Appendix: Molecular Hypergraph Grammar with Its Application to Molecular Optimization

Hiroshi Kajino [1]

Algorithm 1 Production rule extraction

Input: tree decomposition T and node $v_T \in V_T$.
Output: production rule $p = (A, R)$.

Find $\mathrm{pa}(v_T)$ and $\mathrm{ch}(v_T)$.
if $\mathrm{pa}(v_T)$ does not exist **then**
  Set A as the starting hyperedge S.
else
  Set $A = \ell_T^{(V)}(v_T) \cap \ell_T^{(V)}(\mathrm{pa}(v_T))$.
Set $R = H(v_T)$, where the nodes shared with A are set to be the external nodes, $\mathrm{ext}(R)$.
if $\mathrm{ch}(v_T)$ is not empty **then**
  for each child $v'_T \in \mathrm{ch}(v_T)$ **do**
    Add a non-terminal hyperedge to R, which consists of nodes $\ell_T^{(V)}(v_T) \cap \ell_T^{(V)}(v'_T)$ with the non-terminal label.
return (A, R)

A. HRG Inference Algorithm

This section describes the details of the algorithm proposed by Aguiñaga et al. (2016).

The input of the algorithm is a set of hypergraphs, $\hat{\mathcal{H}} = \{H_1, \ldots, H_N\}$, and its output is a hyperedge replacement grammar $\mathcal{G}$ whose language includes the input hypergraphs. The algorithm extracts production rules from each input hypergraph. Let $H \in \hat{\mathcal{H}}$ be any input hypergraph. It first computes a tree decomposition of H, which we denote by T, and picks an arbitrary node of T as its root. Then, for each node $v_T \in V_T$, it applies Algorithm 1 to extract a production rule. Algorithm 1 chooses the triplet of v_T and its parent and children as shown in Fig. 1a, and extracts a production rule that glues $H(v_T)$ to $H(\mathrm{pa}(v_T))$ with non-terminal hyperedges for gluing each child. Figure 1b illustrates the production rule extracted from the triplet shown in Fig. 1a. After applying Algorithm 1 to all of the nodes and all of the input hypergraphs, it removes duplicated production rules and outputs the set of production rules.

[1]MIT-IBM Watson AI Lab; IBM Research, Tokyo, Japan. Correspondence to: Hiroshi Kajino <kajino@jp.ibm.com>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

B. Proofs

This section provides a proof of Theorem 2. Throughout this section, a *node* refers to a hypergraph's node, and a *tree node* refers to a tree decomposition's node.

To prove Theorem 2, we only have to prove that $\mathcal{L}_{\mathsf{HRG}}(\hat{\mathcal{H}}) \subseteq \mathcal{H}(L_H^{(V)}, L_H^{(E)}, c^{(E)})$ holds, because $\hat{\mathcal{H}} \subseteq \mathcal{L}_{\mathsf{HRG}}(\hat{\mathcal{H}})$ has been proven by Aguiñaga et al. (2016).

$\mathcal{L}_{\mathsf{HRG}}(\hat{\mathcal{H}}) \subseteq \mathcal{H}(L_H^{(V)}, L_H^{(E)}, c^{(E)})$ can be proven by combining Lemmata 2, 3, and 4. Lemmata 2 and 3 guarantee that our algorithm preserves the first condition of a molecular hypergraph, and Lemma 4 guarantees that our algorithm preserves the second condition of a molecular hypergraph.

Condition 1. *For each production rule $p = (A, R)$, the degree of external nodes of R is one, and the degree of internal nodes of R is two.*

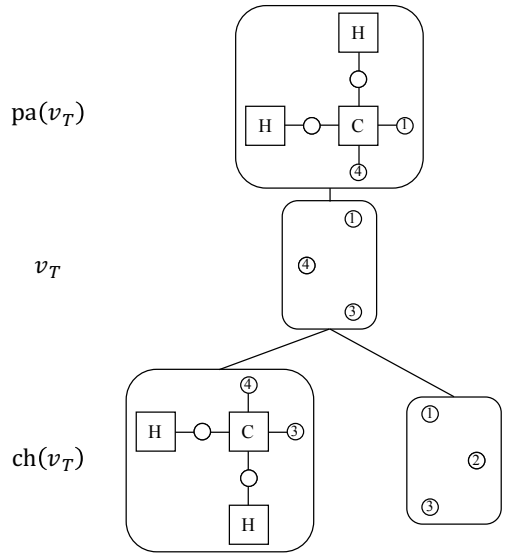
Lemma 2. *HRG inferred by applying our algorithm to a set of 2-regular hypergraphs satisfies Condition 1.*

Lemma 3. *If HRG satisfies Condition 1, then it always generates a 2-regular hypergraph.*

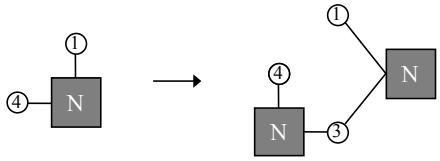
Lemma 4. *HRG inferred by applying our algorithm to a set of cardinality-consistent hypergraphs always generates a cardinality-consistent hypergraph.*

Proof of Lemma 2. Let H be an arbitrary input hypergraph, and $(T, \ell_T^{(V)}, \ell_T^{(E)})$ be its irredundant tree decomposition. Since H is 2-regular, for each $v_H \in V_H$, $T[V_T(v_H)]$ is a single tree node that contains both of the two hyperedges (Fig. 2a, **Case 1**), or $T[V_T(v_H)]$ is a path where each of the leaf tree nodes contains one of two hyperedges adjacent to v_H (Fig. 2b, **Cases 2, 3**). It is sufficient to prove that for each $v_H \in V_H$ and for each $v_T \in T[V_T(v_H)]$, the production rule extracted by running Algorithm 1 satisfies Condition 1. In the following, we fix $v_H \in V_H$ to be an arbitrary node and $e_{H,1}, e_{H,2} \in E_H$ to be the hyperedges incident with v_H.

Case 1. $|V_T(v_H)| = 1$
First, let us assume $|V_T(v_H)| = 1$ and let v_T^* be the only



(a) Triplet of v_T, $\mathrm{pa}(v_T)$, and $\mathrm{ch}(v_T)$ chosen from the tree decomposition in Figure 3.



(b) Production rule extracted from the triplet in Figure 1a. Nodes IDs are added for explanation, and in the algorithm, they are removed except for the correspondence between the nodes in LHS and the external nodes in RHS.

Figure 1. Illustration of Algorithm 1.

node in $V_T(v_H)$, *i.e.*, $V_T(v_H) = \{v_T^*\}$. In this case, v_H cannot be an external node, because $\mathrm{pa}(v_T^*)$ does not contain v_H. Thus, v_H must be an internal node with $e_{H,1}$ and $e_{H,2}$ when it appears in R. In addition, no non-terminal hyperedge will be incident with v_H in R, because none of $\mathrm{ch}(v_T^*)$ contains v_H in it. Therefore, in this case, v_H is an internal node whose degree equals two in R.

Case 2. $|V_T(v_H)| \geq 2$ and v_H is an internal node in R
Such a production rule is made from a triplet $(v_T, \mathrm{pa}(v_T), \mathrm{ch}(v_T))$ such that $v_H \in \ell_T^{(V)}(v_T)$ and $v_H \notin \ell_T^{(V)}(\mathrm{pa}(v_T))$ hold.

If $e_{H,1} \in \ell_T^{(E)}(v_T)$ or $e_{H,2} \in \ell_T^{(E)}(v_T)$ holds, there exists exactly one node $v_T^{(\mathrm{ch})} \in \mathrm{ch}(v_T)$ such that $v_H \in \ell_T^{(V)}(v_T^{(\mathrm{ch})})$ holds. In this case, exactly one non-terminal hyperedge will be connected to v_H, and therefore, the degree of v_H equals two in R.

If $e_{H,1} \notin \ell_T^{(E)}(v_T)$ and $e_{H,2} \notin \ell_T^{(E)}(v_T)$ hold, there exist exactly two nodes $v_{T,1}^{(\mathrm{ch})}, v_{T,2}^{(\mathrm{ch})} \in \mathrm{ch}(v_T)$ such that $v_H \in \ell_T^{(V)}(v_{T,1}^{(\mathrm{ch})})$ and $v_H \in \ell_T^{(V)}(v_{T,2}^{(\mathrm{ch})})$ hold. In this case, exactly two non-terminal hyperedges will be connected to v_H, and therefore, the degree of v_H equals two in R.

Case 3. $|V_T(v_H)| \geq 2$ and v_H is an external node in R
Such a production rule is made from a triplet $(v_T, \mathrm{pa}(v_T), \mathrm{ch}(v_T))$ such that $v_H \in \ell_T^{(V)}(v_T)$ and $v_H \in \ell_T^{(V)}(\mathrm{pa}(v_T))$ hold.

If $e_{H,1} \in \ell_T^{(E)}(v_T)$ or $e_{H,2} \in \ell_T^{(E)}(v_T)$ holds, for each node $v_T^{(\mathrm{ch})} \in \mathrm{ch}(v_T)$, $v_H \notin \ell_T^{(V)}(v_T^{(\mathrm{ch})})$ holds. In this case, no non-terminal hyperedge will be connected to v_H, and therefore, the degree of v_H equals one in R.

If $e_{H,1} \notin \ell_T^{(E)}(v_T)$ and $e_{H,2} \notin \ell_T^{(E)}(v_T)$ hold, there exists exactly one node $v_T^{(\mathrm{ch})} \in \mathrm{ch}(v_T)$ such that $v_H \in \ell_T^{(V)}(v_T^{(\mathrm{ch})})$ holds. In this case, exactly one non-terminal hyperedge will be connected to v_H, and therefore, the degree of v_H equals one in R.

Therefore, for any case, Condition 1 holds, and therefore, Lemma 2 has been proven. □

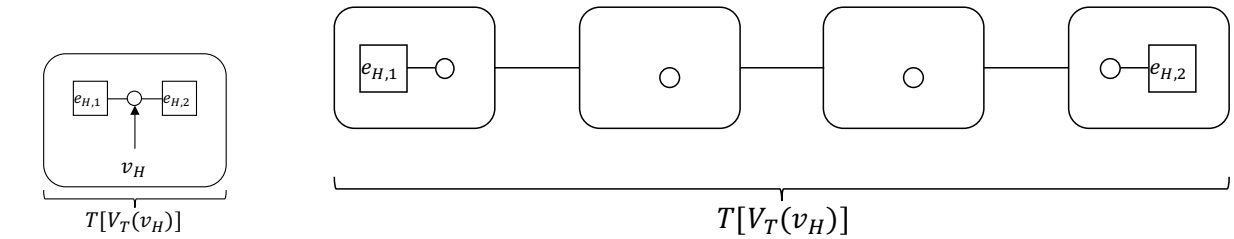
Proof of Lemma 3. Let H be an arbitrary hypergraph that can be derived from the starting symbol S. Note that H may contain non-terminal hyperedges. It is sufficient to prove that H is 2-regular, if the production rules satisfy Condition 1.

If H is directly derivable from the starting symbol S, then it is 2-regular, because Condition 1 guarantees that for any production rule $p = (S, R)$, R is 2-regular.

If H is directly derivable from H' by applying $p = (A, R)$, and if H' is 2-regular and derivable from S, then there exists a non-terminal hyperedge in $e_{H'} \in E_{H'}$ that is labeled as A and is replaced with R to yield H. For each node $v_{H'} \in V_{H'} \backslash e_{H'}$, the replacement does not change the degree, and the degree of $v_{H'}$ equals two in H. For each node $v_R \in V_R \backslash \mathrm{ext}(R)$[1], the replacement does not change the degree, and the degree of v_R equals two in H. For each node $v_{H'} \in e_{H'}$, the replacement first deletes $e_{H'}$, and then, connects the hyperedges adjacent to the external nodes $\mathrm{ext}(R)$. Since the degrees of the external nodes are one, the replacement does not change its degree. The same applies to each external node. Since the degree of each node of H is two, H is proven to be 2-regular. □

Proof of Lemma 4. Let T be a tree decomposition of a molecular hypergraph H. For each $v_T \in V_T$, all of the hyperedges in $\ell_T^{(E)}(v_T)$ are cardinally-consistent, which is guaranteed by the second condition in Def. 1. Therefore, letting the production rule extracted from v_T be $p = (A, R)$,

[1] $\mathrm{ext}(R)$ denotes the set of external nodes in R



(a) Case 1: $|V_T(v_H)| = 1$. (b) Cases 2, 3: $|V_T(v_H)| \geq 2$.

Figure 2. Illustrations of $T[V_T(v_H)]$ in two cases.

all of the hyperedges in R are cardinally-consistent. Since applying such a production rule preserves the cardinality consistency, this lemma has been proven. □

C. Model Configuration

We tune the model configuration using the reconstruction rate on the validation set. Both Enc_N and Dec_N use three-layer GRU (Cho et al., 2014) with 384 hidden units (Enc_N is bidirectional), handling a sequence of production rule embeddings in 128-dimensional space. In Enc_N, the output of GRU is fed into a linear layer to compute the mean and log variance of a 72-dimensional Gaussian distribution, and the latent vector $z \in \mathbb{R}^{72}$ is sampled from it as the output of Enc_N. The encoder and decoder are trained by optimizing the objective function of β-VAE (Higgins et al., 2017) with $\beta = 0.01$ using ADAM (Kingma & Ba, 2015) with initial learning rate 5×10^{-4}. As a predictive model $\hat{f}\colon \mathbb{R}^{72} \to \mathbb{R}$, we employ a linear regression. Whenever target values are available, we jointly train seq2seq VAE and the predictive model.

D. Basic Statistics of MHG

This section reports basic statistics of MHG inferred using the ZINC dataset.

From the ZINC dataset, our algorithm obtains 2,031 production rules, of which 1,424 are starting rules. Each molecule is associated with 25 production rules on average. While the grammar seems to be huge, 2/3 of the starting rules (1,073 rules) are used by less than ten molecules, and in total, 2,355 out of 250k molecules are using these starting rules. If ignoring these rules, our grammar has moderate size.

We also investigate the coverage rate of the language associated with MHG (*i.e.*, how many molecules out of all possible molecules can MHG represent?). While we could not provide any theoretical validation on the coverage rate, we instead approximately estimate the coverage rate by the number of molecules in the test set that cannot be parsed using the grammar inferred using the training set. As a result, 16 out of 5,000 molecules cannot be parsed, and thus, the coverage rate will be 99.68%, and we believe this coverage is sufficiently high.

Algorithm 2 Local Molecular Optimization

In: Mol. graph g_0 and its latent vector z_0, Dec, $\hat{f}$, step size η, similarity measure $\mathrm{sim}(\cdot,\cdot)$, threshold τ, # iterations K.

$g^\star \leftarrow \mathrm{null}$, $y^\star \leftarrow -\infty$
for $k = 1, \ldots, K$ **do**
  $z_k \leftarrow z_{k-1} + \eta \frac{\partial \hat{f}(z_{k-1})}{\partial z}$
  $g_k \leftarrow \mathsf{Dec}(z_k)$, $y_k \leftarrow \hat{f}(z_k)$
  if $\mathrm{sim}(g_0, g_k) \geq \tau$, $g_0 \neq g_k$, and $y_k > y^\star$ **then**
    $g^\star \leftarrow g_k$, $y^\star \leftarrow y_k$
return $(g^\star, y^\star)$

E. Local Molecular Optimization

Local molecular optimization (Algorithm 2) aims to improve the property of a given molecule without modifying it too much. This problem setting is originally proposed by Jin et al. (2018) and is formalized as,

$$m^\star = \mathsf{Dec}\left(\underset{z\colon \mathrm{sim}(m,\mathsf{Dec}(z))\geq\tau}{\arg\max} f(\mathsf{Dec}(z))\right). \quad (1)$$

where $\mathrm{sim}(m, m')$ computes a similarity between molecules m and m', and τ is a similarity threshold. We use Tanimoto similarity with Morgan fingerprint (radius=2). Problem 1 is approximately solved by Algorithm 2, where we substitute our predictive model $\hat{f}$ for the unknown target function f.

Note that the predictive model $\hat{f}$ requires a large data set for training and it is difficult to apply the limited oracle scenario to local molecular optimization. Since the unlimited oracle scenario is out of our scope, we leave this topic in the appendix. We would like to leave a local optimization algorithm tailored for the limited oracle scenario as future work.

Protocol. As a target property, we use a penalized logP:

$$f(m) = \mathrm{logP}(m) - \mathrm{SA}(m). \quad (2)$$

Table 1. Results on local molecular optimization.

	Improvement				Similarity				Success			
δ	0.0	0.2	0.4	0.6	0.0	0.2	0.4	0.6	0.0	0.2	0.4	0.6
JT-VAE	1.91 ±2.04	1.68 ±1.85	0.84 ±1.45	0.21 ±0.71	0.28 ±0.15	0.33 ±0.13	0.51 ±0.10	0.69 ±0.06	97.5%	97.1%	83.6%	46.4%
GCPN	4.20 ±1.28	4.12 ±1.19	2.49 ±1.30	0.79 ±0.63	0.32 ±0.12	0.34 ±0.11	0.47 ±0.08	0.68 ±0.08	100%	100%	100%	100%
Ours	3.28 ±2.19	2.40 ±2.16	1.00 ±1.87	0.61 ±1.20	0.09 ±0.06	0.26 ±0.10	0.52 ±0.11	0.70 ±0.06	100%	86.3%	43.5%	17.0%

We choose 800 molecules with the lowest penalized logP from the test set. For each initial molecule m, we run Algorithm 2 with $\tau \in \{0, 0.2, 0.4, 0.6\}$, $K = 80$, and $\eta = 0.01$. For each threshold τ, we report (i) the mean and standard deviation of the target value improvements, (ii) those of the similarity, and (iii) the success rate, where Algorithm 2 succeeds if the output is not null, *i.e.*, if there exists a modified molecule that satisfies the similarity constraint.

Result. Table 1 reports the scores. First of all, we observe that GCPN outperforms VAE-based methods. This result is reasonable considering that this task assumes the unlimited oracle scenario. Among the VAE-based methods, for any similarity threshold, our method improves the target property better than JT-VAE, which demonstrates the effectiveness of our method over JT-VAE.

F. Molecules Discovered by Global Molecular Optimization

In the following, we illustrate the top 50 molecules found by global molecular optimization in Section 8.2.

References

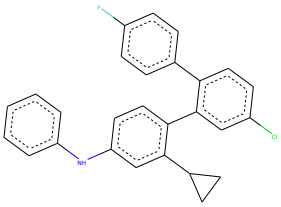
Aguiñaga, S., Palacios, R., Chiang, D., and Weninger, T. Growing graphs from hyperedge replacement graph grammars. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pp. 469–478, 2016.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1724–1734, 2014.

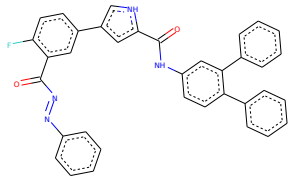
Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. β-VAE: Learning basic visual concepts with a constrained variational framework. In *Proceedings of the Fifth International Conference on Learning Representations*, 2017.

Jin, W., Barzilay, R., and Jaakkola, T. Junction tree variational autoencoder for molecular graph generation. In *Proceedings of the Thirty-fifth International Conference on Machine Learning*, 2018.

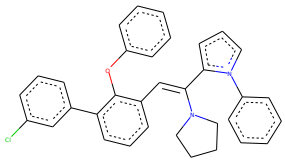
Kingma, D. and Ba, J. Adam: A method for stochastic optimization. In *Proceedings of the Third International Conference on Learning Representations*, 2015.



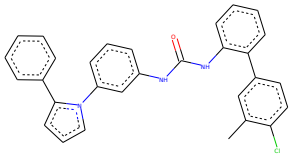
5.558



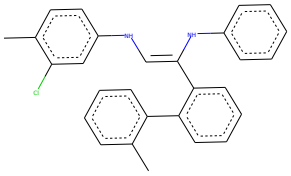
5.401



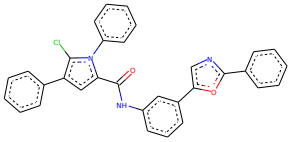
5.344



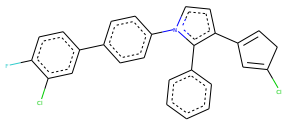
5.289



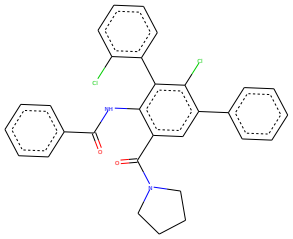
5.063



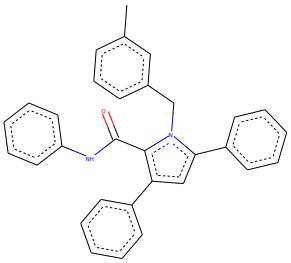
4.979



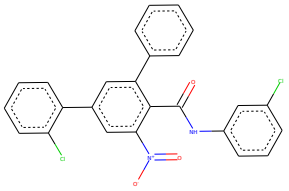
4.811



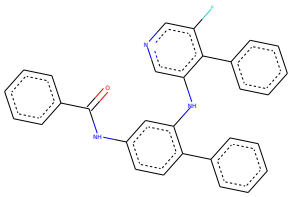
4.779



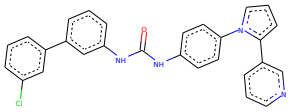
4.775



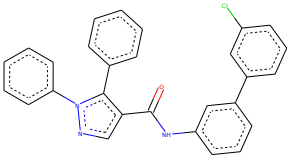
4.730



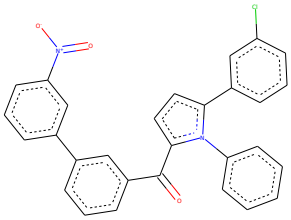
4.712



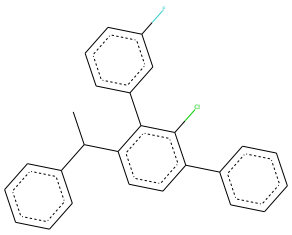
4.641



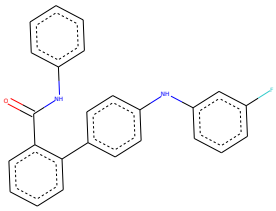
4.617



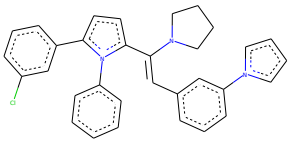
4.598



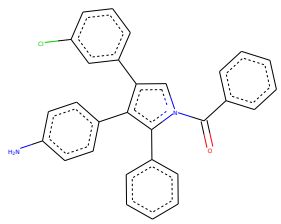
4.595



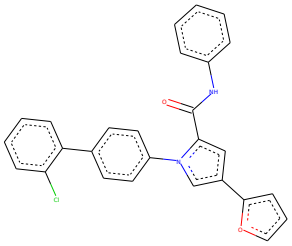
4.555



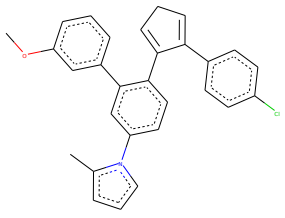
4.546



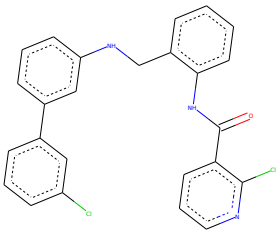
4.538



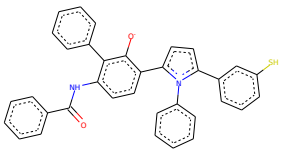
4.484



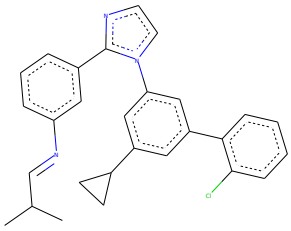
4.464



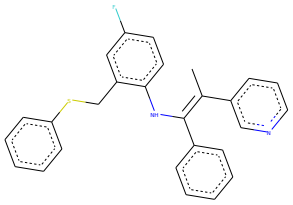
4.450



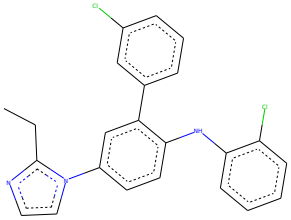
4.443



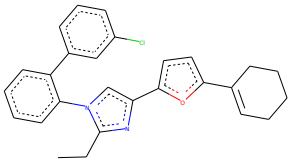
4.408



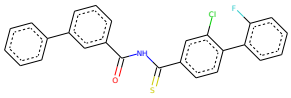
4.404



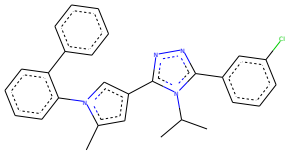
4.374



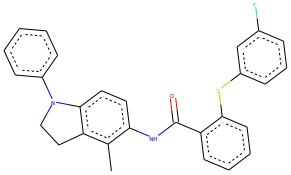
4.362



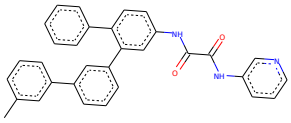
4.354



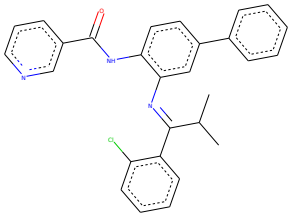
4.351



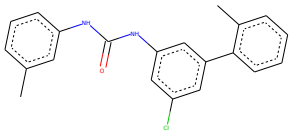
4.349



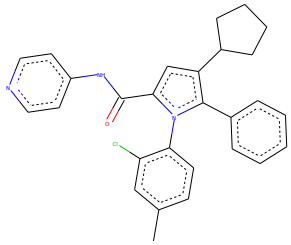
4.341



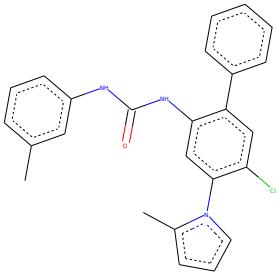
4.335



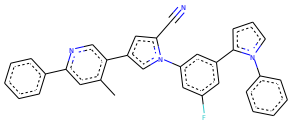
4.294



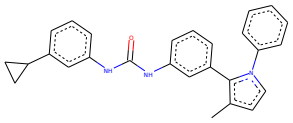
4.274



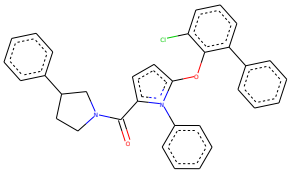
4.265



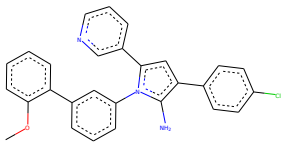
4.259



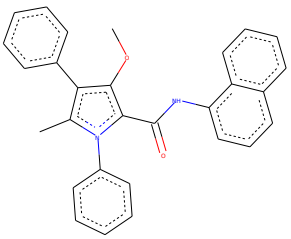
4.258



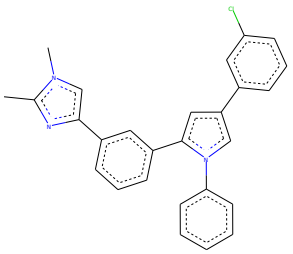
4.242



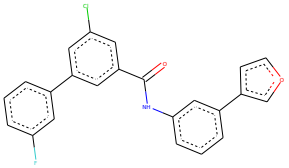
4.240



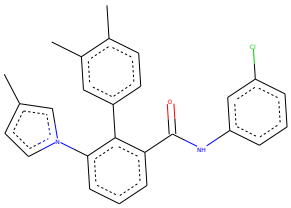
4.233



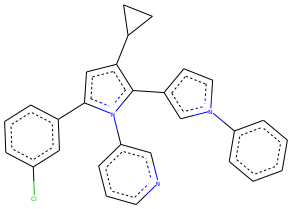
4.209



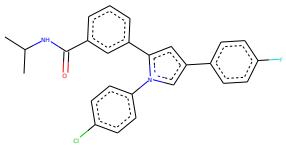
4.202



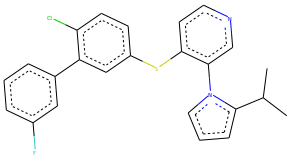
4.196



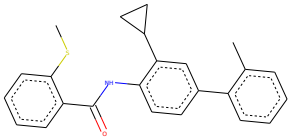
4.185



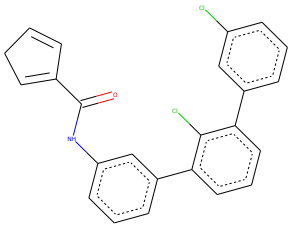
4.171



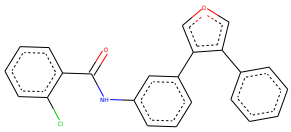
4.169



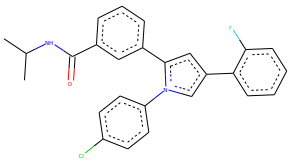
4.163



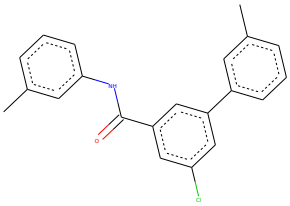
4.153



4.150



4.124



4.124