing Figure. 4.
3. Residual attaching is different. GraphRetro classifies the data of the USPTO-50K dataset and designs residual attachment strategies for the data in simple cases without providing a standard algorithm. We argue that this approach is not generalized to more complicated scenarios. To get out of this dilemma, we provide a generic algorithm in the appendix.

W3 SemiRetro's training efficiency and scalability are highlighted as one of the advantages of the method (see e.g. abstract) and the speed improvements listed in Table 6 seem impressive. It would therefore have been nice to see experiments on larger retrosynthesis datasets, e.g. the larger USPTO one used by Dai el al. (2019), to show the practical advantages of this.

Replay to W3 Thanks for your suggestions. We try to conduct experiments on USPTO-FULL using the same neural model on USPTO-50k. The training process takes 20 hours. In our initial attempts, the top-10 accuracy 62.5% is lower than GLN's 63.7%. We believe that the sub-optimum hyper-parameter setting, e.g., the depth and with of the network, and inadequate template extraction algorithm may lead to low accuracy. Due to time constraints, we prefer to improve the model performance on USPTO-FULL in the future.

W4 In the retrosynthesis experiments there are missing comparisons to state-of-the-art (SOTA) methods, e.g.:

Sun, Ruoxi, et al. "Energy-based View of Retrosynthesis." arXiv preprint arXiv:2007.13437 (2020).

Sacha, Mikołaj, et al. "Molecule edit graph attention network: modeling chemical reactions as sequences of graph edits." Journal of Chemical Information and Modeling 61.7 (2021): 3273-3284. Preprint: https://arxiv.org/abs/2006.15426

Replay to W4 In the revised paper, we have explained the reason for ignoring energy-based methods in section 5.1 and compared it with the molecule edit graph attention network in Table 6.

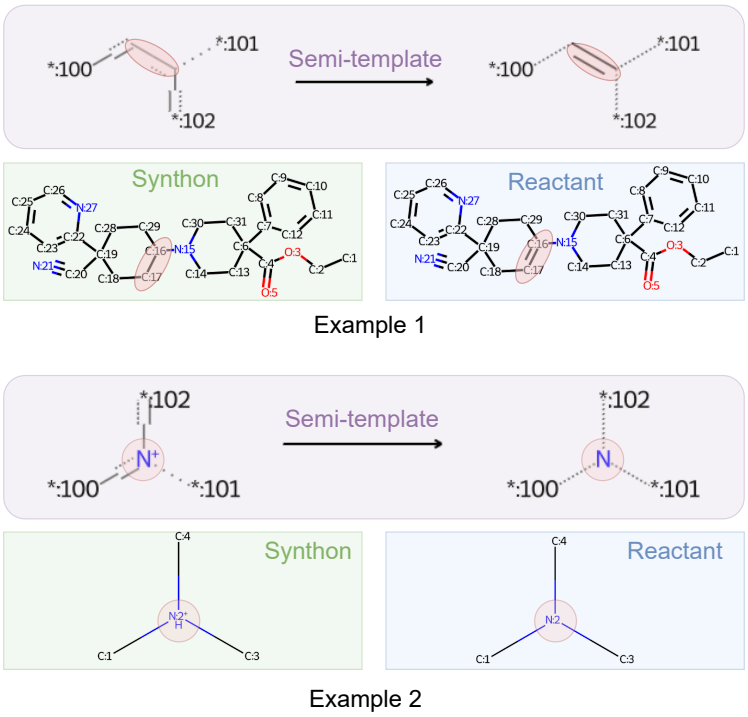

Figure 3: Semi-templates are more expressive than leaving groups. We show two examples that do not include residues because chemical reactions change the types of existing atoms or bonds rather than adding leaving groups. Semi-templates work well in those cases.

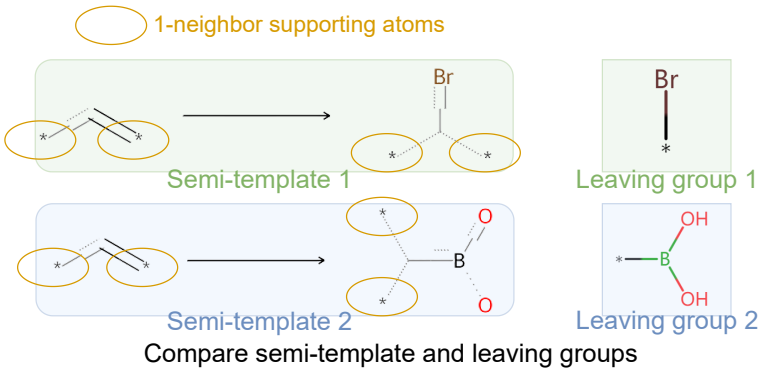

Figure 4: Semi-template uses 1-neighborhood supporting atoms to help the prediction.

Q1' "We cannot provide synthon completion results of (Sun et al., 2020), because this work: ..." Yes, I realize this method is not relevant but I was after a comparison to Somnath et al. (2020). I take your point that sadly no open source code is available, but are the results to be compared to not already provided in Somnath et al. (2020)'s Table 2? (I agree that G2G is an important additional baseline!)

A1' We appreciate the inspirational approach and outstanding results of Somnath et al. (2020). However, we are confused about the implementation details. They said "Given synthons and leaving groups, the attachment process has a 100% accuracy". Despite the cases that the residual cannot represent (seeing Figure. 3), we have no idea how to achieve 100% accuracy, especially when the residue has multiple attaching sites. They provide the idea without standard algorithm, which makes us question the validity of the synthon completion results. We really desire to know their experimental settings and notice that they used to have the source code, but their link is currently disabled, which is regretful for us. For fairness, we seriously treat reproducibility as a significant property, especially when there are some complicated steps, and we will try to contact the authors if possible. Once we confirm the validity of Somnath et al. (2020), we will add the comparison to it.

Q4' Thanks for clarifying! Does this mean that two semi-templates that have the exact same supporting atom set (but involve different transforms) will be ranked the same (or have I misunderstood)? Is this a problem?

A4' Yes, it is. Thans for your a insightful question! For more accurate prediction, we will consider a semi-template structure in the future.

3 REVIEWER XXDN

We appreciate your careful reading and thoughtful review. Below we address the concerns/questions mentioned in the review:

Q1: Are the semi-template in this paper and local reaction template the same stuff?

A1: They are similar but not the same stuff. Semi-template is more simpler than local reaction template. In short, the order of simplification is:

(a) full-template $<$ (b) local reaction template $<$ (c) semi-template.

where full-template contains the global structure of the product, local reaction template contains the local structure of the product, and semi-template contains the local structure of the synthon. The difference between (a) and (b) is that local reaction template removes the redundant global information from full-template. The difference between (b) and (c) is that semi-template breaks local local reaction templates into simpler parts, according to the synthon splitting. Thus, semi-template is more scalable and simpler than local reaction template.

We provide visual examples in Figure.5.

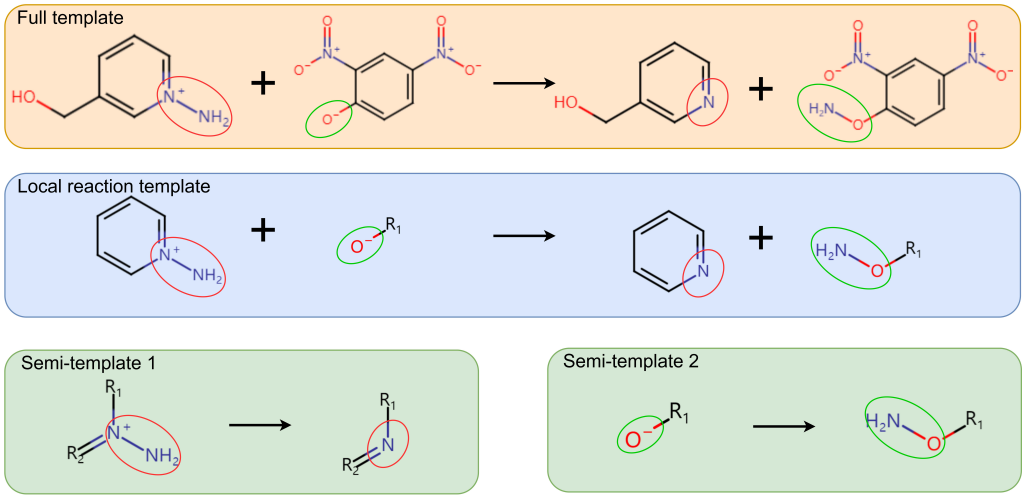

Figure 5: The conceptual illustration of full-template, local reaction template and semi-template. Local reaction template is the simple version of full template, both of which do not break the whole template into several semi-templates.

Q2 The presented DRGAT layer is based on the previous framework, what is the major contribution or novelty of DRGAT that comes from the authors? Could the authors clarify that?

A2 We add shortcut connections from the input to the output in each layer, and concatenate hidden representations of all layers to form the final node representation. We find that directly inputting the original edge features into each layer benefits the learning of edge-controlled attention weights, rather than using transformed edge features from the previous layer. As the multi-head attention and message passing mechanism are widely used in graph learning algorithm, we believe there are contributions in terms of retrosynthesis application.

Q3 The proposed method seems quite generalizable, could the authors also report results on the USPTO-full dataset like GLN to demonstrate the scalability of the proposed method?

A3 At present, we have conducted a part of experiments on USPTO-full dataset. The accuracies on center identification are 71.13% (top-1), 85.12% (top-2), 91.37% (top-3), 95.24% (top-5). More experimental results will be replenished immediately once our experiments are finished.

Q4 What is the detailed composition of atom and bond features mentioned in Table 1? I can not find them in the submission file.

A4 Atom and bond features used in this paper are the same as G2G, seeing Table. 4 and Table. 5 for atom features used in center identification and synthon completion. All tasks use the same bond features, seeing Table. 6. We also provide the Python code below.

Name	Description
Atom type	Type of atom (ex. C, N, O), by atomic number
# Hs	one-hot embedding for the total number of Hs (explicit and implicit) on the atom
Degree	one-hot embedding for the degree of the atom in the molecule including Hs
Valence	one-hot embedding for the total valence (explicit + implicit) of the atom
Aromaticity	Whether this atom is part of an aromatic system.
Ring	whether the atom is in a ring

Table 4: Atom features for center identification.

```
def atom_center_identification(atom):
    return onehot(atom.GetSymbol(), atom_vocab, allow_unknown=True) + \
           onehot(atom.GetTotalNumHs(), num_hs_vocab) + \
           onehot(atom.GetTotalDegree(), degree_vocab, allow_unknown=
                                                 True) + \
           onehot(atom.GetTotalValence(), total_valence_vocab) + \
           [atom.GetIsAromatic(), atom.IsInRing()]
```

Name	Description
Atom type	Type of atom (ex. C, N, O), by atomic number
# Hs	one-hot embedding for the total number of Hs (explicit and implicit) on the atom
Degree	one-hot embedding for the degree of the atom in the molecule including Hs
Ring	whether the atom is in a ring
Ring 3	whether the atom is in a ring of size 3
Ring 4	whether the atom is in a ring of size 4
Ring 5	whether the atom is in a ring of size 5
Ring 6	whether the atom is in a ring of size 6
Ring 6+	whether the atom is in a ring of size larger than 6

Table 5: Atom features for synthon completion.

```
def atom_synthon_completion(atom):
    return onehot(atom.GetSymbol(), atom_vocab, allow_unknown=True) + \
           onehot(atom.GetTotalNumHs(), num_hs_vocab) + \
           onehot(atom.GetTotalDegree(), degree_vocab, allow_unknown=
                                                 True) + \
           [atom.IsInRing(), atom.IsInRingSize(3), atom.IsInRingSize(4),
           atom.IsInRingSize(5), atom.IsInRingSize(6),
           atom.IsInRing() and (not atom.IsInRingSize(3)) and (not atom.
                                                 IsInRingSize(4)) \
           and (not atom.IsInRingSize(5)) and (not atom.IsInRingSize(6))
                                                 ]
```

Name	Description
Bond type	one-hot embedding for the type of the bond
Bond direction	one-hot embedding for the direction of the bond
Stereo	one-hot embedding for the stereo configuration of the bond
Conjugation	whether the bond is considered to be conjugated
Bond length	the length of the bond

Table 6: Bond features.

```
def bond_default(bond):
    return onehot(bond.GetBondType(), bond_type_vocab) + \
           onehot(bond.GetBondDir(), bond_dir_vocab) + \
           onehot(bond.GetStereo(), bond_stereo_vocab) + \
           [int(bond.GetIsConjugated())] + \
           bond_length(bond)
```

Q5 Have the authors removed the original mapping numbers from test data?
A5 We only use the mapping numbers when making groundtruth labels. During the training and evaluating phase, we do not use this information.

Updated A5 We really appreciate this insightful question. Thanks!

We follow the setting of G2G, which does not remove the original mapping numbers from test data but abandons this information during training and evaluation. During data preprocessing, we use the feature extractor of G2G to get the ground-truth center identification labels. Then we abandon atom mapping numbers and extract both node and edge features. We can ensure our results are reproducible under the same setup as G2G.

We find that **G2G achieves high accuracy because they use the directed edge embedding rather than leaking mapping numbers, i.e., using atom mapping numbers as auxiliary inputs.** We track thousands of data samples, find a potential risk sample, check its data flow across G2G's pipeline for hours, and finally find this secret. We provide a typical sample with its original and canonical product smiles as:

	product smiles
original	[c:1]1([NH:15][CH3:14])[n:2][c:3]([Cl:4])[n:5][c:6]([NH:7][CH:8]2[CH2:9][CH2:10][CH2:11][CH2:12]2)[n:13]1
canonical	[CH3:14][NH:15][c:1]1[n:2][c:3]([Cl:4])[n:5][c:6]([NH:7][CH:8]2[CH2:9][CH2:10][CH2:11][CH2:12]2)[n:13]1

Figure. 6 provides a visual example, where the input original and canonical products have the same structure, node features, and edge features, but the predicted results are different. What difference causes this phenomenon? We use handwritten numbers to mark the order of the atoms in the algorithm. Since GNNs are permutation invariant, the node output features of the two products are the same. However, G2G concatenates node features as the edge feature. What is more, G2G uses the directed graph, which means the first node index is smaller than the second node index in an edge. Formally, $e_{i,j} = (v_i, v_j), i < j$, where $e_{i,j}$ is the edge between node i and node j.

Attention! That is the problem! Because G2G uses directed edges, the model may perform differently when there is no consistent definition of this direction. Our experiment finds that the atom mapping numbers do not affect the results, but the graph direction does. Please see Figure. 6 for the detailed illustration.

Thanks for your question again! We believe a consistent direction definition in the original product facilitates solving the center identification problem. Further, automatically defining the consistent edge direction for any product graph and predicting it using neural networks may be a promising

proxy task for more general performance gains. We are going to investigate this in our further research. A simple solution is to set a threshold value for both predictions of the positive (or negative) direction, as shown in Table. 7.

	original		canonical		summary
	positive	negative	positive	negative	
bond [C:1]-[N:15]	> 0	> 0	> 0	> 0	✓
bond [C:14]-[N:15]	< 0	> 0	> 0	< 0	✗

Table 7: A simple solution. We choose the bond as predicted reaction centers if both of its negative and positive direction score are large than 0.

Finally, we emphasize that there two types of existing SOTA graph models:

1. Ones provide open source code, but the original code implementations are controversial or extremely complex, e.g., G2G (Shi et al., 2020), RetroXpert (Yan et al., 2020).
2. The other ones claim the SOAT performance but do not provide open source code, e.g., GraphRetro (Somnath et al., 2021), energy model (Sun et al., 2020).

We aim to make fair and meaningful comparisons of reliable results. Thus we follow the works with open-source code and the same experimental feature extractors and datasets. We strictly follow their specification, from dataset preprocessing to the model API to the result testing. Similar to G2G, our implementation does not use atom mapping numbers but the direction of the original products. This setting may inspire a new center identification problem with well-defined graph directions, compared to the previous direction unknown case. With this in mind, we can further explore how to predict the edge direction and edge centers.

We will eventually provide a solid open-source code to promote the research progress. Moreover, we look forward to seeing more open-source SOTA. Thanks!

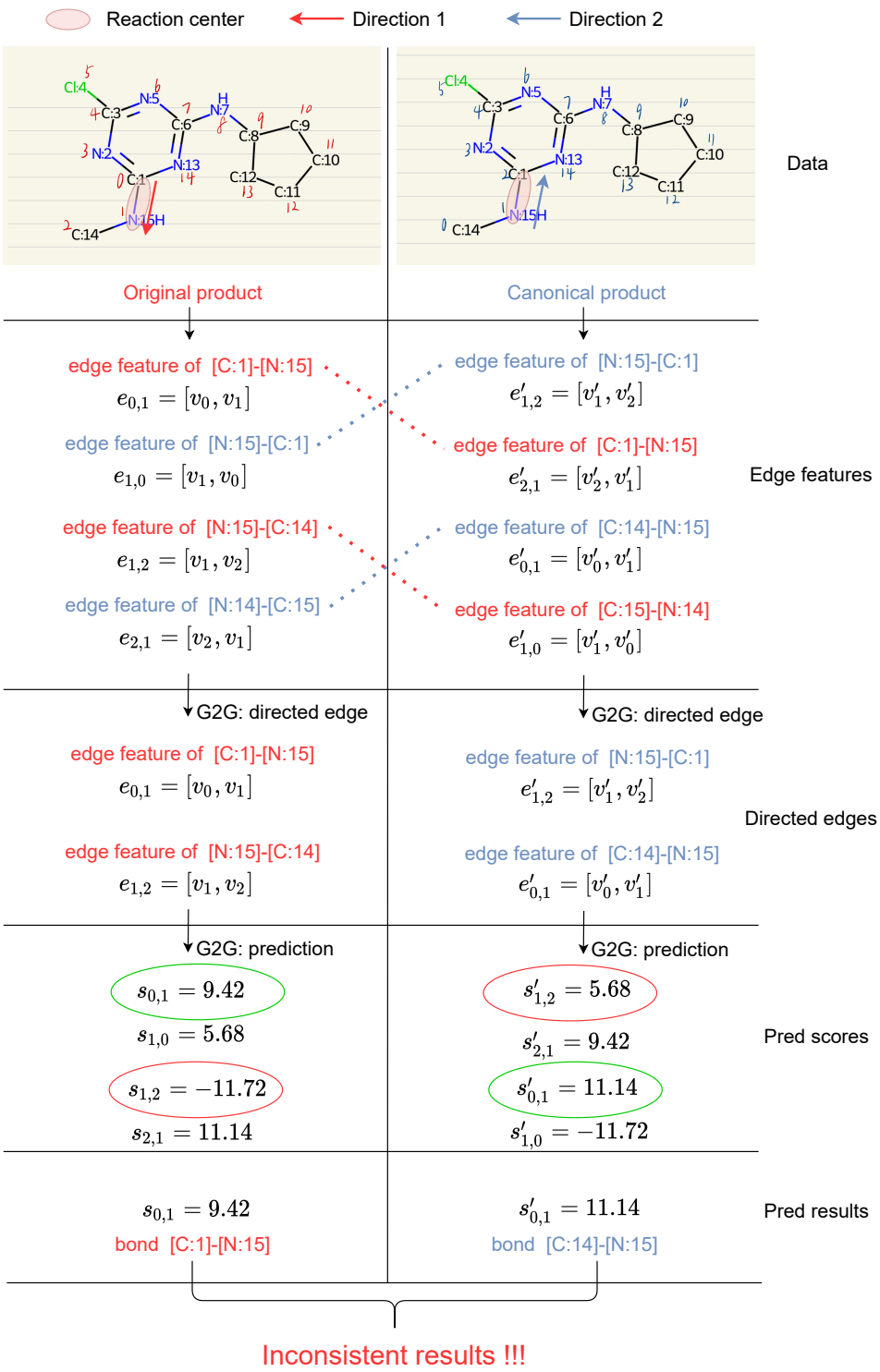

Figure 6: G2G using directed edge embedding. For bond [C:1]-[N:15], we define the positive direction as $[C:1] \rightarrow [N:15]$, and the negative direction as $[N:15] \rightarrow [C:1]$, which are marked in red and blue, respectively. When construcing edge features, G2G choose the positive (or negative) direction for original (or canonical) direction for bond [C:1]-[N:15]. The inconsistant direction setting leads to different edge feature mebeddings, thus change the predictive results.

4 REVIEWER 83GD

Thanks for your comments, we are glad that you enjoy reading this paper.

Q1 The explanation of the DRGAT model only appears in the Figure 3. I expect the revised manuscript add a few sentences/equations to convey the main idea of the model.

A1 Thanks for your constructive suggestion. In our revised version, we have added a detailed description in Sec. 4.1. During message passing, DRGAT extracts source and destination node's features via independent MLPs to consider the bond direction and use the multi-head edge controlled attention mechanism to consider the multi-relational properties. We add shortcut connections from the input to the output in each layer, concatenate hidden representations of all layers to form the final node representation.

Q2 Why GraphRetro and RetroXpert are not studied in the Sec. 5.3?

A2 GraphRetro (Somnath et al., 2020) is difficult to reproduce, especially when open-source code is unavailable (its data preprocessing and residual attaching algorithm are unknown), so we abandon this baseline. RetroXpert Yan et al. (2020) leaks atom mapping information, we doubt this setting is not the same as ours so that we give it up in comparison. We try to reproduce the revised version of RetroXpert, but do not get reasonable results following the official instructions, seeing `https://github.com/uta-smile/RetroXpert`. As G2G provides open-source code, we can conviently implement the same chemical features and code framework to ensure the fairness of experimental settings.

Q3 I found that one of the important previous work is missing from the discussions and the experiments; energy-based models. Please discuss the relationship with the proposed model, and if it is appropriate, compare in experiments.

A3 Thanks for your kind suggestion. In our revised version, the specified discussion have been added in Sec. 5.1. The energy-based approaches (Sun et al., 2020) are ignored because it is more like a plug-and-play training strategy and result filter, focusing on enhancing existing models. For simplicity, we leave the use of energy function in the future and concentrate on comparing original retrosynthesis models.

Q4 Readability: In general, figures are low-visibility. Especially, atoms, bonds, molecules, equations are not displayed well on screens. I hope these figures are improved in the camera ready as much as possible. For example, thick/strong lines and fonts, larger symbols, and so on.

A4 Thanks for your careful suggestion. We have updated all the figures in our paper to improve the readability.

typed: Test top-1: 0.6502092 top-3: 0.8476987 top-5: 0.8851465 top-10: 0.9135984 Test top-1: 0.5598326 top-3: 0.7445607 top-5: 0.7960252 top-10: 0.8353557

REFERENCES

Chence Shi, Minkai Xu, Hongyu Guo, Ming Zhang, and Jian Tang. A graph to graphs framework for retrosynthesis prediction. In *International Conference on Machine Learning*, pp. 8818–8827. PMLR, 2020.

Vignesh Ram Somnath, Charlotte Bunne, Connor W Coley, Andreas Krause, and Regina Barzilay. Learning graph models for template-free retrosynthesis. *arXiv preprint arXiv:2006.07038*, 2020.

Vignesh Ram Somnath, Charlotte Bunne, Connor W. Coley, Andreas Krause, and Regina Barzilay. Learning graph models for retrosynthesis prediction. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. URL `https://openreview.net/forum?id=SnONpXZ_uQ_`.

Ruoxi Sun, Hanjun Dai, Li Li, Steven Kearnes, and Bo Dai. Energy-based view of retrosynthesis. *arXiv preprint arXiv:2007.13437*, 2020.

Chaochao Yan, Qianggang Ding, Peilin Zhao, Shuangjia Zheng, JINYU YANG, Yang Yu, and Junzhou Huang. Retroxpert: Decompose retrosynthesis prediction like a chemist. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 11248–11258. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/819f46e52c25763a55cc642422644317-Paper.pdf.