Atomic-SCS: An Atom-Level Chemical Rule Scoring Tool for Generative Molecular Design

Hejing Huo and Miaomiao Niu *

Posted Date: 13 April 2026

doi: 10.20944/preprints202604.0809.v1

Keywords: atomic-SCS; chemical rule scoring; generative molecular design; atom-level validation; cheminformatics; drug discovery

 doi:10.20944/preprints202604.0809.v1

Article

Atomic-SCS: An Atom-Level Chemical Rule Scoring Tool for Generative Molecular Design

Hejing Huo and Miaomiao Niu *

China Pharmaceutical University, Nanjing, China

* Correspondence: niumm@cpu.edu.cn

Abstract

Generative molecular models such as diffusion models and graph neural networks are widely used in drug design. However, their black-box nature means they lack an explicit understanding of chemical rules (e.g., valency, charge, aromaticity), often generating chemically impossible structures such as pentavalent carbon. To address this issue, this paper proposes Atomic-SCS, an atom-level chemical rule scoring tool based on a symbolic approach. Atomic-SCS does not rely on data-driven training but directly applies IUPAC rules to independently score each atom across four dimensions: valency, charge, aromaticity, and ring strain. It outputs continuous scores (0 = fully compliant, 1 = severe violation), provides atom-level diagnostic reports, and generates prioritized repair suggestions sorted by severity. The tool supports three strictness levels (conservative, balanced, liberal) and three operation modes (assess, diagnose, repair), and can be accessed via a command-line interface or Python API. Validation on 100 normal and 100 problematic molecules shows that Atomic-SCS effectively distinguishes valid from invalid structures (Mann-Whitney U test, $p < 1e-18$). The scoring functions are continuous and can serve as reward signals in generative model training. On a standard CPU, scoring 100 molecules averaged 0.000071 s per molecule. This work provides a rule-based scoring tool for generative molecular design.

Keywords: atomic-SCS; chemical rule scoring; generative molecular design; atom-level validation; cheminformatics; drug discovery

1. Introduction

Deep generative models have shown great potential in molecular design [10]. However, they often produce chemically invalid structures with valency errors, abnormal charges, or broken aromaticity [1,2,8,9]. The most commonly used validation tool, RDKit's SanitizeMol, only provides binary pass/fail judgments, failing to locate problematic atoms or quantify violation severity[3]. The recently released ChemAudit (February 2026) offers molecular-level ML-readiness scores and atom highlighting, but to the best of our knowledge, it still lacks continuous scores and repair suggestions, and its scores are not differentiable [4].

To fill this gap, we present Atomic-SCS, an atom-level, interpretable chemical rule scoring tool. Our main contributions are:

1. Atom-level continuous scores: Based on rules such as quadratic valency penalty and distance-based charge penalty, each atom receives a continuous score between 0 and 1.
2. Atom-level diagnosis and repair suggestions: The tool identifies which atom and which rule is violated, and outputs prioritized repair suggestions as text.
3. Flexible workflows: Three strictness levels and three operation modes are supported for different application scenarios.
4. Open-source and easy to use: Built on Python and RDKit, the tool can be obtained from GitHub and installed via a conda environment.

2. Methods

 doi:10.20944/preprints202604.0809.v1

Atomic-SCS evaluates each atom independently across four dimensions. All dimension scores are output independently without weighted fusion.

- Valency: Valency is checked by comparing the number of connected heavy atoms (atom.GetDegree()) with an expected maximum valence derived from the element and formal charge. For example, carbon has a maximum valence of 4; if it is connected to 5 heavy atoms, the excess is 1. A quadratic penalty is applied: score = min(1.0, (excess^2) * 0.25). This gives 0.25 for one excess and 1.0 for two excesses. This simplification works for organic molecules where hydrogen atoms are implicit.

- Charge: For each element, a predefined list of allowed formal charges is used (e.g., C: -1, 0, 1; N: -2, -1, 0, 1, 2; O: -1, 0, 1; etc.). The tool computes the minimum absolute distance from the atom's actual charge to any allowed charge, then applies a linear penalty: score = min(0.8, min_distance * 0.2). An atom with a disallowed charge receives a penalty proportional to the distance from the nearest allowed charge (capped at 0.8).

- Aromaticity: Relies on RDKit's aromaticity flags. It detects unusual aromatic elements (e.g., sulfur in an aromatic ring) or aromatic atoms that are not part of any ring. Such violations receive a score of 0.2–0.3: an aromatic atom outside any ring receives 0.3, while an atypical aromatic element (e.g., sulfur in an aromatic ring) receives 0.2.

- Ring strain: Only 3- and 4-membered rings are penalized. In balanced mode, a 3-membered ring incurs a total penalty of 0.2, which is evenly distributed among its three atoms (0.0667 per atom); a 4-membered ring incurs a total penalty of 0.1 (0.025 per atom). The total penalty for a ring is evenly distributed among its atoms; if an atom belongs to multiple rings, it takes the maximum penalty among them.

Strictness levels are defined by adjusting compliance thresholds and hypervalence tolerance for sulfur and phosphorus:

- Conservative mode: threshold 0.1, max valence S=4, P=4, 3-ring penalty 0.3, 4-ring penalty 0.15.

- Balanced mode: threshold 0.3, max valence S=6, P=5, 3-ring penalty 0.2, 4-ring penalty 0.1.

- Liberal mode: threshold 0.5, max valence S=6, P=5, 3-ring penalty 0.1, 4-ring penalty 0.05.

Operation modes:

- assess: Outputs molecular score, compliance status, and confidence level.

- diagnose: Outputs per-atom scores and issue descriptions.

- repair: Outputs prioritized repair suggestions as text (no automatic molecular modification).

Score aggregation: The molecular score is computed as the maximum of all atom scores across all rules: molecular_score = max(atom_scores). This design highlights the most severe violation and avoids dilution by large molecules.

3. Results

3.1. Dataset Construction

We constructed two non-overlapping molecule sets:

- Normal set (n = 100): Molecules manually verified to be chemically valid, selected from common drug-like scaffolds. Categories include alkanes (e.g., CC, CCC(C)C), aromatics (e.g., c1ccccc1, c1ccncc1), alcohols and ethers (e.g., CCO, C1CCOC1), carboxylic acids (e.g., CC(=O)O), heterocycles (e.g., c1cncc1, C1CCNC1), and small rings (e.g., cyclopropane, oxetane). All SMILES are provided in the GitHub repository.

- Problematic set (n = 100): Chemically invalid or highly strained structures, collected from known generative model failure cases and manually constructed invalid SMILES. Categories include hypervalency (e.g., pentavalent carbon, trivalent oxygen), extreme formal charges (e.g., carbon with +2), non-aromatic aromatic atoms, ring strain (e.g., cyclopropane,

 doi:10.20944/preprints202604.0809.v1

bicyclo[1.1.0]butane), invalid SMILES syntax, and mixed violations. Full SMILES are also provided in the repository.

Both groups were scored using the balanced strictness mode.

3.2. Scoring Results

The normal group had a median score of 0.0 and a maximum of 0.067 (ring strain in small rings, still below the balanced threshold of 0.3). The problematic group showed a wide distribution from 0 to 1, with a significantly higher median. The Mann-Whitney U test gave $p < 1e\text{-}18$, indicating a highly significant difference.

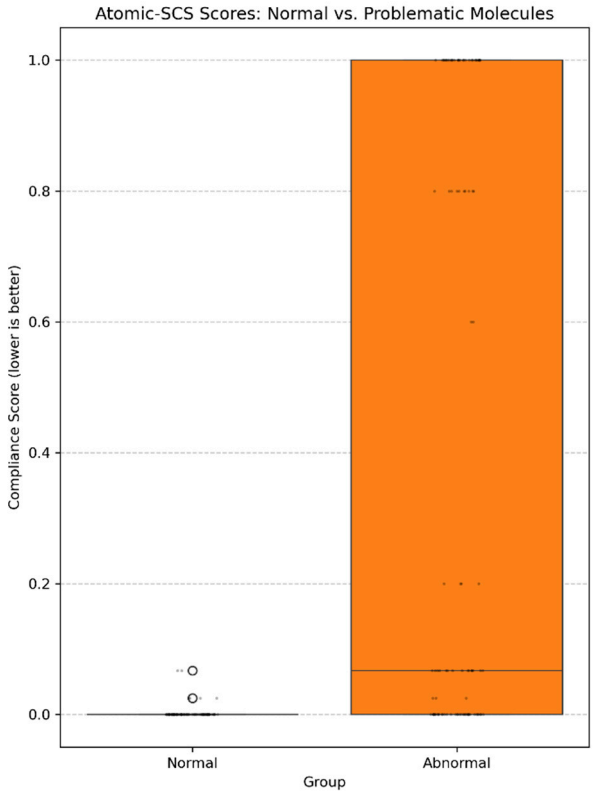

Figure 1. Boxplot of Atomic-SCS scores for normal (n = 100) and problematic (n = 100) molecules. The normal group includes small rings (e.g., cyclopropane, oxetane) which receive non-zero scores due to ring strain; these remain below the balanced-mode compliance threshold of 0.3. The problematic group shows a wide distribution, with severe violations scoring above 0.5. Mann-Whitney U test $p < 1e\text{-}18$.

3.3. Performance

On a single core of an Intel Core i7-14650HX CPU running Ubuntu 22.04 via WSL, scoring 100 molecules from the validation set (average ~30 heavy atoms) took 0.0071 seconds total (0.000071 s per molecule). The tool can easily scale to larger datasets via multiprocessing. A benchmark script (benchmark.py) is provided in the GitHub repository to reproduce this measurement.

4. Discussion and Limitations

Atomic-SCS fills a gap in atom-level chemical rule evaluation. Unlike RDKit's binary judgments and ChemAudit's discrete labels, Atomic-SCS provides continuous scores and prioritized repair suggestions, making it suitable as a constraint signal for generative models [5,6]. Validation results demonstrate its ability to effectively distinguish valid from invalid structures.

Limitations

 doi:10.20944/preprints202604.0809.v1

- Only common elements (H, C, N, O, F, P, S, Cl, Br, I) are fully supported; other elements default to a maximum valence of 4.

- Aromaticity relies on RDKit's built-in algorithm, which may be inaccurate for some fused ring systems.

- Ring strain is only penalized for 3- and 4-membered rings.

- The valency check uses atom.GetDegree() (number of bonded heavy atoms) rather than total valence including hydrogens. This is a simplification suitable for typical organic molecules; for molecules with unusual hydrogen counts, the scoring may be less accurate.

- The repair mode outputs only text suggestions and does not automatically modify molecular structures.

- Installation must be done from source via a conda environment; pip installation is not yet supported.

- Stereochemistry (R/S, E/Z) is ignored; a molecule with wrong chirality will not be penalized.

- Invalid SMILES syntax (e.g., unmatched parentheses) causes immediate failure, returning a score of 1.0.

- Resonance effects are not modeled; carboxylate anions and similar species may receive minor penalties due to formal charge deviations.

Future Work

- Embed the continuous scores of Atomic-SCS as differentiable constraints into generative model training (e.g., as a loss function or reward signal). The scores can be used as rewards in a reinforcement learning framework, or integrated with a differentiable molecular graph encoder.

- Extend the rule library to support more elements and complex chemical patterns.

- Explore neuro-symbolic hybrid models combining the rule base with graph neural networks.

Acknowledgments: The author thanks Prof. Miaomiao Niu for his supervision, guidance, and support throughout this work. The author also thanks the China Pharmaceutical University for providing the research environment.

Data Availability Statement: The source code, documentation, and validation scripts are available on GitHub at https://github.com/huohejing/Atomic-SCS under the MIT License. Validation datasets and scoring results can be generated using the scripts provided in the repository. A benchmark script (benchmark.py) is included to reproduce the performance measurements.

References

1. Olivecrona, M.; Blaschke, T.; Engkvist, O.; Chen, H. Molecular De Novo Design through Deep Reinforcement Learning. J. Cheminform. 2017, 9 (1), 48. DOI: 10.1186/s13321-017-0235-x.
2. Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; Dahl, G. E. Neural message passing for quantum chemistry. In Proceedings of the 34th International Conference on Machine Learning (ICML); 2017; pp 1263-1272. arXiv:1704.01212.
3. RDKit: Open-source cheminformatics. https://www.rdkit.org (accessed 2026-04-07).
4. ChemAudit: An Open-Source Chemical Structure Validation Suite. NFDI4Chem, 2026. https://nfdi4chem.de/introducing-chemaudit/ (accessed 2026-04-05).
5. Geng, C.; Zhang, L.; Zhang, M.; Ye, H.; Zhao, Z.; Si, X. Neural proposals, symbolic guarantees: Neuro-symbolic graph generation with hard constraints. arXiv 2026, arXiv:2602.16954.
6. Christopher, J. K.; Caldei, M.; Liang, J.; Fioretto, F. Neuro-symbolic generative diffusion models for physically grounded, robust, and safe generation. In Proceedings of the Second International Conference on Neuro-Symbolic Systems (NeuS 2025); arXiv:2506.01121.

 doi:10.20944/preprints202604.0809.v1

7. Landrum, G. RDKit: A software suite for cheminformatics, computational chemistry, and predictive modeling. GitHub 2016. https://github.com/rdkit/rdkit
8. Kusner, M. J.; Paige, B.; Hernández-Lobato, J. M. Grammar variational autoencoder. In Proceedings of the 34th International Conference on Machine Learning (ICML); 2017; pp 1945-1954. arXiv:1703.01925.
9. Jin, W.; Barzilay, R.; Jaakkola, T. Junction tree variational autoencoder for molecular graph generation. In Proceedings of the 35th International Conference on Machine Learning (ICML); 2018; pp 2323-2332. arXiv:1802.04364.
10. Elton, D. C.; Boukouvalas, Z.; Fuge, M. D.; Chung, P. W. Deep learning for molecular design—a review of the state of the art. Mol. Syst. Des. Eng. 2019, 4, 828–849. DOI: 10.1039/C9ME00039A.