

# Molecular structure inference from molecular spectra – notes on machine learning methods

# For context

- Computational Biology and Bioinformatics MSc (ETHZ)
- Background in numerical maths
- Machine learning or chemistry – not so much
- One month in
- Today: own analysis of others' work

Inferring molecule structure from its  $^1\text{H}$ -NMR spectrum

# Inferring SMILES from simulated $^1\text{H}$ -NMR multiplets and the sum formula

# Training dataset and transformers model

- Daniel Lowe. *Chemical reactions from US patents (1976–Sep2016)*. [figshare](#), 2017.  
[10.6084/m9.figshare.5104873.v1](#)
- Marvin Alberts et al. “Unraveling Molecular Structure: A Multimodal Spectroscopic Dataset for Chemistry”. In: *NeurIPS 2024 Datasets and Benchmarks Track*. 2024.  
[10.48550/arXiv.2407.17492](#)
- Marvin Alberts et al. *Learning the Language of NMR: Structure Elucidation from NMR spectra using Transformer Models*. 2023.  
[10.26434/chemrxiv-2023-8wxcz](#)

# Training dataset and transformers model

- **Dataset:** USPTO reaction dataset<sup>1</sup>
  - 1,435,481 reactions
  - Realistic molecules & common chemicals
- **Unique molecules:**
  - Initially: 1,675,439
  - Filter:  $5 < \# \text{ heavy atoms} < 35$
  - Allowed elements: C, H, O, N, S, P, Si, B, halogens
  - After filtering: 1,416,499
  - Simulation w/ MestReNova (all spectra): **794,403** molecules
- **Spectra simulated:** <sup>1</sup>H-NMR, IR, <sup>13</sup>C-NMR, HSQC-NMR, MS/MS
- “... **deuterated chloroform** as solvent. Default settings were used”
- **Representation:** SMILES

---

<sup>1</sup>Lowe, 2017 [1]

# Training dataset

## SRC-TRAIN.TXT

- C 15 H 24 N 2 1HNMR | 6.86 6.83 d 1H J 0.68 | 5.98  
5.96 s 1H | 4.22 4.19 s 2H | 3.83 3.74 p 1H J 6.74 |  
3.55 3.50 m 2H | 2.24 2.21 s 3H | 2.05 1.99 m 2H |  
1.35 1.32 s 5H | 1.26 1.22 d 6H J 6.67
- C 30 H 28 F N O 3 1HNMR | 7.39 7.30 m 4H | 7.30 7.05 m  
11H | ... | 2.60 2.46 m 2H
- ...

## TGT-TRAIN.TXT

- C c 1 c c 2 c ( c c 1 N ) N ( C ( C ) C ) ... ( C ) C
- O = C ( C C c 1 c c c c c 1 ) N C ( C c 1 c c c ( O c  
2 c c c c c 2 ) c c 1 ) C ( O ) c 1 c c c ( F ) c c 1
- ...

# Training dataset

## SRC-TRAIN.TXT

- C 15 H 24 N 2 1HNMR | 6.86 6.83 d 1H J 0.68 | 5.98 5.96 s 1H | 4.22 4.19 s 2H | 3.83 3.74 p 1H J 6.74 | 3.55 3.50 m 2H | 2.24 2.21 s 3H | 2.05 1.99 m 2H | 1.35 1.32 s 5H | 1.26 1.22 d 6H J 6.67
- C 30 H 28 F N O 3 1HNMR | 7.39 7.30 m 4H | 7.30 7.05 m 11H | ... | 2.60 2.46 m 2H
- ...

## TGT-TRAIN.TXT with explicit hydrogens (new)

- [CH3] [c] 1 [cH] [c] 2 [c] ( [cH] [c] 1 [NH2] ) [N] ( [CH] ( [CH3] ) [CH3] ) [CH2] [CH2] [C] 2 ( [CH3] ) [CH3 ]
- ...



# Input vocabulary

Constructed from 10k samples

|  $\times 92120$ , J  $\times 52692$ , 1H  $\times 46441$ , 2H  $\times 27470$ , m  $\times 25035$ , s  $\times 14414$ , d  $\times 13282$ , C  $\times 10000$ , H  $\times 10000$ ,  
1HNMR  $\times 10000$ , dd  $\times 9417$ , 3H  $\times 9343$ , O  $\times 9018$ , N  $\times 8443$ , t  $\times 7266$ , 2  $\times 6096$ , ddd  $\times 4820$ , 3  $\times 4683$ ,  
dt  $\times 3241$ , 4H  $\times 3220$ , 4  $\times 2473$ , F  $\times 2346$ , S  $\times 2314$ , Cl  $\times 2132$ , ddt  $\times 1913$ , q  $\times 1820$ , td  $\times 1690$ , 5H  $\times 1627$ ,  
6H  $\times 1622$ , 5  $\times 1430$ , dddd  $\times 1430$ , 0.92  $\times 1393$ , 7.32  $\times 1286$ , dq  $\times 1266$ , 14  $\times 1166$ , tt  $\times 1163$ ,  
17  $\times 1152$ , 16  $\times 1144$ , 15  $\times 1112$ , 19  $\times 1098$ , 18  $\times 1096$ , 2.19  $\times 1064$ , 20  $\times 1060$ , 7.14  $\times 1058$ ,  
p  $\times 1047$ , 1.47  $\times 1040$ , 7.33  $\times 1028$ , 7.69  $\times 1024$ , 12  $\times 1011$ , 2.20  $\times 1009$ , 7.26  $\times 1008$ ,  
13  $\times 994$ , 7.51  $\times 973$ , 11  $\times 956$ , 21  $\times 947$ , Br  $\times 941$ , 8H  $\times 929$ , 2.10  $\times 921$ , 22  $\times 918$ , .....

2019 tokens

a hard-coded dimension in the ML model on the input side

# Output vocabulary

Constructed from 10k samples

C<sub>×94533</sub>, C<sub>×68425</sub>, ( <sub>×39107</sub>, ) <sub>×39107</sub>, 1<sub>×25622</sub>, O<sub>×23680</sub>, 2<sub>×17598</sub>, =<sub>×13822</sub>, N<sub>×11314</sub>, n<sub>×9750</sub>,  
3<sub>×6992</sub>, F<sub>×5100</sub>, Cl<sub>×2717</sub>, -<sub>×2531</sub>, 4<sub>×1826</sub>, S<sub>×1744</sub>, [C@@H]<sub>×1443</sub>, [C@H]<sub>×1257</sub>, Br<sub>×1000</sub>,  
[nH]<sub>×986</sub>, s<sub>×871</sub>, #<sub>×869</sub>, /<sub>×522</sub>, o<sub>×498</sub>, 5<sub>×270</sub>, I<sub>×181</sub>, [C@]<sub>×148</sub>, [N+]<sub>×130</sub>,  
[O-]<sub>×112</sub>, \<sub>×106</sub>, [C@@]<sub>×100</sub>, P<sub>×62</sub>, [N-]<sub>×46</sub>, [n+]<sub>×30</sub>, 6<sub>×8</sub>, [S@]<sub>×7</sub>, [2H]<sub>×4</sub>,  
[C-]<sub>×3</sub>, [PH2]<sub>×1</sub>, [N@@+]<sub>×1</sub>, p<sub>×1</sub>, [S@@]<sub>×1</sub>, [N@]<sub>×1</sub>

43 tokens

a hard-coded dimension in the ML model on the output side

# The “model”: encoder-decoder transformer

- The **encoder** converts input NMR tokens into 512-dimensional vectors via a trainable “embedding” with added “positional encoding” and random “dropout”.
- These embeddings are processed through 4 layers of “self-attention” and feedforward networks, with “residual” (passthrough) connections and layer normalization (no masking).
- The **decoder** generates SMILES tokens in an **autoregressive** manner from a fixed output vocabulary.
- It first embeds the partial SMILES sequence with positional encoding, then, in each of its 4 layers, applies:
  - Masked self-attention (attending only to previous tokens),
  - Cross-attention over the encoder output, and
  - A feedforward network.
- A linear generator converts the decoder output into “logits” over the SMILES vocabulary, interpreted as the probability of the next token.

# The “model”: encoder-decoder transformer

- The **encoder** converts input NMR tokens into 512-dimensional vectors via a trainable “embedding” with added “positional encoding” and random “dropout”.
- These embeddings are processed through 4 layers of “self-attention” and feedforward networks, with “residual” (passthrough) connections and layer normalization (no masking).
- The **decoder** generates SMILES tokens in an **autoregressive** manner from a fixed output vocabulary.
- It first embeds the partial SMILES sequence with positional encoding, then, in each of its 4 layers, applies:
  - Masked self-attention (attending only to previous tokens),
  - Cross-attention over the encoder output, and
  - A feedforward network.
- A linear generator converts the decoder output into “logits” over the SMILES vocabulary, interpreted as the probability of the next token.

# The “model”: encoder-decoder transformer

- The **encoder** converts input NMR tokens into 512-dimensional vectors via a trainable “embedding” with added “positional encoding” and random “dropout”.
- These embeddings are processed through 4 layers of “self-attention” and feedforward networks, with “residual” (passthrough) connections and layer normalization (no masking).
- The **decoder** generates SMILES tokens in an **autoregressive** manner from a fixed output vocabulary.
- It first embeds the partial SMILES sequence with positional encoding, then, in each of its 4 layers, applies:
  - Masked self-attention (attending only to previous tokens),
  - Cross-attention over the encoder output, and
  - A feedforward network.
- A linear generator converts the decoder output into “logits” over the SMILES vocabulary, interpreted as the probability of the next token.

# The “model”: encoder-decoder transformer

- The **encoder** converts input NMR tokens into 512-dimensional vectors via a trainable “embedding” with added “positional encoding” and random “dropout”.
- These embeddings are processed through 4 layers of “self-attention” and feedforward networks, with “residual” (passthrough) connections and layer normalization (no masking).
- The **decoder** generates SMILES tokens in an **autoregressive** manner from a fixed output vocabulary.
- It first embeds the partial SMILES sequence with positional encoding, then, in each of its 4 layers, applies:
  - Masked self-attention (attending only to previous tokens),
  - Cross-attention over the encoder output, and
  - A feedforward network.
- A linear generator converts the decoder output into “logits” over the SMILES vocabulary, interpreted as the probability of the next token.

# The “model”: encoder-decoder transformer

- The **encoder** converts input NMR tokens into 512-dimensional vectors via a trainable “embedding” with added “positional encoding” and random “dropout”.
- These embeddings are processed through 4 layers of “self-attention” and feedforward networks, with “residual” (passthrough) connections and layer normalization (no masking).
- The **decoder** generates SMILES tokens in an **autoregressive** manner from a fixed output vocabulary.
- It first embeds the partial SMILES sequence with positional encoding, then, in each of its 4 layers, applies:
  - Masked self-attention (attending only to previous tokens),
  - Cross-attention over the encoder output, and
  - A feedforward network.
- A linear generator converts the decoder output into “logits” over the SMILES vocabulary, interpreted as the probability of the next token.

# What is “attention”? – a glorified transistor

It is a mechanism that computes a weighted sum of input features, enabling the model to focus on the most relevant parts of a sequence. Introduced in “Attention is all you need” by Vaswani et al. [4].

$$\underset{\longleftrightarrow}{\text{softmax}} \left( \frac{QK^T}{\sqrt{d_k}} \right) V$$

- For each token, a trainable **query** vector is compared to a set of trainable **key** vectors (from the same sequence in self-attention or from the encoder in cross-attention) via a dot-product.
- The resulting similarity scores are scaled and passed through a “softmax” (over the keys) to obtain **attention weights**.
- These weights determine how much each token’s associated trainable **value** vector contributes to the output.



# Training and model predictions

- Model by Alberts et al. [2],  $\sim 30\text{M}$  trainable parameters
- 679'195 training samples, batch size 4k
- Re-trained on NVIDIA  $1 \times \text{A10 GPU}^2$  ( $\sim 35\text{h}$ ), up to 250k batches
- The model generates several hypotheses ranked by “score”, i.e., log-likelihood of the prediction according to the model
- Top-N accuracy evaluated on 1'000 or 10'000 unseen samples
- “Beam search” is a heuristic to find the *most likely* hypotheses; use beams size 100 (i.e., 100 concurrent hypotheses, keep top 10)

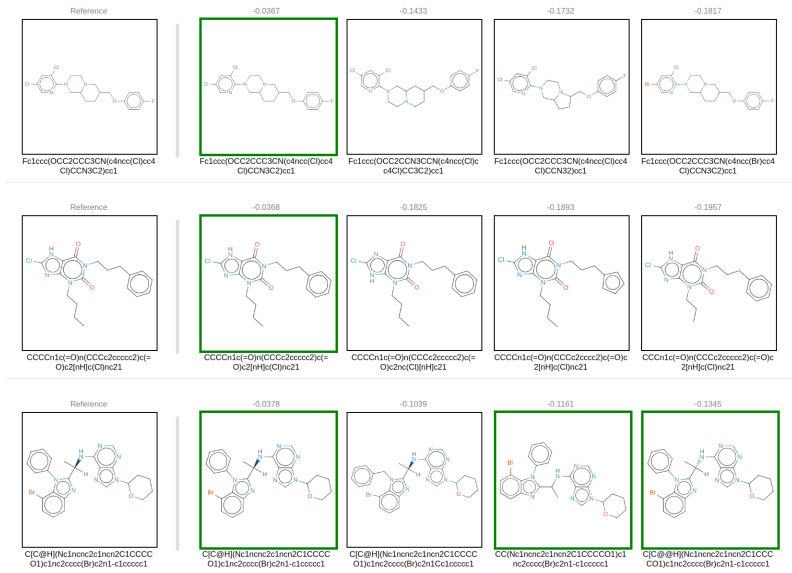
---

<sup>2</sup>Thanks to Lambda Cloud for compute credits

# Evaluation

What is a *correct* prediction by the trained model?

- ✗ Predicted SMILES matches exactly
- ✓ Canonicalize before comparison
- ✓ Chirality-aware
- ✓ Exclude hypotheses with the wrong molecular sum-formula

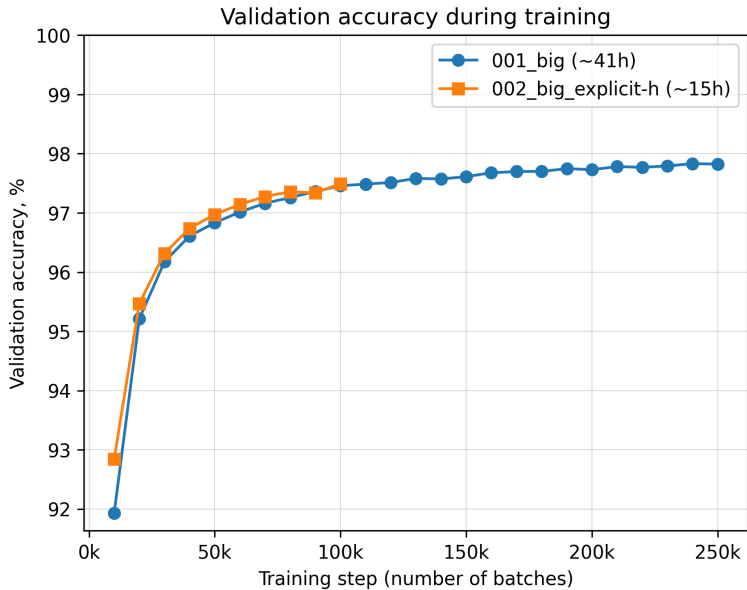


# Evaluation

Validation accuracy during training –

next-token prediction accuracy  
over the unused validation dataset  
with “teacher forcing”

– indicates training progress.



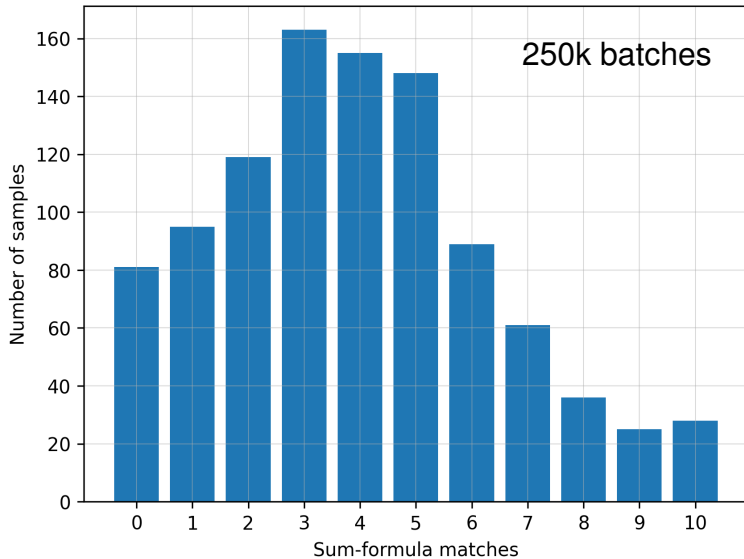
In generating SMILES token-by-token,  
“failures” could mean non-canonical branching

If the probability of wrong token is  $\approx 2\%$   
then among 50 tokens (typical size)  
we have at most one failure  
with probability  $\approx 74\%$ .

# Evaluation

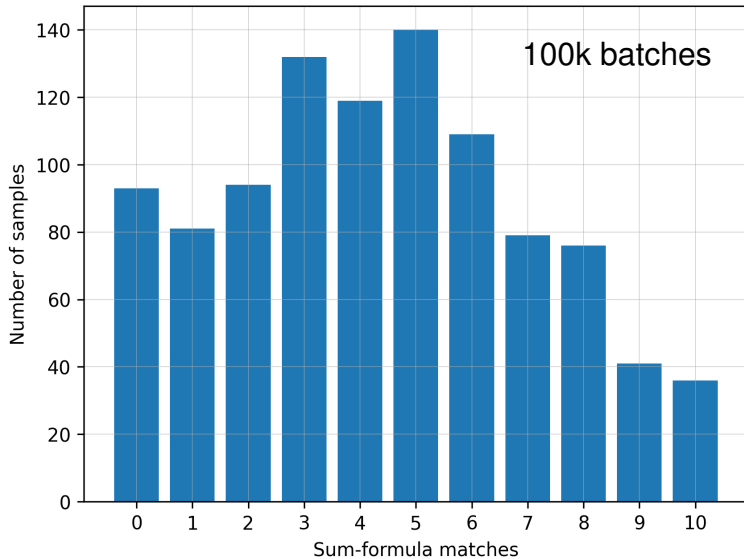
Do predictions have the correct sum-formula?

Sum-formula matches among top-10 predictions

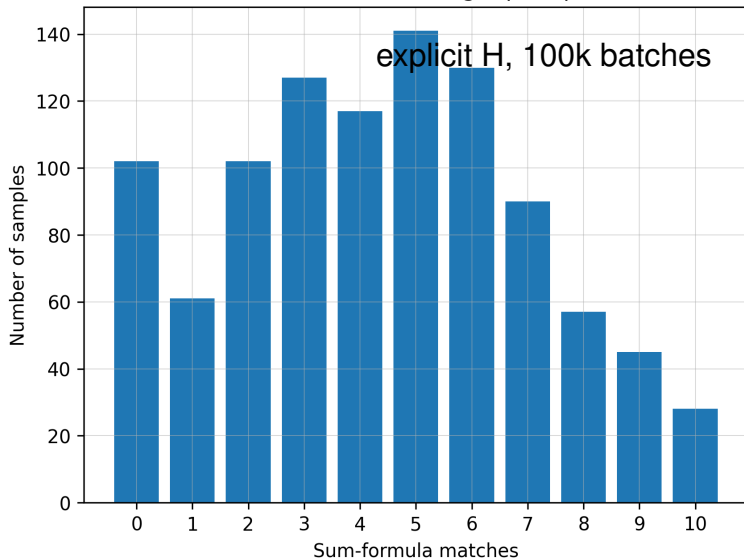




Sum-formula matches among top-10 predictions



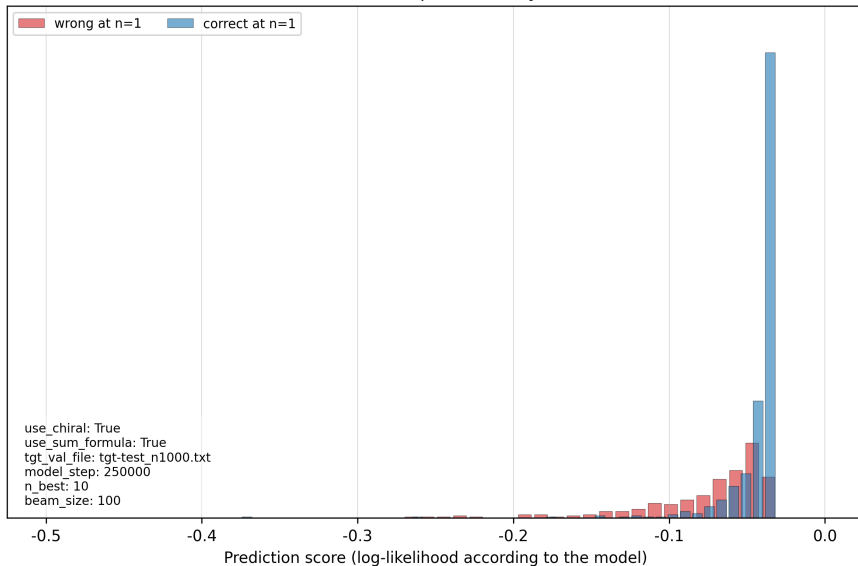
Sum-formula matches among top-10 predictions



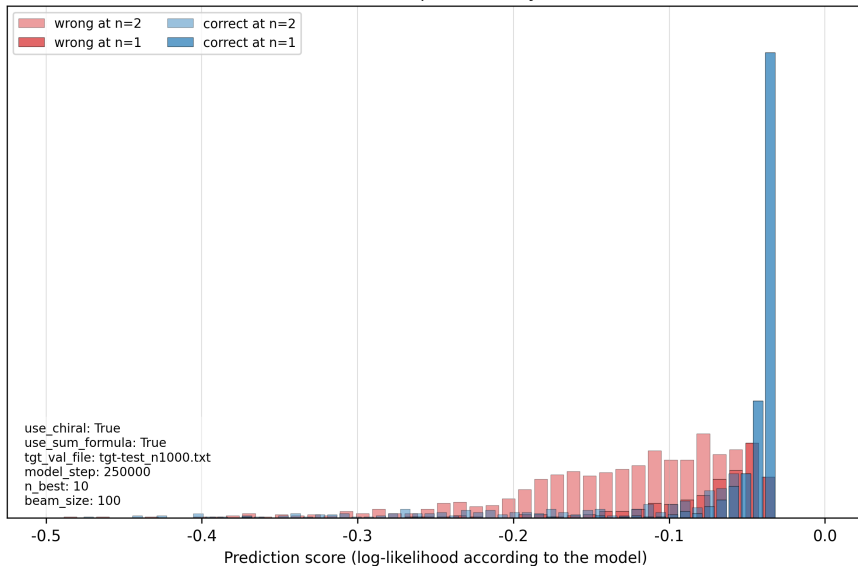
# Evaluation

Which top-n predictions are correct?

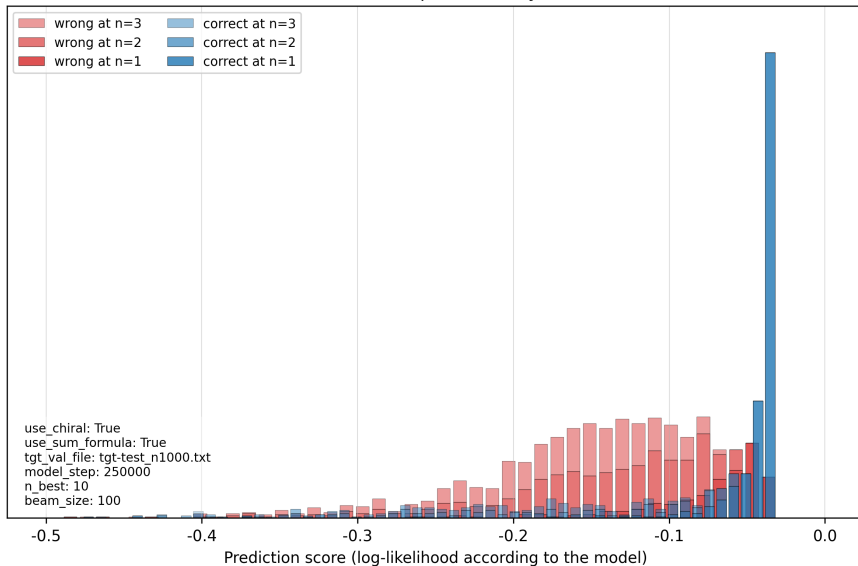
# Inferred SMILES (top-1 accuracy: 62.80%)



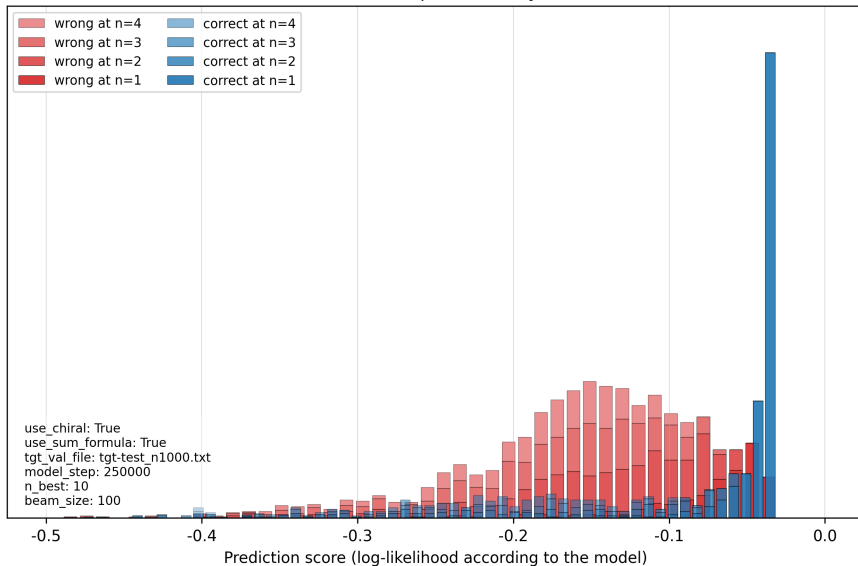
## Inferred SMILES (top-2 accuracy: 71.30%)



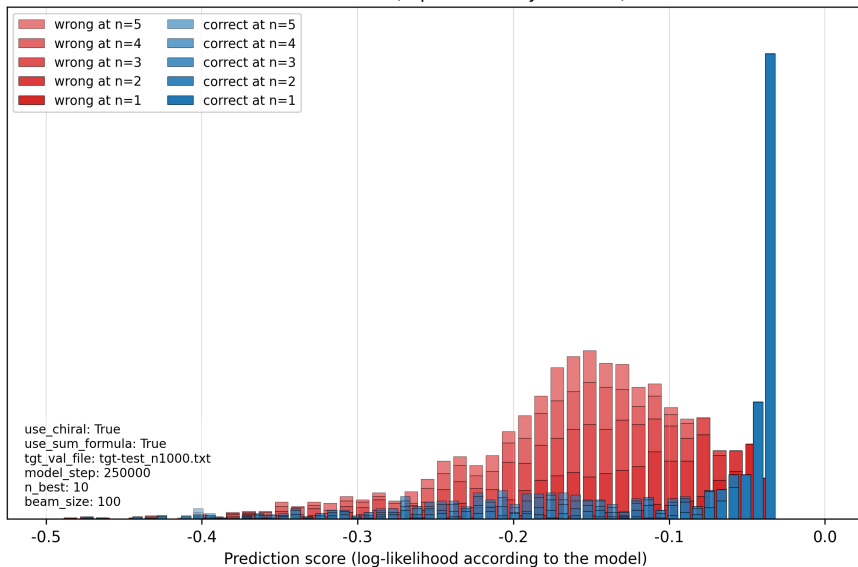
# Inferred SMILES (top-3 accuracy: 74.50%)



# Inferred SMILES (top-4 accuracy: 76.50%)



# Inferred SMILES (top-5 accuracy: 77.30%)

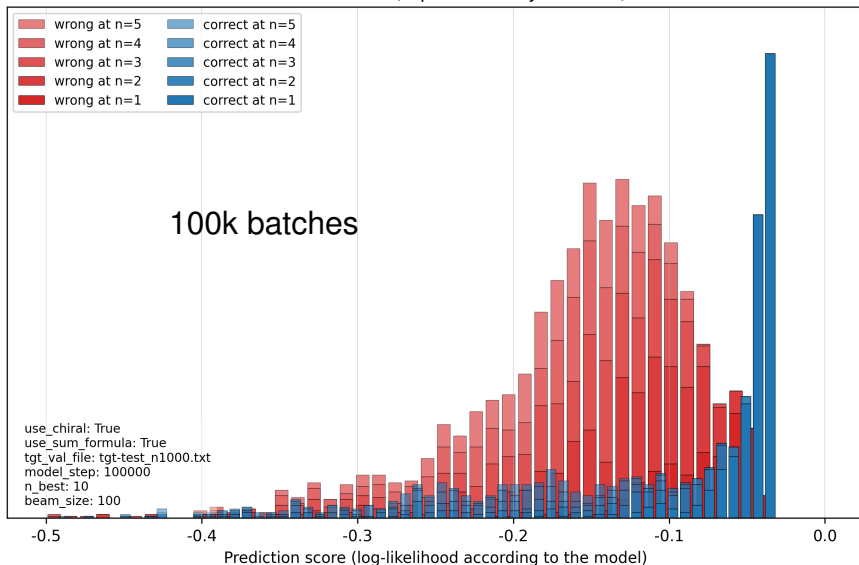




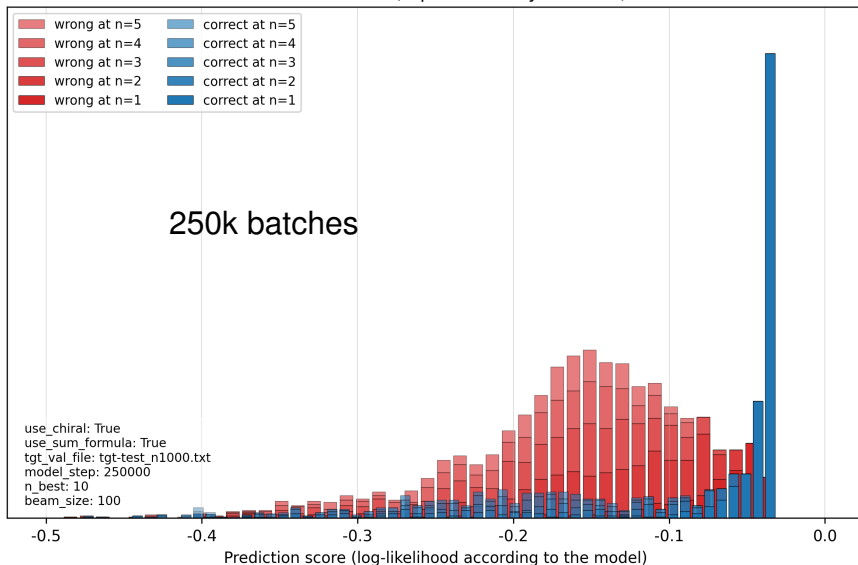
# Evaluation

100k batches vs 250k batches

## Inferred SMILES (top-5 accuracy: 74.70%)



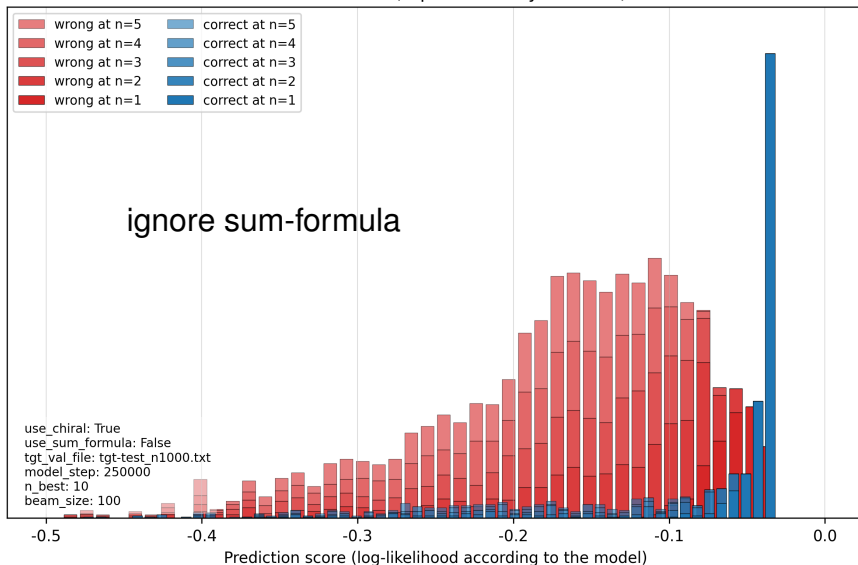
# Inferred SMILES (top-5 accuracy: 77.30%)



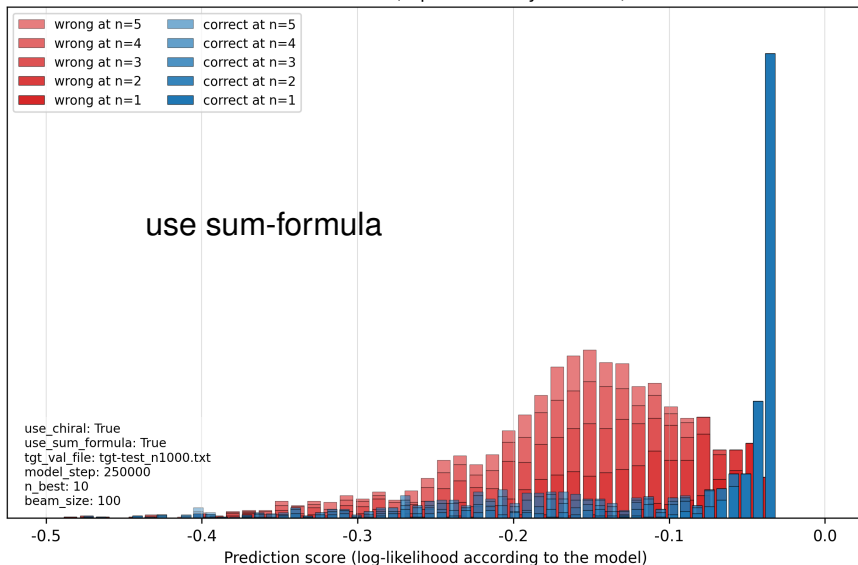
# Evaluation

sum-formula: ignore or use

# Inferred SMILES (top-5 accuracy: 75.80%)



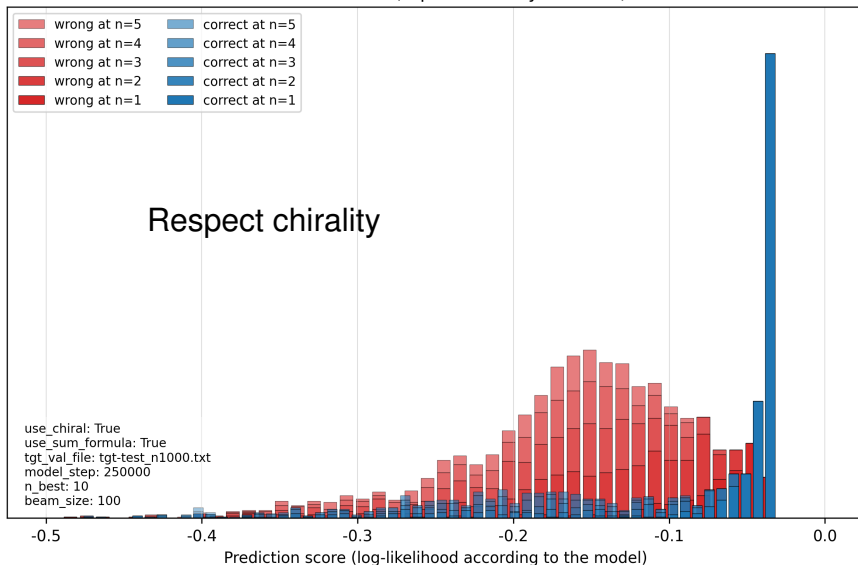
# Inferred SMILES (top-5 accuracy: 77.30%)



# Evaluation

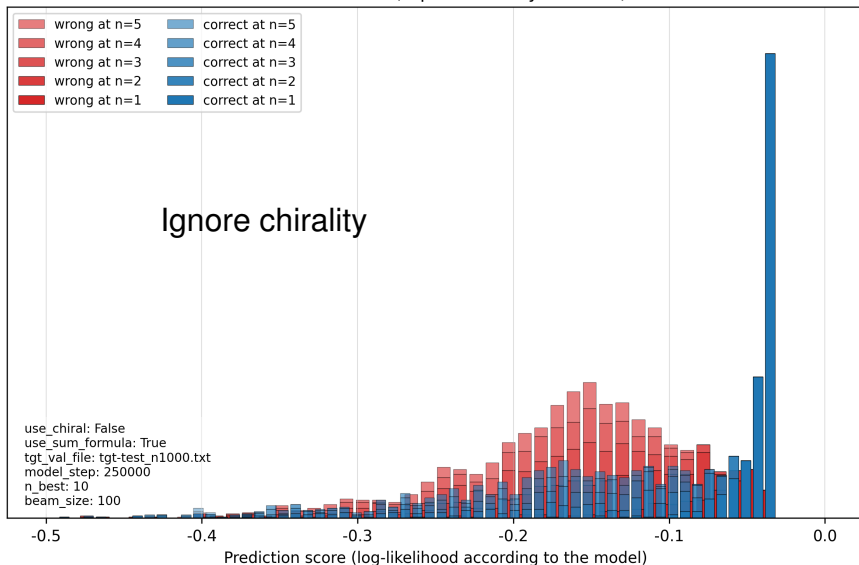
chirality: on vs off

## Inferred SMILES (top-5 accuracy: 77.30%)





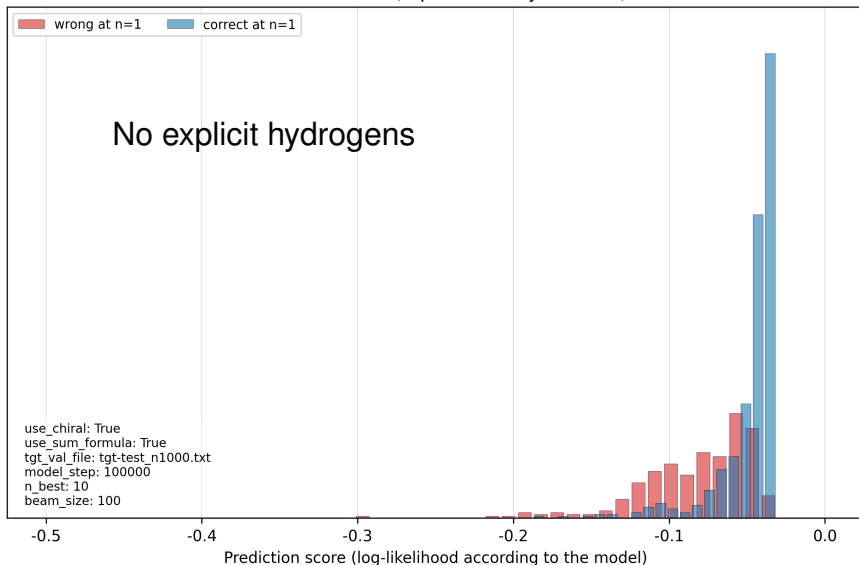
# Inferred SMILES (top-5 accuracy: 78.70%)



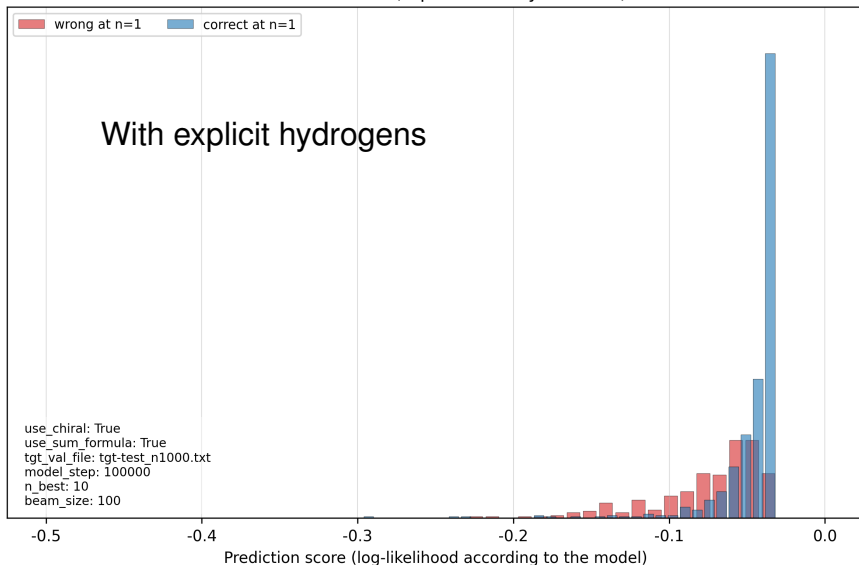
# Evaluation

explicit hydrogens: no vs yes

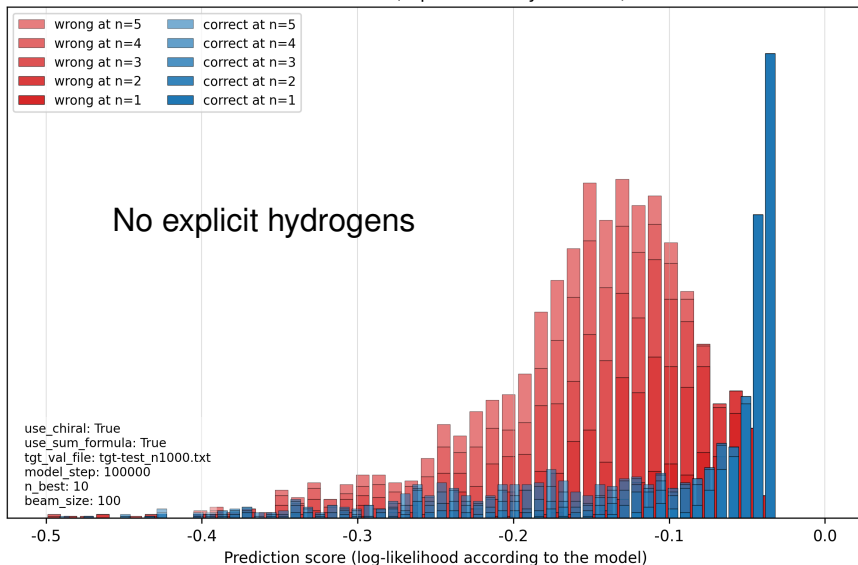
# Inferred SMILES (top-1 accuracy: 58.40%)



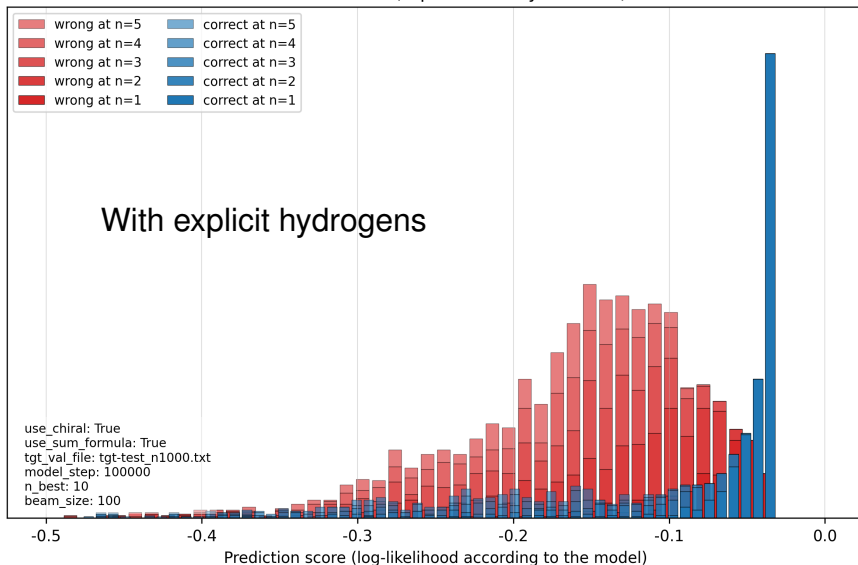
## Inferred SMILES (top-1 accuracy: 59.60%)



## Inferred SMILES (top-5 accuracy: 74.70%)



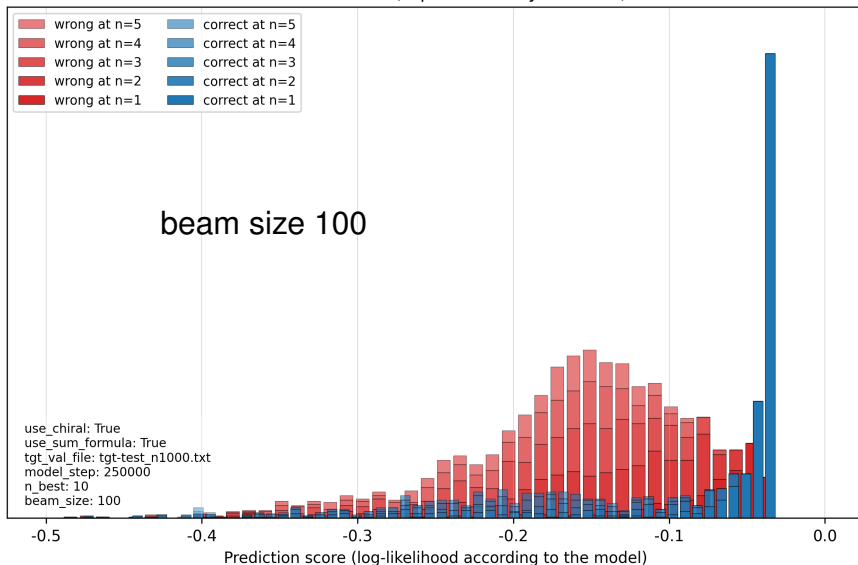
## Inferred SMILES (top-5 accuracy: 74.70%)



# Evaluation

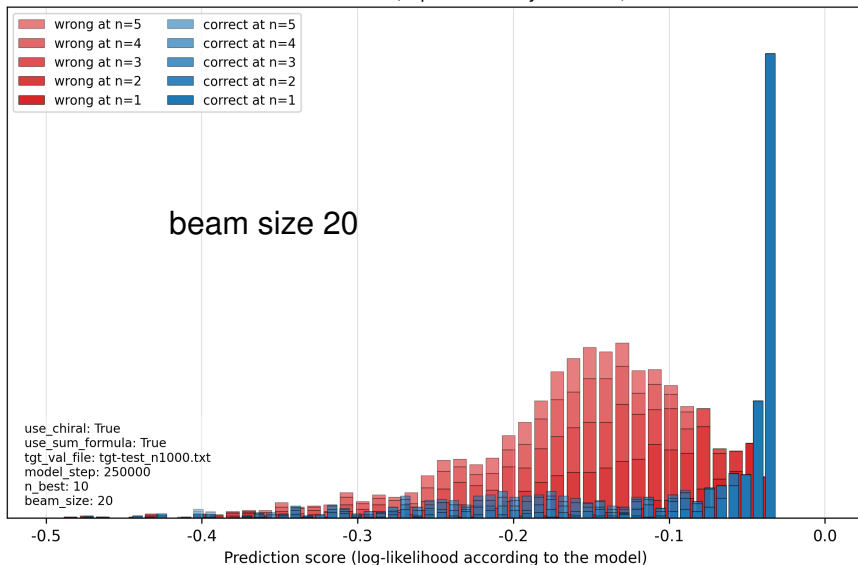
beam size (in hypothesis search): 100 vs 20

# Inferred SMILES (top-5 accuracy: 77.30%)

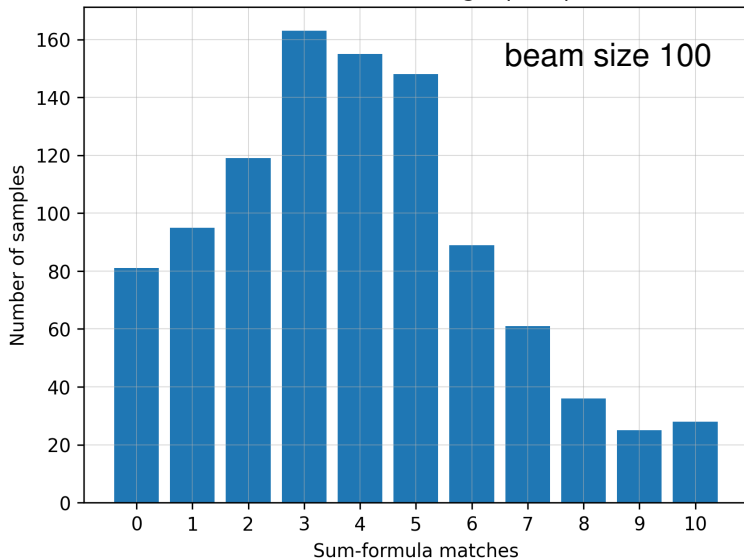




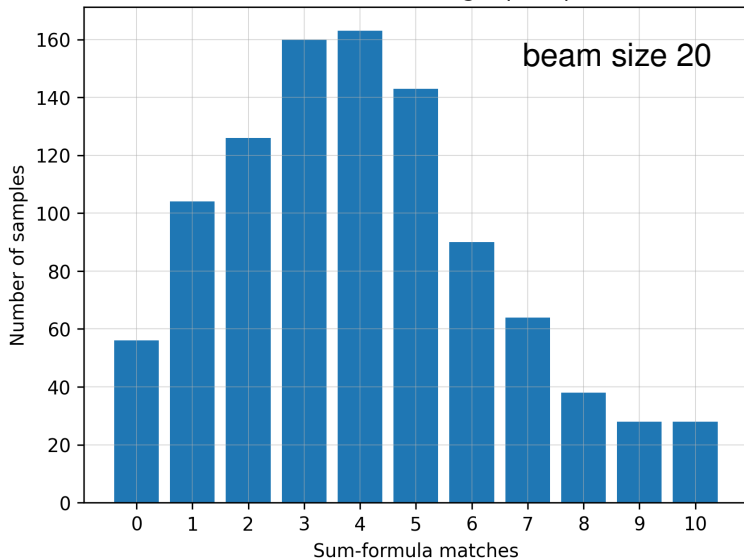
# Inferred SMILES (top-5 accuracy: 80.20%)



Sum-formula matches among top-10 predictions



Sum-formula matches among top-10 predictions



# Evaluation

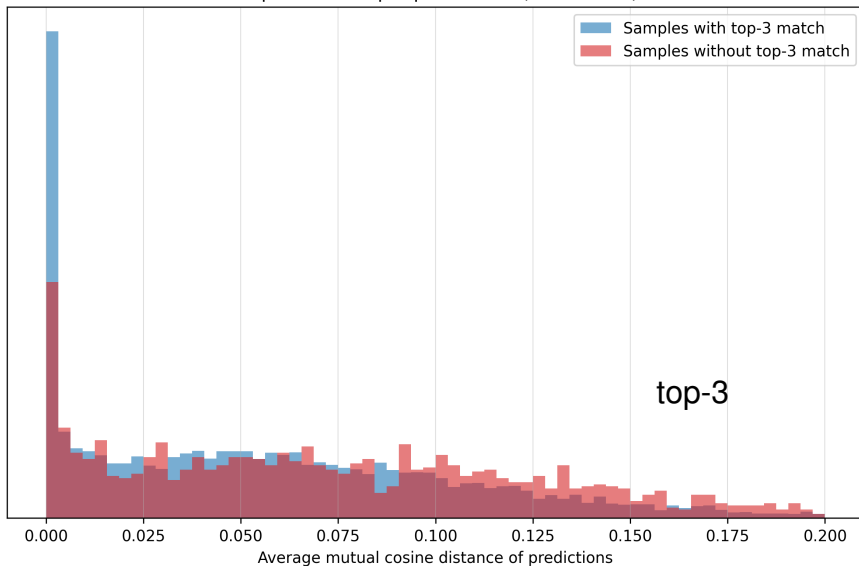
Average mutual cosine distance among top-n predictions,  
or “dispersion”

... computed using the molecule-to-vector mapping ChemBERTa<sup>3</sup>  
(self-supervised transformer pretrained on SMILES)

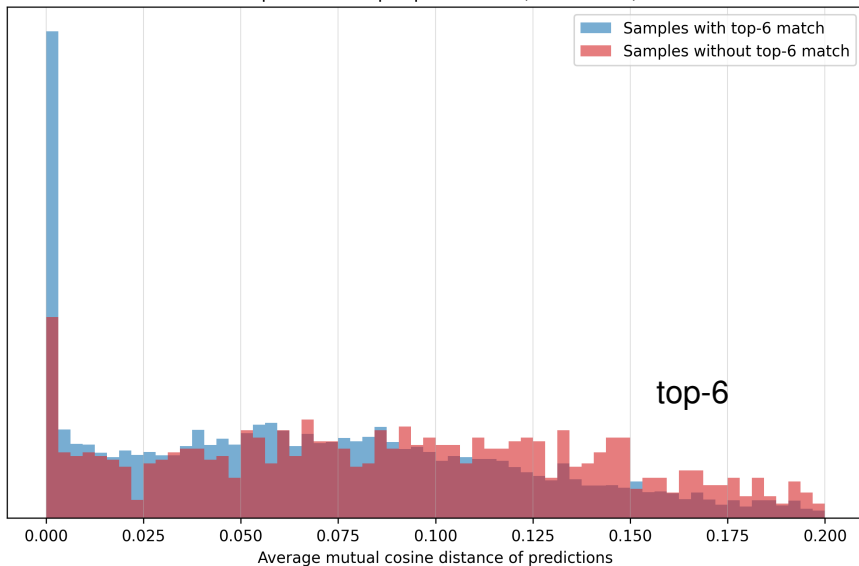
---

<sup>3</sup>Ahmad et al., 2022 [5]

Dispersion of top-3 predictions (normalized)



Dispersion of top-6 predictions (normalized)



Bayes:

$$P(\checkmark \mid d \text{ is small}) = \frac{P(d \text{ is small} \mid \checkmark) P(\checkmark)}{P(d \text{ is small} \mid \checkmark) P(\checkmark) + P(d \text{ is small} \mid \times) P(\times)}$$

Given

$$\frac{P(d \text{ is small} \mid \times)}{P(d \text{ is small} \mid \checkmark)} \approx \frac{1}{2}, \quad P(\checkmark) \approx 80\%, \quad P(\times) \approx 20\%,$$

we have

$$P(\checkmark \mid d \text{ is small}) \approx \frac{80\%}{80\% + \frac{1}{2} \times 20\%} \approx 89\%.$$

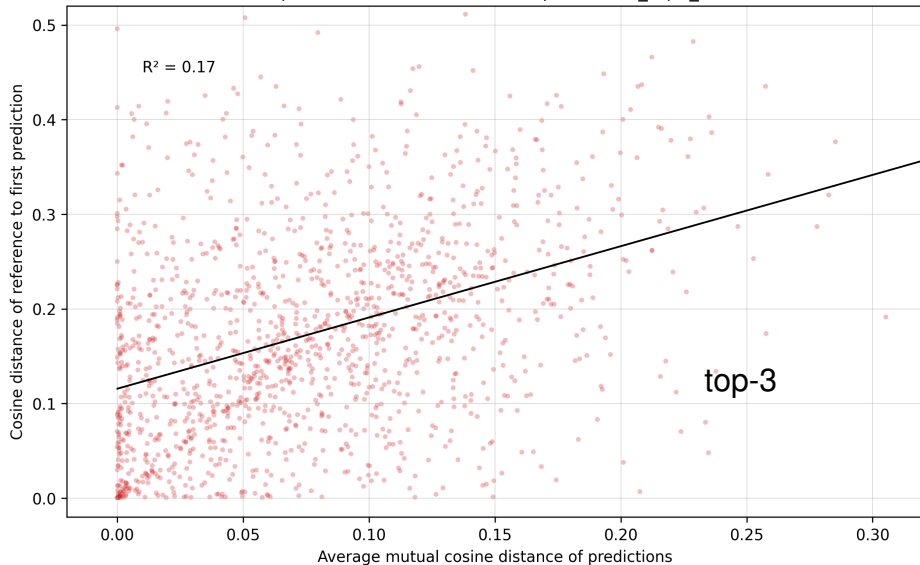
$\rightsquigarrow$  Very small prediction dispersion indicates a top-n match.  
Similarly, high dispersion indicates a mismatch.

# Evaluation

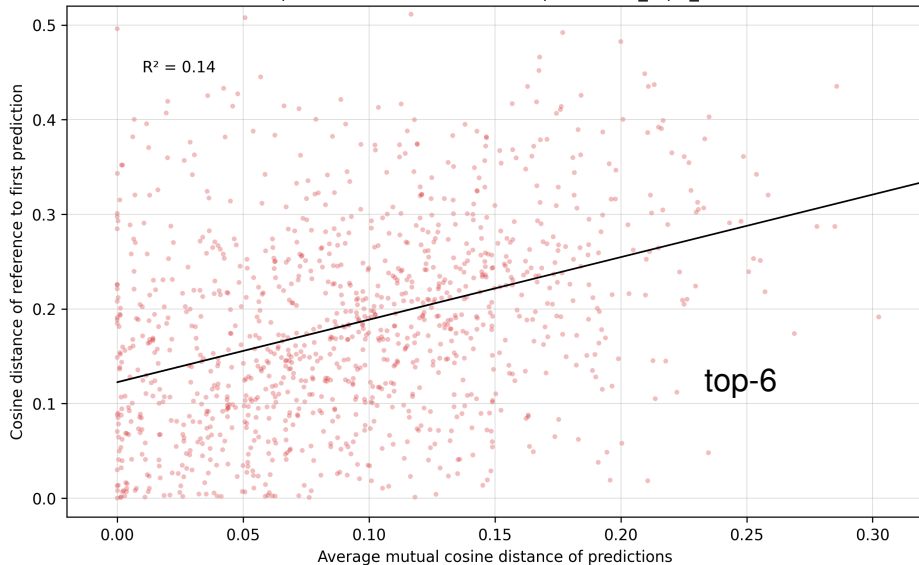
Distance of the top prediction to the reference  
vs  
average distance among top-n predictions



Distance of top-3 to reference vs. their dispersion (is\_topn\_match = False)



Distance of top-6 to reference vs. their dispersion (is\_topn\_match = False)



# Molecular descriptors

We have a baseline confidence of  $\sim 80\%$ .

By looking at chemical properties of the molecules, can we improve our confidence of a top-n match?

In other words, construct a method such that: if it says “match”, we are confident that the transformer model prediction is indeed correct.

This is called “precision”, computed as

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

$\leadsto$  What makes molecules easier/harder to predict?

# Chemical descriptors I

- **Molecular symmetry number:** counts unique canonical atom ranks; a higher value indicates lower overall symmetry. [link]
- **Number of chiral centers:** counts stereogenic atoms that can lead to non-superimposable mirror images. [link]
- **Number of diastereotopic protons:** counts hydrogen atoms in non-equivalent chemical environments (via CIP assignments).
- **Average Gasteiger charge:** computes the mean partial atomic charge using the Gasteiger method, reflecting the electron distribution. [link]
- **Fused ring count:** counts rings that are fused (sharing at least two atoms) with another ring, affecting rigidity and aromaticity.

# Chemical descriptors II

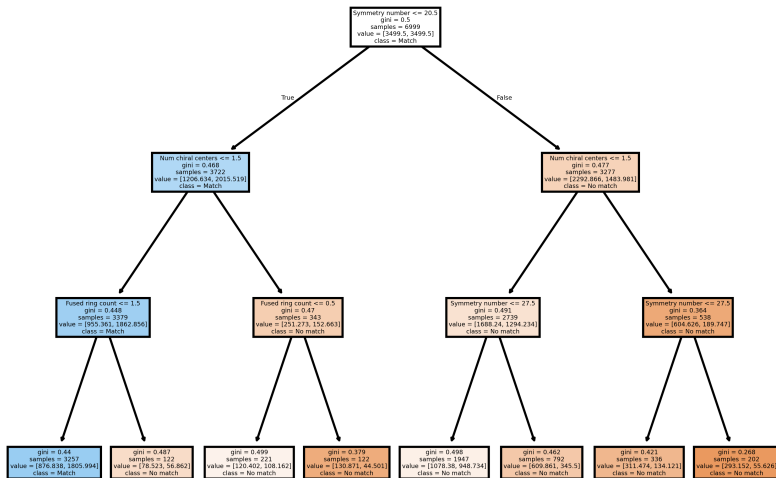
- **Bridgehead protons:** counts hydrogen atoms attached to heavy atoms shared by multiple rings, influencing steric effects and molecular stability.
- **Rotatable bonds:** enumerates bonds that are rotatable (typically single bonds outside rings), determining flexibility. [link]
- **Internal hydrogen bonds:** estimates potential intramolecular hydrogen bonding interactions by pairing donor (N/O with attached H) and acceptor (N/O with degree > 1) atoms.
- **Alpha heteroatom protons:** counts hydrogen atoms attached to carbons adjacent to heteroatoms (O or N).

cf. [code]

# Chemical descriptors – decision tree

We train a “decision tree” on these molecular descriptors to predict whether there is a top-n match. Use the unseen validation dataset.

Caveat: we use the descriptors of the “unknown” reference molecule.



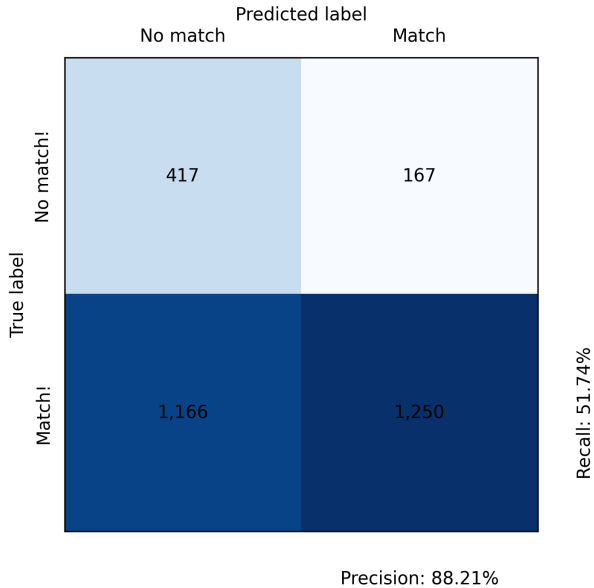
IF:

- Symmetry number  $\leq 20$  (basically, size)
- Num chiral centers  $\leq 1$
- Fused ring count  $\leq 1$

then the decision tree indicates a top-n match.

In this case, we are  $\sim 90\%$  confident that the transformer model got it right.





# In summary

- Slightly better prediction accuracy if:
  - ignore chirality
  - use sum-formula
  - longer training
  - explicit hydrogens
  - beam size 20 instead of 100
  - small prediction dispersion
- Don't understand what makes molecules hard to infer
  - ~> molecule length, chiral centers, fused rings

# Outlook

- Functional groups as descriptors / additional input
- Multimodal input: better data?
- Noisy spectra / multiplets
- Fine-tuning & evaluation on experimental data
- Fine-tuning an off-the-shelf self-supervised model
- “Inductive bias” with graph generation
- Diffusion-like structure generation?
- Cycle consistency: re-simulate spectra from predictions
- Feedback loop (cf. Jonas [6], Devata et al. [7])

thx.

# References I

- [1] Daniel Lowe. *Chemical reactions from US patents (1976–Sep2016)*. figshare, 2017.  
[10.6084/m9.figshare.5104873.v1](https://figshare.com/10.6084/m9.figshare.5104873.v1).
- [2] Marvin Alberts, Oliver Schilter, Federico Zipoli, Nina Hartrampf, and Teodoro Laino. “Unraveling Molecular Structure: A Multimodal Spectroscopic Dataset for Chemistry”. In: *NeurIPS 2024 Datasets and Benchmarks Track*. 2024.  
[10.48550/arXiv.2407.17492](https://arxiv.org/abs/2407.17492).
- [3] Marvin Alberts, Federico Zipoli, and Alain C. Vaucher. *Learning the Language of NMR: Structure Elucidation from NMR spectra using Transformer Models*. 2023.  
[10.26434/chemrxiv-2023-8wxcz](https://chemrxiv.org/10.26434/chemrxiv-2023-8wxcz).

# References II

- [4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. “Attention Is All You Need”. In: *Advances in Neural Information Processing Systems* 30 (2017), pp. 5998–6008. 10.48550/arXiv.1706.03762.
- [5] Walid Ahmad, Elana Simon, Seyone Chithrananda, Gabriel Grand, and Bharath Ramsundar. “ChemBERTa-2: Towards Chemical Foundation Models”. In: *ELLIS Machine Learning for Molecule Discovery Workshop 2021*. 2022. 10.48550/arXiv.2209.01712.
- [6] Eric Jonas. “Deep Imitation Learning for Molecular Inverse Problems”. In: *Advances in Neural Information Processing Systems*. Vol. 32. 2019.

## References III

- [7] Sriram Devata, Bhuvanesh Sridharan, Sarvesh Mehta, Yashaswi Pathak, Siddhartha Laghuvarapu, Girish Varma, and U. Deva Priyakumar. “DeepSPInN – Deep reinforcement learning for molecular structure prediction from infrared and  $^{13}\text{C}$  NMR spectra”. In: *Digital Discovery* (2024). 10.1039/D4DD00008K.

