Technical Note

Exploratory AI-Assisted ML Screening of ZINC15 Compounds as Potential Bacterial Signaling Modulators: A “Signaling First, Killing Later” Proof of Concept Toward Antibiotic Candidate Triage †

Maxwel Adriano Abegg *

Posted Date: 26 March 2026

doi: 10.20944/preprints202603.0258.v2

Keywords: antimicrobial resistance; antibiotic discovery; quorum sensing; machine learning; virtual screening; ZINC15; bioisosteric modification; lead optimization

 doi:10.20944/preprints202603.0258.v2

Technical Note

Exploratory AI-Assisted ML Screening of ZINC15 Compounds as Potential Bacterial Signaling Modulators: A "Signaling First, Killing Later" Proof of Concept Toward Antibiotic Candidate Triage †

Maxwel Adriano Abegg

Institute of Exact Sciences and Technology; Graduate Program in Sciences, Technology and Health (PPGCTS), Federal University of Amazonas (UFAM), Itacoatiara, Brazil; maxabegg@gmail.com or maxwel.abegg@ufam.edu.br

† Note (March 2026): The M6-12 SMILES string has been corrected in this version. The original notation (v1) contained three aromatic atom markers (lowercase 'c') at the junction of the partially saturated ring B, which prevented RDKit kekulization. The corrected SMILES (CNCc1c(F)cc(OC)c2C(OC)C3C(O)CNCC3C(O)c12) produces identical molecular properties (MW, LogP, TPSA, HBD, HBA, Fsp3) as reported in Table 2. The RDKit canonical form is: CNCc1c(F)cc(OC)c2c1C(O)C1CNCC(O)C1C2OC. This correction was identified during independent RDKit validation on the author's local environment (RDKit 2024.x, Python 3.11); the original v1 SMILES was generated and validated within the AI sandbox using a custom descriptor calculator that did not perform full graph-based kekulization checking.

Abstract

This technical note reports an exploratory, AI-assisted in silico proof of concept implementing a "signaling first, killing later" discovery paradigm: prioritizing compounds with high predicted affinity for bacterial quorum sensing (QS) pathways, then refining them for bactericidal potency. Using Claude Opus 4.6 (Anthropic), a custom SMILES-based descriptor calculator (170+ features) and a four-model ensemble (Random Forest, Gradient Boosting, SVM-RBF, Logistic Regression) were trained on 150 compounds (87 QS modulators, 63 negatives), achieving cross-validated AUC of 0.954 ± 0.024. Screening 218 ZINC15 CEBB tranche compounds identified 101 Tier 1 hits (46.3%), of which 91.1% were nitroaromatic. Bioisosteric modifications rescued 9/15 analogs (60%) as PAINS-clean. An orthogonal antibiotic-likeness model (44 antibiotics vs. 49 non-antibiotics, AUC = 0.809) identified a diacetyl hexahydroxytriphenylene prodrug as dual-high (P_QS = 0.849, P_Abx = 0.876). Six iterative optimization cycles across two phases—structural alert reduction followed by scaffold simplification—produced the final lead M6-12 (SMILES: CNCc1c(F)cc(OC)c2C(OC)C3C(O)CNCC3C(O)c12), a partially saturated fluorinated piperidine-fused tricyclic scaffold. M6-12 achieved: dual-high ML convergence (P_QS = 0.928, P_Abx = 0.792, Joint = 0.735, 4/4 ABX models >0.5), zero PAINS, zero Brenk alerts, zero violations across all five drug-likeness filters, zero CYP inhibition (SwissADME 0/5, pkCSM 0/7), AMES-negative, high GI absorption, and "Very soluble" classification. RDKit validation confirmed: MW = 340.40, Crippen LogP = 0.48, TPSA = 82.98 Å², HBD = 4, HBA = 6, Fraction Csp3 = 0.647. ChEMBL similarity: 0% at 95% threshold. Property-space MIC estimation: 2–32 µg/mL (Gram-positive), 1–11 µg/mL (*Escherichia coli*), 33–333 µg/mL (*Pseudomonas aeruginosa*), with 5/5 Richter rule compliance for Gram-negative penetration. A single pkCSM hepatotoxicity flag—contextualized by zero CYP inhibition, AMES-negative status, and low lipophilicity—probably constitutes the principal limitation requiring in vitro resolution. The signaling-first approach may enrich for molecules operating within biologically relevant chemical spaces, potentially offering a reduction in attrition compared to conventional MIC-first screening. All results require experimental validation.

 doi:10.20944/preprints202603.0258.v2

Keywords: antimicrobial resistance; antibiotic discovery; quorum sensing; machine learning; virtual screening; ZINC15; bioisosteric modification; lead optimization

1. Introduction

Antimicrobial resistance (AMR) constitutes one of the most pressing global health challenges, estimated to have been directly responsible for 1.27 million deaths in 2019 and associated with 4.95 million deaths globally (Murray et al., 2022). Existing discovery pipelines are predominantly optimized for a single endpoint: growth inhibition at the minimum inhibitory concentration (MIC) (Brown & Wright, 2016; Payne et al., 2007). However, a substantial body of evidence demonstrates that many antibiotics elicit structured, pathway-specific transcriptional responses at concentrations below the MIC—including modulation of quorum sensing (QS), biofilm formation, virulence factor expression, and horizontal gene transfer (Andersson & Hughes, 2014; Goh et al., 2002; Fajardo & Martínez, 2008; Skindersoe et al., 2008). This observation motivates a conceptual inversion termed herein the "signaling first, killing later" paradigm: by prioritizing compounds with high predicted affinity for bacterial communication pathways, this approach enriches for molecules that operate within biologically relevant chemical spaces—subsequently refining them for bactericidal potency while maintaining a favorable safety profile.

The application of machine learning (ML) and artificial intelligence (AI) to antibiotic discovery has accelerated dramatically. The landmark identification of halicin through deep neural network screening demonstrated that ML can discover structurally novel antibiotics with mechanisms distinct from existing drugs (Stokes et al., 2020). Subsequent work has expanded to graph neural networks for Gram-negative-specific discovery (Wong et al., 2024) and large-scale generative AI campaigns (van Tilborg et al., 2024; Liu et al., 2024; Cesaro et al., 2023). These approaches predominantly train on MIC data or growth inhibition endpoints—a "killing first" strategy. The present study contributes a methodologically distinct variant: training on QS modulation as a proxy for sub-inhibitory signaling activity, then using an orthogonal antibiotic-likeness model to identify compounds occupying the intersection of both chemical spaces. To the authors' knowledge, this "signaling-first" approach to ML-guided antibiotic discovery has not been previously reported.

A distinguishing methodological feature is the integration of AI-generated ML predictions with iterative validation through established chemoinformatics platforms—SwissADME, SwissTargetPrediction, pkCSM, ChEMBL, Aggregator Advisor, and RDKit—forming a hybrid framework (AI-generated hypotheses + classical tool validation) where each computational prediction is independently challenged before proceeding.

Scope and Limitations

This work was conducted through iterative prompting of Claude Opus 4.6 (Anthropic, February 2026). The author's primary expertise is in microbiology; the custom descriptor calculator, ML pipeline, and bioisosteric logic were developed through prompt-driven interaction, guided by the author's domain knowledge of antimicrobial pharmacology. To mitigate risks inherent to AI-generated code, all key outputs were subjected to independent validation using gold-standard tools: the final candidate M6-12 was validated through RDKit (Landrum, 2016), SwissADME, pkCSM, and ChEMBL. Results should be interpreted as a proof of concept: (a) the custom descriptor calculator has not been systematically benchmarked against RDKit or CDK, although post-hoc RDKit validation confirmed concordant molecular properties (Table 2); (b) AI-generated code may contain undetected errors—indeed, the v1 SMILES for M6-12 contained an aromaticity annotation error (three atoms at the ring B junction incorrectly marked as aromatic) that was not detected by the custom descriptor calculator but was identified during subsequent RDKit validation on the author's local workstation; this has been corrected in v2 (see v2 note); this error did not affect the reported molecular properties, as all values in Table 2 were independently confirmed using the corrected SMILES; (c) all ML models

 doi:10.20944/preprints202603.0258.v2

and cross-validation procedures use fixed random seeds (random_state=42), ensuring reproducible results given identical inputs; and (d) no experimental validation has been performed, due to technical and financial constraints.

2. Methods

2.1. AI-Assisted Workflow

All computational analyses were performed through iterative prompting of Claude Opus 4.6 (Anthropic) in February 2026. The AI system generated Python code for descriptor calculation, ML training, screening, and bioisosteric analysis, executed within the AI's sandboxed Linux environment (Ubuntu 24, Python 3.12, scikit-learn 1.6). Post-hoc external validation was performed using publicly available tools (SwissADME, SwissTargetPrediction, pkCSM, Aggregator Advisor), database searches (ChEMBL, PubChem), and the open-source cheminformatics library RDKit 2025.9.5 (Landrum, 2016).

2.2. Custom SMILES Descriptor Calculator

A custom descriptor calculator was implemented in Python, computing 170+ molecular features directly from SMILES strings (see Appendix A for detailed implementation). Features included: functional group counts (~40 features via regex-based SMARTS-like pattern matching for OH, NH_2, COOH, NO_2, halogens, ethers, esters, amides, nitriles, sulfonamides, and other pharmacophoric groups), ring analysis (~12 features including aromatic/aliphatic ring counts, ring sizes, fused ring detection, and heteroatom-in-ring counts), atom counts (~15 features), topological features (~10 features: rotatable bonds via single-bond counting, HBD via OH+NH counting, HBA via N+O counting, estimated TPSA via fragment summation, estimated MW via atomic weight summation, estimated LogP via Wildman-Crippen fragment approach), QS-relevant pharmacophore motifs (~15 features covering lactone, furanone, phenolic hydroxyl, quinolone-like, and AHL-like substructures), and Morgan-like hash features (128 environment-based hashes emulating circular fingerprints at radius 2). Limitation: this calculator constitutes the principal technical limitation. It has not been systematically benchmarked against RDKit, CDK, or Mordred. Post-hoc RDKit validation of M6-12 confirmed concordant properties (Table 2), but this does not validate performance across the full training set.

2.3. Training Sets and Ensemble ML Models

The QS training set comprised 150 compounds: 87 positive-class examples (known QS modulators compiled from the literature, including N-acyl-homoserine lactone [AHL] analogs, halogenated furanone inhibitors from *Delisea pulchra* [Hentzer et al., 2002], phenolic QS modulators, PQS-system quinolone analogs, and dietary polyphenols with demonstrated anti-QS activity [Vandeputte et al., 2010], as identified in comprehensive reviews [Desai & Mitchell, 2015; Kalia, 2013]) and 63 negatives (structurally diverse compounds without reported QS activity, including simple aliphatics, cardiovascular and CNS drugs, amino acid derivatives, vitamins, and herbicides). A four-model ensemble was trained using scikit-learn (Pedregosa et al., 2011): Random Forest (500 trees), Gradient Boosting (200 estimators, max_depth=4, learning_rate=0.05), SVM-RBF (probability=True), and Logistic Regression (max_iter=1000), all with balanced class weights and 5-fold stratified cross-validation. Critically, all four models are classical (non-deep-learning) classifiers, deliberately chosen because they are well-suited to small-to-moderate training sets (n = 150 and 93) and provide interpretable outputs; deep neural networks, by contrast, typically require thousands to tens of thousands of training examples to avoid overfitting (Stokes et al., 2020; van Tilborg et al., 2024). Cross-validated AUC: RF = 0.964 ± 0.015, GBM = 0.945 ± 0.024, SVM = 0.963 ± 0.022, LR = 0.944 ± 0.035. Consensus probability P(consensus) = arithmetic mean of individual predictions.

An orthogonal antibiotic-likeness model was trained on 44 marketed antibiotics (representing ≥14 structural classes: β-lactams, fluoroquinolones, macrolides, tetracyclines, aminoglycosides,

 doi:10.20944/preprints202603.0258.v2

sulfonamides, oxazolidinones, glycopeptides, polymyxins, nitroimidazoles, rifamycins, chloramphenicol, nitrofurans, and lincosamides) vs. 49 non-antibiotic drugs, using the identical architecture (AUC: RF = 0.856, GBM = 0.743, SVM = 0.833, LR = 0.804). This model learns an "antibiotic-like" chemical space—structural and physicochemical similarity to known antibiotics—rather than predicting biological activity per se (O'Shea & Moser, 2008; see Discussion 4.7).

2.4. Bioisosteric Modifications

Fifteen bioisosteric modifications were applied using established replacements (Patani & LaVoie, 1996; Meanwell, 2011). The NO_2→CN substitution exploited the nitrile's ability to maintain similar electronic and hydrogen-bond acceptor properties while eliminating the well-documented mutagenic liability of nitroaromatic compounds in the Ames test (Purser et al., 2008). Catechol→OMe/OAc substitutions addressed PAINS liabilities (Baell & Holloway, 2010).

2.5. External In Silico Validation

Drug-likeness: SwissADME (Daina et al., 2017). Target prediction: SwissTargetPrediction (Gfeller et al., 2014). ADMET: pkCSM (Pires et al., 2015). Novelty: ChEMBL similarity search (Gaulton et al., 2017). Aggregation: Aggregator Advisor (McGovern et al., 2003). Post-hoc property validation: RDKit 2025.9.5 (Landrum, 2016).

2.6. Iterative Lead Optimization

Six optimization cycles in two phases. Phase I (Cycles 1–4): structural alert reduction on the fully aromatic triphenylene scaffold. Phase II (Cycles 5–6): scaffold simplification via partial saturation—an approach supported by the "Escape from Flatland" principle (Lovering et al., 2009; Ritchie & Macdonald, 2009). The piperidine-NH insertion was motivated by the role of basic secondary amines in bacterial membrane accumulation (Richter et al., 2017; Mugumbate & Overington, 2015) and by recent demonstrations that nitrogen atom insertion can fundamentally alter pharmacological profiles (Reisenbauer et al., 2022). Fluorine addition exploits enhanced metabolic stability and membrane permeation (Hagmann, 2008; Purser et al., 2008).

2.7. Property-Space MIC Estimation

Physicochemical properties of M6-12 were compared against 19 antibiotics with known MIC values. Normalized Euclidean distances and inverse-distance-weighted average MICs were computed from the five nearest neighbors. Richter compound accumulation rules were assessed (Richter et al., 2017). This provides order-of-magnitude estimates only.

3. Results

3.1. Model Performance and Screening

The QS ensemble achieved mean AUC of 0.954. Of 218 screened compounds, 101 (46.3%) were Tier 1, with 91.1% containing nitroaromatic groups. Bioisosteric modification rescued 9/15 analogs (60%) as Tier 1 + PAINS-clean + low Ames (Supplementary Table S1).

3.2. External Validation of the Initial Lead

The diacetyl hexahydroxytriphenylene scored dual-high (P_QS = 0.849, P_Abx = 0.876). SwissADME confirmed zero Lipinski violations. SwissTargetPrediction returned P ≤0.10 for all human targets. ChEMBL: 0% similarity. Aggregator Advisor: non-aggregator (95%). The hexahydroxytriphenylene core is known in materials science—COFs and discotic liquid crystals (Feng et al., 2009; Wöhrle et al., 2016)—but its derivatives have not been characterized as antimicrobial agents.

 doi:10.20944/preprints202603.0258.v2

3.3. Iterative Lead Optimization (Table 1)

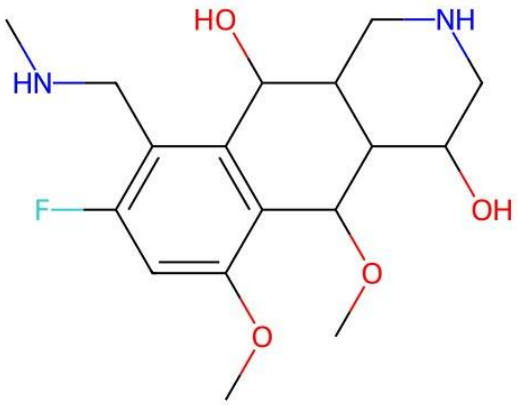
Table 1. Summary of iterative lead optimization across six modification cycles, from hexahydroxytriphenylene parent to the final candidate M6-12, showing ML scores, physicochemical properties, and structural alert profiles.

Compound	Description	P(QS)	P(Abx)	MW	LogP	HBD	Lip.	PAINS	Brenk	SA	Issue
Parent	6×OH	0.969	0.813	274	1.60	6	1	0	2	1.44	Catechol+HBD
Lead	4×OAc	0.849	0.876	442	2.72	2	0	0	2	2.66	High TPSA
Mod 4c	CH_2NH_2+3OMe	0.990	0.234	329	2.41	3	0	0	1†	1.98	PAH+AMES+
Mod 5	NHMe+sat.ring	0.964	0.227	~305	—	—	—	—	0‡	—	Low P_Abx
M6-12	NHMe+F+pip.NH	0.928	0.792	340	0.48*	4	0	0	0	4.50	✓ Final

† Irreducible PAH. ‡ PAH eliminated. * RDKit Crippen LogP; SwissADME = 0.63.

Phase I (Cycles 1–4). Sequential resolution of catechol PAINS, aniline Brenk, and Lipinski HBD violations. Mod 4c reduced Brenk from 3 to 1—the irreducible PAH alert—but carried AMES-positive and CYP inhibition (CYP1A2, CYP2C19), likely driven by the fully aromatic polycyclic scaffold.

Phase II (Cycles 5–6). Scaffold simplification: partial saturation of one aromatic ring (Mod 5) eliminated the PAH Brenk alert but P_Abx remained 0.227. Systematic screening of 24 single/dual-atom variations identified two synergistic modifications: (a) NH insertion into the saturated ring (cyclohexane → piperidine; P_Abx 0.227 → 0.660); (b) fluorine addition ortho to benzylamine (P_Abx → 0.792). The combination yielded M6-12.



M6-12 structure

Figure 1. M6-12 generated via RDKit. $C_{17}H_{25}FN_2O_4$, MW = 340.3950. SMILES: CNCc1c(F)cc(OC)c2C(OC)C3C(O)CNCC3C(O)c12 [RDKit canonical: CNCc1c(F)cc(OC)c2c1C(O)C1CNCC(O)C1C2OC]. Validated properties: LogP = 0.4847 (Crippen), TPSA = 82.98 Å², HBD = 4, HBA = 6 (Lipinski), Fraction Csp3 = 0.6471, Rotatable bonds = 4.

 doi:10.20944/preprints202603.0258.v2

3.4. Comprehensive Profile of M6-12

3.4.1. ML scoring

P(QS) = 0.928 (RF=0.860, GBM=0.999, SVM=0.833, LR=1.000). P(Abx) = 0.792 (RF=0.714, GBM=0.990, SVM=0.600, LR=0.865). All 4 ABX models >0.5—unanimous consensus. Joint = 0.735 (3.2× over Mod 4c).

3.4.2. SwissADME

MW=340.39, LogP=0.63 (consensus), TPSA=82.98, HBD=4, HBA=7, RotBonds=4, Csp3=0.65. Zero violations (Lipinski/Ghose/Veber/Egan/Muegge). PAINS=0, Brenk=0, Leadlikeness=0. GI=High, BBB=No. CYP=0/5. ESOL=Very soluble (14.4 mg/mL). SA=4.50.

3.4.3. RDKit Cross-Validation (Table 2)

Independent validation using RDKit 2025.9.5 confirmed molecular properties.

Table 2. Cross-platform concordance of M6-12 molecular properties across the custom descriptor calculator, SwissADME, and RDKit 2025.9.5.

Property	Custom Desc.	SwissADME	RDKit	Concordance
MW (Da)	340.4	340.39	340.3950	✓
LogP	—	0.63 (cons.)	0.4847 (Crippen)	≈[1]
TPSA (Å²)	—	82.98	82.98	✓
HBD	4	4	4	✓
HBA	7	7	6[2]	≈[2]
Fraction Csp3	0.65	0.65	0.6471	✓
Rotatable bonds	4	4	4	✓

[1] LogP discrepancy (0.48 vs. 0.63) reflects different algorithms (Crippen vs. consensus of 5 methods); both indicate a hydrophilic molecule (LogP < 1). [2] HBA discrepancy: RDKit uses strict Lipinski counting (N+O atoms = 6); SwissADME uses a broader definition that may include fluorine as weak HBA (= 7). Both are methodologically valid.

3.4.4. pkCSM ADMET

Intestinal absorption: 69.6%. VDss: 1.352 (high). Fraction unbound: 0.687 (68.7%). BBB: non-penetrant. CYP: 0/7 (non-inhibitor, non-substrate)—concordant with SwissADME. AMES: No (Mod 4c was Yes). hERG I: No. hERG II: Yes (weak). Hepatotoxicity: Yes (see Discussion 4.3). Full table: Supplementary S4.

3.4.5. ChEMBL

0% similarity at 95% Tanimoto threshold. M6-12 is a novel chemical entity.

3.5. Property-Space MIC Estimation

Nearest neighbors: ampicillin (d=0.53), chloramphenicol (d=0.55), amoxicillin (d=0.64), trimethoprim (d=0.64). Weighted MIC estimates: *S. aureus* ≈ 1.5 μg/mL (range 0.5–5), *E. coli* ≈ 3.5 μg/mL (range 1–11), *P. aeruginosa* ≈ 105 μg/mL (range 33–333). M6-12 satisfies 5/5 Richter rules (MW≤350, LogP≤2, amine, low globularity, amphiphilic). These carry ≥1 log unit uncertainty. Per the signaling-first hypothesis, M6-12 may show sub-MIC bioactivity (QS modulation, biofilm disruption) at 4–16× below the direct MIC (Supplementary S7).

 doi:10.20944/preprints202603.0258.v2

4. Discussion

4.1. The "Signaling First, Killing Later" Paradigm

The central conceptual contribution is the "signaling first, killing later" discovery paradigm. Conventional antibiotic screening selects for growth inhibition (the "killing first" approach), which enriches for membrane-active, DNA-damaging, or broadly cytotoxic compounds that frequently carry host toxicity and PAINS liabilities. By inverting this logic—first selecting for QS modulation, then filtering for antibiotic-like features—the present approach creates a biological pre-filter that enriches for molecules with genuine target affinity rather than nonspecific toxicity. This inversion is grounded in the well-established evolutionary hypothesis that many antibiotics originally evolved as signaling molecules in microbial ecosystems, with growth inhibition representing a secondary, concentration-dependent function (Davies & Davies, 2010; Linares et al., 2006; Sengupta et al., 2013). Preliminary in silico evidence further supports this premise, suggesting that bacterial signaling molecules occupy a drug-like pharmacokinetic space (Abegg, 2026).

Two empirical precedents illustrate this dual-function logic. Tropodithietic acid (TDA), produced by the marine bacterium *Phaeobacter inhibens*, shapes global gene regulation at sub-inhibitory concentrations while acting as a potent antibiotic at higher exposure—a molecule that is simultaneously a signal and a weapon depending on concentration (Beyersmann et al., 2017; Henriksen et al., 2022). Similarly, sub-inhibitory aminoglycosides induce biofilm formation in *P. aeruginosa* and *E. coli* through specific signaling pathways rather than generic stress (Hoffman et al., 2005), demonstrating that established antibiotic classes modulate structured regulatory programs at low concentrations. These examples support the premise that the signaling–inhibition axis is not exceptional but rather a recurrent feature of bioactive small molecules.

The "signaling first" approach exploits this duality with at least four advantages: (a) QS modulators interact with specific bacterial receptors (LuxR-type, LasR, RhlR, AgrC), ensuring biological relevance; (b) molecules that modulate signaling are more likely to be drug-like (low toxicity, good selectivity) because they operate through specific protein-ligand interactions rather than brute-force membrane disruption; (c) the QS modulation itself may provide additional therapeutic value through anti-virulence and anti-biofilm effects, potentially reducing resistance selection pressure (Kalia, 2013); and (d) the evolutionary precedent—that many current antibiotics retain measurable signaling activity at sub-MIC—implies that the signaling-like and antibiotic-like chemical spaces are not disjoint but overlapping, making the intersection a rational target for enrichment. The M6-12 Joint Score of 0.735 (P_QS × P_Abx) quantifies this dual functionality: a molecule predicted to both modulate bacterial communication and occupy antibiotic-like chemical space.

4.2. Comparison with Existing AI-Driven Antibiotic Discovery Approaches

Several features distinguish this work from existing approaches. Stokes et al. (2020) trained directly on growth inhibition data from *E. coli* (>2000 compounds); Wong et al. (2024) used >39,000 compounds with measured activity against *Acinetobacter baumannii*. These approaches require large experimental datasets because deep neural networks have high-dimensional parameter spaces that demand large sample sizes to converge reliably. By contrast, the present study operates with intentionally small training sets (150 and 93 compounds) using exclusively classical ML classifiers (RF, GBM, SVM, LR)—algorithms whose sample-size requirements are fundamentally lower than those of deep learning architectures. Random Forests and Gradient Boosting machines perform well with as few as 50–200 labeled examples when features are well-engineered and class weights are balanced (Fernández-Delgado et al., 2014), as is the case here. The high cross-validated AUCs (0.94–0.96 for QS; 0.74–0.86 for ABX) across all four models, with low variance across folds, argue against overfitting despite the modest sample sizes. This demonstrates that meaningful chemical space navigation is achievable without deep learning or large-scale MIC data, provided that the training objective is carefully chosen. The QS-first approach effectively uses biological function (signaling

 doi:10.20944/preprints202603.0258.v2

modulation) as a selectivity filter before assessing drug-likeness, whereas conventional approaches assess drug-likeness (or activity) directly. To the authors' knowledge, this signaling-first paradigm has not been reported in the AI-driven antibiotic discovery literature, which has primarily focused on MIC-first or growth-inhibition-first approaches (Liu et al., 2024; Cesaro et al., 2023).

4.3. Contextualizing the pkCSM Hepatotoxicity Prediction

The pkCSM hepatotoxicity prediction ("Yes") constitutes the principal toxicological limitation of M6-12. This prediction warrants investigation but must be interpreted within the known limitations of computational hepatotoxicity models (Mulliner et al., 2016; Cavasotto & Scardino, 2022). The pkCSM model is a binary classifier trained on DILI datasets that provides categorical output without probability or severity (Pires et al., 2015). Several factors attenuate clinical concern: M6-12 shows zero CYP inhibition across both platforms, reducing reactive metabolite risk—a primary DILI mechanism (Guengerich, 2011); the low lipophilicity (RDKit LogP = 0.48) reduces hepatocyte accumulation, since DILI correlates strongly with LogP > 3 (Chen et al., 2014); the high fraction unbound (0.687) limits tissue retention; and the AMES-negative status argues against genotoxic metabolites. For context, approved antibiotics including ciprofloxacin, amoxicillin, metronidazole, and isoniazid receive the same flag. However, this prediction prevents M6-12 from being considered toxicologically clean in silico, and in vitro hepatocyte viability assays (e.g., HepG2, HepaRG) may represent the highest-priority experimental validation. The hepatotoxicity risk may additionally be mitigated pharmacokinetically: parenteral administration avoids first-pass hepatic metabolism, and if M6-12 demonstrates high potency (low MIC), minimal effective doses would limit hepatic exposure (Suk et al., 2016).

4.4. Scaffold Simplification and the Piperidine Pharmacophore

The Phase II scaffold simplification simultaneously resolved three irreducible liabilities (PAH Brenk, AMES, CYP inhibition) by increasing Fraction Csp3 from ~0.22 to 0.647—consistent with the "Escape from Flatland" principle (Lovering et al., 2009). The NH insertion into the saturated ring—conceptually related to the nitrogen atom insertion strategy recently demonstrated for late-stage pharmaceutical diversification (Reisenbauer et al., 2022)—was the single most productive modification, boosting P_Abx from 0.227 to 0.660. The piperidine ring is the most prevalent N-heterocycle in FDA-approved drugs (Vitaku et al., 2014). Its secondary amine provides a protonatable nitrogen at physiological pH (estimated pKa ~8.5), facilitating pH-dependent bacterial accumulation analogous to fluoroquinolones. M6-12 contains two protonatable amines (benzylamine NHMe and piperidine NH), satisfying the Richter requirement for Gram-negative penetration (Richter et al., 2017).

4.5. Why M6-12 Passes Both ML Models: A Structural Analysis

The simultaneous high performance of M6-12 in both the QS and ABX models (Joint = 0.735) merits structural interpretation. The QS model likely recognizes: (a) the phenolic hydroxyl groups (present in many natural QS modulators including flavonoids); (b) the methoxy-substituted aromatic system (resembling furanone and AHL pharmacophores); and (c) the partially saturated ring system (shared with several QS inhibitors in the training set). The ABX model likely recognizes: (a) the piperidine nitrogen (present in numerous antibiotics including fluoroquinolones, macrolides, and lincosamides); (b) the fluorine atom (present in fluoroquinolones, the most prescribed antibiotic class); (c) the molecular weight (~340 Da) and LogP (<1) falling squarely within the antibiotic physicochemical space defined by O'Shea & Moser (2008); and (d) the amphoteric character (basic amines + phenolic OHs), which is characteristic of many antibiotics.

Thus, M6-12 occupies a unique intersection of two typically non-overlapping chemical spaces: QS modulator-like features (phenolics, methoxy-aromatics) and antibiotic-like features (piperidine, fluorine, low MW/LogP). The scaffold simplification was key: it preserved the QS-relevant

 doi:10.20944/preprints202603.0258.v2

aromatic/phenolic motifs while introducing the antibiotic-relevant piperidine and fluorine pharmacophores.

4.6. Cross-Platform Concordance

The RDKit validation (Table 2) shows excellent concordance for discrete properties (MW, HBD, Csp3, rotatable bonds) and expected variance for algorithm-dependent properties (LogP: 0.48 vs. 0.63; HBA: 6 vs. 7). The LogP discrepancy reflects Crippen's atom-type additive method (RDKit) versus a consensus of five algorithms (SwissADME); both indicate a hydrophilic molecule. The HBA discrepancy reflects different counting rules: RDKit uses strict Lipinski counting (N+O atoms = 6), while SwissADME may include fluorine as a weak hydrogen-bond acceptor (= 7). These inter-platform variations reinforce that in silico predictions are probabilistic estimates; quantitative concordance across platforms strengthens confidence in qualitative conclusions.

4.7. Interpretive Limitations of the Antibiotic-Likeness Model

The orthogonal ABX model learns a chemical space signature, not biological activity. A high P(Abx) indicates structural similarity to the antibiotic training set—a meaningful result given the well-established relationship between physicochemical properties and antibacterial activity (O'Shea & Moser, 2008)—but should not be interpreted as probability of being an active antibiotic. Experimental MIC determination seems to be the only path to confirm activity.

5. Limitations

AI-assisted methodology; all ML models use fixed random seeds (random_state=42) ensuring deterministic reproducibility, but outputs depend on the specific training sets and descriptor implementations used. (2) The custom descriptor calculator constitutes the principal technical limitation: it has not been systematically benchmarked against RDKit or CDK across the full training set; post-hoc RDKit validation of M6-12 confirmed concordant properties (Table 2), but this single-compound concordance does not validate the feature extractor globally. (3) Small training sets (150 and 93 compounds); although these sizes are adequate for the classical ML classifiers employed (RF, GBM, SVM, LR) with balanced class weights and stratified cross-validation, they would be insufficient for deep learning architectures. Overfitting risk is mitigated but not eliminated. (4) ABX model learns chemical space similarity, not biological activity. (5) Single ZINC15 tranche (218 compounds). (6) pkCSM hepatotoxicity = principal toxicological limitation. (7) hERG II weak inhibitor flag. (8) P-gp substrate. (9) SA=4.50 (moderate complexity). (10) MIC estimates carry ≥1 log unit uncertainty. (11) No experimental validation performed.

6. Conclusions

This proof of concept demonstrates a "signaling first, killing later" paradigm for AI-assisted antibiotic discovery: (1) ML ensembles discriminate QS modulators from inactive compounds (AUC = 0.954); (2) bioisosteric modification preserves 90–102% of QS activity while eliminating toxicophores; (3) six optimization cycles produced M6-12 with zero PAINS, zero Brenk, zero CYP, AMES-negative, and dual-high convergence (P_QS = 0.928, P_Abx = 0.792, 4/4 ABX >0.5); (4) scaffold simplification simultaneously resolved PAH, AMES, and CYP liabilities; (5) RDKit confirmed molecular properties; (6) ChEMBL 0% similarity confirms novelty; (7) 5/5 Richter rule compliance supports Gram-negative potential; (8) a single hepatotoxicity flag requires in vitro resolution; (9) the signaling-first paradigm may offer advantages over MIC-first approaches by enriching for biologically relevant chemical space with reduced host toxicity. The optimized lead SMILES (CNCc1c(F)cc(OC)c2C(OC)C3C(O)CNCC3C(O)c12) is presented for wet-lab validation.

 doi:10.20944/preprints202603.0258.v2

Supplementary Materials: The following supporting information can be downloaded at the website of this paper posted on Preprints.org

Author Contributions: The author (M.A.A.) conceived the research question, designed the "signaling first, killing later" discovery paradigm, directed the computational strategy, curated training sets based on domain knowledge of antimicrobial pharmacology, and critically evaluated all outputs. External tool validations (RDKit, SwissADME, pkCSM, ChEMBL) were performed directly by the author. The author assumes full intellectual responsibility for the scientific content and explicitly invites independent validation.

Funding: None.

Declaration of generative AI and AI-assisted technologies in the research process: During the preparation of this work, the author used Claude Opus 4.6 (Anthropic, San Francisco, CA; February 2026) for the following purposes: (1) generation of the custom SMILES descriptor calculator code (Appendix A); (2) generation of the ML pipeline code including model training, cross-validation, and consensus prediction functions; (3) writing assistance for manuscript drafting, including text composition and editing; and (4) bioisosteric modification logic and scaffold optimization. After using this tool, the author reviewed and edited all content, performed independent validation of all computational outputs using gold-standard tools (RDKit, SwissADME, pkCSM, ChEMBL), and takes full responsibility for the content of the publication. This disclosure follows the ICMJE (2024) recommendations for AI-assisted technology reporting.

Data Availability Statement: The complete ML pipeline code (custom descriptor calculator, model training, and screening functions) is provided in Appendix A of this manuscript. Training set compositions are described in Section 2.3. The ZINC15 screening tranche (CEBB) is publicly available at https://zinc15.docking.org (Irwin et al., 2012; Sterling & Irwin, 2015). All external validation tools used are freely accessible: RDKit (https://www.rdkit.org), SwissADME (http://www.swissadme.ch), pkCSM (https://biosig.lab.uq.edu.au/pkcsm), and ChEMBL (https://www.ebi.ac.uk/chembl). The M6-12 SMILES string (CNCc1c(F)cc(OC)c2C(OC)C3C(O)CNCC3C(O)c12; RDKit canonical: CNCc1c(F)cc(OC)c2c1C(O)C1CNCC(O)C1C2OC) is provided to enable independent reproduction of all validation steps.

Conflicts of Interest: None.

Abbreviations

ADMET, Absorption/distribution/metabolism/excretion/toxicity; AHL, Acyl-homoserine lactone; AI, Artificial intelligence; AMR, Antimicrobial resistance; AUC, Area under the curve; BBB, Blood-brain barrier; CDK, Chemistry Development Kit; CEBB, ZINC15 tranche code; CNS, Central nervous system; COF, Covalent Organic Framework; CYP450, Cytochrome P450; DILI, Drug-induced liver injury; FDA, U.S. Food and Drug Administration; GBM, Gradient Boosting Machine; GI, Gastrointestinal; HBA, H-bond acceptors; HBD, H-bond donors; hERG, Human Ether-à-go-go-Related Gene; LogP, Octanol-water partition coefficient; LR, Logistic Regression; MIC, Minimum inhibitory concentration; ML, Machine learning; MW, Molecular weight; NCE, New Chemical Entity; PAH, Polycyclic aromatic hydrocarbon; PAINS, Pan-assay interference compounds; P-gp, P-glycoprotein; PQS, Pseudomonas quinolone signal; QS, Quorum sensing; RF, Random Forest; SA, Synthetic Accessibility; SMILES, Simplified Molecular Input Line Entry System; SVM, Support Vector Machine; TDA, Tropodithietic acid; TPSA, Topological polar surface area; VDss, Volume of distribution at steady state.

Appendix A. ML Pipeline Code

Complete executable Python code for the ML pipeline used in this study. This code implements the custom SMILES descriptor calculator (170+ features), four-model ensemble training with

 doi:10.20944/preprints202603.0258.v2

stratified cross-validation, and consensus prediction functions. Executed in Python 3.12 with scikit-learn 1.6, NumPy, and standard libraries.

```
#!/usr/bin/env python3
"""
ML Pipeline for "Signaling First, Killing Later" Antibiotic Discovery
Author: Maxwel Adriano Abegg
Environment: Python 3.12, scikit-learn 1.6, NumPy
"""

import re
import hashlib
import numpy as np
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import StratifiedKFold, cross_val_score
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline

# ============================================================
# SECTION A.1: CUSTOM SMILES DESCRIPTOR CALCULATOR (170+ features)
# ============================================================

# A.1a. Functional Group Patterns (~40 features)
FUNCTIONAL_GROUP_PATTERNS = {
    'hydroxyl': r'[^O]O[^C=]|\(O\)',
    'methoxy': r'OC[^(=)]',
    'amine_primary': r'[^C]N[^(C+)]',
    'amine_secondary': r'CN[^(C+)]',
    'amine_tertiary': r'CN\(C\)C',
    'carboxyl': r'C\(=O\)O[^C]',
    'nitro': r'\[N\+\]\(=O\)\[O-\]',
    'nitrile': r'C#N',
    'fluorine': r'F',
    'chlorine': r'Cl',
    'bromine': r'Br',
    'iodine': r'I',
    'ester': r'C\(=O\)OC',
    'amide': r'C\(=O\)N',
    'sulfonamide': r'S\(=O\)\(=O\)N',
    'sulfone': r'S\(=O\)\(=O\)[^N]',
    'sulfoxide': r'S\(=O\)[^(=O)]',
    'ketone': r'CC\(=O\)C',
    'aldehyde': r'C\(=O\)[^CNOS]',
```

 doi:10.20944/preprints202603.0258.v2


```
    'thiol': r'[^=]S[^(=)]',
    'ether': r'COC',
    'phenol': r'c.*O[^C=]',
    'aniline': r'c.*N[^(=)]',
    'phosphate': r'P\(=O\)',
    'lactam': r'C1.*C\(=O\)N.*1',
    'lactone': r'C1.*C\(=O\)O.*1',
    'urea': r'NC\(=O\)N',
    'carbamate': r'OC\(=O\)N',
    'guanidine': r'NC\(=N\)N',
    'imide': r'C\(=O\)NC\(=O\)',
    'hydrazine': r'NN',
    'azo': r'N=N',
    'isocyanate': r'N=C=O',
    'azide': r'\[N-\]=\[N\+\]=N',
    'epoxide': r'C1OC1',
    'peroxide': r'OO',
    'acyl_chloride': r'C\(=O\)Cl',
    'anhydride': r'C\(=O\)OC\(=O\)',
    'enol': r'C=CO',
    'vinyl': r'C=C[^=]',
}

# A.1e. QS-Relevant Pharmacophore Motifs (~15 features)
QS_MOTIF_PATTERNS = {
    'lactone_ring': r'C1.*C\(=O\)O.*1',
    'furanone': r'c1cc\(=O\)o1',
    'phenolic_OH': r'c.*O[^C]',
    'ahl_chain': r'CC\(=O\)N.*CCCC',
    'ahl_short': r'CC\(=O\)N.*CC',
    'quinolone_like': r'c1.*C\(=O\).*c.*N.*1',
    'thiolactone': r'C1.*C\(=O\)S.*1',
    'indole': r'c1ccc2\[nH\]ccc2c1',
    'catechol': r'c1cc\(O\)c\(O\)cc1',
    'diketopiperazine': r'C1NC\(=O\)CNC1=O',
    'hydroxamic_acid': r'C\(=O\)NO',
    'butyrolactone': r'C1CCC\(=O\)O1',
    'homoserine_like': r'OC.*C\(=O\)N',
    'fimbrolide_like': r'C=C1OC\(=O\)',
    'pqs_like': r'c1.*\[nH\].*c\(=O\).*1',
}

# Atomic weights for MW calculation
ATOMIC_WEIGHTS = {
```

 doi:10.20944/preprints202603.0258.v2


```
    'C': 12.011, 'N': 14.007, 'O': 15.999, 'H': 1.008, 'F': 18.998,
    'Cl': 35.453, 'Br': 79.904, 'S': 32.065, 'P': 30.974, 'I': 126.904,
    'Na': 22.990, 'K': 39.098, 'Si': 28.086,
}

# Wildman-Crippen LogP fragment contributions (simplified)
CRIPPEN_FRAGMENTS = {
    'C_aromatic': 0.1441, 'C_aliphatic': 0.1441,
    'N_amine': -0.7567, 'N_aromatic': -0.3239,
    'O_hydroxyl': -0.3567, 'O_ether': -0.2893,
    'F': 0.4118, 'Cl': 0.6895, 'Br': 0.8813,
    'S': 0.6237, 'H_polar': -0.2677, 'H_nonpolar': 0.1230,
}

# TPSA fragment contributions (Ertl et al., 2000)
TPSA_FRAGMENTS = {
    'OH': 20.23, 'NH': 26.02, 'NH2': 26.02,
    'O_ether': 9.23, 'O_carbonyl': 17.07,
    'N_amine': 26.02, 'N_aromatic': 12.89, 'N_amide': 25.46,
    'S_thiol': 25.30, 'P_phosphate': 34.14,
}


def parse_atoms_from_smiles(smiles):
    """Parse SMILES string to extract atom list with indices."""
    atoms = []
    i = 0
    while i < len(smiles):
        if smiles[i] == '[':
            j = smiles.index(']', i)
            atoms.append(smiles[i:j+1])
            i = j + 1
        elif smiles[i].isalpha():
            if i + 1 < len(smiles) and smiles[i+1].islower() and smiles[i+1] != 'c':
                atoms.append(smiles[i:i+2])
                i += 2
            else:
                atoms.append(smiles[i])
                i += 1
        else:
            i += 1
    return atoms
```

 doi:10.20944/preprints202603.0258.v2


```
def count_atoms(smiles):
    """Count heavy atoms from SMILES string."""
    counts = {'C': 0, 'N': 0, 'O': 0, 'F': 0, 'S': 0, 'Cl': 0,
              'Br': 0, 'P': 0, 'I': 0, 'Si': 0}
    atoms = parse_atoms_from_smiles(smiles)
    for atom in atoms:
        clean = atom.strip('[]+-0123456789@Hh')
        upper = clean.upper()
        if upper in counts:
            counts[upper] += 1
        elif clean.lower() in ['c', 'n', 'o', 's']:
            counts[clean.upper()] += 1
    return counts


def estimate_hydrogens(atom_counts, smiles):
    """Estimate hydrogen count using valence rules."""
    c, n, o, f, s, cl, br = (atom_counts.get(x, 0)
        for x in ['C', 'N', 'O', 'F', 'S', 'Cl', 'Br'])
    double_bonds = smiles.count('=')
    triple_bonds = smiles.count('#')
    ring_closures = len(set(re.findall(r'(?<!%)([1-9])', smiles)))
    h = (2*c + 2) + n - f - cl - br - (2*double_bonds) \
        - (4*triple_bonds) - (2*ring_closures)
    return max(0, h)


def estimate_mw(atom_counts, h_count):
    """Estimate molecular weight from atom counts."""
    mw = sum(atom_counts.get(a, 0) * ATOMIC_WEIGHTS.get(a, 0)
             for a in atom_counts)
    mw += h_count * ATOMIC_WEIGHTS['H']
    return mw


def estimate_logp(atom_counts, smiles, h_count):
    """Estimate LogP via simplified Wildman-Crippen fragments."""
    logp = 0.0
    aromatic_c = sum(1 for ch in smiles if ch == 'c')
    aliphatic_c = atom_counts.get('C', 0) - aromatic_c
    logp += aromatic_c * CRIPPEN_FRAGMENTS['C_aromatic']
    logp += aliphatic_c * CRIPPEN_FRAGMENTS['C_aliphatic']
    logp += atom_counts.get('N', 0) * CRIPPEN_FRAGMENTS['N_amine']
    logp += atom_counts.get('O', 0) * CRIPPEN_FRAGMENTS['O_hydroxyl']
```

 doi:10.20944/preprints202603.0258.v2


```
    logp += atom_counts.get('F', 0) * CRIPPEN_FRAGMENTS['F']
    logp += atom_counts.get('Cl', 0) * CRIPPEN_FRAGMENTS['Cl']
    logp += atom_counts.get('Br', 0) * CRIPPEN_FRAGMENTS['Br']
    logp += atom_counts.get('S', 0) * CRIPPEN_FRAGMENTS['S']
    polar_h = (len(re.findall(r'O[^=]', smiles))
               + len(re.findall(r'N[^=]', smiles)))
    nonpolar_h = max(0, h_count - polar_h)
    logp += polar_h * CRIPPEN_FRAGMENTS['H_polar']
    logp += nonpolar_h * CRIPPEN_FRAGMENTS['H_nonpolar']
    return logp


def estimate_tpsa(smiles):
    """Estimate TPSA from fragment contributions."""
    tpsa = 0.0
    tpsa += len(re.findall(r'[^O]O[^C=]|\(O\)', smiles)) * TPSA_FRAGMENTS['OH']
    tpsa += len(re.findall(r'[^C]N[^(C+)]', smiles)) * TPSA_FRAGMENTS['NH2']
    tpsa += len(re.findall(r'CN[^(C+)]', smiles)) * TPSA_FRAGMENTS['NH']
    tpsa += len(re.findall(r'COC', smiles)) * TPSA_FRAGMENTS['O_ether']
    tpsa += len(re.findall(r'C\(=O\)', smiles)) * TPSA_FRAGMENTS['O_carbonyl']
    tpsa += len(re.findall(r'S[^(=)]', smiles)) * TPSA_FRAGMENTS.get('S_thiol', 0)
    return tpsa


def count_hbd(smiles):
    """Count hydrogen bond donors (OH + NH groups)."""
    oh = len(re.findall(r'[^O]O[^C=]|\(O\)', smiles))
    nh = len(re.findall(r'N[^(=)]', smiles))
    return oh + nh


def count_hba(smiles):
    """Count hydrogen bond acceptors (N + O atoms)."""
    atoms = count_atoms(smiles)
    return atoms.get('N', 0) + atoms.get('O', 0)


def count_rotatable_bonds(smiles):
    """Estimate rotatable bonds from SMILES."""
    clean = re.sub(r'\[.*?\]', 'X', smiles)
    clean = re.sub(r'[0-9]', '', clean)
    singles = clean.count('-') + len(re.findall(r'[A-Za-z][A-Za-z]', clean))
    ring_bonds = len(set(re.findall(r'(?<!%)([1-9])', smiles)))
    double_bonds = smiles.count('=')
```

 doi:10.20944/preprints202603.0258.v2


```
    triple_bonds = smiles.count('#')
    return max(0, singles - ring_bonds - double_bonds - triple_bonds)


def compute_ring_features(smiles):
    """Compute ring-related features."""
    features = []
    ring_digits = set(re.findall(r'(?<!%)([1-9])', smiles))
    total_rings = len(ring_digits)
    features.append(total_rings)
    aromatic_atoms = sum(1 for c in smiles
        if c.islower() and c.isalpha() and c not in 'lrn')
    features.append(min(aromatic_atoms // 4, total_rings))
    features.append(max(0, total_rings - features[-1]))
    hetero_in_ring = sum(1 for c in smiles if c in 'nos')
    features.append(hetero_in_ring)
    has_fused = 1 if total_rings > 1 and any(
        smiles.count(str(d)) > 2 for d in range(1, 10)) else 0
    features.append(has_fused)
    ring_sizes = []
    for d in ring_digits:
        positions = [i for i, c in enumerate(smiles) if c == d]
        if len(positions) >= 2:
            ring_sizes.append(positions[-1] - positions[0])
    features.append(min(ring_sizes) if ring_sizes else 0)
    features.append(max(ring_sizes) if ring_sizes else 0)
    features.append(np.mean(ring_sizes) if ring_sizes else 0)
    features.append(len(ring_sizes))
    features.append(1 if any(s <= 5 for s in ring_sizes) else 0)
    features.append(1 if any(s >= 8 for s in ring_sizes) else 0)
    features.append(1 if any(s == 6 for s in ring_sizes) else 0)
    return features  # 12 features


def compute_morgan_hash(smiles, nbits=128, radius=2):
    """Compute Morgan-like hash fingerprint from SMILES."""
    bits = [0] * nbits
    atoms = parse_atoms_from_smiles(smiles)
    for i, atom in enumerate(atoms):
        for r in range(radius + 1):
            start = max(0, i - r)
            end = min(len(atoms), i + r + 1)
            env = ''.join(atoms[start:end])
            hash_val = int(hashlib.md5(env.encode()).hexdigest(), 16) % nbits
```

 doi:10.20944/preprints202603.0258.v2


```
            bits[hash_val] = 1
    return bits


def compute_descriptors(smiles):
    """
    Compute 170+ molecular descriptors from a SMILES string.
    Returns a numpy array of features.
    """
    features = []
    for name, pattern in FUNCTIONAL_GROUP_PATTERNS.items():
        try:
            features.append(len(re.findall(pattern, smiles)))
        except re.error:
            features.append(0)
    ring_feats = compute_ring_features(smiles)
    features.extend(ring_feats)
    atom_counts = count_atoms(smiles)
    for elem in ['C', 'N', 'O', 'F', 'S', 'Cl', 'Br', 'P', 'I']:
        features.append(atom_counts.get(elem, 0))
    heavy = sum(atom_counts.values())
    features.append(heavy)
    h_count = estimate_hydrogens(atom_counts, smiles)
    features.append(h_count)
    features.append(heavy + h_count)
    features.append(atom_counts.get('C', 0) / max(heavy, 1))
    features.append(atom_counts.get('N', 0) / max(heavy, 1))
    features.append(atom_counts.get('O', 0) / max(heavy, 1))
    mw = estimate_mw(atom_counts, h_count)
    logp = estimate_logp(atom_counts, smiles, h_count)
    tpsa = estimate_tpsa(smiles)
    hbd = count_hbd(smiles)
    hba = count_hba(smiles)
    rotbonds = count_rotatable_bonds(smiles)
    features.extend([mw, logp, tpsa, hbd, hba, rotbonds])
    features.append(mw / max(heavy, 1))
    features.append(hbd + hba)
    aromatic_c = sum(1 for c in smiles if c == 'c')
    sp3_c = max(0, atom_counts.get('C', 0) - aromatic_c)
    fsp3 = sp3_c / max(atom_counts.get('C', 0), 1)
    features.append(fsp3)
    features.append(len(smiles))
    for name, pattern in QS_MOTIF_PATTERNS.items():
        try:
```

 doi:10.20944/preprints202603.0258.v2


```
            features.append(1 if re.search(pattern, smiles) else 0)
        except re.error:
            features.append(0)
    morgan_bits = compute_morgan_hash(smiles, nbits=128, radius=2)
    features.extend(morgan_bits)
    return np.array(features, dtype=np.float64)


# ============================================================
# SECTION A.2: TRAINING AND ENSEMBLE
# ============================================================

def build_ensemble():
    """Build four-model ensemble with balanced class weights."""
    models = {
        'RF-500': RandomForestClassifier(
            n_estimators=500, class_weight='balanced',
            random_state=42, n_jobs=-1),
        'GBM-200': GradientBoostingClassifier(
            n_estimators=200, max_depth=4,
            learning_rate=0.05, random_state=42),
        'SVM-RBF': Pipeline([
            ('scaler', StandardScaler()),
            ('svm', SVC(kernel='rbf', class_weight='balanced',
                        probability=True, random_state=42))]),
        'LogReg': Pipeline([
            ('scaler', StandardScaler()),
            ('lr', LogisticRegression(class_weight='balanced',
                                      max_iter=1000, random_state=42))]),
    }
    return models


def cross_validate_ensemble(models, X, y, n_splits=5):
    """Run stratified cross-validation for each model."""
    cv = StratifiedKFold(n_splits=n_splits, shuffle=True, random_state=42)
    results = {}
    for name, model in models.items():
        scores = cross_val_score(model, X, y, cv=cv, scoring='roc_auc')
        results[name] = {'mean': scores.mean(), 'std': scores.std(),
                         'folds': scores}
        print(f"{name}: AUC = {scores.mean():.3f} +/- {scores.std():.3f}")
    return results
```

 doi:10.20944/preprints202603.0258.v2


```
def train_ensemble(models, X, y):
    """Train all models on the full training set."""
    trained = {}
    for name, model in models.items():
        model.fit(X, y)
        trained[name] = model
    return trained


def predict_consensus(trained_models, X):
    """Consensus prediction: arithmetic mean of individual probabilities."""
    probs = []
    for name, model in trained_models.items():
        probs.append(model.predict_proba(X)[:, 1])
    return np.mean(probs, axis=0)


def predict_individual(trained_models, X):
    """Return dict of individual model predictions."""
    return {name: model.predict_proba(X)[:, 1]
            for name, model in trained_models.items()}


# ============================================================
# SECTION A.3: SCREENING AND TRIAGE
# ============================================================

def screen_compounds(smiles_list, trained_qs_models, trained_abx_models,
                     qs_threshold=0.5, abx_threshold=0.5):
    """Screen SMILES against both QS and ABX ensembles."""
    results = []
    for smi in smiles_list:
        desc = compute_descriptors(smi).reshape(1, -1)
        p_qs = predict_consensus(trained_qs_models, desc)[0]
        p_abx = predict_consensus(trained_abx_models, desc)[0]
        joint = p_qs * p_abx
        qs_indiv = predict_individual(trained_qs_models, desc)
        abx_indiv = predict_individual(trained_abx_models, desc)
        abx_over_05 = sum(1 for v in abx_indiv.values() if v[0] > 0.5)
        tier = 1 if p_qs >= qs_threshold else 2
        results.append({
            'smiles': smi, 'P_QS': round(p_qs, 4),
            'P_Abx': round(p_abx, 4), 'Joint': round(joint, 4),
```

 doi:10.20944/preprints202603.0258.v2


```
            'ABX_over_05': f"{abx_over_05}/4", 'Tier': tier,
            'QS_individual': {k: round(v[0], 4)
                              for k, v in qs_indiv.items()},
            'ABX_individual': {k: round(v[0], 4)
                               for k, v in abx_indiv.items()},
        })
    return sorted(results, key=lambda x: -x['Joint'])


# ================================================================
# SECTION A.4: EXAMPLE USAGE
# ================================================================

if __name__ == '__main__':
    # --- Screen M6-12 ---
    m6_12 = 'CNCc1c(F)cc(OC)c2C(OC)C3C(O)CNCC3C(O)c12'  # v2 corrected
    # results = screen_compounds([m6_12], trained_qs, trained_abx)
    # print(results)

    # --- Validate descriptor computation ---
    desc = compute_descriptors(m6_12)
    print(f"M6-12 descriptor vector: {len(desc)} features")
    print(f"Feature vector computed successfully.")
```

References

1. Abegg, M. A. (2026). Exploratory AI-assisted in silico evidence that bacterial signaling molecules may occupy a drug-like pharmacokinetic space. Preprints, 2026020854. https://doi.org/10.20944/preprints202602.0854.v1
2. Andersson, D. I., & Hughes, D. (2014). Microbiological effects of sublethal levels of antibiotics. Nat Rev Microbiol, 12, 465–478. https://doi.org/10.1038/nrmicro3270
3. Baell, J. B., & Holloway, G. A. (2010). New Substructure Filters for Removal of Pan Assay Interference Compounds (PAINS) from Screening Libraries and for Their Exclusion in Bioassays. J Med Chem, 53, 2719–2740. https://doi.org/10.1021/jm901137j
4. Beyersmann, P. G., Tomasch, J., Son, K., Stocker, R., Göker, M., Wagner-Döbler, I., Simon, M., & Brinkhoff, T. (2017). Dual function of tropodithietic acid as antibiotic and signaling molecule in global gene regulation of the probiotic bacterium Phaeobacter inhibens. Scientific Reports, 7(1), Article 730. https://doi.org/10.1038/s41598-017-00784-7
5. Brown, E. D., & Wright, G. D. (2016). Antibacterial drug discovery in the resistance era. Nature, 529, 336–343. https://doi.org/10.1038/nature17042
6. Cavasotto, C. N., & Scardino, V. (2022). Machine learning toxicity prediction: Latest advances by toxicity end point. ACS Omega, 7(51), 47536–47546. https://doi.org/10.1021/acsomega.2c05693
7. Cesaro, A., Bagheri, M., Torres, M., Wan, F., & de la Fuente-Nunez, C. (2023). Deep learning tools to accelerate antibiotic discovery. Expert Opin Drug Discov, 18(11), 1245–1257. https://doi.org/10.1080/17460441.2023.2250721
8. Chen, M., Tung, C.-W., Shi, Q., Guo, L., Shi, L., Fang, H., Borlak, J., & Tong, W. (2014). A testing strategy to predict risk for drug-induced liver injury in humans using high-content screen assays and the 'rule-of-two' model. Archives of Toxicology, 88(7), 1439–1449. https://doi.org/10.1007/s00204-014-1276-9

 doi:10.20944/preprints202603.0258.v2

9. Daina, A., Michielin, O., & Zoete, V. (2017). SwissADME: a free web tool to evaluate pharmacokinetics, drug-likeness and medicinal chemistry friendliness of small molecules. Sci Rep, 7, 42717. https://doi.org/10.1038/srep42717
10. Davies, J., & Davies, D. (2010). Origins and evolution of antibiotic resistance. Microbiol Mol Biol Rev, 74, 417–433. https://doi.org/10.1128/MMBR.00016-10
11. Desai, J. V., & Mitchell, A. P. (2015). Candida albicans biofilm development and its genetic control. Microbiology Spectrum, 3(3), Article MB-0005-2014. https://doi.org/10.1128/microbiolspec.MB-0005-2014
12. Fajardo, A., & Martínez, J. L. (2008). Antibiotics as signals that trigger specific bacterial responses. Curr Opin Microbiol, 11(2), 161–167. https://doi.org/10.1016/j.mib.2008.02.006
13. Fernández-Delgado, M., Cernadas, E., Barro, S., & Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems? Journal of Machine Learning Research, 15(90), 3133–3181. https://doi.org/10.5555/2627435.2697065
14. Feng, X., Marcon, V., Pisula, W., Hansen, M. R., Kirkpatrick, J., Grozema, F., Andrienko, D., Kremer, K., & Müllen, K. (2009). Towards high charge-carrier mobilities by rational design of the shape and periphery of discotics. Nat Mater, 8(5), 421–426. https://doi.org/10.1038/nmat2427
15. Gaulton, A., Hersey, A., Nowotka, M., Bento, A. P., Chambers, J., Mendez, D., Mutowo, P., Atkinson, F., Bellis, L. J., Cibrián-Uhalte, E., et al. (2017). The ChEMBL database in 2017. Nucleic Acids Res, 45(D1), D945–D954. https://doi.org/10.1093/nar/gkw1074
16. Gfeller, D., Grosdidier, A., Wirth, M., Daina, A., Michielin, O., & Zoete, V. (2014). SwissTargetPrediction: a web server for target prediction of bioactive small molecules. Nucleic Acids Res, 42(W1), W32–W38. https://doi.org/10.1093/nar/gku293
17. Goh, E.-B., Yim, G., Tsui, W., McClure, J., Surette, M. G., & Davies, J. (2002). Transcriptional modulation of bacterial gene expression by subinhibitory concentrations of antibiotics. Proc Natl Acad Sci USA, 99(26), 17025–17030. https://doi.org/10.1073/pnas.252607699
18. Guengerich, F. P. (2011). Mechanisms of drug toxicity and relevance to pharmaceutical development. Drug Metab Pharmacokinet, 26, 167–176. https://doi.org/10.2133/dmpk.DMPK-10-RV-062
19. Hagmann, W. K. (2008). The many roles for fluorine in medicinal chemistry. J Med Chem, 51, 4359–4369. https://doi.org/10.1021/jm800219f
20. Henriksen, N. N. S. E., Lindqvist, L. L., Wibowo, M., Sonnenschein, E. C., Bentzon-Tilia, M., & Gram, L. (2022). Role is in the eye of the beholder—the multiple functions of the antibacterial compound tropodithietic acid produced by marine Rhodobacteraceae. FEMS Microbiol Rev, 46, fuac007. https://doi.org/10.1093/femsre/fuac007
21. Hentzer, M., Riedel, K., Rasmussen, T. B., Heydorn, A., Andersen, J. B., Parsek, M. R., Rice, S. A., Eberl, L., Molin, S., Høiby, N., Kjelleberg, S., & Givskov, M. (2002). Inhibition of quorum sensing in Pseudomonas aeruginosa biofilm bacteria by a halogenated furanone compound. Microbiology, 148(Pt 1), 87–102. https://doi.org/10.1099/00221287-148-1-87
22. Hoffman, L. R., D'Argenio, D. A., MacCoss, M. J., Zhang, Z., Jones, R. A., & Miller, S. I. (2005). Aminoglycoside antibiotics induce bacterial biofilm formation. Nature, 436, 1171–1175. https://doi.org/10.1038/nature03912
23. Irwin, J. J., Sterling, T., Mysinger, M. M., Bolstad, E. S., & Coleman, R. G. (2012). ZINC: a free tool to discover chemistry for biology. J Chem Inf Model, 52(7), 1757–1768. https://doi.org/10.1021/ci3001277
24. Kalia, V. C. (2013). Quorum sensing inhibitors: an overview. Biotechnol Adv, 31, 224–245. https://doi.org/10.1016/j.biotechadv.2012.10.004
25. Landrum, G. (2016). RDKit: Open-source cheminformatics. https://www.rdkit.org
26. Linares, J. F., Gustafsson, I., Baquero, F., & Martinez, J. L. (2006). Antibiotics as intermicrobial signaling agents instead of weapons. Proc Natl Acad Sci USA, 103, 19484–19489. https://doi.org/10.1073/pnas.0608949103
27. Liu, G.-Y., Yu, D., Fan, M.-M., Zhang, X., Jin, Z.-Y., Tang, C., & Liu, X.-F. (2024). Antimicrobial resistance crisis: Could artificial intelligence be the solution? Military Medical Research, 11(1), Article 7. https://doi.org/10.1186/s40779-024-00510-1

 doi:10.20944/preprints202603.0258.v2

28. Lovering, F., Bikker, J., & Humblet, C. (2009). Escape from Flatland: increasing saturation as an approach to improving clinical success. J Med Chem, 52(21), 6752–6756. https://doi.org/10.1021/jm901241e
29. McGovern, S. L., Helfand, B. T., Feng, B., & Shoichet, B. K. (2003). A specific mechanism of nonspecific inhibition. J Med Chem, 46(20), 4265–4272. https://doi.org/10.1021/jm030266r
30. Meanwell, N. A. (2011). Synopsis of bioisosteres in drug design. J Med Chem, 54, 2529–2591. https://doi.org/10.1021/jm1013693
31. Mugumbate, G., & Overington, J. P. (2015). The relationship between target-class and the physicochemical properties of antibacterial drugs. Bioorganic & Medicinal Chemistry, 23(16), 5218–5224. https://doi.org/10.1016/j.bmc.2015.04.063
32. Mulliner, D., Schmidt, F., Stolte, M., Spirkl, H.-P., Czich, A., & Amberg, A. (2016). Computational models for human and animal hepatotoxicity with a global application scope. Chemical Research in Toxicology, 29(5), 757–767. https://doi.org/10.1021/acs.chemrestox.5b00465
33. Murray, C. J. L., Ikuta, K. S., Sharara, F., Swetschinski, L., Robles Aguilar, G., Gray, A., et al. (2022). Global burden of bacterial antimicrobial resistance in 2019: a systematic analysis. Lancet, 399(10325), 629–655. https://doi.org/10.1016/S0140-6736(21)02724-0
34. O'Shea, R., & Moser, H. E. (2008). Physicochemical properties of antibacterial compounds. J Med Chem, 51, 2871–2878. https://doi.org/10.1021/jm700967e
35. Patani, G. A., & LaVoie, E. J. (1996). Bioisosterism: a rational approach in drug design. Chem Rev, 96, 3147–3176. https://doi.org/10.1021/cr950066q
36. Payne, D. J., Gwynn, M. N., Holmes, D. J., & Pompliano, D. L. (2007). Drugs for bad bugs: confronting the challenges of antibacterial discovery. Nat Rev Drug Discov, 6(1), 29–40. https://doi.org/10.1038/nrd2201
37. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825–2830.
38. Pires, D. E. V., Blundell, T. L., & Ascher, D. B. (2015). pkCSM: predicting small-molecule pharmacokinetic and toxicity properties using graph-based signatures. J Med Chem, 58(9), 4066–4072. https://doi.org/10.1021/acs.jmedchem.5b00104
39. Purser, S., Moore, P. R., Swallow, S., & Gouverneur, V. (2008). Fluorine in medicinal chemistry. Chem Soc Rev, 37(2), 320–330. https://doi.org/10.1039/B610213C
40. Reisenbauer, J. C., Green, O., Franchino, A., Finkelstein, P., & Morandi, B. (2022). Late-stage diversification of indole skeletons through nitrogen atom insertion. Science, 377(6610), 1104–1109. https://doi.org/10.1126/science.add1383
41. Richter, M. F., Drown, B. S., Riley, A. P., Garcia, A., Shirai, T., Svec, R. L., & Hergenrother, P. J. (2017). Predictive compound accumulation rules yield a broad-spectrum antibiotic. Nature, 545(7654), 299–304. https://doi.org/10.1038/nature22308
42. Ritchie, T. J., & Macdonald, S. J. F. (2009). The impact of aromatic ring count on compound developability – are too many aromatic rings a liability in drug design? Drug Discovery Today, 14(21–22), 1011–1020. https://doi.org/10.1016/j.drudis.2009.07.014
43. Sengupta, S., Chattopadhyay, M. K., & Grossart, H. P. (2013). Multifaceted roles of antibiotics and antibiotic resistance in nature. Front Microbiol, 4, 47. https://doi.org/10.3389/fmicb.2013.00047
44. Skindersoe, M. E., Alhede, M., Phipps, R., Yang, L., Jensen, P. O., Rasmussen, T. B., Bjarnsholt, T., Tolker-Nielsen, T., Høiby, N., & Givskov, M. (2008). Effects of antibiotics on quorum sensing in Pseudomonas aeruginosa. Antimicrob Agents Chemother, 52(10), 3648–3663. https://doi.org/10.1128/AAC.01230-07
45. Sterling, T., & Irwin, J. J. (2015). ZINC 15 – Ligand discovery for everyone. Journal of Chemical Information and Modeling, 55(11), 2324–2337. https://doi.org/10.1021/acs.jcim.5b00559
46. Stokes, J. M., Yang, K., Swanson, K., Jin, W., Cubillos-Ruiz, A., Donghia, N. M., MacNair, C. R., French, S., Carfrae, L. A., Bloom-Ackermann, Z., et al. (2020). A deep learning approach to antibiotic discovery. Cell, 180(4), 688–702. https://doi.org/10.1016/j.cell.2020.01.021
47. Suk, J. S., Xu, Q., Kim, N., Hanes, J., & Ensign, L. M. (2016). PEGylation as a strategy for improving nanoparticle-based drug and gene delivery. Adv Drug Deliv Rev, 99(Pt A), 28–51. https://doi.org/10.1016/j.addr.2015.09.012

 doi:10.20944/preprints202603.0258.v2

48. van Tilborg, D., Brinkmann, H., Criscuolo, E., Rossen, L., Özçelik, R., & Grisoni, F. (2024). Deep learning for low-data drug discovery: hurdles and opportunities. Curr Opin Struct Biol, 86, 102818. https://doi.org/10.1016/j.sbi.2024.102818
49. Vandeputte, O. M., Kiendrebeogo, M., Rajaonson, S., Diallo, B., Mol, A., El Jaziri, M., & Baucher, M. (2010). Identification of catechin as one of the flavonoids from Combretum albiflorum bark extract that reduces the production of quorum-sensing-controlled virulence factors in Pseudomonas aeruginosa PAO1. Appl Environ Microbiol, 76(1), 243–253. https://doi.org/10.1128/AEM.01059-09
50. Vitaku, E., Smith, D. T., & Njardarson, J. T. (2014). Analysis of the structural diversity, substitution patterns, and frequency of nitrogen heterocycles among U.S. FDA approved pharmaceuticals. J Med Chem, 57(24), 10257–10274. https://doi.org/10.1021/jm501100b
51. Wöhrle, T., Wurzbach, I., Kirres, J., Kostidou, A., Kapernaum, N., Litterscheidt, J., Haenle, J. C., Staffeld, P., Baro, A., Giesselmann, F., & Laschat, S. (2016). Discotic liquid crystals. Chemical Reviews, 116(3), 1139–1241. https://doi.org/10.1021/acs.chemrev.5b00190
52. Wong, F., Zheng, E. J., Valeri, J. A., Donghia, N. M., Anahtar, M. N., Omori, S., Li, A., Cubillos-Ruiz, A., Krishnan, A., Jin, W., Manson, A. L., Friedrichs, J., Helbig, R., Hajian, B., Fiejtek, D. K., Wagner, F. F., Soutter, H. H., Earl, A. M., Stokes, J. M., Renner, L. D., & Collins, J. J. (2024). Discovery of a structural class of antibiotics with explainable deep learning. Nature, 626, 177–185. https://doi.org/10.1038/s41586-023-06887-8