Learning Condensed Graph via Differentiable Atom Mapping for Reaction Yield Prediction

Ankit Ghosh [1] **Gargee Kashyap** [1] **Sarthak Mittal** [2] **Nupur Jain** [1] **Raghavan B Sunoj** [1] **Abir De** [2]

Abstract

Yield of chemical reactions generally depends on the activation barrier, i.e., the energy difference between the reactant and the transition state. Computing the transition state from the reactant and product graphs requires prior knowledge of the correct node alignment (i.e., atom mapping), which is not available in yield prediction datasets. In this work, we propose YIELDNET, a neural yield prediction model, which tackles these challenges. Here, we first approximate the atom mapping between the reactants and products using a differentiable node alignment network. We then use this approximate atom mapping to obtain a noisy realization of the condensed graph of reaction (CGR), which is a supergraph encompassing both the reactants and products. This CGR serves as a surrogate for the transition state graph structure. The CGR embeddings of different steps in a multi-step reaction are then passed into a transformer-guided reaction path encoder. Our experiments show that YIELDNET can predict the yield more accurately than the baselines. Furthermore, the model is trained only under the distant supervision of yield values, without requiring fine-grained supervision of atom mapping.

1. Introduction

The *yield* of a chemical reaction is expressed as the percentage of conversion of the reactant(s) to product(s). Early prediction of yield before wet-lab validation of reactions can have an immense impact on ML-driven reaction discovery. It allows the identification and removal of low-yielding reactions, thereby helping in the design and optimization of chemical synthesis. Previous works on yield prediction (Ahneman et al., 2018; Nielsen et al., 2018; Zahrt et al., 2019; Sandfort et al., 2020; Schwaller et al., 2020; 2021c; Singh & Sunoj, 2022; Schleinitz et al., 2022; Lu & Zhang, 2022) have primarily relied on quantum chemically computed molecular descriptors, molecular fingerprints, or SMILES based representation rather than a more holistic representation built on molecular structure itself. In contrast, recent approaches (Saebi et al., 2021; Gong et al., 2021; Kwon et al., 2022; Li et al., 2023) leverage graph neural networks (GNNs) to compute the embedding using molecular graphs.

Yield of a reaction depends strongly on the activation energy, through Arrhenius equation (Arrhenius, 1889), which is associated with the potential energy of a transient chemical entity called the 'transition state'. It is possible to infer the transition state graph from the reactant and product graphs if we know the atom mapping, *i.e.*, the correspondence between the atoms of the reactants and the products (Kim et al., 2024). But most existing yield prediction datasets only include the reactants and products, lacking both transition states and atom mappings. Furthermore, the computation of the true atom mapping from the graph structures of reactants and products is a challenging task. Atom mapping computation involves solving various pairwise graph matching tasks, which are NP hard problems (Astero & Rousu, 2024). In practice, this is usually mitigated by using expert-curated rules. However, such inputs require manual intervention which could, in turn, limit their widespread deployment (Schwaller et al., 2021a). As a result, the existing works on yield prediction do not consider atom mapping or transition states into their model, leaving a significant room for improvement.

1.1. Our contributions

We address these challenges by designing a novel yield prediction network (YIELDNET). Specifically, we make the following contributions.

Differentiable approximation of atom mapping In the context of a chemical reaction, atom mapping refers to an alignment map between the reactant and product nodes, which ensures that the atom composition is preserved. Such an alignment can be obtained by solving a graph matching task between the reactants and the products (Körner & Apostolakis, 2008; Astero & Rousu, 2024). However, this necessitates an exploration of a vast permutation space,

[*]Equal contribution [1]Department of Chemistry, IIT Bombay [2]Department of Computer Science and Engineering, IIT Bombay. Correspondence to: Ankit Ghosh <ankitghosh@iitb.ac.in>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

which is daunting. To address this challenge, we introduce a fully differentiable node alignment network built on the Gumbel-Sinkhorn iterative procedure. This network takes GNN embeddings of the reactants and products as input and outputs an alignment (doubly stochastic or soft permutation) matrix which approximates the atom mapping.

Approximate condensed graph of reaction We use the atom mapping to compute a condensed graph of reaction (CGR) (Varnek et al., 2005; Nugmanov et al., 2019), as a supergraph containing both reactants and products. Owing to such construction, the CGR serves as a surrogate of the transition state. Subsequently, we feed this CGR into a second GNN to compute its embeddings.

Since the atom mapping is approximated by a neural network, the CGR becomes a trainable weighted graph. Therefore, this second GNN must allow differentiation with respect to the input CGR, which is not supported in standard off-the-shelf GNN models. To address this, we design a GNN that relies exclusively on differentiable operations on the input adjacency matrix, enabling end-to-end training using the CGR.

Neural reaction encoder A reaction may involve multiple sequential steps, often referred to as reaction path. Using the computed representations of the CGR for each step in hand, we compute the representation for the entire reaction path. For each elementary step in a multi-step reaction, we concatenate the embeddings of the CGR with *reaction-step encodings*, akin to positional encodings. Finally, these embeddings are fed into a transformer, and the output from the transformer is then used to compute the reaction yield.

Learning under distant supervision The key goal of YIELDNET is to predict reaction yield. While doing so, it learns an approximate atom mapping using our node alignment network. Notably, YIELDNET is trained solely under the distant supervision of the yield values, without any fine grained supervision of ground truth atom mapping or CGR. Since transition state plays an important role in reaction yield, our approach of using the continuous approximation of the CGR as a surrogate for transition state from the differentiable atom mapping, can enhance the inductive bias of the model.

Our experimental evaluation across multiple datasets show that YIELDNET is able to outperform several baselines by a significant margin. Furthermore, we observe that YIELDNET can effectively approximate atom mapping under the supervision of only reaction yields.

2. Related work

Apart from the aforementioned work on the yield prediction, in recent years, there has been an increasing interest in designing ML models for atom mapping (Schwaller et al., 2021a; Nugmanov et al., 2022; Astero & Rousu, 2024) and transition state (Pattanaik et al., 2020; Jackson et al., 2021; Makoś et al., 2021; Choi, 2023; Duan et al., 2023; Kim et al., 2024; Duan et al., 2025). However, the problem setup of these works is very different from ours. For instance, the aforementioned works on transition states focus only on predicting the transition states but not yield. To do so, they use the fine-grained supervision of ground truth transition states for training. Similarly, the goal of Nugmanov et al. (2022); Astero & Rousu (2024) is to predict the atom mapping, after training under the supervision of fine-grained ground truth atom mapping. In another work on atom mapping, Schwaller et al. (2021a) employed a large transformer over SMILES representations. In contrast to the above works, our key goal is to predict the yield. Hence, we train our network on datasets containing only yield values, not any ground truth atom mapping or transition states.

Our work is also related to graph neural networks (GNNs), other applications of ML in chemistry, and attention mechanisms in different domains. We briefly review each of them as follows.

Representation learning for graphs Graphs are structured objects that are different from images and texts. They need specialized representation learning methods to compute the graph embeddings. These graph representation learning methods can be broadly divided into two categories. The first set of works consists of transductive models, where the node embeddings are computed independently of each other (Grover & Leskovec, 2016; Perozzi et al., 2014). This requires us to train $O(|V|)$ embeddings separately for each node in a graph. The second set of works consists of inductive models, referred to as graph neural networks or GNNs (Gilmer et al., 2017; Hamilton et al., 2017; Kipf & Welling, 2017; Veličković et al., 2018; Zhang & Chen, 2018), that employ message passing neural networks with shared parameters. At the outset, their goal is to collect information from the neighborhoods at different distances from a node and cast it into a low dimensional representation vector. Such methods are widely used in link prediction (Zhang & Chen, 2018), node classification (Kipf & Welling, 2017), graph matching (Li et al., 2019b; Bai et al., 2019) *etc*.

ML applications in chemistry In recent years, there has been a surge of work that uses machine learning in a wide variety of chemical applications (Hirohara et al., 2018; Bjerrum, 2017; Liu et al., 2018; Huang et al., 2020; Honda et al., 2019; Wang et al., 2019c; Chithrananda et al., 2020). Several works have represented molecules as a customized string called SMILES (Anderson et al., 1987) and then applied CNNs (Hirohara et al., 2018) or sequence encoders (Bjerrum, 2017; Liu et al., 2018; Huang et al., 2020; Honda et al., 2019; Wang et al., 2019c; Chithrananda et al.,

2020). However, the same molecule can result in multiple SMILES strings, which might affect the expressivity of the trained model. Therefore, a wide body of work represents a molecule as a molecular graph and designs graph based machine learning models (Kearnes et al., 2016; Schütt et al., 2018; Qiao et al., 2020; Klicpera et al., 2020b;a; Fuchs et al., 2020; Samanta et al., 2020; Jin et al., 2018; Cao & Kipf, 2018). Such graph based models for molecular graphs can be used for generative modeling (Samanta et al., 2020; Jin et al., 2018; Cao & Kipf, 2018), property predictions (Yang et al., 2019; Axelrod & Gomez-Bombarelli, 2022; Liu et al., 2021; Chithrananda et al., 2020) *etc*.

Learning alignment in the context of graphs Graph matching naturally leads to a quadratic assignment problem (Anstreicher, 2003; Gold & Rangarajan, 1996; Caetano et al., 2009). Fey et al.; Zanfir & Sminchisescu (2018); Wang et al. (2019b) provide a learning model for training graph matching, under distant supervision. In last few years, there are some works on learning node to node alignment for subgraph matching (Roy et al., 2022b; Ramachandran et al., 2024; Raj et al., 2025), maximum common subgraph computation (Kriege et al., 2019; Roy et al., 2022a), graph edit distance (Jain et al., 2024; Li et al., 2019b), graph rerpresentation learning (Roy et al., 2021).

3. Preliminaries

Notations We represent a single-step reaction as $A + B \rightarrow Y + Z$, where $R = \{A, B\}$ are the reactants and $I = \{Y, Z\}$ are the products. Here, I typically consists of intermediates in multi-step reactions. Each step involves at most two reactants and two products, consistent with our datasets. For any molecular graph $G_A = (V_A, E_A)$, we use $\boldsymbol{V}_A \in \mathbb{R}^{|V_A| \times d_V}$ and $\boldsymbol{E}_A \in \mathbb{R}^{|E_A| \times d_E}$ to denote the node and edge feature matrices, respectively. In practice, we build the node feature for u, i.e., $\boldsymbol{V}_A[u, :]$ using the atomic number, degree of connectivity, hybridization, etc., and the edge feature for (u, v), i.e., $\boldsymbol{E}_A[(u, v), :]$ using the type of bonds respectively. We denote the reactant and product graphs as $G_R = (V_R, E_R)$ and $G_I = (V_I, E_I)$, and their adjacency matrices as Adj_R and Adj_I, respectively. We denote the set of $N \times N$ permutation matrices as $\mathcal{P}_N$. While $\boldsymbol{P}$ represents both the hard permutation matrix and the doubly stochastic matrix. For a hard permutation matrix $\boldsymbol{P}$, the alignment map π satisfies $\boldsymbol{P}[u, u'] = 1 \implies \pi[u] = u'$.

Reaction and their components A reaction path r comprises n elementary steps, with each step represented as $A_i + B_i \rightarrow Y_i + Z_i$, where Y_i and Z_i are intermediates. Inputs A_i and B_i may be initial reactants, catalysts, or intermediates from prior steps (Y_j or Z_j, $j < i$). The final product Y_n from the n-th step is the reaction's desired output, whose yield is the focus. Thus, r follows a sequential structure:

$$r = \{A_1 + B_1 \rightarrow Y_1 + Z_1, \cdots, A_n + B_n \rightarrow Y_n + Z_n\} \quad (1)$$

At step $i \leq n$, we denote $R_i = \{A_i, B_i\}$ and $I_i = \{Y_i, Z_i\}$. Chemical yield for a reaction can be defined as the observed amount of the desired final product (Y_n in Eq. (1)), which is generally less than the theoretical maximum amount of the product given by stoichiometric calculations.

We do incorporate both catalyst and solvent, when present, into the reactant set R. Temperature was excluded as it is invariant in the case of high-throughput experimentation datasets (e.g., DF and NS datasets (Section 5)), which maintain consistent reaction conditions throughout, or varies mildly for others (e.g., SC). Integrating temperature into the node/edge features might enhance CGR or yield prediction quality.

Activation energy and its relationship with yield In an elementary step reaction $R \rightarrow I$, reactants R form a transient and unstable chemical entity (shown as TS in Figure 1), before producing I. This chemical entity, called as transition state, has the highest potential energy in the reaction, and the energy required to overcome this barrier is the *activation energy* ($\Delta\mathcal{E}^{\ddagger}$). A lower activation energy leads to a faster reaction rate and higher yield (Kozuch & Shaik, 2011).

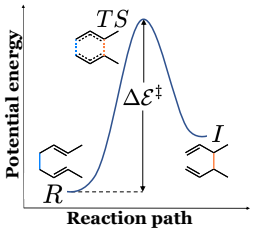

Figure 1: Reaction path vs potential energy

Atom mapping In a chemical reaction, only the bonds in the reactants R are rearranged—broken or newly formed— to produce the products I, while the atoms themselves remain conserved. This conservation establishes a correspondence, represented as a permutation $\pi : V_I \rightarrow V_R$, mapping the atoms in the products to those in the reactants. Here, π is the atom mapping.

Our goal We are given a set of training dataset $\mathcal{D}$ containing multi-step reactions along with the ground truth yield values, i.e., $\mathcal{D} = \{(r_1, \mathrm{yield}(r_1)), ..., (r_{|\mathcal{D}|}, \mathrm{yield}(r_{|\mathcal{D}|}))\}$. Any reaction r has n elementary steps as described in Eq. (1), where each elementary step contains a set of two-dimensional representations of the molecular graphs $\{R_i = (A_i, B_i), I_i = (Y_i, Z_i)\}_{i \in [n]}$ during both training and test. Our goal is to develop a yield prediction model that can approximate atom mapping and utilize this mapping to compute a surrogate for the transition state, enabling accurate yield prediction for a new reaction.

4. Proposed approach

For brevity of exposition, we first outline a yield prediction model for an ideal setup, where the atom mapping is known.

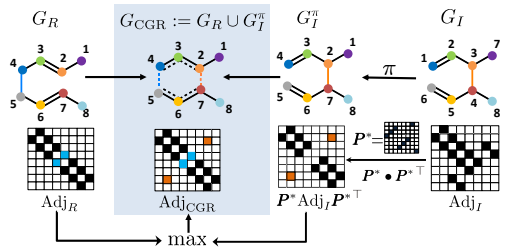

Figure 2: Top row: G_{CGR} is computed as the supergraph containing G_R and $G_I^\pi \cong G_I$, where G_I^π is obtained by applying atom-mapping π on V_I. Bottom row: Adjacency matrix of CGR is computed as $\max\left(\text{Adj}_R, \boldsymbol{P}^*\text{Adj}_I\boldsymbol{P}^{*\top}\right)$, where $\boldsymbol{P}^*$ is 0/1 permutation matrix corresponding to π. Given G_R and G_I, π can be obtained by minimizing the number of edges in the supergraph $G_R \cup G_I^\pi$: this renders in the maximum overlap between G_R and G_I^π.

Then, we introduce YIELDNET, a neural model that first learns an approximate atom mapping when its ground truth is unavailable, and then computes a condensed graph of reaction (CGR) as a surrogate for the transition state.

4.1. Yield prediction when the atom mapping is known

Activation energy directly impacts yield, making it imperative to incorporate transition states in yield prediction models. If the atom mapping π is known, we can design a yield prediction model in the following steps, which will enable us to express the yield as a trainable function of the CGR that approximates the corresponding transition state.

(1) Given the reactants R and the products I in an elementary step reaction, we permute V_I using the atom mapping π to construct $G_I^\pi = (\pi(V_I), E_I(\pi(V_I)))$ which is isomorphic to G_I. Then, we compute the graph of transition state as the condensed graph $G_{\text{CGR}} := G_R \cup G_I^\pi$, *i.e.*, a supergraph which subsumes the structures of both G_R and G_I (Figure 2). Suppose $\boldsymbol{P}^* \in \mathcal{P}_N$ is the gold permutation matrix corresponding to π, where $N = |V_R| = |V_I|$ obtained after padding. This allows us to approximate the adjacency matrix of the CGR using the adjacency matrices Adj_R and Adj_I as follows.

$$\text{Adj}_{\text{CGR}} = \max\left(\text{Adj}_R, \boldsymbol{P}^*\text{Adj}_I\boldsymbol{P}^{*\top}\right). \tag{2}$$

(2) We compute the node embeddings of G_{CGR_i} for each elementary step $i \leq n$ of an $n-$step reaction (1).
(3) We feed these node embeddings into a sequence-to-sequence encoder to compute the yield.

4.2. YIELDNET model

Overview For most real-life chemical datasets, the atom mapping π or the permutation matrix $\boldsymbol{P}^*$ is typically not readily available. Hence, we design neural networks to approximate atom mapping (item (1) above), which leads to an approximate CGR for each reaction step. We achieve this in the following steps, as illustrated in Figure 3.

(I) We integrate a graph neural network (GNN) that computes both node embeddings and edge embeddings.
(II) We feed these node embeddings into a differentiable node alignment network, Align, to obtain an alignment matrix $\boldsymbol{P} \approx \boldsymbol{P}^*$.
(III) To ensure the differentiability of the graph structures of CGR, we use the edge embeddings to obtain a smooth approximation of the adjacency matrix of CGR.
(IV) We compute the embeddings of CGR using an input-differentiable GNN.
(V) Finally, we combine the embeddings of CGRs of each elementary step using a sequence encoder and obtain the yield.

In the following we describe the above steps in details.

Embedding computation using GNN Given a multistep reaction r, each elementary step consists of a reactant graph G_R, a product graph G_I and their node and edge features $\boldsymbol{V}_R, \boldsymbol{E}_R$ and $\boldsymbol{V}_I, \boldsymbol{E}_I$, each with N nodes, obtained after padding. We apply a graph neural network GNN_θ with parameter θ on these graphs to compute their node embeddings $\boldsymbol{H}_\bullet$, and edge embeddings $\boldsymbol{M}_\bullet$, as follows:

$$\begin{aligned}\boldsymbol{H}_R, \boldsymbol{M}_R &= \text{GNN}_\theta(G_R, \boldsymbol{V}_R, \boldsymbol{E}_R),\\ \boldsymbol{H}_I, \boldsymbol{M}_I &= \text{GNN}_\theta(G_I, \boldsymbol{V}_I, \boldsymbol{E}_I).\end{aligned} \tag{3}$$

Here $\boldsymbol{H}_\bullet \in \mathbb{R}^{N\times d}$ and $\boldsymbol{M}_\bullet \in \mathbb{R}^{N\times N\times D}$. For brevity, we present our analysis with $D = 1$ and defer the general discussion using tensor form in Appendix C. To elaborate, for the reactants R, $\boldsymbol{H}_R[u,:] = \boldsymbol{h}_R(u) \in \mathbb{R}^d$ is the embedding of node u, while $\boldsymbol{M}_R[u,v]$ is the message value $m_R(u,v)$ for an edge $(u,v) \in E_R$ and zero for non-edges $(u,v) \notin E_R$. Similarly, we compute $\boldsymbol{H}_I$ and $\boldsymbol{M}_I$.

Differentiable approximation of atom mapping To address the challenge of learning $\boldsymbol{P}^*$, *i.e.*, the permutation matrix corresponding to the unknown atom mapping, we adopt a data-driven approach. Ideally, $\boldsymbol{P}^*$ should be a binary 0/1 matrix, but such discrete values attenuate gradient signals and prevent backpropagation. To overcome this, we approximate $\boldsymbol{P}^*$ using a node alignment network Align. This network takes the node embeddings of the reactant and product graphs, $\boldsymbol{H}_R$ and $\boldsymbol{H}_I$ as input, and outputs an alignment matrix $\boldsymbol{P}$ which approximates $\boldsymbol{P}^*$. The matrix $\boldsymbol{P}$ is a continuous doubly stochastic matrix, which enables smooth end-to-end optimization.

$$\boldsymbol{P} = \text{Align}(\boldsymbol{H}_R, \boldsymbol{H}_I) \tag{4}$$

$$= \text{Sinkhorn}\left(-\sum_{\ell=1}^{d}[\max(\boldsymbol{H}_R[u,\ell], \boldsymbol{H}_I[u',\ell])]_{u,u'}\right) \tag{5}$$

Here, $\text{Sinkhorn}(\cdot)$ performs iterative Sinkhorn normalizations on the input matrix (Cuturi, 2013; Mena et al., 2018). Given a temperature $\lambda > 0$, an input matrix $\boldsymbol{C}$ and the number of iterations T, $\text{Sinkhorn}(\boldsymbol{C}) = \text{Sinkhorn}^{(T)}(\boldsymbol{C})$

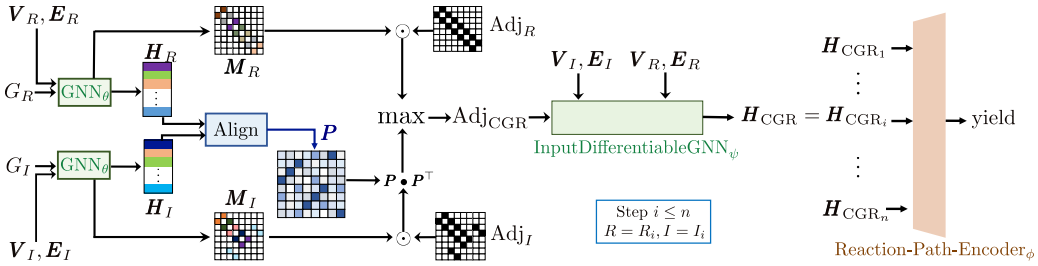

Figure 3: Illustration of YIELDNET. For each reaction step i, we process the reactants R and products I using a GNN, GNN_θ, to obtain node embeddings $\boldsymbol{H}_R, \boldsymbol{H}_I$ and edge embeddings $\boldsymbol{M}_R, \boldsymbol{M}_I$. We feed them into a fully differentiable node alignment network, Align, to obtain an alignment matrix $\boldsymbol{P}$ which approximates the permutation matrix corresponding to the atom mapping. Then, we use $\boldsymbol{P}$ to obtain a continuous approximation of the adjacency matrix of CGR, Adj_{CGR}. Next, we feed this approximate Adj_{CGR} with the node and features into a new GNN ($\text{InputDifferentiableGNN}_\psi$) to obtain $\boldsymbol{H}_{\text{CGR}_i}$, the node embedddings of CGR. Finally, a reaction path encoder use $\{\boldsymbol{H}_{\text{CGR}_i}\}$ to predict the yield.

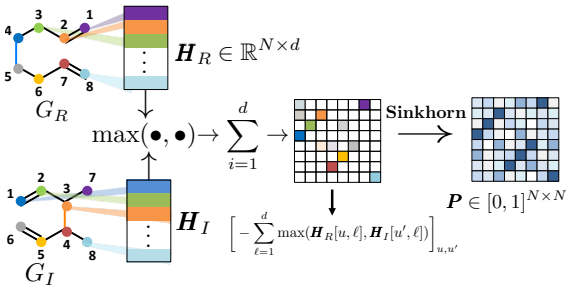

Figure 4: Differentiable approximation of atom mapping using node alignment network Align. It takes $\boldsymbol{H}_R$ and $\boldsymbol{H}_I$, the node embeddings of G_R and G_I as input; and, outputs an alignment matrix $\boldsymbol{P}$, by computing iterative Sinkhorn iterations on the matrix $[-\sum_\ell \max(\boldsymbol{H}_R[u,\ell], \boldsymbol{H}_I[u',\ell])]_{u.u'}$.

is computed as follows:

$$\text{Sinkhorn}^{(0)}(\boldsymbol{C}) = \exp(\boldsymbol{C}/\lambda) \tag{6}$$

$$\text{Sinkhorn}^{(t+1)}(\boldsymbol{C}) = \text{RowDiv}(\text{ColDiv}(\text{Sinkhorn}^{(t)}(\boldsymbol{C}))). \tag{7}$$

Here, RowDiv (ColDiv) indicates division by sum of rows (columns). Note that the limit $\lim_{T\to\infty} \text{Sinkhorn}^{(T)}(\boldsymbol{C})$ converges to a doubly stochastic matrix and the limit $\lim_{\lambda\to 0} \lim_{T\to\infty} \text{Sinkhorn}^{(T)}(\boldsymbol{C})$ converges to a permutation matrix. We run the iterations in Eq. (7) a total of $T = 10$ times.

Rationale behind our design choice: Here, we justify the choice of node alignment network in Eq. (5) by connecting this design decision to a combinatorial heuristic for determining atom mapping. From the combinatorial viewpoint, we can compute the permutation matrix $\boldsymbol{P}^*$ corresponding to atom mapping, by minimizing the size of the supergraph containing the structures of R and I (Figure (2)). In terms of the adjacency matrices, this may be written as:

$$\boldsymbol{P}^* = \underset{\boldsymbol{P}\in\mathcal{P}_N}{\text{argmin}} \sum_{u,v} \max\left(\text{Adj}_R, \boldsymbol{P}\text{Adj}_I\boldsymbol{P}^\top\right)[u,v]. \tag{8}$$

The above optimization problem is a quadratic assignment problem (QAP) due to the quadratic involvement of $\boldsymbol{P}$, and is NP-hard as well. To tackle this challenge, we view a graph as the "set" of node embeddings and relax the optimization (8) into the problem of minimizing the approximate size of the superset containing $\boldsymbol{H}_R$ and $\boldsymbol{H}_I$. Specifically, we seek to solve: $\min_{\boldsymbol{P}\in\mathcal{P}_N} \sum_{\ell=1}^{d} \max(\boldsymbol{H}_R, \boldsymbol{P}\boldsymbol{H}_I)[\ell]$. This is same as solving the following optimization task, due to the fact that in each row of $\boldsymbol{P}$, exactly one element equals to one.

$$\min_{\boldsymbol{P}\in\mathcal{P}_N} \sum_{\ell=1}^{d} \sum_{u,u'} \max\Big(\boldsymbol{H}_R[u,\ell], \boldsymbol{H}_I[u',\ell]\Big)\boldsymbol{P}[u,u']. \tag{9}$$

The above problem is a linear optimal transport (OT) problem (Kuhn, 1955; Villani, 2008) and is solvable in polynomial time. However, the optimal solution of the above optimization (9) is still a 0/1 hard permutation matrix, which diminishes gradient signals. To address this challenge, we perform a further relaxation of the optimization (9) and solve the following entropy regularized linear OT problem.

$$\min_{\boldsymbol{P}} \sum_{\ell=1}^{d} \sum_{u,u'} \max\Big(\boldsymbol{H}_R[u,\ell], \boldsymbol{H}_I[u',\ell]\Big)\boldsymbol{P}[u,u'] - \lambda \cdot \text{Entropy}(\boldsymbol{P}), \tag{10}$$

$$\text{such that: } \boldsymbol{P} \ge 0, \boldsymbol{P}\mathbf{1} = \boldsymbol{P}^\top\mathbf{1} = 1. \tag{11}$$

Here, $\lambda > 0$ is the regularizer coefficient. As shown by Cuturi (2013); Mena et al. (2018), our alignment matrix $\boldsymbol{P}$ obtained using Eq. (5) is the solution of the above OT problem (10) – (11). Therefore, our node alignment network Align (4) can approximate the true atom mapping, enhancing the overall inductive bias of our model.

Continuous approximation of CGR Having computed our alignment matrix $\boldsymbol{P}$ (5), we make a continuous approximation of Adj_{CGR} (2), which serves as a surrogate of the transition state of the underlying reaction. This is achieved by performing elementwise multiplication of Adj_R and Adj_I with the messages collected from edge embeddings $\boldsymbol{M}_R$ and $\boldsymbol{M}_I$, generated by GNN_θ in Eq. (3), *i.e.*,

$$\text{Adj}_{\text{CGR}} \approx \max(\text{Adj}_R \odot \boldsymbol{M}_R, \boldsymbol{P}\text{Adj}_I \odot \boldsymbol{M}_I\boldsymbol{P}^\top) \tag{12}$$

Even without this continuous approximation, the discrete adjacency matrix $\max(\text{Adj}_R, \boldsymbol{P}\text{Adj}_I\boldsymbol{P}^\top)$ is differentiable, since the matrix $\boldsymbol{P}$ is now the output of the node alignment

network. However, there are two key advantages of using the continuous approximation (12).

(1) $\boldsymbol{M}_\bullet$ can model transient bonds: CGR represents the transient state, where the edges represent transient fractional bonds. This can be captured better using continuous weights $\boldsymbol{M}$ rather than binary 0/1 values.

(2) Continuous weights enhance backpropagation: Binary adjacency matrices suppress gradient signals, limiting model training, whereas the continuous weights help in backpropagation through the GNN. For instance, if instead of $\text{Adj}_I \odot \boldsymbol{M}_I$, we use only the binary adjacency matrix Adj_I, then the gradient term will involve $\boldsymbol{P}\text{Adj}_I \frac{d\boldsymbol{P}^\top}{d\theta} + \frac{d\boldsymbol{P}}{d\theta}\text{Adj}_I\boldsymbol{P}^\top$. Since $\boldsymbol{P}$ involves iterative normalization starting with exponentials, $\frac{d\boldsymbol{P}}{d\theta}$ includes products like $\boldsymbol{P}[u,u']\boldsymbol{P}[v,v']$, due to the fact that $\frac{d}{do_k} e^{o_j}/\sum_i e^{o_i} = -(e^{o_j}/\sum_i e^{o_i})(e^{o_k}/\sum_i e^{o_i})$ when $k \neq j$. Thus, $\boldsymbol{P}\text{Adj}_I \frac{d\boldsymbol{P}^\top}{d\theta}$ contains terms of the form $\boldsymbol{P}[u,u']\boldsymbol{P}[v,v']\boldsymbol{P}[w,w']$, with each $\boldsymbol{P}[.,.] \in [0,1]$, which weakens the gradient signals.

Instead, in our approximation (12), the gradient will involve: $\boldsymbol{P}\text{Adj}_I \odot \frac{d\boldsymbol{M}_I}{d\theta}\boldsymbol{P}^\top + \boldsymbol{P}\text{Adj}_I \odot \boldsymbol{M}_I\frac{d\boldsymbol{P}^\top}{d\theta} + \frac{d\boldsymbol{P}}{d\theta}\text{Adj}_I \odot \boldsymbol{M}_I\boldsymbol{P}^\top$. The first term will include entries like $\boldsymbol{P}[u,u']\boldsymbol{P}[v,v'][\frac{d\boldsymbol{M}_I^\top}{d\theta}][w,w']$. While the last two terms result in the entries like $\boldsymbol{P}[u,u']\boldsymbol{P}[v,v']\boldsymbol{P}[w,w']\boldsymbol{M}[\omega,\omega']$, instead of $\boldsymbol{P}[u,u']\boldsymbol{P}[v,v']\boldsymbol{P}[w,w']$ in binary adjacency matrix representations. These terms with $\boldsymbol{M}$ will enhance the gradient signals.

Embedding computation for CGR Next, we compute the node embeddings of the continuous approximation of the CGR, derived in Eq. (12). We first compute the initial node and edge features $\boldsymbol{V}_{\text{CGR}}$ and $\boldsymbol{E}_{\text{CGR}}$, which are required to initialize the GNN embeddings and the message propagation process. These features are composed of two key components. The first component is identical to $\boldsymbol{V}_R$ and $\boldsymbol{E}_R$, *i.e.*, the initial features of the reactants, reflecting the reactants' role in transitioning to the product via the condensed graph. These features are critical as they capture the reactants' structure prior to transformation. The second component is the difference between the features of the reactant set R and the product set I, which captures the effect of the reacting atoms and edges. To compute this difference, the nodes and edges of the reactants are aligned with those of the products using the approximate atom mapping $\boldsymbol{P}$ derived from Eq. (5). Therefore, we have:

$$\boldsymbol{V}_{\text{CGR}} = [\boldsymbol{V}_R; \boldsymbol{V}_R - \boldsymbol{P}\boldsymbol{V}_I]\,, \tag{13}$$

$$\boldsymbol{E}_{\text{CGR}} = \left[\boldsymbol{E}_R; \boldsymbol{E}_R - \boldsymbol{P}\boldsymbol{E}_I\boldsymbol{P}^\top\right]. \tag{14}$$

Next, we input the features $\boldsymbol{V}_{\text{CGR}}, \boldsymbol{E}_{\text{CGR}}$ and the weighted adjacency matrix Adj_{CGR} (12), into a specialized GNN model to compute the node embeddings for the CGR. This process requires the GNN to support backpropagation through both the input adjacency matrix Adj_{CGR} and the features $\boldsymbol{V}_{\text{CGR}}$ and $\boldsymbol{E}_{\text{CGR}}$. However, standard GNN implementations assume fixed, discrete graph structures and do not allow differentiation through the input graph. To overcome this limitation, we design a custom GNN that performs message passing and aggregation using tensorized, differentiable operations on the entries of Adj_{CGR}, $\boldsymbol{V}_{\text{CGR}}$, and $\boldsymbol{E}_{\text{CGR}}$. We denote this GNN as $\text{InputDifferentiableGNN}_\psi$, parameterized by ψ, which computes the embeddings $\boldsymbol{H}_{\text{CGR}} \in \mathbb{R}^{N\times d_H}$ as shown below. Appendix C contains more details.

$$\boldsymbol{H}_{\text{CGR}} = \text{InputDifferentiableGNN}_\psi(\text{Adj}_{\text{CGR}}, \boldsymbol{V}_{\text{CGR}}, \boldsymbol{E}_{\text{CGR}}) \tag{15}$$

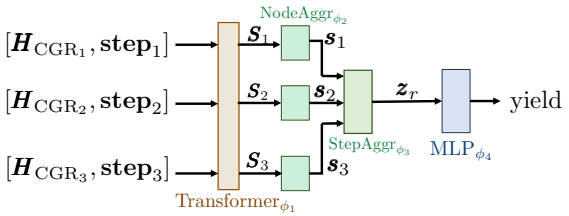

Figure 5: Reaction path encoder. For an n-step reaction r with $n = 3$, we encode each step $i \leq n$ by concatenating $\boldsymbol{H}_{\text{CGR}_i}$, the node embeddings of CGR_i, with a step encoding. A transformer then processes this to obtain $\boldsymbol{S}_i$. Next, $\boldsymbol{S}_i$ is aggregated into $\boldsymbol{s}_i$ via NodeAggr_{ϕ_2}, and $\{\boldsymbol{s}_i : i \leq n\}$ is passed through StepAggr_{ϕ_3} to compute the reaction embedding $\boldsymbol{z}_r$. We use an MLP on $\boldsymbol{z}_r$ to predict the yield.

Reaction path encoder For a multistep reaction r, the elementary steps occur in a unique sequence. Therefore, encoding the reaction path requires that CGR of each step should be associated with a signal that captures its position in the reaction path. However, the node embeddings $\boldsymbol{H}_{\text{CGR}}$ for the different steps are independent of the ordering of the steps. Therefore, to embed condensed graphs in a sequence-aware manner, we concatenate $\boldsymbol{H}_{\text{CGR}}$ with *step-encodings* similar to positional embeddings and pass them through a layer of transformer. Given a n-step reaction r, we first compute the sequence-aware node embeddings $\boldsymbol{S}_i$:

$$\overline{\boldsymbol{H}}_i = [\boldsymbol{H}_{\text{CGR}_i}, \textbf{step}_i]\,, \quad i \in \{1,..,n\} \tag{16}$$

$$\boldsymbol{S}_1, ..., \boldsymbol{S}_n = \text{Transformer}_{\phi_1}\left(\overline{\boldsymbol{H}}_1, ..., \overline{\boldsymbol{H}}_n.\right) \tag{17}$$

Here, CGR_i is the CGR for the i-th elementary step. Next, we aggregate $\boldsymbol{S}_i$ to their graph level embedding $\boldsymbol{s}_i$ using a NodeAggr, which is a permutation invariant set encoder (Vinyals et al., 2016). We compute $\boldsymbol{s}_i$ as:

$$\boldsymbol{s}_i = \text{NodeAggr}_{\phi_2}(\boldsymbol{S}_i[1,:], ..., \boldsymbol{S}_i[N,:]), \text{ for } i \leq n. \tag{18}$$

We could alternatively apply NodeAggr to $\boldsymbol{H}_{\text{CGR}_i}$ first, followed by the Transformer. However, this approach would aggregate the node embeddings of a CGR into its graph embedding before introducing interactions. Hence, this would only allow interactions between CGRs at a more coarse level. In contrast, applying the Transformer before NodeAggr allows the model to capture finer interactions between atoms within the CGRs across different steps.

Next, we aggregate all the graph embeddings $\{\boldsymbol{s}_i : i \leq n\}$

to obtain the reaction embedding $\boldsymbol{z}_r$ using StepAggr_{ϕ_3}, which is another set encoder of the same architecture as proposed by (Vinyals et al., 2016). Finally, we feed the $\boldsymbol{z}_r$ into a multilayer perceptron (MLP) to predict reaction yield value for the underlying multistep reaction r.

$$\boldsymbol{z}_r = \text{StepAggr}_{\phi_3}(\boldsymbol{s}_1, ..., \boldsymbol{s}_n) \tag{19}$$

$$\text{yield} = \text{MLP}_{\phi_4}(\boldsymbol{z}_r). \tag{20}$$

Here, the set $\phi = \{\phi_1, \phi_2, \phi_3, \phi_4\}$ is the collection of all trainable parameters.

4.3. Training

We conduct end-to-end training of the entire network to minimize the mean squared error (MSE) between the predicted and gold yield values, which serves as the loss function. Note that our node alignment network does not explicitly ensure that atoms of the reactants only match with the same atoms in the products, e.g., it does not ensure carbon matches with carbon and not oxygen. We ensure this goal by explicit regularization described as follows.

Regularizer to constrain atom mapping We introduce a regularizer that penalizes the $\boldsymbol{P}[u, v]$ values for those (u, v) pairs where u and v have different atomic identities (or, atomic numbers) since an atom doesn't convert to another atom throughout the reaction. Specifically, we introduce a matrix $\Lambda \in \mathbb{R}^{N \times N}$ where $\Lambda[u, v] = \mathbb{I}(\text{atom}_R(u) \neq \text{atom}_I(v))$ and $\mathbb{I}(\cdot)$ is the indicator function. Then, we compute the regularizer $\text{Reg}(R, I) = ||\boldsymbol{P} \odot \Lambda||_F^2$ for each CGR to limit the exploration space of $\boldsymbol{P}$, where $\boldsymbol{P}$ is computed in Eq. (5).

Loss Given a set of reactions $\mathcal{D} = \{(r_1, \text{yield}(r_1)), ..., (r_{|\mathcal{D}|}, \text{yield}(r_{|\mathcal{D}|}))\}$, we minimize the sum of the squared error between predicted yield, $\text{yield}_{\theta,\psi,\phi}(r)$ computed using Eq. (20) and the ground truth yield $\text{yield}(r)$, in the presence of the atom mapping regularizer as described above. Given a regularization parameter ρ, the training optimization becomes:

$$\min_{\theta,\psi,\phi} \sum_{r \in \mathcal{D}} \Big[(\text{yield}_{\theta,\psi,\phi}(r) - \text{yield}(r))^2 + \rho \sum_{(R,I) \in r} \text{Reg}(R, I)\Big] \tag{21}$$

5. Experiments

In this section, we provide a comprehensive evaluation of our method across eight datasets. Appendix E contains additional experiments and results on more datasets including USPTO dataset (Lowe, 2017). Our code is available in https://github.com/ankitthreo/YieldNet.git.

5.1. Experimental setup

Datasets We carry out our experiments using eight datasets. They include – (1) GP dataset which is derived from Gas-phase Isomerization reactions (Grambow et al., 2020b); five datasets derived from catalytic asymmetric N, S-acetal formation reaction (Zahrt et al., 2019), *viz.*, (2) NS1, (3) NS2, (4) NS3, (5) NS4, (6) NS5; one dataset on (7) Suzuki coupling reaction (SC); and, another dataset on (8) Deoxyflurorination (Nielsen et al., 2018) (DF). GP is the simplest dataset, where each reaction is single-step reaction and the ground truth atom mapping is available. The remaining seven datasets involve multi-step reactions and *do not* contain any ground truth atom mapping. Availability of atom mapping in GP helps us to better evaluate our atom mapping approximator and understand the effect of CGR on yield prediction, which is not possible for the other more commonplace datasets. Appendix D contains more details about the datasets.

Baselines We evaluate our model against several baselines including (1) YieldBERT (Schwaller et al., 2021c), (2) DeepReac+ (Gong et al., 2021) and several variants of graph neural networks: (3) graph convolutional network (GCN) (Kipf & Welling, 2017), (4) heterogenous graph transformer (HGT) (Hu et al., 2020), (5) topology adaptive graph convolutional networks (TAG) (Du et al., 2017) and (6) graph isomorphism network (GIN) (Xu et al.).

Evaluation metrics We partitioned the datasets into 70% training, 10% validation, and 20% test folds. We generated ten random splits using different random seeds. For each split, we measure the performance in terms of mean absolute error (MAE) between the predicted and ground truth yield values on the test set, and then average it across all ten splits to report the overall performance.

5.2. Results

Comparison with yield predictor baselines First, we compare YIELDNET against the baselines across eight datasets. We also report on a skyline variant (YIELDNET (sky)), which predicts yield using the true CGR, instead of an approximation. Since true CGR is only available in GP, skyline performance is limited to only GP dataset. Table 1 shows the results in terms of MAE. We make the following observations. (1) YIELDNET (sky) outperforms YIELDNET, highlighting CGR's importance in yield prediction. (2) YIELDNET outperforms all baselines in seven out of eight datasets; and, in five datasets, the performance gains are statistically significant. The baselines do not explicitly approximate CGR and instead use the embeddings of the reactants and products as-they-are, leading to weaker performance. (3) There is no consistent second-best model; GCN, TAG, DeepReac+, and YieldBERT share the second-best across datasets.

Evaluation of our atom mapping approximation Next, we compare our approximate atom mapping computed using Eq. (5), with four existing atom mapping methods. They include (1) RDKit (Landrum, 2013): the alignment strategy built into the RDKit library, (2) RXNMapper: a pre-trained

Model	GP	NS1	NS2	NS3	NS4	NS5	SC	DF
GCN	38.057 ± 0.337	11.162 ± 0.735	9.416 ± 1.045	8.813 ± 0.857	8.777 ± 0.221	4.453 ± 0.293*	10.682 ± 0.396	12.516 ± 0.329
HGT	39.405 ± 0.231	13.385 ± 0.844	9.573 ± 1.016	9.421 ± 0.835	8.629 ± 0.249	4.561 ± 0.277	13.665 ± 0.334	19.500 ± 0.257
TAG	36.887 ± 0.412	13.246 ± 0.760	9.560 ± 0.994	9.232 ± 0.860	8.603 ± 0.230	4.547 ± 0.286	12.603 ± 0.450	18.150 ± 0.447
GIN	38.307 ± 0.450	13.318 ± 0.825	9.588 ± 1.017	9.152 ± 0.860	8.617 ± 0.217	4.568 ± 0.285	12.899 ± 0.389	17.126 ± 0.479
DeepReac+	27.837 ± 0.346	12.985 ± 1.452	10.729 ± 0.690	9.968 ± 0.981*	9.439 ± 0.775*	5.125 ± 0.301*	16.953 ± 1.800	14.022 ± 1.694
YieldBERT	40.910 ± 0.300	12.446 ± 0.820	9.593 ± 1.002	8.780 ± 0.918*	9.236 ± 0.290	4.532 ± 0.255	10.437 ± 0.328	12.470 ± 0.412
YIELDNET	23.152 ± 0.393	9.245 ± 0.518	8.387 ± 0.907	7.914 ± 0.931	7.015 ± 0.495	4.382 ± 0.249	8.751 ± 0.438	6.941 ± 0.192
YIELDNET (sky)	17.396 ± 0.357	NA	NA	NA	NA	NA	NA	NA

Table 1: Comparison of yield prediction performance for YIELDNET against all the competitive baselines, *viz.*, GCN (Kipf & Welling, 2017), HGT (Hu et al., 2020), TAG (Du et al., 2017), GIN (Xu et al.), DeepReac+ (Gong et al., 2021), YieldBERT (Schwaller et al., 2021c), on the 20% test examples, across all datasets. Performance is measured in terms of Mean Absolute Error (MAE). YIELDNET(sky) represents a skyline of our model, where we use the true CGR. Only GP dataset contains the true CGR and therefore, such skyline performance is not available for other datasets. Numbers in green (yellow) indicate the best (second best) performer. Our improvement in performance over the next best baseline, where YIELDNET is the best performer, is statistically significant with p-value < 0.05, except in the cases marked with *.

Model	NS1	NS2	NS3	SC
RDKit	9.910	8.845	8.567	–
RXNMapper	9.610	8.871	8.470	10.046
Random	11.195	8.984	8.695	10.223
Attention	9.548	9.024	8.432	8.826
YIELDNET	9.245	8.387	7.914	8.751

Table 2: Comparison between different atom mapping strategies in terms of MAE of yield prediction

node alignment model (Schwaller et al., 2021a) on a large set of reactions, (3) Random: Uniformly sampled alignment, and (4) Attention: an attention network between reactants and products based on Astero & Rousu (2024), that induces a non-injective alignment, unlike our proposed alignment network which outputs an approximate injective alignment. Here, RDKit requires a well-defined reaction template obtained through expert-curated rules and subsequent manual inspection of atom mapping for each reaction in the dataset. Consequently, we cannot generate such templates for SC datasets due to the high diversity of reactions. In Table 2, we report the results in terms of the MAE of the predicted yield. We make the following observations: (1) Our method outperforms the alternatives. (2) The attention network outperforms RXNMapper, RDKit and Random in most cases. However, since it induces a non-injective mapping, it assigns a product atom to multiple reactant atoms with high probability. As a result, our method outperforms it, since our alignment network provides the atom mapping through a doubly stochastic matrix, which in turn induces an injective mapping (one atom in R is mapped to exactly one atom I and vice-versa). (3) The random permutation generation strategy performs the worst. This underscores the necessity of good approximation of atom mapping.

Next, we evaluate the accuracy of our learned atom mapping by measuring its proximity to the ground-truth atom mapping available in the GP dataset— the only dataset that provides such ground-truth mappings. Specifically, we compute the average Frobenius norm of $||\boldsymbol{P} - \boldsymbol{P}^*||_F$ across all reactions, where $\boldsymbol{P}^*$ is the node permutation matrix representing the ground-truth atom mapping and $\boldsymbol{P}$ is the learned node alignment matrix. We compare our method with attention-based method (Astero & Rousu, 2024), which was the best alternative to our model, as shown in Table 2. Our method gives the average error using $||\boldsymbol{P}-\boldsymbol{P}^*||_F = \mathbf{3.708}$, whereas attenion based method gives $||\boldsymbol{P}-\boldsymbol{P}^*||_F = 5.096$. Notably, our method achieves such high accuracy in atom mapping prediction solely through end-to-end learning from yield data, without any access to the ground-truth atom mapping.

Interplay of atom mapping with predicted yield Here, we examine the extent to which the performance of YIELDNET can be attributed to its ability to learn the atom mapping. We investigate the relationship between the mismatch in prediction error ΔAE_r, and the difference between the true and learned atom mapping: $\Delta\boldsymbol{P}_r$ for each test reaction r as follows. We consider GP datasets since it is the only dataset containing atom mapping. Our approach begins by training a different variant of our model where we replace $\boldsymbol{P}_r$ for each reaction r with $\boldsymbol{P}^*_r$, *i.e.*, the permutation matrix corresponding to the ground truth atom mapping. We call this model as *reference model*. Then, for the test dataset, we gather all the learned alignments $\boldsymbol{P}_r$ from our trained model $\text{yield}_{\theta,\psi,\phi}$ and transform them into hard permutation matrices $\boldsymbol{P}_{\text{hard},r}$ via the Hungarian algorithm. Next, we apply both $\boldsymbol{P}^*_r$ and $\boldsymbol{P}_{\text{hard},r}$ to the reference model to obtain the predictions $\text{yield}_{\boldsymbol{P}^*_r}(r)$ and $\text{yield}_{\boldsymbol{P}_{\text{hard},r}}(r)$ for each reaction r in the test set. Note that $\text{yield}_{\boldsymbol{P}^*_r}$ gives high accuracy as we leak the true atom mapping into the model. Next, we calculate the difference in the absolute errors (AE) for each test reaction as $\Delta\text{AE}_r = |\text{AE}^*{}_r - \text{AE}_{\text{hard},r}|$ where $\text{AE}^*{}_r = |\text{yield}_{\boldsymbol{P}^*_r}(r) - \text{yield}(r)|$ and $\text{AE}_{\text{hard},r} = |\text{yield}_{\boldsymbol{P}=\boldsymbol{P}_{\text{hard},r}}(r) - \text{yield}(r)|$ for the gold yield values $\text{yield}(r)$. Finally, we quantify the dissimilarity between two permutation matrices using $\Delta\boldsymbol{P}_r = ||\boldsymbol{P}^*_r - \boldsymbol{P}_{\text{hard},r}||_F$

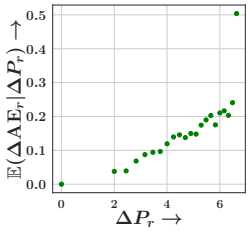

Figure 6: ΔAE_r vs $\Delta\boldsymbol{P}_r$

and plot the correlation between $\mathbb{E}(\Delta\text{AE}_r|\Delta\boldsymbol{P}_r)$ and $\Delta\boldsymbol{P}_r$, by averaging ΔAE_r at particular $\Delta\boldsymbol{P}_r$. Figure 6 shows a strong correlation between $\Delta\boldsymbol{P}_r$ and ΔAE_r, indicating a strong association between the learned alignments' quality and the model's predictive performance.

Ablation study on CGR representations Here, instead of incorporating the embeddings of the CGR into the reaction path encoder, we directly feed the embeddings of the reactants R_i and the products I_i into the encoder. Specifically, we omit both our node alignment network Align and CGR representation network $\text{InputDifferentiableGNN}$, and instead input the embeddings from the GNN into the transformer, alongside the step encoder. Table 3 summarizes the results. We observe that incorporating CGR embeddings into the reaction path encoder is more effective than directly utilizing the embeddings of the reactants and products.

Method	**NS1**	**NS2**	**NS3**	**SC**
w/o CGR	11.356	9.459	9.342	10.686
YIELDNET	9.245	8.387	7.914	8.751

Table 3: Effect of ablation of CGR (MAE).

Ablation study on reaction encoder components Here, we investigate the benefits offered by different components of our reaction path encoder (Eq. (16)—Eq. (20)). Specifically, we focus on ablations of its two key components, *viz.*, the transformer (17) and StepAggr (19). Table 4 summarizes the results for four multi-step reaction datasets measured in terms of MAE of the predicted yield. We make the following observations: (1) the transformer encoder contributes to the improved accuracy across most cases; (2) utilizing Seq2Seq for sets (Vinyals et al., 2016) yields better performance than DeepSet (Zaheer et al., 2017) and simple sum aggregation methods; (3) ablation on the set encoder has a stronger impact on MAE than the transformer because the set encoder is applied in the final layer.

Method	NS1	NS2	NS3	SC
$\text{StepAggr} = \text{DeepSet}$	9.643	9.008	8.283	8.551
$\text{StepAggr} = \text{SumAggr}$	10.192	8.858	8.349	8.853
Without transformer	9.467	8.680	8.164	8.628
YIELDNET	9.245	8.387	7.914	8.751

Table 4: Effect of Ablation on reaction path encoder components (MAE).

Effect of ablation on the regularizer Reg(R, I) Here, we evaluate the effect of the regularizer $\text{Reg}(R, I)$ in Eq. (21). In particular, we compare the results of our default regularized training against the scenario when ρ, the coefficient of $\text{Reg}(R, I)$, is set to zero. Table 5 shows the results in terms of MAE. We observe that the inclusion of regularization, i.e., the presence of constraints, benefits the yield prediction.

Method	**NS1**	**NS2**	**NS3**	**SC**
$\rho = 0$	9.929	8.590	8.035	8.810
YIELDNET	9.245	8.387	7.914	8.751

Table 5: Effect of ablation of regularizer $\text{Reg}(R, I)$ in Eq. (21) (MAE).

6. Conclusion

In this work, we present YIELDNET, a novel model for yield prediction of the final product in multi-step reactions. The energy of the transition state plays a critical role in determining reaction yield. The graph structure of the transition state can be inferred using atom mapping between the reactants and products. As most datasets lack atom mapping, we employ a differentiable node alignment network, which can be trained end-to-end using supervision from yield rather than fine-grained atom mapping labels.

Our work opens several promising directions for future research. YIELDNET can be used to predict free energy, reaction rates, and more. Yield depends on factors beyond activation energy, like reaction time, purification procedure, etc., which current datasets lack. While the datasets also lack the underlying atom mapping, the graph structures of the reactants and products allowed us to approximate it. A key direction for future work is to incorporate the other factors into our model when such datasets become available.

Impact Statement

This paper offers a valuable tool optimizing reaction yields. It can be used in prediction of yield when the datasets contain molecular graph structures. However, it is imperative to acknowledge the possibility of incorrect predictions by any yield prediction model, which demands human interventions. Additionally, misuse of the model may lead to predictions for undesirable compounds, potentially causing adverse societal consequences. Careful attention to ethical guidelines is paramount to ensure a responsible and positive impact on the model.

References

Ahneman, D. T., Estrada, J. G., Lin, S., Dreher, S. D., and Doyle, A. G. Predicting reaction performance in c–n cross-coupling using machine learning. *Science*, 360 (6385):186–190, 2018.

Anderson, E., Veith, G. D., and Weininger, D. *SMILES, a line notation and computerized interpreter for chemical structures*. US Environmental Protection Agency, Environmental Research Laboratory, 1987.

Anstreicher, K. M. Recent advances in the solution of quadratic assignment problems. *Mathematical Programming*, 97:27–42, 2003.

Arıcı, H., Sündü, B., Fırıncı, R., Ertuğrul, E., Özdemir, N., Cetinkaya, B., Metin, Ö., and Günay, M. E. The synthesis of new peppsi-type n-heterocyclic carbene (nhc)-pd (ii) complexes bearing long alkyl chain as precursors for the synthesis of nhc-stabilized pd (0) nanoparticles and

their catalytic applications. *Journal of Organometallic Chemistry*, 934:121633, 2021.

Arrhenius, S. Über die dissociationswärme und den einfluss der temperatur auf den dissociationsgrad der elektrolyte. *Zeitschrift für physikalische Chemie*, 4(1):96–116, 1889.

Astero, M. and Rousu, J. Learning symmetry-aware atom mapping in chemical reactions through deep graph matching. *Journal of Cheminformatics*, 16(1):46, 2024.

Axelrod, S. and Gomez-Bombarelli, R. Geom, energy-annotated molecular conformations for property prediction and molecular generation. *Scientific Data*, 9(1):185, 2022.

Azpiroz, R., Sharma, P., Pérez-Flores, F. J., Gutierrez, R., Espinosa-Pérez, G., and Lara-Ochoa, F. Stable ferrocenyl-nhc pd (ii) complexes: Evidence of ch- h/π interaction and mo bonding in solution. *Journal of Organometallic Chemistry*, 848:196–206, 2017.

Bai, Y., Ding, H., Bian, S., Chen, T., Sun, Y., and Wang, W. Simgnn: A neural network approach to fast graph similarity computation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pp. 384–392, 2019.

Benayad, Z., David, R., and Stirnemann, G. Prebiotic chemical reactivity in solution with quantum accuracy and microsecond sampling using neural network potentials. *Proceedings of the National Academy of Sciences*, 121 (23):e2322040121, 2024.

Bi, H., Wang, H., Shi, C., Coley, C., Tang, J., and Guo, H. Non-autoregressive electron redistribution modeling for reaction prediction. In *International Conference on Machine Learning*, pp. 904–913. PMLR, 2021.

Bjerrum, E. J. SMILES enumeration as data augmentation for neural network modeling of molecules. *arXiv preprint arXiv:1703.07076*, 2017.

Bradshaw, J., Kusner, M., Paige, B., Segler, M., and Hernández-Lobato, J. A generative model for electron paths. In *7th International Conference on Learning Representations, ICLR 2019*, volume 7. International Conference on Learning Representations (ICLR), 2019.

Burés, J. and Larrosa, I. Organic reaction mechanism classification using machine learning. *Nature*, 613(7945): 689–695, 2023.

Caetano, T. S., McAuley, J. J., Cheng, L., Le, Q. V., and Smola, A. J. Learning graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31 (6), 2009.

Çakır, S., Kavukcu, S. B., Karabıyık, H., Rethinam, S., and Türkmen, H. C (acyl)–c (sp 2) and c (sp 2)–c (sp 2) suzuki–miyaura cross-coupling reactions using nitrile-functionalized nhc palladium complexes. *RSC advances*, 11(60):37684–37699, 2021.

Cao, N. D. and Kipf, T. MolGAN: An implicit generative model for small molecular graphs, 2018.

Chen, B., Li, C., Dai, H., and Song, L. Retro*: learning retrosynthetic planning with neural guided a* search. In *International conference on machine learning*, pp. 1608–1616. PMLR, 2020.

Chen, M.-T. and Kao, Z.-L. Effect on orthometallation of nhc palladium complexes toward the catalytic activity studies in suzuki coupling reaction. *Dalton Transactions*, 46(47):16394–16398, 2017.

Chen, S. and Jung, Y. A generalized-template-based graph neural network for accurate organic reactivity prediction. *Nature Machine Intelligence*, 4(9):772–780, 2022.

Chen, S., An, S., Babazade, R., and Jung, Y. Precise atom-to-atom mapping for organic reactions via human-in-the-loop machine learning. *Nature Communications*, 15(1): 2250, 2024.

Chen, Z., Ayinde, O. R., Fuchs, J. R., Sun, H., and Ning, X. G 2 retro as a two-step graph generative models for retrosynthesis prediction. *Communications Chemistry*, 6 (1):102, 2023.

Chithrananda, S., Grand, G., and Ramsundar, B. Chemberta: Large-scale self-supervised pretraining for molecular property prediction. *arXiv preprint arXiv:2010.09885*, 2020.

Choi, S. Prediction of transition state structures of gas-phase chemical reactions via machine learning. *Nature Communications*, 14(1):1168, 2023.

Choi, S., Kim, Y., Kim, J. W., Kim, Z., and Kim, W. Y. Feasibility of activation energy prediction of gas-phase reactions by machine learning. *Chemistry–A European Journal*, 24(47):12354–12358, 2018.

Coley, C. W., Barzilay, R., Jaakkola, T. S., Green, W. H., and Jensen, K. F. Prediction of organic reaction outcomes using machine learning. *ACS central science*, 3(5):434–443, 2017a.

Coley, C. W., Rogers, L., Green, W. H., and Jensen, K. F. Computer-assisted retrosynthesis based on molecular similarity. *ACS central science*, 3(12):1237–1245, 2017b.

Coley, C. W., Jin, W., Rogers, L., Jamison, T. F., Jaakkola, T. S., Green, W. H., Barzilay, R., and Jensen, K. F. A

graph-convolutional neural network model for the prediction of chemical reactivity. *Chemical science*, 10(2): 370–377, 2019.

Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26:2292–2300, 2013.

Dai, H., Li, C., Coley, C., Dai, B., and Song, L. Retrosynthesis prediction with conditional graph logic network. *Advances in Neural Information Processing Systems*, 32, 2019.

Dastgir, S., Coleman, K. S., Cowley, A. R., and Green, M. L. Synthesis, structure, and temperature-dependent dynamics of neutral palladium allyl complexes of annulated diaminocarbenes and their catalytic application for c- c and c- n bond formation reactions. *Organometallics*, 29(21):4858–4870, 2010.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. pp. 4171–4186, 2019.

Diebolt, O., Jurcik, V., Correa da Costa, R., Braunstein, P., Cavallo, L., Nolan, S. P., Slawin, A. M., and Cazin, C. S. Mixed phosphite/n-heterocyclic carbene complexes: synthesis, characterization and catalytic studies. *Organometallics*, 29(6):1443–1450, 2010.

Du, J., Zhang, S., Wu, G., Moura, J. M., and Kar, S. Topology adaptive graph convolutional networks. *arXiv preprint arXiv:1710.10370*, 2017.

Duan, C., Du, Y., Jia, H., and Kulik, H. J. Accurate transition state generation with an object-aware equivariant elementary reaction diffusion model. *Nature Computational Science*, 3(12):1045–1055, 2023.

Duan, C., Liu, G.-H., Du, Y., Chen, T., Zhao, Q., Jia, H., Gomes, C. P., Theodorou, E. A., and Kulik, H. J. Optimal transport for generating transition states in chemical reactions. *Nature Machine Intelligence*, 7(4):615–626, 2025.

Eyring, H. The activated complex in chemical reactions. *The Journal of Chemical Physics*, 3(2):107–115, 1935.

Fey, M., Lenssen, J. E., Morris, C., Masci, J., and Kriege, N. M. Deep graph matching consensus. In *International Conference on Learning Representations*.

Fortunato, M. E., Coley, C. W., Barnes, B. C., and Jensen, K. F. Data augmentation and pretraining for template-based retrosynthetic prediction in computer-aided synthesis planning. *Journal of chemical information and modeling*, 60(7):3398–3407, 2020.

Fuchs, F. B., Worrall, D. E., Fischer, V., and Welling, M. Se(3)-transformers: 3d roto-translation equivariant attention networks. *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

Gao, H., Struble, T. J., Coley, C. W., Wang, Y., Green, W. H., and Jensen, K. F. Using machine learning to predict suitable conditions for organic reactions. *ACS central science*, 4(11):1465–1476, 2018.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *International conference on machine learning*, pp. 1263–1272. PMLR, 2017.

Gold, S. and Rangarajan, A. A graduated assignment algorithm for graph matching. *IEEE Transactions on pattern analysis and machine intelligence*, 18(4):377–388, 1996.

Gong, Y., Xue, D., Chuai, G., Yu, J., and Liu, Q. Deepreac+: deep active learning for quantitative modeling of organic chemical reactions. *Chemical Science*, 12(43):14459–14472, 2021.

Grambow, C. A., Pattanaik, L., and Green, W. H. Deep learning of activation energies. *The journal of physical chemistry letters*, 11(8):2992–2997, 2020a.

Grambow, C. A., Pattanaik, L., and Green, W. H. Reactants, products, and transition states of elementary chemical reactions based on quantum chemistry. *Scientific data*, 7 (1):137, 2020b.

Grover, A. and Leskovec, J. node2vec: Scalable feature learning for networks. In *SIGKDD*, 2016.

Guan, Y., Coley, C. W., Wu, H., Ranasinghe, D., Heid, E., Struble, T. J., Pattanaik, L., Green, W. H., and Jensen, K. F. Regio-selectivity prediction with a machine-learned reaction representation and on-the-fly quantum mechanical descriptors. *Chemical science*, 12(6):2198–2208, 2021.

Hagberg, A., Swart, P., and S Chult, D. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

Hamilton, W. L., Ying, R., and Leskovec, J. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 1025–1035, 2017.

Han, F., Xu, Y., Zhu, R., Liu, G., Chen, C., and Wang, J. Highly active nhc–pd (ii) complexes for cross coupling of aryl chlorides and arylboronic acids: an investigation of the effect of remote bulky groups. *New Journal of Chemistry*, 42(9):7422–7427, 2018.

Hartmann, C. E., Nolan, S. P., and Cazin, C. S. Highly active [pd (μ-cl)(cl)(nhc)] 2 (nhc= n-heterocyclic carbene) in the cross-coupling of grignard reagents with aryl chlorides. *Organometallics*, 28(9):2915–2919, 2009.

Heid, E. and Green, W. H. Machine learning of reaction properties via learned representations of the condensed graph of reaction. *Journal of Chemical Information and Modeling*, 62(9):2101–2110, 2021.

Hirohara, M., Saito, Y., Koda, Y., Sato, K., and Sakakibara, Y. Convolutional neural network based on SMILES representation of compounds for detecting chemical motif. *BMC bioinformatics*, 19(19):83–94, 2018.

Honda, S., Shi, S., and Ueda, H. R. Smiles transformer: Pre-trained molecular fingerprint for low data drug discovery. *arXiv preprint arXiv:1911.04738*, 2019.

Hoque, A., Das, M., Baranwal, M., and Sunoj, R. B. Reactaivate: A deep learning approach to predicting reaction mechanisms and unmasking reactivity hotspots. *arXiv preprint arXiv:2407.10090*, 2024a.

Hoque, A., Surve, M., Kalyanakrishnan, S., and Sunoj, R. B. Reinforcement learning for improving chemical reaction performance. *Journal of the American Chemical Society*, 146(41):28250–28267, 2024b.

Hu, Z., Dong, Y., Wang, K., and Sun, Y. Heterogeneous graph transformer. In *Proceedings of the web conference 2020*, pp. 2704–2710, 2020.

Huang, K., Fu, T., Glass, L. M., Zitnik, M., Xiao, C., and Sun, J. DeepPurpose: A deep learning library for drug-target interaction prediction. *Bioinformatics*, 2020.

Izquierdo, F., Corpet, M., and Nolan, S. P. The suzuki–miyaura reaction performed using a palladium–n-heterocyclic carbene catalyst and a weak inorganic base. *European Journal of Organic Chemistry*, 2015(9):1920–1924, 2015.

Jackson, R., Zhang, W., and Pearson, J. Tsnet: predicting transition state structures with tensor field networks and transfer learning. *Chemical Science*, 12(29):10022–10040, 2021.

Jain, E., Roy, I., Meher, S., Chakrabarti, S., and De, A. Graph edit distance with general costs using neural set divergence. *Advances in Neural Information Processing Systems*, 37:73399–73438, 2024.

Jin, W., Coley, C., Barzilay, R., and Jaakkola, T. Predicting organic reaction outcomes with weisfeiler-lehman network. *Advances in neural information processing systems*, 30, 2017.

Jin, W., Barzilay, R., and Jaakkola, T. Junction tree variational autoencoder for molecular graph generation. *ICML*, 2018.

Joung, J. F., Fong, M. H., Roh, J., Tu, Z., Bradshaw, J., and Coley, C. W. Reproducing reaction mechanisms with machine-learning models trained on a large-scale mechanistic dataset. *Angewandte Chemie International Edition*, 63(43):e202411296, 2024.

Kantchev, E. A. B., O'Brien, C. J., and Organ, M. G. Palladium complexes of n-heterocyclic carbenes as catalysts for cross-coupling reactions—a synthetic chemist's perspective. *Angewandte Chemie International Edition*, 46 (16):2768–2813, 2007.

Karataş, M. O. Cycloheptyl substituted n-heterocyclic carbene peppsi-type palladium complexes with different n-coordinated ligands: involvement in suzuki-miyaura reaction. *Journal of Organometallic Chemistry*, 899:120906, 2019.

Kearnes, S., McCloskey, K., Berndl, M., Pande, V., and Riley, P. Molecular graph convolutions: moving beyond fingerprints. *Journal of computer-aided molecular design*, 30(8):595–608, 2016.

Kim, S., Woo, J., and Kim, W. Y. Diffusion-based generative ai for exploring transition states from 2d molecular graphs. *Nature Communications*, 15(1):341, 2024.

King-Smith, E., Faber, F. A., Reilly, U., Sinitskiy, A. V., Yang, Q., Liu, B., Hyek, D., and Lee, A. A. Predictive minisci late stage functionalization with transfer learning. *Nature Communications*, 15(1):426, 2024.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

Klicpera, J., Giri, S., Margraf, J. T., and Günnemann, S. Fast and uncertainty-aware directional message passing for non-equilibrium molecules. *arXiv preprint arXiv:2011.14115*, 2020a.

Klicpera, J., Groß, J., Günnemann, S., et al. Directional message passing for molecular graphs. In *ICLR*, pp. 1–13, 2020b.

Körner, R. and Apostolakis, J. Automatic determination of reaction mappings and reaction center information. 1. the imaginary transition state energy approach. *Journal of chemical information and modeling*, 48(6):1181–1189, 2008.

Kozuch, S. and Shaik, S. How to conceptualize catalytic cycles? the energetic span model. *Accounts of Chemical Research*, 44(2):101–110, 2011.

Kriege, N. M., Humbeck, L., and Koch, O. Chemical similarity and substructure searches. In *Encyclopedia of Bioinformatics and Computational Biology*. Academic Press, 2019.

Kuhn, H. W. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

Kumar, S., Shaikh, M. M., and Ghosh, P. Palladium complexes of amido-functionalized n-heterocyclic carbenes as effective precatalysts for the suzuki–miyaura c–c cross-coupling reactions of aryl bromides and iodides. *Journal of Organometallic Chemistry*, 694(26):4162–4169, 2009.

Kuriyama, M., Matsuo, S., Shinozawa, M., and Onomura, O. Ether-imidazolium carbenes for suzuki–miyaura cross-coupling of heteroaryl chlorides with aryl/heteroarylboron reagents. *Organic letters*, 15(11): 2716–2719, 2013.

Kwon, Y., Lee, D., Choi, Y.-S., and Kang, S. Uncertainty-aware prediction of chemical reaction yields with graph neural networks. *Journal of Cheminformatics*, 14:1–10, 2022.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Landrum, G. Rdkit documentation. *Release*, 1(1-79):4, 2013.

Lemm, D., Von Rudorff, G. F., and Von Lilienfeld, O. A. Machine learning based energy-free structure predictions of molecules, transition states, and solids. *Nature Communications*, 12(1):4468, 2021.

Li, B., Su, S., Zhu, C., Lin, J., Hu, X., Su, L., Yu, Z., Liao, K., and Chen, H. A deep learning framework for accurate reaction prediction and its application on high-throughput experimentation data. *Journal of Cheminformatics*, 15 (1):72, 2023.

Li, D.-H., He, X.-X., Xu, C., Huang, F.-D., Liu, N., Shen, D.-S., and Liu, F.-S. N-heterocarbene palladium complexes with dianisole backbones: Synthesis, structure, and catalysis. *Organometallics*, 38(12):2539–2552, 2019a.

Li, X., Che, Y., Chen, L., Liu, T., Wang, K., Liu, L., Yang, H., Pyzer-Knapp, E. O., and Cooper, A. I. Sequential closed-loop bayesian optimization as a guide for organic molecular metallophotocatalyst formulation discovery. *Nature Chemistry*, pp. 1–9, 2024.

Li, Y., Gu, C., Dullien, T., Vinyals, O., and Kohli, P. Graph matching networks for learning the similarity of graph structured objects. In *International conference on machine learning*, pp. 3835–3845. PMLR, 2019b.

Liu, B., Ramsundar, B., Kawthekar, P., Shi, J., Gomes, J., Luu Nguyen, Q., Ho, S., Sloane, J., Wender, P., and Pande, V. Retrosynthetic reaction prediction using neural sequence-to-sequence models. *ACS central science*, 3 (10):1103–1113, 2017.

Liu, S., Alnammi, M., Ericksen, S. S., Voter, A. F., Ananiev, G. E., Keck, J. L., Hoffmann, F. M., Wildman, S. A., and Gitter, A. Practical model selection for prospective virtual screening. *Journal of chemical information and modeling*, 59(1):282–293, 2018.

Liu, S., Qu, M., Zhang, Z., Cai, H., and Tang, J. Multi-task learning with domain knowledge for molecular property prediction. In *NeurIPS 2021 AI for science workshop*, 2021.

Lowe, D. Chemical reactions from us patents (1976-sep2016). *Figshare Dataset*, 6084:m9, 2017.

Lu, D.-D., He, X.-X., and Liu, F.-S. Bulky yet flexible pd-peppsi-ipentan for the synthesis of sterically hindered biaryls in air. *The Journal of Organic Chemistry*, 82(20): 10898–10911, 2017.

Lu, J. and Zhang, Y. Unified deep learning model for multitask reaction predictions with explanation. *Journal of chemical information and modeling*, 62(6):1376–1387, 2022.

Lv, H., Zhu, L., Tang, Y.-Q., and Lu, J.-M. Structure–activity relationship of n-heterocyclic carbene–pd (ii)–imidazole complexes in suzuki–miyaura coupling between 4-methoxyphenyl chloride and phenylboronic acid. *Applied Organometallic Chemistry*, 28(1):27–31, 2014.

M. Bran, A., Cox, S., Schilter, O., Baldassari, C., White, A. D., and Schwaller, P. Augmenting large language models with chemistry tools. *Nature Machine Intelligence*, pp. 1–11, 2024.

Makoś, M. Z., Verma, N., Larson, E. C., Freindorf, M., and Kraka, E. Generative adversarial networks for transition state geometry prediction. *The Journal of Chemical Physics*, 155(2), 2021.

Mena, G., Belanger, D., Linderman, S., and Snoek, J. Learning latent permutations with gumbel-sinkhorn networks. In *International Conference on Learning Representations*, 2018.

Mercan, D., Cetinkaya, E., and Cetinkaya, B. Influence of ch3 substituents on tetrahydropyrimidin2ylidene: sigma donating properties and in situ catalytic activities of precursor salts/pd (oac) 2 system for c–c coupling reactions.

Journal of Organometallic Chemistry, 696(7):1359–1366, 2011.

Micksch, M., Tenne, M., and Strassner, T. Cyclometalated 2-phenylimidazole palladium carbene complexes in the catalytic suzuki–miyaura cross-coupling reaction. *Organometallics*, 33(15):3966–3976, 2014.

Navarro, O., Kaur, H., Mahjoor, P., and Nolan, S. P. Cross-coupling and dehalogenation reactions catalyzed by (n-heterocyclic carbene) pd (allyl) cl complexes. *The Journal of organic chemistry*, 69(9):3173–3180, 2004.

Navarro, O., Marion, N., Oonishi, Y., Kelly, R. A., and Nolan, S. P. Suzuki- miyaura, α-ketone arylation and dehalogenation reactions catalyzed by a versatile n-heterocyclic carbene- palladacycle complex. *The Journal of Organic Chemistry*, 71(2):685–692, 2006.

Nielsen, M. K., Ahneman, D. T., Riera, O., and Doyle, A. G. Deoxyfluorination with sulfonyl fluorides: navigating reaction space with machine learning. *Journal of the American Chemical Society*, 140(15):5004–5008, 2018.

Nippa, D. F., Atz, K., Müller, A. T., Wolfard, J., Isert, C., Binder, M., Scheidegger, O., Konrad, D. B., Grether, U., Martin, R. E., et al. Identifying opportunities for late-stage ch alkylation with high-throughput experimentation and in silico reaction screening. *Communications Chemistry*, 6(1):256, 2023.

Nippa, D. F., Atz, K., Hohler, R., Müller, A. T., Marx, A., Bartelmus, C., Wuitschik, G., Marzuoli, I., Jost, V., Wolfard, J., et al. Enabling late-stage drug diversification by high-throughput experimentation with geometric deep learning. *Nature Chemistry*, 16(2):239–248, 2024.

Nugmanov, R., Dyubankova, N., Gedich, A., and Wegner, J. K. Bidirectional graphormer for reactivity understanding: neural network trained to reaction atom-to-atom mapping task. *Journal of Chemical Information and Modeling*, 62(14):3307–3315, 2022.

Nugmanov, R. I., Mukhametgaleev, R. N., Akhmetshin, T., Gimadiev, T. R., Afonina, V. A., Madzhidov, T. I., and Varnek, A. Cgrtools: Python library for molecule, reaction, and condensed graph of reaction processing. *Journal of chemical information and modeling*, 59(6): 2516–2521, 2019.

O'Brien, C. J., Kantchev, E. A. B., Valente, C., Hadei, N., Chass, G. A., Lough, A., Hopkinson, A. C., and Organ, M. G. Easily prepared air-and moisture-stable pd–nhc (nhc= n-heterocyclic carbene) complexes: a reliable, user-friendly, highly active palladium precatalyst for the suzuki–miyaura reaction. *Chemistry–A European Journal*, 12(18):4743–4748, 2006.

Organ, M. G., Çalimsiz, S., Sayah, M., Hoi, K. H., and Lough, A. J. Pd-peppsi-ipent: An active, sterically demanding cross-coupling catalyst and its application in the synthesis of tetra-ortho-substituted biaryls. *Angewandte Chemie International Edition*, 48(13):2383–2387, 2009.

Ouyang, J.-S., Li, Y.-F., Huang, F.-D., Lu, D.-D., and Liu, F.-S. The highly efficient suzuki–miyaura cross-coupling of (hetero) aryl chlorides and (hetero) arylboronic acids catalyzed by "bulky-yet-flexible" palladium–peppsi complexes in air. *ChemCatChem*, 10(2):371–375, 2018.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Pattanaik, L., Ingraham, J. B., Grambow, C. A., and Green, W. H. Generating transition states of isomerization reactions with deep learning. *Physical Chemistry Chemical Physics*, 22(41):23618–23626, 2020.

Peh, G.-R., Kantchev, E. A. B., Er, J.-C., and Ying, J. Y. Rational exploration of n-heterocyclic carbene (nhc) palladacycle diversity: A highly active and versatile precatalyst for suzuki–miyaura coupling reactions of deactivated aryl and alkyl substrates. *Chemistry–A European Journal*, 16(13):4010–4017, 2010.

Pereira, O., Ruth, M., Gerbig, D., Wende, R. C., and Schreiner, P. R. Leveraging limited experimental data with machine learning: Differentiating a methyl from an ethyl group in the corey–bakshi–shibata reduction. *Journal of the American Chemical Society*, 2024.

Perozzi, B., Al-Rfou, R., and Skiena, S. Deepwalk: Online learning of social representations. In *KDD*, pp. 701–710, 2014.

Qian, Y., Guo, J., Tu, Z., Coley, C. W., and Barzilay, R. Rxnscribe: A sequence generation model for reaction diagram parsing. *Journal of Chemical Information and Modeling*, 63(13):4030–4041, 2023.

Qiao, Z., Welborn, M., Anandkumar, A., Manby, F. R., and Miller III, T. F. Orbnet: Deep learning for quantum chemistry using symmetry-adapted atomic-orbital features. *The Journal of chemical physics*, 153(12):124111, 2020.

Raccuglia, P., Elbert, K. C., Adler, P. D., Falk, C., Wenny, M. B., Mollo, A., Zeller, M., Friedler, S. A., Schrier, J., and Norquist, A. J. Machine-learning-assisted materials discovery using failed experiments. *Nature*, 533(7601): 73–76, 2016.

Raj, V., Roy, I., Ramachandran, A., Chakrabarti, S., and De, A. Charting the design space of neural graph representations for subgraph matching. In *The Thirteenth International Conference on Learning Representations*, 2025.

Ramachandran, A., Raj, V., Roy, I., Chakrabarti, S., and De, A. Iteratively refined early interaction alignment for subgraph matching based graph retrieval. In Globerson, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J., and Zhang, C. (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 77593–77629. Curran Associates, Inc., 2024.

Roy, I., De, A., and Chakrabarti, S. Adversarial permutation guided node representations for link prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 9445–9453, 2021.

Roy, I., Chakrabarti, S., and De, A. Maximum common subgraph guided graph retrieval: Late and early interaction networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022a.

Roy, I., Velugoti, V. S. B. R., Chakrabarti, S., and De, A. Interpretable neural subgraph matching for graph retrieval. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pp. 8115–8123, 2022b.

Saebi, M., Nan, B., Herr, J., Wahlers, J., Wiest, O., and Chawla, N. Graph neural networks for predicting chemical reaction performance. 2021.

Sahin, Z., Akkoç, S., İlhan, İ. Ö., and Kayser, V. Palladium n-heterocyclic carbene complexes: Synthesis from benzimidazolium salts and catalytic activity in carbon-carbon bond-forming reactions. *JoVE (Journal of Visualized Experiments)*, (125):e54932, 2017.

Samanta, B., De, A., Jana, G., Gómez, V., Chattaraj, P. K., Ganguly, N., and Gomez-Rodriguez, M. Nevae: A deep generative model for molecular graphs. *Journal of machine learning research. 2020 Apr; 21 (114): 1-33*, 2020.

Sandfort, F., Strieth-Kalthoff, F., Kühnemund, M., Beecks, C., and Glorius, F. A structure-based platform for predicting chemical reactivity. *Chem*, 6(6):1379–1390, 2020.

Scetbon, M., Cuturi, M., and Peyré, G. Low-rank sinkhorn factorization. In *International Conference on Machine Learning*, pp. 9344–9354. PMLR, 2021.

Schleinitz, J., Langevin, M., Smail, Y., Wehnert, B., Grimaud, L., and Vuilleumier, R. Machine learning yield prediction from nicolit, a small-size literature data set of nickel catalyzed c–o couplings. *Journal of the American Chemical Society*, 144(32):14722–14730, 2022.

Schütt, K. T., Sauceda, H. E., Kindermans, P.-J., Tkatchenko, A., and Müller, K.-R. Schnet–a deep learning architecture for molecules and materials. *The Journal of Chemical Physics*, 148(24):241722, 2018.

Schwaller, P., Gaudin, T., Lanyi, D., Bekas, C., and Laino, T. "found in translation": predicting outcomes of complex organic chemistry reactions using neural sequence-to-sequence models. *Chemical science*, 9(28):6091–6098, 2018.

Schwaller, P., Laino, T., Gaudin, T., Bolgar, P., Hunter, C. A., Bekas, C., and Lee, A. A. Molecular transformer: a model for uncertainty-calibrated chemical reaction prediction. *ACS central science*, 5(9):1572–1583, 2019.

Schwaller, P., Vaucher, A. C., Laino, T., and Reymond, J.-L. Data augmentation strategies to improve reaction yield predictions and estimate uncertainty. 2020.

Schwaller, P., Hoover, B., Reymond, J.-L., Strobelt, H., and Laino, T. Extraction of organic chemistry grammar from unsupervised learning of chemical reactions. *Science Advances*, 7(15):eabe4166, 2021a.

Schwaller, P., Probst, D., Vaucher, A. C., Nair, V. H., Kreutter, D., Laino, T., and Reymond, J.-L. Mapping the space of chemical reactions using attention-based neural networks. *Nature machine intelligence*, 3(2):144–152, 2021b.

Schwaller, P., Vaucher, A. C., Laino, T., and Reymond, J.-L. Prediction of chemical reaction yields using deep learning. *Machine learning: science and technology*, 2 (1):015016, 2021c.

Segler, M. H. and Waller, M. P. Neural-symbolic machine learning for retrosynthesis and reaction prediction. *Chemistry–A European Journal*, 23(25):5966–5971, 2017.

Sheldon, R. A., Brady, D., and Bode, M. L. The hitchhiker's guide to biocatalysis: recent advances in the use of enzymes in organic synthesis. *Chemical science*, 11(10): 2587–2605, 2020.

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

Shields, B. J., Stevens, J., Li, J., Parasram, M., Damani, F., Alvarado, J. I. M., Janey, J. M., Adams, R. P., and Doyle, A. G. Bayesian reaction optimization as a tool for chemical synthesis. *Nature*, 590(7844):89–96, 2021.

Singh, S. and Sunoj, R. B. A transfer learning protocol for chemical catalysis using a recurrent neural network

adapted from natural language processing. *Digital Discovery*, 1(3):303–312, 2022.

Somnath, V. R., Bunne, C., Coley, C., Krause, A., and Barzilay, R. Learning graph models for retrosynthesis prediction. *Advances in Neural Information Processing Systems*, 34:9405–9415, 2021.

Song, Y., Zheng, S., Niu, Z., Fu, Z.-H., Lu, Y., and Yang, Y. Communicative representation learning on attributed molecular graphs. In *IJCAI*, volume 2020, pp. 2831–2838, 2020.

Struble, T. J., Coley, C. W., and Jensen, K. F. Multitask prediction of site selectivity in aromatic c–h functionalization reactions. *Reaction Chemistry & Engineering*, 5 (5):896–902, 2020.

Tu, T., Sun, Z., Fang, W., Xu, M., and Zhou, Y. Robust acenaphthoimidazolylidene palladium complexes: highly efficient catalysts for suzuki–miyaura couplings with sterically hindered substrates. *Organic letters*, 14(16):4250–4253, 2012.

Tu, Z. and Coley, C. W. Permutation invariant graph-to-sequence model for template-free retrosynthesis and reaction prediction. *Journal of chemical information and modeling*, 62(15):3503–3513, 2022.

Varnek, A., Fourches, D., Hoonakker, F., and Solov'ev, V. P. Substructural fragments: an universal language to encode reactions, molecular and supramolecular structures. *Journal of computer-aided molecular design*, 19:693–703, 2005.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks. In *International Conference on Learning Representations*, 2018.

Villani, C. Optimal transport: Old and new, vol. 338: Springer science & business media, 2008.

Vinyals, O., Bengio, S., and Kudlur, M. Order matters: Sequence to sequence for sets. In *ICLR (Poster)*, 2016.

Wang, J. Y., Stevens, J. M., Kariofillis, S. K., Tom, M.-J., Golden, D. L., Li, J., Tabora, J. E., Parasram, M., Shields, B. J., Primer, D. N., et al. Identifying general reaction conditions by bandit optimization. *Nature*, 626(8001): 1025–1033, 2024.

Wang, M., Zheng, D., Ye, Z., Gan, Q., Li, M., Song, X., Zhou, J., Ma, C., Yu, L., Gai, Y., Xiao, T., He, T., Karypis, G., Li, J., and Zhang, Z. Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*, 2019a.

Wang, R., Yan, J., and Yang, X. Learning combinatorial embedding networks for deep graph matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3056–3065, 2019b.

Wang, S., Guo, Y., Wang, Y., Sun, H., and Huang, J. Smiles-bert: large scale unsupervised pre-training for molecular property prediction. In *Proceedings of the 10th ACM international conference on bioinformatics, computational biology and health informatics*, pp. 429–436, 2019c.

Wei, J. N., Duvenaud, D., and Aspuru-Guzik, A. Neural networks for the prediction of organic chemistry reactions. *ACS central science*, 2(10):725–732, 2016.

Weininger, D. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1):31–36, 1988.

Weisfeiler, B. and Leman, A. The reduction of a graph to canonical form and the algebra which appears therein. *nti, Series*, 2(9):12–16, 1968.

Wu, L., Drinkel, E., Gaggia, F., Capolicchio, S., Linden, A., Falivene, L., Cavallo, L., and Dorta, R. Room-temperature synthesis of tetra-ortho-substituted biaryls by nhc-catalyzed suzuki–miyaura couplings. *Chemistry–A European Journal*, 17(46):12886–12890, 2011.

Xia, Q., Shi, S., Gao, P., Lalancette, R., Szostak, R., and Szostak, M. [(nhc) pdcl2 (aniline)] complexes: Easily synthesized, highly active pd (ii)–nhc precatalysts for cross-coupling reactions. *The Journal of Organic Chemistry*, 86(21):15648–15657, 2021.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *International Conference on Learning Representations*.

Yan, C., Ding, Q., Zhao, P., Zheng, S., Yang, J., Yu, Y., and Huang, J. Retroxpert: Decompose retrosynthesis prediction like a chemist. *Advances in Neural Information Processing Systems*, 33:11248–11258, 2020.

Yang, J. Heteroleptic (n-heterocyclic carbene)–pd–pyrazole (indazole) complexes: Synthesis, characterization and catalytic activities towards c–c and c–n cross-coupling reactions. *Applied Organometallic Chemistry*, 31(10): e3734, 2017.

Yang, K., Swanson, K., Jin, W., Coley, C., Eiden, P., Gao, H., Guzman-Perez, A., Hopper, T., Kelley, B., Mathea, M., et al. Analyzing learned molecular representations for property prediction. *Journal of chemical information and modeling*, 59(8):3370–3388, 2019.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. Deep sets. *Advances in neural information processing systems*, 30, 2017.

Zahrt, A. F., Henle, J. J., Rose, B. T., Wang, Y., Darrow, W. T., and Denmark, S. E. Prediction of higher-selectivity catalysts by computer-driven workflow and machine learning. *Science*, 363(6424):eaau5631, 2019.

Zanfir, A. and Sminchisescu, C. Deep learning of graph matching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2684–2693, 2018.

Zhang, M. and Chen, Y. Link prediction based on graph neural networks. In *NeurIPS*, 2018.

Zhang, Y., Zhang, R., Ni, C., Zhang, X., Li, Y., Lu, Q., Zhao, Y., Han, F., Zeng, Y., and Liu, G. Nhc-pd (ii)-azole complexes catalyzed suzuki–miyaura cross-coupling of sterically hindered aryl chlorides with arylboronic acids. *Tetrahedron Letters*, 61(10):151541, 2020.

Zhou, Z., Li, X., and Zare, R. N. Optimizing chemical reactions with deep reinforcement learning. *ACS central science*, 3(12):1337–1344, 2017.

Learning Condensed Graph via Differentiable Atom Mapping for Reaction Yield Prediction (Appendix)

A. Limitations

1. Currently our work is limited to reactions having a reasonable number of atoms ($\sim 200-250$). However, implementing our model on larger molecular reactions or catalysis (Sheldon et al., 2020) has some drawbacks. Computation of $\boldsymbol{P}$ using Sinkhorn-iterations (5) has a complexity of $\mathcal{O}(N^2)$, where N is the number of total atoms of reactant(s) or product(s) in a reaction step. Thus, computing $\boldsymbol{P}$ in such cases would be challenging. However, this issue can be mitigated by low-rank factorization (Scetbon et al., 2021) for Sinkhorn routine, which reduces the complexity to $\mathcal{O}(N)$.
2. The standard yield prediction datasets currently have only a single reaction path r for reactions leads to one single product. Thus, we consider only single reaction path-based reactions to train our model. However, our method can be adapted to take multiple reactions into account, where we can add more stochasticity into Sinkhorn layers to enable generation of multiple permutations from the same set of reactants and products. They will lead to multiple atom mapping and CGRs, which in turn can generate multiple reaction paths.
3. The standard yield prediction datasets contain only molecular graphs of any molecular component and these datasets are commonly used in previous baselines. Hence, our model, as of now, does not incorporate information about geometry of the molecules. However, our model can be easily adapted to incorporate geometry by extending our GNN into the spatial graphs and taking into account three-dimensional coordinates for our atom mapping approximators.

B. Additional discussion on related work

Apart from the general applications in chemistry mentioned in Section 2, there has been a recent interest in modeling chemical reactions (Bradshaw et al., 2019; Benayad et al., 2024; Joung et al., 2024; Hoque et al., 2024a) and solving reaction-related problems (Schwaller et al., 2021b; Burés & Larrosa, 2023). These also include retrosynthetic planning (Segler & Waller, 2017; Liu et al., 2017; Coley et al., 2017b; Dai et al., 2019; Shi et al., 2020; Chen et al., 2020; Yan et al., 2020; Fortunato et al., 2020; Somnath et al., 2021; Tu & Coley, 2022; Chen et al., 2023), reaction outcome prediction (Wei et al., 2016; Coley et al., 2017a; Jin et al., 2017; Schwaller et al., 2018; 2019; Coley et al., 2019; Struble et al., 2020; Guan et al., 2021; Bi et al., 2021; Chen & Jung, 2022; Nippa et al., 2023; 2024; King-Smith et al., 2024; Pereira et al., 2024), reaction optimization (Raccuglia et al., 2016; Zhou et al., 2017; Gao et al., 2018; Shields et al., 2021; Wang et al., 2024; Li et al., 2024; Hoque et al., 2024b), atom mapping (Schwaller et al., 2021a; Nugmanov et al., 2022; Chen et al., 2024; Astero & Rousu, 2024), transition state geometry prediction (Pattanaik et al., 2020; Lemm et al., 2021; Makoś et al., 2021; Choi, 2023; Duan et al., 2023; Kim et al., 2024; Duan et al., 2025), etc (Qian et al., 2023; M. Bran et al., 2024). In the case of atom mapping, RXNMapper (Schwaller et al., 2021a), and GraphormerMapper (Nugmanov et al., 2022) are the two pioneers of deep learning-based atom-mapping tools. RXNMapper uses post-processing on the attention heads, learned through an unsupervised task of reaction SMILES reconstruction. Due to this unsupervised nature, RXNMapper needs a substantial amount of data and a large transformer encoder-based model (Lan et al.). On the other hand, GraphormerMapper is pre-trained in multiple levels on different tasks, making the model very computationally expensive. As both models use attention-guided paths for atom mapping, these mappings aren't injective *i.e.*, there are multiple atom mapping possible for one single reaction instance and that should not happen in real case scenarios. Recently, Chen et al. (2024) use active learning for the atom mapping task, which is also attention-based. In another recent work, AMNet (Astero & Rousu, 2024) makes the atom mapping injective by imposing an extra layer of WL test algorithm (Weisfeiler & Leman, 1968) on top of a non-injective graph attention. However, YIELDNET model – (1) tries to approximate atom mapping as a doubly-stochastic matrix, which by architecture yields injective mapping, (2) is relatively simple. Our key goal is to predict yield which is also different from the above described atom mapping models. We achieve this by designing an architecture, that approximates atom-mapping on the fly.

C. Neural architectures different components of YIELDNET

C.1. Architecture of GNN_θ

Our GNN_θ is built upon a communicative message passing network (CMPNN) (Song et al., 2020). In contrast to a simple message-passing neural network (MPNN) (Gilmer et al., 2017) or directed message-passing neural network (DMPNN) (Yang et al., 2019), where either only node states or edge states are updated independently in each iteration, our model updates both node and edge states simultaneously. Moreover, the update states for nodes and edges are treated as interdependent instead of treated independently. We elaborate on each step, starting from initial features as input to the final embedding as output.

In the following, we describe the embedding computation only for the reactant set R, which can subsequently be extended for the product set I. GNN_θ takes input graph G_R as $\mathrm{Adj}_R \in \{0,1\}^{N\times N}$, node features $\boldsymbol{V}_R \in \mathbb{R}^{N\times d_V}$, and edge features $\boldsymbol{E}_R \in \mathbb{R}^{N\times N\times d_E}$. We use two different MLPs for each of the features to transform into initial embeddings, $\mathrm{nodeEmb}_0(u) \in \mathbb{R}^D$ and $\mathrm{edgeEmb}_0(u,v) \in \mathbb{R}^D$ accordingly,

$$\mathrm{nodeEmb}_0(u) = \mathrm{MLP}_{\theta_1}(\boldsymbol{V}_R)[u] \tag{22}$$

$$\mathrm{edgeEmb}_0(u,v) = \mathrm{MLP}_{\theta_2}(\mathrm{Adj}_R \odot \boldsymbol{E}_R)[u,v] \tag{23}$$

Here, $\mathrm{MLP}_{\theta_\cdot}$ are just single-layer networks followed by a ReLU activation function. In Eq. (23), $\mathrm{Adj}_R \in \{0,1\}^{N\times N}$ and $\boldsymbol{E}_R \in \mathbb{R}^{N\times N\times d_E}$. Here, in $\odot$ operation, we first broadcast Adj_R into the third dimension to have an adjacency tensor of dimension $N \times N \times d_E$ and then perform Hadamard product with $\boldsymbol{E}_R$. Next, we perform message passing for k number of propagation layers. For a $k \in \{1,..,K-1\}$, the message passing steps are given by,

$$\boldsymbol{\mu}_k(v) = \sigma(\max_{u\in\mathrm{nbr}(v)} \mathrm{edgeEmb}_{k-1}(u,v)) \odot \textstyle\sum_{u\in\mathrm{nbr}(v)} \mathrm{edgeEmb}_{k-1}(u,v) \tag{24}$$

$$\mathrm{nodeEmb}_k(v) = \mathrm{nodeEmb}_{k-1}(v) + \boldsymbol{\mu}_k(v); \quad \mathrm{nodeEmb}_k(v) \in \mathbb{R}^D \tag{25}$$

$$\boldsymbol{\nu}_k(u,v) = \mathrm{Adj}_R[u,v] \cdot \Big(\mathrm{nodeEmb}_k(v) - \mathrm{edgeEmb}_{k-1}(v,u)\Big) \tag{26}$$

$$\mathrm{edgeEmb}_k(u,v) = \beta_k(u,v) \cdot \ \mathrm{ReLU}\Big(\mathrm{edgeEmb}_0(u,v) + \theta_{3,k} \cdot \boldsymbol{\nu}_k(u,v)\Big) \tag{27}$$

$$\text{where } \beta_k(u,v) = \frac{\exp\Big(\mathrm{Adj}_R[u,v] \cdot \mathrm{nodeEmb}_k(u)^\top \mathrm{nodeEmb}_k(v)\Big)}{\sum_{u'} \exp\Big(\mathrm{Adj}_R[u',v] \cdot \mathrm{nodeEmb}_k(u')^\top \mathrm{nodeEmb}_k(v)\Big)} \tag{28}$$

The above computations were performed for $k \in \{1,..K-1\}$. For $k = K$, we update the embeddings as follows:

$$\boldsymbol{\mu}_K(v) = \sigma(\max_{u\in\mathrm{nbr}(v)} \mathrm{edgeEmb}_{K-1}(u,v)) \cdot \textstyle\sum_{u\in\mathrm{nbr}(v)} \mathrm{edgeEmb}_{K-1}(u,v) \tag{29}$$

$$\mathrm{nodeEmb}_K(v) = \mathrm{MLP}_{\theta_4}\Big(\boldsymbol{h}_0(v), \mathrm{nodeEmb}_{K-1}(v), \boldsymbol{\mu}_K(v)\Big); \quad \mathrm{nodeEmb}_K(v) \in \mathbb{R}^d \tag{30}$$

$$\mathrm{edgeEmb}_K(u,v) = \mathrm{edgeEmb}_{K-1}(u,v); \quad \mathrm{edgeEmb}_K(v) \in \mathbb{R}^D \tag{31}$$

Finally, we compute node embedding vector $\boldsymbol{h}_R(u)$ per node u, node embedding matrix $\boldsymbol{H}_R \in \mathbb{R}^{N\times d}$, edge embedding vector $\boldsymbol{m}_R(u,v)$ for edge (u,v), edge embedding matrix $\boldsymbol{M}_R \in \mathbb{R}^{N\times N\times D}$, as follows:

$$\boldsymbol{h}_R(u) = \mathrm{nodeEmb}_K(u) \tag{32}$$

$$\boldsymbol{m}_R(u,v) = \mathrm{edgeEmb}_K(u,v) \tag{33}$$

$$\boldsymbol{H}_R[u,:] := \boldsymbol{h}_R(u) \tag{34}$$

$$\boldsymbol{M}_R[u,v,:] := \boldsymbol{m}_R(u,v) \tag{35}$$

C.2. Architecture of $\mathrm{InputDifferentiableGNN}_\psi$

This architecture is similar to the GNN_θ, except that we realize this in tensor space and use adjacency tensor $\mathrm{Adj}_{\mathrm{CGR}} \in \mathbb{R}^{N\times N\times D}$. This is because during the continuous approximation of $\mathrm{Adj}_{\mathrm{CGR}}$ in Eq. (12), we need $\boldsymbol{M}_R$, $\boldsymbol{M}_I \in \mathbb{R}^{N\times N\times D}$ (3). Thus, to obtain $\mathrm{Adj}_{\mathrm{CGR}}$, we need to do $\odot$ operation *i.e.*, broadcasting $\mathrm{Adj}_\bullet \in \mathbb{R}^{N\times N}$ into a third dimension to have an adjacency tensor of $N \times N \times D$ followed by Hadamard multiplication with corresponding $\boldsymbol{M}_\bullet$. Finally, we obtain an $\mathrm{Adj}_{\mathrm{CGR}} \in \mathbb{R}^{N\times N\times D}$ through Eq. (12). Now, similar to the GNN_θ, to perform the Eq. (23), we feed $\mathrm{Adj}_{\mathrm{CGR}}$ into an MLP

to bring its final dimension from D to d_E same as $\boldsymbol{E}$, so that the Hadamard product in Eq. (23) is compatible.

$$\widetilde{\text{Adj}}_{\text{CGR}} = \text{MLP}_{\psi_0}(\text{Adj}_{\text{CGR}}) \tag{36}$$

$$\text{nodeEmbT}_0 = \text{MLP}_{\psi_1}(\boldsymbol{V}_R) \tag{37}$$

$$\text{edgeEmbT}_0 = \text{MLP}_{\psi_2}(\widetilde{\text{Adj}}_{\text{CGR}} \odot \boldsymbol{E}_{\text{CGR}}) \tag{38}$$

Then the tensorized version for Eqs. (24)-(28) for $k \in \{1, ..., K-1\}$ is written as (xT indicates tensorized version of a vector $\boldsymbol{x}$ described in GNN_θ):

$$\mu\text{T}_k[v,:] = \sigma\left(\max_u \text{edgeEmbT}_{k-1}[u,v,:]\right) \odot \sum_u \text{edgeEmbT}_{k-1}[u,v,:] \tag{39}$$

$$\text{nodeEmbT}_k[v,:] = \text{nodeEmbT}_{k-1}[v,:] + \mu\text{T}_k[v,:] \tag{40}$$

$$\nu\text{T}_k[u,v,:] = \text{Adj}_{\text{CGR}}[u,v,:] \odot \text{nodeEmbT}_k[v,:] - \text{edgeEmbT}^\top_{k-1}[u,v,:] \tag{41}$$

$$\text{edgeEmbT}_k[u,v,:] = \beta_k[u,v,:] \odot \text{ ReLU}\Big(\text{edgeEmb}_0[u,v,:] + \psi_{3,k} \cdot \nu\text{T}_k[u,v,:]\Big) \tag{42}$$

$$\text{where } \beta\text{T}_k[u,v,:] = \frac{\exp\Big((\text{nodeEmbT}_k\, \text{nodeEmbT}_k^\top) \odot \text{Adj}_{\text{CGR}}\Big)[u,v,:]}{\sum_{u'} \exp\Big((\text{nodeEmbT}_k\, \text{nodeEmbT}_k^\top) \odot \text{Adj}_{\text{CGR}}\Big)[u',v,:]} \tag{43}$$

For $k = K$, we update the embeddings as:

$$\mu\text{T}_K[v,:] = \sigma(\max_u \text{edgeEmbT}_{K-1}[u,v,:]) \odot \sum_u \text{edgeEmbT}_{K-1}[u,v,:] \tag{44}$$

$$\text{nodeEmbT}_K[v,:] = \text{MLP}_{\psi_4}\Big(\text{nodeEmbT}_0[v,:], \text{nodeEmbT}_{K-1}[v,:], \mu\text{T}_K[v,:]\Big) \tag{45}$$

Finally, $\boldsymbol{H}_{\text{CGR}}$ is computed as $\text{nodeEmbT}_K \in \mathbb{R}^{N \times d_H}$.

C.3. Details about neural path encoder

Neural path encoder is composed of four components $\text{Transformer}_{\phi_1}$, NodeAggr_{ϕ_2}, StepAggr_{ϕ_3}, and MLP_{ϕ_4}.

1. $\text{Transformer}_{\phi_1}$ Before going to $\text{Transformer}_{\phi_1}$ encoder, $\boldsymbol{H}_{\text{CGR}_i}$ supposed to have a concatenation with $\mathbf{step_i}$, which is similar to the concept of positional encoding (Vaswani et al., 2017). Here, $i \leq n$, where n is the number of total steps in reaction path r. However, we used one-hot positional encoding, which leads us to $\mathbf{step_i} \in \{0,1\}^{N \times n}$. Thus, according to Eq. (16), $\overline{\boldsymbol{H}}_i \in \mathbb{R}^{N \times (d_H + n)}$. As our $\text{Transformer}_{\phi_1}$ is a single transformer encoder layer having the same encoder layer architecture as Vaswani et al. (2017), our final $\boldsymbol{S}_i \in \mathbb{R}^{N \times (d_H + n)}$ (17).
2. NodeAggr_{ϕ_2} The main architecture is the same as Seq2Seq for sets or known as Set2Set (Vinyals et al., 2016). This type of aggregation inflates the final output dimension of embeddings to twice the input dimension *i.e.* $2(d_H + n)$. Thus, we use a Linear layer to reduce back the output dimension to the same as the input dimension $(d_H + n)$. This breaks down the Eq. (18) into the following

$$\boldsymbol{s}_i = \text{Linear}_{\phi_{2b}}\Big(\text{Set2Set}_{\phi_{2a}}(\boldsymbol{S}_i[1,:], ..., \boldsymbol{S}_i[N,:])\Big), \quad \text{for } i \leq n. \tag{46}$$

3. StepAggr_{ϕ_3} This architecture is also similar to NodeAggr_{ϕ_2}. Thus, Eq. (19) can be rewritten as,

$$\boldsymbol{z}_r = \text{Linear}_{\phi_{3b}}\Big(\text{Set2Set}_{\phi_{3a}}(\boldsymbol{s}_1, ..., \boldsymbol{s}_n)\Big); \quad \boldsymbol{z}_r \in \mathbb{R}^{(d_H + n)} \tag{47}$$

4. MLP_{ϕ_4} The final MLP_{ϕ_4} layer is a $\text{Linear} - \text{ReLU} - \text{Linear}$ (LRL) layer, the input $\boldsymbol{z}_r$ has a final dimension of $(d_H + n)$.

D. Additional details about experimental setup

D.1. Dataset preparation

To assess the efficacy of our model, we focus on four reactions that have garnered recent attention. These reactions— Gas-phase isomerization (GP) (Grambow et al., 2020b), deoxyflurorination (DF) (Nielsen et al., 2018), asymmetric N,S-acetal formation (NS) (Zahrt et al., 2019), and Suzuki coupling reaction (SC)— exhibit diverse data sizes and varying number of steps.

Each reaction in these datasets is annotated with sets of reactants, intermediates, and products based on mechanistic considerations. The measured yields ranging from 0% to 100% are compiled from prior wet-lab experimental data, except in the case of the GP dataset, which we will discuss shortly. Several of these datasets, including (DF) (Nielsen et al., 2018) and asymmetric N, S-acetal formation (NS) (Zahrt et al., 2019), are obtained through High-Throughput Experimentation (HTE). The experimental process for HTE typically involves selecting a limited set of reactants and conducting reactions on *all possible combinations within those chosen reactants.* In contrast, real-life datasets contain a significantly larger pool of diverse reactants, with only a small fraction of the potential reactant combinations explored. This discrepancy results in a substantial reduction in dataset size for real-life scenarios (Saebi et al., 2021; Schleinitz et al., 2022). Additionally, real-life datasets exhibit higher sparsity, making it less likely to encounter a reactant from the test sample in the training set.

Due to the above issues, we create subsets of these HTE datasets, which create multiple low throughput (LTE) datasets, making them closer to reality. Specifically, we derive GP, GP1, GP2 from the gas-phase isomerization dataset; NS1, NS2, NS3, NS4, NS5 from the NS dataset; DF1, DF2 and DF3 from DF dataset. Moreover, we collect a new dataset consisting of samples of the reaction type Suzuki Coupling (SC) using the reaction data from ~ 50 peer reviewed publications. In contrast to the existing HTE datasets, this dataset is low throughput, *i.e.*, they consist of reactions of only a limited number of combinations between the reactants. We also use USPTO from Lowe (2017). These finally lead to total fifteen datasets.

D.1.1. List of datasets

The fifteen datasets used are: GP, GP1, GP2, USPTO, NS1, NS2, NS3, NS4, NS5, NS, DF1, DF2, DF3, DF, and SC. Among them, we present GP, NS1, NS2, NS3, NS4, NS5, SC and DF in the main part and present the rest in this Appendix. In the following part, we describe the datasets in full detail.

D.1.2. Datasets derived from gas phase isomerization

The gas-phase isomerization reaction dataset (Grambow et al., 2020b) contains around 12000 elementary step organic reactions involving small molecules composed solely of C, H, N and O atoms. The dataset has the activation energy for the reaction as the label. Previous attempts to predict activation energies involved using fingerprints and graph-based models (Choi et al., 2018; Grambow et al., 2020a; Heid & Green, 2021). We transformed the activation energy ($\Delta\mathcal{E}^{\ddagger}$) into yield values, by using the Eyring equation (Eyring, 1935). This approximates a rate constant (γ) value from activation energy. We then assume every elementary step reaction following first-order kinetics to get the yield value. The steps are shown below. Notations used in Eqs. (48)— (51) are only limited to this section and may share overlap with symbols used in other parts of the paper. Here, $\Delta G^{\ddagger}$ represents Gibbs free energy of activation (or assumed to be activation energy) in Jmol^{-1}. k_B, h, R are the Boltzmann constant (1.38×10^{-23} JK^{-1}), Planck constant (6.626×10^{-34}Js), and universal gas constant (8.314 JKmol^{-1}) respectively. Here, t here is the reaction time, we take a fix t of 1 hour i.e., 3600s.

$$\gamma = \frac{k_B \text{Temp}}{h} \exp\left(\frac{-\Delta G^{\ddagger}}{R \cdot \text{Temp}}\right) \tag{48}$$

$$\text{yield} = (1 - \exp(-\gamma t)) \times 100 \tag{49}$$

Temp is the reaction temperature. In our case, we consider a fixed Temp $= 1105.26$K for our GP dataset. We have activation energy in this dataset in terms of kcal mol^{-1}. To convert it to a proper unit, we multiply our activation energy value by a factor of 4183 and feed the values in place of $\Delta G^{\ddagger}$. After all these, our final equation becomes,

$$\gamma = \frac{1.38 \times 10^{-23} \times 1105.26}{6.626 \times 10^{-34}} \exp\left(\frac{-\Delta\mathcal{E}^{\ddagger} \times 4183}{8.314 \times 1105.26}\right) \tag{50}$$

$$\text{yield} = (1 - \exp(-\gamma \times 3600)) \times 100 \tag{51}$$

We derive GP and its other two variants GP1 and GP2 based on clustering of the gas-phase reaction dataset. We observe that the reactants in GP1 mostly contain very small, 3 or 4-membered rings (e.g., cyclopropanes, oxiranes, aziridines,

cyclobutanes, etc.). Similarly, reactants in GP2 are mostly substituted *five-membered aromatic heterocycles* (e.g., pyrrole, pyrazole, imidazole, furan, etc.). In total, we have used three datasets GP (main), GP1 (Appendix), and GP2 (Appendix).

D.1.3. DATASETS DERIVED FROM DEOXYFLUORINATION

The deoxyfluorination (DF) reaction is an important method for converting inexpensive alcohol to the corresponding fluorinated compounds by using sulfonyl fluorides, in the presence of a suitable base (Nielsen et al., 2018). Here, the whole dataset represents a collection of combination reactions involving 37 unique alcohols, 5 sulfonyl fluoride, and 4 bases, totaling 740 reactions and their measured yields. Previous studies on yield prediction for this reaction employed the random forest (Nielsen et al., 2018) and NLP-based transfer learning (Singh & Sunoj, 2022). Apart from the full dataset (DF), we have extracted three subsets, each containing 200 instances, from the full dataset. We create these subsets based on the yield labels. Yields have a skewed distribution, where yield has a mean around 40. We create three subsets having an average yield of low (10), medium (75) and high (90). We find that there are different predominant compounds. For instance:

(1) The dataset DF1 has 2-(4-hydroxycyclohexyl)isoindoline-1,3-dione, 2,3,4,6,7,8,9,10-octahydropyrimido[1,2-a]azepine, 4-nitro benzenesulfonyl fluoride as predominant reactants.

(2) The dataset DF2 has 3-([1,2,4]triazolo[1,5-a]pyrimidin-6-yl)propan-1-ol, 2-(tert-butyl)-1,1,3,3- tetramethylguanidine, 1,1,2,2,3,3,4,4,4-nonafluorobutane -1-sulfonyl fluoride predominant reactants.

(3) The dataset DF3 has 3-(4,5-diphenyloxazol-2-yl)propan-1-ol, 2-(tert-butyl)-1,1,3,3- tetramethylguanidine, 1,1,2,2,3,3,4,4,4-nonafluorobutane -1-sulfonyl fluoride as predominant reactants.

In total, we have used four datasets DF (main), DF1 (Appendix), DF2 (Appendix), and DF3 (Appendix).

D.1.4. DATASETS DERIVED FROM ASYMMETRIC N, S-ACETAL FORMATION

The asymmetric N, S-acetal formation reaction (NS) is a catalytic transformation involving the addition of a thiol to aldimines. Zahrt et al. (2019) released a dataset comprising 1075 such examples, where the labels are in terms of activation barrier difference. Later, Singh & Sunoj (2022) released another version of the prior dataset, which comprises 1027 samples having labels as percentage values. This dataset consists of 5 thiols, 15 different imines as substrates, and 43 different chiral phosphoric acids as catalysts (Cat). The label for this dataset is expressed in terms of enantioselectivity, which is effectively the yield of the major enantiomer produced in the reaction. Similarly to the previous dataset, we have sampled the complete dataset into five subsets of size 300. Like DF, we create datasets with average yield of 25, 40, 50, 60, 75.

(1) The dataset NS1 has 2,6-dibromo-4-hydroxy-8,9,10,11,12,13,14,15-octahydrodinaphtho[2,1-d:1',2'-f][1,3,2] dioxaphosphepine-4-oxide; (E)-N-(2,4-dichlorobenzylidene)benzamide; ethanethiol as predominant reactants.

(2) The dataset NS2 has 2,6bis(anthracen-9-ylmethyl)-4hydroxydinaphtho[2,1-d:1',2'-f][1,3,2] dioxaphosphepine-4-oxide; (E)-N-benzylidenebenzamide; cyclohexanethiol as predominant reactants.

(3) The dataset NS3 has 2,6-bis(anthracen-9-ylmethyl)-4-hydroxydinaphtho[2,1-d:1',2'-f][1,3,2] dioxaphosphepine-4-oxide; (E)-N-(naphthalen-1-ylmethylene)benzamide; cyclohexanethiol as predominant reactants.

(4) The dataset NS4 has 4-hydroxy-2,6-bis(2-(naphthalen-2-yl)phenyl)-8,9,10,11,12,13,14,15-octahydrodinaphtho [2,1-d:1',2'-f][1,3,2]dioxaphosphepine-4-oxide; (E)-N-(4-methoxybenzylidene)benzamide; cyclohexanethiol as predominant reactants.

(5) The dataset NS5 has 4-hydroxy-2,6-bis(4-methoxyphenyl) dinaphtho[2,1-d:1',2'-f][1,3,2] dioxaphosphepine-4-oxide; (E)-N-(4-methoxybenzylidene)benzamide; 2-methylbenzenethiol} as predominant reactants.

We also utilized the full dataset (abbreviated as NS) to check the performance. In total, we have six datasets NS1 (main), NS2 (main), NS3 (main), NS4 (main), and NS5 (main), and NS (Appendix).

D.1.5. DATASETS DERIVED FROM SUZUKI COUPLING REACTION

The palladium-catalyzed Suzuki Cross-coupling (SC) is a versatile method for generating biaryl products—ubiquitous in natural compounds, pharmaceuticals, and chiral ligands (Kantchev et al., 2007). This dataset comprising 481 reactions is manually curated from 25 publications filtered out from numerous reports within this reaction class (Çakır et al., 2021; O'Brien et al., 2006; Navarro et al., 2006; Li et al., 2019a; Peh et al., 2010; Chen & Kao, 2017; Lu et al., 2017; Micksch

et al., 2014; Diebolt et al., 2010; Tu et al., 2012; Ouyang et al., 2018; Han et al., 2018; Zhang et al., 2020; Karataş, 2019; Yang, 2017; Lv et al., 2014; Organ et al., 2009; Xia et al., 2021; Navarro et al., 2004; Wu et al., 2011; Sahin et al., 2017; Kuriyama et al., 2013; Hartmann et al., 2009; Dastgir et al., 2010; Arıcı et al., 2021; Azpiroz et al., 2017; Kumar et al., 2009; Mercan et al., 2011; Izquierdo et al., 2015). The selection criteria were specifically tailored to a type of catalyst where palladium is intricately bound to an N-heterocyclic carbene (NHC) ligand, with additional selection criteria involving heteroatoms (N, O, and P) within the catalyst structure. The dataset is labeled based on the yield of the resulting biaryl product. We used the SC dataset in our main paper.

D.1.6. Datasets derived from USPTO

This is a US patent chemical reaction dataset collected by Lowe (2017). Later the dataset is filtered to classify the reaction classes (Schwaller et al., 2021b). The reaction dataset has around 44k reactions with yield labels and reaction classes. We use these reaction classes to filter out the dataset. With the further exclusion of reactions with missing reagents, we selected 1700 reaction samples from the original dataset. Next, we select only those reactions which occur through a two-step reaction pathway. For instance, the protection of functional groups (e.g., alcohol, amines), functional group interconversion, including nucleophilic substitution, etc. Finally, we curated 1150 two-step reaction samples for our study. We used the USPTO dataset in the Appendix.

D.2. Details about our implementations of the baselines

While implementing the baselines, we performed few modifications to ensure that the comparison between our method and the baselines is fair— (1) number of parameters is approximately same; and (2) no component of a baseline is exposed/pre-trained in external dataset, since that would give additional signals to them, which is not provided to our method and rest of the baselines.

GCN Here in GNN, the message function employs the graph convolution layer (GCN) (Gilmer et al., 2017), which collects adjacency node embeddings through edges. We have developed other versions of it by replacing this GCN with alternative node-based convolution message functions. Finally, following Kwon et al. (2022), we predict the yield by taking molecular graphs of reactants and products as input. It aggregates reactant vectors, concatenates them with the product vector, and channels them through an FNN to predict the yield. We set the dimension of node embeddings as 18.

HGT Here in GNN, we utilize a Heterogeneous Graph Transformer-based Convolution (HGT) as our message function to treat the molecular graph as a heterogeneous graph (Hu et al., 2020). Then, we used the same method proposed by Kwon et al. (2022), which is described in the context of GCN. We set the dimension of node embeddings as 17.

TAG Here in GNN, we utilize TAG, which is based on the topology adaptive graph convolution network (Du et al., 2017). Then, we used the same method proposed by Kwon et al. (2022), which is described in the context of GCN. We set the dimension of node embeddings as 20.

GIN Here in GNN, we employ the GIN message convolution function, inspired by graph isomorphism networks, showcasing a Graph Neural Network (GNN) that exhibits comparable power to the Weisfeiler-Lehman (WL) test for isomorphism (Xu et al.). Then, we used the same method proposed by Kwon et al. (2022), which is described in the context of GCN. We set the dimension of node embeddings as 20.

DeepReac+ The overall architecture of the (Gong et al., 2021) model is founded on a two-level graph attention convolution framework (Veličković et al., 2018). In the initial stage, the graphs for each reaction component are encoded using a graph attention network (GAT)-based GNN. Later, these individual component graphs are treated as nodes within a fully connected reaction graph, using the graph embeddings serving as node features. Another layer of GAT is then applied to this complete graph. Finally, the embeddings of all components are fed into a FFN network. We set the dimension of node embeddings as 48.

YieldBERT YieldBERT uses a previously trained BERT encoder (Devlin et al., 2019) to forecast chemical reaction yield as a function of reaction SMILES (Weininger, 1988). But that would expose the model to external datasets possibly containing examples in test data too. Hence, we train BERT encoder along with a regressor, end-to-end for yield prediction. We set the dimension of hidden size as 12.

D.3. Implementation details

Training We maintain a 70:10:20 split for training, validation, and testing across all models, employing 10 different splits for robust evaluation. We kept the batch size b same across all models. We set $b = 50$ for GP datasets and $b = 8$ for the rest. We train each model with 100 epochs and evaluate their performance on the test dataset, selecting the epoch with the lowest validation MAE. We use Adam optimizer for each model. We use Noam learning rate with 2 warmup epochs and an initial and final learning rate of 10^{-4} and a maximum learning rate of 10^{-3}. Additionally, we keep the regularizer parameter ρ fixed at a value of 0.1 in our model.

Model

1. GNN_θ. We set the dimension of node embeddings, $d = 20$ (30) and dimension of edge embeddings, $D = 20$ (31).
2. Align. We set temperature $\lambda = 0.1$ in Eq. (6), Gumbel noise factor 1.0, and Sinkhorn iterations $T = 10$.
3. $\text{InputDifferentiableGNN}_\psi$. It shares parameters with GNN_θ except ψ_0 and ψ_4 (Appendix C). We set $d_H = 20 - n$, where n is the number of steps.
4. $\text{Transformer}_{\phi_1}$. As $d_H = 20 - n \implies (d_H + n) = 20$, which is the input and output dimension of the $\text{Transformer}_{\phi_1}$. We set the number of heads to 5, and the feedforward dimension to 2048.
5. NodeAggr_{ϕ_2} and StepAggr_{ϕ_3}. As described in the Appendix C, the set encoders of Eqs. (46) and (47) of both the aggregator modules are borrowed off-the-shelf from Vinyals et al. (2016), where we have input dimension $(d_H + n) = 20$, number of layers 1, number of iterations 3. The final output of the Aggr modules will be of the dimension of $(d_H + n) = 20$.
6. MLP_{ϕ_4}. We have input dimension of $(d_H + n) = 20$ and output dimension of 1 as per Eq. (20).

Baseline For a fair comparison, we maintain a similar number of parameters across all the models. We used the same learning rate for all GNN models and YieldBERT of 10^{-4}. For DeepReac+, the learning rate is 10^{-3}.

Number of parameters The number of parameters for multi-step reactions is around 112k, and almost 26k for a single step one. The number of parameters is almost maintained throughout all the models (Table 6).

#Parameters	GP	rest
GCN	26825	115753
HGT	28405	151895
TAG	27965	108725
GIN	27165	105525
DeepReac+	27074	119522
YieldBERT	28861	110797
YIELDNET	26478	112583

Table 6: Number of parameters for all models

D.4. Hardware details

All the models are trained on NVIDIA A100 80GB GPU. All the models are fully based on PyTorch (Paszke et al., 2019). We run in Ubuntu 20.04.6 LTS machine having 2TB RAM with 64 bit CPU and AMD EPYC 7742 64-Core Processor.

D.5. License

We utilize CMPNN (Song et al., 2020), which comes under MIT License. For baseline comparisons, we use dgl-based (Wang et al., 2019a) GCN (Kwon et al., 2022), HGT (Hu et al., 2020), TAG (Du et al., 2017), GIN (Xu et al.), and DeepReac+ (Gong et al., 2021) - all of which come under the Apache 2.0 License. YieldBERT (Schwaller et al., 2021b) comes under MIT License. The Gas-Phase reaction datasets (Grambow et al., 2020b) are licensed under CC-BY 4.0. The USPTO dataset used here comes under the MIT License. The original USPTO dataset (Schwaller et al., 2021b) by Lowe (2017) comes under CC0 1.0 License. RXNMapper (Schwaller et al., 2021a) tool comes under MIT License. RDKit tool comes under BSD-3-Clause License. NS (Zahrt et al., 2019) and DF (Nielsen et al., 2018) datasets used here, are available in (Singh & Sunoj, 2022). However, we couldn't find licenses for those datasets. PyTorch (Paszke et al., 2019) and NetworkX (Hagberg et al., 2008) both come under BSD-3-Clause License.

E. Additional results

E.1. Comparison in yield prediction in terms of RMSE

From Table 7 we observe a similar trend as Table 1 in RMSE for seven out of eight datasets. A different trend in RMSE for GP is due to using MAE on the validation set during early stopping. The metric used for early stopping was MAE in the validation set, which is why it may show an alternate trend for RMSE in GP. Like Table 1, YIELDNET (sky) (skyline variants of our model, which uses true CGR for yield prediction) outperforms other models, reflecting CGR's importance in yield prediction.

Model	GP	NS1	NS2	NS3	NS4	NS5	SC	DF
GCN	42.778 ± 0.276	15.080 ± 1.151	12.090 ± 1.011	10.438 ± 0.864*	11.667 ± 0.286	6.631 ± 0.532	16.314 ± 0.762	16.245 ± 0.392
HGT	43.442 ± 0.181	17.550 ± 1.038	12.241 ± 1.012	11.222 ± 0.758	11.639 ± 0.334	6.805 ± 0.552	19.136 ± 0.612	23.358 ± 0.195
TAG	41.698 ± 0.336	17.470 ± 0.972	12.212 ± 0.993	10.953 ± 0.798	11.575 ± 0.318	6.766 ± 0.563	18.792 ± 0.661	22.177 ± 0.473
GIN	42.712 ± 0.351	17.449 ± 1.052	12.285 ± 1.011	10.760 ± 0.799	11.424 ± 0.290	6.782 ± 0.552	18.467 ± 0.635	21.320 ± 0.428
DeepReac+	35.748 ± 0.375	16.349 ± 1.474	13.048 ± 0.699	12.041 ± 0.960	12.526 ± 0.921	6.992 ± 0.391*	19.852 ± 1.604	17.188 ± 1.680
YieldBERT	44.733 ± 0.260	16.438 ± 1.153	12.090 ± 0.992	10.955 ± 0.894*	11.971 ± 0.342	6.803 ± 0.557	16.104 ± 0.620	15.993 ± 0.384
YIELDNET	38.195 ± 0.436	12.415 ± 0.866	10.701 ± 0.966	9.702 ± 0.969	9.383 ± 0.491	6.507 ± 0.535	14.029 ± 0.795	9.231 ± 0.258
YIELDNET(sky)	31.598 ± 0.448	NA	NA	NA	NA	NA	NA	NA

Table 7: Comparison of yield prediction performance for YIELDNET against all the competitive baselines, *viz.*, GCN (Kipf & Welling, 2017), HGT (Hu et al., 2020), TAG (Du et al., 2017), GIN (Xu et al.), DeepReac+ (Gong et al., 2021), YieldBERT (Schwaller et al., 2021c), on the 20% test examples, across all datasets. Performance is measured in terms of Root Mean Squared Error (RMSE). Numbers in green (yellow) indicate the best (second best) performer. Our improvement in performance over the next best baseline, where YIELDNET is the best performer, is statistically significant with p-value < 0.05, except in the cases marked with *.

E.2. Statistical significance test

To check the statistical significance of our results, we perform the paired t-test between YIELDNET and each baseline. We report the p-value of MAE and RMSEs for Table 1 and Table 7 in Table 8. If the p-value between the performance metrics is lower than the 5e-2 margin, we consider the corresponding performance significant.

p-value for MAE table

Model	GP	NS1	NS2	NS3	NS4	NS5	SC	DF
GCN	1.5e-10	3.6e-03	7.7e-03	2.7e-02	1.6e-02	3.7e-01*	1.6e-05	1.1e-07
HGT	5.1e-11	3.1e-05	1.7e-03	2.3e-03	2.5e-02	3.7e-02	1.0e-07	5.5e-12
TAG	2.1e-09	1.7e-05	1.6e-03	4.6e-03	2.7e-02	3.9e-02	3.3e-06	4.2e-10
GIN	1.0e-09	2.8e-05	1.8e-03	5.8e-03	2.0e-02	1.1e-02	7.8e-07	4.8e-09
DeepReac+	3.3e-06	3.7e-03	2.4e-03	6.5e-02*	1.8e-02	9.3e-02*	1.0e-03	6.5e-03
YieldBERT	9.6e-11	1.8e-04	4.3e-04	9.8e-02*	1.5e-02	4.9e-02	1.0e-03	3.9e-07

p-value for RMSE table

Model	GP	NS1	NS2	NS3	NS4	NS5	SC	DF
GCN	3.7e-06	4.1e-03	3.5e-03	9.2e-02*	4.8e-03	3.1e-02	3.0e-04	8.2e-08
HGT	1.6e-06	9.8e-06	1.9e-03	6.0e-03	5.5e-03	8.3e-03	4.1e-06	4.6e-12
TAG	1.5e-04	8.7e-06	1.9e-03	1.5e-02	5.8e-03	2.7e-02	1.1e-05	2.8e-10
GIN	6.7e-06	1.3e-05	1.7e-03	2.5e-02	7.9e-03	2.5e-02	6.9e-06	6.5e-10
DeepReac+	3.2e-02	4.9e-03	3.1e-03	2.6e-02	7.9e-03	1.7e-01*	2.1e-02	4.6e-03
YieldBERT	3.0e-07	1.9e-04	1.0e-03	5.2e-02*	6.5e-03	3.1e-02	2.0e-03	7.7e-08

Table 8: p-value for MAE and RMSEs for all baseline models compared to our model. Our improvement in performance over the next best baseline, where YIELDNET is the best performer, is statistically significant with p-value < 0.05, except the cases marked with *.

E.3. Additional results on comparison across various atom-to-atom alignment methods

Here, we report the MAE and RMSE with standard error for all atom-to-atom alignment methods introduced in Section 5.2. Table 9 and 10 report the performances, which reveal that our alignment method is the best performer in the majority of the cases. Due to diversity in the reaction for GP and SC, they don't follow a common reaction template, thus atom-mapping using RDKit aren't feasible for them.

Mean Absolute Error (MAE)

Model	GP	NS1	NS2	NS3
RDKit	–	9.910 ± 0.630	8.845 ± 0.873	8.567 ± 0.931
RXNMapper	20.604 ± 0.433	9.610 ± 0.602	8.871 ± 0.871	8.470 ± 0.970
Random	25.187 ± 0.407	11.195 ± 0.890	8.984 ± 0.993	8.695 ± 0.796
Attention	25.730 ± 0.758	9.548 ± 0.414	9.024 ± 0.894	8.432 ± 0.934
YIELDNET	23.152 ± 0.393	9.245 ± 0.518	8.387 ± 0.907	7.914 ± 0.931

Mean Absolute Error (MAE)

Model	NS4	NS5	SC	DF
RDKit	7.237 ± 0.375	4.533 ± 0.246	–	7.187 ± 0.208
RXNMapper	6.871 ± 0.406	4.677 ± 0.315	10.046 ± 0.434	6.879 ± 0.173
Random	7.991 ± 0.256	4.504 ± 0.243	10.223 ± 0.474	10.595 ± 0.244
Attention	6.955 ± 0.259	4.317 ± 0.220	8.826 ± 0.399	7.593 ± 0.220
YIELDNET	7.015 ± 0.495	4.382 ± 0.249	8.751 ± 0.438	6.941 ± 0.192

Table 9: Comparison between different alignment strategies in terms of MAE of yield prediction. Brief results were presented in Table 2. Numbers in green (yellow) indicate the best (second best) performer.

Root Mean Squared Error (RMSE)

Model	GP	NS1	NS2	NS3
RDKit	–	13.133 ± 0.802	11.381 ± 0.910	10.451 ± 0.912
RXNMapper	35.429 ± 0.455	12.769 ± 0.854	11.366 ± 0.852	10.365 ± 1.002
Random	37.428 ± 0.482	14.827 ± 1.185	11.467 ± 0.998	10.852 ± 0.772
Attention	39.513 ± 0.659	12.703 ± 0.713	11.484 ± 0.933	10.500 ± 0.933
YIELDNET	38.195 ± 0.436	12.415 ± 0.866	10.701 ± 0.966	9.702 ± 0.969

Root Mean Squared Error (RMSE)

Model	NS4	NS5	SC	DF
RDKit	9.972 ± 0.599	6.732 ± 0.557	–	9.770 ± 0.334
RXNMapper	9.483 ± 0.740	6.984 ± 0.684	15.486 ± 0.773	9.202 ± 0.221
Random	10.908 ± 0.421	6.644 ± 0.562	15.419 ± 0.839	14.098 ± 0.324
Attention	9.531 ± 0.421	6.423 ± 0.565	13.719 ± 0.710	10.533 ± 0.382
YIELDNET	9.383 ± 0.491	6.507 ± 0.535	14.029 ± 0.795	9.231 ± 0.258

Table 10: Comparison between different alignment strategies in terms of RMSE of yield prediction. Brief results were presented in Table 2. Numbers in green (yellow) indicate the best (second best) performer.

E.4. Additional results on ablation study on CGR representations

Here, we present the MAE and RMSE values for the ablation study on CGR computation introduced in Section 5.2, which extends the results of Table 3. From Table 11 we observe that the influence of CGR computation is benefitting on the yield prediction task for almost all of the datasets.

E.5. Additional results on ablation study of reaction path encoder components

Next, we present the MAE and RMSE values for the ablation study on components of the reaction encoder for all the datasets (Section 5.2), which extends the results of Table 4. From Table 12 we can observe that the influence of the SetEncoders is way more significant than that of the Transformer. Since the GP is a single-step reaction dataset, these ablation studies are not applicable to it.

Mean Absolute Error (MAE)

Method	GP	NS1	NS2	NS3	NS4	NS5	SC	DF
w/o CGR	24.921 ± 0.427	11.356 ± 0.702	9.459 ± 0.953	9.342 ± 0.822	7.726 ± 0.428	4.495 ± 0.270	10.686 ± 0.453	9.064 ± 0.242
YIELDNET	23.152 ± 0.393	9.245 ± 0.518	8.387 ± 0.907	7.914 ± 0.931	7.015 ± 0.495	4.382 ± 0.249	8.751 ± 0.438	6.941 ± 0.192

Root Mean Squared Error (RMSE)

Method	GP	NS1	NS2	NS3	NS4	NS5	SC	DF
w/o CGR	36.686 ± 0.353	15.249 ± 0.881	12.075 ± 0.949	11.316 ± 0.775	10.506 ± 0.54	6.679 ± 0.585	16.134 ± 0.748	11.827 ± 0.276
YIELDNET	38.195 ± 0.436	12.415 ± 0.866	10.701 ± 0.966	9.702 ± 0.969	9.383 ± 0.491	6.507 ± 0.535	14.029 ± 0.795	9.231 ± 0.258

Table 11: Ablation study CGR computation. Brief results were introduced in Table 3. Numbers in green indicate the best performer.

Mean Absolute Error (MAE)

Model	NS1	NS2	NS3	NS4	NS5	SC	DF
StepAggr = DeepSet	9.643 ± 0.703	9.008 ± 0.915	8.283 ± 0.948	6.124 ± 0.202	4.395 ± 0.253	8.551 ± 0.292	6.712 ± 0.264
StepAggr = SumAggr	10.192 ± 0.743	8.858 ± 0.806	8.349 ± 0.909	6.937 ± 0.490	4.433 ± 0.262	8.853 ± 0.309	7.165 ± 0.183
Without transformer	9.467 ± 0.516	8.680 ± 0.786	8.164 ± 0.849	6.407 ± 0.224	4.461 ± 0.252	8.628 ± 0.504	6.448 ± 0.140
YIELDNET	9.245 ± 0.518	8.387 ± 0.907	7.914 ± 0.931	7.015 ± 0.495	4.382 ± 0.249	8.751 ± 0.438	6.941 ± 0.192

Root Mean Squared Error (RMSE)

Model	NS1	NS2	NS3	NS4	NS5	SC	DF
StepAggr = DeepSet	12.493 ± 0.930	11.594 ± 0.912	10.102 ± 0.928	8.297 ± 0.250	6.451 ± 0.543	13.117 ± 0.584	8.924 ± 0.309
StepAggr = SumAggr	13.341 ± 0.998	11.413 ± 0.916	10.078 ± 0.910	9.222 ± 0.596	6.62 ± 0.615	13.442 ± 0.551	9.531 ± 0.223
Without transformer	12.407 ± 0.727	10.871 ± 0.845	10.055 ± 0.878	8.900 ± 0.424	6.577 ± 0.577	13.485 ± 0.777	8.761 ± 0.182
YIELDNET	12.415 ± 0.866	10.701 ± 0.966	9.702 ± 0.969	9.383 ± 0.491	6.507 ± 0.535	14.029 ± 0.795	9.231 ± 0.258

Table 12: Ablation study on different components of the reaction encoder. Brief results were introduced in Table 4. Numbers in green (yellow) indicate the best (second best) performer.

E.6. Additional results on ablation study on regularizer

In this section, we present the MAE and RMSE values for the ablation study on regularizer $\text{Reg}(R, I)$ by taking the hyperparameter $\rho = 0$ (Section 5.2), which extends the results of Table 5. From Table 13 summarizes that the addition of the regularizer improves the performance in almost all of the cases.

Mean Absolute Error (MAE)

Method	GP	NS1	NS2	NS3	NS4	NS5	SC	DF
$\rho = 0$	23.650 ± 0.465	9.929 ± 0.627	8.590 ± 0.885	8.035 ± 0.903	7.162 ± 0.168	4.397 ± 0.243	8.810 ± 0.308	8.959 ± 0.704
YIELDNET	23.152 ± 0.393	9.245 ± 0.518	8.387 ± 0.907	7.914 ± 0.931	7.015 ± 0.495	4.382 ± 0.249	8.751 ± 0.438	6.941 ± 0.192

Root Mean Squared Error (RMSE)

Method	GP	NS1	NS2	NS3	NS4	NS5	SC	DF
$\rho = 0$	39.666 ± 0.494	13.228 ± 0.820	10.892 ± 0.940	9.919 ± 0.943	9.815 ± 0.313	6.524 ± 0.547	13.351 ± 0.540	11.942 ± 0.822
YIELDNET	38.195 ± 0.436	12.415 ± 0.866	10.701 ± 0.966	9.702 ± 0.969	9.383 ± 0.491	6.507 ± 0.535	14.029 ± 0.795	9.231 ± 0.258

Table 13: Ablation study on regularizer $\text{Reg}(R, I)$ in Eq. (21). Brief results were introduced in Table 5. Numbers in green indicate the best performer.

E.7. Comparison between different ways to compute alignment matrix

Here, we compare an alternative way to construct alignment matrix $\boldsymbol{P}$ mentioned in Eq. (5) by choosing $\boldsymbol{C}[u, v] = \boldsymbol{h}_R(u)^\top \boldsymbol{h}_I(v)$ (referred as $\boldsymbol{C}_{\text{dot}}$) to our chosen one, $\boldsymbol{C}[u, v] = -\sum_{\ell=1}^{d} \max\left(\boldsymbol{h}_R(u), \boldsymbol{h}_I(v)\right)[\ell]$ (referred as $\boldsymbol{C}_{\max}$). From Table 14 we observe almost similar performances for both the C matrices.

Mean Absolute Error (MAE)

Method	GP	NS1	NS2	NS3	NS4	NS5	SC	DF
$\boldsymbol{C}_{\text{dot}}$	22.735 ± 0.543	8.709 ± 0.416	8.654 ± 0.885	8.121 ± 0.896	6.606 ± 0.247	4.365 ± 0.267	8.481 ± 0.330	6.964 ± 0.132
$\boldsymbol{C}_{\max}$	23.152 ± 0.393	9.245 ± 0.518	8.387 ± 0.907	7.914 ± 0.931	7.015 ± 0.495	4.382 ± 0.249	8.751 ± 0.438	6.941 ± 0.192

Root Mean Squared Error (RMSE)

Method	GP	NS1	NS2	NS3	NS4	NS5	SC	DF
$\boldsymbol{C}_{\text{dot}}$	37.639 ± 0.639	11.459 ± 0.599	11.010 ± 0.938	10.002 ± 0.916	9.167 ± 0.463	6.514 ± 0.618	12.984 ± 0.601	9.322 ± 0.165
$\boldsymbol{C}_{\max}$	38.195 ± 0.436	12.415 ± 0.866	10.701 ± 0.966	9.702 ± 0.969	9.383 ± 0.491	6.507 ± 0.535	14.029 ± 0.795	9.231 ± 0.258

Table 14: Performance of YIELDNET with different ways to calculate $\boldsymbol{P}$ mentioned in Eq. (5)

E.8. Comparison against competitive baselines with additional datasets

Mean Absolute Error (MAE)

Model	GP1	GP2	USPTO	DF1	DF2	DF3	NS
GCN	34.771 ± 0.399	37.316 ± 0.309	34.336 ± 0.345*	9.610 ± 0.692*	10.098 ± 0.775	10.451 ± 0.823	6.542 ± 0.139*
HGT	38.345 ± 0.241	41.798 ± 0.276	34.767 ± 0.320	11.246 ± 0.843	10.721 ± 0.889	10.989 ± 0.916	8.608 ± 0.218
TAG	36.504 ± 0.292	39.757 ± 0.239	34.693 ± 0.321	11.200 ± 0.774	10.553 ± 0.919	10.877 ± 0.919	9.083 ± 0.132
GIN	36.184 ± 0.184	39.008 ± 0.280	34.896 ± 0.304	11.169 ± 0.839	10.373 ± 0.871	10.610 ± 0.805	8.883 ± 0.155
DeepReac+	29.700 ± 0.210	32.577 ± 0.402	36.890 ± 0.994	12.302 ± 1.693*	10.665 ± 0.928	10.091 ± 0.689	9.450 ± 0.905
YieldBERT	39.587 ± 0.278	44.072 ± 0.280	35.352 ± 0.356	10.126 ± 0.822*	9.912 ± 0.925	10.475 ± 0.656	7.837 ± 0.154
YIELDNET	24.269 ± 0.610	27.563 ± 0.618	33.460 ± 0.474	8.832 ± 0.747	8.552 ± 0.894	8.815 ± 0.384	6.278 ± 0.194

Root Mean Squared Error (RMSE)

Model	GP1	GP2	USPTO	DF1	DF2	DF3	NS
GCN	40.945 ± 0.347	42.132 ± 0.300*	38.257 ± 0.383*	13.165 ± 0.963*	12.065 ± 0.688	13.523 ± 0.842	9.055 ± 0.214*
HGT	42.667 ± 0.209	44.633 ± 0.239	37.913 ± 0.374*	14.38 ± 1.072	12.575 ± 0.819	14.172 ± 0.836	11.454 ± 0.251
TAG	41.551 ± 0.221	43.536 ± 0.317*	37.970 ± 0.397*	14.274 ± 1.011	12.602 ± 0.831	14.054 ± 0.835	11.974 ± 0.185
GIN	41.836 ± 0.226	42.983 ± 0.257*	38.006 ± 0.385*	14.399 ± 1.071	12.296 ± 0.784	13.741 ± 0.748	11.764 ± 0.225
DeepReac+	37.576 ± 0.241*	40.961 ± 0.371	42.876 ± 1.157*	15.618 ± 1.724*	13.136 ± 0.942	12.902 ± 0.839*	12.028 ± 0.917
YieldBERT	44.099 ± 0.165	46.380 ± 0.220	38.446 ± 0.396*	13.669 ± 1.146*	12.110 ± 0.822	13.529 ± 0.726	10.678 ± 0.217
YIELDNET	38.862 ± 0.622	42.852 ± 0.659	41.191 ± 1.112	12.181 ± 0.903	10.790 ± 0.865	11.813 ± 0.478	8.777 ± 0.310

Table 15: Comparison of yield prediction performance for YIELDNET against all the competitive baselines, *viz.*, GCN (Kipf & Welling, 2017), HGT (Hu et al., 2020), TAG (Du et al., 2017), GIN (Xu et al.), DeepReac+ (Gong et al., 2021), YieldBERT (Schwaller et al., 2021c), on the 20% test examples, across the additional datasets. Performance is measured in terms of MAE (top half) and RMSE (bottom half). Numbers in green (yellow) indicate the best (second best) performer. Numbers with * indicate that either the model outperforms YIELDNET in that case or the difference is not statistically significant (i.e. p-value > 0.05). The results reveal the same observations as in Table 1.

In this section, we present the MAE and RMSE for all the methods on the additional datasets *viz.*, NS4, NS5, DF1, DF2, DF3, and NS. Table 15 summarizes the results, which shows that our method outperforms the baselines, similar to Table 1. For the rest of the datasets YIELDNET outperforms the baselines by a significant margin in most cases. We observe the results are more prominent in the case of MAE as compared to RMSE.

E.9. Results on approximating the gold permutation for additional datasets

We compute $||\boldsymbol{P} - \boldsymbol{P}^*||_F$ (mentioned in Section 5.2) for attention and YIELDNET for two additional GP1 and GP2 datasets

Model	GP	GP1	GP2
Attention	5.096	4.611	4.496
YIELDNET	3.708	3.627	3.165

Table 16: Comparison of $||\boldsymbol{P} - \boldsymbol{P}^*||_F$.

as true labels of permutations for Gas-phase reaction datasets are available. Table 16 summarizes that the atom mapping approximation in YIELDNET is superior to that of attention.

E.10. Results on the interplay between the quality of learned alignment and predicted yield for additional datasets

We plot $\mathbb{E}(\Delta\text{AE}_r|\Delta\boldsymbol{P}_r)$ vs $\Delta\boldsymbol{P}_r$ same as Figure 6 for additional GP1 and GP2 datasets in Figure 7. This shows a similar trend i.e. high correlation between error in yield prediction and atom-to-atom alignment for GP1 and GP2 datasets.

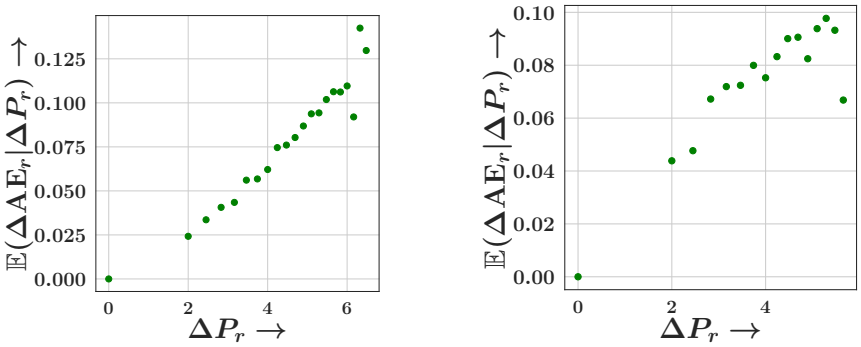

Figure 7: $\mathbb{E}(\Delta\text{AE}_r|\Delta\boldsymbol{P}_r)$ vs $\Delta\boldsymbol{P}_r$ plot for **GP1** (left) and **GP2** (right) dataset

E.11. Visualization of condensed graphs where they show the exact transition states

Condensed graphs serve as surrogates of the transition states. We are visualizing some cases for the GP dataset, where they reflect the exact transition state. We select the reactions with low MAE values and collect the alignment matrix, $\boldsymbol{P}$ corresponding to those reactions. Then, we apply the Hungarian algorithm to get hard permutation matrices, $\boldsymbol{P}_{\text{hard}}$. We obtain the Adj_{CGR} as follows, $\text{Adj}_{\text{CGR}} = \max(\text{Adj}_R, \boldsymbol{P}_{\text{hard},r}\text{Adj}_I\boldsymbol{P}_{\text{hard},r}^{\top})$. Finally from the connections obtained from the Adj_{CGR}, we draw the molecular structures of condensed graphs as shown in Figure 8.

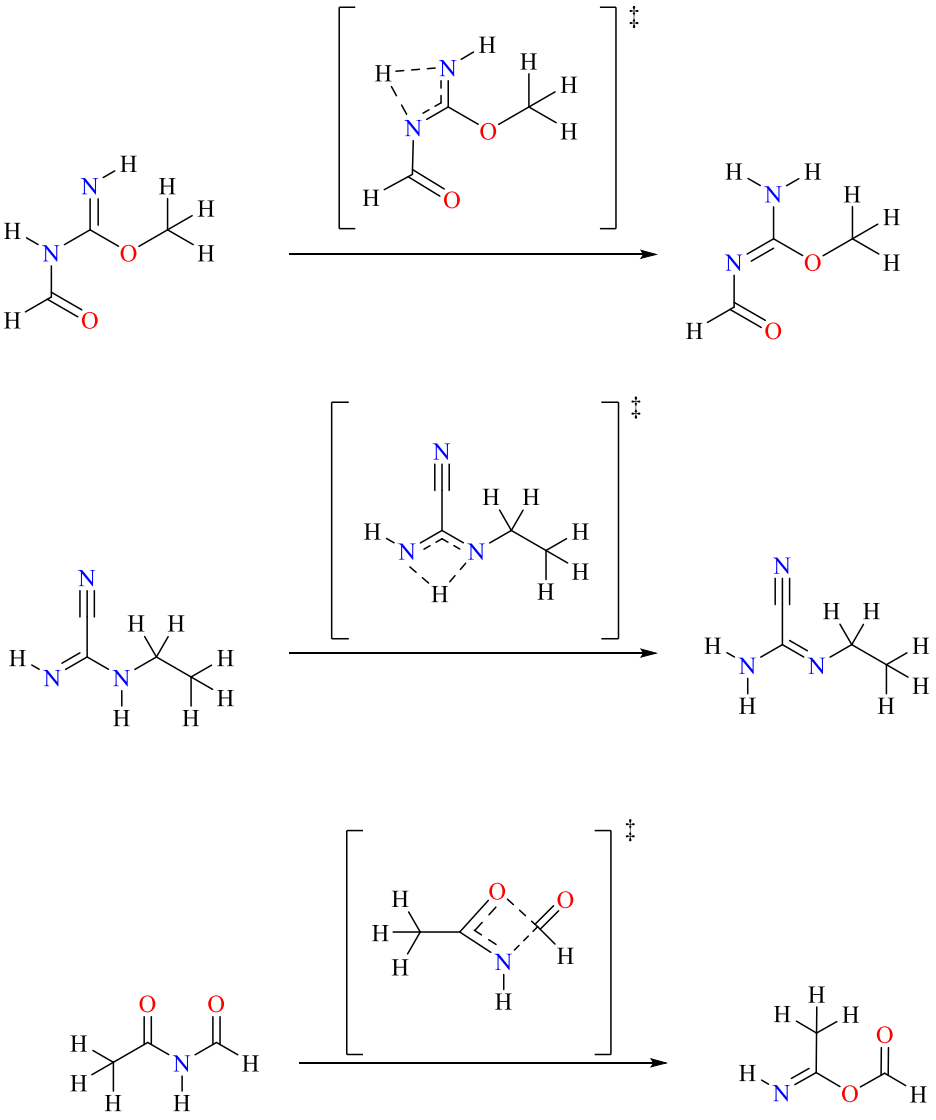

Figure 8: Representative examples of the condensed graphs where they show the exact transition states, (shown on the top of the arrow in each case) as generated by YIELDNET during yield prediction.

We use NetworkX (Hagberg et al., 2008) to get the connections between atoms from Adj_{CGR}. After obtaining those NetworkX raw graphs we manually draw figures of Figure 8 in ChemDraw for better visualization, maintaining the connectivity obtained through the NetworkX graphs.

E.12. Time complexity

We also report the inference time for the NS3 dataset with the test size of 60 and batch size of 8. From Table 17, it is clear that inference time for our model is comparable with the some of the GNN based models (HGT and TAG) and not as high as YieldBERT.

Model	Inference time in sec
GCN	1.727 ± 0.010
HGT	4.684 ± 0.087
TAG	3.109 ± 0.023
GIN	1.693 ± 0.020
DeepReac+	0.900 ± 0.026
YieldBERT	84.983 ± 0.186
YIELDNET	3.440 ± 0.298

Table 17: Inference time for different methods