



Generation of novel molecules and reactions by chemography-guided artificial intelligence

William Bort

► To cite this version:

William Bort. Generation of novel molecules and reactions by chemography-guided artificial intelligence. Other. Université de Strasbourg, 2023. English. NNT : 2023STRAF047 . tel-04405740

HAL Id: tel-04405740

<https://theses.hal.science/tel-04405740>

Submitted on 19 Jan 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ÉCOLE DOCTORALE DES SCIENCES CHIMIQUES Chimie
de la matière complexe – UMR7140

THÈSE présentée par :

William BORT

soutenue le : **22 Novembre 2023**

pour obtenir le grade de : **Docteur de l'Université de Strasbourg**

Discipline/ Spécialité : **Chimie**

**Génération de nouvelles molécules et réactions
par intelligence artificielle guidée par la
chémographie**

THÈSE dirigée par :

M. VARNEK Alexandre

Professeur, Université de Strasbourg

M. MARCOU Gilles

Maître de Conférences (HDR), Université de Strasbourg

RAPPORTEURS :

M. MONTES Matthieu

Professeur, Conservatoire National des Arts et Métiers

M. LANGER Thierry

Professeur, Université de Vienne

AUTRES MEMBRES DU JURY :

Mme. DOUGUET Dominique

Chargée de Recherche (HDR), Université Côte d'Azur

Pour Hélène.

Acknowledgements

La rédaction de cette thèse a été pour moi l'une des épreuves les plus difficiles à laquelle j'ai pu me confronter. Ces pages, lourdement chargées de sens, représentent à la fois une fin et un début. Il est difficile d'exprimer à quel point je suis reconnaissant d'avoir côtoyé durant ces dernières années des personnes toutes plus exceptionnelles les unes que les autres qui m'ont tant appris et m'ont aidé à grandir, scientifiquement, humainement et émotionnellement.

Je voudrais en premier lieu exprimer ma gratitude à tous mes collègues du laboratoire de Chémoinformatique, présents et passés. En particulier, je voudrais remercier mes superviseurs : Pr. Alexandre Varnek et Dr. Gilles Marcou pour leur patience inébranlable, leurs encouragements et leur compréhension dans les moments les plus difficiles. Je suis infiniment reconnaissant d'avoir pu travailler dans un environnement aussi expérimenté, rigoureux, et motivant aux côtés de grands scientifiques. Je voudrais également remercier Dr. Arkadii Lin pour son aide, sa créativité, son expertise et sa rigueur sans relâche, avec qui j'ai tant appris. Je n'oublie pas Dr. Fanny Bonachera, dont la porte est toujours restée ouverte et qui a su me prêter une oreille bienveillante et de précieux conseils, ainsi qu'un support technique inestimable. J'aimerais également remercier Dr. Dragos Horvath, Dr. Olga Klimchuk, Dr. Igor Baskin, Dr. Iuri Casciuc, Dr. Yuliana Zabolotna, Dr. Timur Madzhidov, Dr. Tagir Akhmetshin, Dr. Helena Perez Pena, Sai Prashanth Santhapuri, Louis Plyer, Regina Pikalyova, Karina Pikalyova, Shamkhal Baybekov, Maxim Shevelev et toutes les belles personnes que j'ai pu rencontrer dans ce laboratoire.

J'ai bien évidemment une pensée pour mes Pokémons Rares : Mélissa (Psy), Guillaume (Roche), Franck (Ténèbres), Sacha (Feu), Baptiste (Eau), Jordan (Sol), Fred (Poison), Jonathan (Spectre), Ειρήνη (Dragon) et tous les allègres complices qui m'ont accompagné durant mes années d'étude, dans les plus belles joies, les plus grandes tristesses, dans tous ces moments qui font la beauté d'une vie. Une pensée également à tous mes compagnons de basket : Marie, Brice, Clément, et tous les autres avec qui je partage ma passion et bien plus encore.

Je me dois aussi de mentionner Tali, espiègle compagne de mon quotidien dont les péripéties et tribulations ne manquent jamais de vivifier mes journées.

Pour conclure, j'aimerais remercier ma famille, et en particulier mes parents, pour m'avoir transmis leurs valeurs et leurs forces qui me permettent de persévérer jour après jour.

Contents

1	Résumé en français.....	9
1.1	Introduction	9
1.2	Résultats et Discussions.....	12
2	General Introduction	33
2.1	Sequence-to-Sequence Neural Networks	37
2.2	Generative Topographic Mapping	57
3	Exploring the latent space of an Autoencoder.....	69
3.1	Introduction	69
3.2	Methods	71
3.3	Results	73
3.4	Conclusion.....	87
4	An Autoencoder coupled with Generative Topographic Mapping for the discovery of novel reactions	89
4.1	Summary.....	106
5	Linking the latent space of an Autoencoder with another descriptor space	109
5.1	ISIDA2SMI	111
5.2	Multimodal Deep Boltzmann Machines.....	131
5.3	Stargate-GTM.....	158
5.4	Combination of ISIDA landscapes	170
5.5	Conditional Variational Autoencoder (ACoVAE)	177
6	General Conclusion & Perspectives	195
7	List of Abbreviations	199
8	References	203

1 Résumé en français

1.1 Introduction

La recherche de nouveaux composés ayant un potentiel médicamenteux est à la base de la recherche dans le domaine médicinal. Il est nécessaire d'explorer l'espace chimique des molécules afin de pouvoir isoler les médicaments de demain. De ce fait, chaque année, les bases de données chimiques commerciales et publiques voient leur nombre de molécules augmenter significativement grâce, par exemple, à de nouvelles voies de synthèses, à la chimie combinatoire ou aux outils informatiques appliqués au recensement de nouveaux composés. L'augmentation de la taille de ces bases de données entraîne également une augmentation des coûts computationnel et énergétiques pour leur stockage et lors de leur criblage. Et pourtant, malgré cette croissance exponentielle, l'espace chimique « découvert » qu'elles occupent reste encore minuscule par rapport à la taille de l'espace chimique des molécules « drug-like » potentiellement synthétisables (estimée aux alentours de 10^{33} composés^[1]). Il est donc important de développer de nouveaux outils permettant l'exploration efficace de l'espace chimique dont le potentiel pour la chimie médicinale est incontestable.

L'arrivée de nouveaux outils d'intelligence artificielle en chimie a ouvert la voie à de nouvelles méthodes très performantes dans les domaines du design et de la découverte de nouveaux composés d'intérêt pour la chimie médicinale^[2]. Un type d'architecture en particulier a été plébiscité pour sa simplicité et son efficacité : L'Autoencodeur (AE)^[3]. Le principe de ce dernier est d'ajuster simultanément les paramètres de deux processus : l'un codant et l'autre décodant. Le premier est utilisé pour coder des structures de molécules en vecteurs numériques appelés vecteurs latents. Le second doit convertir ces vecteurs latents en structures de molécules. L'espace latent peut ensuite être exploré et utilisé pour générer de nouveaux composés^[4-6]. Durant ce processus, le choix d'un vecteur latent est critique pour générer une structure chimique pertinente, par exemple un composé actif pour un projet de conception de médicament. Plusieurs architectures dérivées des Autoencodeurs initiaux sont aujourd'hui très populaires dans le domaine de la génération de composés, et peuvent être catégorisés en deux larges familles : D'un côté, les modèles qui sont entraînés avec des bibliothèques de molécules hyper-spécifiques à une application et les modèles plus généraux entraînés sur de larges bases

de données diverses. Les modèles très spécifiques ont l'avantage de générer des composés à haut potentiel (activité par rapport une protéine par exemple) mais doivent être réentraînés à chaque changement d'objectif. Les modèles plus généraux, eux, permettent de générer des molécules plus diverses en explorant de plus larges zones de l'espace chimique. Néanmoins, les espaces chimiques latents étant hautement multidimensionnels, ils sont difficiles à visualiser, explorer et échantillonner.

La Cartographie Topographique Générative (GTM)^[7] est une méthode de réduction de la dimensionalité qui permet de visualiser sur des cartes 2D des espaces multi-dimensionnels. La GTM assigne en tout point de la carte une probabilité de présence à une molécule au lieu de fixer sa position à un seul point. Ces probabilités peuvent être utilisées pour définir des cartes de densité de l'espace chimique. En prenant en compte les étiquettes (classes « actif » / « inactif », propriétés physico-chimiques, etc.) associées aux données, la GTM produit des paysages, i.e. des cartes représentant les valeurs de ces étiquettes par des codes couleurs, analogues à des cartes de géographie. Il devient trivial de cibler des zones pertinentes de l'espace chimique dans lesquelles la génération de nouveaux composés aura de grandes chances de proposer de nouvelles molécules d'intérêt (Figure 1).

Les vecteurs latents basés sur l'interprétation d'un AE ont montré qu'ils ont la capacité de correctement séparer actifs et inactifs. Néanmoins, étant basée sur une interprétation séquentielle de chaînes de caractères SMILES, l'organisation de l'espace chimique en résultant est segmenté par les règles sémantiques des codes SMILES. Ce n'est pas le cas avec des descripteurs structuraux calculés sur des graphes moléculaires : ces derniers sont donc plus efficaces. De plus, ils sont modulables, ce qui permet de les adapter plus finement à des tâches de modélisation QSAR et d'intégrer des connaissances antérieures. Si des vecteurs de descripteurs moléculaires sont plus efficaces pour prédire des propriétés à partir de structures chimiques, en revanche, il n'existe pas jusqu'à présent de procédure pour générer des structures chimiques correspondant à des vecteurs descripteurs moléculaires. Dans cette optique, il est intéressant de combiner la versatilité et les performances des espaces chimiques construits sur des descripteurs structuraux aux capacités d'un autoencodeur pour générer des structures chimiques.

Cette thèse a donc deux objectifs principaux. Dans un premier temps, des méthodes de cartographie aux espaces latents des AutoEncodeurs ont été combinés pour mieux rationaliser

l'organisation de ces espaces latents et en permettre l'exploration. En particulier, pour la première fois, un autoencodeur en combinaison avec la cartographie a été utilisé pour générer de nouvelles transformations chimiques. Dans un second temps, cette thèse présente les résultats des recherches visant à convertir des descripteurs moléculaires structuraux en structures chimiques par l'intermédiaire d'un autoencodeur. Ceci est indispensable quand les vecteurs latents d'un autoencodeur sont moins performants par rapport aux descripteurs moléculaires sélectionnés pour une modélisation QSAR. Une méthode est donc proposée pour permettre la génération de composés avec des propriétés chimiques et des descripteurs structuraux précis.

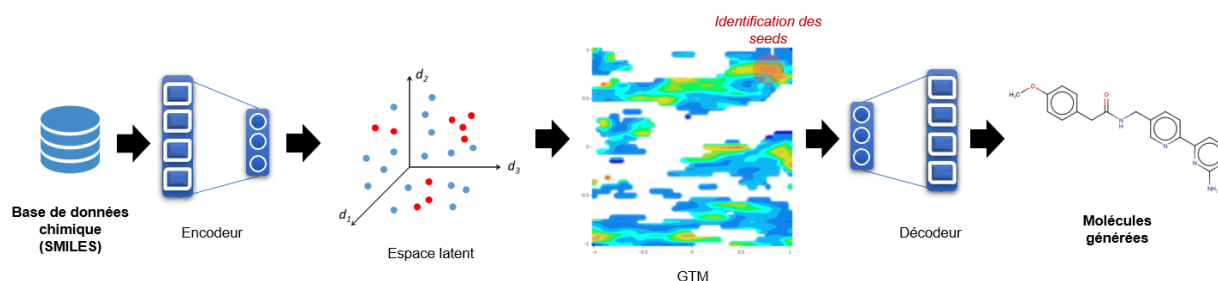


Figure 1. Processus de création d'un espace latent basé sur une base de données chimique encodée en SMILES via un Autoencodeur. L'espace latent est ensuite visualisé à travers la GTM pour permettre l'échantillonnage de l'espace chimique dans les régions plus intéressantes.

1.2 Résultats et Discussions

1.2.1 Etude de l'espace latent d'un autoencodeur LSTM

Les architectures de type AutoEncodeur (AE) restent des systèmes « boîtes noires » et la compréhension de leur fonctionnement interne est encore incomplète, en particulier pour les applications en chimie. Le but de cette étude était d'approfondir la compréhension de l'entraînement et de l'organisation de l'espace latent d'un AE. Un AE muni de couches Long Short-Term Memory (LSTM) a été entraîné sur la base de données ChEMBL23 - environ 1.5 millions de composés. Les molécules de la base de données ont été utilisées sous forme de SMILES canoniques.

Il a d'abord été montré que l'entraînement des modèles n'était pas entièrement reproductible avec les équipements classiquement utilisés (cartes graphiques) dans ce type de recherches. Malgré ces différences dans la création des espaces latents, l'organisation des molécules dans l'espace chimique reste comparable d'un modèle à l'autre si les paramètres sont les mêmes. De plus, des projections de structures chimiques représentées par des SMILES différents ont été effectuées pour vérifier l'existence d'une dépendance de l'ordre des caractères composant le SMILES dans l'interprétation du réseau de neurones.

De nombreux paysages GTM ont ensuite été construits pour visualiser la répartition de certaines propriétés comme la densité de présence de molécules, la distance au feuillet de la GTM (son centre) et des propriétés physico-chimiques. Il a été possible grâce à ces cartes de prouver que le modèle est capable de regrouper des composés en familles chimiques. Ces paysages ont mis en évidence des différences d'organisation dans l'espace latent de l'AE par rapport à des descripteurs structuraux tels que ISIDA^[8].

Afin d'analyser les capacités génératives de l'AE, 1000 chaînes SMILES ont été systématiquement générées sur chaque nœud de la GTM et comparées aux densités observées dans la base de données ChEMBL23. Des paysages ont ensuite été construits pour visualiser le pourcentage de molécules valides générées ainsi que leur profil de propriété, tels qu'ils résultent de ChEMBL23. Un comparatif entre paysages « réels » et paysages « générés » (Figure 2) a

permis de vérifier la capacité du modèle à créer des molécules réalistes, et de comprendre les facteurs pouvant influencer l'efficacité du processus génératif.

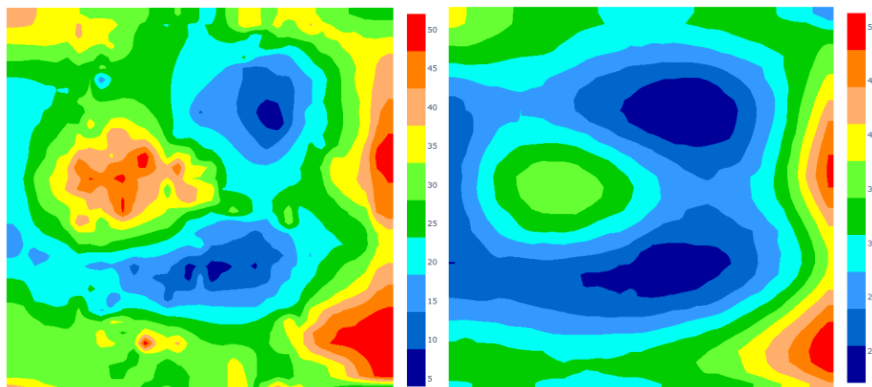


Figure 2. Nombre d'atomes lourds pour les composés de ChEMBL (à gauche) et pour les composés générés (à droite)

Ces études de densité et de profils de propriétés a permis la mise en évidence d'un « lissage des propriétés ». Les profils de propriétés physico-chimiques des molécules ChEMBL et des molécules générées ont une correspondance claire d'une carte à l'autre, néanmoins les profils de propriétés des molécules générées ont tendances à être plus lisses et indiquent que la génération de composés se fait par moyennage des zones peuplées aux alentours.

1.2.2 Utilisation d'un autoencodeur couplé aux cartes topographiques génératives pour la découverte de nouvelles réactions

La recherche de nouvelles réactions est intimement liée au processus de design de médicaments. L'augmentation des possibilités de transformations chimiques facilite la synthèse des nouveaux composés, pour des applications industrielles par exemple. Les réactions chimiques étant des systèmes impliquant plusieurs molécules et des conditions, elles sont donc plus difficiles à modéliser.

Grâce à la technologie des Condensed Graph of Reaction (CGR)^[9], il a été possible d'étudier la base de données USPTO^[10] qui référence presque 2.5 millions de réactions issues d'une base de données de brevets. Un CGR représente une réaction sous forme de pseudo-molécule où les réactifs et les produits sont combinés en un seul ensemble. Ces pseudo-molécules sont ensuite exprimées sous forme de CGR où les changements dans les liaisons sont inclus à l'aide de caractères spéciaux, reprenant les bases de la grammaire SMILES en incluant des modifications pour tenir compte des spécifications des réactions (Figure 3). Les CGR, couplés à une modification de l'architecture des AutoEncodeurs classiques a permis l'entraînement d'un modèle génératif pour des réactions chimiques.

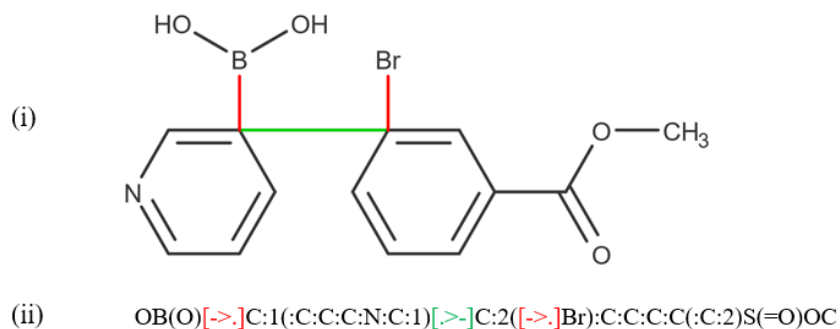


Figure 3. (i) Représentation schématique d'une réaction de Suzuki sous forme de CGR. La liaison verte indique une liaison créée lors de la réaction, les liaisons rouges indiquent des liaisons brisées pendant la réaction. (ii) SMILES-CGR correspondant à la réaction (i), [$>.$] indique une transformation d'une liaison simple vers une absence de liaison. [$>.$] indique le passage d'une absence de liaison vers une liaison simple.

La combinaison de ce modèle avec l'outil GTM a permis de cartographier l'espace chimique des réactions et de générer des types de réactions spécifiques à partir de positions sur une carte. Une grande quantité de réactions de type Suzuki a été générée, puis filtrée à l'aide

d'une nouvelle méthode de détection de nouveautés. Cette méthode exploite les « centres de réactions », autrement dit l'ensemble des atomes et liaisons directement impliqués par une transformation chimique. Cette définition a été étendue pour créer les « environnements de réactions » qui eux correspondent non seulement au centre de réaction mais qui incluent également tous les atomes directement liés au centre de réaction.

Parmi les réactions valides restantes après filtrage, 13 réactions ont été identifiées comme potentiellement nouvelles et ne figurant pas dans la base de données USPTO. Parmi celles-ci, 5 ont ensuite été identifiées dans d'autres bases de données, vérifiant la capacité du modèle à générer des réactions cohérentes chimiquement. La faisabilité de ces réactions a été confirmée par des calculs de DFT de l'enthalpie de réaction en phase gazeuse.

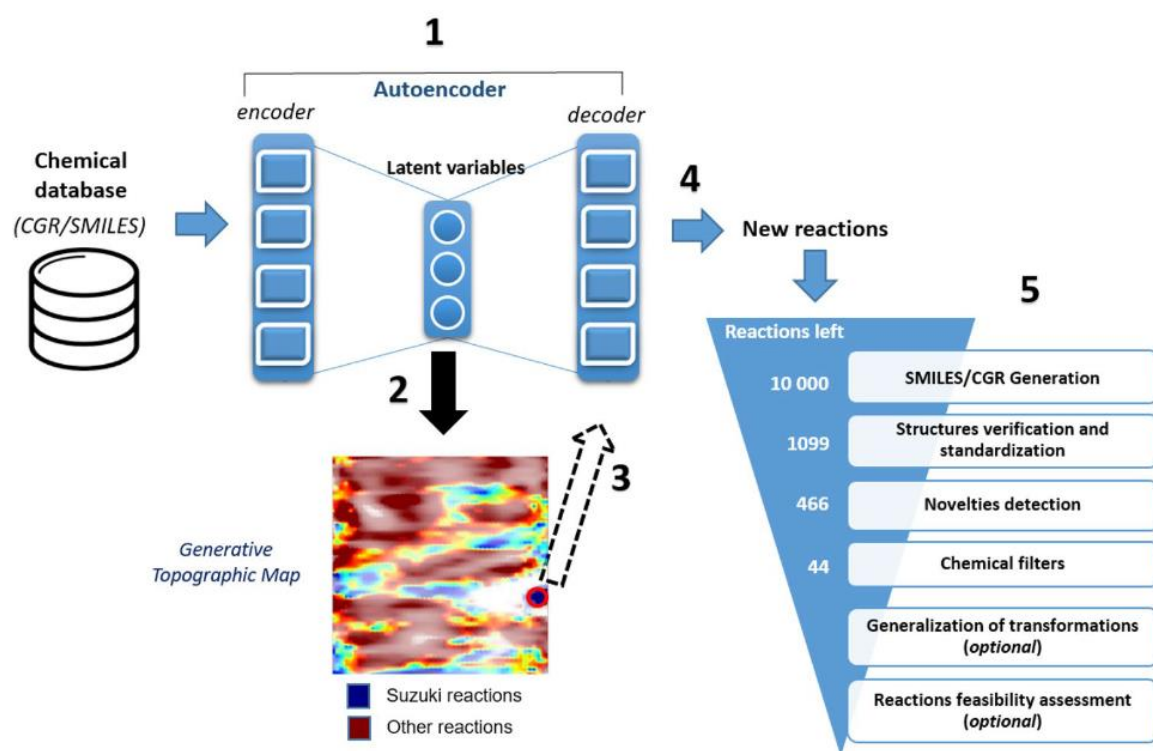


Figure 4. Processus résumant l'utilisation d'un modèle d'AutoEncodeur couplé au cartes topographiques génératrices pour la génération de nouvelles réactions chimiques. Le modèle est entraîné sur la base de données USPTO sous forme de graphes condensés de réaction (1) puis les vecteurs latents sont utilisés pour la construction d'une carte topographique générative (2). Des zones d'intérêt sont ensuite sélectionnées sur cette carte (3) et sont utilisées pour générer des réactions (4). Après plusieurs filtres (5), on obtient des réactions potentiellement nouvelles et chimiquement vraisemblables, confirmées dans la bibliographie et des calculs DFT.

1.2.3 Liaison entre l'espace latent d'un AutoEncodeur et un autre espace de descripteurs

La génération de jeux de données possédant certaines propriétés ciblées est un problème central en Chémoinformatique. Plutôt que de générer au hasard des structures moléculaires ou de chercher des composés intéressants dans des bases de données toujours plus grandes, « à la recherche d'une aiguille dans une botte de foin », il est préférable de pouvoir choisir les régions de l'espace chimique susceptibles d'abriter les structures chimiques qui satisfont les critères désirés par un utilisateur et d'échantillonner ces régions pour générer des structures pertinentes. Les méthodes à base d'AE et de paysages GTM présentées précédemment permettent de biaiser la génération de structures sur des zones riches en composés biologiquement actifs. Mais la sémantique propre au code SMILES des structures chimiques fragmente l'espace chimique de l'AE de façon arbitraire et peu contrôlable ce qui rend plus complexe l'exploration de l'espace latent en résultant. Cela résulte en un contrôle plus difficile des structures générées et de leurs propriétés.

Les descripteurs moléculaires structuraux ne présentent pas ce même défaut. Ils peuvent être adaptés selon le type de structure, le type de cibles et/ou le type d'application. Leur versatilité les rends donc beaucoup plus robustes et applicables efficacement à une plus grande diversité de problèmes. Du fait de leur surjectivité, il est néanmoins impossible d'associer un vecteur de descripteurs à une seule structure. En pratique, plusieurs molécules peuvent avoir le même vecteur de descripteur, ce qui complexifie énormément la tâche d'entraîner un modèle d'AE.

Cependant, les bénéfices potentiels à la réalisation d'un modèle génératif où les propriétés structurelles et physico-chimiques sont solidement contrôlées est très intéressant. Il serait donc utile de combiner la versatilité et la robustesse des descripteurs structuraux classiques avec le pouvoir génératif des réseaux de neurones.

SMI2ISIDA

Une méthode d'inversion directe de modèle de QSAR a été proposée, basé sur une architecture simple. Un réseau de neurones artificiels basé sur une succession de couches LSTM (les mêmes qui sont utilisées dans les AutoEncodeurs) a pour objectif de traduire un vecteur de descripteurs moléculaires ISIDA en SMILES (Figure 5). La difficulté du projet réside dans la problématique de la multimodalité des vecteurs ISIDA par rapport aux chaînes SMILES. Un vecteur de descripteurs ISIDA peut correspondre à plusieurs chaînes SMILES, ce qui rend l'entraînement d'un modèle délicat.

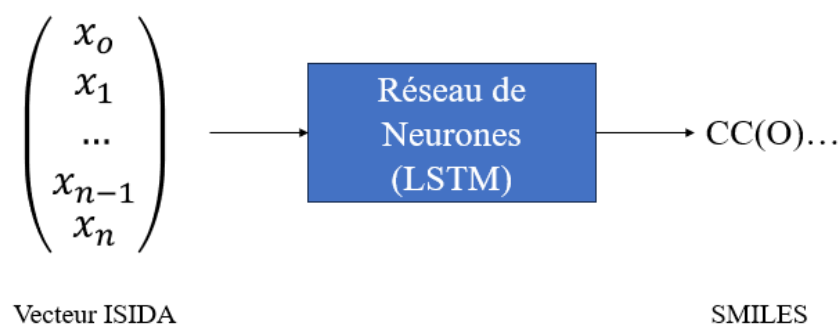


Figure 5. Représentation schématique du processus désiré de passage de vecteur ISIDA à SMILES.

Afin de pouvoir vérifier le bon fonctionnement du modèle, le vecteur ISIDA correspondant au SMILES reconstruit devait être comparé au vecteur ISIDA initial. Cela impliquait, durant l'entraînement du modèle, le calcul constant de vecteurs ISIDA par un script. Ce script étant très chronophage et demandeur en puissance de calcul, il n'était pas envisageable de le lancer des millions de fois pendant la phase d'entraînement. Une idée a donc été introduite de créer le modèle inverse (SMILES vers ISIDA) qui, une fois entraîné, aurait la tâche de remplacer ledit script.

Un modèle capable de prédire un vecteur ISIDA à partir d'une chaîne SMILES était donc une étape nécessaire au projet global.

Une analyse en profondeur a été réalisée sur l'architecture choisie, et de nombreuses variations de la même idée ont été testées, ainsi qu'un deuxième type d'architecture basé sur l'augmentation de données SMILES. Ces différents tests ont montré que les types d'architectures utilisés étaient dans l'incapacité de faire un lien entre séquences de caractères

et compte de descripteurs, les fragments avec le plus de variabilité étant constamment mal prédits.

Machines de Boltzmann

Le projet consiste à créer un modèle capable d'associer aux vecteurs de descripteurs moléculaires structuraux, des vecteurs de l'espace latent d'un AE. Pour cela, une Machine de Boltzmann dite multimodale a été développée.

Une machine de Boltzmann^[11] est un réseau de neurones basé sur un concept d'énergie, et composé d'unités binaires constituant un réseau complètement connexe. Une machine de Boltzmann est un modèle non-supervisé qui optimise la vraisemblance des données d'entraînement. Une machine de Boltzmann peut être inversée : à partir d'un état de la couche cachée (vecteur latent), le vecteur d'entrée correspondant est reconstruit. Par exemple, si le vecteur latent correspond à un benzaldéhyde, une machine pourrait potentiellement le traduire en SMILES, une autre en graphe, et une dernière donner son nom IUPAC. Les différentes machines sont également capables de transformer une modalité en « idée conceptuelle », permettant ainsi la traduction (Figure 6).

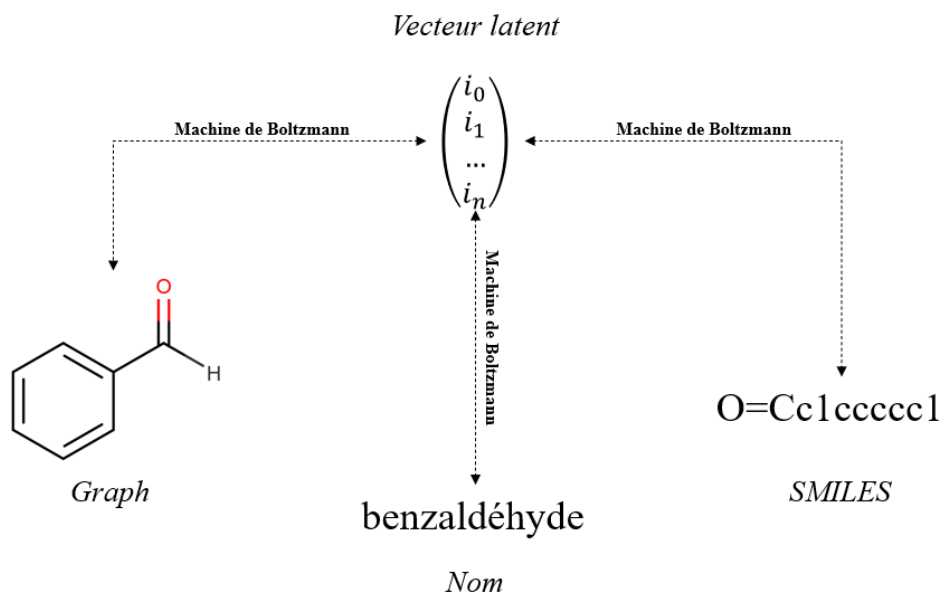


Figure 6. Représentation schématique du fonctionnement d'une Machine de Boltzmann Multimodale. Le modèle est constitué de différentes Machines de Boltzmann représentant différentes modalités connectées au même vecteur latent. Les liaisons étant dans les deux sens, la traduction d'une modalité à une autre est possible.

Dans le cas présent, deux machines de Boltzmann sont entraînées séparément sur les mêmes structures chimiques : l'une utilisant des vecteurs de descripteurs moléculaires ISIDA de ces structures, l'autre utilisant les vecteurs latents de l'AE précédemment entraîné. Une couche intermédiaire, permet de prédire un vecteur latent étant donné un vecteur de descripteurs moléculaire, et inversement. Cette architecture est dite « multimodale ». Le vecteur latent fonctionne comme une « idée conceptuelle » de l'objet en question, et les différentes machines connectées à ce vecteur latent servent à traduire cette idée dans différentes modalités.

La construction et l'entraînement de la Machine de Boltzmann Multimodale implique la construction et l'entraînement de Machines de Boltzmann individuelles : une pour les vecteurs latents de l'AE et l'autre pour les vecteurs ISIDA. Ce processus s'est fait graduellement, en augmentant au fur et à mesure la taille des modèles individuels (passage de Machines de Boltzmann restreintes à Machines de Boltzmann profondes) tout en optimisant les paramètres au fur et à mesure. Le but final étant de connecter les deux machines et de les entraîner plus finement en commun en les reliant par la couche latente.

Les faibles performances des modèles séparés pour la simple tâche de reconstruction des vecteurs et le coût en ressources et en temps nécessaires pour entraîner ce type de modèle étant trop hauts, le projet n'a pas pu aboutir. Encore une fois, le modèle était dans l'incapacité de prédire avec précision les comptes de descripteurs ISIDA à haute variation lors de la reconstruction.

Stargate GTM

Stargate-GTM^[12] est une méthode basée sur la GTM qui permet à deux espaces de descripteurs d'être coentraînés. Deux jeux de données sont présentés au modèle dont les individus se correspondent l'un à l'autre. Une carte GTM est construite sur chaque jeu de données, mais au cours de l'entraînement, chaque carte doit satisfaire les contraintes issues de la topologie de chacun des deux jeux de données, avec une pondération (Figure 7).

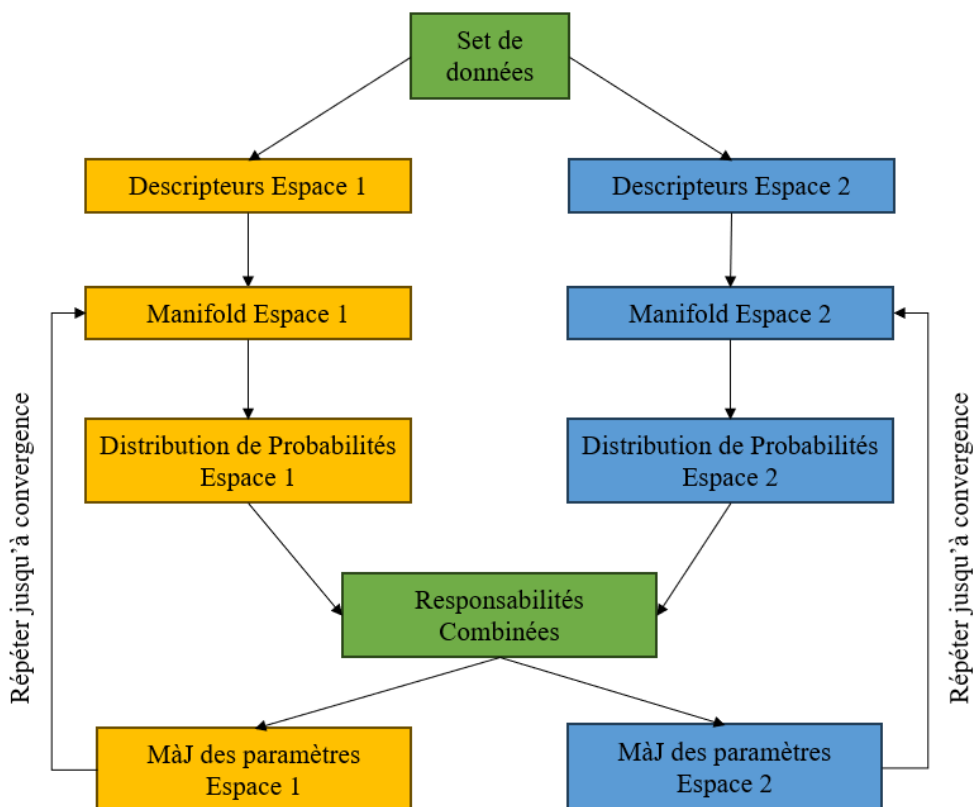


Figure 7. Processus d'entraînement de Stargate-GTM.

Finalement, les cartes se correspondent : une localisation sur une carte se traduit par un ensemble de responsabilités (densité de probabilité de présence d'une donnée) sur l'autre. Comme précédemment, les vecteurs de descripteurs et les vecteurs latents correspondants du AE sont utilisés. La projection d'une molécule sur la carte construite sur un espace de descripteur permet d'estimer les responsabilités correspondantes dans l'espace latent de l'AE qui peuvent ensuite être décodées en structures chimiques.

La comparaison des cartes basées sur les descripteurs ISIDA et les vecteurs latents d'un AE ont permis d'observer les différences de distribution de probabilités des deux espaces. Les deux types de carte présentent des similarités au niveau de la distribution de la densité de population dans certains cas, mais les zones de haute densité sur les cartes ISIDA sont systématiquement beaucoup plus concentrées que les zones de haute densité sur les cartes de vecteurs latents qui sont souvent bien plus étalées. Une étude approfondie des positions des composés sur les deux types de carte a montré que la correspondance des distributions de probabilités n'est pas respectée dans le cas d'une Stargate-GTM ISIDA-Latent, suggérant l'incompatibilité des deux espaces de descripteurs dans le cadre d'un lien direct.

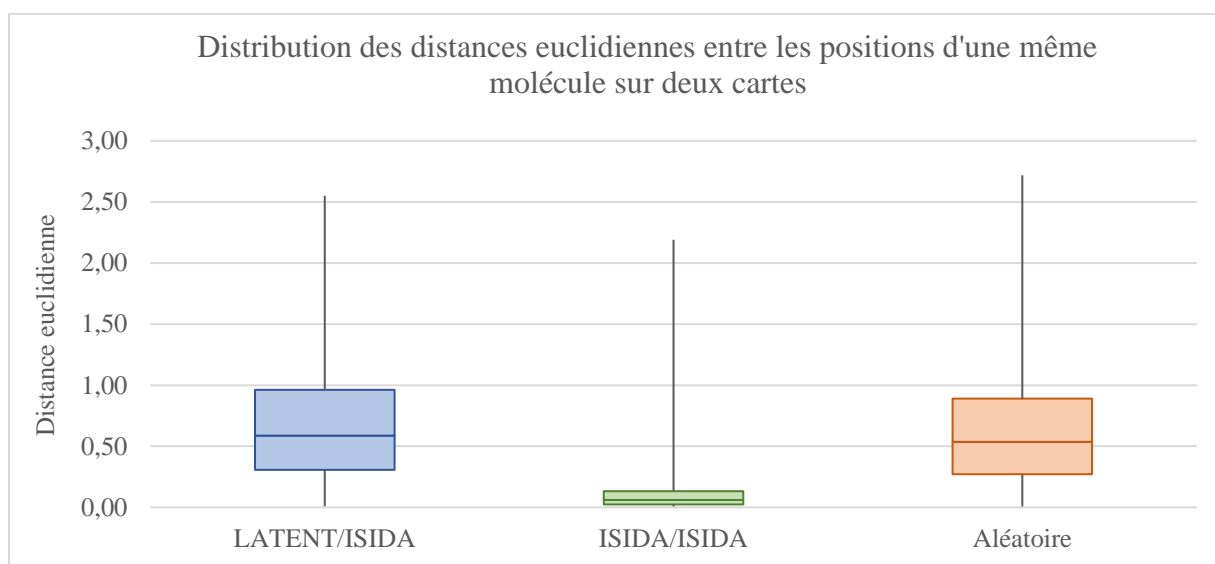


Figure 8. Distribution des distances euclidiennes entre les positions d'une même molécule sur deux cartes entraînées ensemble par Stargate-GTM. Aléatoire correspond à la distance entre deux molécules prises au hasard sur deux cartes ISIDA entraînées ensemble.

Combinaison de paysages ISIDA

Cette méthode est uniquement basée sur la GTM. Une carte représentant l'activité sur une cible biologique (ChEMBL3717) construite sur des vecteurs ISIDA a été combinée à une série de cartes de l'espace latent de l'AE colorées par valeurs de descripteurs moléculaires ISIDA. Un vecteur de descripteurs optimum sur la carte ChEMBL3717 est ensuite utilisé pour réaliser des requêtes sur les différentes cartes de l'espace latent. Celles-ci ont permis d'identifier une zone dans l'espace latent de l'AE potentiellement liée à une zone d'activité sur ChEMBL3717 dans l'espace des descripteurs moléculaires ISIDA. Cette zone de l'espace latent de l'AE a été exploitée pour générer 10.000 structures dont la correspondance avec la zone identifiée dans l'espace des descripteurs moléculaires ISIDA a été analysée.

Ces différentes tentatives n'ont pas été fructueuses. Les molécules générées à partir des vecteurs latents de l'AE ne correspondent pas aux composés décrits avec les descripteurs moléculaires ISIDA. La correspondance entre espace latent d'un AE et les descripteurs moléculaires ISIDA, si elle est théoriquement attendue, apparaît donc très difficile à formaliser. Cette conclusion est renforcée sans équivoque lorsque sont calculés le coefficient de corrélation de Hilbert-Schmidt entre l'espace latent de l'AE et différents espaces de descripteurs moléculaires. Une telle corrélation n'existe quasiment pas ce qui signifie qu'une relation entre ces deux espaces est nécessairement très non-linéaire.

Conditional Variational AutoEncoder (CVAE)

Une nouvelle architecture de réseaux de neurones employant la technologie de l'attention retrouvée dans les couches Transformers^[13] a été développée au cours d'une collaboration entre le laboratoire Chemoinformatique et l'Université de Kazan. Les séquences SMILES servent d'entrée à un Conditional Variational Autoencoder (CVAE). Un Variational Autoencoder (VAE) fait correspondre aux vecteurs d'entrée une distribution de probabilité dans l'espace latent. Ceci offre des meilleures garanties de continuité dans l'espace latent d'un VAE en comparaison d'un AE : une perturbation d'un vecteur latent est moins susceptible de produire de grands changements dans la structure chimique générée correspondante. Cela ne résout pas les problèmes de fragmentation de l'espace chimique en raison de la sémantique des SMILES, mais combiné à une architecture semblable à celle des Transformers, améliore sensiblement les capacités de reconstruction des graphes des molécules. Enfin, les vecteurs de descripteurs ISIDA sont utilisés pour conditionner l'espace latent du VAE. Au travers d'une couche d'attention multi-entrée (Multi-Head Attention, MHA) ces vecteurs de descripteurs biaisent l'échantillonnage de l'espace latent de la VAE (Figure 9).

Un vecteur de descripteurs moléculaires ISIDA peut ensuite être utilisé en requête pour générer des vecteurs latents qui sont ensuite décodés en structures chimiques dont les descripteurs moléculaires ISIDA sont similaires au vecteur demandé. La couche MHA est l'élément qui permet d'introduire la non-linéarité indispensable pour faire correspondre ces espaces chimiques latents et descripteurs moléculaires.

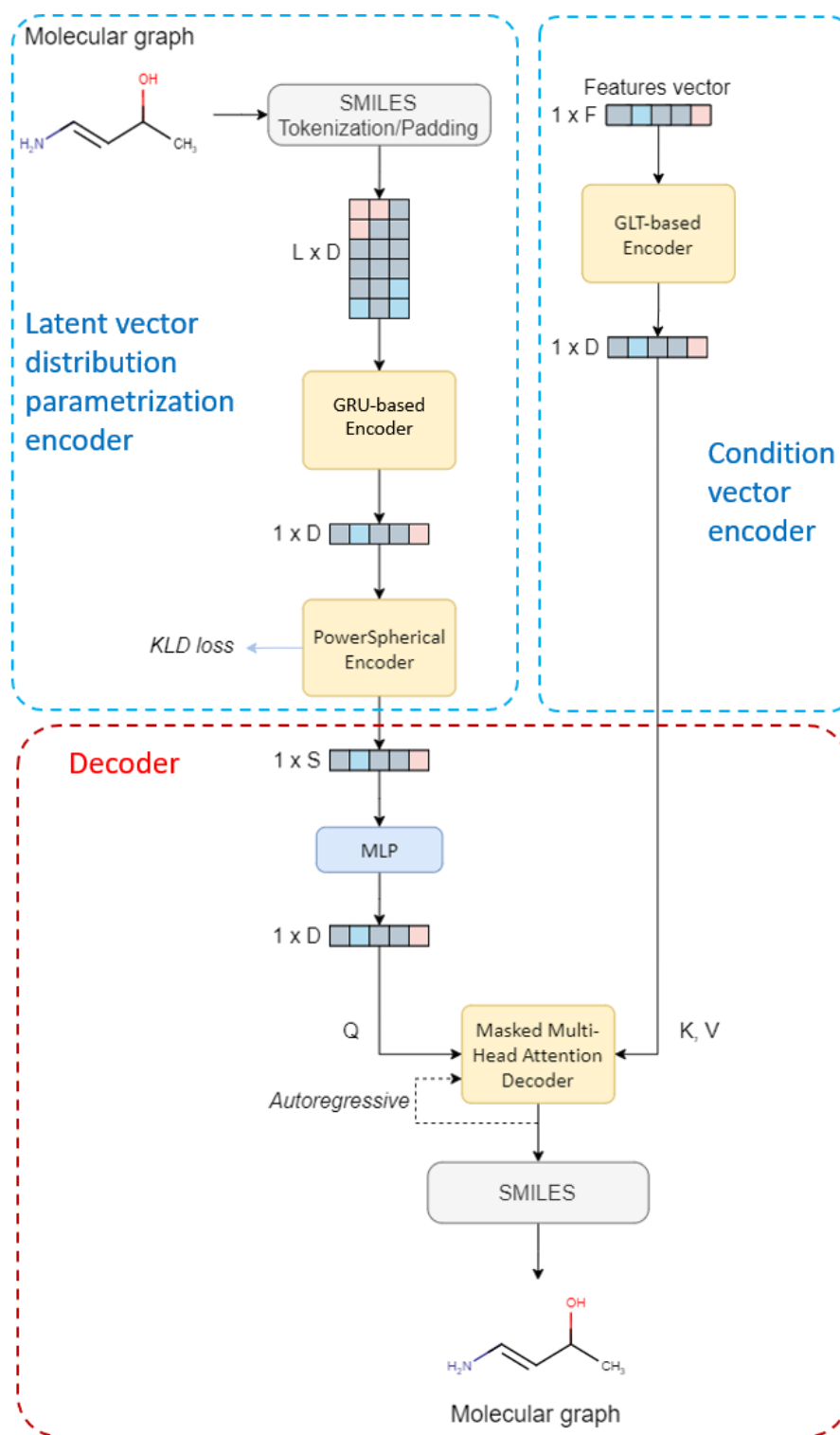


Figure 9. Architecture du modèle CVAE développé qui permet d'échantillonner des structures chimiques dont les descripteurs moléculaires correspondent à une requête. Ici, les descripteurs structuraux ISIDA sont utilisés. La couche d'attention multi-entrée (Multi-Head Attention) est indispensable pour prendre en charge la relation très non-linéaire entre l'espace chimique des descripteurs moléculaires et l'espace latent de la VAE.

Ce modèle a été entraîné sur la base de données ChEMBL23 sous forme de SMILES canoniques en combinaison avec des vecteurs ISIDA utilisés pour construire une « carte universelle ».^[14]

Plusieurs méthodes ont été testées pour sélectionner des vecteurs de descripteurs ISIDA correspondant à une haute activité contre la protéine tyrosine kinase ABL (ChEMBL1862). Des vecteurs optimisés par Algorithme Génétique (GA) basés sur des prédictions des modèles de régression à vecteurs supports (Support Vector Regression, SVR), des vecteurs sélectionnés à partir de molécules étant reconnues comme actives et des vecteurs sélectionnés sur un paysage GTM de l'activité sur ChEMBL1862 ont été utilisés comme « seed » pour la génération. Les différentes méthodes de sélection de vecteurs ont permis la génération de différents profils de molécules (Table 1).

Les molécules générées via l'algorithme génétique montrent une tendance à être très similaires structurellement, avec un potentiel d'activité très élevé. Cette tendance est retrouvée lors de la génération à partir de molécules actives, le potentiel de ces molécules étant légèrement moins élevé qu'avec la méthode algorithme génétique. Les molécules générées à partir de la méthode GTM montrent une plus grande diversité mais un potentiel actif moins haut. Le profil des bibliothèques de molécules générées peut donc être modulé selon la méthode de sélection de vecteurs choisis.

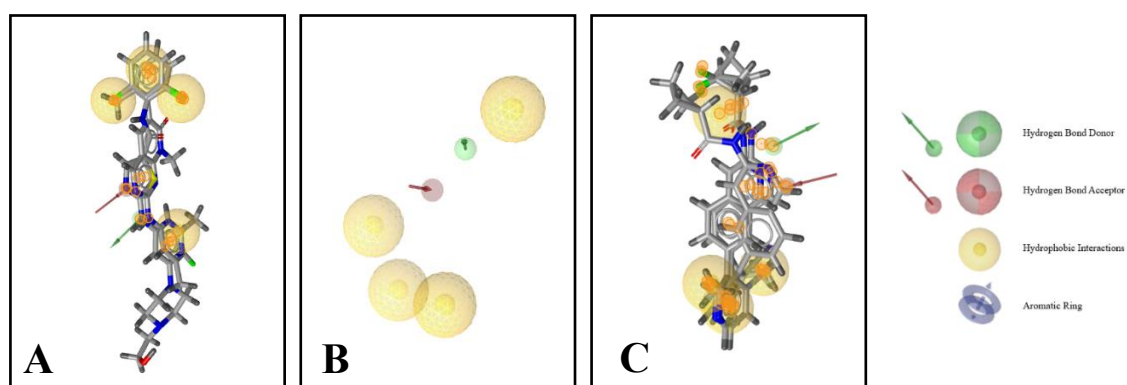


Figure 10. (a) Modèle pharmacophore aligné avec les structures cristallines des deux ligands existants. (b) Modèle pharmacophore (c) Potentiels hits issus de la génération par GA alignés avec le modèle pharmacophore

Le potentiel de ces composés a été confirmé par des études pharmacophoriques (Figure 10) et de docking (Figure 11).

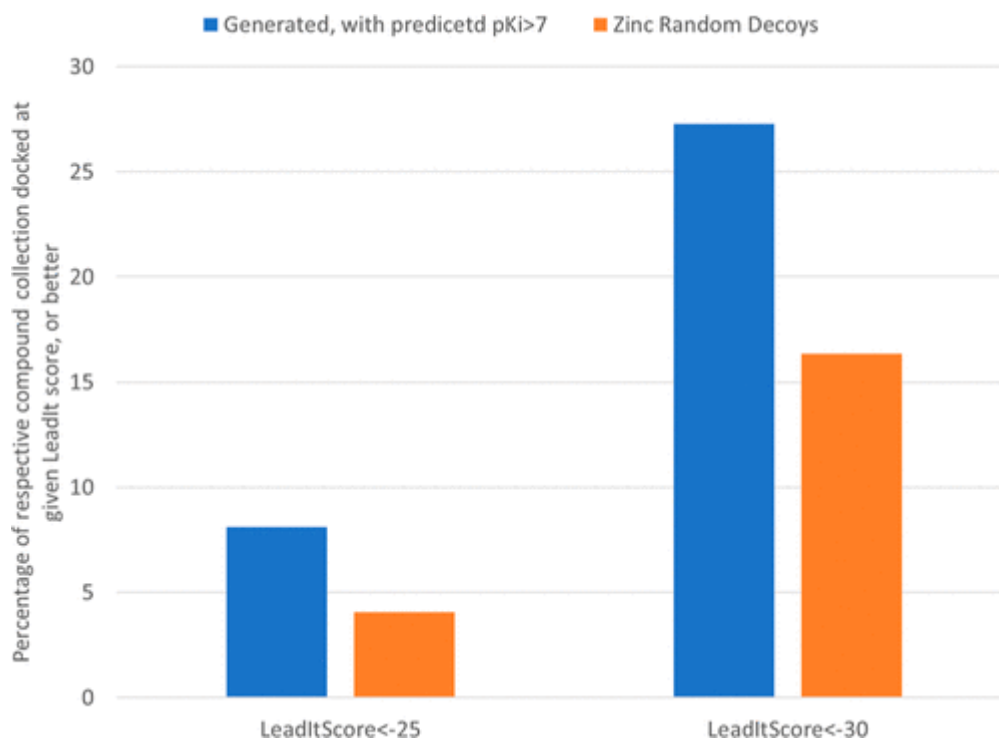
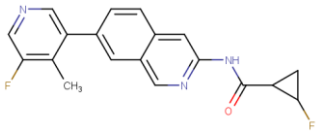
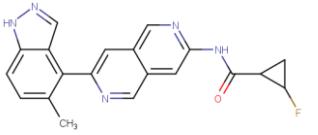
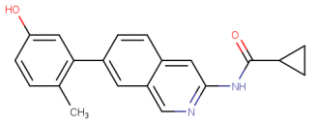
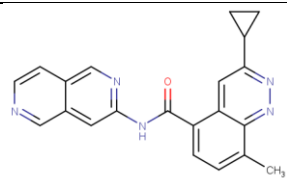
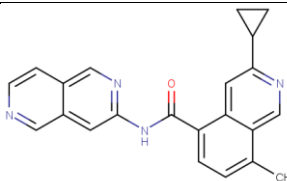
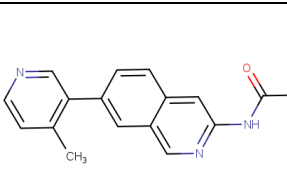
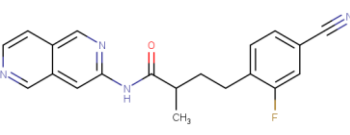
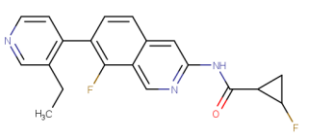
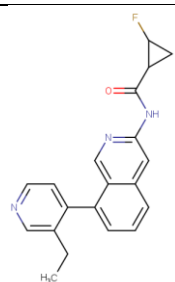
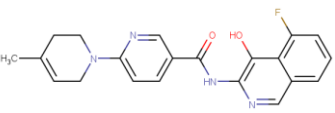
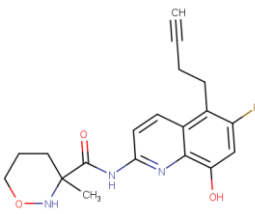
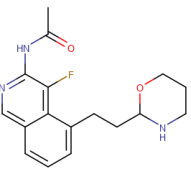


Figure 11. Pourcentage de molécules générées à partir de la GTM et de la SVR (bleu) et pour les leurres ZINC (orange) ayant un score de docking LeadIT comparable à celui d'actifs validés expérimentalement.

Ainsi, l'architecture développée a permis d'établir un lien entre modèle génératif et espace de descripteurs structuraux et par conséquent représente un outil efficace pour effectuer des études de QSAR inverse.

Table 1. Exemples de molécules issues de ChEMBL, et de molécules générées à l'aide des différentes méthodes de sélection de vecteurs. Les valeurs correspondent à l'activité prédite sur la cible ChEMBL1862. Exemples of ChEMBL and generated compounds (with different vector selection methods) with their associated predicted activity against the ChEMBL1862 target.

Molécules ChEMBL		
		
10.73	10.70	10.70
Molécules générées à partir du GA		
		
10.20	9.84	9.82
Molécules générées à partir d'une molécule existante active		
		
10.08	9.45	9.35
Molécules générées à partir de la GTM		
		
7.88	7.84	7.83

1.2.4 Conclusion

Les outils de cartographie couplés à l'espace latent des AE ont permis de montrer que l'organisation de cet espace permet le regroupement général de familles chimiques et la génération de composés structurellement proches des molécules réelles présentes aux alentours. Il a aussi été possible de montrer la dépendance de la topologie de l'espace latent des AE à la sémantique du codage SMILES.

Grâce à l'utilisation de graphes condensés de réaction avec un autoencodeur, il a été possible de générer de nouvelles réactions. Les nouveaux types de transformations chimiques ont été identifiées en utilisant des motifs structuraux CGR correspondant aux cœurs de réaction.

Les tentatives pour faire correspondre des descripteurs moléculaires aux vecteurs latents d'un autoencodeur se sont soldés par des échecs qui ont mis en évidence le caractère non trivial d'une telle relation. Cette observation a été renforcée par les observations effectuées sur les compatibilités des espaces chimiques à l'aide du coefficient d'Hilbert-Schmidt.

Ceci a conduit au développement d'une nouvelle architecture combinant espace de descripteurs ISIDA et AE variationnel qui a finalement permis de générer des structures chimiques dont les vecteurs de descripteurs structuraux correspondaient aux contraintes exigées. Il a été possible de montrer que les composés proposés ont montré de bons résultats tant au niveau de la similarité structurelle que de l'activité biologique potentielle.

1.2.5 Liste des Présentation

Bort, W., Baskin, I. I., Gimadiev, T., Mukanov, A., Nugmanov, R., Sidorov, P., Marcou, G., Horvath, D., Klimchuk, O., Madzhidov, T. & Varnek, A. Discovery of novel chemical reactions by deep generative recurrent neural network. *GGMM SFCi (Group of Graphism and Molecular Modeling & French Society of Chemoinformatics)* à Lille, 31 Septembre 2021.

Poster

Bort, W., Baskin, I. I., Gimadiev, T., Mukanov, A., Nugmanov, R., Sidorov, P., Marcou, G., Horvath, D., Klimchuk, O., Madzhidov, T. & Varnek, A. De novo design of chemical transformations using deep neural networks. *Journée Scientifique des doctorants UMR7140* à Strasbourg, 5 Mai 2021. **Oral**

1.2.6 Liste des Publications

Bort, W., Baskin, I. I., Gimadiev, T., Mukanov, A., Nugmanov, R., Sidorov, P., Marcou, G., Horvath, D., Klimchuk, O., Madzhidov, T. & Varnek, A. Discovery of novel chemical reactions by deep generative recurrent neural network. *Sci Rep* **11**, 3178 (2021). <https://doi.org/10.1038/s41598-021-81889-y>

Bort, W., Mazitov, D., Horvath, D., Bonachera, F., Lin, A., Marcou, G., Baskin, I. I., Madzhidov, T., Varnek, A. Inverse QSAR: Reversing Descriptor-Driven Prediction Pipeline Using Attention-Based Conditional Variational Autoencoder. *J. Chem. Inf. Model.* **62** (22), 5471-5484 (2022). <https://doi.org/10.1021/acs.jcim.2c01086>

2 General Introduction

The continuous search for new potential drugs in medicinal chemistry is a never-ending quest. Far from the early days of medicine and its eat-the-plant-then-see-what-happens approach, today's drug design methods involve the efficient navigation^[15–17] of so-called “drug-like chemical space”^[18–20] which designates the ensemble of all drug-like molecules, existing or tangible. Explored areas of chemical space are a result of centuries of research in Chemistry^[21], supported by the developments of more and more advanced extraction, synthetic and analytic methods. The more recent advances in computing technologies have accelerated this process, notably thanks to new methods like combinatorial chemistry^[22] or high-throughput screening (HTS)^[23]. Each year, the number of new compounds reported increases, constantly expanding the size of the known chemical universe. Historically, Virtual Screening (VS) of existing databases using an arsenal of different tools like pharmacophores^[24,25], docking^[26,27], QSAR^[28–30] or molecular dynamics^[31,32] has been the dominant method for efficient exploration and hit discovery.

However, not unlike our actual universe, it seems the size of the charted areas amounts to very little compared to the vastness of uncharted territories. The size of drug-like chemical space has been estimated to contain between 10^{23} and 10^{60} compounds^[1,33,34]. Comparatively, the biggest commercially available database, Enamine REAL^[35], contains 29 billion compounds. The novel drugs of the future may be hidden among these yet unknown molecules but most of the current discovery methods rely on existing databases of listed compounds. Therefore, although very successful, VS methods are, by definition, limited in their scope of research by the borders of current knowledge. Recent studies of chemical space^[36,37] have sparked a strong interest in its untapped potential and a renewed interest in methods aiming to explore unknown areas of chemical space.

The most popular of these methods to benefit from this newfound interest was De Novo drug design^[38–40]. De Novo design aims to generate compounds from scratch with desirable physicochemical and physiological properties. Early “structure-based” de novo tools used algorithms to identify and map potential binding zones, then stochastically grow molecular structures inside the pre-mapped protein pockets^[41–43] either atom by atom^[44] or fragment by

fragment^[45] and implied the pre-existent knowledge and comprehension of a protein binding pocket through X-Ray crystallography^[46,47], NMR^[48], or electron microscopy^[49,50]. Atom-based methods resulted in a much larger and more diverse number of potential candidates but with questionable synthetic accessibility, while fragment-based methods generated a lower number of more feasible compounds but relied on existing fragment datasets which still limits the exploration potential. So-called “Ligand-based” methods were developed in parallel for protein targets with no available solved structures and relied on known ligands to recreate pharmacophore-based pseudo-receptors or perform direct similarity design^[51]. The advantage of de novo design is that the generated molecules follow precise binding criteria, have certain designated structural features or physicochemical properties, and are usually novel. With it, the usual QSAR workflow could be inverted to generate compounds from desirable given characteristics. However, the prior knowledge necessary to build a molecule from smaller building blocks in a mapped protein pocket and the computational costs necessary to power the algorithms, have limited the scope of applications of the method.

In parallel with the developments in De Novo design, the “Renaissance” of Artificial Intelligence (AI) took place in the early 21st century long after the first introductory experiments by Newell and Simon in 1956^[52]. Having passed through a couple of “winters” in the 70s and 80s due to the lack of results, lack of funding, and unrealistic expectations of end-users^[53], the field of Artificial Intelligence began to gain some new traction at the beginning of the 21st century fuelled by the reducing costs and rapid increase in computing power^[54]. The popularization of Machine Learning (ML) methods using statistical data to form predictions made its way to the field of chemistry to form Cheminformatics^[55–57] and very quickly the exponential increase in global data saw the emergence of a new trend in Artificial Intelligence: Deep Learning (DL).

Deep Learning is a subset of Machine Learning that uses large Artificial Neural Networks (ANN) architectures to handle large amounts of data with reduced preprocessing and gained popularity in the era of Big Data. DL methods have been used in various tasks like natural language processing^[58], image recognition^[59] or even protein folding^[60] and naturally made its way to chemical applications like drug discovery^[61]. One of the many fields impacted by the democratization of DL algorithms was De Novo design. Around 2017, several different types of Deep Generative Models, initially used for language translation^[62] or chat bots^[63], were used

in combination with the Simplified Molecular-Input Line-Entry System^[64] (SMILES) or Graph-based representations to generate new molecules^[65–70].

The mass generation of novel chemical compounds via DL algorithms raised the issue of synthetic accessibility. For hits to be considered viable by the pharmaceutical industries, they must be reachable through cheap and simple reactions using readily available building blocks. Classical rule-based reaction prediction algorithms exist^[71,72] but require the manually-inputted reaction rules and constant expert supervision. To accompany the rise in compound generation, DL models predicting synthetic pathways and synthetic accessibility were developed^[72–74]. The goal was to not only accelerate the retrosynthesis process to match the speed of molecular generation, but also to harness the strong pattern recognition capabilities of DL algorithms to find new potential reaction pathways, unseen by synthetic chemists.

Nowadays, two main trends of generative methods can be characterized: Models based on specific scoring functions aiming to create highly focused libraries, or more general models based on the creation and exploration of a model-based chemical latent space. Architectures based on scoring functions include Generative Adversarial Networks (GANs)^[69], Adversarial AutoEncoders (AAEs)^[5], or any model based on Reinforcement Learning (RL)^[75] or Transfer Learning (TL)^[65]. The role of the scoring function is to orient the generation process towards a very specific subset of, for example, active compounds against a specific target. Generated compounds are compared to existing actives using a set of predefined criteria, resulting in high scores if the criteria are met. The advantage is that molecules obtained this way have a strong potential to be highly active and the structural features and properties outputted by the model can be controlled. However, this also implies that each model is hyper-specific to a unique target and must be retrained if the objective changes, increasing computing and temporal costs. Architectures based on the exploration of a learned latent space are more universal and can be tasked to generate more varied compounds^[76] since the models are trained with large varied molecular datasets. Even though they may correctly separate classes, navigating these “AI-chemical spaces” in search for active clusters remains a challenge due to their highly dimensional nature.

“Chemography”, a combination of Chemistry and Geography, is the art of mapping chemical spaces to facilitate their exploration^[77] and was initially based on the use of Principal Component Analysis^[78–80] (PCA). Highly dimensional chemical spaces could therefore be

reduced to easily readable and comprehensible 2D maps. Different methods of dimensionality reduction techniques exist like previously stated PCA, LDA^[81], t-SNE^[82] or even Autoencoders^[3], a particular type of Deep Neural Network. Another one of these techniques, Generative Topographic Mapping^[7] (GTM), based on Self-Organizing Maps^[83] (SOM), is non-linear and probabilistic which makes it well adapted to handle large amounts of chemical data. Due to its nature, GTM allows the creation of smooth landscapes, in which areas of chemical spaces can be coloured according to the properties of the compounds residing there, like physicochemical properties or biological activity. QSAR models based on GTM showed the potential of this method to find active zones in chemical space and isolate compounds of interest^[84–86].

GTM has been successfully used before in combination with Deep Neural Networks to navigate the chemical space of a generative model, isolate active areas and generate compounds with good activity potential^[87] for a particular target. Although successful in this particular application, there is no certainty that latent descriptors can be successfully used to separate actives and inactives in all cases. Still today, little is known of the construction mechanism and the organization of NN-based chemical spaces, and the robustness and flexibility of latent descriptors compared to classical structural descriptors. In contrast, ISIDA^[8] descriptors have been shown to be versatile in terms of active and inactive separation for several hundred different biological targets.^[14] However, the many-to-one nature of classical structural descriptors renders the simple act of going from descriptor vector to molecule impossible. Therefore, both classical and latent-based methods have strong advantages but each lacks one essential feature. By combining the two, it may be possible to obtain a universal generative model, able to efficiently separate classes for a variety of different targets and navigate latent space in search for active clusters to sample from.

This thesis is therefore dedicated firstly to the analysis of the construction of chemical spaces by deep neural networks, especially Autoencoders, and their generative ability in terms of active compounds and novel reactions. The second part of the thesis is orientated towards the development of a method to harness the generative power of neural networks to couple it with the versatility and efficiency of classical structure based ISIDA molecular descriptors to allow the controlled generation of molecules with desired activities, structures, and properties, reversing the classical QSAR methodology.

2.1 Sequence-to-Sequence Neural Networks

Sequence-to-Sequence (Seq2Seq) models are a type of Recurrent Neural Networks^[88] (RNN) first introduced by Sutskever and al. in 2014^[89]. The initial function of seq2seq models was Natural Language Processing (NLP), specifically English to French translation. However, the architecture was later derived for many other applications in different domains, like text summarisation^[90,91], image processing and captioning^[92], conversational models^[93,94], and even music generation^[95]. These types of models have been extensively used in Cheminformatics and drug design in recent years^[66].

Seq2Seq models are composed of two parts: An Encoder and a Decoder, which are two separate entities, but linked and trained simultaneously. The role of the encoder is to process a variable-length input vector and generate a fixed-length “latent” numerical vector which encapsulates contextual information about the input data. The decoder receives the latent vector and regenerates a variable-length sequence from the given context. For example, in the case of an English to French language translation task, the encoder receives a sentence in English and outputs a numerical vector which conceptually represents the sentence in a latent space. Then, a decoder trained to navigate said latent space can use that vector to output a sentence in any language it was trained on, for example, French.

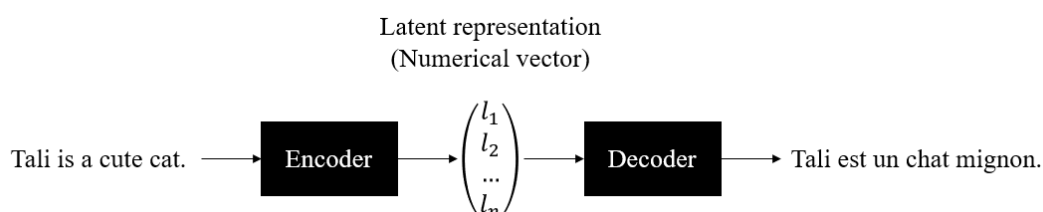


Figure 12. Schematic representation of the basic function of a Seq2Seq model on an English to French translation task.

Encoders and decoders used to be constructed with simple feed forward RNN architectures when Seq2seq models were first introduced. However, vanishing gradients quickly became an issue when trying to train deep architectures^[96], causing models to struggle to maintain contextual links between words far apart in a sequence. To tackle that issue, Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) “cells” were introduced in 1997^[97] and 2014^[62] respectively. LSTM solved the issue of vanishing gradients by allowing

the model to selectively access memory states not only from the last input but from earlier inputs as well. GRUs were developed later as a simpler and faster alternative to LSTMs which performed almost comparably^[98].

Some Seq2Seq architectures are trained to simply reconstruct their input while minimizing the reconstruction error. These types of architectures are called Autoencoders, and their main purpose is to learn in an unsupervised manner a “meaningful” higher representation of the input data. However, if the dimensionality of the latent vector, meaning the vector which is given to the decoder by the encoder, is higher than dimensionality of the input, then the model will simply learn an identity function. To avoid the problem of the model learning to simply “copy” its input, different regularization methods exist, like in sparse^[99], denoising^[100] or contractive^[101] AEs. Those methods work well if the dimensionality of the latent vector is equal or higher to the dimensionality of the data. Another regularization technique consists in making the dimensionality of the latent vector lower than the input data, creating a “bottleneck” which forces some information loss and trains the model to keep the most vital representation and context and infer the missing information (Figure 13).

Even with the introduction of LSTMs and GRUs, a study by Cho and al.^[102] showed encoder-decoder architectures still had a strong dip in performance when dealing with very long sentences. The encasement of the entire context into a fixed-length vector was pinpointed as the main source of error and led to the development of Attention-based seq2seq architectures^[13,103,104]. During the generation process, the Attention mechanism allows the model to selectively concentrate on relevant parts of the source data where the most important information is located. The model can then predict new words according to the global or local context vectors, depending on the Attention type, as well as all previously generated words, thus improving performance on long sentences and translational efficiency altogether.

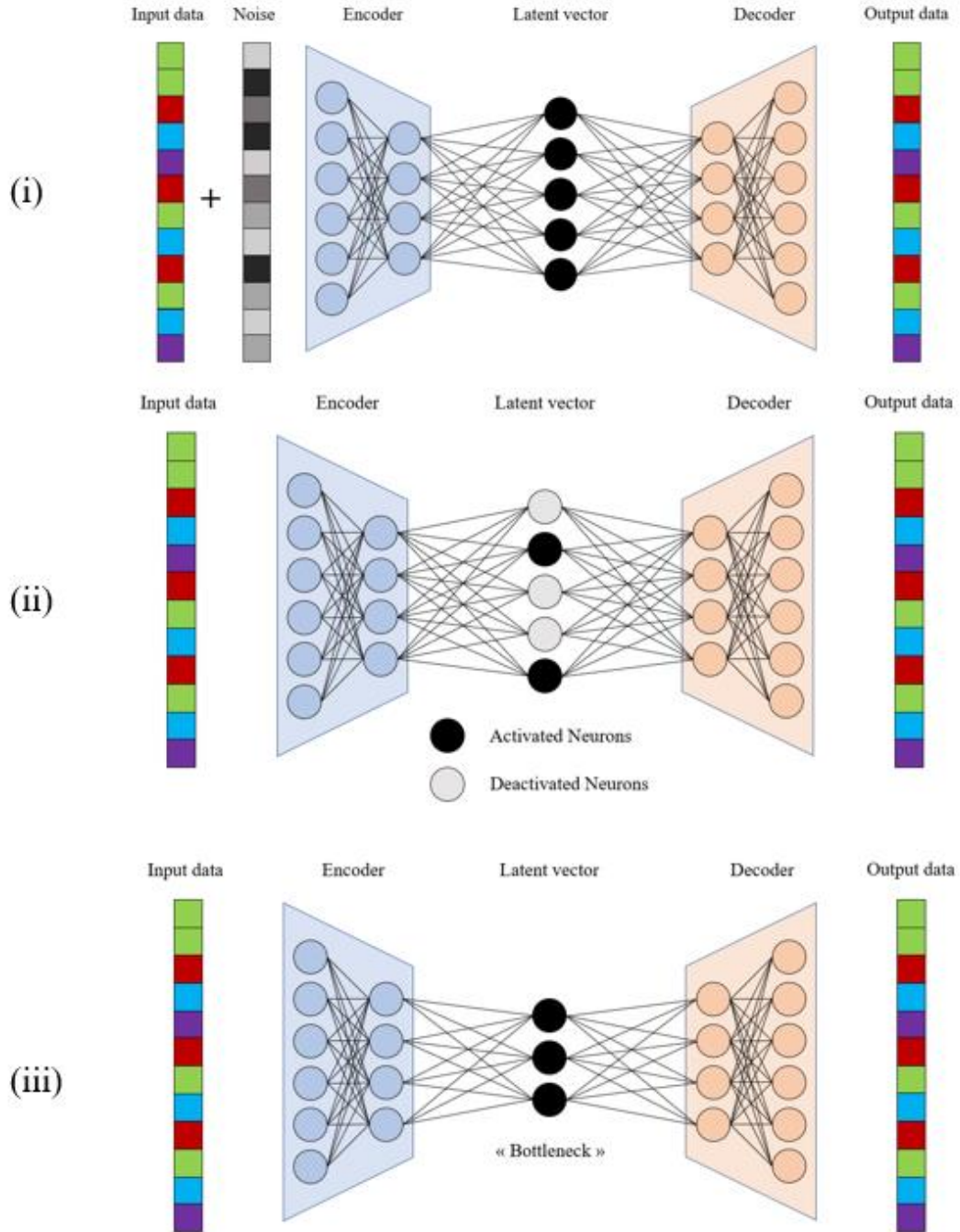


Figure 13. Examples of autoencoder regularization techniques. (i) represents the denoising autoencoder, where a noise vector is added to the input, but the output is compared to the clean input, forcing the model to make the difference between useful information and noise. (ii) represents a sparse autoencoder. Random values are “deactivated” (set to 0) between encoder and decoder, resulting in information loss and forcing the model to infer from incomplete information. (iii) represents an autoencoder with a low-dimensional latent vector, called a “bottleneck” which compresses the information. The decoder must infer the output from this compressed information.

2.1.1 RNNs, LSTMs, GRUs, Attention

Recurrent Neural Networks

A Recurrent Neural Network takes as input a sequence of vectors $\mathbf{X} = (\mathbf{X}_0, \mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_n)$ and processes the vectors one by one, starting with the first one. At time step t , the model processes the input \mathbf{X}_t as well the state vector \mathbf{h}_{t-1} resulting from the previous iteration to generate an output \mathbf{Y}_t and an updated state vector \mathbf{h}_t . Figure 14 shows a schematic representation of the process.

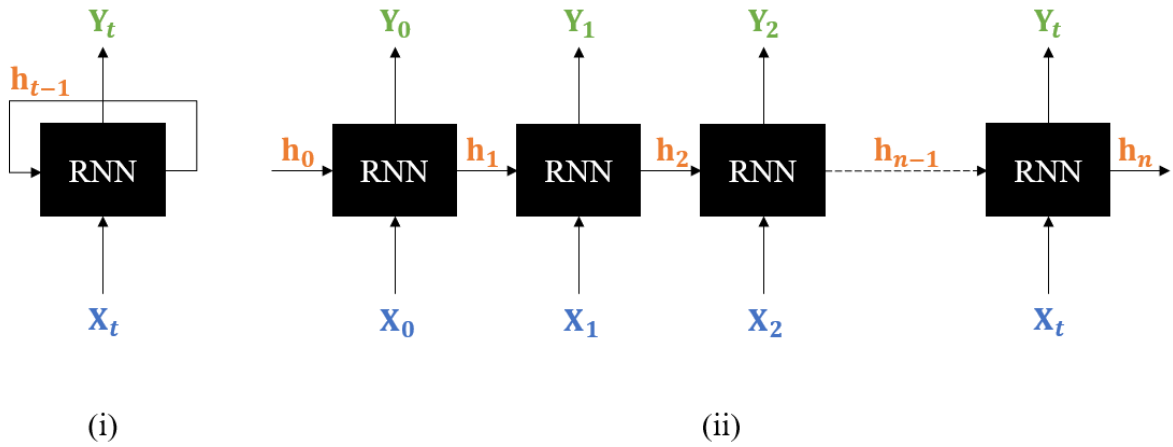


Figure 14. Schematic representation of a folded (i) and unfolded (ii) RNN. The folded scheme shows the feedback loop, current input and previous states are used to compute the output and the updated states. The unfolded scheme allows more readability and a step-by-step understanding of the process.

The current state vector \mathbf{h}_t is a function of \mathbf{X}_t and \mathbf{h}_{t-1} , and the output vector \mathbf{Y}_t is a function of the current state vector \mathbf{h}_t . If $\boldsymbol{\theta}$ represents the trainable parameters of the model then the system evolves as follows:

$$\mathbf{h}_t = f_{state}(\mathbf{X}_t, \mathbf{h}_{t-1}, \boldsymbol{\theta}) \quad (2.1)$$

$$\mathbf{Y}_t = f_{output}(\mathbf{h}_t, \boldsymbol{\theta}) \quad (2.2)$$

Typically, the following equations are used to compute the states and outputs:

$$\mathbf{h}_t = \Phi_{state}(\mathbf{W}_{xh}^T * \mathbf{X}_t + \mathbf{W}_{hh}^T * \mathbf{h}_{t-1} + \mathbf{b}_h) \quad (2.3)$$

$$\mathbf{Y}_t = \Phi_{output}(\mathbf{W}_{yh}^T * \mathbf{h}_{t-1} + \mathbf{b}_y) \quad (2.4)$$

\mathbf{b}_h and \mathbf{b}_y are the biases for the hidden layer and the output respectively, \mathbf{W}_{xh} , \mathbf{W}_{hh} and \mathbf{W}_{yh} are the weights matrices associated with the layer connections. Φ_{state} and Φ_{output} are non-linear activation functions, usually tanh in the case of Φ_{state} and sigmoid, softmax or the rectified linear unit (ReLU) for Φ_{output} depending on the application and the desired output form.

During training, the loss function L of the model is calculated as the sum of losses at each time step as follows:

$$L = \sum_{t=0}^T l(\mathbf{Y}_t^{\text{data}}, \mathbf{Y}_t^{\text{pred}}) \quad (2.5)$$

The nature of the individual loss function l depends on the form of the output, the context of training and the required task. When dealing with sequences of words or characters which are a classification problem, binary or categorical cross-entropy are common choices. Mean Square Error (MSE) or Mean Absolute Error (MAE) are mostly used in regression tasks. The loss is backpropagated through the model at each time step. During this step-by-step backpropagation, the global gradient which is a multiplication of localized gradients can become exponentially high or low if the model is deep (many layers), leading to the exploding or vanishing gradients issues respectively. Long-term dependencies between words in a sequence can therefore be affected if the gradient becomes smaller, as the sentence gets longer.

Long Short-Term Memory and Gated Recurrent Units

To solve the problem of long term-dependencies, Hochreiter and Schmidhuber^[97] introduced the LSTM in 1997. An LSTM works like an RNN but adds different connexions to the cell and a new variable: the cell state. As explained previously, hidden states in RNNs keep the context from previous inputs, but the longer the character chain is, the less effective that

context will be. In the LSTM, hidden states work as short-term memory, and cell states as long-term memory which prevents vanishing gradients and greatly strengthens long-term dependencies. Comparative schemes of a regular RNN and an LSTM are shown in Figure 15.

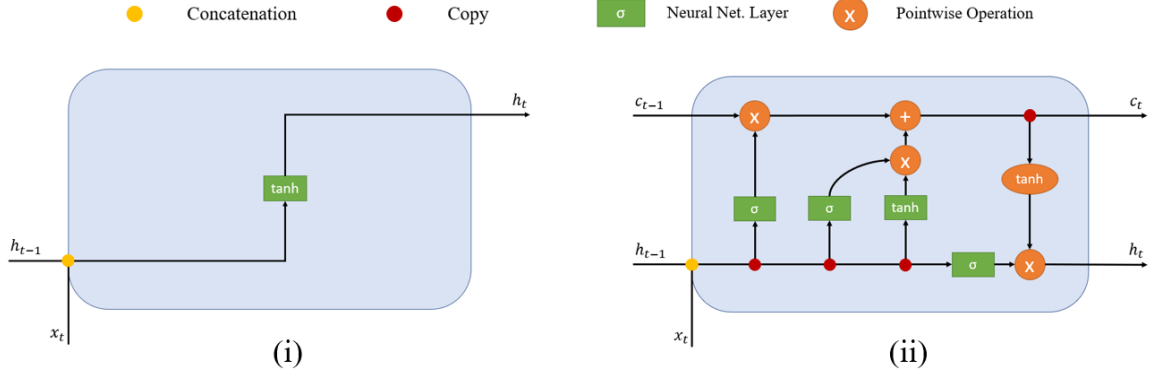


Figure 15. Schematic representations of a regular RNN (i) and the LSTM (ii). The main difference is the cell state c_t which is not present in a regular RNN.

In contrast to the regular RNN where hidden states and input are simply concatenated and passed to an activation (\tanh) layer, the LSTM takes the input and hidden states and passes that signal through “gates” which are composed of an activation layer and either a multiplicative or additive pointwise operation (three red dots on the bottom line in Figure 15, (ii)). These gates decide how much of the new information x_t and short-term memory h_{t-1} to add to the cell state c_{t-1} which serves as the long-term memory. c_t is then multiplied to h_{t-1} to create the output h_t . The first gate is a “forget” gate, which removes or diminishes the importance of certain information in the cell state; the second and third gates work as a “remember” gate, which adds (via the additive operation) new useful information to the cell state.

GRUs function a bit differently, as they combine long-term and short-term memory in the same hidden states h_t (Figure 16). The result is a simpler RNN model with less variables and parameters to optimize which makes it faster to train and use.

The complex mathematical equations and functions governing both these models will not be described here since they are not useful for the understanding of the work, however a precise and thorough mathematical explanation of the feed-forward and backpropagation processes can be found in the literature^[105].

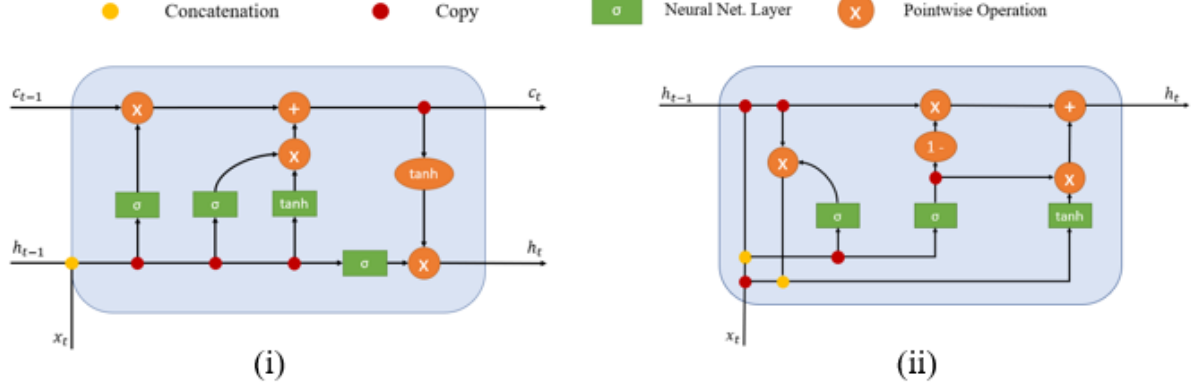


Figure 16. Comparison between schematic representations of the LSTM (i) and the GRU (ii). The GRU does not have a cell state like the LSTM but combines short-term and long-term dependencies into the hidden states.

Different types of RNNs can be distinguished depending on the input and output dimensionality as well as the time steps. The different types are summed up in Table 2.

Context vectors created when reading a sequence from beginning to end are used by the decoder to generate new items from past context. RNNs using this method are called Unidirectional as they only read information one way. Bidirectional RNNs^[106] read the sequence from beginning to end, and from end to beginning and output two hidden states vector, one forward and one backward. The forward vector, just like in a Unidirectional RNN, captures past context during the generation process while the backward vector captures future context. The combination of both context vectors during the generation process allows past and future context to be considered when sampling new words. General hidden states at time step t , h_t are expressed as a concatenation of forward states h_t^f and backward states h_t^b such that:

$$h_t = [h_t^f, h_t^b] \quad (2.6)$$

In traditional RNNs and Autoencoders, the intermediate hidden states of the encoder h_t are always given to the next time step but are not stored individually. Instead, only the final hidden states (and/or cell states in the case of LSTMs) are given to the decoder. This means that the whole context is stored in a single fixed-length vector as shown in Figure 17.

Table 2. Different types of RNNs with their schematic representation and examples of their usage. T_x and T_y represent the length of the sequences \mathbf{X} and \mathbf{Y} respectively; two different model types for different usages can be isolated if the sequences lengths are the same or different.

Type	Representative scheme	Usage example
One-to-Many		Music Generation, Image captioning.
Many-to-One		Sentiment Analysis, Stock price forecasting
Many-to-Many $T_x = T_y$		Named entity recognition
Many-to-Many $T_x \neq T_y$		Language translation

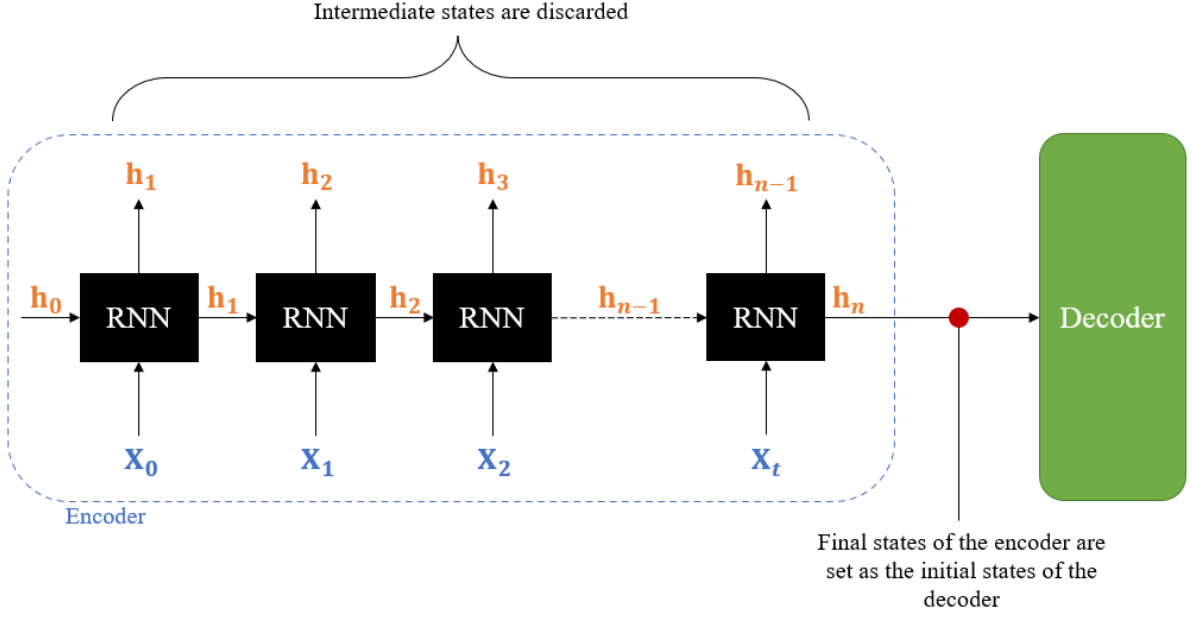


Figure 17. Schematic representation of a Classical RNN. At each time step, the input \mathbf{X}_t and previous hidden states h_{t-1} are given back to the model to generate new hidden states h_t . At the end of the n time steps, the hidden states h_n are transferred to the decoder. The intermediate state h_1 to h_{n-1} are discarded.

Attention

The first Attention mechanism was developed by Bahdanau and al.^[103] in 2014, to solve the issue of long term dependencies in DNNs. The Attention mechanism is based on using the intermediate hidden states h_t to create a dynamic context vector which is then given to the decoder instead of the usual final hidden states. This dynamic context vector is different for each time step of the decoder and allows to form localized connections between source and target sequences. Simply put, for each time step, the context vector has a higher influence from elements in the source sequence that are relevant to that time step specifically, instead of having the same context for all time steps.

During decoding, the basic equations of the Bahdanau attention mechanism are as follows: Let Y_{t-1} and s_{t-1} respectively be the output of the decoder and the hidden states of the decoder at time step $t - 1$. The hidden states h_i are pre-computed from the input $\mathbf{X} = (\mathbf{X}_0, \mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_n)$ by the decoder. At each time step, an attention score $e_{t,i}$ for each hidden state h_i is calculated with the hidden state of the previous output step s_{t-1} following the equation:

$$e_{t,i} = f(s_{t,i}, h_i) \quad (2.7)$$

with f an alignment function which in this case is additive:

$$f(s_{t,i}, h_i) = \mathbf{w}^T \tanh(\mathbf{W}[h_i; s_{t-1}]) \quad (2.8)$$

$[A; B]$ being the concatenation of vectors A and B.

Or

$$f(s_{t,i}, h_i) = \mathbf{w}^T \tanh(\mathbf{W}_1 h_i + \mathbf{W}_2 s_{t-1}) \quad (2.9)$$

\mathbf{w} , \mathbf{W}_1 and \mathbf{W}_2 are weights which are trained alongside with the rest of the model.

Once the alignment scores $e_{t,i}$ are calculated, a softmax function is applied to obtain the corresponding attention weights:

$$\alpha_{t,i} = \text{softmax}(e_{t,i}) = \frac{\exp(e_{t,i})}{\sum_{i=0}^n \exp(e_{t,i})} \quad (2.10)$$

$\alpha_{t,i}$ can therefore be considered as probability or weights values, indicating how important the hidden state h_i is to the next output Y_t and next output state s_t . Finally, the context vector c_t is computed using the combination of $\alpha_{t,i}$ and h_i as follows:

$$c_{t,i} = \sum_{i=0}^n \alpha_{t,i} h_i \quad (2.11)$$

The context vector is then given to the decoder along with the previous output y_{t-1} and previous states s_{t-1} to compute the new output y_t .

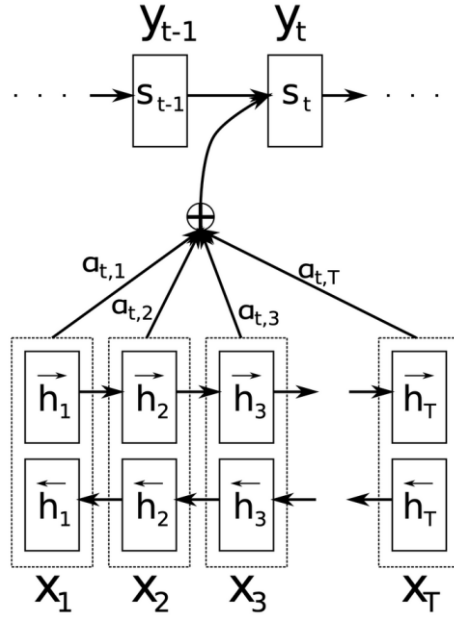


Figure 18. Schematic representation of the process of computing the Bahdanau attention taken from the original publication^[103]. This image shows the process for a bidirectional RNN.

The original Bahdanau attention can be considered as additive since the alignment functions concatenate or add the states together. Luong and al. improved on the original design by using multiplicative alignment functions^[104]. Attention mechanisms can be separated in two categories: local and global. Global Attention (Bahdanau and Luong) makes use of all intermediate states to generate the context vector while local attention selects the most important hidden states in the source sequence to generate the next word in the target sequence^[107].

In 2017, Vaswani and al. introduced a general attention mechanism which does not require the model to be recurrent^[13] and was based only on positional embeddings and multi-head attention (multiple instances of single attention vectors are calculated, concatenated, and projected to give a single context vector). These models, called Transformers, achieved great results in many different fields^[108].

2.1.2 Molecular Representations for Seq2Seq architectures

Seq2seq architectures require sequences of data as input, as opposed to graph-based architectures which work with encoded molecular graphs^[109,110]. To ensure that seq2seq models can be used in chemical applications, chemical compounds must be converted in sequential machine-readable format. Common molecular names like “propane” or “benzaldehyde” would be simple inputs, however they bring no information on the structure or properties of the compound. In contrast, IUPAC nomenclature would be more systematic and complete, but the resulting names can be lengthy and hardly interpretable. Several different forms of molecular representation were therefore developed throughout the years to try and strike a compromise between chemical information retention, simple interpretability, and performance optimization.

SMILES

The Simplified Molecular Input Line Entry System (SMILES)^[64] is a method of encoding molecular structures in the form of a sequence of characters representing the succession of atoms in a molecule. SMILES are built by selecting a starting atom and going through connections in the molecular graph, each time adding the corresponding atom character to the chain. Different atoms, bond types, branching, etc... are all handled using simple characters which make SMILES easily readable and a very light option for storage.

Any atom can be selected as the starting point when building SMILES, which means that one molecule can have several different associated SMILES in a one-to-many relationship. As such, so-called “canonization” algorithms^[111,112] were developed to make the generation process systematic and reproducible, so that each molecule is associated with a single “canonical” SMILES.

InChI

The International Chemical Identifier (InChI)^[113] represents molecular structures as a unique combination of machine-readable layers of character strings. The basis of the InChI representation is the core layer which describes the “skeleton” of the molecule. Different layers may then be added, separated by a “/”, that each provide different chemical information. InChI in essence are a very versatile but quite complex way to represent molecules. Attempts at using InChI to train models showed that the high complexity factor resulted in inferior performance when compared to SMILES-based models^[114,115].

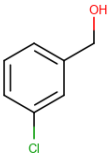
DeepSMILES

SMILES strings may sometimes be very long in the case of structures with several cycles or branching paths. In some of those cases, one opening parenthesis may be at the beginning of the SMILES string when the closing parenthesis is at the end, same with cycle numbers. These placements introduce long-distance dependencies in the sequential representation which are sometimes difficult to apprehend for seq2seq architectures. During reconstruction, small errors can appear when closing cycles or ending branches which can render the SMILES chemically invalid. DeepSMILES^[116] were developed to solve this problem by using single symbols for both cycles and branching paths. Branching paths are indicated by closing parentheses, the number of which indicates the size of the branching. The issue of pairing ring closure is handled by using a single symbol at the ring closing location, indicating the number of atoms in the ring.

SELFIES

As previously stated, small differences or changes in SMILES can result in invalid strings which are not associated with any existing chemical structures, which makes this representation quite prone to errors. This is problematic when trying to use generative neural networks as the probability-based reconstruction may incorrectly handle long-term dependencies and place certain characters where they shouldn't be. SELFIES^[117] are an adaptation of SMILES, robust to small changes and errors. SELFIES are based on formal Chomsky type-2 grammar and localize branches and rings, so that instead of indicating the beginning and the end of a ring or branch, the corresponding symbols indicate the length of the features. All SELFIES character sequences correspond to valid molecules and every molecule can be expressed as a SELFIE.

Table 3. Example of each type of representation for (3-chlorophenyl)methanol.

2D Structure	
SMILES (Unique)	<chem>OCc1cccc(Cl)c1</chem>
InChI	<chem>InChI=1S/C7H7ClO/c8-7-3-1-2-6(4-7)5-9/h1-4,9H,5H2</chem>
DeepSMILES	<chem>OCccccCl)c6</chem>
SELFIES	<chem>[O][C][C][=C][C][=C][C][Branch1][C][Cl][=C][Ring1][#Branch1]</chem>

2.1.3 One-hot encoding

Seq2Seq architectures cannot directly input sequence of characters or words however, as these are incompatible with the numerical transformations which take place in the model. Instead, they should be “encoded” into machine-readable format in the form of vectors, matrices or tensors which can be manipulated through mathematical functions. Sentences in any language are built using a finite list of words organized in a certain order to create meaning and sense. When predicting the next word in a sentence, the model must choose between a certain number of possibilities, making this a categorical problem.

By assigning an integer value to each word or character in the dictionary, sentences can be encoded into numerical vectors. This is called “integer encoding” and is easily reversible, making it the simplest encoding method. The latter works well when numerical values assigned to the data maintain an ordinal relationship present initially in the data. For example, if the task is to encode reviews for a movie where the possibilities are “bad”, “average”, “good”, then encoding them as 1, 2 and 3 makes sense since they are ordered. However, when working with data with no ordinal relationship, this method of encoding can cause biases. Encoding “cat”, “parrot” and “bison” as 1, 2 and 3 implies that a cat is closer to a parrot than a bison which establishes ordered connections that are not present in the initial data. Moreover, when dealing with large dictionaries or large numbers of possibilities, the encoding integers can become very large which can cause memory and performance issues.

Another method to encode sentences or sequences of characters is called one-hot encoding. In this case, each word or character in the sequence is represented as a vector of length N , N being the size of the dictionary. The vector encodes the presence or absence of each word for the current instance, as shown in Table 4.

Table 4. Example of the one-hot encoding of the word “cute” for a dictionary of size 7. The resulting vector has a dimensionality of 7.

<i>Tali</i>	<i>cat</i>	<i>dog</i>	<i>ugly</i>	<i>cute</i>	<i>a</i>	<i>is</i>
0	0	0	0	1	0	0

Using this method, entire sentences can be encoded into matrices of binary values as shown in Table 5. Note that these matrices can get quite large depending on the size of the given

dictionary, but the binary values allow for simpler processing and low memory usage. Furthermore, there is no bias caused by arbitrary values being assigned to random data.

Table 5. Example of the one-hot encoding of a sentence of 5 words with a dictionary of size 7. Words in blue are the sentence, while words in green represent the dictionary. The resulting matrix has a dimensionality of (7, 5).

	Tali	is	a	cute	cat
Tali	1	0	0	0	0
cat	0	0	0	0	1
dog	0	0	0	0	0
ugly	0	0	0	0	0
cute	0	0	0	1	0
a	0	0	1	0	0
is	0	1	0	0	0

This method can be easily extended to chemical data, particularly SMILES. By considering them as a “sentence” of chemical “words” they can be encoded as naturally as languages. A SMILES database contains a finite number of possible characters which form a dictionary, used to create chemically meaningful sequences. The process is the same as before, as shown in Table 6.

Table 6. Example of the one-hot encoding of the but-3-en-2-ol SMILES with a simple dictionary of 6 characters.

	C	C	(O)	C	=	C
C	1	1	0	0	0	1	0	1
O	0	0	0	1	0	0	0	0
N	0	0	0	0	0	0	0	0
(0	0	1	0	0	0	0	0
)	0	0	0	0	1	0	0	0
=	0	0	0	0	0	0	1	0

Even though the process of forming the one-hot matrices is the same in both cases, differences in the handling of natural languages and SMILES may be noted. On the one hand, SMILES sequences can be very long, up to hundreds of characters. With each SMILES character being treated as if it were a word, this would be equivalent to having sentences of 100 or 200 words. As a comparison, the average length of a sentence in the “Harry Potter” books is around 12 words^[118], emphasizing the need for very strong long-term dependencies when

dealing with chemical information. Errors in natural languages can still lead to understandable sentences if one word is missing, misplaced, or mistranslated, but a single error in a SMILES string can lead to a completely different molecule in certain cases, or more likely to a meaningless character sequence with no chemical meaning. On the other hand, dictionaries for SMILES are much smaller than dictionaries containing words for natural languages. The latter can contain thousands of entries, rendering the one-hot matrices very large, while the former may contain 30-60 possible characters depending on the given task. The problematic of having very long sentences is thus alleviated somewhat by the rather low number of possible atoms and functions.

2.1.4 Molecular generation with Seq2Seq architectures

During training, SMILES are modified to add a start and end character at the beginning and end of the SMILES. For example, the simple C1CCCCC1 would become !C1CCCCC1E if “!” was the start character and “E” the end character. SMILES are then encoded into one-hot vectors and the model is trained, sometimes using the Teacher Forcing (TF) method^[119]. As an example for the generation procedure, an Autoencoder which must reconstruct its input will be considered.

The encoder part is only used during training or for generating latent vectors corresponding to inputted molecules. The generation procedure is only done by the decoder. The latter receives as input a latent vector corresponding to a certain SMILES string along with a start token (“!” in our example). The model then predicts the probability of each possible character to be the next one using the softmax function, based on the given latent vector. The sampled character is added to the sequence and given back to the decoder to predict the next character and so on, until the model predicts an “E” which signifies that the string has ended. An example of the process is shown in Figure 19.

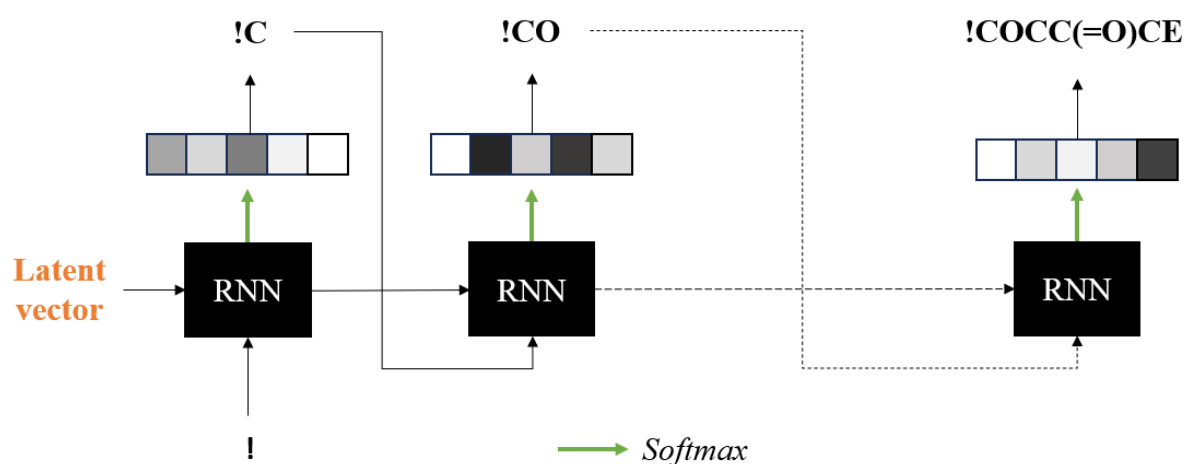


Figure 19. Schematic representation of the generative process. The start character is given to the decoder with the latent vector to predict the next character, based on a probability distribution of all characters (represented by the rectangle). In this case, the most probable character is selected and added to the existing sequence. The latter is then injected back into the neural network to continue predict characters until reaching the “E” which signifies the end.

2.1.5 Seq2Seq Architectures in Drug Design and Reaction Planning

Segler’s work on generating focused libraries with basic RNNs^[65] was the first case of molecular generation of focused datasets using a SMILES-based seq2seq architecture. The method was based on an LSTM-based stacked RNN using Transfer Learning (TL)^[120]. TL or “fine-tuning” consists in training the model on a large, varied molecular database. Then, the model is retrained on a smaller, more specific dataset according to the given task so that the knowledge acquired on the bigger dataset can be used on the more specific task. Segler and al. trained their RNN on the entire ChEMBL database first, then on the more specific target dataset obtained from ChEMBL. Since the first publication, other teams have used TL with LSTM-based stacked RNN^[121–124] to generate focused datasets. Another method is Reinforcement Learning (RL)^[125]. RL is a training algorithm based on applying a scoring function to the output of the model and rewarding or punishing the model according to the score. When applied to drug design, the model generates compounds and a scoring function applies a score to the generated molecule according to preferences in structure, properties, etc... The model is then

rewarded for generating molecules which fit the desired properties and punished otherwise. RNNs coupled with RL have been used to generate potential actives against biological targets^[66,126]. Forcing the model to generate compounds with desired properties can also be achieved by training the model to make the connection between properties and chemical structure. During training, the model is given, alongside the SMILES string, corresponding properties or structural features associated with the given SMILES. When sampling, the model only takes the property vector and outputs a SMILES corresponding to a compound with the desired properties. These types of models are called Conditional Recurrent Neural Networks (CRNN) and have been used to generate active compounds against the DRD2 receptor^[127].

More complex architectures like AEs became quite popular due to their ability to create an explorable latent space in which any area of interest can be sampled using cartography^[87] or Particle Swarm Optimization (PSO)^[128]. Due to the discrete nature of the latent spaces associated to vanilla AEs, VAEs were preferred for latent space navigation^[114,129]. VAEs were also coupled with TL^[126] to design ligands for the dopamine type 2 and the 5-hydroxytryptamine type 1A receptors. CVAEs also achieved good results in the generation of actives against biological targets^[4,130].

Generative Adversarial Networks^[131] are a combination of two separate models trained together: The first model, called the generator, generates compounds that are given to the second model, the discriminator. The discriminator receives the compounds from the generator and compounds from a dataset of real molecules and must learn to make the difference between them. The generator is trained to “fool” the discriminator by creating compounds that resemble the real dataset. Once the discriminator cannot make the difference between real and generated compounds, then sampling the generator gives compounds which are structurally very similar to the real compounds. GANs were successfully used in focused datasets generation^[69,132–134] and latent space exploration^[6]. Instead of comparing molecular structures like in the case of GANs, an AE can be modified to create an Adversarial Autoencoder which compares the data distribution of the AE with a prior distribution, the goal being to bring prior and latent distributions closer so that the generative process is fuelled by latent vectors in interesting areas of latent space^[5].

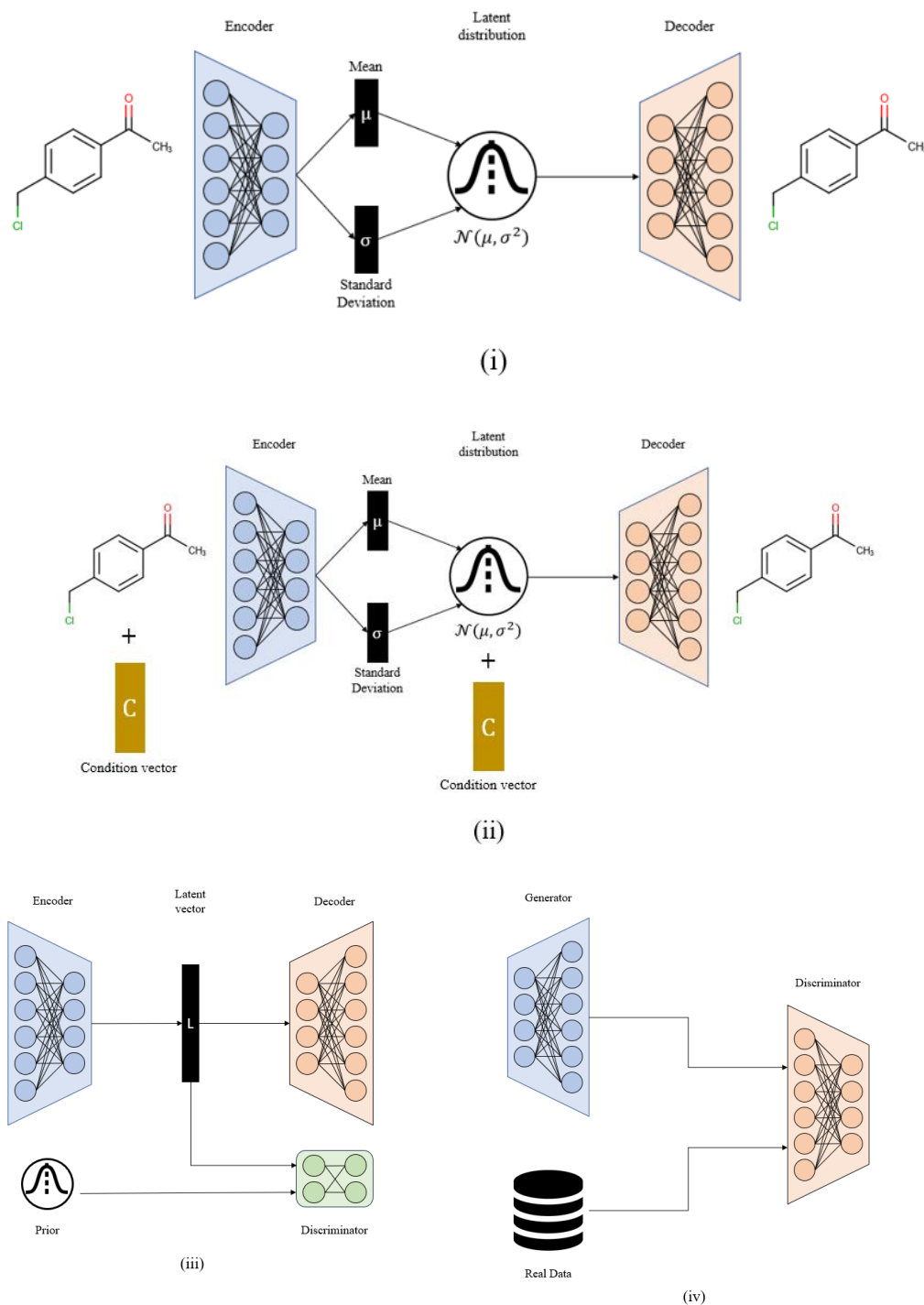


Figure 20. (i) Variational Autoencoder. Instead of the unique latent vector generated by a vanilla AE, the encoder generates two vectors: mean and standard deviation of a normal distribution from which the vector given to the decoder is sampled. (ii) Conditional Variational Encoder. A condition vector describing structural fragments or physicochemical properties is given to the encoder at the same time as the molecular structure and concatenated with the latent vector sampled from the gaussian distribution. (iii) Adversarial AutoEncoder. The encoder generates a latent vector which is compared with a prior distribution by a discriminator. (iv) Generational Adversarial Network. The generator creates molecular structures which are compared to real compounds by a discriminator which is trained in parallel.

The developments of sequence-to-sequence architectures achieving target compound design meant that large datasets of potentially new molecules were created. However, being predicted active is not the only requirement for potential drugs. Simple and relatively cheap synthetic pathways must also be found to ensure that the interesting molecules are also easily accessible synthetically. Reactions are a lot more complex than molecules however, since reactions often imply multiple reagents and products, with a correspondence between chemical entities before and after the reaction. This seemingly complex issue can be simplified slightly by thinking of reactions as a translation from reagents to products, and since seq2seq architectures were initially developed for translation tasks, they can naturally be adapted to solve the problem. GRU and LSTM-based seq2seq models were successfully used to predict products directly from reagents^[135,136] and retrosynthetically predict reagents from products^[137–139]. Reactions remain very difficult to handle and the prediction of novel reactions through neural networks is still in its infancy.

2.2 Generative Topographic Mapping

Generative Topographic Mapping (GTM) is a probabilistic dimensionality reduction technique akin to Self-Organizing Maps^[140], first published by Bishop and al. in 1998^[7]. As a method of dimensionality reduction, visualization and analysis of chemical latent space, GTM has successfully been used for the analysis of large data collections^[85,141–143]. The algorithm performs non-linear projections from an initial N-dimensional descriptor space onto a 2D latent space called manifold which is inserted into the data. The manifold itself is a flexible surface composed of Gaussian Radial Basis Functions (RBFs). It is inserted into the densest regions of the frameset where it adapts by assuming the general shape of the data distribution. Data points are projected onto the manifold via grid points called nodes, and the manifold is then unfolded and flattened into a 2D map.

Each data point is associated to all nodes via a set of responsibilities which encode the position of the data point on the map. The higher the responsibility in regard to a certain node, the closer the data point will be to that location. These responsibility vectors can then be used to create landscapes, 2D maps associated to a certain property or activity where each node is coloured according to the value of the given property or activity.

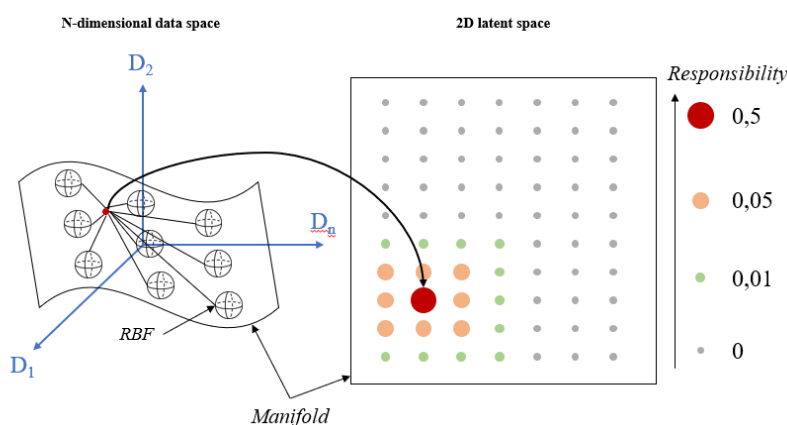


Figure 21. General overview of Generative Topographic Mapping. The data point is projected onto the manifold via the RBFs with a probability to reside in each node. The responsibilities for one data point are normalized over the entire map, meaning the sum of responsibilities is equal to 1.

2.2.1 GTM Algorithm

Generative Topographic Mapping is a probabilistic method, as stated previously. The manifold is composed of a set of M Radial Basis Functions (RBFs, Gaussian functions in this implementation), forming a probability distribution sampled using K nodes. The mapping function \mathbf{Y} used to map items from the latent space of dimension L (in this case, $L = 2$) to the initial space of dimension D is the following:

$$\mathbf{Y} = \boldsymbol{\Phi} \mathbf{W} \quad (2.12)$$

$\boldsymbol{\Phi}$ corresponds the $K \times M$ matrix regrouping the evaluation of each RBF position in relation to each node with the formula:

$$\Phi_{mk} = \exp \left(-\frac{\|\mathbf{x}_k - \boldsymbol{\mu}_m\|^2}{2\sigma} \right) \quad (2.13)$$

With \mathbf{x}_k the position of the node, $\boldsymbol{\mu}_m$ the fixed position of the RBF, and σ the variance associated to the Gaussian functions. A set of K , M , and σ parameters are associated to a constant $\boldsymbol{\Phi}$ matrix and influence the resolution of the model. The deformation of the manifold to adapt to the data is described by the trainable weight matrix \mathbf{W} (2.12) of size $M \times D$, which defines the manifold in the initial D -dimensional space.

Thus, the multiplication of the $\boldsymbol{\Phi}$ matrix (K, M) defining the relations between RBF centres and nodes in 2-dimensional latent space and the \mathbf{W} matrix (M, D) defining the placement of the RBF centres in the initial D -dimensional data space gives the \mathbf{Y} matrix (K, D) defining the shape of the manifold in the initial space. During training, the weights shift to move the nodes closer to the data points, searching for the shape that will best fit the data distribution thus improving the resulting latent representation.

The data distribution is usually defined by a set of N data points called a frameset, which is a representative subset of a usually larger dataset, but sufficient for the training process to capture the general shape of the distribution. The first step of training is the initialization of the weight matrix \mathbf{W} , which can be done randomly but the application of PCA is the preferred method. Simply put, the manifold is inserted “flat” into the data following the eigenvectors of

the two first principal components resulting from a PCA on the frameset. The process is governed by the following equation:

$$\mathbf{W} = \Phi^{-1}(\mathbf{X}\mathbf{U}) \quad (2.14)$$

The \mathbf{X} matrix ($K, 2$) defines the position of the nodes following the two eigenvectors resulting from the PCA and \mathbf{U} ($2, D$) defines the two eigenvectors in initial space. The resulting matrix $\mathbf{X}\mathbf{U}$ gives the coordinates of the nodes in D -dimensional space. To obtain initialized weights, this matrix needs to be multiplied by the inverse of the Φ matrix, essentially defining the positions of the RBF centres in the initial space following the principal components. Once the manifold is inserted into its initial position, each compound of the frameset is projected onto the manifold using the following equation:

$$p(\mathbf{t}|\mathbf{x}_k, \mathbf{W}, \beta) = \left(\frac{\beta}{2\pi}\right)^{-\frac{D}{2}} \exp\left(-\frac{\beta}{2} \|\mathbf{y}_k - \mathbf{t}\|^2\right) \quad (2.15)$$

Equation (2.15) describes the probability density (or likelihood) of a data point \mathbf{t} to be associated with the node k of coordinates \mathbf{x}_k in the latent space. \mathbf{y}_k corresponds to the coordinates of nodes in D -dimensional space calculated using equation (2.12). β is the inverse of the variance of the distribution, initialized based on the third component of the PCA, and optimized during the training procedure. Equation (2.15) can be integrated over all nodes to obtain the likelihood of data point \mathbf{t} against the entire manifold, which is a quality indicator of the manifold's representation of that particular data point.

$$p(\mathbf{t}|\mathbf{W}, \beta) = \frac{1}{K} \sum_{k=1}^K p(\mathbf{t}|\mathbf{x}_k, \mathbf{W}, \beta) \quad (2.16)$$

In practice, the likelihood is not used as is, but its natural logarithm (LLh) is preferred.

$$LLh(\mathbf{t}, \mathbf{W}, \beta) = \ln(p(\mathbf{t}|\mathbf{W}, \beta)) \quad (2.17)$$

Summing the likelihood of the N data points contained in the frameset gives a global value for the quality of the manifold fit and is used as an objective function to optimize the weight matrix \mathbf{W} .

$$LLh(\mathbf{W}, \beta) = \sum_{n=1}^N LLh(\mathbf{t}_n, \mathbf{W}, \beta) \quad (2.18)$$

Expectation-Maximization Algorithm

Based on this objective function, the Expectation-Maximisation algorithm is run to optimize the values of \mathbf{W} and β . The Expectation step first evaluates the normalized responsibilities of the N data points \mathbf{t}_n on every node \mathbf{x}_k with equation (2.21), creating a matrix of responsibilities \mathbf{R} of dimension (K, N) :

$$r_{kn} = \frac{p(\mathbf{t}_n | \mathbf{x}_k, \mathbf{W}, \beta)}{\sum_{k'}^K p(\mathbf{t}_n | \mathbf{x}_{k'}, \mathbf{W}, \beta)} \quad (2.19)$$

$$\mathbf{R} = \begin{pmatrix} r_{1,1} & \cdots & r_{1,N} \\ \vdots & \ddots & \vdots \\ r_{K,1} & \cdots & r_{K,N} \end{pmatrix} \quad (2.20)$$

The second part of the Expectation step is to calculate, for each node, the sum of responsibilities g of the N data points. The result of this summation is expressed as a diagonal matrix and not a one-dimensional vector since the basis must be changed in the Maximization step to go from the 2-Dimensional space of the nodes to the D -dimensional space of the RBF centres via the Φ matrix.

$$g_{kk} = \sum_{n=1}^N r_{kn} \quad (2.21)$$

$$\mathbf{G} = \begin{pmatrix} g_{11} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & g_{KK} \end{pmatrix} \quad (2.22)$$

The Maximization step uses the previously calculated \mathbf{R} and \mathbf{G} matrices, the matrix \mathbf{T} describing the N data points in the initial D -dimensional space, a unit matrix \mathbf{I} and a regularization coefficient λ to compute the updated parameter matrix \mathbf{W}' . Based on the latter, a new value β' for the width is also computed.

$$\mathbf{W}' = (\Phi^T \mathbf{G} \Phi + \lambda \mathbf{I})^{-1} \Phi^T \mathbf{R} \mathbf{T} \quad (2.23)$$

$$\frac{1}{\beta'} = \frac{1}{ND} \sum_{n=1}^N \sum_{k=1}^K r_{kn} \|y(\mathbf{x}_k, \mathbf{W}') - \mathbf{t}_n\|^2 \quad (2.24)$$

Both updated width and updated parameters are reinjected into the Expectation step for a new iteration. An updated $LLh(\mathbf{W}', \beta')$ is computed and compared to the initial $LLh(\mathbf{W}, \beta)$ using a simple gradient indicated in equation (2.25). The training process is stopped when the gradient becomes smaller than the limit (0.001 in this case).

$$\frac{LLh(\mathbf{W}', \beta') - LLh(\mathbf{W}, \beta)}{LLh(\mathbf{W}, \beta)} \leq 0.001 \quad (2.25)$$

GTM ReSample

In certain cases, after the manifold has been trained, the number of nodes may be incompatible with certain applications. For example, the number of nodes in a small manifold may be too little for separating some species, leading to confusing graphical representations. It is possible to change the number of nodes in the manifold by simply reassigning a matrix of node positions from the matrix \mathbf{Y} describing the positions of the RBFs in initial space which do not change since the manifold is well trained and embedded into the data. The probability distributions of the data points can then be recalculated using the new set of node coordinates.

2.2.2 GTM Landscapes

Visualization and modelling of the data is done by using the aforementioned responsibilities to create “landscapes” depicting the data distribution, with the possibility to enhance the displayed information by using colour coding related to properties or in the case of chemical information, activity as well. Depending on the goal and the type of model needed, three types of landscapes can be defined: Density, Class, and Property (see Figure 22).

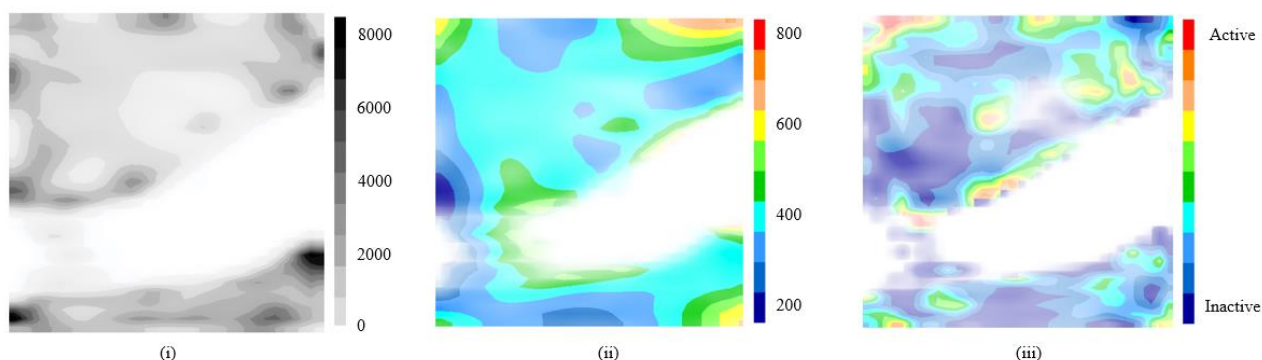


Figure 22. Examples of density, property, and class landscapes. (i) shows a density landscape with very populated areas represented in dark grey or black and lightly populated/unpopulated areas represented in light grey or white. (ii) shows a property landscape, molecular weight in this case, with heavy molecules coloured in red and lighter molecules coloured in blue. (iii) shows a class landscape with zones containing active molecules coloured in red and zones containing inactive molecules coloured in blue. The zones coloured in green contain both active and inactive molecules with similar cumulated responsibility values.

Density landscapes are simply a representation of the cumulated responsibilities in each node.

$$q_k = \sum_{n=1}^N r_{kn} \quad (2.26)$$

For property landscapes, the property value for each node q_k can be calculated by multiplying the property value q_n of each compound in the node by its responsibility r_{kn} , summing all those values per node, and dividing by the cumulated responsibilities.

$$q_k = \frac{\sum_{n=1}^N q_n * r_{kn}}{\sum_{n=1}^N r_{kn}} \quad (2.27)$$

Once the property landscape is created it can also serve as a regression tool. By projecting a new data point \mathbf{t}' on the landscape and obtaining its responsibility vector $\mathbf{r}_{t'}$ with components $r_{kt'}$ for each node, the predicted value for the property of the compound is calculated as follows:

$$q_{t'} = \sum_{k=1}^K q_k * r_{t'} \quad (2.28)$$

In other words, the predicted property is a sum of all properties of the nodes where the data point was projected, weighted by the probability of the data point to be in each node.

In class landscapes, the value for each node is equivalent to the probability of finding a data point of a certain class in it:

$$P(c_i|x_k) = \frac{P(x_k|c_i) * P(c_i)}{\sum_j P(x_k|c_j) * P(c_j)} \quad (2.29)$$

$$P(x_k|c_i) = \frac{\sum_{n=1}^N r_{nk}^{c_i}}{N_{c_i}} \quad (2.30)$$

$$P(c_i) = \frac{N_{c_i}}{N_{tot}} \quad (2.31)$$

Here, $r_{nk}^{c_i}$ is the responsibility of a data point n in node k with class c_i , N_{c_i} is the number of data points with the c_i class and N_{tot} is the total amount of data points.

In the same way as property landscapes, class landscapes can also be used as predictions tools with a formula resembling the property landscape equation. The value for the class of a projected compound \mathbf{t}' can be expressed as:

$$P(c_i|\mathbf{t}') = \sum_{k=1}^K P(c_i|x_k) * r_{kt'} \quad (2.32)$$

Transparency combined with colouring in both property and class landscapes help visualize the data density. It is however sometimes less easily readable in this format, thus the need for a density landscape.

2.2.3 ISIDA descriptors

The basic idea of any QSAR model in chemoinformatics is to link activity or property to structural features with the following general formula: *activity/property* = *f(structure)*. GTM is therefore commonly built on structural fragment descriptors and the landscapes coloured according to chemical properties or activities to achieve that relationship. These descriptors must be carefully selected to have the best possible maps.

ISIDA^[8] descriptors encode molecular structures as specific subgraph counts where nodes correspond to atoms and can be coloured by different local physico-chemical properties such as pH-dependent pharmacophores and electrostatic potential. The vertices in the subgraphs correspond to the bonds; bond information can either be represented or ignored. Different fragmentation schemes can be applied to create different types of subgraphs: linear features, feature pairs, circular features, or feature trees. As such, the different types of colouring, bond information and fragmentation schemes offer a vast range of different levels in which the chemical information may be encoded. This plethora of choices for descriptors allows the creation of very strong GTM models which can be adapted to any biological target.

2.2.4 Combining GTM and Autoencoders

GTM is commonly used with ISIDA descriptors or structural descriptors in general, however the method can technically be applied to any data vector. Using the encoder part of an Autoencoder and SMILES encoded in one-hot-vectors, it is possible to create a high-dimensional latent space of chemical structures where each molecule resides with a coordinates vector which can be used as descriptors. These descriptors can then be used like any other to train GTM models. The added benefit of this combination of methods is the ability to “generate” molecules from selected zones of interest on landscapes. Once the manifold is trained and the activity or property landscape is plotted, coordinates in 2D space of any area can be isolated and reverted to high-dimensional initial space using the reversibility of GTM. These coordinates can be used as input for the decoder to generate molecules residing on the manifold in that area of space. A representative scheme of this process is shown in Figure 23.

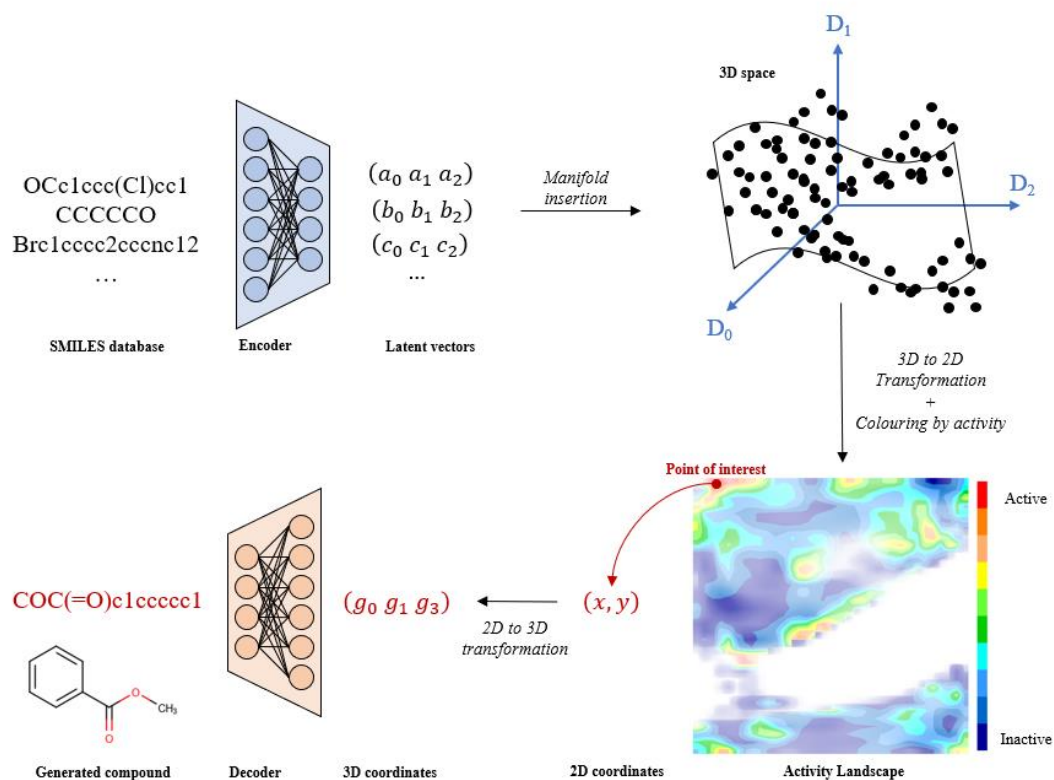


Figure 23. Schematic process of the training process of a GTM based on latent vectors and its usage in the case of generating molecules from zones of interest. The initial space is defined here with 3 dimensions for readability and ease of understanding, however in practice any dimensionality could be used.

By using this method, it is possible to generate molecules inside a data “cluster” of interest, however the coordinates extracted only belong to the manifold which adapts itself to the data distribution but may not cover it entirely as shown in Figure 24, (i). To improve the generation process to cover more of the data distributions, multiple vectors coordinate around the initial point must be sampled. Let $\mathbf{x} = (x_0, x_1, \dots, x_D)$ be a vector of coordinates in initial space of dimension D obtained from the manifold. A new vector \mathbf{y} is obtained by multiplying each coordinate x_i of the initial vector by a random number r sampled from a gaussian distribution centered on 1.

$$y_i = x_i * r, \quad r \sim \mathcal{N}(1, \beta) \quad (2.33)$$

The distribution is centered on 1 so that the multiplication creates a small shift from the initial position. The width of the gaussian must be carefully selected to fit the data distribution. The parameter β in equation (2.33) is the same as in equation (2.4) and describes the width of the RBF centers of the manifold which are optimized to fit the data. Naturally, this value is reused as a sampling width for the new vectors.

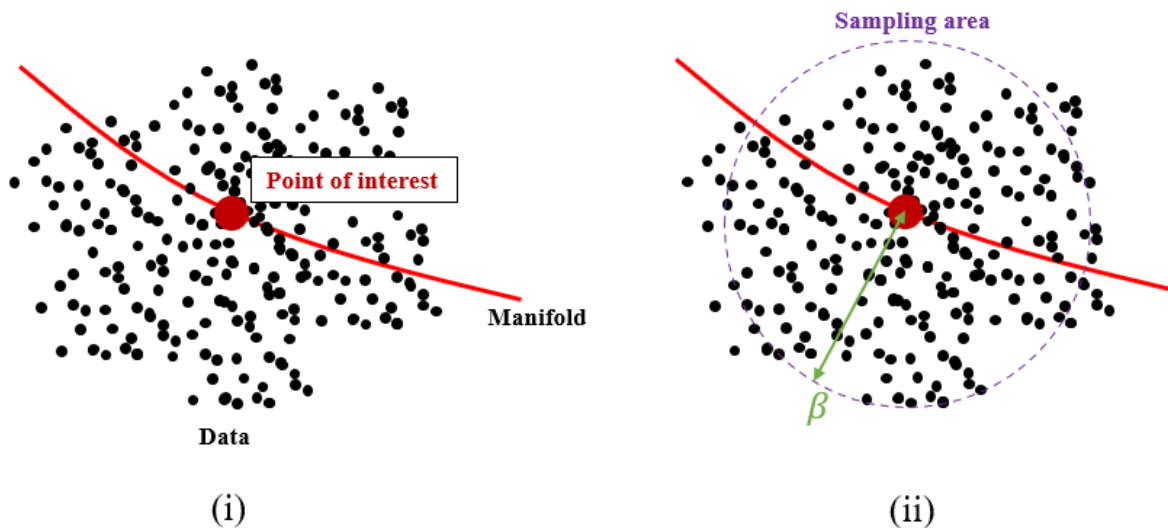


Figure 24. Schematic representation of a manifold passing through a cluster of data points. If that zone on the 2D map is considered of interest, only coordinates on the manifold can be selected for the generation process (i). By multiplying the coordinates of the points of interest in initial space by a gaussian distribution, it is possible to sample not only on the manifold but also the entire data cluster (ii).

3 Exploring the latent space of an Autoencoder

3.1 Introduction

The Autoencoder is a classical but still very promising architecture in deep learning^[144,145]. Its purpose is to encode an object (e.g. a SMILES string) to a latent vector, and then to decode the latent vector back to the initial object's representation as close as possible. Thus, two products of the AEs can be delineated: a latent vector, and decoder's prediction. The first one may be used as a vector of descriptors of a new kind in various machine-learning applications, while the second one allows generating new chemical structures.

AEs have been already used to conduct ligand-based virtual screening using reconstruction errors as scoring^[146]. Nowadays, various modifications of AEs are mainly employed in training QSAR/QSPR models^[147]. In the context of classical QSAR/QSPR models, AEs have no obvious advantage in comparison to classical descriptors (e.g. ISIDA^[8] or ECFP^[148]). However, they become a revolutionary technology when they are combined with a machine-learning method that supports producing a vector of descriptors for a given chemical subspace. Namely, AEs can be used to build maps of chemical space by applying dimensionality reduction algorithms to the latent vectors corresponding to chemical compounds. On such a map, a user may delineate a zone with a desired property, and then generate new latent vectors that correspond the selected zone. After, the decoder of the AE can decode it to a set of SMILES strings corresponding to the newly generated latent vectors. The described methodology has been already applied by Sattarov et al. to perform de novo design of molecules with desired properties using GTM^[87].

AEs can also be used to generate chemical structures by picking them from different parts of the latent space and feeding the latent vectors to the decoder. In this case, VAEs, in which a molecule is mapped to a Gaussian distribution over the latent space from which latent vectors can be sampled, are usually used. This has been implemented by Gómez-Bombarelli et al.^[114] using the latent vectors sampled from the latent space of a VAE. This allows the generation of chemical structures along a trajectory in the latent space either (i) between the latent vectors corresponding to predetermined chemical structures or (ii) from a chosen latent vector in the direction of increase (or decrease) of the considered molecular property (e.g., biological

activity). In the latter, the chemical structure is being optimized to achieve the desired properties. In order to conduct optimization in the right direction, Blaschke et al. used an additional Gaussian processes structure-activity model^[68]. To perform multi-objective optimization in the latent space of a VAE, Zhavoronkov et al. applied reinforcement learning^[149]. One can create AEs that can generate chemical structures with desired properties without the need to perform optimization in the latent space. This can be done, in particular, using conditional VAEs in which the property vector is concatenated with the latent vector to feed the decoder^[4]. In this case, the generation of new chemical structures with desired properties can be performed by sampling from the prior distribution in the latent space, augmenting the generated vectors with the vector of desired properties and converting them to chemical structures using the decoder. New chemical structures with desired properties can also be generated using a semi-supervised VAE trained on a set of existing chemical structures with properties known only for a part of them^[130].

AEs can be used not only to produce latent vectors from string representations of chemical structures but also directly from molecular graphs. Several types of VAEs have been developed for this purposes: JT-VAE^[150], CGVAE^[151], GraphVAE^[152], NeVAE^[153]. Special types of hetero-encoders can also be used to translate between molecular graphs^[154]. An obvious advantage of this approach is the formal correctness of the graphs generated by decoding latent vectors, because in this case AEs do not need to learn the syntax of languages for representing chemical structures, such as SMILES, IUPAC names, etc., using a very large number of examples. For SMILES, this problem can be partially solved with the help of a special autoencoder, GrammarVAE, which is aware of the grammar of the SMILES line notation^[155].

Thus, in the literature there is a significant number of publications devoted to the use of AEs for constructing a chemical latent space and its use to build predictive SPR models and de novo design of chemical compounds with desired properties. Meanwhile, the visualization of chemical latent space using data analysis and visualization methods, i.e cartography are still in infancy. Although some publications reported data distributions on 2-dimensional maps obtained with the dimensionality reduction methods like PCA or GTM, a systematic study of chemical latent space of a Vanilla SMILES-based AEs has not yet been carried out. Such exploration is needed to get answers on the following questions:

1. Is the latent space of an autoencoder consistent?
2. Do similar compounds possess similar latent vectors, and vice versa?
3. Can new chemical structures be generated from any part of the latent space?
4. How novel are the structures generated in different parts of the latent chemical space?
5. What benefits can be gained from the analysis of the latent chemical space?

Different types of AEs and different ways of structures representations may lead to different latent spaces. The purpose of this work is to provide an example of such an analysis hoping that the conclusions drawn will be general in nature, and the methodology used to implement this can be applied in other cases. An AE model was combined with GTM to map, visualize and explore the chemical space of a latent space constructed by a neural network, in the hopes to get a better understanding of the rules governing these relatively new spaces.

3.2 Methods

Visualizing and sampling latent space of the AutoEncoder using GTM

Using the encoder part of the trained AE model developed by Sattarov and al^[87], the entire ChEMBL23 database was encoded into 256-dimensional latent vectors by extracting the bottleneck vectors associated to each input SMILES. These 256-dimensions latent vectors can be seen as a type of molecular descriptors and as such, can be used to train GTM models. Latent vectors were transformed into SVM format and filtered according to their standard deviation. Each descriptor with a standard deviation less than 2% of the maximum standard deviation of all descriptors was eliminated. Additionally, a step of minmax scaling was applied. Filtering and scaling are necessary here to ensure the proper training of GTM. Once GTM was trained, density and property landscapes were built to visualize the latent space of the AE in 2D.

Using the reversibility of GTM, selected coordinates in 2D can be reverted to the initial latent space. By making use of this property and reverting the scaling and filtering applied to train the GTM model, new latent vectors can be obtained, which correspond to the coordinates of nodes. These new latent vectors may be given to the decoder part of the AE, which will generate SMILES associated with these new vectors. However, it is not interesting to only sample from the exact coordinates of the node due to the properties of GTM and its data distribution. Compounds projected in a node, often are not located exactly on the exact

coordinates of the node but scattered around at a certain distance. During training, the GTM algorithm inserts the manifold through the data distribution and nodes can be seen as a sort of average of data distribution. Therefore, sampling from the exact coordinates of a node would be counterproductive since it means only one vector can be sampled, which may or may not correspond to a valid molecule, and the actual data distribution may be slightly distant from these coordinates. To solve this issue, a latent vector corresponding to a node is multiplied by a random vector, issued from an isotropic multivariate normal (Gaussian) distribution, centered on 1 with a width of β . This β value which corresponds to the optimal sampling distance from the node, is equal to the width of the RBF functions in GTM.

1000 SMILES were generated for each node of the GTM model. Generated SMILES were filtered using RDKit^[156], removing any duplicates and invalid SMILES which would correspond to unfeasible chemical structures. Novelty was assessed by comparing the generated compounds to the training database. Any generated molecule absent in the training set was considered novel.

The general workflow of the study was the following: 1) Train the AE, 2) Build GTM landscapes, 3) Sample systematically, and 4) Analyze sampled molecules in terms of chemical properties, novelty, and general distribution in latent space.

Shannon Entropy

The Shannon Entropy of GTM landscapes is used to compare the homogeneity of the distribution of data. Comparing this metric over different AE models and therefore different GTM landscapes based on those models indicates if the models have a similar distribution of data or rather have a completely different organization of latent space. Shannon entropy is computed using the accumulated responsibilities of all compounds projected on a map. For each node, the accumulated responsibility is calculated by adding each molecule's contribution to that node and dividing by the total number of molecules projected on the map.

The Shannon entropy is computed as:

$$E = \frac{\sum_K CumR_k \log (CumR_k)}{\log (K)} * 100$$

With $CumR_K$ being the cumulated responsibility vector for each node k and K the total number of nodes. E ranges from 0 to 100, where 0 means that all molecules are projected in the same node, while 100 means that the projections cover the chemical space equally and uniformly.

3.3 Results

Reproducibility of latent space

The first step to analyse the chemical space of an AE is to test its consistency. It is vital that the results of any experiment be reproducible, and that means that training several AE models from the exact same parameters and training/validation sets should give the exact same results. To verify this, four different AE models were trained, by fixing any random number generation to a given seed in the initialization and selecting the exact same training/validation set split. Results are shown in Table 7.

Table 7. Training and validation reconstruction rates, and number of descriptors after filtering according to standard deviation for all 4 models.

Model	1	2	3	4
Train Reconstruction	99.23%	99.23%	99.54%	98.88%
Val Reconstruction	98.61%	98.25%	98.62%	97.94%
Number of descriptors after filtering	66	60	75	62

Surprisingly, although extremely similar, the results are different for all 4 models. The most surprising is the difference in the number of descriptors remaining after filtering. A difference of up to 15 descriptors can be seen between models 2 and 4. This indicates that fixing the random number generation and the input data to be the same in all cases was not sufficient to obtain 4 times the same model. On further inspection, it becomes clear that the problem is due to the way the calculations are processed on GPU. Calculations done on GPU have a lower precision (32 vs 64 bits) than CPU in this case, significantly accelerating the calculations. However, this also means that small differences in calculations can appear due to the loss of precision. This issue has been studied and a solution has been developed^[157] which could be

implemented in future work to fix it. Over the course of an entire process of AE training, these differences are accentuated to the point where the models become quite different in terms of metrics. To see if these differences in metrics have an impact on the latent space, the 4 density landscapes corresponding to the ChEMBL database for the 4 models were computed and shown in Figure 25.

Even though metrics show the models have different reconstruction rates and significantly different numbers of descriptors, the data distributions in latent space are similar. High density zones, located in the same areas of latent space can be found in all 4 maps, especially in the corners. The biggest difference is the seemingly “empty” region of space in models 3 and 4 which seems to be less present in models 1 and 2. However, although these areas in models 3 and 4 are not densely populated, they are not completely empty, and similar areas of lower density can be found in the same spot in model 1, albeit smaller, and slightly shifted towards the bottom-right corner in model 2. The comparison of the 4 density landscapes indicates that the model may differ slightly in terms of parameters and metrics, but latent space has a similar organization in the 4 of them.

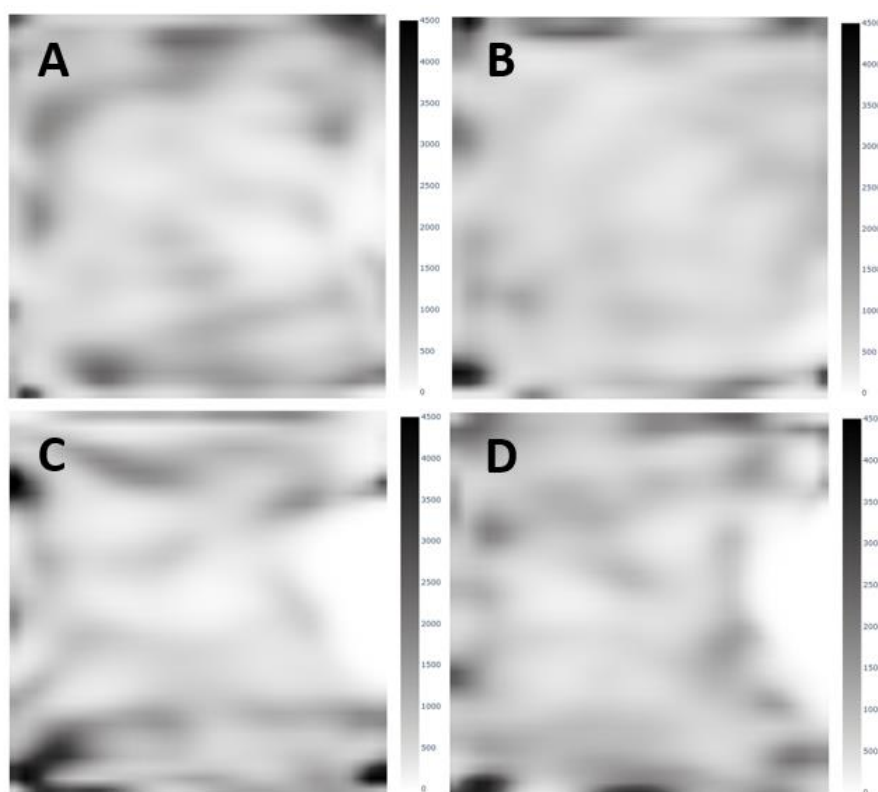


Figure 25. Density Landscapes for the 4 models. (A) model1, (B) model2 (C) model3 (D) model4

Computing the Shannon Entropy (SE) for all 4 models (Table 8) further confirms that the 4 models have a very similar data distribution across latent space. All 4 models have very similar and very high values of SE, indicating that the data is homogeneously distributed across latent space. Model 2 in particular presents no “very low” density zone and few high density areas, making it the most uniformly distributed latent space. In contrast, model 3 shows a lower value of SE (even though still very high) since there is an area of lower density on the right (see Figure 25C) and seemingly more visible high density areas.

Table 8. Shannon entropy computed for the density landscapes of the different models. 0 means that all compounds are projected into the same node, 100 means that compounds are evenly spread across latent space.

Model 1	Model 2	Model 3	Model 4
98.99	99.22	97.55	98.21

SE calculations show that even though the 4 models are slightly different in terms of organization of latent space, and uniformity of data distribution, they are still very similar. Thus, in the rest of the analysis only model 1 will be considered.

Order dependance of SMILES strings in data distribution

One essential idea when generating a descriptor set is that similar compounds need to have similar descriptors and thus, similar position in latent space. Since the AE does not function in terms of structural descriptors but character sequence, it is interesting to consider if the similarity principle is kept in latent space. The idea is that, since the AE considers character sequences, character order may be an important factor in the latent vectors calculation process. Very similar molecules may have very different canonical SMILES strings, with different starting points, which may be a problem for the AE. To test this, the same compound shown in Table 9 was expressed 3 different ways by randomizing the SMILES string using RDKit. Even though they relate to the same molecule, the three SMILES strings are organized differently with very different starting points.

Table 9. SMILES A, B, and C randomized from the given ChEMBL molecule.

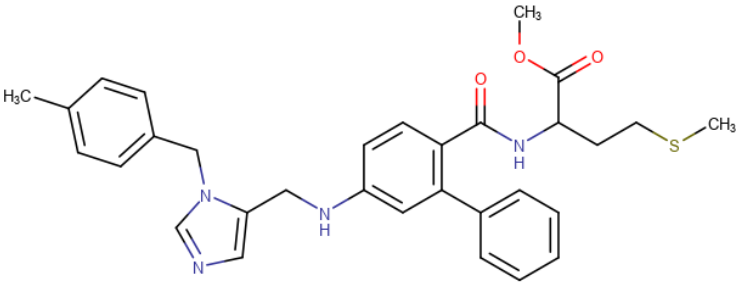
		
A	B	C
<chem>c1cc(C(NC(C(OC)=O)CCS</chem> <chem>C)=O)c(-</chem> <chem>c2ccccc2)cc1NCc1cncn1Cc</chem> <chem>1ccc(C)cc1</chem>	<chem>N(Cc1cncn1Cc1ccc(C)cc1)c</chem> <chem>1cc(-</chem> <chem>c2ccccc2)c(C(=O)NC(C(OC</chem> <chem>)=O)CCSC)cc1</chem>	<chem>c1cc(C)ccc1Cn1cnc1CNc1ccc(</chem> <chem>C(=O)NC(CCSC)C(OC)=O)c(-</chem> <chem>c2ccccc2)c1</chem>

Figure 26 shows the projections of the three SMILES strings on the density landscape of model 1. The same molecule is projected in completely different areas of chemical space, the only reason being the difference in its SMILES representation. This is a consequence of the way AEs deal with input data. Since latent vectors are based on a sequence of characters, changing that sequence of characters also completely changes the values of the latent vectors. Bidirectional LSTM cells read input from both sides, mitigating this problem in some cases where the SMILES starts from the “other side” of the molecule, however, when the SMILES strings are completely different and start from very dissimilar positions, the AE is incapable of relating all the different character strings to the same compound. The similarity principle is therefore not necessarily obeyed in all cases with the latent space of an AE.

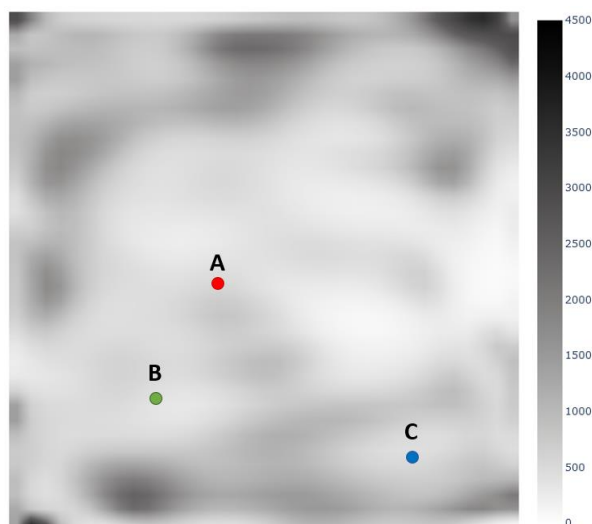


Figure 26. Projections of SMILES A, B and C on the density landscape of model 1

Data distribution in the chemical latent space in model 1

The GTM density landscape constructed for the ChEMBL23 database is shown in Figure 27. The data density is mostly evenly distributed with no empty areas. Some zones of higher density can be observed, mostly located in the corners of the map. The log likelihood value in GTM indicates the “closeness” of a given compound to the manifold. The closer the compound is to the manifold, the higher the loglikelihood will be. The loglikelihood landscape shown in Figure 27 (RIGHT) indicates that compounds on the edge of the map are further from the manifold than the compounds in the center. Especially the low-density zone on the right side of the map seems to be quite far from the manifold.

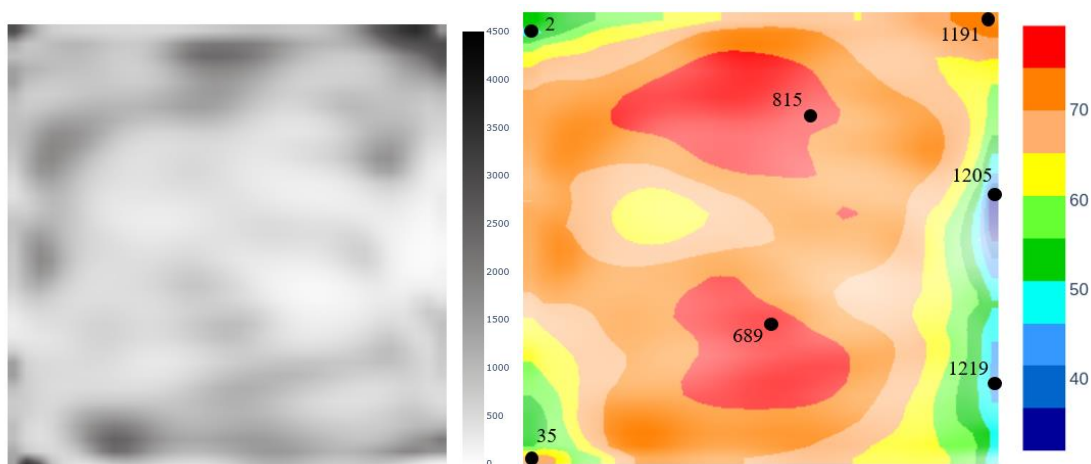
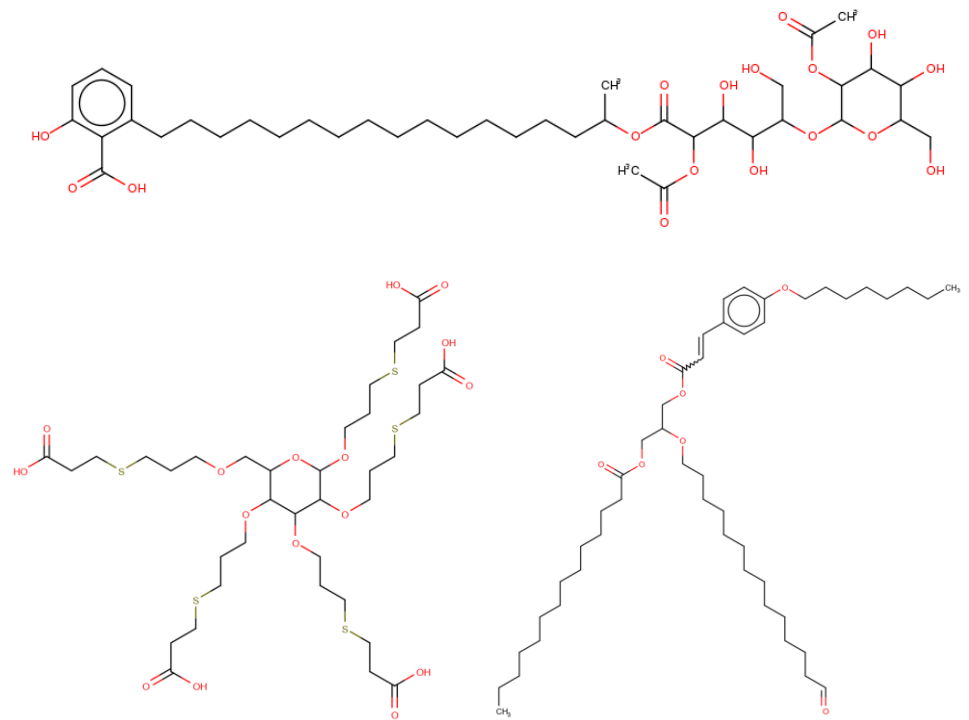
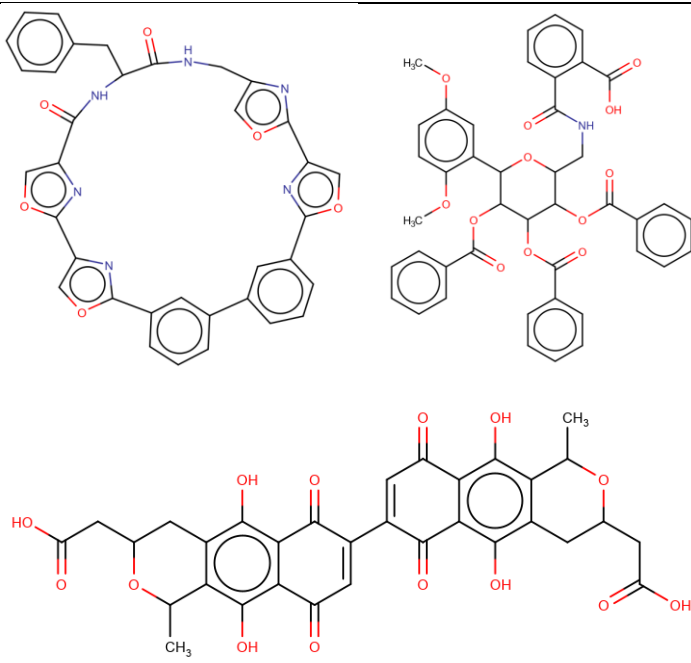


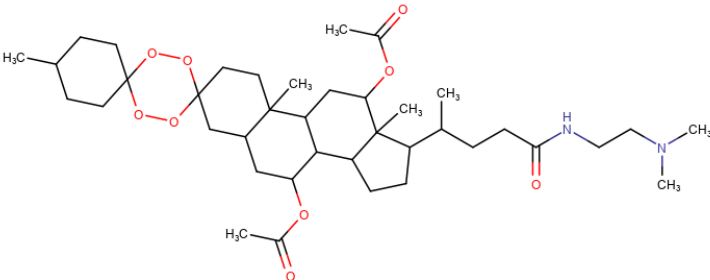
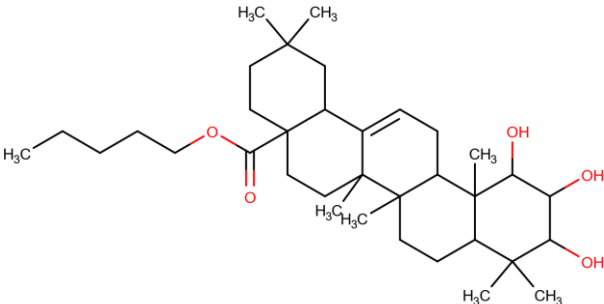
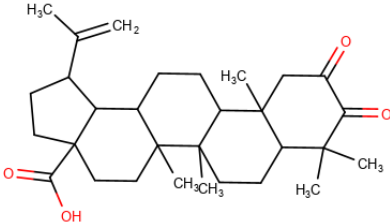
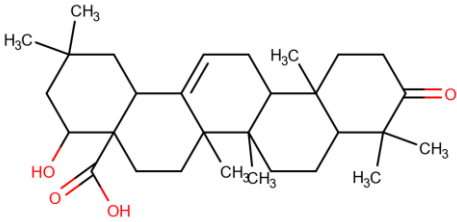

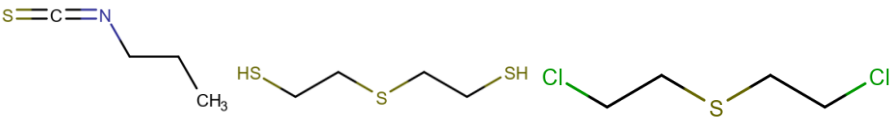
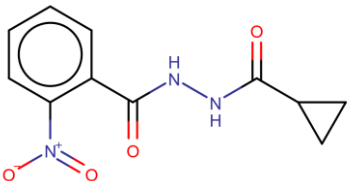
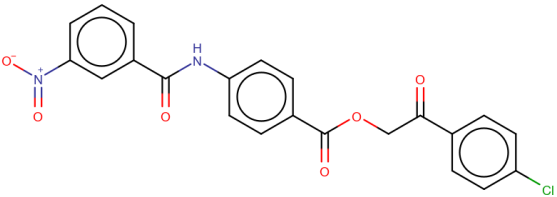
Figure 27. (LEFT) Density landscape for ChEMBL compounds for model 1. (RIGHT) Log Likelihood of projections of the ChEMBL compounds on the map, with nodes of interest annotated.

The zones of low LLh indicate data distributions which the manifold had more difficulty adapting to. This could mean that the data cluster is located far away from the rest of the data distribution or that the cluster is very sparse and spread out and the manifold was inserted in the middle. The low density around node 1205 confirms the hypothesis of a very sparse cluster being the reason for the low LLh. Molecules located in that node almost always contain one to four long carbon chains sometimes containing heteroatoms as is shown in Table 10. This type of compound is quite peculiar in terms of SMILES string (long repetitions of “C” character) and somewhat rare, which would explain their distance from the rest of the data distribution. Node 1219 is more densely populated and contains large cyclic structures or structures with a high number of fused cycles and large amounts of alcohol, carbonyl, or carboxyl functions as well as very low amounts of heteroatoms. The combination of all these factors may have shifted the latent vectors away from the main data cluster and be the cause of the low LLh. Node 2, a high density area (Figure 27), contains steroid-like structures and structures with fused rings but low amounts of aromatic rings (Table 12). The 3 zones of low log likelihood coincide with areas of high molecular weight. Higher molecular weight could imply larger number of characters and further differentiate these clusters from the rest of the data. Interestingly only 3 out of the 4 nodes with high LLh seem to correlate with low molecular weights (815, 1191, 689) with very small molecules in nodes 815 and 689. Node 1191 is populated by molecules containing peptide bonds which are common in drug-like compounds and could explain the good manifold coverage. Node 35 is a very high-density node, containing steroid-like structures although smaller than in node 2. The density of the node and easily repeatable pattern could

mean that this area of latent space is very densely packed, meaning the manifold could adapt well to the data.

Table 10. Examples of molecules for the nodes of interest shown in Figure 27. The nodes are split in two categories: Low loglikelihood (blue) and high loglikelihood (orange)

Type	Node	Examples
Low LLh	1205	
	1219	

	2	 
High LLh	35	 
	815	
	689	
	1191	 

Justifying the manifold coverage and the positions of certain functional groups and patterns remains very complex even with GTM landscapes as a visualization tool. Several factors, like molecular weight (which is related to the number of characters in the SMILES string), repeated patterns (peptide bonds, carbonyl groups, etc...), density of the data clusters, aromaticity, branching and overall complexity of the SMILES string seem to influence the data distribution. Although difficult, it is possible by taking all properties into account to somewhat understand the distribution of compounds in latent space, even if a complete understanding is still impossible.

Generative abilities of the AE

Above, only the maps indicating the distribution of the chemical compounds taken from the ChEMBL database were considered. Such maps could have been obtained using any other set of molecular descriptors. An important feature of using continuous autoencoder latent vectors as descriptors is the ability to convert them into chemical structures using the decoder, which allows for the generation of new molecules. The questions arise: (1) can correct chemical structures be generated from any point in the autoencoder latent space, and (2) what factors influence the generative process?

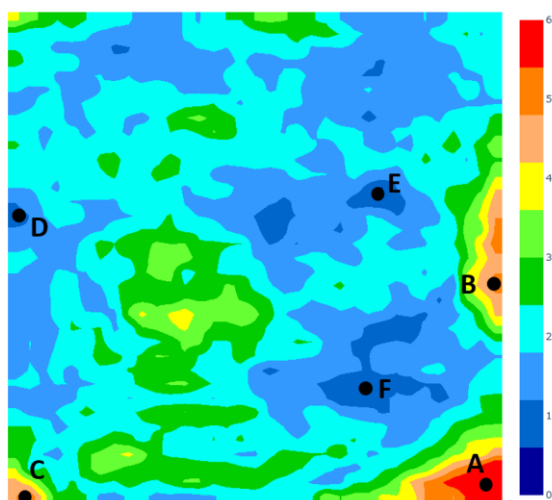


Figure 28. Distribution of percentage of validity of generated SMILES for each node.

As can be seen on Figure 28, there are 6 zones A, B and C (high validity zones) and D, E, F (low validity zones). A quick analysis of the molecules in these zones show a trend and indicate why this phenomenon exists.

D, E and F compounds show a recurrent pattern. Molecules generated from low validity zones tend to have fused, sometimes aromatic, ring structures. A simple ring; like in molecules A, B, C; is not intrinsically difficult for the AE to reproduce since the grammar of an isolated ring is fairly simple to realize and it is a pattern easily recognizable for an AE. However, when the rings are fused together, the grammar of the SMILES starts to become more complex with more and bigger numbers being introduced, parentheses, etc. and the model starts struggling to generate correct compounds which are chemically sound, which in turn makes the validity rate go well below the high validity rate of molecules possessing simple, isolated rings.

Distribution of the valid SMILES rate (*i.e.*, percentage of correct chemical structures generated for each of the nodes) is shown in Figure 28, in which the color indicates the percentage of chemically valid structures generated for the corresponding GTM node. Red nodes have high validity percentage, while blue nodes have low percentage. The generative ability of the model is very unevenly distributed on the map, zones as high as 60% validity rate coexist with zones showing about 9-10% validity rate. Interestingly, we can compare this landscape to the property landscapes on the left shown in Table 12. For example, it is possible to observe that the model is performing well in some zones where the number of aromatic rings per molecule is low, which we have explained before. Low number of rings and aromatic rings seem to be playing a big factor in the capacity of the model to generate chemically feasible compounds. Interestingly, the lower left corner of the chemical space is populated with ChEMBL compounds possessing a high number of rings, however, the generated compounds do not possess the same amount of rings. The generation process seems to have bypassed the requirement for a high number of rings and generated compounds with single rings and long carbon chains, which are present just above this area in the ChEMBL latent space.

One important aspect is the ability of the model to generate molecules with properties close to the properties of the ChEMBL compounds. Comparisons shown in Table 12, show that the model can mostly recreate the topology of the ChEMBL landscape although with sometimes different scales. The landscape of number of heavy atoms is well recreated. The two main zones with smaller molecules in the ChEMBL distribution on the left also appear on the right, and the zones containing big molecules also match. These matches can be found on all other property landscapes and confirms the ability of the model to correctly predict molecules which fit the area of latent space selected for sampling.

Table 11. Examples of molecules extracted from zones of high validity (A, B, C) and low validity (D, E, F)

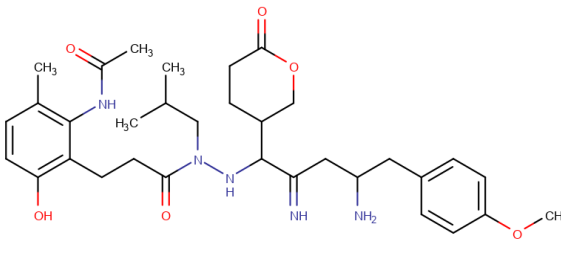
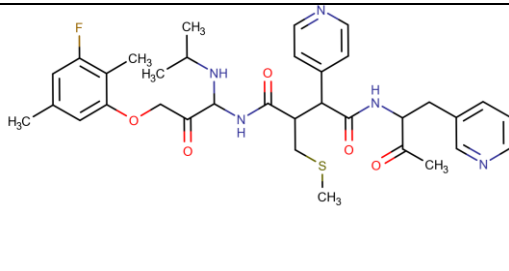
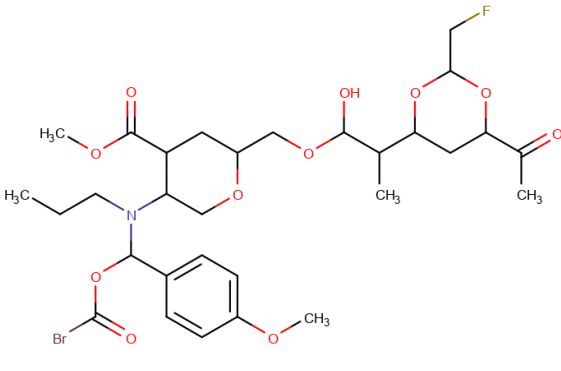
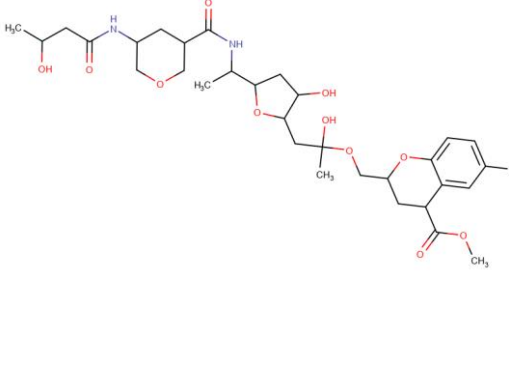
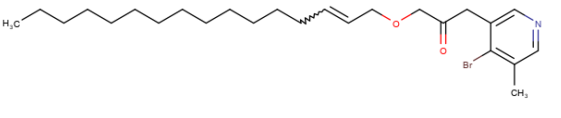
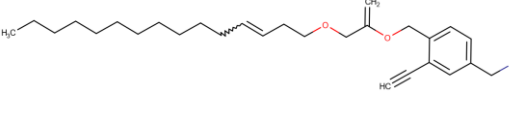
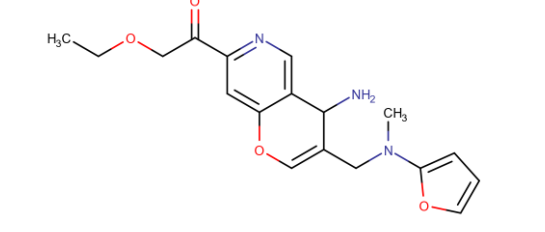
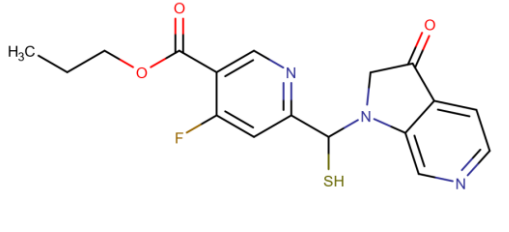
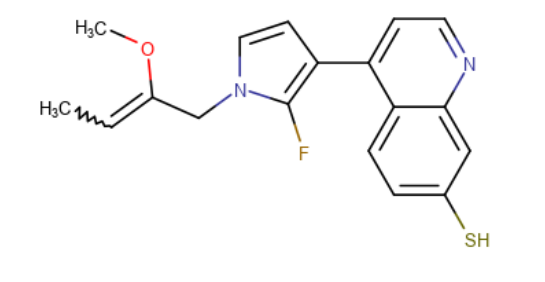
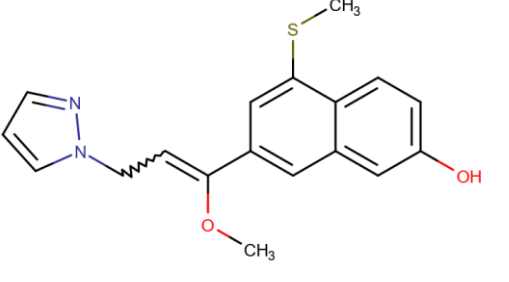
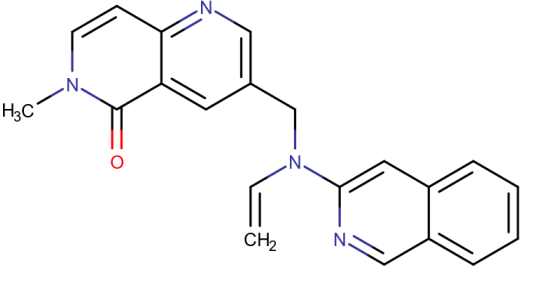
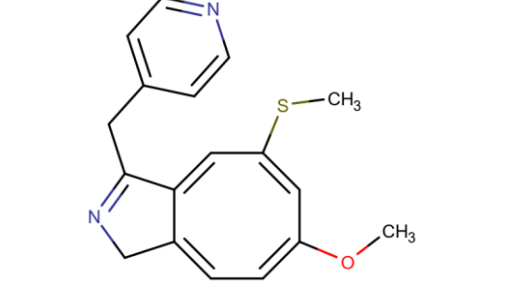
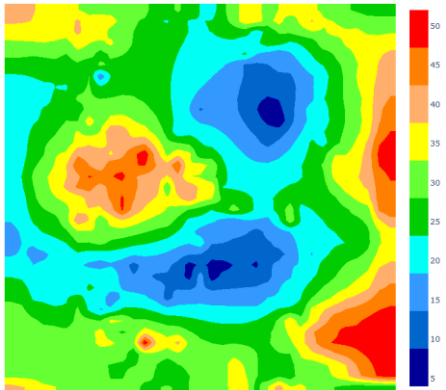
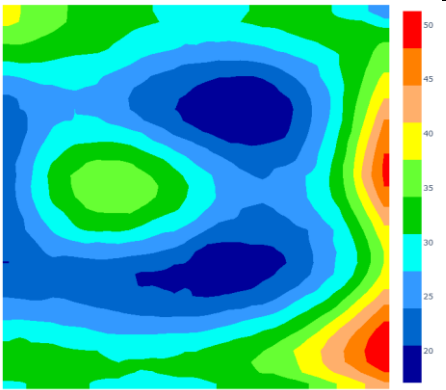
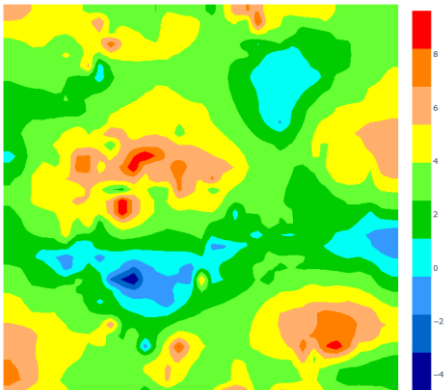
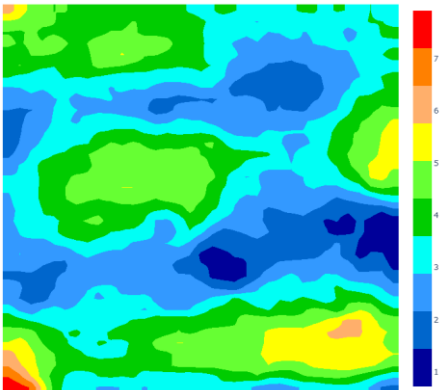
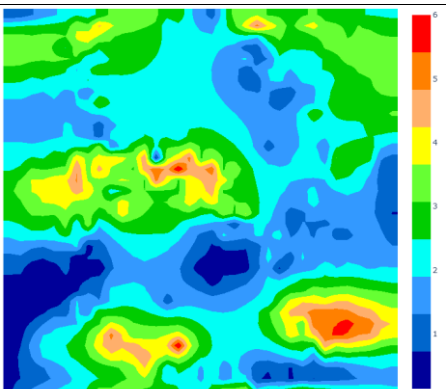
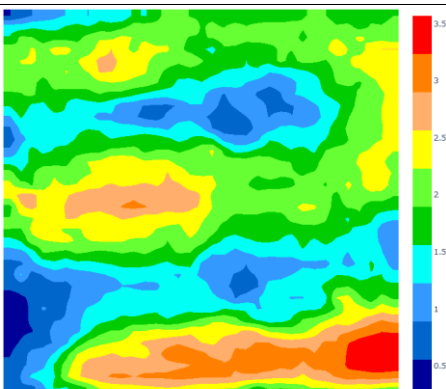
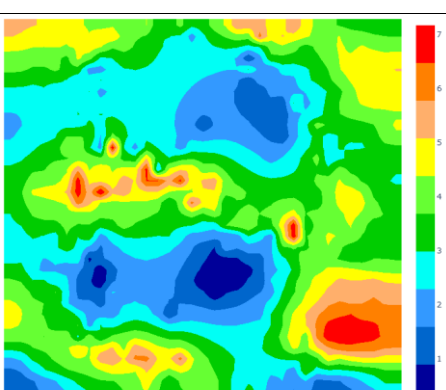
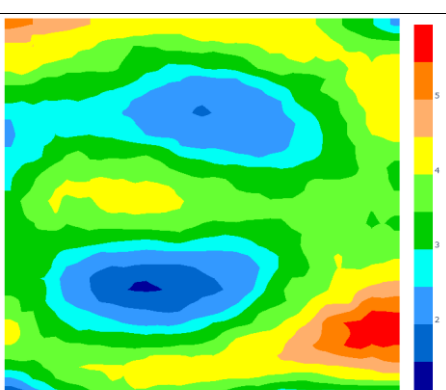
A		
B		
C		
D		
E		
F		

Table 12. Map comparison for different properties between ChEMBL and generated compounds. The maps are on different scales for better visualization.

Property	ChEMBL compounds	Generated compounds
Number of heavy atoms		
LogP		
Aromatic Rings		
Total Rings		

Neighborhood preservation for generated structures in the chemical latent space

Another important aspect of the generation process is the ability of the model to generate compounds which are located in the area of chemical space they are generated from, meaning that the positions of the initial sampled vector and the positions of the actual generated compounds are close. Figure 29 shows the density landscape of model 1 with 4 sets of generated compounds projected onto them as well as the zone the sampling was done in. These sets of generated compounds were selected from the high and low validity zones presented above. We selected two high validity zones and two low validity zones to compare the impact of a “struggle” from the model on the neighborhood preservation of generated structures. As shown in Figure 29, the generated compounds are all projected in the correct area of chemical space where they were sampled from, showing that the generation of new molecules is correctly calibrated to focus on very specific parts of chemical space.

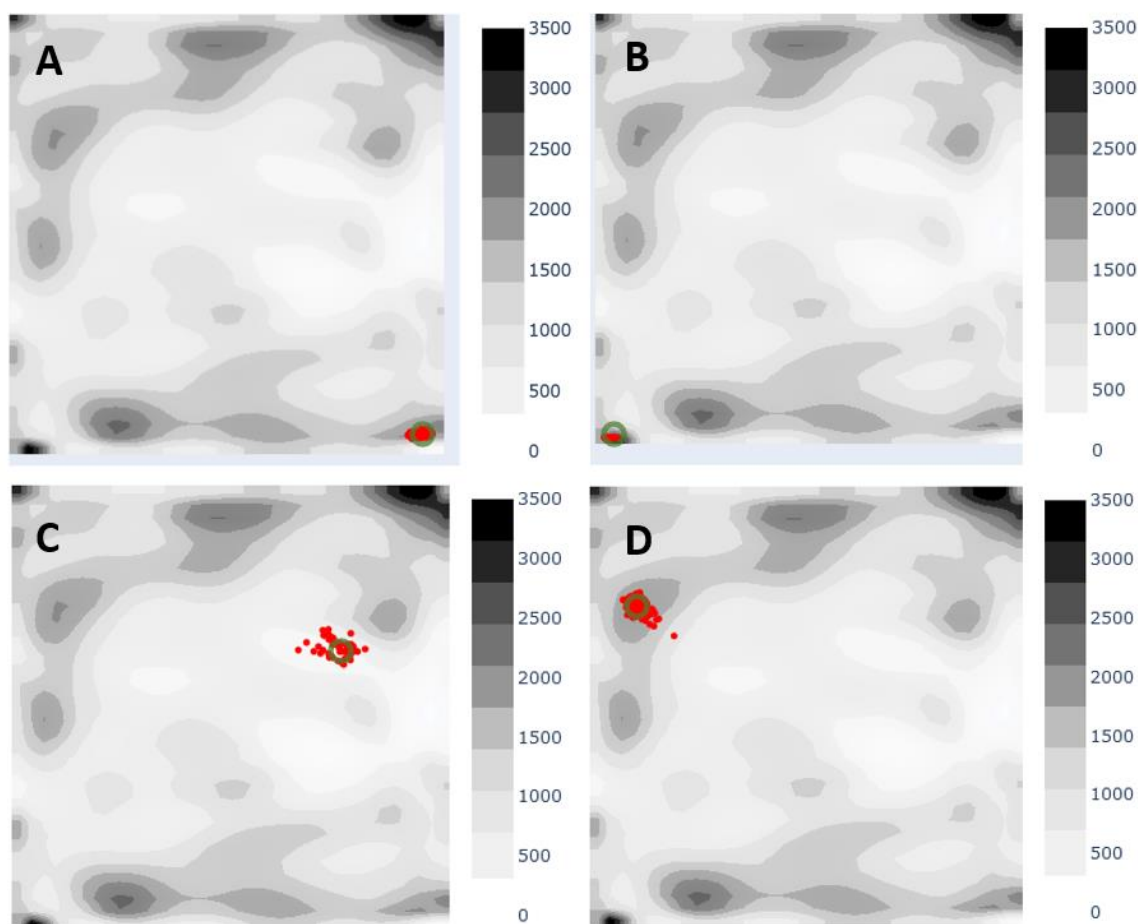


Figure 29. Projections of generated compounds back on the density landscape based on model 1. The green circles represent the area where molecules were sampled. (A) Molecules sampled from node 1189. (B) Node 70. (C) Node 923. (D) Node 115.

Distribution of the novelty rate of generated structures in chemical latent space

The comparison of generated compounds with the training database showed that all generated compounds were considered novel. No matches were found within ChEMBL. The model was able to generate completely new structures, that managed to be projected in the correct region of chemical space. It is then possible to imagine that any region of chemical space represented on GTM may be filled with novel compounds using the AE's generative ability.

3.4 Conclusion

Combining cartography and structure generation by autoencoders to explore chemical space is a promising method to facilitate the drug discovery. On the one hand, generative models can be built to create novel structures with desired properties due to the ability of GTM landscapes to reveal the most promising zones in the chemical space for generating new molecules. Visualization of the distribution of various important properties over the chemical latent space in autoencoders, their comparison with each other can provide valuable information and lead to a better understanding the performance of generative models. On the other hand, the maps of the chemical space indicate the gaps in the training-set distributions, while the trained autoencoders might be able to fill these gaps and provide us with a more complete vision of the chemical space. Developing this kind of models is critical in the quest to discover interesting, usable, novel structures because not only do we have maps detailing the current state of the universe, but we can now send “explorers” in the areas we seem interesting, either to discover “unfound land” or to search deeper in an already discovered part.

Using this approach applied to the chemical latent space of the sequence-to-sequence autoencoder trained on the ChEMBL structures, we have demonstrated in this work that the chemical structures are very evenly distributed in its latent space. New molecules can be generated by sampling in the latent space from the Gaussian distributions centered at GTM nodes and using the decoder to transform them to the SMILES strings representing chemical structures. Chemical structures generated from a given node are similar to the training structures residing in the same node unless the data density is too small. The generation process depends on several factors, like complex ring structures, aromaticity or branching which seem to play a big part in the ability of the model to generate correct structures.

4 An Autoencoder coupled with Generative Topographic Mapping for the discovery of novel reactions

The generation of potentially active compounds using an AE guided by GTM has been achieved before by Sattarov and al.^[87] using an Autoencoder with a Bidirectional LSTM-based encoder and a Unidirectional LSTM-based decoder. Encoder and Decoder were linked by a bottleneck, creating a regularized latent space (Figure 30).

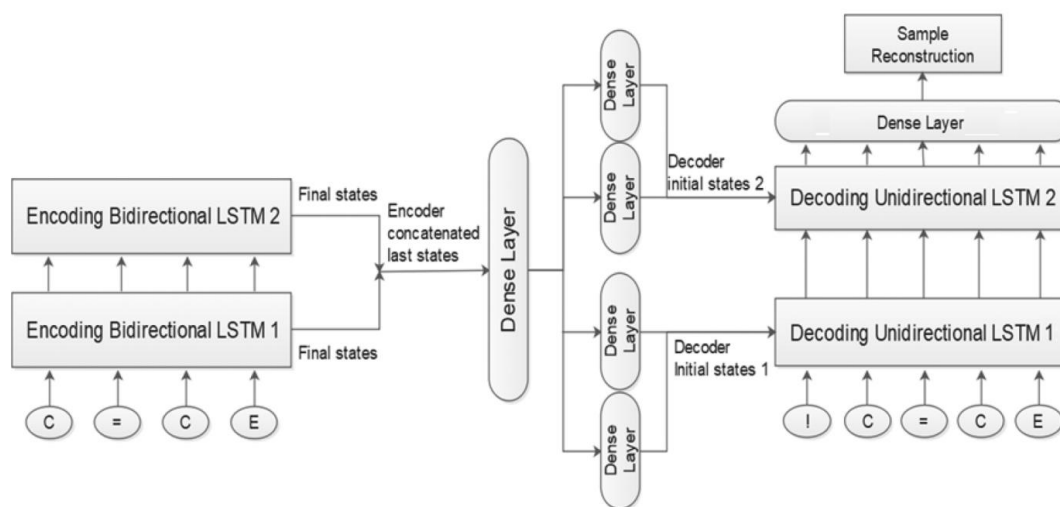


Figure 30. Schematic representation taken from the article of the Autoencoder architecture used.

The model was trained on the entire ChEMBL23 database and visualized on 2D landscapes via GTM. In particular, the adenosine a2A receptor (ChEMBL251) was selected as the target for the generative process, and the assessment using Balanced Accuracy of the related latent-based activity landscape showed good separation and predictive power, on par with classical descriptors. The model successfully managed to generate potentially active compounds from the coordinates of active clusters identified using GTM.

As stated previously, the handling of chemical reactions by seq2seq architectures is very difficult due to the complexity of chemical reaction systems involving reagents and products. However, the simplicity and good results obtained with the previous model on molecules prompted an interest in the generation of reactions using a slightly modified architecture. Reactions, just like molecules, can be expressed as character strings using reaction SMILES, which represent reagents and products separated by “.” and “>” characters (Figure 31).

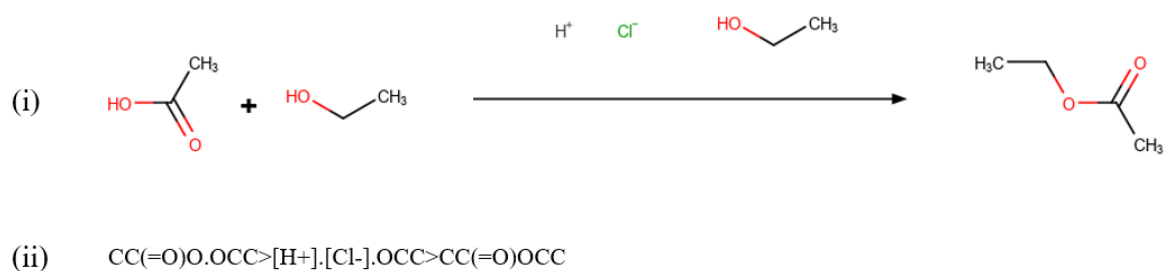


Figure 31. Chemical reaction (i) and its associated reaction SMILES (ii).

This representation can get cumbersome when several reagents, products or conditions are engaged or if the reagents and products are big structures. This causes problems with long-term dependencies and can induce errors in reconstruction or sampling. Condensed Graphs of Reactions^[9] (CGR) are a simpler and much more lightweight alternative to reaction SMILES which allow the representation of reactions in the form of pseudo-molecules which are better adapted to a usage with NN architectures.

In this work, a method of representing reactions as a pseudo-molecule called Condensed Graphs of Reactions was used in combination with a specially adapted AE architecture and GTM to map the latent space of reactions, navigate it and generate novel reactions. Reaction novelty was assessed by the newly introduced concept of Reaction Centre and Reaction Environment which consider the atoms affected by the changes in bonds as a fingerprint for the reaction type.

Several “novel” reactions, absent from the training set were isolated by the filtering process, and their feasibility was assessed by quantum calculations of reaction heat.



OPEN

Discovery of novel chemical reactions by deep generative recurrent neural network

William Bort¹, Igor I. Baskin^{1,2,4}, Timur Gimadiev³, Artem Mukanov², Ramil Nugmanov², Pavel Sidorov³, Gilles Marcou¹, Dragos Horvath¹, Olga Klimchuk¹, Timur Madzhidov² & Alexandre Varnek^{1,3}✉

The “creativity” of Artificial Intelligence (AI) in terms of generating *de novo* molecular structures opened a novel paradigm in compound design, weaknesses (stability & feasibility issues of such structures) notwithstanding. Here we show that “creative” AI may be as successfully taught to enumerate novel *chemical reactions* that are stoichiometrically coherent. Furthermore, when coupled to reaction space cartography, *de novo* reaction design may be focused on the desired reaction class. A sequence-to-sequence autoencoder with bidirectional Long Short-Term Memory layers was trained on on-purpose developed “SMILES/CGR” strings, encoding reactions of the USPTO database. The autoencoder latent space was visualized on a generative topographic map. Novel latent space points were sampled around a map area populated by Suzuki reactions and decoded to corresponding reactions. These can be critically analyzed by the expert, cleaned of irrelevant functional groups and eventually experimentally attempted, herewith enlarging the synthetic purpose of popular synthetic pathways.

The discovery of new organic reactions has always been in the focus of synthetic organic chemistry. Each new reaction enriches the arsenal of synthetic tools and opens new horizons in the development and optimization of new drugs and materials. Such reactions are often given the names of their discoverers, which is the highest recognition of their contribution to organic chemistry. Most of the new reactions have been discovered by plain luck, and it has been up to the chemists to notice the discovery and apply their “chemical intuition” to study it in detail¹. The beginning of a systematic approach to the search for new reactions was laid in 1967 by Balaban, who applied the graph theory for systematical enumeration of pericyclic reactions proceeding through a 6-membered transition state². In the 1970s, these studies were significantly expanded by Hendrickson³, Arens^{4–6}, Zefirov, and Tratch^{7,8} who considered various formal schemes describing bonds redistribution for different types of pericyclic reactions. Another approach implemented in the IGOR^{1,9} and IGOR2¹⁰ programs concerned the algebraic model of constitutional chemistry developed by Dugundji and Ugi¹¹. This approach supports the hierarchical representation of organic reactions and deals explicitly with heteroatoms and charges, keeps track of rings in molecules¹⁰. Its application led to the discovery of previously unknown reactions: the thermal decomposition of α -formyl-oxy ketones^{1,9}, and the formation of a cage molecule from N-methoxycarbonyl homopyrrole and tropone¹⁰. Then, an alternative method based on the generation of the complete sets of non-isomorphic spanning subgraphs of a given graph was suggested. With the help of this approach, new carbene reaction¹² and two new elimination reactions leading to the formation of synthetically important dienes¹³ were discovered. The formal-logical approach to organic reactions⁷ implemented in the SYMBEQ¹⁴ and ARGENT^{15,16} software was used to discover substituted furans¹⁴.

Despite great expectations, no significant progress in computer-aided reaction design was achieved; approaches, algorithms, and software tools reported so far have not found any widespread popularity among organic chemists. The work with those tools required both extensive knowledge in synthetic organic chemistry and a well-developed intuition to turn abstract schemes of bonds redistribution into specific chemical reactions with particular reagents, catalysts, and experimental conditions. This explains why all reactions computationally

¹Laboratory of Chemoinformatics, UMR 7140 CNRS, University of Strasbourg, 1, rue Blaise Pascal, 67000 Strasbourg, France. ²Laboratory of Chemoinformatics and Molecular Modeling, Butlerov Institute of Chemistry, Kazan Federal University, Kremlyovskaya str. 18, 420008 Kazan, Russia. ³Institute for Chemical Reaction Design and Discovery (WPI-ICReDD), Hokkaido University, Kita 21 Nishi 10, Kita-ku, Sapporo 001-0021, Japan. ⁴Department of Materials Science and Engineering, Technion – Israel Institute of Technology, 3200003 Haifa, Israel. ✉email: varnek@unistra.fr

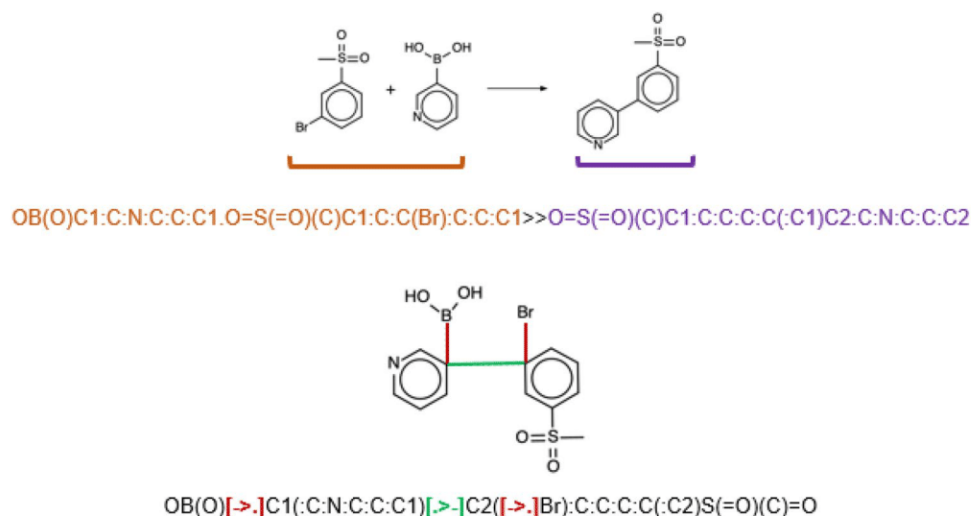


Figure 1. An example of Suzuki coupling reaction (*top*) and its condensed graph (CGR, *bottom*). Reaction SMILES and SMILES/CGR are given underneath. The reaction SMILES features reactants (in orange), and products (in purple). Atom-to-atom mapping is not provided. In the SMILES/CGR broken single bonds are encoded as [\rightarrow .] (in red), while the created C–C bond is [\rightarrow .] (in green). The colon (:) represents aromatic bonds. See Supporting Information for the details.

discovered so far were relatively simple (mainly thermal pericyclic reactions). We believe that real progress in the discovery of new chemical reactions can be achieved by deep learning from big data¹⁷. Recently, Segler et al. reported a chemical synthesis planning system based on deep neural networks and symbolic AI trained on a big collection of known synthetic reactions¹⁸. This tool, however, implements automatic extraction of transformation rules (“templates”) from known chemical reactions and therefore, in principle, cannot “suggest” not yet seen transformations. Several template-free techniques based on recurrent neural networks and transformers were successfully implemented. They operate in sequence-to-sequence translation mode¹⁹, in which SMILES of products were directly predicted from SMILES of reactants^{20,21} and vice versa^{22–24}. Interesting chemistry knowledge driven approaches aiming to predict organic reactions outcomes from given reactants were proposed by Coley et al.²⁵ and in Baldi’s group^{26,27}. Although, discovery of new chemical transformations cannot be excluded, this is not an objective of such type of calculations. To our knowledge, no new types of chemical reactions resulted from the “reactants-to-products” models were reported in the literature so far.

Generative models based on recurrent deep neural networks were successfully used to generate novel chemical structures^{28–37}. Recently, we have demonstrated that the structures of molecules possessing desirable properties could be generated using a combination of autoencoder with Generative Topographic Map built on the latent vectors²⁶. In order to apply this approach to chemical reactions, they must be encoded by SMILES strings. However, conventional reaction SMILES can hardly be used because: (i) they are much longer, and (ii) atom-to-atom mapping (AAM) needed for reaction center identification, adds a further layer of complexity. The autoencoder would have to learn not only semantics and syntax of SMILES but also the AAM rules.

Earlier, we showed that *in silico* chemical reaction handling can be significantly simplified by the Condensed Graph of Reaction (CGR) approach³⁸, in which the structures of reactants and products are merged into a single graph (Fig. 1). The CGR edges correspond either to standard chemical bonds or to “dynamic” bonds describing transformations. In such a way, one can consider a CGR as a pseudomolecule for which some types of molecular descriptors can easily be computed followed by their application in data analysis and statistical modeling tasks³⁹. Thus, this approach was successfully applied to similarity searching in reaction databases^{38,40}, building quantitative structure–reactivity models^{41–44}, assessment of tautomer distributions^{45,46}, prediction of activity cliffs⁴⁷, classification of enzymatic transformations⁴⁸, prediction of reaction conditions^{49,50}, etc. Here, for the first time, we introduce dedicated SMILES strings encoding CGRs (SMILES/CGR), see their detailed description in Supporting Information. Moreover, the CGR (and, hence, SMILES/CGR) contains information about the reaction center and its close neighborhood⁵¹.

Basically, CGRs are nothing but “molecules” with “exotic” bond orders for the changing bonds—thus, let us “teach” *de novo* molecular design tools on how to generate new reactions! Following a workflow recently used for the generation of novel molecular structures potentially possessing desirable biological activity³⁰, we have chosen to focus here on the generation of “Suzuki-like” putative chemical transformations. The Suzuki coupling reaction was chosen because this reaction is widely used in organic synthesis, and, therefore, its new variants implying different leaving groups and reaction centers could be of interest for synthetic chemists. From the technical point of view, Suzuki reactions constitute a sizeable part of the USPTO database which assures satisfactory knowledge extraction upon the model training. Reaction center of Suzuki reaction can be represented by a SMILES string BC.QL>>B.CQ.L (where Q = C, N, O, S, Si and L is a leaving group). In our simulation we expect that AI may suggest realistic and unseen combinations of Q and L.

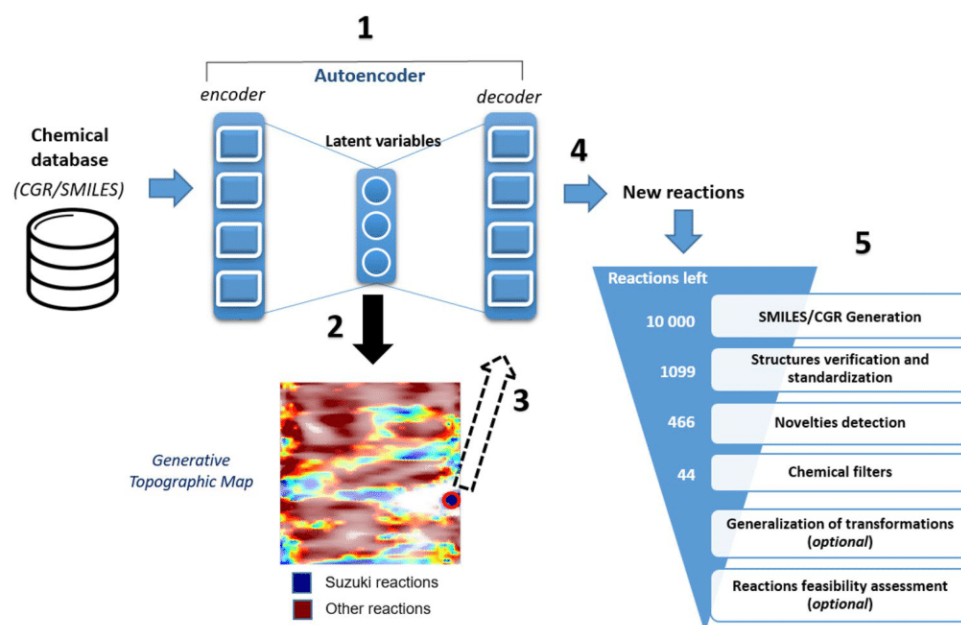


Figure 2. Modeling workflow for generation of new reactions consists of five main steps: (1) training sequence-to-sequence autoencoder on the USPTO database of chemical reactions; (2) building of Generative Topographic Map (GTM) using the autoencoder latent variables and preparation of GTM class landscape; (3) selecting on GTM a zone populated to Suzuki coupling reactions and identification of related autoencoder latent vectors; (4) sampling from the autoencoder latent space and generation of new reactions; and, (5) post-processing step. On the Generative Topographic Map, larger transparency levels correspond to lower density. The color code renders the (binary: Suzuki vs Other) reaction class distribution. Thus, zones in dark blue are exclusively populated by Suzuki reactions, zones in dark red are exclusively populated by other types of reactions; while intermediate colors correspond to reaction space areas hosting both categories, in various ratios. The red circle indicates the zone from which virtual Suzuki reactions were sampled.

A sequence-to-sequence neural network with Bidirectional Long Short-Term Memory⁵² layers trained on SMILES/CGR achieved the ability to convert SMILES/CGR to their latent vectors (“encode”) and back (“decode”). Generative Topographic Mapping (GTM) was used to visualize the latent space in 2D and to detect a cluster mostly populated with Suzuki reactions (Fig. 2). Then, virtual chemical reactions were generated by sampling the targeted zone followed by the decoding of associated latent vectors to SMILES/CGR. Notice that visualization is not strictly required for clusters identification, but may significantly help to choose a cluster from which the sampling is performed.

Results

Reaction sampling from generative topographic map. A set of 2 424 306 reactions, extracted and curated from the USPTO database⁵³, was rendered as CGRs and then as SMILES/CGR strings used to feed the autoencoder. The latter was trained on some 2 million reactions and validated on 450 thousand reactions. The reconstruction rate (a ratio of correctly reconstructed SMILES/CGR) was 98.4% and 97.8% at the training and validation stage, respectively. This is slightly less than reconstruction rates of plain molecular SMILES by state-of-the-art encoders/decoders, but it can be explained by larger complexity and length of SMILES/CGR and an additional source of error: the errors of atom-to-atom mapping in some entries. SMILES/CGR is intrinsically more difficult to learn, with dynamical bonds, dynamical atoms and formal coordination numbers exceeding atomic valency representing novel degrees of freedom in the syntax. Unbalanced or erroneous entries may pass the standardization protocols and thus negatively impact generated SMILES/CGR quality. Nevertheless, reconstruction rates are robust and although LSTM has relatively short memory compared to some other neural networks architectures like transformers and, therefore, may fail to learn relatively complex structural motifs, the bidirectional LSTM used in our work seems to perform acceptably well.

The latent vectors for 100 000 randomly selected reactions were used to construct a Generative Topographic Map (GTM) using in-house software⁵⁴. Then the entire USPTO database was projected onto the map, on which several zones predominantly populated by Suzuki reactions were identified, as shown in Fig. 2.

Random latent vectors were sampled from one of these zones with the highest relative population of Suzuki reactions. As expected, the sampling procedure led to virtual transformations of a similar type. Finally, 10,000 text strings have been generated, followed by their analysis using a complex post-processing protocol (Fig. 2). At the structures verification and standardization stage, the CGRtools.v3 tool was used to discard invalid SMILES/CGR and to perform valence and aromaticity check. This reduced the dataset to 1099 reactions (some 11% of generated text strings) in which structures of reactants and products were correct. This value is similar to that

(15–20%) observed for the SMILES strings in our previous studies devoted to generation of individual molecules. Clearly, not every latent space vector corresponds to a valid structure. However, since invalid SMILES/CGR can be discarded algorithmically, they are not a liability but a manageable consequence of exploratory sampling.

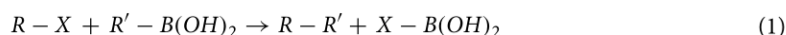
Also, the USPTO reactions are unevenly distributed in terms of types. Deep learning typically focuses on dense clusters allowing it to extensively capture their associated syntactic rules. The reliability of different combinations of leaving group **L** and a coupling partner **Q** in the reaction center BC, $QL > B.CQ.L$ suggested by AI depends not only on the training set size but also on its diversity, i.e., on the presence in the training set related examples. The USPTO dataset contains very few reactions with **Q** = O, S and Si which may explain the relatively high rate of invalid SMILES/CGR strings.

Reaction novelty analysis. The main interest of in silico reaction generation is the proposal of novel reactions that a human mind would not spontaneously think of. However, unlike individual compounds, where novelties can be identified as unique scaffolds or particular structural motifs³⁰, the definition of reaction novelty was not discussed in the literature. The most descriptive part is the reaction center (**RC**)⁵⁵, i.e. atoms and bonds directly involved in the transformation. Thus, we consider two levels of reaction novelty: (i) the reaction center is unknown (not present in the training set); (ii) reaction center is known, but its closest neighborhood (1st atoms and bonds near the RC, **RC + 1**) is new. The latter can be extended to a more distant neighborhood (*n* atoms and bonds away, **RC + n**), but in this work, we only focus on the reaction center and the closest neighbors. To decide whether a reaction is novel, these substructural reaction motifs are encoded by a hashing function as reaction signatures and are compared to all signatures extracted from the initial dataset.

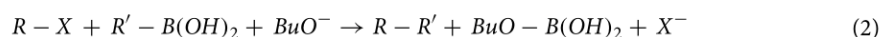
Among 1099 reactions selected using the post-processing workflow (Fig. 2), 436 contain new reaction center **RC** and 30 reactions are novel at first neighborhood level **RC + 1**. Some generated reactions have two or more distinct reaction centers, i.e. represent multistep transformations. Note that “novelty” defined as the absence of reaction center from the training set data is per se meaningful, as an illustration of the “creativity” of this Artificial Intelligence, i.e. its ability to generate original reaction centers which can be submitted for empirical feasibility assessment to human experts. Unfortunately, “novelty” as the absence of reaction center from both the training set and public reaction databases is not easy to interpret, for it may both mean that (a) such reactions were tried, but failed and thus were not published or (b) reactions were never explored, thus represent a real asset of innovation. The choice not to publish failed reactions is a major drawback in training reactivity models⁴⁴.

Reactions curation and generalisation. A close look at the generated reactions reveals several serious drawbacks: (i) unbalanced reaction equations, (ii) presence of likely unstable groups (e.g., $R_3S(=O)H$ and $R-PH(=O)-OR'$), and, (iii) transformations which require harsh reaction conditions (e.g., breaking of a C–C bond), or kinetically unfavorable reactions (e.g., cleavage of a leaving group with carbon at attachment point). Some reactions can be corrected or discarded using some heuristic rules (“Chemical Filters”).

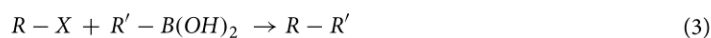
Output of unbalanced reactions is a direct consequence of the training set composition: almost all USPTO reactions are also unbalanced, e.g., leaving groups are almost never reflected in the reaction equation present in the database. The application of the CGR technology may implicitly solve that problem. Indeed, within the CGR formalism, heavy atoms in reactants and products are implicitly conserved—as the same graph is simply interpreted differently in terms of dynamical bond status in order to convert it to reagents or to products, respectively. Even if the initial reaction was not stoichiometrically balanced (see example in SI), its CGR representation will be—in so far the conversion of an unbalanced transformation to CGR succeeds to produce the correct CGR of the balanced process. However, as the exact state of the leaving group cannot be deduced from the training set, in silico generated CGRs may occasionally decode into reactions by simply substituting a broken bond by a hydrogen atom, leading to a disbalance in terms of implicit hydrogens. This is seen in the example from Fig. 3A in which the products contain 2 hydrogens more than reactants. Furthermore, the postulated product $BH(OH)_2$ is highly reactive, thus unrealistic. Formally, this is a rather “creative” in silico interpretation of the Suzuki reaction pattern, in which the organic halide $R-X$ is replaced by an amide group: the acyl fragment is assimilated to “R” while the benzylamine is the leaving group X. Formally, a balanced Suzuki process could be formulated as either



or, more realistically, with inclusion of the required alkoxy base, typically BuO^- :



Unfortunately, a sketchily written Suzuki transformation, carelessly ignoring the inorganic leaving groups:



converts to a CGR corresponding to

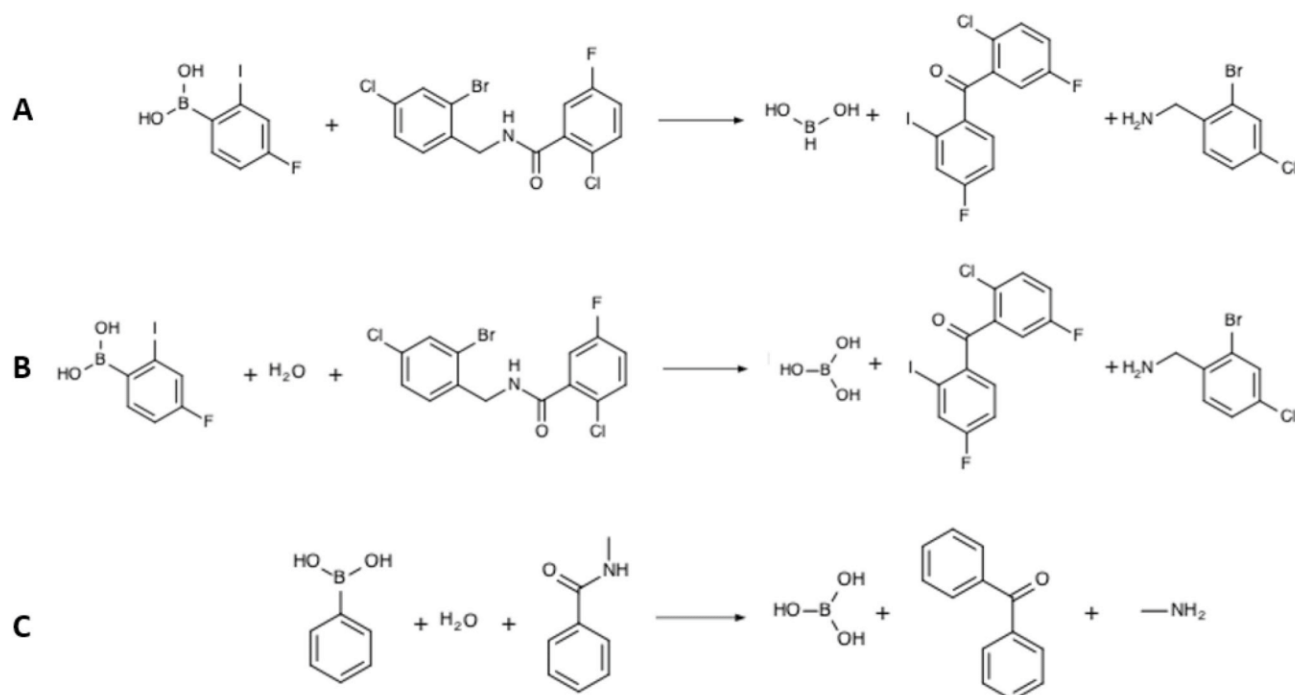
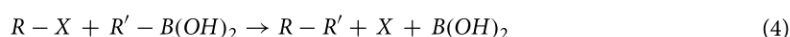


Figure 3. Example of generated chemical reaction with a new reaction center as is (A), balanced by the addition of a water molecule as a reactant (B), and its simplified form (C). Notice that the aminobenzylic leaving group suggested by the autoencoder for generated reaction looks unrealistic.

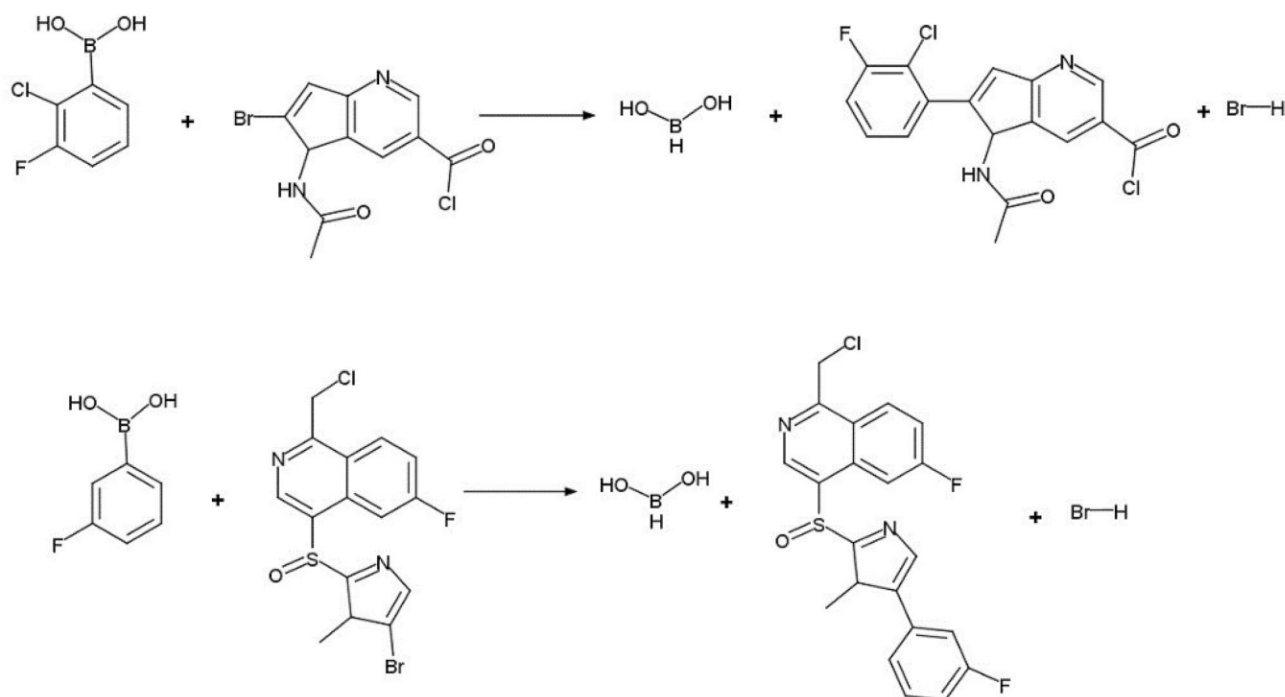


in which the unsatisfied valences of X and B are interpreted by cheminformatics tools as implicit hydrogens. This explains why the AI tool is inclined to generate formal reactions of type (4), which are nothing but a biased interpretation of Suzuki processes, corrupted by intrinsic representation errors in USPTO database entries. The addition of a water molecule as a “formal” basic species leads to a fully balanced reaction (Fig. 3B). Although water is not a perfect base for the Suzuki reaction, it helps to correctly represent the boron-containing leaving group in reaction equations. In silico generated reactions that cannot be “corrected” in this way have been discarded.

We also decided to discard the unfeasible under normal conditions transformations consisting in the cleavage of a C–C bond and assuming a carbon-centric leaving group. Application of these heuristics reduced a considered set of novel reactions to 44 including 31 reactions with new **RC** and 13 reactions with new **RC + I**.

The question arises whether we need to consider explicit chemical structures of generated reactants and products. In our opinion, this is not firmly required if one focuses on the detection of new reaction transformations identified by **RC** or **RC + I** structural motifs. In this case, a “simplified” reaction in which substrates contain only atoms of reaction center and their closest environment (including second neighbors) could be sufficient, see Fig. 3C. Notice that such simplified reactions correspond to general reactivity patterns. Particular reactants can be selected by chemists as a function of availability, intended conditions, reactivity concerns, etc. For example, the reaction in Fig. 3C looks unfeasible, but it becomes more realistic if the amine leaving group were strengthened by binding to strong electron acceptors (for example, trifluoromethanesulfonyl) or by quaternization.

Notice that the majority of generated reactions have known **RC** and **RC + I** motifs. All belong to the Suzuki coupling type, as exemplified below.



Reactions with new reaction centers. 31 reactions featuring a total of 13 distinct reaction centers not seen in USPTO were generated (see Table 1 and Table S3 in Supporting Information). Substructural searching with **RC** as a query in the much larger CAS REACT database (SciFinder) resulted in retrieval of several reactions similar to those “discovered” by Artificial Intelligence. In particular, this concerns reactions with C–Br⁵⁶ bond formation and C–Si⁵⁷ coupling, and C–C coupling with N-containing⁵⁸ and F⁵⁹ leaving groups, as well as C–O coupling with organosilicon leaving groups. In total, 5 out of 13 new reaction centers discovered computationally, were found in SciFinder reactions (Table S5 in Supporting Information). Since none of them were used for the autoencoder training, these generated reactions were pure “imagination” of AI. Thus, several “novel” reactions (4 in Table 1, 5, and 7 in Table S5) correspond to a quite interesting C–N bond cleavage with amine as leaving group. A similar reaction has recently been discovered experimentally by Weires et al.⁶⁰ who shown that the formation of amides facilitated nickel-catalyzed cleavage of C–N bonds accompanied by C–C coupling (reaction 12 in Table S3). Experimental analogues of C–Si coupling reactions generated by the model (reactions 8 and 9 in Table 1, reactions 19–26 in Table S3) were found in SciFinder (reaction 9 in Table 1 and 19 in Table S5). In the experiment, bromotriarylsilane was used as template⁵⁷ (reaction 19 in Table S5) whereas our tool proposed less stable di-substituted silane bromide possibly with heteroatoms surrounding silicon (reactions 8 and 9 in Table 1). The organosilicon leaving group proposed for the C–O coupling (reaction 11 in Table 1) is very similar to that reported by Kori et al.⁶¹ Fluoro-Suzuki reaction proposed by the autoencoder (reaction 5 in Table 1) was observed experimentally in the study by Chi et al.⁶² Reaction 7 in Table 1 is not a coupling but boron substitution by bromine; it has been experimentally discovered using N-bromosuccinimide as a donor of bromine in the study by Thiebes et al.⁵⁶ (reaction 17 in Table S5). However, from the structural point of view, its reaction center looks similar to “classical” Suzuki type reactions (boron substitution by carbon or heteroatom).

Some of the reactions still look unfeasible, e.g., the O–I compound seems quite unstable (reaction 2 in Table 1). Nonetheless, such compounds are listed as commercially available (e.g. CAS Nos 3240-34-4, 1338247-47-4). Sulfur-containing compounds are generally unsuitable for Suzuki catalysts. Their generation can be explained by an excessive model’s “creativity”, which can be hardly controlled in the employed neural network architecture.

Reactions with a new environment of known reaction centers (RC + 1). Following the novelty detection procedure, 13 reactions that correspond to 3 known reaction centers but an original first environment (**RC + 1**) were detected, see Table 2 and Table S4 in SI. Two similar reactions have been found in SciFinder. Although the simplified reaction 1 in Table 2 looks unfeasible, a more suitable leaving group might render it possible. For instance, in hydrogenation conditions, a catalyst can facilitate reductive cleavage of C–O bond (in esters, carbamates, benzyl ethers, etc.) followed by a coupling (as in reaction 10 in Table S6).

The use of alkyl and acyl bromides in C–C coupling in reaction 3 (Table 2), was observed experimentally (see reaction 13 in Table S6 in SI). Reaction 2 in Table 2 looks quite feasible because synthesis of acyl iodides was reported in the literature (e.g., CAS 191340-22-4 and CAS 1332596-80-1) whereas carboniodidates can be provided by some vendors (e.g., Enamine BBV-109267542 or BBV-109267541). Notice that similar reactions with chloroformates have been also found in Reaxys⁶³.

Reaction center SMILES	Simplified reaction	References
1 <chem>O.BC.N>>O.B.CN</chem>	<p>Reaction 1: Phenylboronic acid (<chem>O.B(O)(O)c1ccccc1</chem>) reacts with N-cyanodiphenylamine (<chem>N#C#N.Oc1ccccc1N(c2ccccc2)c3ccccc13</chem>) and water (<chem>H2O</chem>) to form a pinacol boronate ester (<chem>Oc1ccccc1B(O)(O)c2ccccc2</chem>) and a nitrile (<chem>HO-C#N</chem>).</p>	^a
2 <chem>O.BC.OI>>O.BC.OI</chem>	<p>Reaction 2: Phenylboronic acid (<chem>O.B(O)(O)c1ccccc1</chem>) reacts with phenyl iodide (<chem>Ic1ccccc1</chem>) and water (<chem>H2O</chem>) to form a pinacol boronate ester (<chem>Oc1ccccc1B(O)(O)c2ccccc2</chem>) and hydrogen iodide (<chem>HI</chem>).</p>	^a
3 <chem>O.BC.CS>>O.B.CC.S</chem>	<p>Reaction 3: Phenylboronic acid (<chem>O.B(O)(O)c1ccccc1</chem>) reacts with 2-pyridylmethyl sulfide (<chem>c1ccc(cc1)Scc2ccncc2</chem>) and water (<chem>H2O</chem>) to form a pinacol boronate ester (<chem>Oc1ccccc1B(O)(O)c2ccncc2</chem>) and a thiol (<chem>SH</chem>).</p>	^a
4 <chem>O.BC.CN>>O.B.CC</chem>	<p>Reaction 4: Phenylboronic acid (<chem>O.B(O)(O)c1ccccc1</chem>) reacts with phenylamine (<chem>Nc1ccccc1</chem>) and water (<chem>H2O</chem>) to form a pinacol boronate ester (<chem>Oc1ccccc1B(O)(O)c2ccccc2</chem>) and ammonia (<chem>NH3</chem>).</p>	³⁸
Continued		

Reaction center SMILES	Simplified reaction	References
5 O.BC.CF>>OB.CC.F		59
6 O.BC.[Si]S>>OB.C[Si].S		a
7 O.BC.BrN>>BO.CBr.N		56
8 O.BC.[Si]Br>>OB.C[Si].Br		a
9 O.BC.[Si]Br>>OB.C[Si].Br		57
Continued		

Reaction center SMILES	Simplified reaction	References
10 O.BC.SBr > > OB.CS.Br		^a
11 O.BC.O[Si] > > OB.CO.[Si]		⁶¹

Table 1. Examples of simplified Suzuki-type reactions with a new reaction center together with corresponding conventional reaction SMILES notation. The right column refers to similar types of reactions found in SciFinder. A complete list of simplified reactions is given in SI. ^aReaction centers for which no information in the literature was found.

Reaction center SMILES	Simplified reaction	References
1 <chem>BC.O.CO>>O.CC.BO</chem>		64
2 <chem>BC.O.Cl>>I.CC.BO</chem>		65
3 <chem>BC.O.CBr>>Br.CC.BO</chem>		66

Table 2. Examples of simplified Suzuki-type reactions with the RC present in the training set but in a new chemical environment. The right column refers to similar types of reactions found in SciFinder. A complete list of simplified reactions is given in SI.

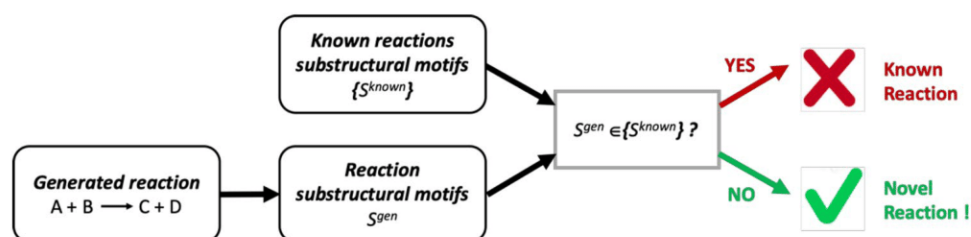


Figure 4. Reactions novelty detection workflow. Substructural motifs S^{gen} (RC , $RC + 1$, $RC + 2$, ...) are extracted from the query CGR and compared with those for known reactions $\{S^{known}\}$. In such a way, motifs belonging to novel reactions will easily be identified.

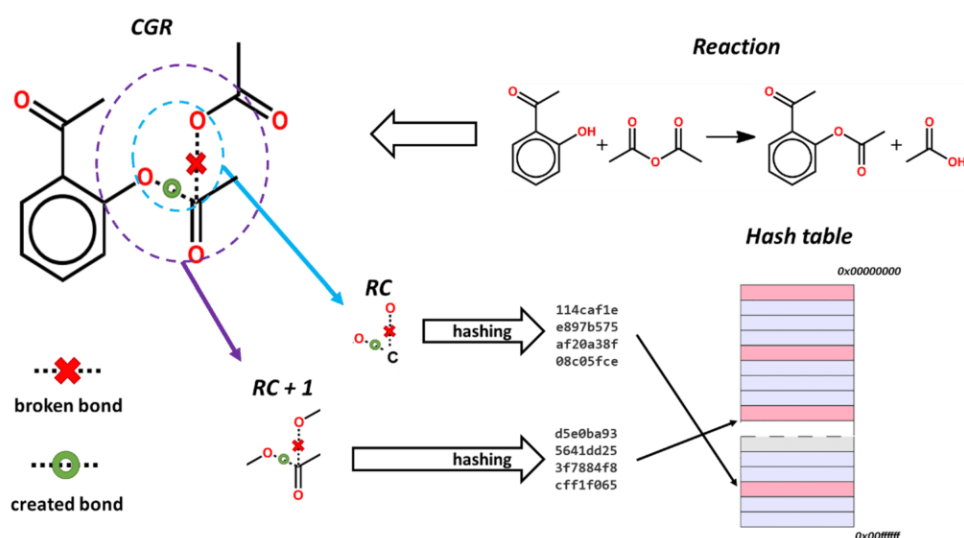


Figure 5. Preparation of a collection of reaction signatures as hash codes. From a CGR generated from a given reaction, substructural motifs containing reaction center (RC), or reaction center with n neighboring bonds and atoms ($RC + n$, here $n = 1$) can be extracted. Each motif is encoded by a hashing function into a unique hash code—reaction signature. The ensemble of unique hash codes for all reactions in the database is stored in the hash table.

Reactions feasibility assessment. Strictly speaking, reaction feasibility is defined by both kinetic and thermodynamic factors. However, according to the Bell-Evans-Polanyi principle^{66,67}, in a series of similar reactions, the trend of activation energies follows the trend of reaction enthalpies. Thus, favorable thermodynamics, namely reaction enthalpy (ΔH), can be considered as weak proof of reaction feasibility. A series of gas-phase DFT calculations were performed to assess ΔH for all simplified reactions with new **RC** and **RC + 1**. According to our estimations, almost all reactions are exothermic except for four reactions with Si-containing substrates in which ΔH is positive but close to zero (see Tables S3 and S4 in Supporting Information). This shows that all new computer-generated reactions are feasible, at least, as far as DFT-based thermodynamics estimates can tell. Since DFT is a rather time-consuming method and can hardly be applied for thousands of generated reactions, we also performed a rough estimation of ΔH using the tabulated bond energies in reactants and products^{68–70}. Although calculated in such a way reaction enthalpies poorly correlate with the DFT values, they generally provide similar conclusions concerning reaction feasibility (see Tables S3 and S4 in Supporting Information).

Conclusion

Here we present the first attempt to generate new chemical reactions using a combination of Condensed Graph of Reaction, Generative Topographic Mapping, and sequence-to-sequence autoencoder. To feed the autoencoder, special reaction SMILES strings (SMILES/CGR) were conceived and implemented. In order to discard the seemingly unfeasible reactions, a special 4-steps post-processing procedure has been implemented. It includes: (i) stoichiometric balancing of reaction equations, (ii) reduction of substrates structure to their simplified form, (iii) discarding chemically infeasible transformations using suggested heuristics (“Chemical filters”), and (iv) assessment of synthetic feasibility using quantum mechanics calculations. The effectiveness of the suggested approach was demonstrated on the example of Suzuki-like coupling reactions. Among generated reactions we discovered transformations with 13 new reaction centers which did not occur in the training set. Five out of 13

transformations were then found in the reaction databases (not used in the model training), thus showing the reliability of our approach to generate new synthetically feasible reactions.

This study reveals that creativity of Artificial Intelligence is rather limited. Deep learning neural networks, at least, in their current state, are not able to invent completely new type of chemical transformations but rather propose unseen and sometimes not trivial variations of existing ones. Thus, in this study novel (in the context of the training set) C–N, C–O, C–S and C–Si bond formation reactions, as well as nitrogen- and sulfur-containing leaving groups have been suggested by the model. We believe that this opens a way to propose putatively new synthetic pathways in a way that is not affected by the bias of human expertise—with all the benefits and the pitfalls this may bring. It should also be noted that compared to theory-driven quantum chemical models, data-driven DNN is much less time consuming and, practically, is not limited by the reactants size. The more data are used in the neural network training, the more realistic the predicted reactions are. Since the sizes of reaction databases are rapidly growing up, deep learning approach has an obvious perspective as a tool for discovery of novel reactions.

Methods

Datasets and data curation. The dataset used in this project comes from United States Patents and Trademark Office database (1976 to 2016) extracted by Lowe⁵³. It contains about 3.5 million reactions. The initial dataset was preprocessed with *in-house* scripts based on the CGRtools library⁵¹. The curation includes the standardization (aromatization and functional group standardization), removal of empty reactions (those where the products and reactants are the same, or no reactants or products are recorded) and reactions with valence errors. For curated reactions, atom-to-atom mapping (AAM) was performed using the ChemAxon Automapper tool which is a part of the JChem toolkit⁷¹. The mapped reactions were converted into CGRs and their reaction centers were extracted with the CGRtools. In total, 165 879 different reaction centers were obtained. Since AAM errors lead to incorrect reaction centers, which are usually rare, only highly populated reaction centers were selected. Thus, the resulting dataset consisted of some 2.5 million reactions (approximately 70% of the initial dataset) which corresponds to 300 most frequent reaction centers.

According to our estimations⁷², the ChemAxon Automapper tool leads to the erroneous AAM for some 25% of USPTO reactions. Most of those concern cycloadditions with complex reaction centers. As far as Suzuki errors are concerned, this error is around 3%.

Notice that practically all USPTO reactions are stoichiometrically unbalanced. This doesn't prevent to build Condensed Graph of Reaction, but, in some cases, may lead to erroneous atom-to-atom mapping.

Reaction data treatment. CGRtools library (version 3)⁵¹ was used for the reactions cleaning, their transformation to CGRs, conversion of CGRs into SMILES/CGR, and processing of generated SMILES/CGR back into reactions.

SMILES/CGR notation. Generally, SMILES/CGR follows the OpenSMILES rules⁷³. They differ from regular Daylight SMILES in terms of aromatic atoms and ring closure specification and introduce special “dynamic” bonds and atoms characterizing chemical transformations. Dynamic bonds in CGR characterizing chemical transformations have special labels representing changes in bond orders. Dynamic atom corresponds to change of formal charge or radical state of this atom in reaction. Detailed information about SMILES/CGR syntax is given in Supporting Information. SMILES/CGR generation and parsing, including preparation of canonic SMILES/CGR, are implemented into CGRtools Python library⁵¹.

Reaction generation algorithm. The network architecture previously applied for molecular SMILES generation³⁰ has been used in this study. It is based on the autoencoder architecture introduced by Xu et al.⁷⁴. SMILES/CGR transformed into sequences of one-hot encoded characters with padding to constant length (256) were used to feed the encoder. Symbols within square brackets (conventional or dynamic atoms or dynamic bonds) were considered as a single symbol within tokenization. The encoder consists of two bidirectional Long Short-Term Memory (LSTM)⁵² layers (128 nodes each), while the decoder is composed of two forward LSTM layers (256 nodes each). The bottleneck dense layer between the encoder and the decoder transforms the states of the encoder LSTMs into latent variables to subsequently feed them to the decoder; it consists of 128 nodes. Finally, the decoder outputs are transformed back to one-hot encoded characters via a single dense layer.

The autoencoder was trained in batch mode, where batches of “one-hot”-encoded sequences were generated on-the-fly from training set SMILES/ CGR strings. The Adam optimizer was used for training, initial learning rate was set to 0.005, and batch size was set to 256 samples per batch. The learning rate was reduced during training if there were no improvement in the validation loss for two epochs. The training was terminated after 34 epochs when no improvements in test set reconstruction accuracy was observed. To generate latent variable vectors for eventual decoding, we use the Generative Topographic Mapping method. It is a non-linear dimensionality reduction method that has been successfully used for chemical space analysis^{54,74–82}, comparison of chemical libraries⁸³, building classification^{43,74–77,80,84}, and regression^{85,86} models via activity landscapes, as well as for solving the “inverse” QSAR problem⁸⁷. The GTM algorithm operates by embedding a nonlinear two-dimensional manifold into a D-dimensional descriptor space and calculating the distribution of objects of initial space on these two dimensions. In this work, we utilize the autoencoder's latent vectors as an initial descriptor space. Once a map for the entire USPTO database was constructed, the zones corresponding to the desired reaction type (Suzuki reaction) were located, from which the latent vectors for virtual reactions were sampled. These new vectors fed the trained decoder resulting in new SMILES/CGR strings.

Novelty detection. Novelty detection is based on the comparison of hashed reaction signatures corresponding to reaction centers (RC) and their environment between the database of known reaction (here, USPTO database) and the reactions generated by the autoencoder (Fig. 4). Encoding chemical reactions by CGR significantly simplifies RC detection. Thus, substructural motifs involving the reaction center (RC, RC + 1, RC + 2, ...) can easily be extracted from CGR (see Fig. 5). Since any operations with molecular graphs are time-consuming, each substructural motif was encoded by a unique hash code⁵¹—a reaction signature uniquely identifying given transformation. In this case, the novelty detection is reduced to the comparison of signature (hash code) of a generated reaction with those of known reactions (Fig. 4). The suggested procedure assures fast and precise novelty detection.

Reaction enthalpy calculations. The difference in energies between reactants and products is calculated in several steps⁸⁸. First, a conformer with the lowest energy is generated for each compound in the reaction using the ChemAxon cxcalc module. Then, the geometry of each compound was optimized using the Priroda16 program with PBE exchange and correlation functional⁸⁹, and the built-in triple-zeta split valence basis set 3z, which is equivalent to Schäfer's TZVP basis set⁹⁰. Relativistic and solvent effects were neglected. The Priroda16 program was chosen as it is one of the fastest DFT software due to the efficient evaluation of density functional exchange–correlation terms based on the electron density expansion⁹¹. Final energy values were extracted for optimized structures and used for calculation of reaction enthalpy. The additive scheme for estimating reaction enthalpies was implemented using the tabulated chemical bonds increments^{68–70}.

Data availability

The dataset used in this project comes from the publicly available United States Patents and Trademark Office database (Lowe, <https://doi.org/10.17863/CAM.16293>). Curated USPTO dataset is available on GitHub: <https://github.com/Laboratoire-de-Chemoinformatique>. All data preprocessing procedures are described in the Methods section and are based on freely available CGRtools library.

Code availability

CGRtools library is used for data preprocessing and creation and treatment of chemical reactions as SMILES/CGR, and is freely available (<https://github.com/cimm-kzn/CGRtools>). Autoencoder model code and the ISIDA/GTM tool are available upon request.

Received: 8 August 2020; Accepted: 6 January 2021

Published online: 04 February 2021

References

- Herges, R. Reaction planning: Computer-aided reaction design. *Tetrahedron Comput. Methodol.* **1**, 15–25 (1988).
- Balaban, A. T. Chemical graphs. 3. Reactions with cyclic 6-membered transition states. *Rev. Roum. Chim.* **12**, 875–902 (1967).
- Hendrickson, J. B. The variety of thermal pericyclic reactions. *Angew. Chem. Int. Ed. English* **13**, 47–76 (1974).
- Arens, J. F. A formalism for the classification and design of organic reactions. I. The class of (– +)n reactions. *Recl. des Trav. Chim. des Pays-Bas* **98**, 155–161 (1979).
- Arens, J. F. A formalism for the classification and design of organic reactions. II. The classes of (+ –)n + and (– +)n – reactions. *Recl. des Trav. Chim. des Pays-Bas* **98**, 395–399 (1979).
- Arens, J. F. A formalism for the classification and design of organic reactions III. The class of (+ –)nC reactions. *Recl. des Trav. Chim. des Pays-Bas* **98**, 471–483 (1979).
- Zefirov, N. S. & Tratch, S. S. Formal-logical approach to multicentered processes with cyclic electron transfer. *Match* **3**, 263–264 (1977).
- Zefirov, N. S. S., Tratch, S. S. S. & Trach, S. S. Systematization of tautomeric processes and formal-logical approach to the search for new topological and reaction types of tautomerism. *Chem. Scr.* **15**, 4–12 (1980).
- Bauer, J., Herges, R., Fontain, E. & Ugi, I. IGOR and computer assisted innovation in chemistry. *Chimia (Aarau)*. **39**, 43–53 (1985).
- Bauer, J. IGOR2: A PC-program for generating new reactions and molecular structures. *Tetrahedron Comput. Methodol.* **2**, 269–280 (1989).
- Dugundji, J. & Ugi, I. An algebraic model of constitutional chemistry as a basis for chemical computer programs. In *Computers in Chemistry* 19–64 (Springer-Verlag, Berlin, 1973).
- Herges, R. Reaction planning: Prediction of new organic reactions. *J. Chem. Inf. Comput. Sci.* **30**, 377–383 (1990).
- Herges, R. & Hooock, C. Reaction planning: Computer-aided discovery of a novel elimination reaction. *Science* **255**, 711–713 (2020).
- Zefirov, N. S., Baskin, I. I. & Palyulin, V. A. SYMBEQ program and its application in computer-assisted reaction design. *J. Chem. Inf. Comput. Sci.* **34**, 994–999 (1994).
- Zefirov, N., Tratch, S. & Molchanova, M. The argent program system: A second-generation tool aimed at combinatorial search for new types of organic reactions. *Math. Comput. Chem.* **46**, 253–273 (2002).
- Molchanova, M. S., Tratch, S. S. & Zefirov, N. S. Computer-aided design of new organic transformations: Exposition of the ARGENT-1 program. *J. Phys. Org. Chem.* **16**, 463–474 (2003).
- Baskin, I. I., Madzhidov, T. I., Antipin, I. S. & Varnek, A. A. Artificial intelligence in synthetic chemistry: Achievements and prospects. *Russ. Chem. Rev.* **86**, 1127–1156 (2017).
- Segler, M. H. S., Preuss, M. & Waller, M. P. Planning chemical syntheses with deep neural networks and symbolic AI. *Nature* **555**, 604 (2018).
- Sutskever, I., Vinyals, O. & Le, Q. V. Sequence to sequence learning with neural networks. *Adv. Neural. Inf. Process. Syst.* **4**, 3104–3112 (2014).
- Nam, J. & Kim, J. Linking the Neural Machine Translation and the Prediction of Organic Chemistry Reactions. Preprint at *arXiv* <https://arxiv.org/abs/1612.09529> (2016).
- Schwaller, P., Gaudin, T., Lányi, D., Bekas, C. & Laino, T. “Found in Translation”: Predicting outcomes of complex organic chemistry reactions using neural sequence-to-sequence models. *Chem. Sci.* **9**, 6091–6098 (2018).
- Liu, B. *et al.* Retrosynthetic reaction prediction using neural sequence-to-sequence models. *ACS Cent. Sci.* **3**, 1103–1113 (2020).
- Karpov, P., Godin, G. & Tetko, I. V. A transformer model for retrosynthesis. *Lect. Notes Comput. Sci.* **11731**, 817–830 (2019).

24. Schwaller, P. *et al.* Predicting retrosynthetic pathways using a combined linguistic model and hyper-graph exploration strategy. (2019) doi:<https://doi.org/10.26434/chemrxiv.9992489.v1>.
25. Coley, C. W., Barzilay, R., Jaakkola, T. S., Green, W. H. & Jensen, K. F. Prediction of organic reaction outcomes using machine learning. *ACS Cent. Sci.* **3**, 434–443 (2017).
26. Fooshee, D. *et al.* Deep learning for chemical reaction prediction. *Mol. Syst. Des. Eng.* **3**, 442–452 (2018).
27. Kayala, M. A. & Baldi, P. ReactionPredictor: Prediction of complex chemical reactions at the mechanistic level using machine learning. *J. Chem. Inf. Model.* **52**, 2526–2540 (2012).
28. Xue, D. *et al.* Advances and challenges in deep generative models for de novo molecule generation. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **9**, e1395 (2019).
29. Xu, Y. *et al.* Deep learning for molecular generation. *Fut. Med. Chem.* **11**, 567–597 (2019).
30. Sattarov, B. *et al.* De novo molecular design by combining deep autoencoder recurrent neural networks with generative topographic mapping. *J. Chem. Inf. Model.* **59**, 1182–1196 (2019).
31. 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.* **4**, 828–849 (2019).
32. Blaschke, T., Olivecrona, M., Engkvist, O., Bajorath, J. & Chen, H. Application of generative autoencoder in de novo molecular design. *Mol. Inform.* **37**, 1700123 (2018).
33. Sanchez-Lengeling, B. & Aspuru-Guzik, A. Inverse molecular design using machine learning: Generative models for matter engineering. *Science* (80-) **361**, 360–365 (2018).
34. Jorgensen, P. B., Schmidt, M. N. & Winther, O. Deep generative models for molecular science. *Mol. Inform.* **37**, 1700133 (2018).
35. Gupta, A. *et al.* Generative recurrent networks for de novo drug design. *Mol. Inform.* **37**, 1700111 (2018).
36. Segler, M. H. S. & Waller, M. P. Modelling chemical reasoning to predict and invent reactions. *Chem. A Eur. J.* **23**, 6118–6128 (2017).
37. Brown, N., Fiscato, M., Segler, M. H. S. & Vaucher, A. C. GuacaMol: Benchmarking models for de novo molecular design. *J. Chem. Inf. Model.* **59**, 1096–1108 (2019).
38. Hoonakker, F., Lachiche, N., Varnek, A. & Wagner, A. A representation to apply usual data mining techniques to chemical reactions illustration on the rate constant of SN2 reactions in water. *Int. J. Artif. Intell. Tools* **20**, 253–270 (2011).
39. Varnek, A., Fourches, D., Hoonakker, F. & Solovev, V. P. Substructural fragments: An universal language to encode reactions, molecular and supramolecular structures. *J. Comput. Aided. Mol. Des.* **19**, 693–703 (2005).
40. Hoonakker, F., Lachiche, N., Varnek, A. & Wagner, A. A representation to apply usual data mining techniques to chemical reactions. *Lect. Notes Comput. Sci.* **6097**, 318–326 (2010).
41. Madzhidov, T. I. *et al.* Structure-reactivity relationships in terms of the condensed graphs of reactions. *Russ. J. Org. Chem.* **50**, 459–463 (2014).
42. Madzhidov, T. I. *et al.* Structure-reactivity relationship in bimolecular elimination reactions based on the condensed graph of a reaction. *J. Struct. Chem.* **56**, 1227–1234 (2015).
43. Gimadiev, T. *et al.* Bimolecular nucleophilic substitution reactions: Predictive models for rate constants and molecular reaction pairs analysis. *Mol. Inform.* **38**, 1800104 (2019).
44. Glavatskikh, M. *et al.* Predictive models for kinetic parameters of cycloaddition reactions. *Mol. Inform.* **38**, 1800077 (2019).
45. Gimadiev, T. R. *et al.* Assessment of tautomer distribution using the condensed reaction graph approach. *J. Comput. Aided. Mol. Des.* **32**, 401–414 (2018).
46. Gimadiev, T. R. *et al.* Prediction of tautomer equilibrium constants using condensed graphs of reaction. in *Second Kazan Summer School on Chemoinformatics* 34 (2015).
47. Horvath, D. *et al.* Prediction of activity cliffs using condensed graphs of reaction representations, descriptor recombination, support vector machine classification, and support vector regression. *J. Chem. Inf. Model.* **56**, 1631–1640 (2016).
48. Latino, D. A. R. S. & Aires-de-Sousa, J. Classification of chemical reactions and chemoinformatic processing of enzymatic transformations. *Methods Mol. Biol.* **672**, 325–340 (2011).
49. Madzhidov, T. I. *et al.* Artificial neural networks model for assessment of optimal conditions of hydrogenation reactions. in *In 22nd European Symposium on Quantitative Structure-Activity Relationships*. 186 (2018).
50. Marcou, G. *et al.* Expert system for predicting reaction conditions: The Michael reaction case. *J. Chem. Inf. Model.* **55**, 239–250 (2015).
51. Nugmanov, R. I. *et al.* CGRtools: Python library for molecule, reaction, and condensed graph of reaction processing. *J. Chem. Inf. Model.* **59**, 2516–2521 (2019).
52. Hochreiter, S. & Schmidhuber, J. Long short-term memory. *Neural Comput.* **9**, 1735–1780 (1997).
53. Lowe, D. M. M. Extraction of chemical structures and reactions from the literature. *Doctoral Thesis* (University of Cambridge, 2012). doi:<https://doi.org/10.17863/CAM.16293>.
54. Gaspar, H. A. *et al.* Generative topographic mapping approach to chemical space analysis. *ACS Symp. Ser.* **1222**, 211–241 (2016).
55. Chen, W. L., Chen, D. Z. & Taylor, K. T. Automatic reaction mapping and reaction center detection. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **3**, 560–593 (2013).
56. Thiebes, C., Thiebes, C., Prakash, G. K. S., Petasis, N. A. & Olah, G. A. Mild preparation of haloarenes by ipso-substitution of arylboronic acids with N-halosuccinimides. *Synlett* **2**, 141–142 (1998).
57. Park, J. *et al.* Indole compound, compound for organic electric element containing derivative thereof, organic electric element using same, and corresponding electronic device. PCT/KR2013/003289. (2013).
58. Zong, Y., Hu, J., Sun, P. & Jiang, X. Synthesis of biaryl derivatives via a magnetic Pd-NPs-catalyzed one-pot diazotization-cross-coupling reaction. *Synlett* **23**, 2393–2396 (2012).
59. Luo, Z.-J., Zhao, H.-Y. & Zhang, X. Highly selective Pd-catalyzed direct C–F bond arylation of polyfluoroarenes. *Org. Lett.* **20**, 2543–2546 (2018).
60. Weires, N. A., Baker, E. L. & Garg, N. K. Nickel-catalyzed Suzuki–Miyaura coupling of amides. *Nat. Chem.* **8**, 75–79 (2016).
61. Kori, M. *et al.* Fused thiadiazine derivatives as AMPA receptor potentiators and their preparation and use for the treatment of diseases. *PCT Int. Appl.* **16**, 2012020848 (2012).
62. Chi, Y. & Lin, J. Iridium complex, OLED using the same, and nitrogen-containing tridentate ligand having carbene unit. *Faming Zhuanli Shenqing* 106928281 <https://patents.google.com/patent/US10153442B2> (2017).
63. Duan, Y.-Z. & Deng, M.-Z. Palladium-catalyzed cross-coupling reaction of arylboronic acids with chloroformate or carbamoyl chloride. *Synlett* **02**, 355–357 (2005).
64. Dindarloo Inaloo, I., Majnooni, S., Eslahi, H. & Esmailpour, M. Nickel(II) Nanoparticles Immobilized on EDTA-Modified Fe₃O₄. SiO₂ Nanospheres as Efficient and Recyclable Catalysts for Ligand-Free Suzuki–Miyaura Coupling of Aryl Carbamates and Sulfamates. *ACS Omega* **5**, 7406–7417 (2020).
65. Chakraborty, J., Nath, I. & Verpoort, F. Pd-nanoparticle decorated azobenzene based colloidal porous organic polymer for visible and natural sunlight induced Mott–Schottky junction mediated instantaneous Suzuki coupling. *Chem. Eng. J.* **358**, 580–588 (2019).
66. Bell, R. P. & Hinshelwood, C. N. The theory of reactions involving proton transfers. *Proc. R. Soc. London. Ser. A Math. Phys. Sci.* **154**, 414–429 (1936).
67. Evans, M. G. & Polanyi, M. Further considerations on the thermodynamics of chemical equilibria and reaction rates. *Trans. Faraday Soc.* **32**, 1333–1360 (1936).

68. Cottrell, T. L. *The strengths of chemical bonds*. (Butterworths Scientific Publications, 1958).
69. Darwent, B. deB. *Bond dissociation energies in simple molecules*. (1970).
70. Benson, S. W. III. Bond energies. *J. Chem. Educ.* **42**, 502 (1965).
71. ChemAxon. Chemical Structure Representation Toolkit. (2019).
72. Lin, A. I. *et al.* Atom-to-Atom Mapping: A Benchmarking Study of Popular Mapping Algorithms and Consensus Strategies. <https://doi.org/10.26434/chemrxiv.13012679.v1> (2020).
73. James, C. A. OpenSMILES specification. www.opensmiles.org (2016).
74. Xu, Z., Wang, S., Zhu, F. & Huang, J. Seq2seq Fingerprint. in *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics - ACM-BCB '17* 285–294 (ACM Press, 2017). doi:<https://doi.org/10.1145/3107411.3107424>.
75. Gimadiev, T. R., Madzhidov, T. I., Marcou, G. & Varnek, A. Generative topographic mapping approach to modeling and chemical space visualization of human intestinal transporters. *Bionanoscience* **6**, 464–472 (2016).
76. Klimenko, K., Marcou, G., Horvath, D. & Varnek, A. Chemical space mapping and structure-activity analysis of the ChEMBL antiviral compound set. *J. Chem. Inf. Model.* **56**, 1438–1454 (2016).
77. Sidorov, P., Gaspar, H., Marcou, G., Varnek, A. & Horvath, D. Mappability of drug-like space: Towards a polypharmacologically competent map of drug-relevant compounds. *J. Comput. Aided. Mol. Des.* **29**, 1087–1108 (2015).
78. Maniyar, D. M., Nabney, I. T., Williams, B. S. & Sewing, A. Data visualization during the early stages of drug discovery. *J. Chem. Inf. Model.* **46**, 1806–1818 (2006).
79. Owen, J. R., Nabney, I. T., Medina-Franco, J. L. & López-Vallejo, F. Visualization of molecular fingerprints. *J. Chem. Inf. Model.* **51**, 1552–1563 (2011).
80. Kireeva, N. *et al.* Generative topographic mapping (GTM): Universal tool for data visualization, structure-activity modeling and dataset comparison. *Mol. Inform.* **31**, 301–312 (2012).
81. Glavatskikh, M. *et al.* Visualization and analysis of complex reaction data: The case of tautomeric equilibria. *Mol. Inform.* **37**, 1800056 (2018).
82. Horvath, D., Marcou, G. & Varnek, A. Generative topographic mapping approach to chemical space analysis. 167–199 (2017). doi:https://doi.org/10.1007/978-3-319-56850-8_6.
83. Gaspar, H. A., Baskin, I. I., Marcou, G., Horvath, D. & Varnek, A. Chemical data visualization and analysis with incremental generative topographic mapping: Big data challenge. *J. Chem. Inf. Model.* **55**, 84–94 (2015).
84. Gaspar, H. A. *et al.* Generative topographic mapping-based classification models and their applicability domain: Application to the biopharmaceutics drug disposition classification system (BDDCS). *J. Chem. Inf. Model.* **53**, 3318–3325 (2013).
85. Gaspar, H. A., Baskin, I. I., Marcou, G., Horvath, D. & Varnek, A. GTM-based QSAR models and their applicability domains. *Mol. Inform.* **34**, 348–356 (2015).
86. Baskin, I. I., Solovev, V. P., Bagaturyants, A. A. & Varnek, A. Predictive cartography of metal binders using generative topographic mapping. *J. Comput. Aided. Mol. Des.* **31**, 701–714 (2017).
87. Gaspar, H. A., Baskin, I. I., Marcou, G., Horvath, D. & Varnek, A. Stargate GTM: Bridging descriptor and activity spaces. *J. Chem. Inf. Model.* **55**, 2403–2410 (2015).
88. Gimadiev, T. R., Klimchuk, O., Nugmanov, R. I., Madzhidov, T. I. & Varnek, A. Sydnone-alkyne cycloaddition: Which factors are responsible for reaction rate? *J. Mol. Struct.* **1198**, 126897 (2019).
89. Perdew, J. P., Burke, K. & Ernzerhof, M. Generalized gradient approximation made simple. *Phys. Rev. Lett.* **77**, 3865–3868 (1996).
90. Schäfer, A., Huber, C. & Ahlrichs, R. Fully optimized contracted Gaussian basis sets of triple zeta valence quality for atoms Li to Kr. *J. Chem. Phys.* **100**, 5829–5835 (1994).
91. Laikov, D. N. Fast evaluation of density functional exchange-correlation terms using the expansion of the electron density in auxiliary basis sets. *Chem. Phys. Lett.* **281**, 151–156 (1997).

Acknowledgements

RN, IB, TM are grateful to Russian Science Foundation (Project No 19-73-10137) for the support of CGRtools library development.

Author contributions

All authors contributed to the code development, obtaining the results and preparation the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary Information The online version contains supplementary material available at <https://doi.org/10.1038/s41598-021-81889-y>.

Correspondence and requests for materials should be addressed to A.V.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



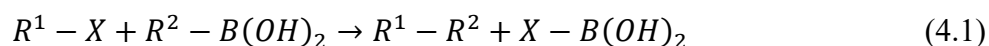
Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

© The Author(s) 2021

4.1 Summary

In this work, we combine a specially adapted LSTM-based Vanilla Autoencoder with Condensed Graphs of Reactions and Generative Topographic Mapping to create a model capable of encoding chemical reactions in a latent space.

CGR encode whole reaction systems into pseudo-molecules with SMILES-like representation, making them perfectly adapted for seq2seq architectures. A curated dataset extracted and curated from the USPTO database containing about 2.4 million reactions was encoded into CGR and given as input to a vanilla LSTM-based Autoencoder. The Autoencoder achieved a reconstruction rate of around 98% which is on par with the reconstruction rates achieved by Vanilla SMILES-based Autoencoders. Once trained, the created latent space was plotted using GTM and coloured according to reaction type using reaction centres, which classify reactions into certain categories depending on the atoms implied in the bond changes. The focus was put on Suzuki reactions, of the form:



10.000 Random latent vectors sampled from regions populated in majority by this type of reactions were given to the decoder for the generation process, out of which 1099 were found to be correct (11% validity rate). Among the 1099 correct reactions, 31 had reaction centres not seen in the training database, which indicates some kind of “creativity” from the AI. 13 of these reaction types were found in external databases or published articles, corresponding to 3 different reaction centres, showing that the model can recreate existing reactions without having them in the initial training data.

The feasibility of the 13 reactions was tested using gas-phase DFT calculations of reaction enthalpy, which showed that all the generated chemical reactions were feasible, at least as far as DFT estimations can tell.

5 Linking the latent space of an Autoencoder with another descriptor space

Ultimately, the goal for any drug design process is to be able to control the activity, structural features, properties, and novelty of the generated compounds, allowing chemists to obtain molecules perfectly fitting the needed profile for a given task. It is difficult with vanilla AEs to generate structures possessing desired properties, since there is no control over the organization of latent space. While still meaningful, the interpretation of SMILES strings by a sequential RNNs can hardly be compared to the level of information coded into structural descriptors. An autoencoder's latent space is therefore less adapted to tackle the variety of existing targets than a modulable, adaptable range of structural descriptors. However, generative models seeded by these molecular descriptors (like ISIDA) have not yet been developed.

The combination of generative autoencoder models with the robustness and versatility of chemical space built on ISIDA descriptors would allow more control over the generated structures. Easily understandable, robust, and versatile coordinates in ISIDA space, optimally chosen for the needed task could be translated into latent coordinates corresponding to areas in the space of an autoencoder which could then be sampled to generate focused datasets. By using latent vectors as a sort of “middleman”, the generation of novel compounds from structural descriptors would be possible. This inverse QSAR process could be more efficient and complement the screening of large databases with more classical methods.

In this chapter, several methods aiming to link ISIDA descriptor spaces with the latent space of an autoencoder were proposed and tested.

- 1) **ISIDA2SMI.** A simple LSTM-based model which aims to directly translate ISIDA descriptor vectors to SMILES of corresponding molecules.
- 2) **Multimodal Deep Boltzmann Machine.** A probability-based model composed of two independent reconstructive architectures linked by a “context” layer. One architecture trains to reconstruct ISIDA vector, the other reconstructs latent vectors. They are linked by a “context” layer in which the information can pass, effectively working as a translator between ISIDA and latent spaces.

- 3) **Stargate-GTM.** A GTM-based approach where two manifolds are trained together, one in latent space, one in ISIDA space, describing a mixed probability distribution. The position of a compound on one landscape results in a distribution in the second landscape via the use of a mapping function.
- 4) **Combination of ISIDA landscapes.** A combination of GTM landscapes in ISIDA space, corresponding to desirable properties, were used to build a query vector, used to find a valid position in latent space to sample, which would correspond to the initial ISIDA vector.
- 5) **Constrained Variational Autoencoder (CVAE).** A CVAE architecture was developed, using ISIDA descriptors as condition vectors. It consists in sampling the marginal probability distribution of a variational autoencoder using ISIDA vectors of compounds with desirable properties as conditions.

Insight and knowledge about latent space compatibility and the handling of structural descriptors by classical NN architecture were gathered through the exploration of these many strategies. Finally, a satisfactory architecture has been found capable of linking both chemical spaces with good results.

5.1 ISIDA2SMI

In this work, the aim is to create a link between a space of latent descriptors obtained from an AE and another “target” descriptor space. These target descriptors must therefore be carefully selected since the organization of chemical space highly depends on the type of descriptor used.

The first very basic solution proposed to the problem was to simply “translate” ISIDA descriptors into SMILES strings using a LSTM-based model (ISIDA2SMI). ISIDA descriptors used in the construction of previously published Universal Maps (UM) were favoured in this context for their versatility and good predictive power over many biological targets.

The issue with ISIDA descriptors is that they are, as well as most molecular descriptors, not unique. For example, the ChEMBL25 database encoded into UM4 descriptors (e.g., IA-2-7; sequences of 2-7 atoms, dimension 6520) contains around 10% of duplicates: several compounds resulting in the same molecular descriptor vector. A canonical-to-canonical SMILES AE learns to associate in a 1-to-1 relationship, meaning one latent vector corresponds to one SMILES string. However, the bijection is guaranteed only in the frame of the training sets: new SMILES strings may result in the same latent vectors. If one ISIDA vector has two SMILES associated to it, this ambiguity is challenging for training an autoencoder employing categorical cross entropy as the loss function.

To solve this issue, we proposed to generate several SMILES strings for a same ISIDA vector during the training stage. The model would then understand that it does not need to reconstruct one exact SMILES, but that several possibilities exist. The aim was to minimize the smallest squared Euclidian distance between the given ISIDA vector and the ISIDA vectors of the generated compounds. This resulted in a sort of multi-instance learning, to adapt a “one-to-one” generative model to work in a “one-to-many” fashion as shown Figure 32.

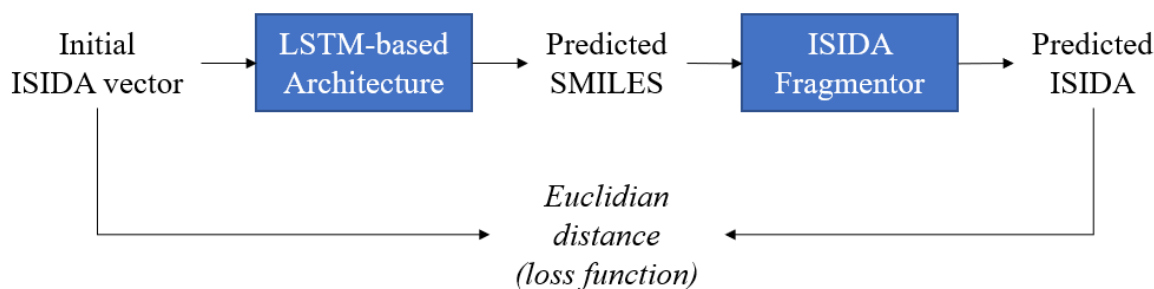


Figure 32. Initial idea for a one-to-many ISIDA to SMILES generative model. The initial ISIDA vector is passed through the model, and a corresponding SMILES is generated. Next, its ISIDA vector can be computed and compared to the initial one using the Euclidean distance as a metric. The latter can be used as a loss function which would need to be minimized.

However, problems appeared when trying to compute the ISIDA vector for the predicted SMILES during the training process. The FRAGMENTOR software had to be called repeatedly for every batch to compute the loss. Depending on the type of descriptors used, that would also necessitate colouring by ChemAxon^[158]. This resulted in a resource-demanding and computationally inefficient process.

A workaround has been proposed to solve this issue. It consists in building a “FRAGMENTOR neural network” (FRAGMENTOR-NN) model whose task would be to generate a valid ISIDA vector from a given SMILES. The trained FRAGMENTOR-NN model could then replace the actual FRAGMENTOR in the initial model, and significantly speed up the training process as shown in Figure 33. This workaround does mean that errors in the FRAGMENTOR-NN will propagate on the general loss of the ISIDA2SMI model.

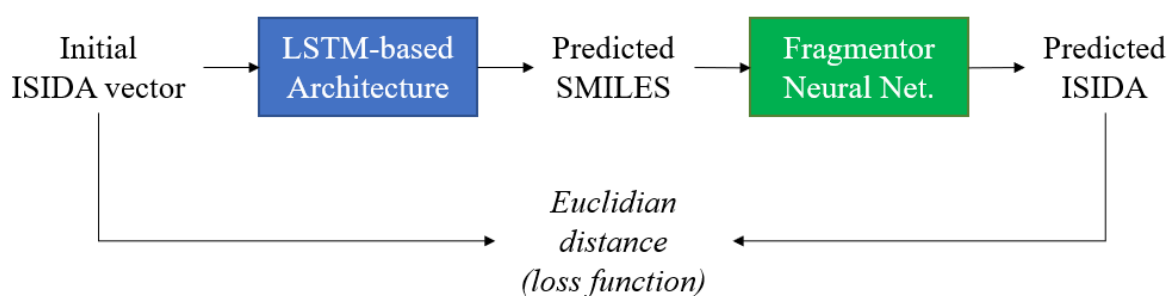


Figure 33. Updated model with the FRAGMENTOR-NN replacing the actual FRAGMENTOR.

5.1.1 Methods

Data

As input data, the ChEMBL25 database (1,669,377 molecules) was converted into ISIDA descriptors using the UM4 fragmentation scheme. IA-2-7 corresponds to sequences of 2-7 atoms and fit the most with the sequential nature of the processing of character strings by LSTM cells. They also are the most readable and most easily understandable. Training needed to be accelerated since this was a prototyping phase with a great amount of testing needed. Therefore 500,000 random compounds (about 30% of the database) were selected for the training and validation sets. The 500,000 compounds were split into training and validation sets with a ratio of 90%/10%. 166,597 additional compounds were extracted as an external test set, separate from training and validation. Table 13 summarizes the data separation.

Table 13. Summary of the data separation.

ChEMBL25	1,669,377
Internal (Training/Validation) Set	500,000 (random)
Training Set	450,000 (90% of Internal Set, random)
Validation Set	50,000 (10% of Internal Set, random)
External Test Set	166,597 (10% of ChEMBL, random)

SMILES strings above 150 characters were removed, the rest was transformed into canonical form by ChemAxon. Additional filters were applied to remove rare or unsuitable atoms (Mg, K, Ga, Ge, Ti, etc...) and charged atoms or isotopes.

Descriptor filtering

Descriptors were filtered according to their standard deviation using an in-house script to eliminate features which had no variation in their appearance in the database. The script calculates the standard deviation of all descriptors across the dataset (6520 descriptors in this case). A threshold is then selected to identify descriptors with almost no fluctuations. The threshold on the standard deviation was calculated at 0.27 which corresponds to 2% of the maximum overall standard deviation. All descriptors which had a standard deviation value below the threshold were eliminated. 371 descriptors remained after filtering; it is important to keep in mind that this number is rather low since only 500,000 compounds out of the 1.6M were kept to accelerate training.

Descriptor standardization

The remaining 371 descriptors were standardized across the training set using the following equation:

$$z = \frac{x - \mu}{\sigma}$$

σ and μ being standard deviation and mean respectively calculated across the whole training set for each descriptor.

Architecture

The architecture is composed of N stacked LSTM, followed by M fully connected Dense layers. The schematic representation of the architecture is depicted in Figure 34.

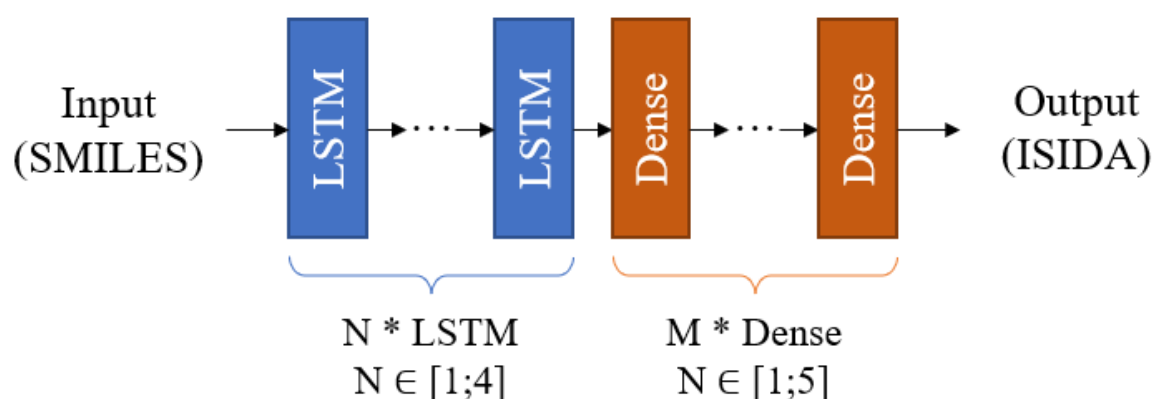


Figure 34. Schematic representation of the model architecture. During the testing procedure, the number of consecutive LSTM cells ranged from 1 to 4, followed by 1 to 5 fully connected Denser layers. The last dense layer had the same dimension as the chosen ISIDA vector and worked as the output of the model.

By analogy with the Euclidean distance, we used the Mean Squared Error calculated between the ISIDA vector of the input SMILES and the ISIDA vector received from the output of the model as the loss function.

$$MSE = \frac{1}{n} \sum_{i=1}^n (X_i - Y_i)^2$$

n is the size of the ISIDA vector, X_i and Y_i are the components of the initial and resulting ISIDA vectors respectively.

The improvement of the model was done in gradual steps: in the early stages, a systematic analysis of all possible combinations of N * LSTM and M * Dense Layers with a fixed set of starting parameters was performed. Using a maximum of 4 LSTMs and 5 Dense layers, 20 models per step were trained. Using this methodology, the best model for the different inputs/starting parameters could be isolated. Filtering and/or standardization of ISIDA descriptors as well as SMILES randomization were also tested to check their influence on model performance.

Parameters initialization

Parameters used across all experiments (unless specifically stated otherwise) are reported in Table 14. Only the dimension of the Dense layers varied, following the size of the ISIDA vectors.

Table 14. Parameters used across all experiments in the SMI2ISIDA models.

Learning rate*	Batch size	Dimension of LSTM cell	Dimension of dense layers	Activation function in Dense layers	Loss function
0.001	256	256 (Bidirectional)	Equal to the dimension of ISIDA vectors	ReLu	MSE

**Learning rate is divided by half every time the validation loss does not improve*

Influence of descriptors on the reconstruction error

Understanding which descriptors induce the most error in the model is an important step in the optimization process. A workflow to isolate the most problematic fragments was designed. 10,000 ISIDA vectors were predicted from random compounds in the test set and compared to the actual vectors calculated from the same compounds. The absolute difference between the initial and predicted ISIDA vectors was then calculated using the following formula:

$$d = |x_{init} - y_{pred}|$$

These calculations resulted in 10,000 “difference” vectors where each element of each vector corresponded to the absolute difference between initial and reconstructed fragment. Mean and standard deviation were then computed for these vectors. A schematic representation of these calculations is shown in Figure 35.

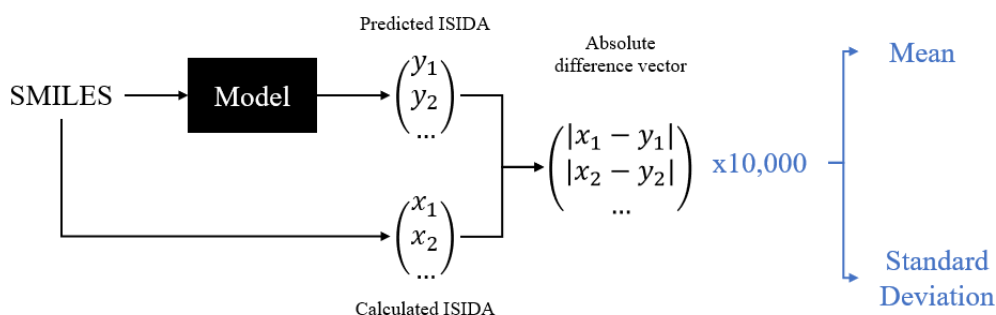


Figure 35. Workflow for the calculations of the absolute difference vectors between calculated and predicted ISIDA vectors. The 10,000 resulting vectors are then studied using Mean and Standard deviation to isolate the most problematic descriptors.

5.1.2 “Raw” descriptors, unique SMILES

The first step of the analysis was to use “raw” descriptors (meaning unfiltered, non-standardized) with canonical SMILES. The 20 models were trained, and results are reported in Table 15. The dimension of ISIDA descriptor vectors is 6520.

Table 15. Minimum validation loss achieved during training with the different numbers of LSTM cells and Dense Layers. Green represents the best model, red represents the worse.

# of Dense \ # of LSTMs	1	2	3	4
1	0.0279	0.0209	0.0249	0.0800
2	0.0348	0.0277	0.0369	0.0468
3	0.0418	0.0321	0.0316	0.0329
4	0.0448	0.0342	0.0364	0.0376
5	0.0588	0.0386	0.0416	0.0370

The best model isolated in this analysis was 2 LSTM cells and 1 Dense layer. For this type of descriptors, deeper models performed worse.

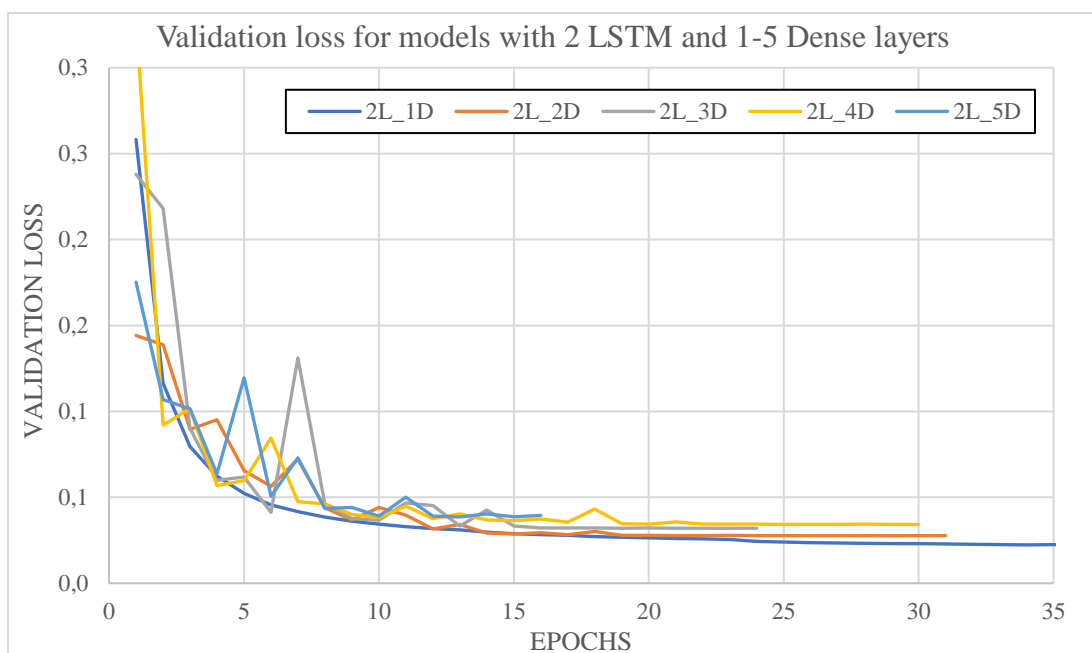


Figure 36. Validation loss during training for models with 2 LSTM cells and 1-5 Dense layers. Training is stopped when the loss doesn't improve for multiple epochs, meaning some models train faster than others.

Table 15 shows that adding Dense Layers rendered the models less accurate compared to the baseline with 1. However, the increasing depth seemed to accelerate training. On average, with this reduced dataset of 500,000 compounds, one epoch took around 3.5 minutes to be processed which meant that the difference in training time was at least 1.5 hours between a model with 1 Dense layer and a model with 5. These training times were still reasonable even in the eventuality of a bigger dataset; thus, it was decided that the best model in terms of loss reduction - ('2L_1D'; Figure 36) – would be selected for further testing.

The initial set of parameters used for the previous analysis were standard parameters regularly used with this kind of neural networks but are not necessarily the best. Thus, another analysis was performed, this time varying Batch Size and Learning rate. Results are shown in Table 16.

Table 16. Minimum validation loss achieved during training with the 2L_1D model with different combinations of starting parameters. The initial parameters that were used to first train this model are shown in light blue. The best model obtained is shown in green.

Batch size \ Learning rate	0.0005	0.001	0.005
	0.0222	0.0227	0.0639
128	0.0222	0.0227	0.0639
256	0.0220	0.0209	0.0456
512	0.0190	0.0233	0.1025

Table 16 shows that the model performance also depended on starting parameters. The performance increased when the learning rate was lowered, and the best performance was achieved with a batch size of 512 compounds.

The resulting model was evaluated on 10,000 random compounds out of the 166,597 in the external test set by computing MSE between initial and reconstructed vectors. On the external test set, MSE amounted to **0.0221** which is close to the validation MSE of **0.0190** showing no sign of overfitting.

Table 17. Top 5 descriptors with highest mean absolute difference value.

Fragment number	Fragment SMILES	Mean absolute difference
38	CCCCNCC	1.02868
208	CCCCC	1.00118
20	CCCCC	0.98502
103	CCCCCNC	0.97515
211	CCCCCCC	0.96136

Table 18. Top 5 descriptors with highest standard deviation of absolute difference value.

Fragment number	Fragment SMILES	STD of absolute difference
211	CCCCCCC	1.66083
697	CCCCCCF	1.37969
203	CCNCNCC	1.36696
208	CCCCC	1.34939
38	CCCCNCC	1.23589

As is clearly visible in Table 17 and Table 18, reconstruction of longer fragments caused significant errors. Combining the mean and standard deviation of Fragment n°38 for example, gave a maximum absolute difference of 2.2. In practice, this meant that the model may predict that such a fragment count is wrong by 2 units or more. Recurring, significant errors in the counting of several descriptors meant the model was not able to reconstruct any ISIDA vectors entirely and perfectly. The model is therefore not satisfactory to replace an algorithmic calculation of the descriptors such as in the FRAGMENTOR software.

To understand the fluctuations in the reconstructed data, a small analysis of the initial training data was carried out. The standard deviation of each of the 6520 descriptors was computed, sorted, and plotted in Figure 37.

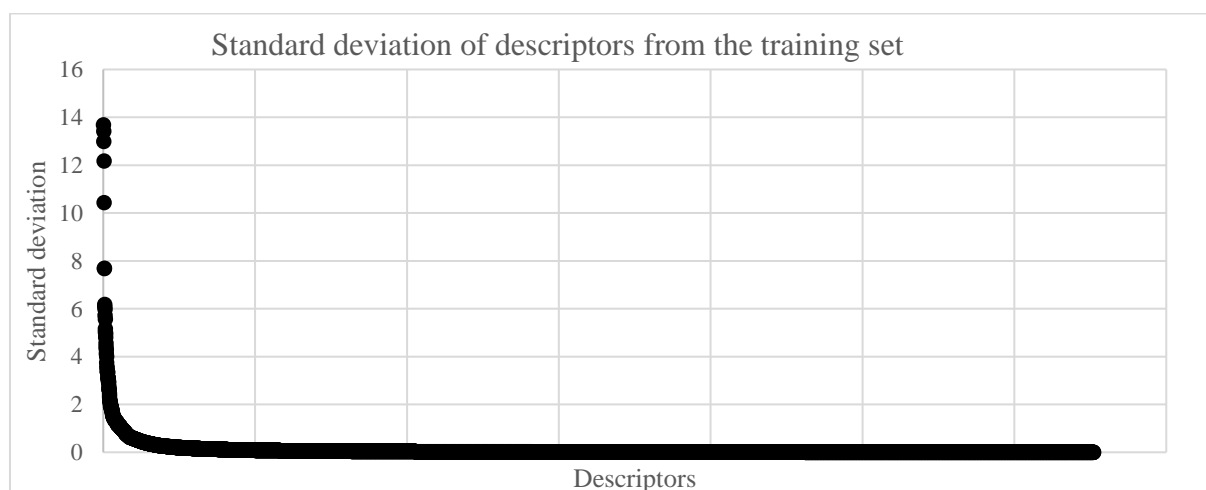


Figure 37. Graph of standard deviation for each of the 6520 descriptors of the training set (500,000 compounds), sorted by increasing standard deviation. Most of the descriptors from the training set have a standard deviation of 0.

A large portion of descriptors had an almost-0 standard deviation in the training set. These descriptors were easily predicted to their average and near constant value. However, the good prediction performances might hinder the training to fit those descriptors with larger variance. The low variance descriptors were therefore filtered out in the following.

5.1.3 Filtered descriptors, unique SMILES

The training set was filtered according to standard deviation, leaving 371 descriptors remaining as shown in Figure 38.

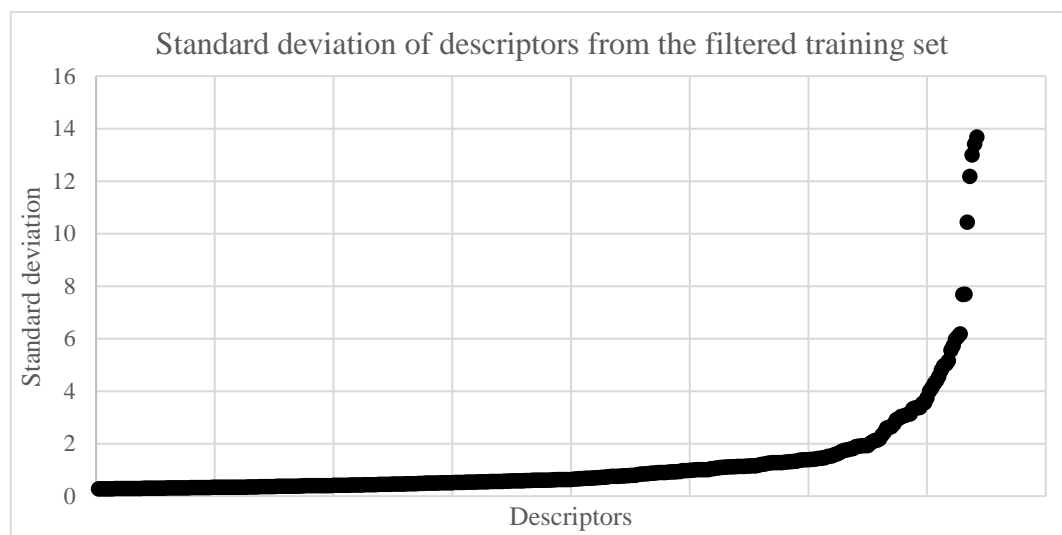


Figure 38. Graph of standard deviation for each of the 371 descriptors of the training set remaining after filtering, sorted by increasing standard deviation.

These descriptors, which can be considered more “meaningful” were used to reproduce the systematic analysis of the architecture to find the best model. The absence of constant descriptors induces an increase of the loss function in the inverse proportion of the decrease of the number of descriptors. The number of descriptors was divided by a factor of almost 2 after filtering, but the MSE only went up by a factor of 10 which means that the filtering process allowed for a better model. This difficulty to compare the performances of models trained on all descriptors and on filtered descriptors only, underlines the defects of the MSE as loss function.

Table 19. Minimum validation loss achieved during training with different numbers of LSTM cells and Dense Layers. The best validation loss achieved is shown in green.

# of LSTMs # of Dense	1	2	3	4
1	0.514	0.228	0.246	2.965
2	0.531	0.248	0.221	0.194
3	0.466	0.263	0.266	0.278
4	0.555	0.385	0.256	0.317
5	0.532	0.380	0.291	0.351

The best isolated model was 4 LSTM and 2 Dense Layers, with a validation MSE of **0.194** (see Table 19).

The model performed an MSE of **0.258** on the external test set.

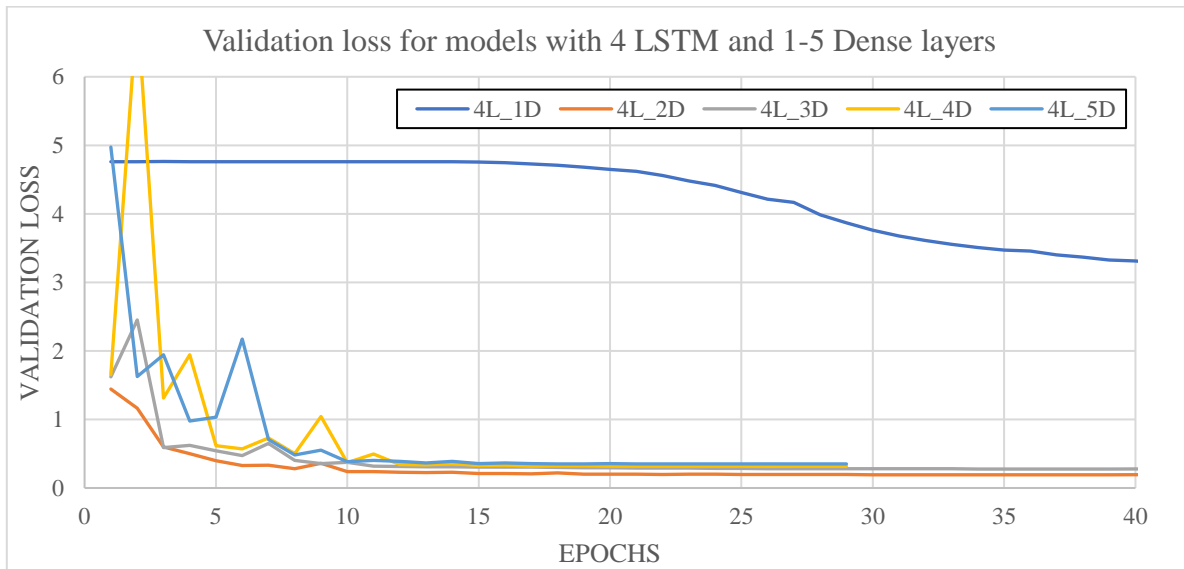


Figure 39. Validation loss during training for models with 4 LSTM cells and 1-5 Dense layers.

Figure 39 confirms the tendency observed in the last experiment. Models with a higher number of Dense layers tend to train faster but are less accurate. Models with 2 or 3 dense layers took around 15 more epochs to train than models with 4 and 5. The model with 1 Dense layer could not reduce validation loss to a performing level, most likely due to overfitting. For further testing, the best model, 4L_2D was retained.

Exactly like the previous analysis, the reconstruction capacity of the trained architecture was tested on 10,000 compounds from the external test set. Mean and standard deviation of absolute difference were computed for all 371 remaining descriptors.

Table 20. Top 5 descriptors with highest mean absolute difference value.

Fragment number	Fragment SMILES	Mean absolute difference
10	CCCCC	0.75550
8	CCCC	0.75093
135	CCCCCC	0.74331
28	CCCCNCC	0.72690
66	CCCCCNC	0.71237

Table 21. Top 5 descriptors with highest standard deviation of absolute difference value.

Fragment number	Fragment SMILES	STD of absolute difference
137	CCCCCCC	1.21887
135	CCCCCC	1.18268
203	CCCSCNCC	1.07405
150	CCCNCC	1.05791
8	CCCC	1.05507

Comparing Table 17 with Table 20 and Table 18 with Table 21 shows that the filtering process was helpful. The mean of the absolute difference dropped about 0.25 in the top 5 and in general through the data. The most problematic fragment for the standard deviation was the same in both non-filtered and filtered experiments and the error dropped from 1.66 to 1.22 which is an improvement of 0.4. All other in top 5 showed an improvement of 0.2. Still, the standard deviation and the mean combined meant that descriptors could be predicted up to 2 units away from their real values.

One unexpected issue came to light when metrics for the model according to the length of fragments were computed. The model seemed unable to count single atoms as shown in Figure 40.

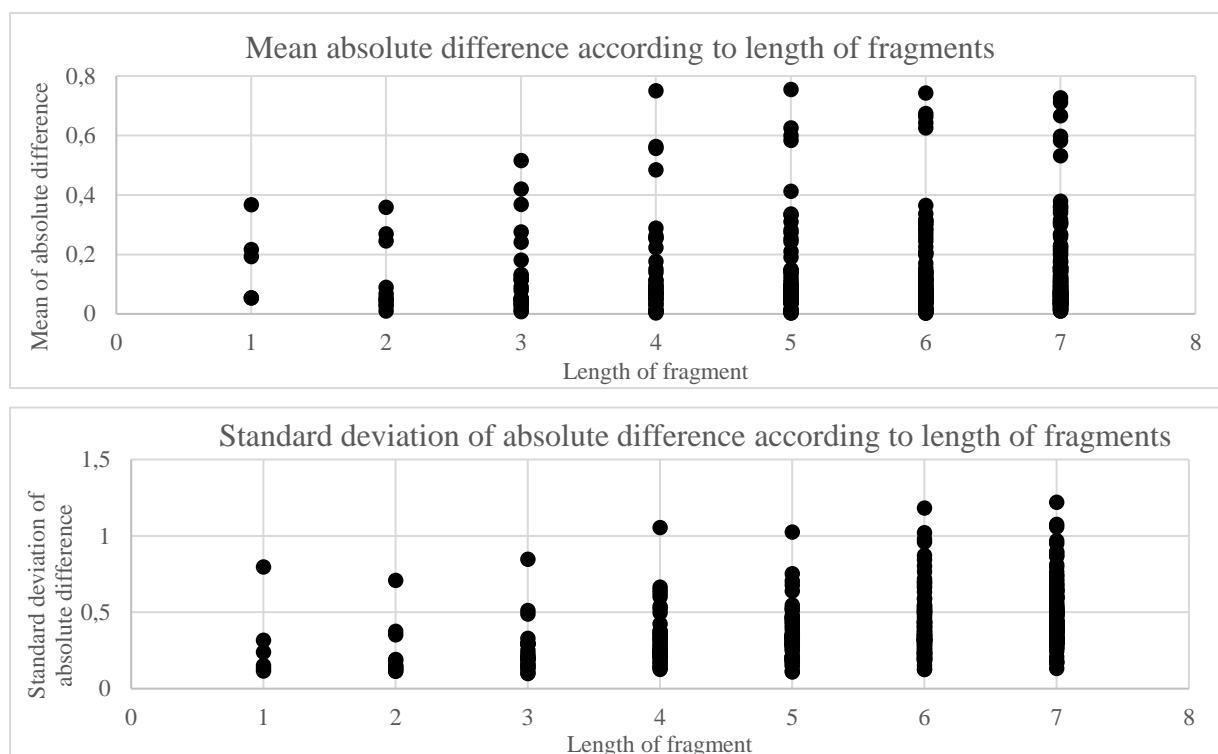


Figure 40. Graphs of mean and standard deviation of the absolute difference according to fragment length. Descriptors are fragment counts. Even some atoms are miscounted by the model.

Counting atoms should be done easily since the correlation between number of characters and number of atoms in the molecule is linear. An inability of the architecture to fulfil the simplest task shed doubt on its ability to reproduce more complex counting. To test that the model could count atoms, the output was modified from ISIDA vectors to only single atom counts with a vector of dimension 9.

9 atoms were considered: C, N, O, S, F, I, Br, Cl, P

A counting accuracy of about 95% was achieved. However, looking at the results it was found that the accuracy of the model when counting molecules containing Br and I was 0%, and P was only 68.18%. These high errors appeared because molecules containing these 3 atoms were not well represented in the dataset. Filtering all molecules containing the Br, I and P atoms from the test set raised the counting accuracy to 99.94%. This shows that the neural network architecture is not able to represent a simple concept such as counting characters in a SMILES string. Shifting the focus from counting atoms to counting sequences of variable lengths induces more errors.

One idea to optimize the model was to use standardized descriptors, so that all 371 descriptors had the same mean and standard deviation, which was the next step for the analysis. The idea was that standardization may help because if all descriptors vary in the same way, the model must learn to understand these variations and link them to the differences in the SMILES string instead of learning by heart possible values for each descriptor according to their standard deviation.

5.1.4 Standardized, filtered descriptors, unique SMILES

The model that performed best on the previous step was used as an initial benchmark (4L_2D). It achieved a validation loss **0.410** which is significantly worse than the with non-standardized descriptors. The top 5 most problematic descriptors in terms of Mean and Standard Deviation of the absolute difference are reported in Table 22 and Table 23. Note that these differences are recalculated by removing the standardization to be able to compare the results with the non-standardized results which are shown in Table 20 and Table 21.

Table 22. Top 5 descriptors with highest mean absolute difference value (standardized descriptors) along with their standard deviation in the initial dataset.

Fragment number	Fragment SMILES	Mean absolute difference	STD in training set (rank)
8	CCCC	29.089	13.68 (1 st)
11	CCC	26.556	10.43 (5 th)
35	C	24.940	7.688 (6 th)
12	CC	23.078	7.676 (7 th)
10	CCCCC	19.028	13.41 (2 nd)

Table 23. Top 5 descriptors with highest standard deviation of absolute difference value (standardized descriptors) along with their standard deviation in the initial dataset.

Fragment number	Fragment SMILES	STD of absolute difference	STD in training set (rank)
8	CCCC	28.220	13.68 (1 st)
11	CCC	26.266	10.43 (5 th)
35	C	24.938	7.688 (6 th)
12	CC	23.033	7.676 (7 th)
10	CCCCC	17.253	13.41 (2 nd)

Values for the error are an order of magnitude larger than the errors in unstandardized descriptors. The first explanation for this phenomenon is that reverting the standardization to calculate the absolute differences multiplies the errors in reconstruction by the standard deviation of the descriptor, making them larger. The higher the standard deviation for a descriptor, the higher the calculated absolute difference will be as is shown in Table 22 and Table 23. The highest errors in reconstruction correlate with the highest standard deviation of the actual descriptors. This implies that the standardized error is well balanced across the dataset which could be explained by looking at an example of descriptors represented in Figure 41.

Non-standardized SVM

5:3 6:1 7:1 8:8 9:1 10:2 11:9 12:9 13:3 14:1 17:2 18:3 19:2 24:4 26:2 29:5 30:2 33:3
34:1 35:11 36:4 37:2 41:1 45:1 57:1 58:2 59:3 60:3 65:2 66:2 72:2 74:2 75:1 80:2
84:1 87:1 89:2 98:1 99:1 100:1 106:1 127:2 130:2 172:6 173:4 174:2 175:1 220:1 238:3
239:3 244:1 275:6 276:6 297:6 371:0

Standardized SVM (shortened)

1:-0.1711 2:-0.1322 3:-0.1785 4:-0.1876 5:-0.814 6:-0.7348 7:-0.6723 8:0.2532 9:-0.6424
10:0.2281 11:0.0361 12:-0.0694 13:0.3056 14:-0.3779 15:-0.3303 16:-0.4947 17:-0.5584
18:-0.5663 [...] 367:-0.0169 368:-0.0168 369:-0.0458 370:-0.0346 371:-0.0072

Figure 41. Comparison between non-standardized (blue) and standardized (green) ISIDA descriptors. Note that the standardized have been shortened as they would be too long to represent. The shortened representation is enough to try to explain the switch in tendency. The notation follows the libSVM notation. The data are given as tuples, the first number I the ID of the descriptor and the second one, after the “:” character, is the value of the descriptor. Null values are not written.

In the non-standardized SVM (Figure 41), only fragments which are contained in the molecules are represented, other fragments that have a value of 0 are not represented (and have a 0 value for the model). In the standardized version, ALL descriptors are represented from 1 to 371 and have non-zero values in similar numerical ranges. It might be that since descriptors are always represented no matter what in a similar fashion, the model had trouble associating a certain SMILES pattern with its position in the descriptor vector. With standardized descriptors, all descriptors have the same standard deviation, and thus the model treats them as equivalent and minimizes the loss in an equivalent manner. Which means that when they are converted back to their initial values, the descriptors with the highest standard deviation and mean become the descriptors with the highest error, which is exactly what can be observed in Table 22 and Table 23.

Standardization did not in fact help with the reconstruction error but worsened the model performance by basically confusing it as to which descriptors were associated with which character sequences in the SMILES strings. Understanding why the model could not count simple carbon chains fragments meant that a deeper understanding of the interpretation process of the model was necessary. Figure 42 shows two examples of a possible explanation for the miscounting of simple fragments.

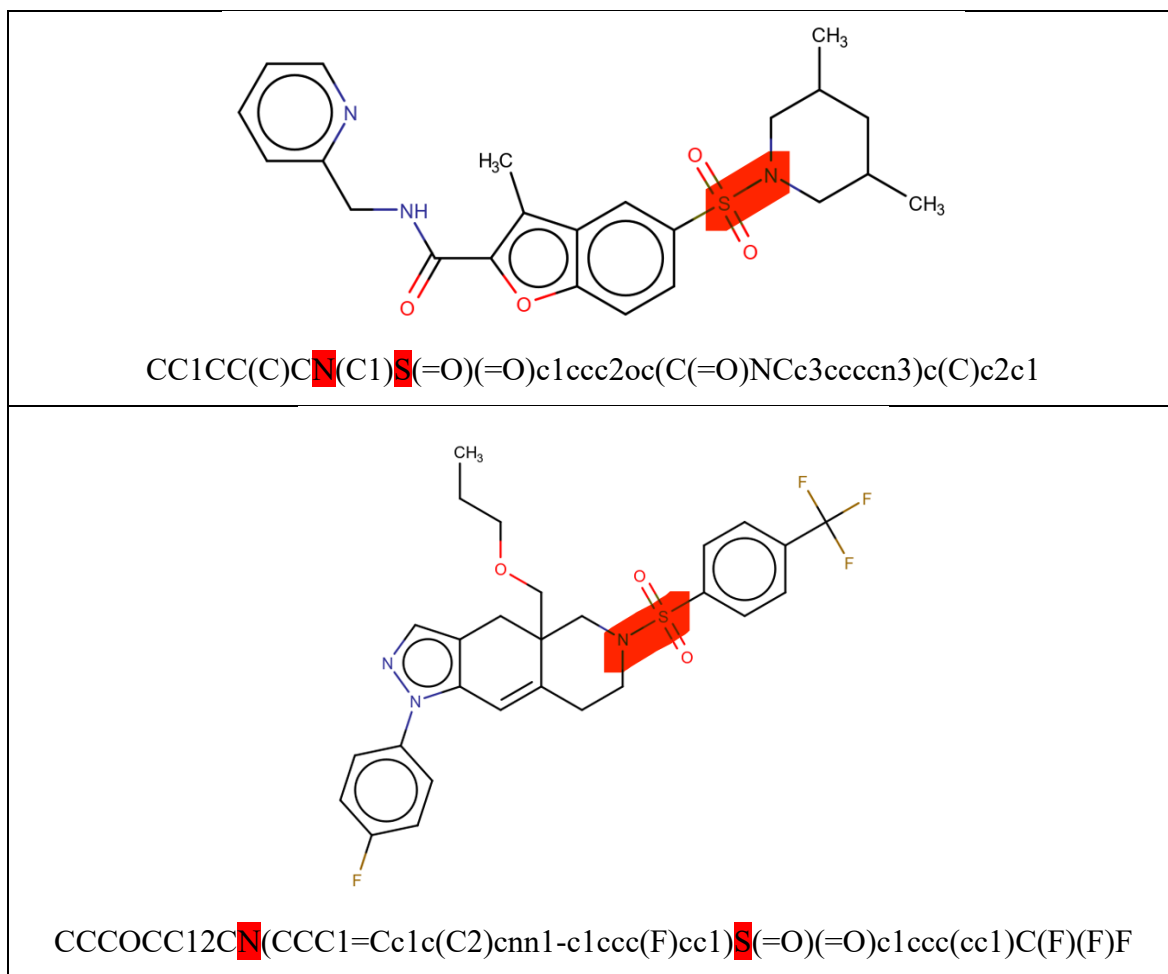


Figure 42. Example of molecules where 2 atoms are adjacent to each other in the molecule but are separated by a variable number of characters in the associated SMILES string.

In both these molecules and associated SMILES, the NS fragment is highlighted in red. Due to the construction of the SMILES, the N and the S are not adjacent to each other even though they share a bond in the molecule. The model therefore has more difficulty making the connection between them and counting them as a fragment. This is a limitation of SMILES and LSTMs that was already observed in the canonical-to-canonical SMILES autoencoder. When generating large cyclic structures, the model had trouble connecting cycle indicators that were very distant in the SMILES string, in multi-cyclic structures for example where several cycles are opened and then closed.

5.1.5 Filtered descriptors, enumerated SMILES

To solve this issue, a data augmentation strategy was adopted through the randomization of SMILES. Through randomization and using a larger number of SMILES strings for each molecule, fragments that could be separated in a certain SMILES might be adjacent in another different SMILES. The model could therefore detach itself from the SMILES sequence and get a higher representation of the molecule represented by the SMILES. SMILES Randomization was performed using RDKit, by transforming a SMILES string into a molecule object, extracting, and shuffling atom numbers and recreating a SMILES string. For each SMILES in the ChEMBL database, 10 random SMILES were generated and associated with the same filtered, non-standardized SVM. Again, based previous knowledge, a 4 LSTM, 2 Dense layer model was selected.

Training resulted in a validation loss of **0.370** which is significantly worse than the **0.194** obtained with the best model with non-randomized, non-standardized SMILES. The same data augmentation strategy has not been investigated on the filtered descriptors dataset due to a lack of time.

Another architecture was tried to fit to the 10 enumerated SMILES per compounds: each alternative SMILES being assigned to an independent channel then fused in a dedicated layer. The architecture tested is shown in Figure 43.

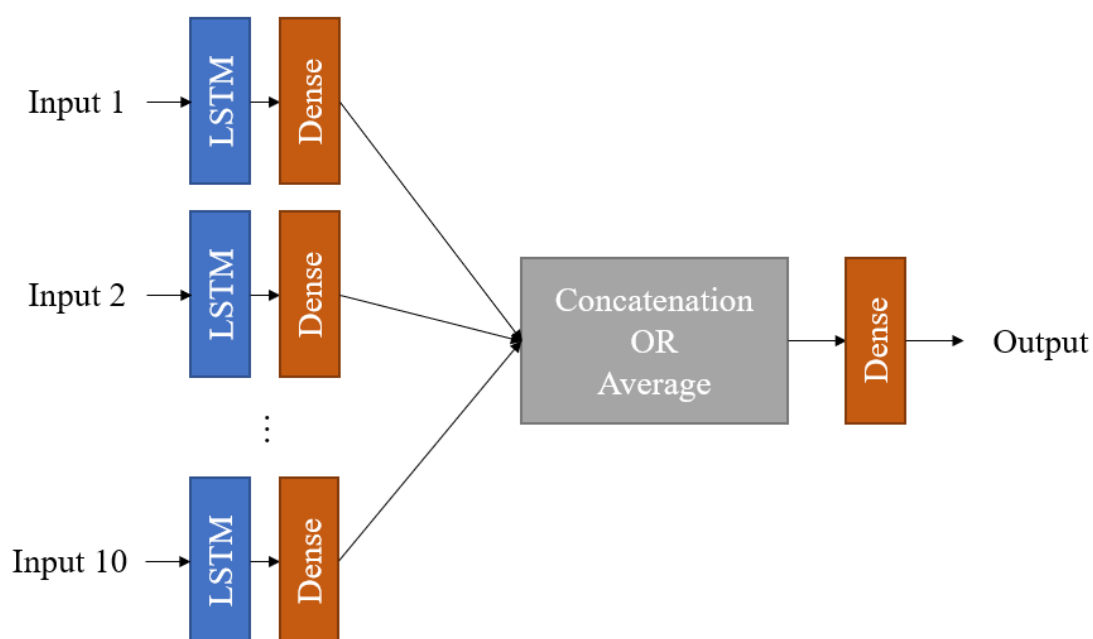


Figure 43. Schematic representation of the changed architecture. Each input corresponds to a different randomized SMILES string. Then, all outputs from the 10 LSTM+Dense duos, are either concatenated or averaged (two different models). Then this new vector is passed through the output dense layer to shape it with the corresponding size.

Results of both models are reported in Table 24. This architecture produced worse models than previous attempts.

Table 24. Minimum validation loss achieved for both types of model.

Model type	Minimum validation loss achieved
<i>Concatenation</i>	0.520
<i>Average</i>	2.387

5.1.6 Conclusion

The aim of this project was to train a model that could compute ISIDA molecular descriptors accurately and faster using a machine learning model. The resulting models produced too many errors to be considered as a replacement to an algorithmic exact calculation.

It was clearly highlighted that SMILES strings combined with LSTM cells have strong limitations. The grammar of SMILES seems to be more complex for a model to apprehend than classical human language grammar. For example, a simple phrase in English will always have: Subject + verb + agreement in that order. This predictable behaviour is easy to identify and learn for a Deep Neural Network. The language of molecules it seems, is much more complex and a simple architecture was not able to crack the secrets of the SMILES representation.

The impact of SMILES randomization, descriptor filtering and standardization was tested on the process of learning. It seems preferable to work on counts without further transforming the molecular descriptors. Filtering out low variance descriptors seems beneficial to improve both the speed of the training and the accuracy of the model.

Data augmentation using alternative SMILES representation could be a possible solution to improve the model. Complexification and better control over the regularization of the model could also be tested. A graph-based approach could probably be a better fit, since all molecular connections would be considered by the model.

5.2 Multimodal Deep Boltzmann Machines

The Multimodal Deep Boltzmann Machine (MDBM) is an architecture that was considered in the scope of Constrained Generation, with the aim to create an ISIDA to SMILES model which would be able to generate SMILES strings from vectors with selected fragment counts and property values. A strategy to approach the problem of constrained generation is to link ISIDA structural descriptors and the latent vectors of an AE, the latter being prepared to decode SMILES strings. The mapping between ISIDA and an AE feature space is ensured, here, by an MDBM.

Boltzmann Machines are models with pairwise interacting units that update their states over time in a probabilistic manner depending on the states of adjacent units. They can be regarded as stochastic versions of Hopfield networks^[159]. The most striking feature of this architecture is that it contains only one visible layer that is used as both input and output of the network. Restricted Boltzmann Machines^[160] (RBM) and Deep Boltzmann Machines^[161] (DBM) are special types of Boltzmann machines in which the interactions are done between layers of units. In the case of a RBM, the interactions are limited to the visible and the hidden set of units, no connections are allowed inside the visible or the hidden layer. A DBM has multiple hidden unit layers communicating sequentially: meaning communication is allowed between layers but still not in the layers.

Isolated Boltzmann Machines can work in the same way as an AE, reconstructing its input from a latent vector, however in this case there is no separation between encoder and decoder since the visible layer serves as both input and output. The visible units take the input and the information travels deeper into the model, layer by layer until reaching the deepest layer, where it can go backwards towards the visible units. The deepest layer serves as a latent representation of the input data, setting the state of the deep hidden layer, the model can generate a data vector that can be read in the visible layer.

Multimodal Deep Boltzmann Machines^[162] are a combination of several DBMs trained to reconstruct their input after creating a latent representation of said input. The different DBMs work in different modalities, where the same concepts are expressed in different manners (e.g. the word “cat”, images of cats, sounds of cats). All the deepest layers of the different DBMs are connected by a single layer, which serves as a unique, latent representation of all the modalities

of the same concept. Thus, from one latent vector, the model generates new data in any of the input modalities.

ISIDA and latent descriptor vectors can be seen as two different modalities of the same molecule and could therefore be linked by an MDBM. The goal is to generate an AE latent vector, that can be decoded as SMILES from an ISIDA vector representation.

5.2.1 Boltzmann Machines

A classical Boltzmann Machine (BM) is an energy-based neural network, composed of symmetrically connected binary units. The symmetry means that artificial neurons are unique mathematical functions of the values from the adjacent artificial neurons, in contrast to a feed-forward neural network (like an autoencoder), where artificial neurons use two different functions, in forward and backward mode. For this reason, there is no direction in a BM, only an arbitrary choice to define which are the visible and the hidden units. Binary units mean that an artificial neuron can have two states: 1 or 0. A classical BM is represented in Figure 44.

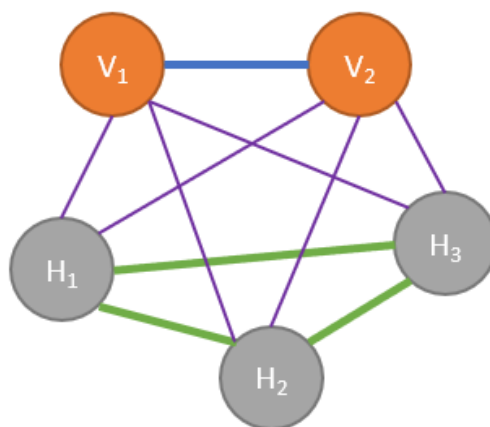


Figure 44. Simple scheme of a classical Boltzmann Machine. Orange neurons represent visible units which take the input and give the output. Grey neurons represent hidden units which, similarly to an autoencoder, are supposed to gather higher representations and the underlying “meaning” of the data distribution. Bold lines represent connections between units of the same layer. Coloured lines represent the different interactions between units. Green lines are interconnections inside a hidden layer, the blue line represents the interconnection inside the visible layer and the purple lines represent connections between two layers.

As can be seen in Figure 44, in a BM all units are connected, and there are interconnections between units of the same layer. This architecture is designed to model an unknown probability density function from a sample dataset. The mathematical form of the modelled probability distribution is a Boltzmann law, hence the Boltzmann Machine name. The energy is a function of the configuration of the units of the model.

$$E(\mathbf{v}, \mathbf{h}; \theta) = -\frac{1}{2} \mathbf{v}^T \mathbf{L} \mathbf{v} - \frac{1}{2} \mathbf{h}^T \mathbf{J} \mathbf{h} - \mathbf{v}^T \mathbf{W} \mathbf{h} - \mathbf{b}_v \mathbf{v} - \mathbf{b}_h \mathbf{h} \quad (5.1)$$

With \mathbf{v}, \mathbf{h} the state of the visible and hidden units respectively, $\theta = \{\mathbf{L}, \mathbf{J}, \mathbf{W}, \mathbf{b}_v, \mathbf{b}_h\}$ the parameters of the model to fit. The terms $\mathbf{b}_v, \mathbf{b}_h$ are the biases of visible and hidden units respectively (threshold of activation for both units). \mathbf{L} and \mathbf{J} account for intra-layer interactions and \mathbf{W} for visible-hidden layers interactions. Colours in the equation refer to the colours in Figure 44.

From this energy function, and using the Boltzmann Distribution, the probability of activation of each visible and hidden unit can be inferred. In turn this probability is used to set the state of the corresponding artificial neuron. Unfortunately, BM do not currently benefit from any algorithmic acceleration to train. But RBMs do, and for this reason, are preferred.

5.2.2 Restricted Boltzmann Machines

A Restricted Boltzmann Machine (RBM) is a version of the Boltzmann Machine where interconnections inside the same layers have been removed as shown in Figure 45.

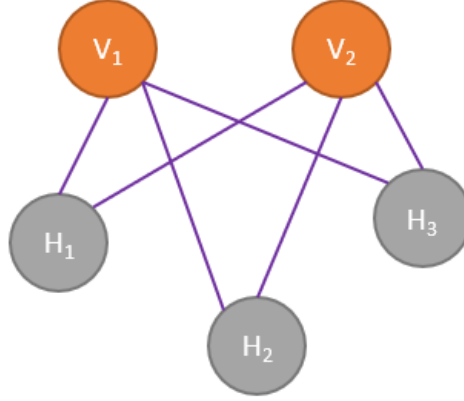


Figure 45. Simple scheme of a Restricted Boltzmann Machine. Interconnections between units of the same layer have been removed. As a result, this architecture looks more like a classical feed-forward neural network.

Since interconnections have been removed, the energy equation is simpler:

$$E(\mathbf{v}, \mathbf{h}; \theta) = \mathbf{v}^T \mathbf{W} \mathbf{h} - \mathbf{b}_v \mathbf{v} - \mathbf{b}_h \mathbf{h} \quad (5.2)$$

We modified this function to use real valued artificial neurons for the visible layer and binary valued artificial neurons for the hidden layer. The visible layer artificial neurons follow a multivariate normal distribution with the assumption of independence. The energy function becomes:

$$E(\mathbf{v}, \mathbf{h}) = \frac{1}{2} (\mathbf{v} - \mathbf{b}_v)^T \mathbf{\Sigma}^{-1} (\mathbf{v} - \mathbf{b}_v) - \mathbf{v}^T \mathbf{\Sigma}^{-1} \mathbf{W} \mathbf{h} - \mathbf{b}_h^T \mathbf{h} \quad (5.3)$$

Where $\mathbf{\Sigma} = \begin{pmatrix} \sigma_1^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_V^2 \end{pmatrix}$ is the diagonal covariance matrix of shape (V, V) where V is the number of visible units and σ_i^2 are the variances of each input descriptor (a ISIDA fragment count or one of the AE latent vector coordinate). Now, instead of visible units being restrained

to values 0 or 1, the values for each unit V_i will be sampled from a gaussian distribution with the corresponding variance. To train the model, we use a process called Gibbs sampling. The variances are pre-computed.

To counter the problem of sampling in real space which might lead to the gaussian distributions to have an area superior or inferior to 1, Hinton suggested to normalize the data to have mean 0 and standard deviation of 1 which has been done. The normalization was done using the following transformation:

$$\mathbf{y} = \frac{\mathbf{x} - \bar{\mathbf{x}}}{\delta} \quad (5.4)$$

Here, \mathbf{x} is an ISIDA/latent vector, $\bar{\mathbf{x}}$ is the mean, and δ the standard deviation of the data.

5.2.3 Training a Restricted Boltzmann Machine

As an example for this explanation, we will use the output of an encoder which are latent vectors of dimension 256. This data will be fed in batches to the RBM, and we will assume a batch size of 100 (i.e. 100 molecules per batch). Therefore, our input matrix \mathbf{X} has shape (100, 256). We will also assume 256 visible units and 20 hidden units.

The training process is done using Gibbs Sampling as previously mentioned, which works as follows:

1. Sample hidden states from input
2. Sample visible values from hidden states
3. Sample hidden states from sampled visible values
Steps II-III can be repeated k times if needed (in this example, only one iteration was done because it has been shown that $k=1$ gives good results).
4. Calculate and apply gradients (from step 2 and 3)
5. Update parameters

The details of each step will be described in the following chapters.

Sampling hidden states from input passed in visible units

First, we calculate the probabilities for each hidden unit to be activated:

$$\mathbf{P} = P(\mathbf{H} = 1 \mid \mathbf{X}) = \text{sigmoid}(\mathbf{b}_h + \mathbf{X}\Sigma^{-1}\mathbf{W}) \quad (5.5)$$

$\mathbf{b}_h = (b_h^1, b_h^2, \dots, b_h^{20})$ is the vector of hidden biases

\mathbf{H} corresponds to the states of the hidden units

\mathbf{W} the weights matrix of $\text{shape} = (256, 20)$

The decision to activate or not a hidden unit is made by sampling out of the Bernoulli distribution using \mathbf{P} .

As a result, we obtain a matrix of hidden states \mathbf{H} :

$$\mathbf{H} = f(\mathbf{P}), \text{shape} = (100, 20) \quad (5.6)$$

Sampling visible from hidden

The process is rather different because visible units are not binary but real and use a gaussian distribution. The process consists in creating a gaussian distribution to sample from using our newly calculated hidden states.

The probability distribution \mathbf{Q} from which we sample visible states can be expressed as follows:

$$\mathbf{Q} = P(\mathbf{V} \mid \mathbf{H}) = \mathcal{N}(\mathbf{V}; \mathbf{H} \mathbf{W}^T + \mathbf{b}_v, \Sigma), \text{shape} = (100, 256) \quad (5.7)$$

where $\mathcal{N}(x; \mu, \sigma^2)$ is the gaussian distribution with mean μ and variance σ^2 ,

$\mathbf{b}_v = (b_v^1, b_v^2, \dots, b_v^{256})$ is the vector of visible biases

\mathbf{W} is the same matrix of weights.

By sampling random numbers from \mathbf{Q} we get \mathbf{V} of shape $(100, 256)$ which is the matrix of states of the visible units. These states can be reverted to our input space by reverting the normalization process: $\mathbf{V}_{reverted} = \delta \mathbf{V} + \bar{x}$

Sampling hidden from the sampled visible layer state

And again, for the third step of Gibbs sampling, new hidden states are calculated the same way as before:

$$\mathbf{P}' = P(\mathbf{H} = 1 \mid \mathbf{V}) = \text{sigmoid}(\mathbf{b}_h + \mathbf{V}\Sigma^{-1}\mathbf{W}), \text{shape} = (100, 20) \quad (5.8)$$

Calculating gradients

The gradients are computed by calculating the derivative of the log-likelihood against every parameter.

Visible bias :

$$\Delta \mathbf{b}_v = \text{mean}\left((\mathbf{X} - \mathbf{b}_v)\Sigma^{-1} - (\mathbf{V} - \mathbf{b}_v)\Sigma^{-1}\right) * lr, \text{shape} = (256) \quad (5.9)$$

With lr the learning rate

Means are calculated across the batches.

Hidden bias:

$$\Delta \mathbf{b}_h = \text{mean}(\mathbf{P} - \mathbf{P}') * lr, \text{shape} = (20) \quad (5.10)$$

Weights matrix:

$$\Delta \mathbf{W}^T = \text{mean}(\mathbf{P}^T \Sigma^{-1} \mathbf{X} - \mathbf{P}'^T \Sigma^{-1} \mathbf{V}) * lr, \text{shape} = (256, 20) \quad (5.11)$$

Gradients are applied and variables will be changed as:

$$\mathbf{b}_v = \mathbf{b}_v + \Delta \mathbf{b}_v; \mathbf{b}_h = \mathbf{b}_h + \Delta \mathbf{b}_h; \mathbf{W} = \mathbf{W} + \Delta \mathbf{W} \quad (5.12)$$

The training process continues until we reach convergence on RMSE. Reconstruction rate could also be used for ISIDA vectors. In this case, RMSE makes more sense since latent vectors have 10 decimal precision.

As the training process is stochastic, it has to be softly ended. To this end, the learning rate is lowered at each epoch in order.

$$\text{learning rate} = f(\text{epoch}) = \frac{1}{e^{\left(\frac{\text{epoch}}{a} + b\right)}} \quad (5.13)$$

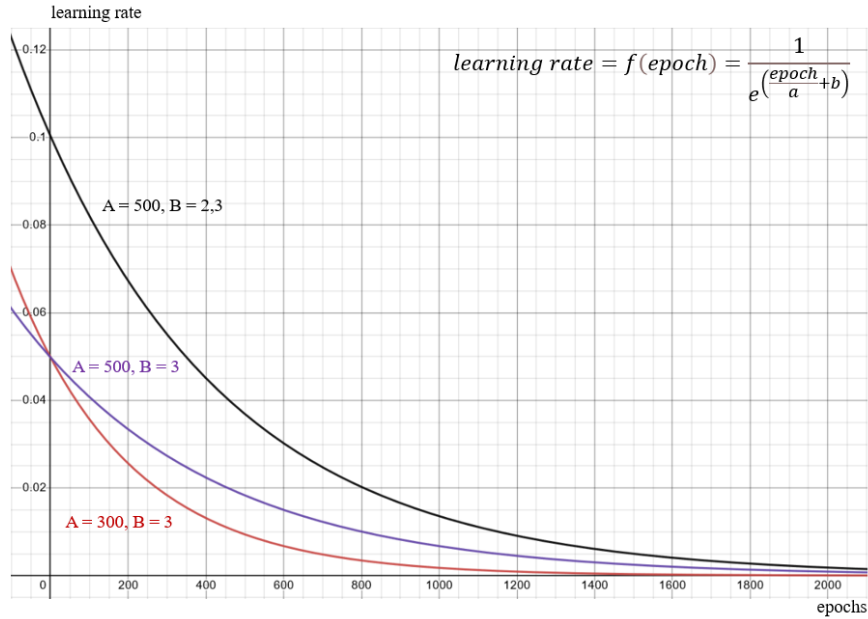


Figure 46. Influence of a and b parameters on the evolution of the learning rate. Parameter “ a ” controls the rate of decrease while “ b ” controls the starting point.

Parameters initialization

From Melchior’s publication we gathered the following parameters initialization

\mathbf{b}_v , visible biases, are initialized to the mean of the data because they tend to that value

\mathbf{b}_h , hidden biases, are initialized as:

$$\mathbf{b}_h = -\frac{\left| \left| \mathbf{b}_v + \mathbf{W}_{*j} \right| \right|^2 - \left| \left| \mathbf{b}_v \right| \right|^2}{\sigma_j^2} + \ln(\tau) \quad (5.14)$$

With $\tau = 0.01$ and σ_j^2 the variance of the j^{th} column of the matrix $\mathbf{\Sigma}$.

$$w_{ij} \text{ are sampled in } U\left(-\frac{\sqrt{6}}{\sqrt{20+256}}, \frac{\sqrt{6}}{\sqrt{20+256}}\right) \quad (5.15)$$

With U a uniform distribution

Σ is initialized to a diagonal of 1.

5.2.4 Multimodal Deep Boltzmann Machine

To build the Multimodal Deep Boltzmann Machine, two separate Deep Boltzmann Machines must be built and optimized first. One model specific to latent vectors and one model specific to ISIDA vectors. Both objectives for the models are to reconstruct their input after passing through their respective latent representation layers. Once optimal parameters for both models have been found, they can be connected using a common layer at the top and trained with both inputs.

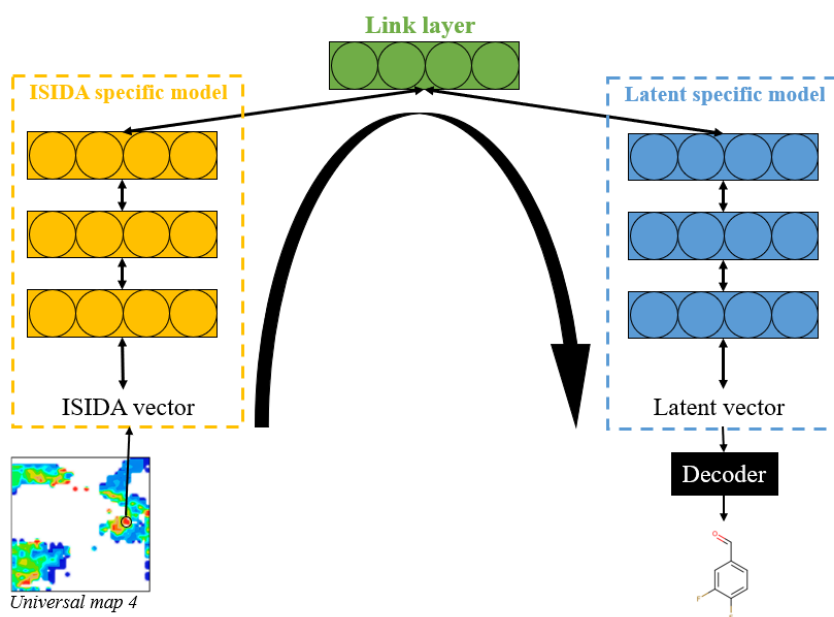


Figure 47. Schematic representation of the Multimodal Deep Boltzmann Machine with both specific models linked by the higher representation layer on top. Details are added to show the model in usage mode, sampling vectors from ISIDA-based GTM, finding their equivalent in latent vectors and decoding them into molecules with controlled structure.

5.2.5 Data

Two types of experiments were performed with two datasets of different sizes. Type A experiments used 10.000 vectors sampled randomly, either from the output of the encoder of a classical canonical-to-canonical autoencoder or from ISIDA descriptors. In both cases molecules that were encoded were extracted from the ChEMBL23 database. In the case of the latent vectors, SMILES were initially transformed in their canonical form and then given to the encoder.

5.2.6 Latent vectors model efficiency metric: SMILES reconstruction rate

To verify that the model performs well in reconstruction tasks for the latent vectors, a metric was needed since the Euclidian distance between latent vectors used during training is not easily understandable. Both latent vectors and reconstructed latent vectors (after passing through) the RBM/DBM were fed to the decoder and decoded into SMILES. These SMILES were then compared and sorted into three categories: Perfect match, meaning that the SMILES before and after reconstruction was the same; invalid SMILES, meaning the encoder generated a meaningless SMILES string given the reconstructed latent vector; and imperfect reconstruction, meaning the output of the decoder was a valid SMILES string but didn't correspond to the input. Imperfect reconstructions were more thoroughly studied by calculating the Tanimoto coefficient between initial and reconstructed SMILES using Morgan-4 Fingerprints. Figure 48 shows a schematic representation of the comparison process.

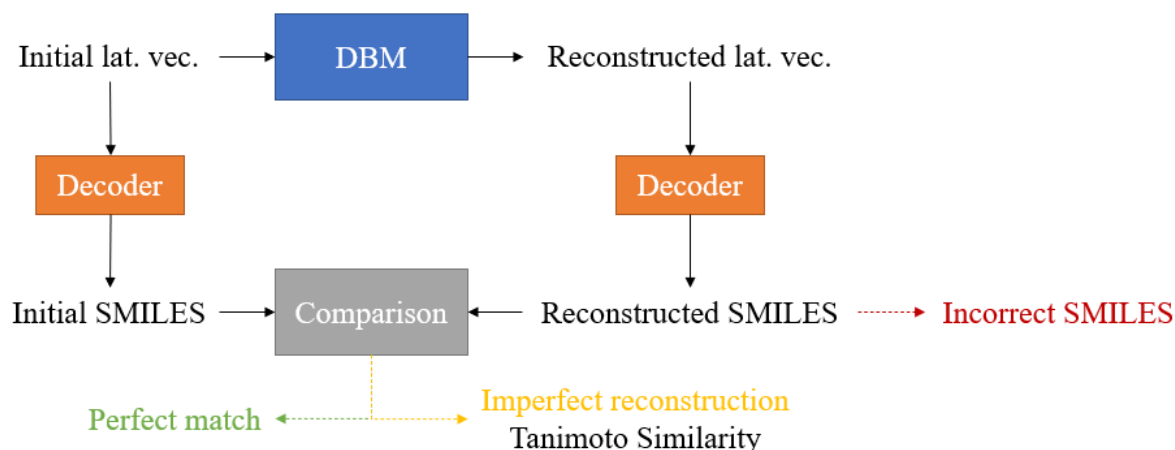


Figure 48. Schematic representation of the efficiency metric for the part of the DBM trained for reconstructing latent vectors. Reconstructed SMILES are sorted into three categories according to their validity and comparison to the initial SMILES.

5.2.7 ISIDA vectors model efficiency metric: Descriptor fluctuation

The initial ISIDA vector is compared to the ISIDA descriptors vector from the reconstructed SMILES. We term these the “reconstructed ISIDA vectors”. The ISIDA descriptor type used contains only fragment counts (i.e., all descriptors have whole numbers), therefore a perfectly reconstructed vector would have the same values as the initial one when all descriptors are rounded to the nearest integers. This means that the error tolerance for correctly reconstructing a descriptor is ≤ 0.5 .

5.2.8 Parameters and architecture optimization – Latent vectors

A thorough analysis was performed to find optimal parameters for both types of models, starting with the latent-specific model. The optimization was done layer by layer. The first step was to find good parameters for a one hidden layer model (RBM), then for a two-layer and maybe three-layer model.

The first experiment of the analysis was performed on a simple RBM with 1 hidden layer of a variable dimension as shown in Figure 49.

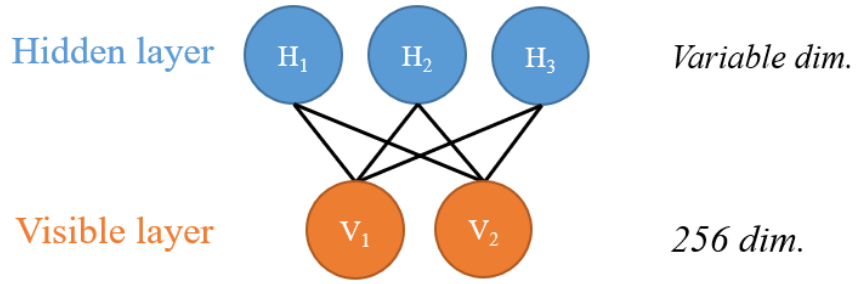


Figure 49. Schematic representation of the simple RBM used in the first experiment. The visible layer has a dimension of 256 corresponding to the size of the input latent vectors. The hidden layer has variable dimension.

Parameters were initialized to the values shown in Table 25.

Table 25. Parameters used for the training of the RBM. Only the dimension of the hidden layer varied.

Hidden Dimension	Batch size	Epochs	A	B	Start Learning Rate
<i>Variable</i>	256	1000	300	3	0.05

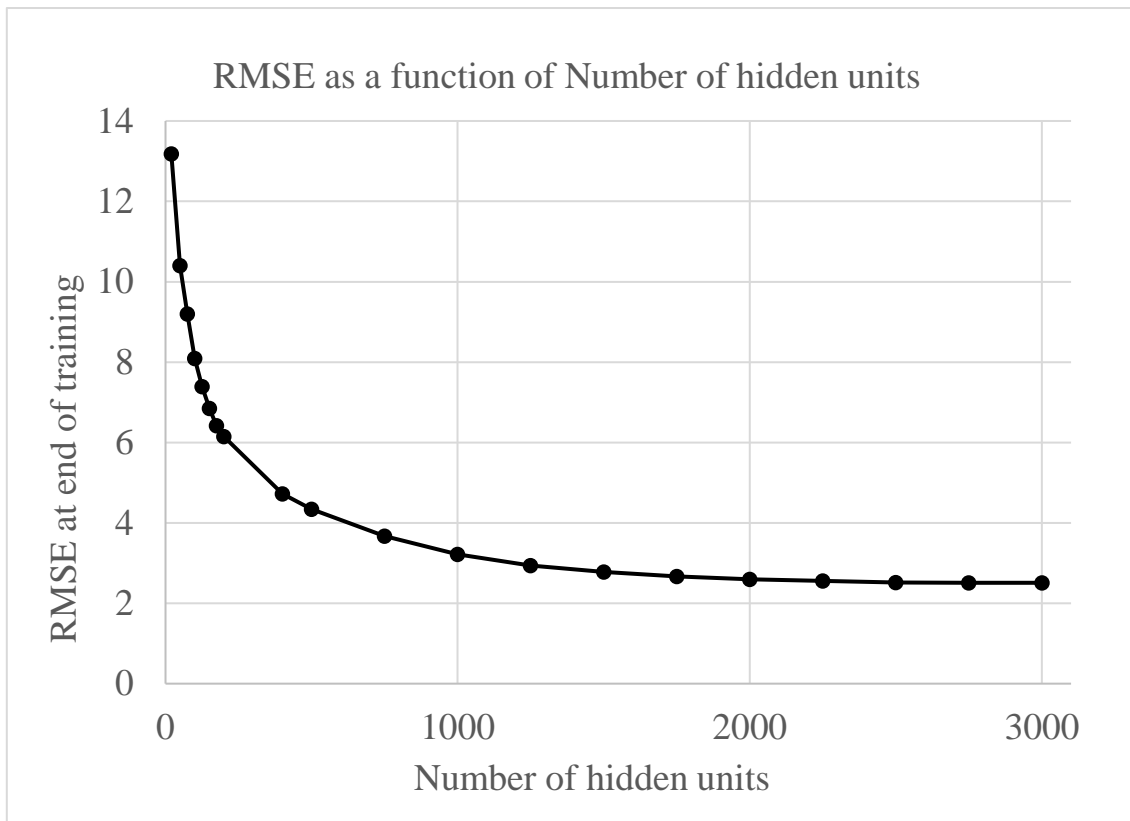


Figure 50. Graph representing the evolution of the RMSE at the end of training in relation the number of hidden units in the model.

Figure 50 shows that increasing dimensions in the hidden layer helped decrease the RMSE significantly. A plateau was reached at around $RMSE=2.5$ for a dimension of 2500-3000. The hidden layer of the RBM was set at a dimension of 3000 for the computation of the reconstruction analysis.

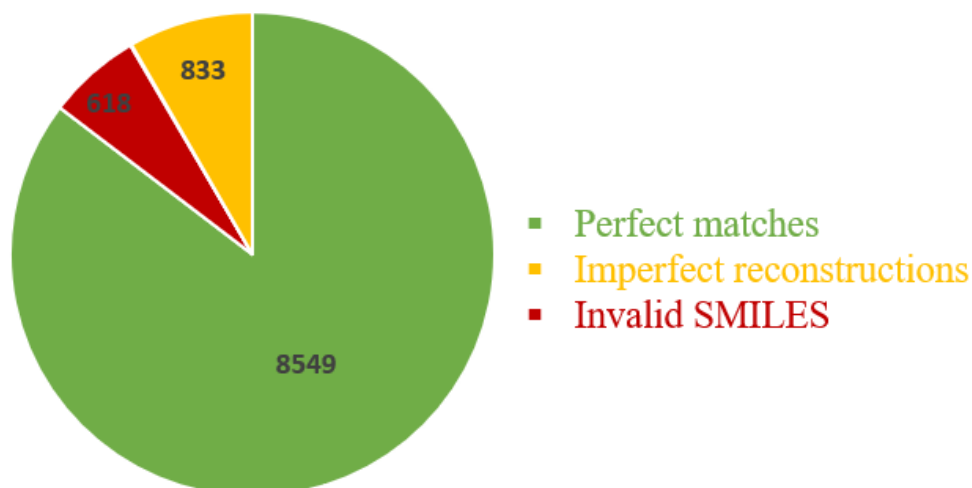


Figure 51. Graph showing the proportion of perfect matches, imperfect reconstructions, and invalid SMILES for the best RBM found in the systematic analysis.

Figure 51 shows that the initial version of the model achieved a perfect reconstruction rate of 85.49%. In comparison to classical canonical-to-canonical autoencoders which generally achieve upwards of 95%, this value is disappointing but encouraging considering the simplicity of the model. Tanimoto coefficients were computed (Figure 52) and showed that most of the imperfect reconstructions had a Tc around 0.5 – 0.7 with a few going above 0.9. Small changes to the latent vectors seemed to induce changes of variable degrees to the closeness of the reconstructed compounds, showing the discrete characteristic of latent space and the tendency of the autoencoders to have a poor chemical space organization in terms of structural similarity.

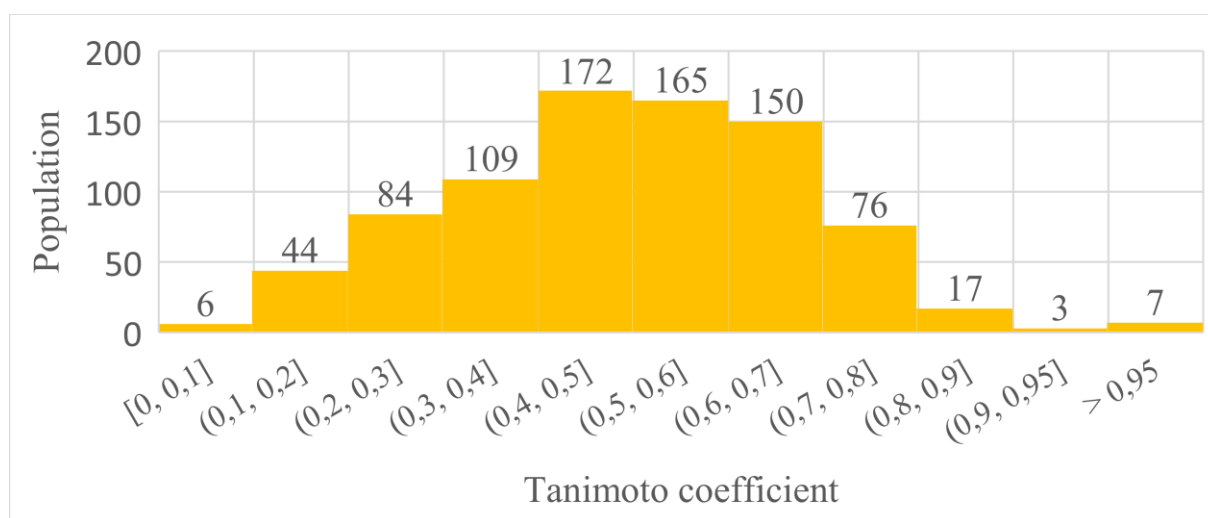


Figure 52. Distribution of Tanimoto coefficients for the 833 imperfect reconstructions.

The second experiment was performed by having the first hidden dimension fixed at 3000 and the second hidden dimension varying as shown in **Figure 53**.

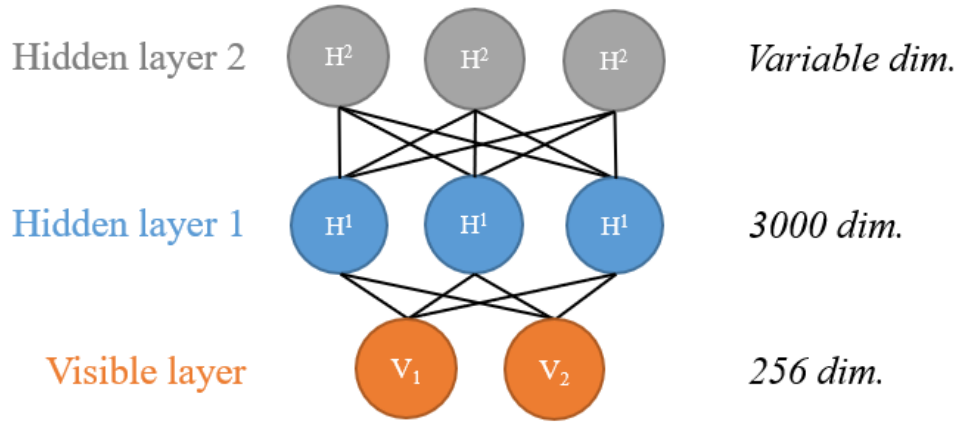


Figure 53. Schematic representation of the DBM evolved from the previous RBM. Visible and first hidden layers are fixed to constant values while the second hidden layer varies.

Parameters for the models in the second experiment were slightly modified compared to the first experiment. The number of epochs was scaled up from 1000 to 3000 and the decrease of the learning rate was slowed by modifying the A value in the learning rate function from 300 to 500 as shown in Table 26.

Table 26. Parameters used for the training of the DBM, only the dimension of the 2nd hidden dimension varied.

Hidden dimension 1	Hidden dimension 2	Number of epochs	Batch size	A	B
3000	75	3000	256	500	3

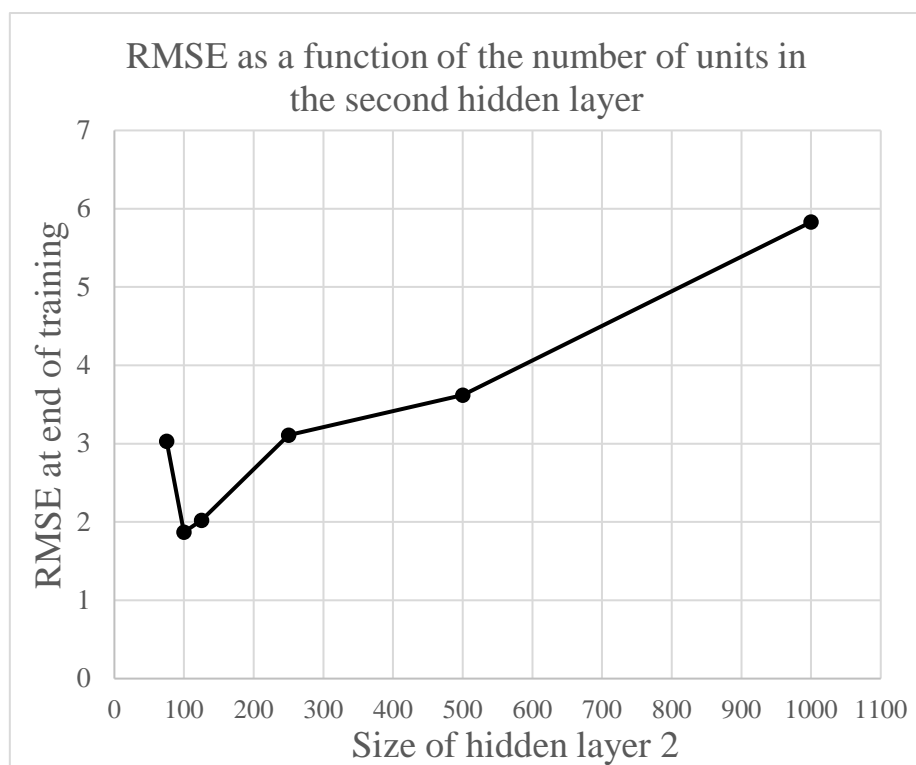


Figure 54. Graph representing the evolution of RMSE at the end of training as a function of the dimension of the second hidden layer.

Results of the second experiment show that the second hidden layer does not benefit from a large hidden dimension. The best result was found for a 100-dimension layer as shown in Figure 54. The Reconstruction rate analysis was again performed using the 10.000 training compounds.

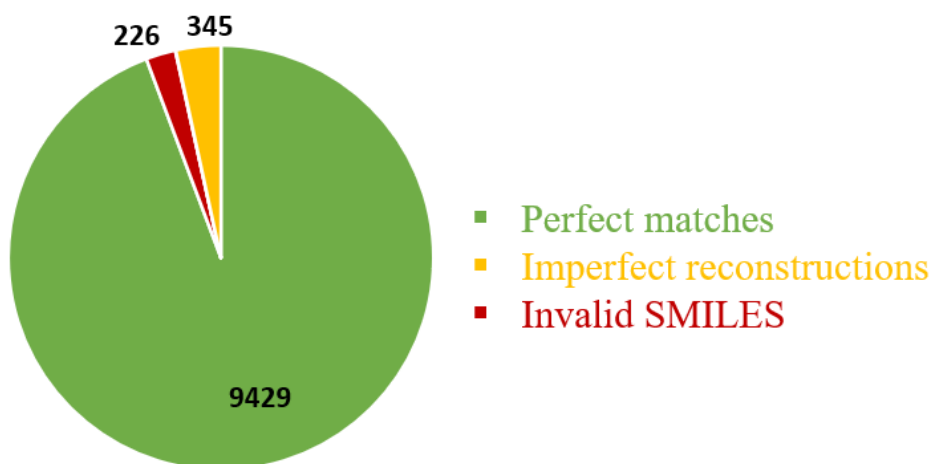


Figure 55. Graph showing the proportion of perfect matches, imperfect reconstructions, and invalid SMILES for the best DBM found in the systematic analysis.

This model achieved a 94.29% perfect reconstruction rate as shown in Figure 55. The breakdown of the Tc for the imperfect reconstructions shows the same trend as the previous experiment (Figure 56).

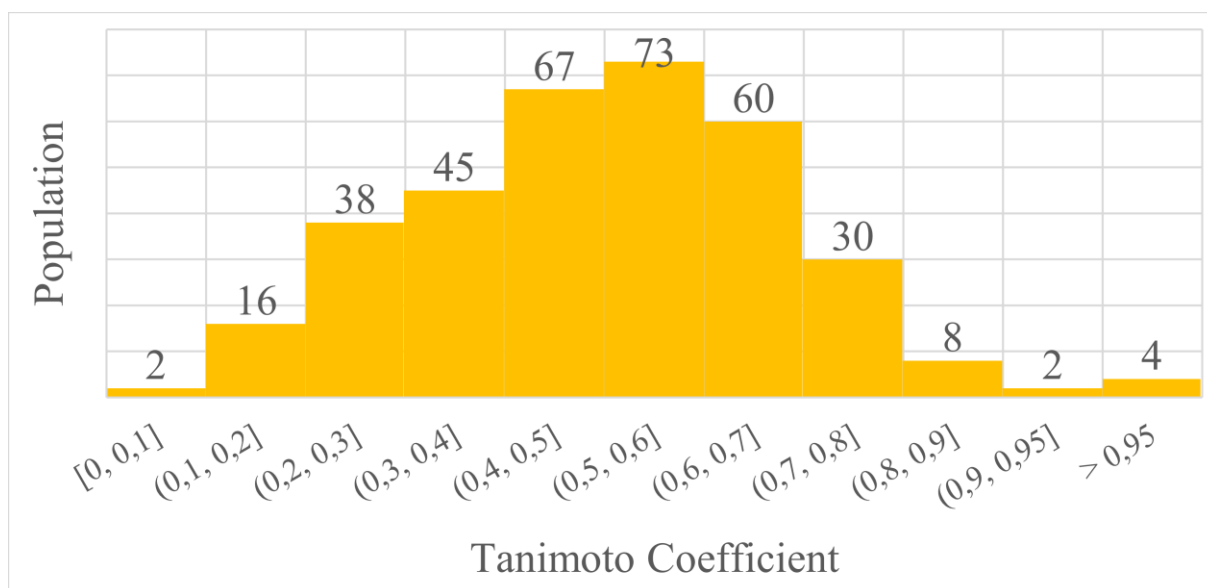


Figure 56. Distribution of Tanimoto coefficients for the 345 imperfect reconstructions.

Having had satisfying results with a small training database, an upscaling to a bigger database of 100.000 randomly selected compounds was used to train the model and 10.000 external compounds were used as a validation set. The model achieved a RMSE of around 2.4

for both sets during training and validation. The reconstruction rate analysis was performed on the 10.000 external compounds as shown in Figure 57.

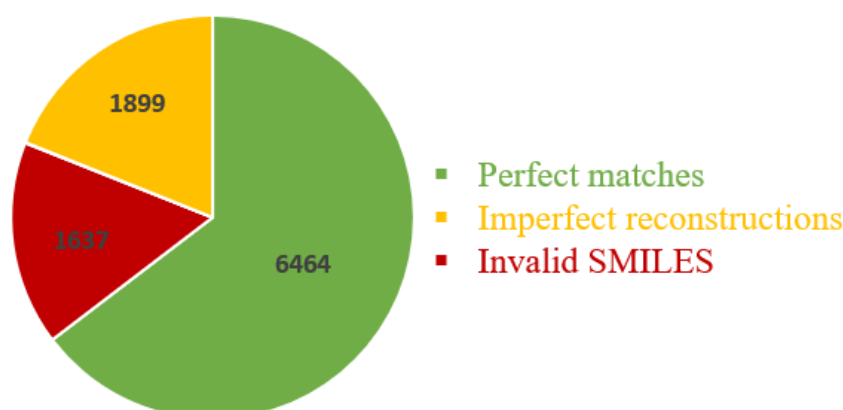


Figure 57. Proportion of perfect matches, imperfect reconstructions, and invalid SMILES for the best DBM found in the previous analysis.

Unfortunately, scaling up the databases by a factor of 10 reduced the perfect reconstruction rate to 64.64% which is 30% lower than the previous results. This may be due to overfitting or it could also be the result of a capacity issue for the architecture. It was able to accommodate 10.000 compounds previously, but was not able to encode 10 times more.

5.2.9 Parameters optimization – ISIDA vectors

Having gathered some preliminary results for the latent vectors, the focus was shifted to ISIDA descriptors to compare results. The method for finding the best parameters was slightly changed from a 2-step method to a 1-step method where both dimensions for hidden layer 1 and hidden layer 2 could vary. By using different methods of parameter optimization for ISIDA and latent vectors, the goal was to get a better understanding of the relation between the two hidden layers and the performance of the model. Setting each hidden layer's parameters sequentially could amount to minimizing a two-variable function by freezing one variable, minimizing the other, then doing the opposite. This is an easy solution but with no guarantee to end up in a minimum for the loss function. The parameters obtained this way may not be the most optimal. Therefore, both dimensions of the hidden layers were varied at the same time, effectively minimizing the loss function using the two variables.

Table 27. Results for the training of the DBM with 2 hidden layers on ISIDA vectors. All experiments were done for a visible dimension of 402. Parameters a and b control the decline of the learning rate, a gives the rate of decline and b controls the learning rate start. The lower a is, the faster the learning rate declines. For all experiments, A was set to 500, the batch size was set to 256, and the number of epochs was set to 2000. The best experiment is highlighted in green, the experiment used in the analysis is highlighted in blue.

Experiment Number	Hidden dimension 1	Hidden dimension 2	B	Start Learning Rate	Euclidian Distance
1	400	400	3.7	0.025	14.60
2	400	100	3.7	0.025	14.56
3	400	10	3.7	0.025	13.62
4	400	1000	3.7	0.025	14.56
5	400	2000	3.7	0.025	12.78
6	400	2000	3	0.05	14.54
7	400	2000	2.3	0.1	12.78
8	400	2000	1.61	0.2	11.96
9	500	2000	1.61	0.2	11.13
10	750	2000	1.61	0.2	9.62
11	1000	2000	1.61	0.2	8.61
12	1500	2000	1.61	0.2	7.33
13	2000	2000	1.61	0.2	6.53
14	2000	3000	1.61	0.2	6.44
15	2000	4000	1.61	0.2	6.41
16	2000	5000	1.61	0.2	6.43
17	3000	5000	1.61	0.2	5.43
18	4000	5000	1.61	0.2	5.04
19	5000	5000	1.61	0.2	4.85
20	6000	5000	1.61	0.2	4.76
21	7000	5000	1.61	0.2	4.65
22	8000	5000	1.61	0.2	4.53
23	9000	5000	1.61	0.2	4.49
24	10000	5000	1.61	0.2	4.38
25	15000	5000	1.61	0.2	4.01
26	17500	5000	1.61	0.2	3.98
27	20000	5000	1.61	0.2	4.03
28	17500	7500	1.61	0.2	13.11

The best RMSE obtained was for experiment 26. However, experiment 25 got very close results with 2500 less hidden units in the first hidden layer. This difference in number of hidden units also implies a difference in computing time. Thus, for timing purposes the fluctuation analysis was computed with the model of experiment 25.

The standard deviation analysis was performed on experiment 25 as explained above:

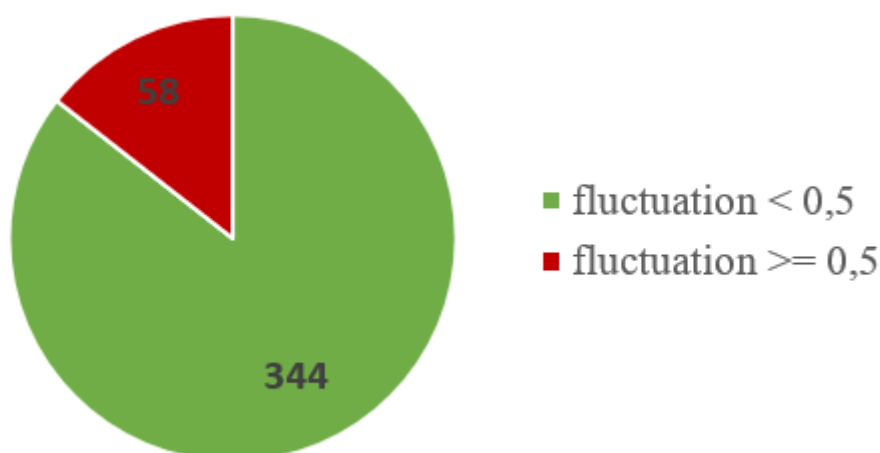


Figure 58. Proportion of descriptors according to their reconstruction fluctuation. 85.6% of descriptors have a possible variation of less than 0.5.

86% of the descriptors had a fluctuation less than 0.5, meaning that they were correctly reconstructed on the training dataset (Figure 58). The remaining 58 descriptors were above the fluctuation threshold, a large part having an error close to 1 but some of the descriptors had a fluctuation above 2 (Figure 59). This result was not better than our previous attempt. The combination of the errors from the latent model and the ISIDA models seriously hinder the ability to train a MDBM model.

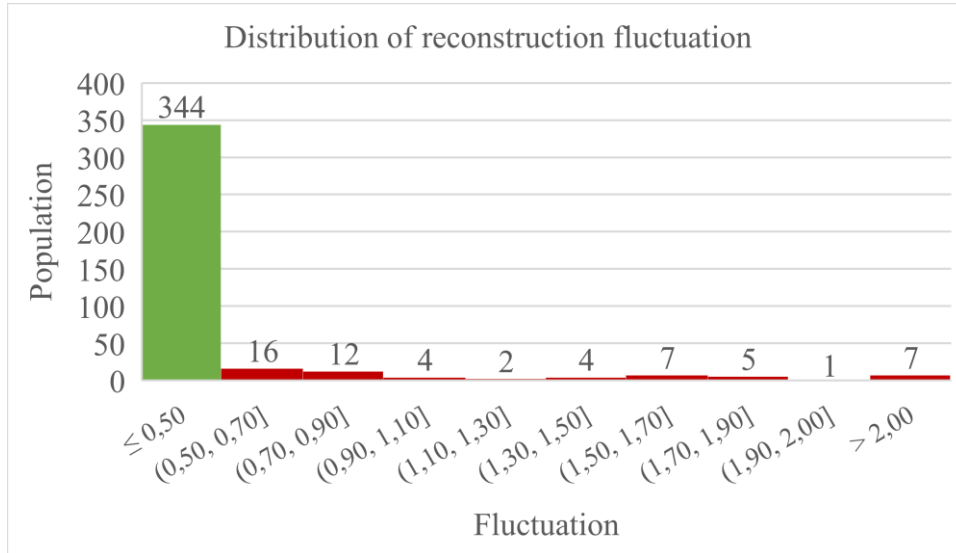


Figure 59. Histogram of the distribution of reconstruction fluctuation calculated between initial and reconstructed ISIDA vectors. Bars are coloured according to results of Figure 58.

In an attempt to improve the ISIDA DBM, a third layer was added to create a deeper DBM in hopes that the increased depth and higher representation would improve the model performance.

Table 28. Results for the training of the DBM with 3 hidden layers on ISIDA vectors. All experiments were done for a visible dimension of 402. A was set to 500, B to 1.61, the starting learning rate was 0.2. The number of epochs was set to 2000.

Experiment Number	Hidden dimension 1	Hidden dimension 2	Hidden dimension 3	RMSE
1	15000	5000	1000	111.45
2	15000	5000	2000	116.13
3	15000	5000	3000	108.08
4	500	500	500	18.97
5	300	300	300	14.85
6	100	100	100	17.13
7	100	100	1000	16.87

Initial experiments (1, 2 & 3, Table 28) with large hidden layers did not train well at all. In following experiments (4 to 7), the dimensions of all layers were drastically reduced. The thought process was that in a 2-hidden layer model, the lack of depth was compensated by the

large size of the hidden layers. Adding more depth meant that the layers' size could be reduced. Smaller layers resulted in a reduction of the RMSE by a factor of almost 10, unfortunately, these experiments still did not produce comparable results to the DBM with 2 hidden layers. The deeper DBM did not improve on the 2-hidden layer DBM, therefore the previous architecture was used to train a model on 100.000 training vectors and 10.000 validation vectors.

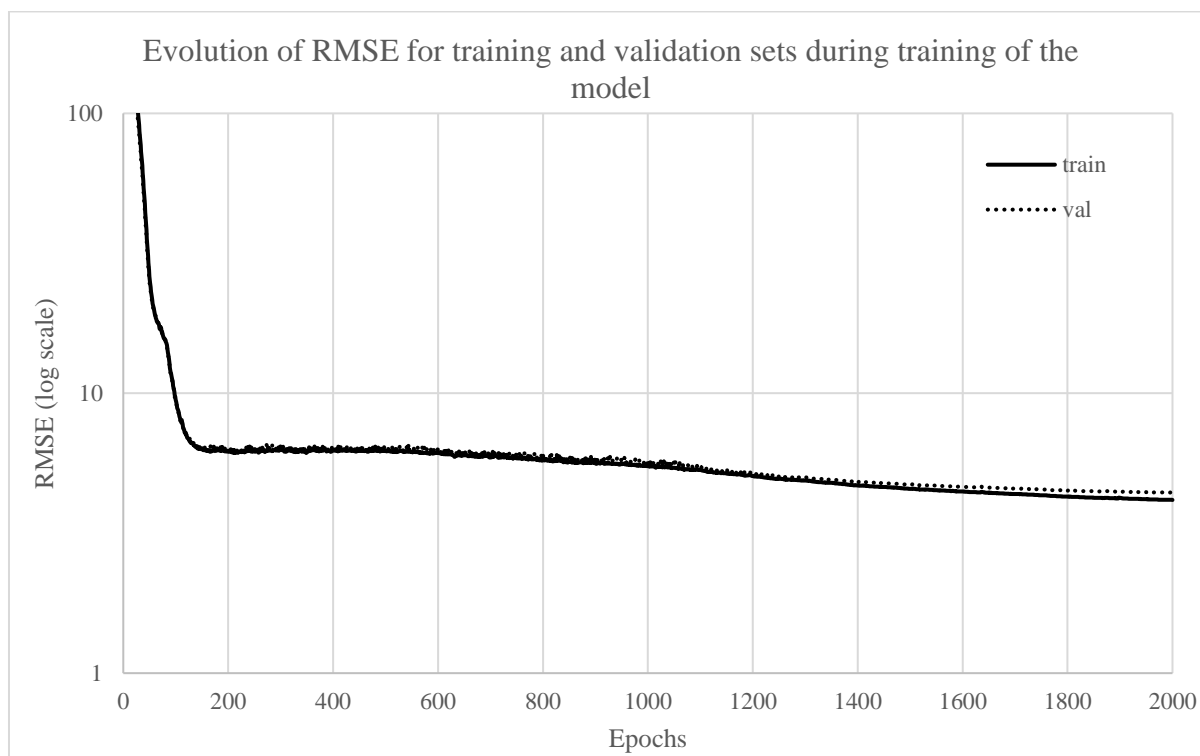


Figure 60. Evolution of RMSE for training and validation sets during training of the model.

No sign of overfitting can be observed on Figure 60. Both training and validation sets achieve comparable RMSE values.

Table 29. Best RMSE achieved during training for training set and validation set.

Best training loss achieved	Best validation loss achieved
4.16	4.42

The model used for this phase (experiment 25) achieved a RMSE of 4.01 on the training set with 10.000 compounds. Here, with 10x more data, it achieved 4.16 on training and 4.42 on validation (Table 28), which is very close to the preliminary results.

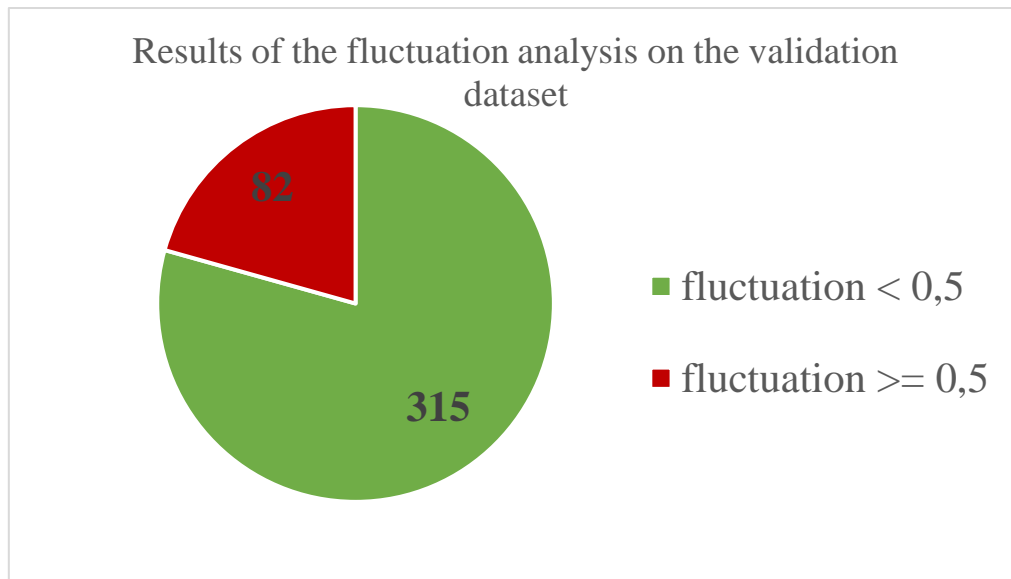


Figure 61. Proportion of descriptors according to their reconstruction fluctuation on the validation dataset.

79% of the descriptors are correctly reconstructed (Figure 61) which is a 7% decrease compared to the preliminary results on the small training set. 82 descriptors have a fluctuation of more than 0.5, 22 of them with 2 or more which is a slight increase compared to the 12 before. (Figure 62).

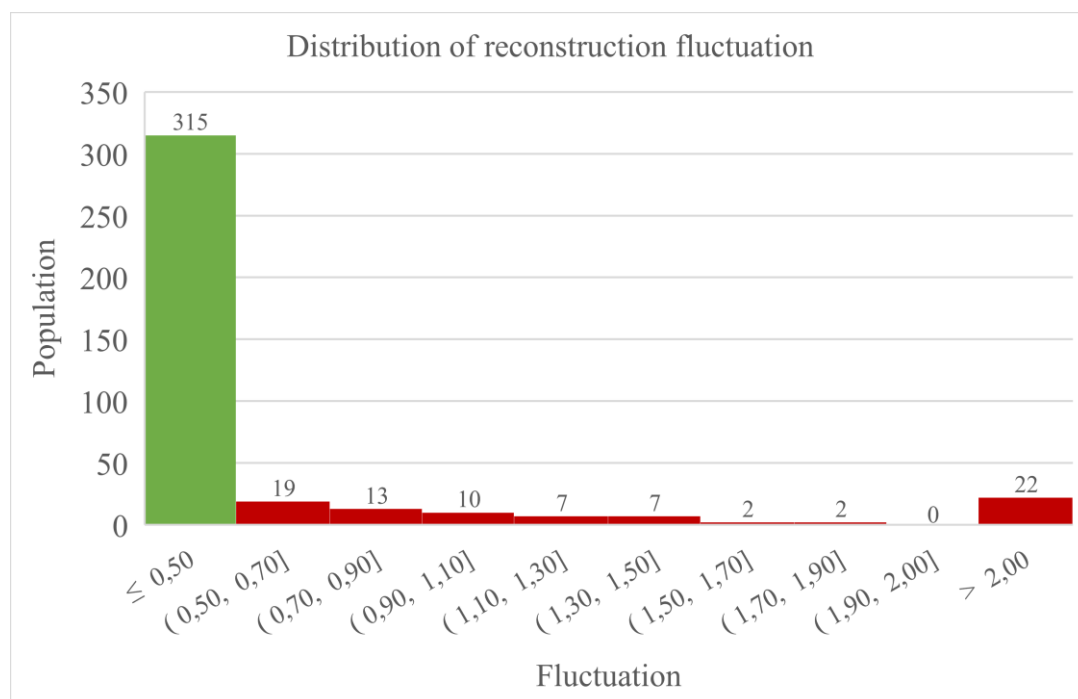


Figure 62. Histogram of the distribution of reconstruction fluctuation calculated between initial and reconstructed ISIDA vectors. Bars are colored according to results of Figure 61.

A quick comparison between initial and reconstructed ISIDA vectors of the validation set showed that *no vectors* was perfectly which amounts to a 0% reconstruction rate. This is expected since about 20% of the descriptors had a general fluctuation above the threshold and the perfect reconstruction implies 397 simultaneous successful predictions – which is very unlikely. To have better insight we computed each descriptor's "occurrence" over the validation dataset. Occurrence is a measurement of how much the descriptor is used in a dataset. It is computed as follows:

$$\text{occurrence} = \frac{\text{number of times descriptor is not 0 in data}}{\text{length of dataset}}$$

For example, if the occurrence of a descriptor is 0.5 that means that the fragment is found in 50% of the molecules in the dataset. The occurrence of each descriptor was plotted against its fluctuation, and each descriptor was coloured according to its standard deviation. The results are shown in Figure 63.

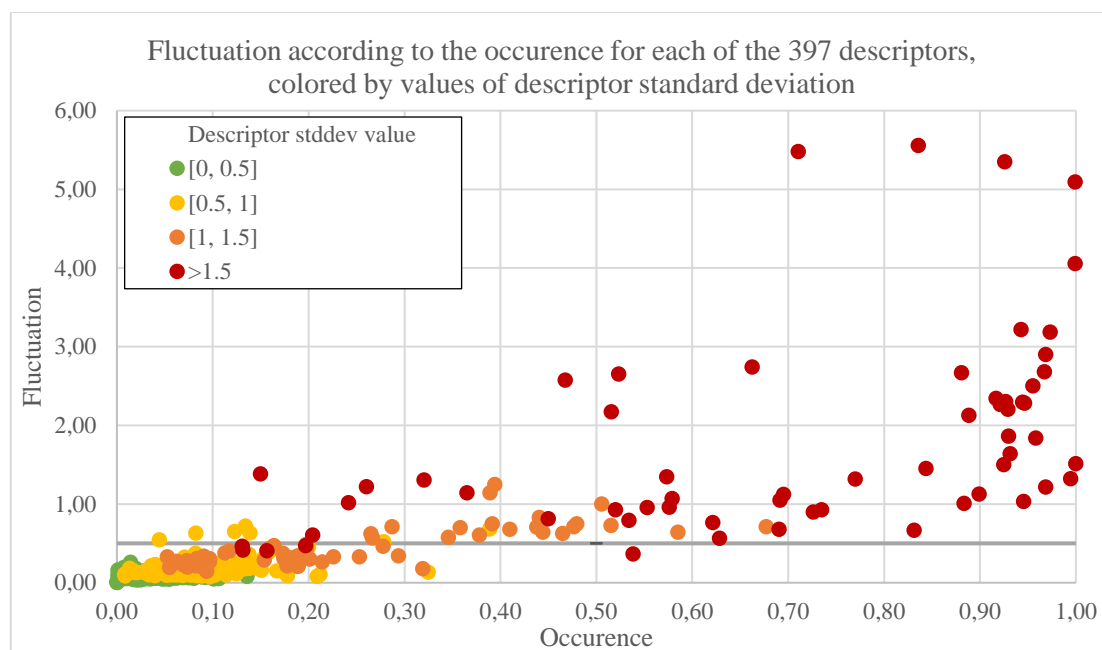


Figure 63. Fluctuation of each descriptor according to its occurrence in the validation dataset. The grey line represents the 0.5 fluctuation limit. Each dot is colored according to the standard deviation of the descriptor itself. The grey line indicates the threshold of fluctuation = 0.5.

There appears to be a correlation between the occurrence of a descriptor in the dataset, its fluctuation, and its standard deviation. Descriptors with the biggest occurrences are the most badly predicted by the model, probably because of their high variation as indicated by the high values for standard deviation. Since those descriptors are not correctly predicted it is highly

unlikely that a perfect reconstruction will ever occur, explaining the poor performances in terms of reconstruction rate. This phenomenon is highly reminiscent of the issues caused by the variation of ISIDA vectors in the SMI2ISIDA project where the variation and tendencies of the most frequent fragments were unable to be captured and reproduced by neural network. It seems difficult for these two architectures to link ISIDA descriptors to a different representation, SMILES in the case of SMI2ISIDA or an abstract numerical vector in the case of this project.

A last-ditch effort to improve model performance included adding one more layer which was unsuccessful and upgrading the optimizer of the model from Adam to AdaBelief. The AdaBelief Optimizer is an improvement over the Adam Optimizer which is widely used in many different types of Deep Learning models. The Adam Optimizer improves on the learning by calculating moments which considers not only the current gradients but also the past gradients. This method helps the model avoid local minima in search of the global minimum. Gradients for the standard deviations, hidden biases, visible biases and weights were calculated using this method instead of the simple gradient descent. The model was trained on the small training set and compared to the best model trained with the Adam optimizer, also on the small training set.

Table 30. Parameters for the best model found in the initial analysis.

Hidden Dimension 1	Hidden Dimension 2	Number of epochs	Batch size	A	B	Start Learning rate
15000	5000	5000	256	500	1.61	0.2

Parameters in Table 30 were found to be the best for the initial analysis and were therefore reused to train the AdaBelief model. Without AdaBelief, the model achieved a Euclidian distance of 4.01 on the training set. The addition of AdaBelief on the same network with the same parameters lowered the final Euclidian distance to 2.22 (Table 31).

Table 31. RMSE at the end of training with and without the AdaBelief Optimizer for a model with the same parameters on the small training dataset.

With AdaBelief	Without AdaBelief
2.22	4.01

A fluctuation analysis was performed on the AdaBelief model and results were compared to the initial model (Figure 64).

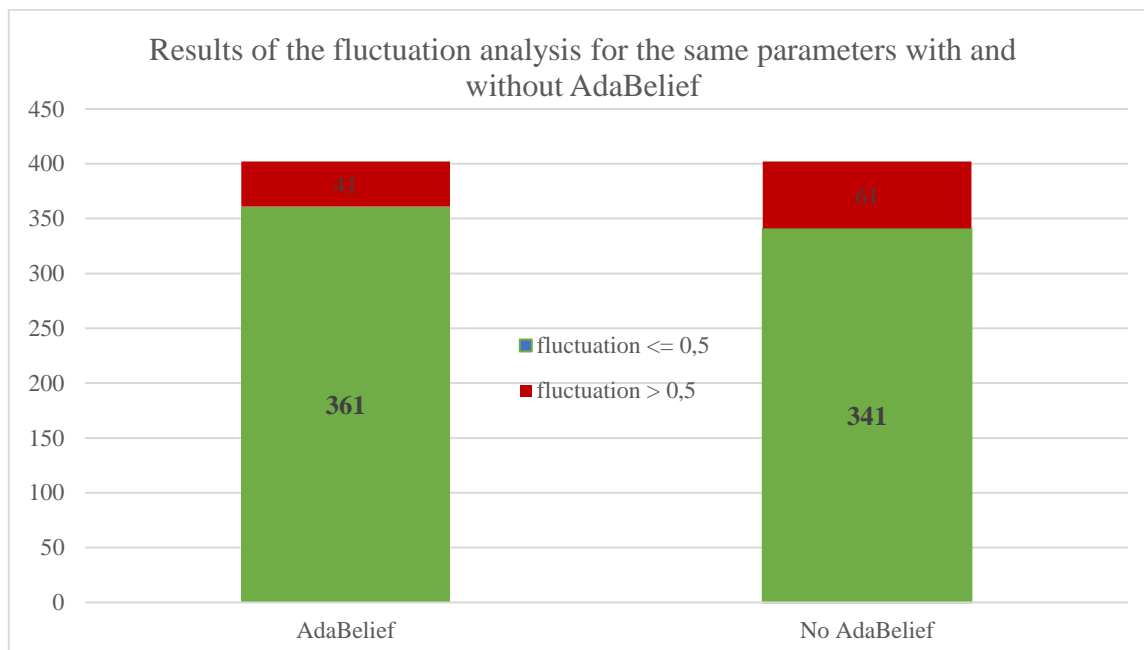


Figure 64. Proportion of results for a model with the same parameters with AdaBelief (left) and without AdaBelief (right).

The addition of the AdaBelief optimizer improved the number of correctly reconstructed descriptors from 85% to 90%, putting 20 more descriptors under the fluctuation threshold. However, when fluctuation is plotted against occurrence (Figure 65), similar problems appear, mainly that the most popular and important descriptors are miscounted.

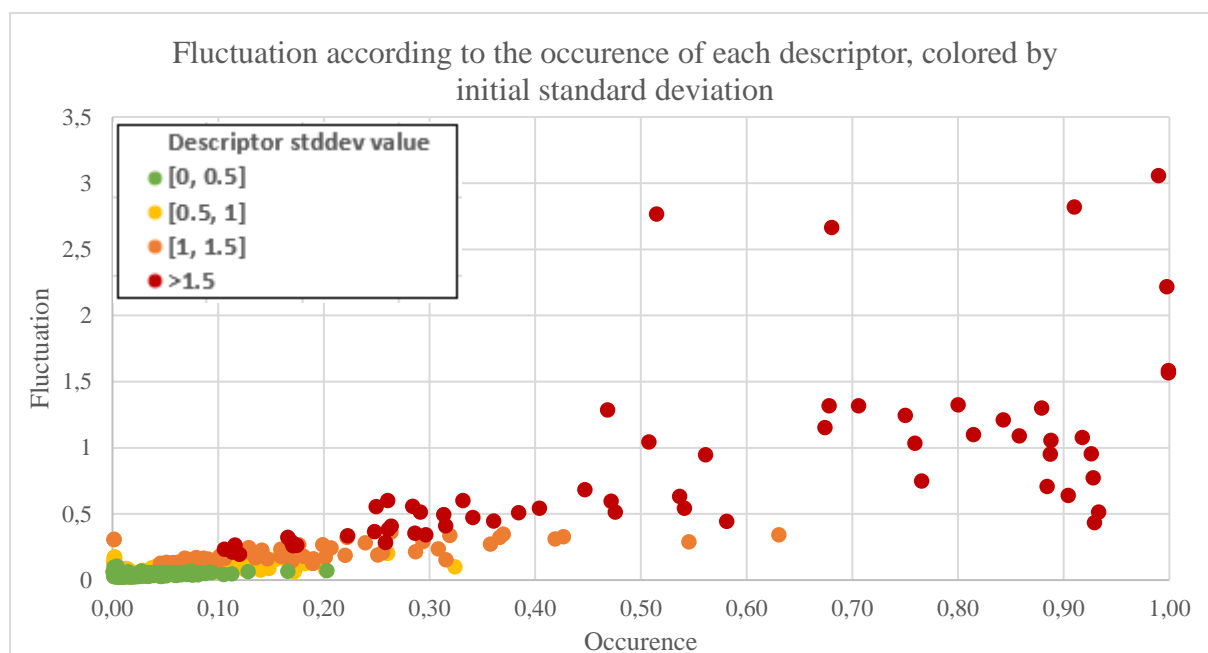


Figure 65. Fluctuation of each descriptor according to its occurrence in the training dataset. Each dot is colored according to the standard deviation of the descriptor itself.

5.2.10 Conclusion

Neither of the separate models could reproduce their input with enough precision to be used in a working MDBM. Latent vectors could potentially be optimized to obtain satisfactory results, or at least increase the reconstruction rate with more time. It could have been beneficial to work with a principal component transform of ISIDA descriptors vectors and the AE latent vectors in order for the covariance matrix in equation (5.3) to better describe the distribution of the input. Besides, ISIDA descriptors are counts, so a standard DBM based on integer could have been attempted, for instance, by mapping the counts binary vectors. Defining more relevant metrics and loss function to train these architectures in the context of generating molecular structures could also be explored. Structural descriptors in general seem to be problematic for neural networks to correctly handle. Methods based on manipulating them by having to reconstruct or predict them like SMI2ISIDA and MDBM remain unsuccessful despite many attempts.

5.3 Stargate-GTM

5.3.1 Introduction

Stargate-GTM (S-GTM) is a tool based on Generative Topographic Mapping that allows two different descriptor spaces to be connected. On the premise that the same data points are present in both spaces, two manifolds can be trained simultaneously each of them satisfying topological constraints from both datasets. The mapping of the two spaces is done by using the GTM manifold of map 1 with patterns on map 2 and reversely. This procedure actually emphasizes the consistency between the two data spaces which respective GTM are co-trained. Here, one of the data space is the ISIDA descriptors vector space and the second it the AE latent space – that can be readily decoded as SMILES strings. In this way, a compound represented by an ISIDA descriptors vector is represented by the ISIDA-space GTM responsibilities. These responsibilities are decoded using the manifold of the AE-space GTM in AE latent space vector. These latent vectors would then be fed to a generative model to create compounds localized in active areas of ISIDA space.

5.3.2 Methodology

Stargate-GTM

Stargate-GTM builds a model using two initial spaces instead of one like in the conventional GTM. Two manifolds are fitted in the two different spaces and the individual probability distributions are combined to obtain a joint probability distribution. The manifolds are constructed so that each node in the 2D latent space is associated with the RBFs of both manifolds.

During training, manifolds are optimized together using joint responsibilities. These are obtained from the individual probability distributions for Space 1 and Space 2 respectively: $p(\mathbf{t}_n^{Space1} | \mathbf{x}_k, \mathbf{W}^{Space1}, \beta^{Space1})$ and $p(\mathbf{t}_n^{Space2} | \mathbf{x}_k, \mathbf{W}^{Space2}, \beta^{Space2})$ computed using the two mapping functions from the manifolds \mathbf{Y}^{Space1} and \mathbf{Y}^{Space2} . In the

same way as regular GTM, individual responsibilities are initially computed during the expectation step using the following equations:

$$r_{kn}^{Space1} = p(\mathbf{x}_k | \mathbf{t}_n^{Space1}, \mathbf{W}^{Space1}, \beta^{Space1}) \quad (5.16)$$

$$r_{kn}^{Space1} = \frac{p(\mathbf{t}_n^{Space1} | \mathbf{x}_k, \mathbf{W}^{Space1}, \beta^{Space1})}{\sum_{k'}^K p(\mathbf{t}_n^{Space1} | \mathbf{x}_{k'}, \mathbf{W}^{Space1}, \beta^{Space1})} \quad (5.17)$$

$$r_{kn}^{Space2} = p(\mathbf{x}_k | \mathbf{t}_n^{Space2}, \mathbf{W}^{Space2}, \beta^{Space2}) \quad (5.18)$$

$$r_{kn}^{Space2} = \frac{p(\mathbf{t}_n^{Space2} | \mathbf{x}_k, \mathbf{W}^{Space2}, \beta^{Space2})}{\sum_{k'}^K p(\mathbf{t}_n^{Space2} | \mathbf{x}_{k'}, \mathbf{W}^{Space2}, \beta^{Space2})} \quad (5.19)$$

Combined responsibilities R_{kn} are then computed as follows:

$$R_{kn} = \frac{p(\mathbf{t}_n^{Space1} | \mathbf{x}_k, \mathbf{W}^{Space1}, \beta^{Space1})^{w^{Space1}} * p(\mathbf{t}_n^{Space2} | \mathbf{x}_k, \mathbf{W}^{Space2}, \beta^{Space2})^{w^{Space2}}}{\sum_{k'} p(\mathbf{t}_n^{Space1} | \mathbf{x}_{k'}, \mathbf{W}^{Space1}, \beta^{Space1})^{w^{Space1}} * p(\mathbf{t}_n^{Space2} | \mathbf{x}_{k'}, \mathbf{W}^{Space2}, \beta^{Space2})^{w^{Space2}}} \quad (5.20)$$

w^{Space1} and w^{Space2} are user-defined weight parameters governing the importance of each probability distribution. They are real values ranging from 0 to 1 and their combined values always equal to 1 so that: $w^{Space2} = 1 - w^{Space1}$. The shapes of the manifold are adjusted until convergence similarly to a simple GTM.

Data Preparation

5000 compounds were randomly selected from ChEMBL23 and encoded into their corresponding ISIDA vectors (sequences of 2 to 7 atoms, I-A—2-7) and latent vectors. The autoencoder model used to generate the latent vectors was the same that was previously used in the MDBM project. Descriptors were filtered according to standard deviation (2% of max). This resulted in 421 remaining ISIDA descriptors (out of 6520 initially) and 133 remaining latent descriptors (out of 256 initially). Both these datasets served as Stargate’s framesets.

Stargate-GTM Training

As previously explained, one of the important parameters for this analysis are the user-defined weights w^{ISIDA} and w^{LATENT} . Depending on these values, the impact of one of the two spaces can be more important than the other. The most natural idea would be to give equal importance to both data spaces with $w^{ISIDA} = w^{LATENT} = 0.5$. However, there is no indication that this would ensure good results, or that other combinations may not perform better. Therefore, 19 different models were trained, in which only the weights parameters fluctuated while other parameters were set to values known for ensuring a viable training process. The values for w^{ISIDA} and w^{LATENT} in each experiment, and other fixed parameters are reported in Table 32.

Table 32. Weights distribution for all experiments with other fixed GTM parameters.

Exp. N°	w^{ISIDA}/w^{LATENT}	Exp. N°	w^{ISIDA}/w^{LATENT}	Exp. N°	w^{ISIDA}/w^{LATENT}
1	0.05 / 0.95	8	0.40 / 0.60	14	0.70 / 0.30
2	0.10 / 0.90	9	0.45 / 0.55	15	0.75 / 0.25
3	0.15 / 0.85	10	0.50 / 0.50	16	0.80 / 0.20
4	0.20 / 0.80	11	0.55 / 0.45	17	0.85 / 0.15
5	0.25 / 0.75	12	0.60 / 0.40	18	0.90 / 0.10
6	0.30 / 0.70	13	0.65 / 0.35	19	0.95 / 0.05
7	0.35 / 0.65				
Number of RBFs		Number of Nodes		RBF width	
225		1600		0.5	
				Regularization	
				0.63	

After training, the manifolds were resampled using the GTM ReSample tool to a size of 625 nodes since 1600 nodes was an unnecessary large number for displaying such small spaces. During resampling, the training data was projected on the manifold and 2D map coordinates were calculated. Finally, density landscapes for both manifolds were created using the resampled manifold and the training data.

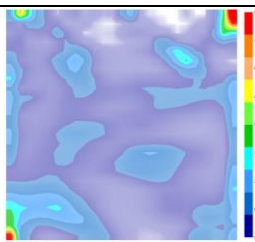
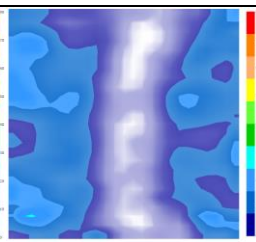
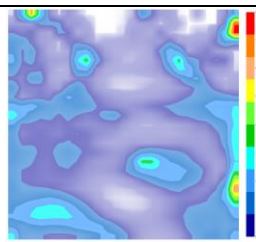
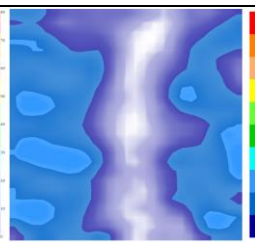
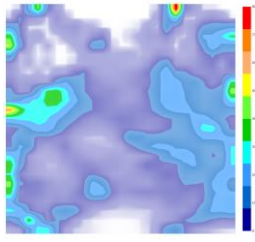
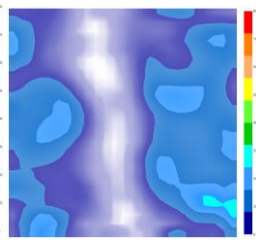
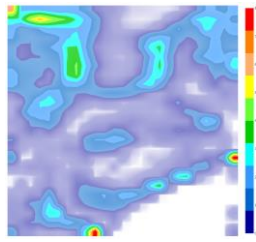
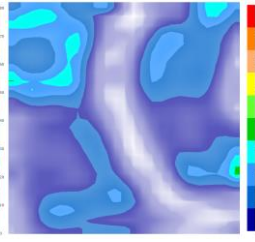
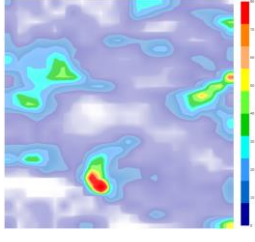
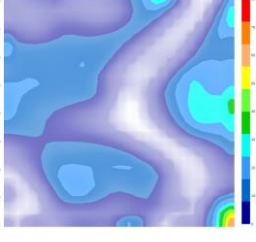
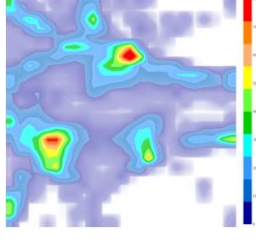
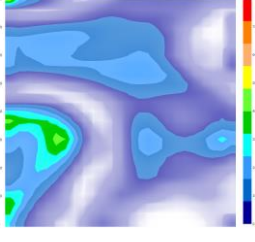
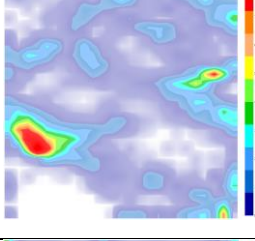
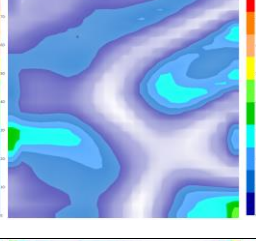
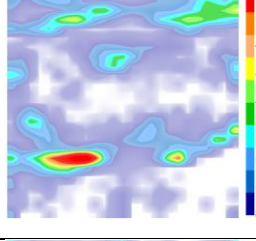
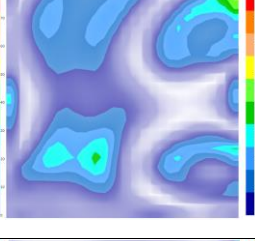
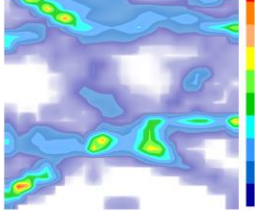
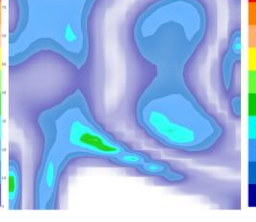
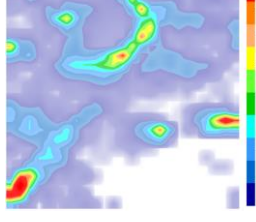
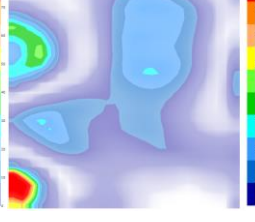
Hilbert-Schmidt Independence Criterion

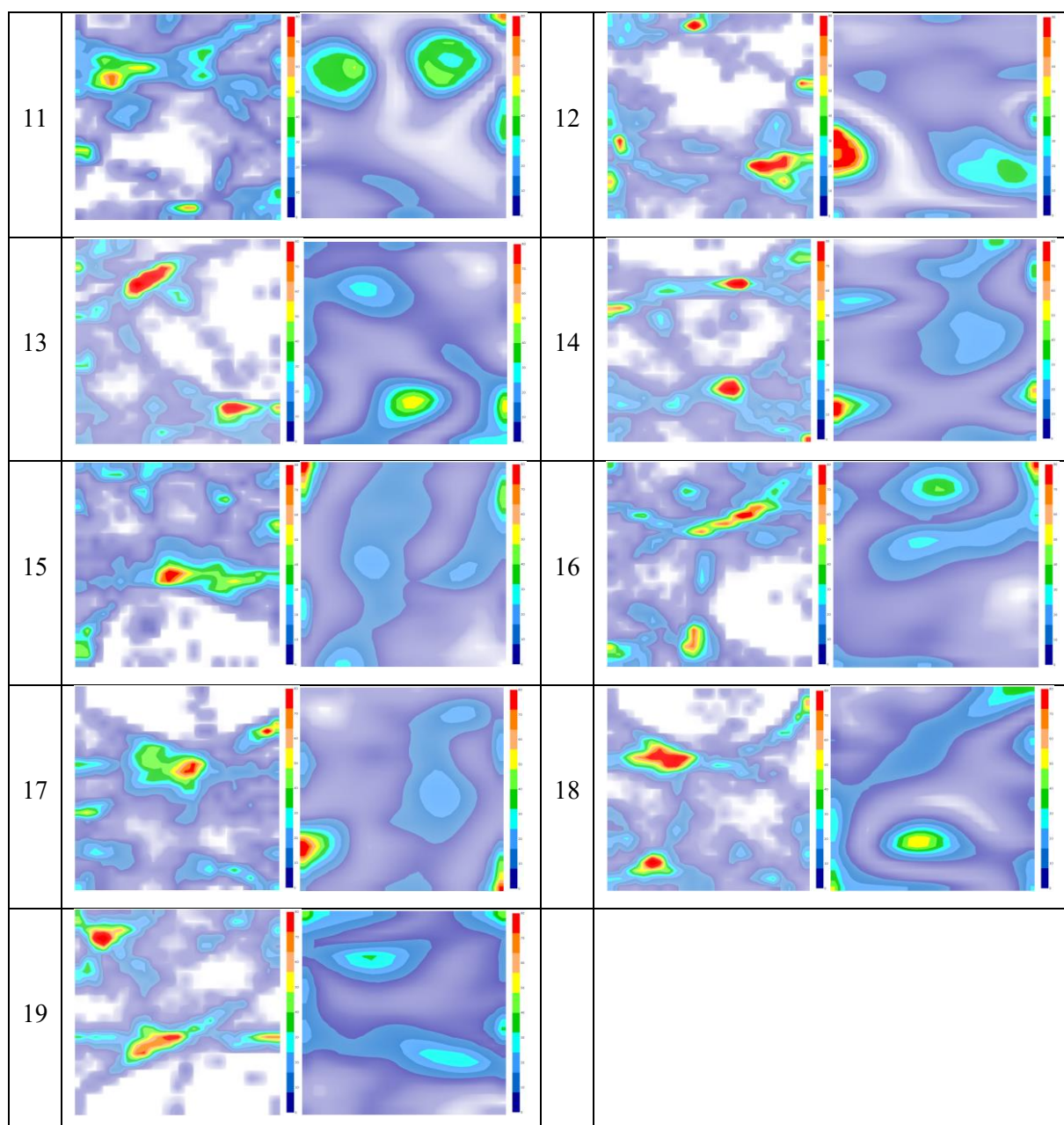
The Hilbert Schmidt Independence Criterion^[163] (HSIC) is a value used to measure the independence between two multivariate distributions expressing different modalities. In such situation, only the kernels, measuring the similarity between instances sampled from each distribution can be compared. This allows to account for potential non-linear dependence between the tested distributions. Simply put, the output of the calculation will tend towards 0 if the two spaces are independent, and 1 if they are statistically dependent.

5.3.3 Results

Each couple of datasets was projected on its corresponding manifolds and the resulting density landscapes are regrouped and compared per experiment in Table 33.

Table 33. Comparison between density landscapes for ISIDA (left) and latent (right) datasets for each experiment. Only the 5000 training compounds were projected. All density scales are set to the same values and range from 0 (dark blue/white) to 80 (red).

E	ISIDA	LATENT	E	ISIDA	LATENT
1			2		
3			4		
5			6		
7			8		
9			10		



The comparison of density landscapes for all experiments shows several tendencies. From one map to the other, it is often possible to recognise patterns that are smoothly modified as with weight parameter value. The chemical content of these patterns is stable: the same molecules are found in the map in a consistent manner from one map to the other, with a small change of the weight value.

Besides, the effect of the co-training is visible, as pattern structures from the ISIDA maps can be retrieved, in an altered version in the AE maps and reversely. This is most visible

between the experiment 3 and 17, with a weight parameter with all values of the weight parameter in the range [0.15, 0.75].

ISIDA landscapes often present areas of high density (80+ red zones) scattered around the map while those high-density zones are a lot rarer in latent landscapes and may only be observed in experiments 10, 12 and 14. This may indicate that AE latent vectors tend to be more spread out in terms of probability distribution compared to ISIDA vectors. White zones corresponding to empty areas of chemical space are more common and well defined in ISIDA landscapes, compared to the AE latent density landscapes.

These differences in density distribution mean that, visually, ISIDA and latent landscapes in each experiment do not share obvious similar features. In a more ideal situation, when the two spaces represented are topologically similar, the Stargate-GTM densities can look very much alike. To illustrate it, two example datasets from ChEMBL were used containing the same 1263 compounds with two different sets of ISIDA descriptors (dimensionality 280 and 637). The comparison of the resulting landscapes is shown in Figure 66.

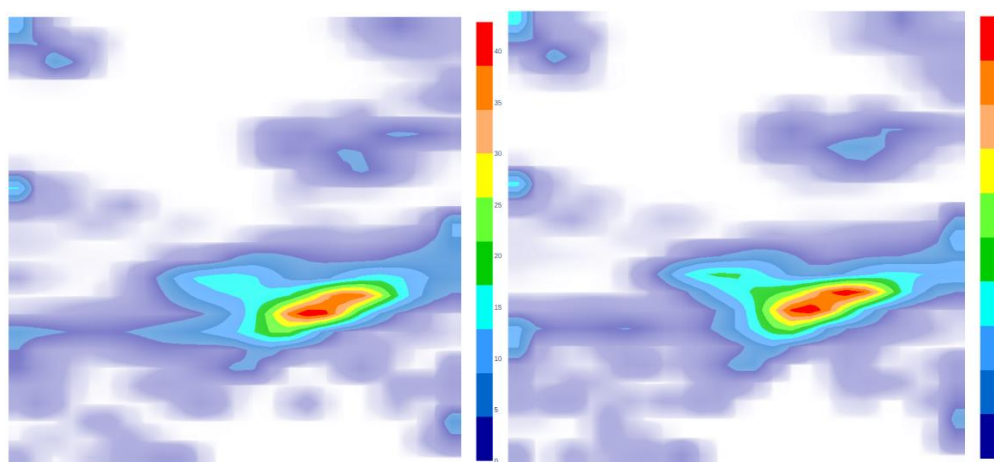


Figure 66. Density landscapes created using StargateGTM for the example datasets. Both density scales are the same and range from 0 (dark blue/white) to 45 (red).

The comparison of the two example datasets shows that both density distributions are almost the same. All empty areas and low-density areas are replicated in both landscapes and

the high-density area has the same shape and density value in both maps. In comparison to this example, the results obtained with ISIDA, and latent vectors are visually disappointing.

In order to map one space to the other, it is expected that the responsibilities from one space can be decoded using the manifold of the other space. Hence, the localisation of a compound in one map, should correspond to the location of related compounds in the second map.

As an initial experiment to verify this, 9 random molecules were selected and their 2D coordinates in the ISIDA and latent maps from Experiment 3 were extracted. Experiment 3 was chosen because it seemed to be the most similar in terms of visual comparison. The same process was done for the example datasets combining two ISIDA descriptors sets. Results are shown in Figure 67.

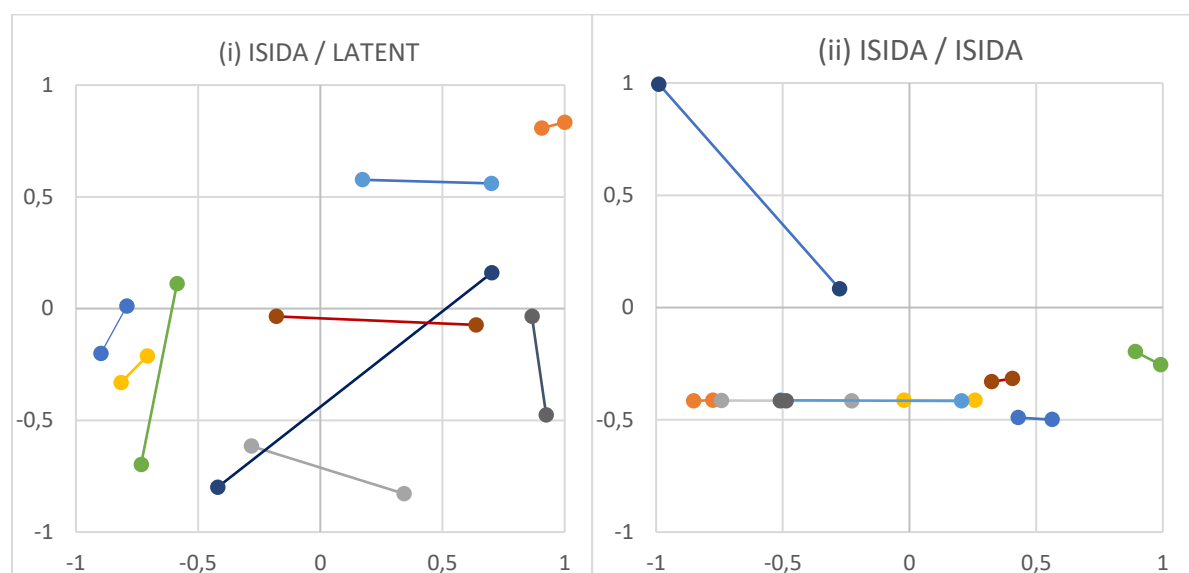


Figure 67. Positions of 9 randomly selected compounds for the ISIDA / LATENT S-GTM (i) and ISIDA / ISIDA S-GTM (ii). Each molecule has 2 points of the same colour corresponding to their projection in the two spaces (ISIDA/LATENT and ISIDA/ISIDA), which are linked for visualization. The graphs are squares representing the map area. Coordinates range from -1 to 1 on both axes.

Most compounds extracted from the ISIDA/LATENT S-GTM have highly different positions on the map except for arguably three (orange, yellow and dark blue). On the other hand, only 2 of the randomly selected compounds have different positions on the ISIDA/ISIDA S-GTM maps. This emphasizes that compounds do not share close positions in the ISIDA/LATENT S-GTM.

This observation is systematically studied in Figure 68. The Euclidean distance $d = \sqrt{(x_{latent} - x_{isida})^2 + (y_{latent} - y_{isida})^2}$ between the position of a molecule on one map and the position of the same molecule on the other map is calculated for all compounds in the dataset. The distribution of the distances is reported for each pairs of maps obtained using the various values of the weight parameter. For comparison, the distance for the ISIDA/ISIDA related maps and when association the coordinate of a compound on the ISIDA map to a random compound position in the AE map.

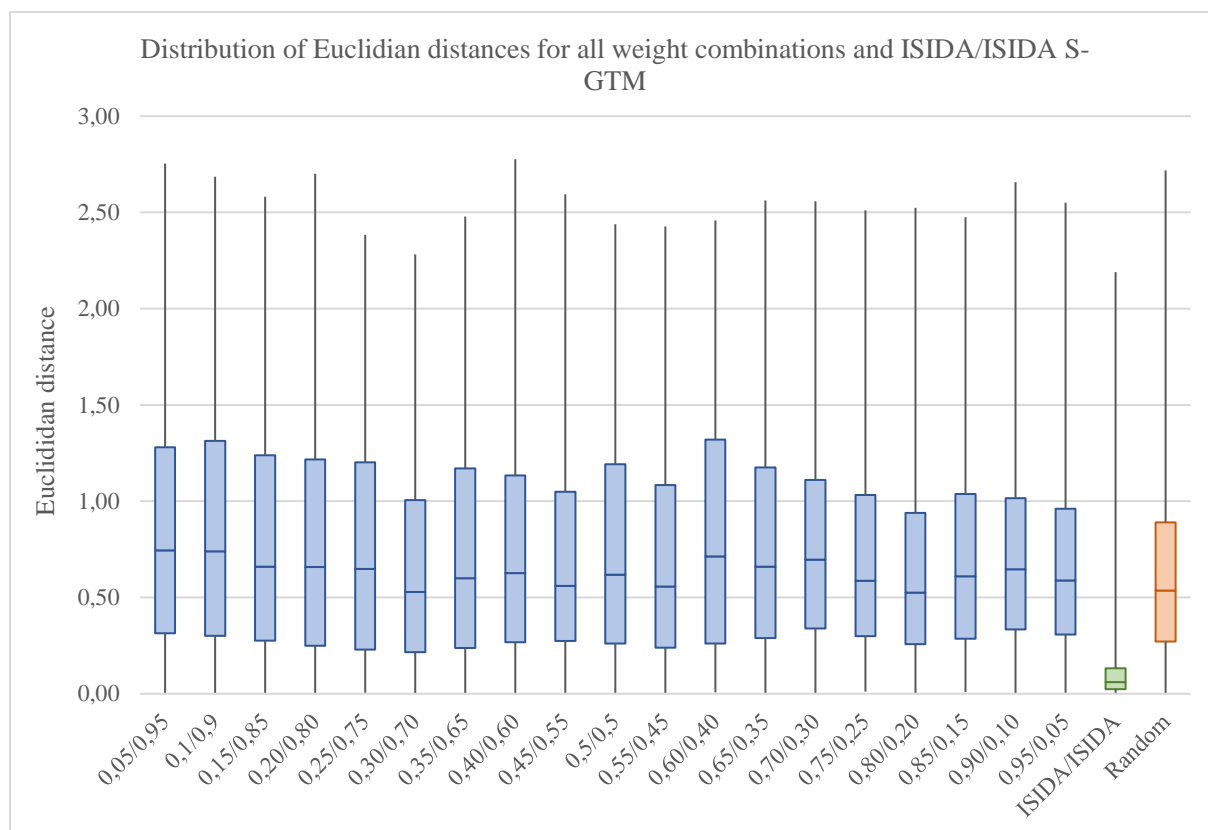


Figure 68. Box plots generated for the distribution of Euclidian distances between the coordinates of the same compounds on the two maps of S-GTM. Blue box plots correspond to ISIDA/LATENT S-GTMs with different combinations of weight parameters (0.05/0.95 means that $w^{ISIDA} = 0.05$ and $w^{LATENT} = 0.95$). The green box plot shows the results for the ISIDA/ISIDA distances, and the orange box plot shows the results for the randomized ISIDA-ISIDA experiment.

Figure 68 shows that in terms of Euclidean distances between the same compounds, ISIDA/LATENT S-GTMs perform much worse than the example ISIDA/ISIDA model and are comparable to a random situation. A compound in the ISIDA map is usually localized on the map: the responsibilities are concentrated on a small number of nodes. The AE maps encode the molecules in responsibility patterns that cover large portions of the map. This reflects the

differences in the dimensions of the two chemical spaces and the significant differences between the two datasets distributions of pairwise distances that can hardly be reconciliated. Thus, the comparison of the projections of the compounds on the maps is not very relevant and can be misleading. Yet, it reflects strikingly the low correlation between the AE descriptors space and the ISIDA descriptors space. The fragment-oriented interpretation through ISIDA descriptors (or fragment descriptors in general) is very different from the interpretation a Seq2Seq architecture makes of a SMILES string. These two different “interpretations” are more radically different than the difference between for example, sequence-based fragments or centroid-based fragments since they both use molecular fragments. The complexity of the mathematical equations governing the calculations of latent vectors makes it extremely difficult to understand exactly in essence what information is stored inside, however the results of this analysis suggest that they may not describe chemical space in the same manner as a classical fragment-based approach.

Hilbert Schmidt Independence Criterion

To confirm that hypothesis and have a better understanding of the issue, the “compatibility” or dependence of the two descriptor spaces should be measured. For Stargate-GTM to be more relevant, the same similarity principles should apply in both chemical spaces. If this is not the case, then the basic construction of the spaces are so different that linking the two hardly seems achievable. The normalized HSIC was calculated for the ISIDA/LATENT datasets and the ISIDA/ISIDA datasets and are reported in Table 34 using a cosine kernel.

Table 34. Results for the HSIC calculations for both datasets in their respective spaces

ISIDA/LATENT	ISIDA/ISIDA
0.159	0.642

The HSIC value between the ISIDA/LATENT descriptors is quite low compared to the ISIDA/ISIDA value. This means that ISIDA vectors and LATENT vectors are almost completely independent. Since different principles are applied when constructing the respective chemical spaces, their organization is completely different.

For the sake of comparison, the HSIC values (the “compatibility” of descriptor spaces) were calculated among different ISIDA descriptors and the AE latent descriptors. Results are shown in Table 35.

Table 35. Values of HSIC calculated among a set of simple ISIDA descriptors and the latent space. IA(2- n) means sequences of atoms of length 2 to n . IAB(2- m) means atom-centered fragments with a radius of 2 to m atoms.

	Latent	IA(2-2)	IA(2-3)	IA(2-4)	IA(2-5)	IA(2-6)	IA(2-7)	IAB(2-2)	IAB(2-3)	IAB(2-4)
Latent	1									
IA(2-2)	0,102	1								
IA(2-3)	0,106	0,970	1							
IA(2-4)	0,099	0,926	0,980	1						
IA(2-5)	0,099	0,889	0,889	0,989	1					
IA(2-6)	0,098	0,859	0,859	0,968	0,993	1				
IA(2-7)	0,096	0,838	0,838	0,950	0,981	0,996	1			
IAB(2-2)	0,074	0,799	0,799	0,798	0,781	0,762	0,747	1		
IAB(2-3)	0,057	0,364	0,364	0,428	0,457	0,467	0,468	0,591	1	
IAB(2-4)	0,06	0,153	0,153	0,201	0,233	0,254	0,267	0,288	0,662	1

All sequence-based chemical spaces have highly correlated descriptor spaces which is expected since the smaller descriptors spaces are contained in the bigger ones so that $IA(2-2) \in IA(2-3) \in IA(2-4) \in \dots \in IA(2-7)$ and the combinations of smaller fragments can manage to describe bigger fragments. Interestingly, atom-centered fragments of length 2 share very high HSIC values with both longer atom-centered fragments and sequence-based fragments. Due to the short nature of these descriptors, they share many fragments with sequence-based descriptors which is not the case for longer atom-centered fragments. This also explains the rather low HSIC values between IAB(2-2) and other IAB descriptors (0,591 and 0,288 for IAB(2-3) and IAB(2-4) respectively). More importantly, we see that latent vectors have very low HSIC values with both sequence-based and atom-centered descriptors. This confirms that the space of latent vectors is not constructed in the same way at all compared to fragment-based descriptors. Interestingly however, the HSIC values are slightly higher between latent vectors and sequence-based descriptors (around 0,1 for all instances of sequence-based fragments compared to 0,06-0,07 for atom-centered fragments). This difference, although very slight makes sense in the context of the interpretation of sequences of characters by a neural network. In the case of a carbon chain for example, the sequences of atoms or sequences of characters would describe the same molecule. The difference then would come from the interpretation of ramifications and cycles in the molecule.

5.3.4 Conclusion

The application of Stargate-GTM was not sufficient to create a link between the latent space of an autoencoder and a chemical space based on fragment-based structural descriptors. The basic principle of the method, which is that the same molecule should have comparable responsibilities on both maps was not observed. Although the projections of the compounds on both maps differ, it is possible that the responsibilities of the AE map could overlap with the responsibilities of the ISIDA. This analysis is left for future work. Yet, this means that an ISIDA vector would be translated in a complicated responsibility pattern in the AE space, that could be translated in a potentially diverse set of chemical structures resulting in a loss of the control of the generated chemical structures to sample a desired region of the chemical space.

These results did however confirm that the interpretation of chemical structures through artificial neural networks and molecular descriptors is completely different and leads to completely different chemical spaces that follow different principles and neighbourhood behaviours. These different organizations suggest an explanation of the difficulties met so far while designing models able to generate chemical structures corresponding to a given molecular descriptor vector.

5.4 Combination of ISIDA landscapes

5.4.1 Introduction

Since autoencoder latent space and ISIDA descriptor space behave technically like two independent multidimensional variables, method seeking to find a correlation or dependence between the two cannot be applied. This makes the task of generating compounds with selected properties and structural features through neural networks and cartography challenging. “Simple” solutions like an artificial neural network or Stargate-GTM could not be used for this application. For this reason it has been proposed to relax the constraints in the definition of the targeted region of an ISIDA chemical space to be sampled. This definition is based on Generative Topographic Mapping, and uses a combination of several ISIDA landscapes to create a “query” vector containing desirable properties, which can then be used with a neural network to generate interesting compounds.

5.4.2 Methodology

The first part of the process was to select a biological target to create an activity landscape, then select an active zone in which molecules should be generated.

Data

The ChEMBL3717 (Hepatocyte growth factor receptor) target was selected for this project. 4176 compounds with known activities were extracted from ChEMBL24 and encoded into ISIDA descriptors using IA—2-7 fragmentation schemes (sequences of atoms, length 2 to 7). 6520 descriptors were filtered down to 728 by removing all descriptors with a standard deviation of less than 2% of the maximum. An activity landscape of the ChEMBL3717 target based on the previously trained Universal Map 4 manifold was created, as well as 728 descriptor value landscape on which the entire ChEMBL24 database was projected and each of the 728 maps were coloured according to the value of one descriptor.

An activity landscape of the ChEMBL3717 was also created based on the latent vectors of a previously trained Autoencoder model. Additionally, 728 landscapes have been prepared on the AE GTM, one for each of the ISIDA fragment descriptors

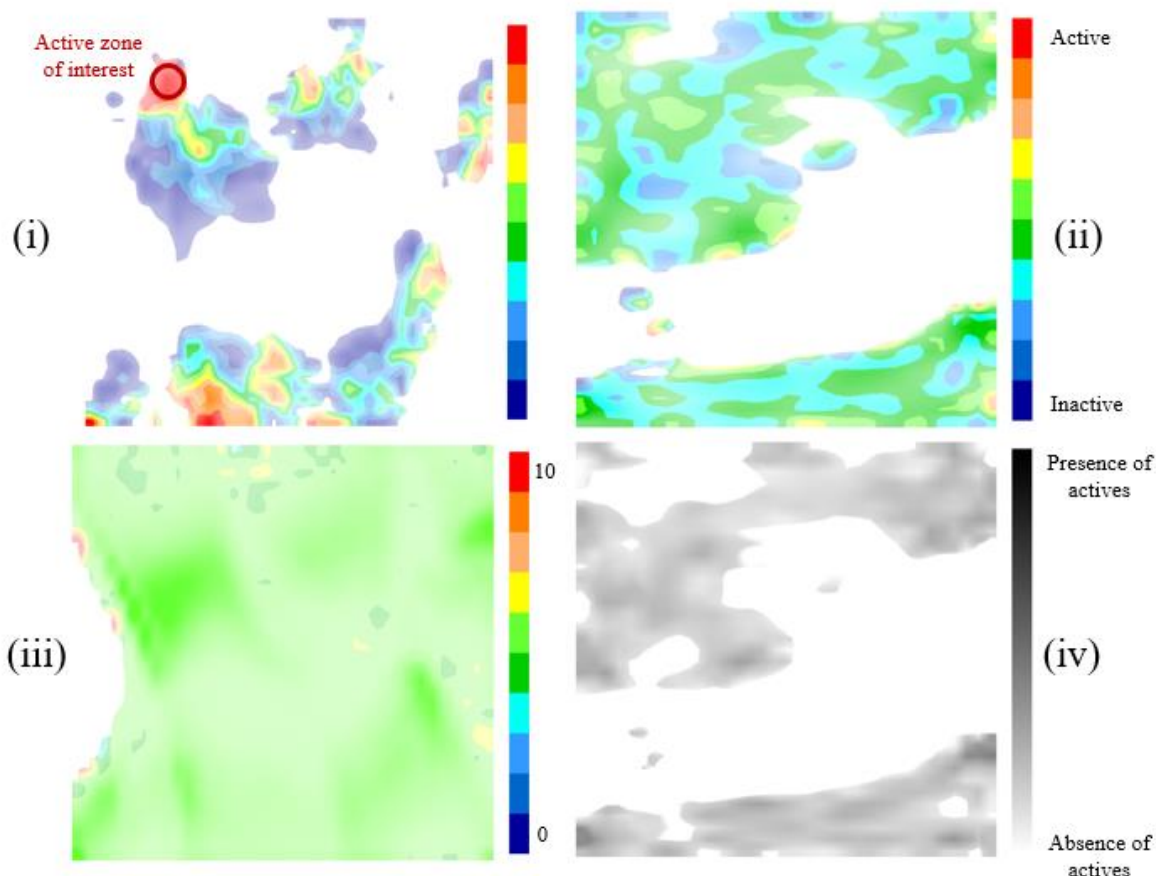


Figure 69. (i) Activity landscape for the ChEMBL3717 target based on the Universal Map 4 manifold. The circled area shows the most clearly separated, dense active area which was selected as the area of interest. The central node in the circle and the 8 nodes surrounding it were used to generate the “query” vector. (ii) Activity landscape for the ChEMBL3717 target based on the latent vectors generated by an autoencoder model. (iii) Landscape based on the Universal Map 4 manifold where all compounds from ChEMBL24 were projected and coloured by the value of one descriptor. This map is one of the 728 generated and shows the occurrence of the CCCC descriptor. (iv) Density landscape where only active compounds against ChEMBL3717 were projected. The darker the area, the denser in terms of actives it is.

Figure 69 shows clear separated active areas on the ChEMBL3717 ISIDA-based. One of these areas was selected as shown on the figure and 9 nodes were isolated as the target location. Additionally, the min and max values of the 728 “single descriptor value” were recorded from the compounds localized in those 9 nodes (an example is shown in Figure 69,

(iii)). This resulted in 728 ranges of descriptor values corresponding to the selected active zone for the ChEMBL3717 target.

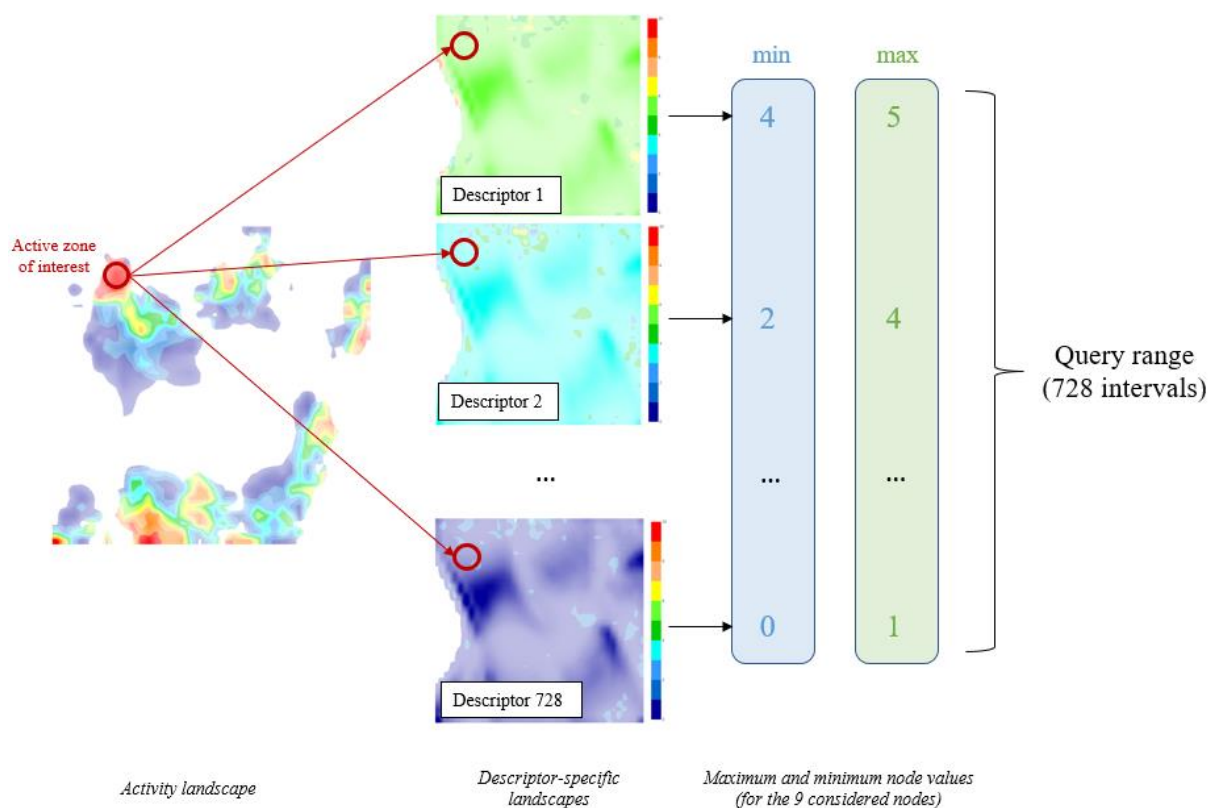


Figure 70. Creation process of the “query”. Each of the 728 descriptors landscape is checked on node 273 and surrounding (9 nodes in total) for its descriptor value. Maximum and minimum values of each descriptor in the 9 nodes are extracted which gives a range for a descriptor in this area of the map. The combination of all 728 ranges gives the query, which corresponds to the potential values of each descriptor in the active area. The query indicates which values the descriptors should have so that the compound is projected into the active area.

As is illustrated in Figure 70, the query gives the values that the descriptors should have if a compound were to be projected in that area. The query cannot be used directly to generate molecules unfortunately; however, it can be linked to the latent space of an autoencoder to try and identify a zone which would have the same descriptor values. If such a zone can be identified, then compounds corresponding the query can be generated. Therefore, the content of each node of the AE latent vectors is checked for its compatibility with the query range of values for the ISIDA molecular descriptors. Each node of the AE latent vectors landscape is compared to the range of the corresponding descriptor, resulting in a “correspondence” vector. This vector has the same dimension as the number of nodes of the latent landscape and assigns

a 1 to nodes which values correspond to the given descriptor range and a 0 in the other case (Figure 71).

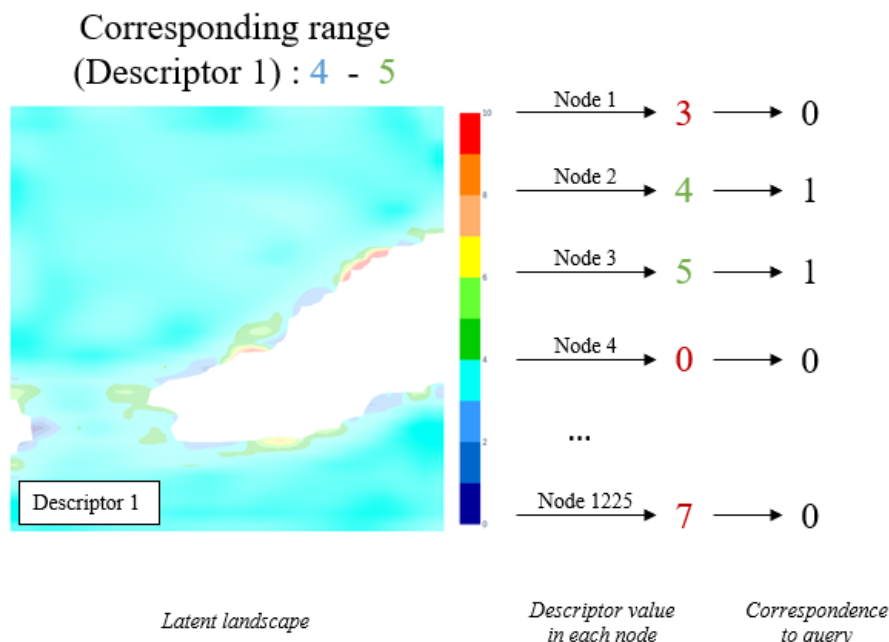


Figure 71. Example of creation of a “node correspondence” vector for a latent landscape representing a particular fragment. Each node is probed for its descriptor value which is compared to the query for that fragment. If the node fits the query, then vector will get a one, else it will be a 0.

Descriptors which had a query range of (0 – 0) were removed which left 71 “meaningful” descriptors: for each node, it contains a 0 if the node is not compatible with the range of the molecular descriptor and a 1 otherwise. Adding the 71 “node correspondence” vectors results in a single “cumulated node correspondence” vector. With this vector, one can locate the nodes of the latent landscape having the highest correspondence to the query ranges. The node with the highest correspondence was selected and used as a seed to generate chemical structures.

5.4.3 Results

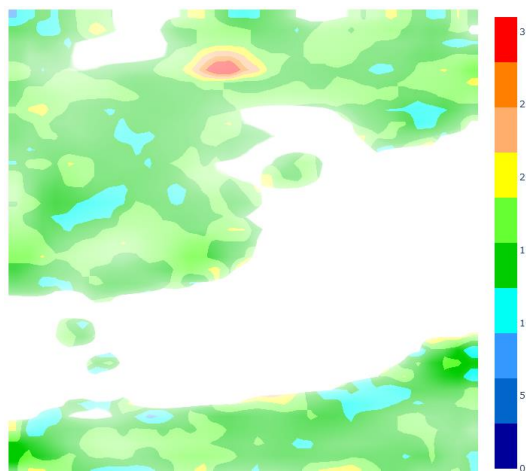


Figure 72. Latent landscape coloured using the cumulated “node correspondence vector” generated from the 728 LATENT landscape. Note that only 71 meaningful descriptors remained after filtering. The maximum is found in node 565 with 34 matching descriptors.



Figure 73. Density landscape for the 1662 generated compounds projected on the ISIDA manifold. As we can see, projected compounds are not near the wanted node.

As shown in Figure 72, one specific area of latent space had the biggest correspondence to the active area in ISIDA space. On average, the correspondence any given point on the landscape was about 21-25%. The red area on the figure has about 48% correspondence which is twice bigger. 10.000 vectors from this node were therefore sampled (node 565) which gave

1662 valid SMILES. These SMILES were reprojected on the initial ISIDA landscape to see if they were in the correct area of ISIDA space (Figure 73).

The ISIDA molecular descriptors of the generated compounds were computed and projected on the ISIDA GTM. However, it appeared that the generated compounds did not cover the initially selected region, the node 273. The selected node has a density equal to 0 and the projections are not in the active area. The compounds with the highest rate of correspondence to the query (about 50%, 35 descriptors out of 71) had strange and chemically non-sensical structures which could be filtered out (Figure 74). Meaningful chemical structures were also obtained but with a lower correspondence rate. (Figure 75)

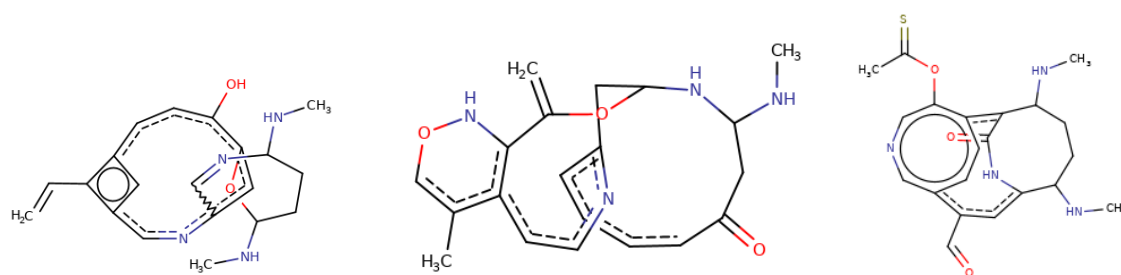


Figure 74. Three of the compounds with the highest correspondence to the query. (35, 34 and 33 corresponding descriptors from left to right)

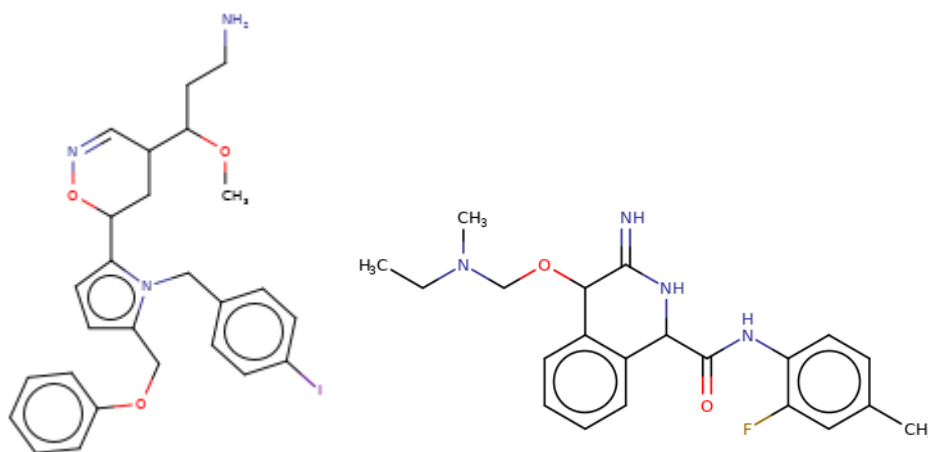


Figure 75. Two examples of more feasible and stable compounds, which had 11 matching descriptors (left) and 14 matching descriptors (right).

5.4.4 Conclusion

The more complex approach of the “query” vectors using a combination of ISIDA and latent landscapes was not successful either. The method seems to run into the same issues as previous methods, especially Stargate-GTM. A complete incompatibility of spaces, which makes it impossible for two zones to be similar in terms of molecular structure. We observed that the generated compounds with reasonable structures had very few descriptors in common with the query vector.

5.5 Conditional Variational Autoencoder (ACoVAE)

Linking the latent space of an Autoencoder with a separate descriptor space could not be performed by training the Autoencoder space separately. Vanilla Autoencoder latent space based on SMILES string has a completely different latent space construction and structure than structural descriptors, making it impossible to simply create a bridge between the two. However, by imposing the link during the training of the model using condition vectors, it is possible to force the neural network to adapt to a different set of descriptors.

In this work, a Conditional Variational Autoencoder was developed containing 3 important features:

- i) A GRU-based variational encoder encodes SMILES into latent vectors.
- ii) A descriptor vector corresponding to the inputted SMILES string is transformed into a condition vector and concatenated with the latent vector obtained from the VAE.
- iii) A powerful attention-based decoder translates the concatenated vector into a SMILES string.

With a model capable of generating compounds from ISIDA descriptors, the goal was to select the best candidates for the generation of actives against the ChEMBL1862 target. Three methods were used:

- a) A GTM based on descriptors from a Universal map (force-field type colouring of sequences of atoms) was built and coloured according to the activity against the ChEMBL1862 target. The zones with the highest concentration of actives were selected and the corresponding ISIDA vectors were used as candidates.
- b) A Genetic Algorithm based on an SVR model predicting the activity of a descriptor vector against ChEMBL1862 was used to find the optimal candidates.
- c) The descriptor vector of the best known active against ChEMBL1862 was used as a candidate.

All three candidate selection methods returned “seeds” which were used to generate several thousand compounds, which were screened for their activity potential by pKi calculations using an SVR model, pharmacophore search and docking.

Inverse QSAR: Reversing Descriptor-Driven Prediction Pipeline Using Attention-Based Conditional Variational Autoencoder

William Bort, Daniyar Mazitov, Dragos Horvath, Fanny Bonachera, Arkadii Lin, Gilles Marcou, Igor Baskin, Timur Madzhidov, and Alexandre Varnek*



Cite This: *J. Chem. Inf. Model.* 2022, 62, 5471–5484



Read Online

ACCESS |



Metrics & More

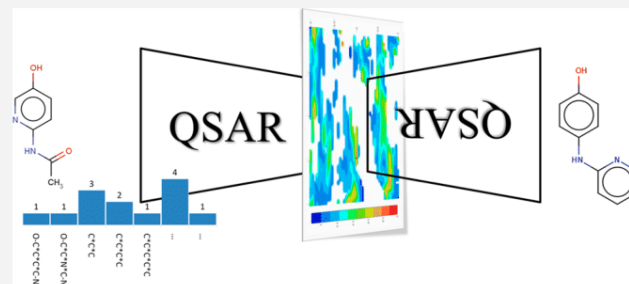


Article Recommendations



Supporting Information

ABSTRACT: In order to better formalize it, the notorious inverse-QSAR problem (finding structures of given QSAR-predicted properties) is considered in this paper as a two-step process including (i) finding “seed” descriptor vectors corresponding to user-constrained QSAR model output values and (ii) identifying the chemical structures best matching the “seed” vectors. The main development effort here was focused on the latter stage, proposing a new attention-based conditional variational autoencoder neural-network architecture based on recent developments in attention-based methods. The obtained results show that this workflow was capable of generating compounds predicted to display desired activity while being completely novel compared to the training database (ChEMBL). Moreover, the generated compounds show acceptable druglikeness and synthetic accessibility. Both pharmacophore and docking studies were carried out as “orthogonal” *in silico* validation methods, proving that some of *de novo* structures are, beyond being predicted active by 2D-QSAR models, clearly able to match binding 3D pharmacophores and bind the protein pocket.



1. INTRODUCTION

Predictive quantitative structure–activity/property relations (QSAR/QSPR)¹ are regression or classification models that are able to compute, upon input of a molecular structure, an estimate of the activity/property value the compound is expected to display. One may formulate the above as activity = $f(\text{structure})$, where function f needs first to be calibrated in order to have $f(\text{structure})$ returning accurate approximations of known activity values. If the above holds, then *inverse mapping* would allow to retrieve the “optimal” chemical structure(s), maximizing the expectancy of having an activity matching the input argument, that is, the desired activity level needed to achieve success in the current research project.

Since the first pioneering linear regression model by Hansch and Leo,² procedures to “fit,” e.g., machine learn $f(\text{structure})$, have progressed to the point of routine calibration of nonlinear models based on a plethora of machine learning methods (support vector machines, partition trees, neural networks—to cite only the most popular^{3–7}).

Typically, the *structure* argument in $f(\text{structure})$ is the molecular graph with vertices colored by chemical elements and edges colored by bond types. Since $f(\text{structure})$ returns a real number, it is obvious that the information content of the input molecular graph could first be translated in this process into some purely numerical representation—a vector of N real numbers \vec{D} known as the “molecular descriptor vector.” In classical QSAR, the two formal steps, descriptor calculation \vec{D}

= $\theta(\text{structure})$ and model fitting, activity = $\mu(\vec{D})$ are clearly separated into successive steps, and hence activity = $\mu(\theta(\text{structure})) = f(\text{structure})$. Hence, the inverse QSAR problem may be conceptualized as a succession of two formal steps:^{8–10}

1. finding descriptor vectors (“seed vectors”) matching the desired activity level: $\vec{D} = \mu^{-1}(\text{activity})$
2. finding the structures that correspond to the \vec{D} above: structure = $\theta^{-1}(\vec{D})$

Since $\mu: \mathbb{R}^N \rightarrow \mathbb{R}$, searching extremal points of $\mu(\vec{D})$ is a standard optimization problem, and albeit solving may prove challenging when μ is highly nonlinear or if N is large, this step of inverse QSAR is conceptually an easy one.

By contrast, step 2 is both technically and conceptually hard—to the point that, until recently, the typical way to discover molecules with activity values matching a desired activity level is to enumerate candidate structures and apply, to each, the QSAR model until all input candidates were herewith “virtually screened^{11,12}” or until enough events $f(\text{structure}) \approx$ desired activity occurred, e.g., “virtual hits” were found. Virtual

Received: August 27, 2022

Published: November 4, 2022



screening (VS), however, is limited by the choice of candidate structures either from public/commercial databases or from user-designed virtual libraries. In contrast to systematic VS, sampling techniques of chemical structures consider molecular structure as evolvable.^{13–15} This is *de novo* design,^{16–23} which fundamentally differs from VS by the fact that structures are not a predefined library but are generated and/or modified “on the fly” by some automated molecular structure editor.

The recent advent of deep neural networks (DNNs), able to extract information from arbitrary “brute” data and herewith learn to recognize patterns, had a major impact in the field of QSAR.^{24–28} The idea of DNNs is mimicking a human brain in which neurons communicate by generating and passing signals. Along with many applications of DNNs, Rana *et al.*²⁹ reviewed the application of the simplest example of DNN models—multilayer perceptron (MLP)—to disease diagnostics. MLP was also shown as a method to build successive QSAR models.³⁰ Later, parsing a chemical structure given in the form of a SMILES string by DNNs using the natural language processing technique was proposed as a new approach for QSAR model training.³¹ This success was not the last, and soon graph convolutional networks were proposed as a replacement of recurrent neural networks (RNNs) in QSAR modeling.³² As the research domain is in full effervescence, an exhaustive overview of already envisaged DNN architectures is beyond the scope of this article. The reader is encouraged to access the most recent reviews.³³

Some DNN architectures, namely, autoencoders, relate input structure (simply rendered as SMILES³⁴) to activity within a unique computational framework, apparently bypassing the need for molecular descriptors in QSAR. *De facto*, SMILES string encoder architectures first translate structure to a “latent” real vector \vec{L} , which the associated decoder would use to regenerate the SMILES. Thus, \vec{L} is nothing but a machine-generated molecular descriptor vector. Therefore, the decoder is a deep-learning-based model based on latent space descriptors \vec{L} implicitly allowing for a solution to the inverse problem.

So far, the majority of QSAR models are still based on classical, human expert-designed descriptors. This is first due to historical reasons, latent space descriptors \vec{L} being very new. However, expert-designed descriptors \vec{D} may still have a key advantage over the former (such as atom order invariance, which may be an issue in \vec{L} spaces—and their support of relatively small training sets in contrast to “big data”-dependent DNN approaches). So far, only a few attempts to convert arbitrary descriptor space \vec{D} back to structure have been described. One work³⁵ reports two distinct RNN-driven approaches labeled PCB (physchem-based) and FPB (finger-print-based). The former inputs a vector of predicted physicochemical properties (including a QSAR-predicted bioactivity value) to generate SMILES strings of compounds matching these properties. The latter uses Morgan fingerprints for input. Similarly, a transformer architecture has been implied to “translate” various classical chemoinformatics fingerprints back to structure.³⁶ Both works can be considered as examples of “hard” inverse QSAR approaches and were successfully used to generate structures in the neighborhood of known actives. However, they stopped short of coupling “easy” and “hard” QSAR problems in order to investigate how their approaches would cope with input vectors corresponding to optima of the QSAR landscape, not to already known molecules.

For the above reasons, the current contribution wishes to explore the feasibility of a genuine solution for the inverse QSAR problem for models based on classical, expert-defined molecular descriptors. The core of this work consists in the development of an attention-based conditional variational autoencoder (ACoVAE) based on transformer architecture. Given the seed vectors of ISIDA fragment descriptors, the ACoVAE generates corresponding molecules.

We have used two types of in-house generated QSAR models of ABL tyrosine kinase 1 (ChEMBL1862) activity:

1. Support vector regression (SVR) models for the inhibition constant (pK_i) using \vec{D} = ISIDA^{37,38} circular fragment counts. Seed vectors prepared with the help of a genetic algorithm used to sample \vec{D} space with predicted pK_i value as fitness.

Additionally, the descriptor vector of the molecule possessing the highest affinity (“lead molecule” LM) from the ChEMBL1862 set was also used as a seed vector.

2. Generative topographic mapping (GTM)-based predictive activity class landscapes using the “universal” map³⁹ based on \vec{D} = force field-type colored⁴⁰ ISIDA atom sequence counts. Sampling of \vec{D} was performed around the coordinates of active-enriched nodes of the landscape.

The inverse QSAR problem is considered solved if (i) the obtained structures are valid and chemically feasible and (ii) the obtained structures are submitted to classical forward QSAR model prediction and return conveniently high activity values.

Here, the ultimate goal was to obtain *de novo* structures that are perceived by a QSAR model to be highly active—whether they really are active or not is a question of underlying model quality, not of the quality of the inverse QSAR approach. Nevertheless, an alternative orthogonal *in silico* validation of these structures as ligands of the considered targets has been performed by pharmacophore analysis with the LigandScout⁴¹ program and by docking using both LeadIT⁴² and S4MPLE⁴³ approaches.

2. METHODS

2.1. ACoVAE. The proposed ACoVAE transformer model is shown in Figure 1. It consists of three main parts:

- (1) During the training procedure, a GRU-based encoder parametrizes a random latent vector distribution based on the training set SMILES. Hyperspherical distribution with zero mean and variance equal to 1 is used as target latent vector distribution;
- (2) A condition vector encoder uses a grouped linear transformation (GLT) layer⁴⁴ to transform initial descriptor vectors to a conditional latent vector;
- (3) A standard autoregressive multihead attention decoder⁴⁵ translates from condition and random latent vectors to SMILES. A more detailed architecture of the network is given in Supporting Information, Figures S1 (training stage) and S2 (inference stage). During the training, a SMILES strings and their corresponding descriptor vectors are used to train the ACoVAE. A reparameterization trick for latent vector sampling is used to train the network end-to-end. In the inference stage, the latent vector is sampled from a prior (0, 1) hyperspherical distribution, and a desired descriptor vector is used as

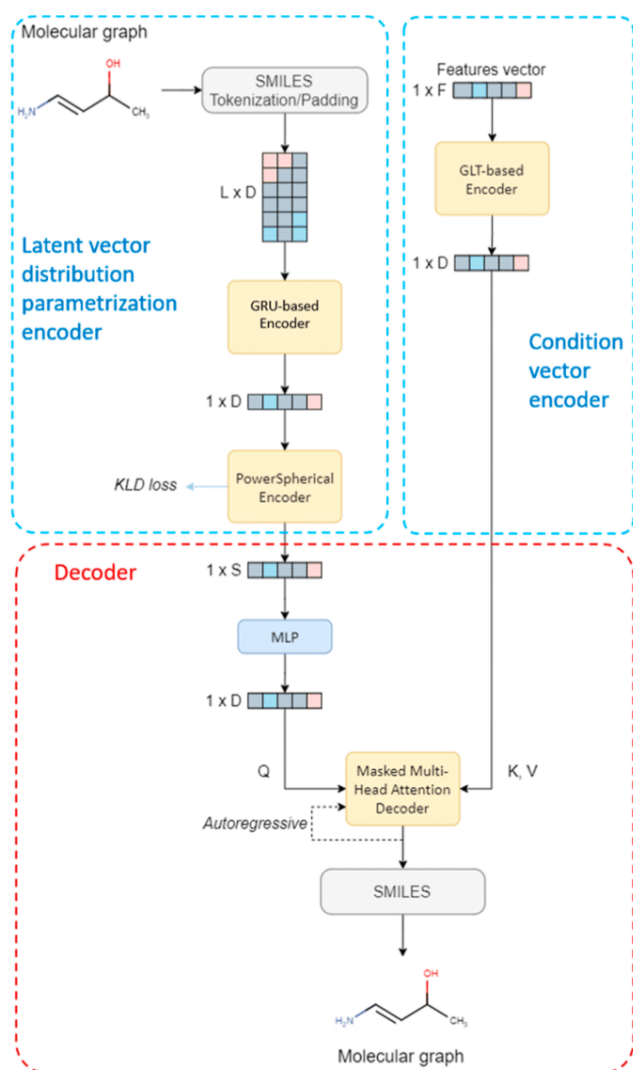


Figure 1. General scheme of the ACoVAE architecture used in this study. The GRU-based encoder (top left) parametrizes SMILES into latent vectors following a hyperspherical distribution, which is used upon inference for random sampling. The descriptor vector which is used as a condition in the generation is embedded by a GLT layer (top right). Autoregressive transformer is used to decode random latent vectors and combined conditions into SMILES strings. A detailed representation of all three networks is given in the [Supporting Information](#).

condition. Based on the random and condition vector, the decoder generates a wanted SMILES. Notice, that alternative SMILES for a given condition descriptor vector can be generated both (i) by running inference stage with different random vectors sampled from a prior distribution and (ii) by sampling different text strings using categorical sampling from token probabilities predicted by the transformer for a given random and condition vector.

The proposed architecture of the ACoVAE transformer was inspired by the one proposed by Lin *et al.*⁴⁶ In a similar way, a random latent vector is fed as a START token. However, substantial changes were introduced which helped us to achieve better performance. In our architecture, a random latent vector is encoded directly using a GRU, while Lin *et al.* used a trick with a priori undefined random distribution

parameterized by a separate network. Additionally, a hyperspherical uniform distribution was preferred to a standard Gaussian one because during the tuning stage, the former performed better. A von Mises–Fisher distribution is commonly used for sampling from hyperspherical uniform distribution⁴⁷ with the reparameterization trick. However, we found that the power spherical distribution⁴⁸ used instead of von Mises–Fisher one allows a speeding up of the learning process without loss of the performance. Application of a GLT transformation layer⁴⁹ better translates the descriptor vector into the internal representation used by the decoder network than MLP. Finally, inspired by the GELU approximation,⁵⁰ new activation function FTswishG resulted from some modifications of the previously reported FTswish⁵¹ was used throughout the ACoVAE network

$$\text{FTswishG} = \text{RELU}(x) \times \text{sigmoid}(1.702x) - 0.2 \quad (1)$$

According to our tests, it gives better results compared to the ReLU, GeLU, and FTswish activation functions. In such a way, our ACoVAE transformer architecture is a novel one, having only a few in common with the one proposed by Lin *et al.*⁴⁶ The designed architecture is implemented using the TensorFlow framework and can be readily retrained for other descriptor types. It is available on our GitHub storage <https://github.com/Laboratoire-de-Chemoinformatique/ACoVAE>.

2.2. SVR Models. A series of ligands for ABL tyrosine kinase (ChEMBL1862) from the ChEMBL v.23 database was standardized using a protocol reported by Sidorov *et al.*³⁹ SVR models for thermodynamic instability constants of protein–ligand complexes ($\text{p}K_i$) were generated using the evolutionary *libsvm* model tuner,⁵² which supports selection of the best suited descriptor space yielding to best performance models as a key hyperparameter. The best-suited ISIDA fragmentation schemes were defined together with the SVR-specific parameters (kernel type, cost, γ , *etc.*) optimizing model quality. The models were built on a training set containing 739 molecules and validated on a test set of 82 molecules. The test set data were collected from recent publications posterior to model training. The best model relies on IIRAB-1-3 ISIDA fragment count descriptors (7372 atom-centered fragments with a radius of 1 to 3 atoms with restricted fragmentation) and the Gaussian kernel option. It displayed a reasonable performance in cross-validation ($R^2 = 0.79$ and $\text{RMSE} = 0.70$) and on the test set ($R^2 = 0.80$ and $\text{RMSE} = 0.67$).

Computation of the “optimal” seed vectors has been confided to an evolutionary heuristic browsing through the \vec{D} space in search of vectors maximizing computed $\text{p}K_i$ values. The “chromosome” of the approach is a 20-dimensional integer vector in which loci may contain either zero or a number denoting a training set compound. The vector encoded by such a chromosome is taken as the mean $\langle \vec{D} \rangle$ of descriptor vectors of the training set compounds mentioned in the chromosome (a compound may be mentioned several times in different loci, which amounts to increasing its weight in the computed average). The fitness score of the chromosome is nothing but the corresponding $\text{p}K_i = \text{SVR}(\langle \vec{D} \rangle)$ to be maximized. Hence, the evolutionary algorithm is bound to find, by applying cross-over and mutation operators, chromosomes enumerating optimal sets of training set compounds, with the property that the centroid of the descriptor vector of the set is predicted to correspond to high affinity values. The procedure was applied for each SVR model for 150,000 generations. Sampled “high-affinity” $\langle \vec{D} \rangle$ values

were used as the condition vector for the ACoVAE decoder. Details about evolutionary model building can be found in our publication,⁵² which also provides instruction on how to obtain and download that tool. Here, it was used with default setup, meaning 12-fold-repeated three-fold cross-validation (with steadily reshuffled cross-validation tiers at every iteration). The model fitness score was the mean cross-validated determination coefficient $\langle Q^2 \rangle$ penalized by 1 standard deviation, $\text{fitness} = \langle Q^2 \rangle - \sigma(Q^2)$.

2.3. GTM Landscape-Driven Models. GTM is a dimensionality reduction technique developed by Bishop *et al.*^{53,54} The method performs a nonlinear projection of an N -dimensional space onto a 2D latent space. The former corresponds to the descriptor space, where each molecule is defined by an N -dimensional molecular descriptor vector. The 2D latent space corresponds to a manifold which is defined by a set of radial basis functions and evaluated on sample points called “nodes.” Simply put, the manifold can be seen as a rubber band that can be folded in N -dimensions during training to fit the data distribution in a way maximizing its coverage of the space zones populated by relevant items (the “frame set”). Any compound can subsequently be projected on the manifold. For visualization purposes, the manifold is “unfolded” into a 2D plane, organizing the nodes into a square grid. GTM is a probabilistic method, meaning that compounds are fuzzily projected on all nodes of the manifold. As such, an item is associated with (“resident in”) each node with different probabilities. The sum of the probabilities—technically named responsibilities—over all nodes of the manifold equals 1. In practice, this means that one compound will be defined by a responsibility “pattern” potentially involving several nodes instead of being confined to one node only. When projecting compounds of experimentally known properties, neighborhood behavior⁵⁵ (NB) compliance implies that residents of the same node should have related property values, so that the node may be seen to “represent” that local average property, and “colored” accordingly. Resulting property “landscapes” are nothing but NB-driven QSAR models: the property of any external item can be predicted from the “local color” of the landscape zone onto which it is projected. In this work, the fuzzy class landscapes (monitoring the likelihood to classify as “active” with respect to a target) were employed. They were based on the previously published⁵⁶ universal map #1 (UM1)—the first of a series of GTMs parameterized (using ChEMBL data), such as to maximize their “polypharmacological competence,” that is, their ability to host a large battery of highly predictive fuzzy class landscapes associated with diverse biological targets. Note that landscape-based QSAR models are parameter-free (the landscapes are built by projection of existing structure–activity data on the given manifold in an unsupervised manner). Therefore, landscape-based QSAR models are implicitly available as soon as the supporting structure–activity data are available.

The structure–activity data set associated with the ChEMBL1862 target was projected on the manifold of the first universal map UM1⁵⁶ and was seen to “spontaneously” segregate into zones populated predominantly by “actives” and “inactives,” respectively. This map was built based on ISIDA⁴⁰ atom sequence counts with a length of two to three atoms labeled by CVFF force field types and formal charge status (IAFF-2-3-FC). Recall that construction of activity landscapes on a given GTM manifold is not supervised but a purely deterministic procedure. The separation proficiency of the

considered manifold was obtained by repeated leave-1/3-out cross-validation, in which iteratively two-third of the items are projected on the map in order to “color” the activity class landscape, whereas the remaining one-third of compounds *a posteriori* projected onto that landscape and have their activity classes assigned on basis of their residential zones in the landscape. Cross-validated balanced accuracy was 0.78, significantly above the randomness threshold of 0.5. The structure–activity dataset is herewith proven to be robust and modelable by both machine-learning (SVR) and neighborhood analysis-based mapping.

Activity class landscape for ChEMBL1862 was used to identify zones in the chemical space in which “active” compounds tend to cluster preferentially. Note that the label “active” was assigned to compounds with the ~25% highest affinity values according to the initial automated data curation procedure used for universal map fitting. The GTM nodes n in which active compounds were seen to preferentially reside were identified as key points if

$$\frac{\sum_{c \in \text{Actives}} R_{cn}}{\sum_{\text{all } c} R_{cn}} \gg \frac{N_{\text{Actives}}}{N_{\text{all}}} \quad (2)$$

R_{cn} represents the responsibility of compound c with respect to node n , summed over actives (numerator) and over all training compounds (denominator), with the ratio representing the fuzzy-logic propensity to expect an active “resident” in node n . This propensity should be much higher than the baseline propensity to encounter an active throughout the training set (top nodes were selected according to the ratio of summed responsibilities). Coordinates of these key nodes correspond to vectors in ISIDA descriptor chemical space zones expected to harbor active compounds. The Gaussian neighborhoods of key node vectors were sampled by generating a multidimensional Gaussian distribution with a width of $w = 0.05$. Several vectors were generated from the initial node vector using this method.

2.4. Solution of Inverse QSAR Problem: The ACoVAE Algorithm. Sampling with the ACoVAE transformer is accomplished by giving a descriptor vector to the trained decoder part of the model. Each descriptor vector, which corresponds to the “condition” part of the ACoVAE, is combined with a batch of random vectors from a power spherical distribution, which serves as the basis for the latent space. Each descriptor vector/random latent vector combination returns a sample of generated SMILES. Categorical sampling is the preferred method of generation since it allows, for the same input, to explore different possibilities, thus maximizing the generative “coverage.” Therefore, the batch of latent vectors returns a batch of generated SMILES. For example, for one descriptor vector concatenated with 200 different sampled random vectors with a batch size of 512, the algorithm returns $200 \times 512 = 102,400$ generated SMILES. In such a way, a given descriptor vector can be used several times leading to different SMILES. In-house CGRtools⁵⁷ software is used to verify the validity of the generated text string, directly removing any incoherent or incorrect SMILES.

The following parameters were analyzed when monitoring the pertinence of the inverse QSAR approach:

1. *Validity* = #valid SMILES/#all generated text strings, which measures success to generate a syntactically valid SMILES string (assessed by CGRtools), starting from the input “high-affinity” $\langle \vec{D} \rangle$ vectors.

2. *Feasibility* assessing chemical feasibility and drug-likeness according to Ertl⁵⁸ and QED⁵⁹ indices.
3. *Novelty*. A compound generated with ACoVAE is considered “novel” if it is not contained in the training database.

A coherence between the ISIDA descriptor vector recalculated for the generated SMILES string and the input vector at the source of that SMILES was assessed using the Tanimoto similarity score.

2.5. Filtering of Nonvalid SMILES Strings. During the sampling procedure, output SMILES were parsed and standardized using CGRtools. Then, they were transformed into Kekulé form followed by verification of valences. If no error detected, the SMILES strings were rearomatized and then written to the output. Failure of any step in this workflow leads to discarding the given text string as invalid SMILES.

3. RESULTS AND DISCUSSION

3.1. Finding Candidate Descriptor Vectors Associated with High Affinity. For the SVR model, the evolutionary sampler of the ISIDA descriptor space outlined in Section 2.2 is very fast to visit “high-affinity” $\langle \vec{D} \rangle$ values. Points in the ISIDA descriptor space corresponding to predicted pK_i values close to the ones of the most active compounds included in the training set can be discovered in matter of tens of minutes on Linux workstations with the following specification: Intel Xeon Silver 4214 2.20 GHz, 48 cores, 64 GB RAM, Ubuntu 18.04.6 LTS. However, the discovery of points with activities predicted to be *better* than the one of the best training compounds was never achieved despite of the total run times of the order of 48 h, resulting in >150 K visited $\langle \vec{D} \rangle$ values. On the one hand, it is not clear whether such points may actually exist—SVR may suffer (in particular when based on the Gaussian kernel) from the “regression towards the mean” effect, consisting of systematic underestimation of high and overestimation of low property values. Moreover, it is even less likely that points where the SVR model nevertheless predicts a value beyond the largest observed pK_i would actually be located within the “fragment control bounding box” defining the applicability domain⁵⁴ (AD) of the model. Given the fact that herein visited $\langle \vec{D} \rangle$ values are generated as means of descriptor vectors of randomly selected subsets of compounds, these points are guaranteed within the bonding box AD (each vector element D_i will be larger or equal than the minimal and, respectively, smaller or equal than the maximal D_i value ever encountered within the training set). Third, the top affinities for all these targets are already within the 0.1 nM range—discovery of significantly more potent molecules is extremely unlikely in this context. Therefore, the five visited $\langle \vec{D} \rangle$ values corresponding to the highest predicted pK_i scores (comparable but not better than the affinity of the most active compound) were used to tackle the inverse QSAR problem (see Figure 2).

As a complementary study to the inverse-SVR descriptor selection, the most active ChEMBL compound shown in Table 2 (compound A) was selected as a seed to show the difference between the generation from optimized vectors and a real active molecule.

For the GTM-based activity class predictors, two nodes that were most highly enriched in “active” residents were selected, as represented in Figure 3. Candidate descriptor vectors were obtained by augmenting the D space coordinates of these nodes with Gaussian noise as described in the Methods section

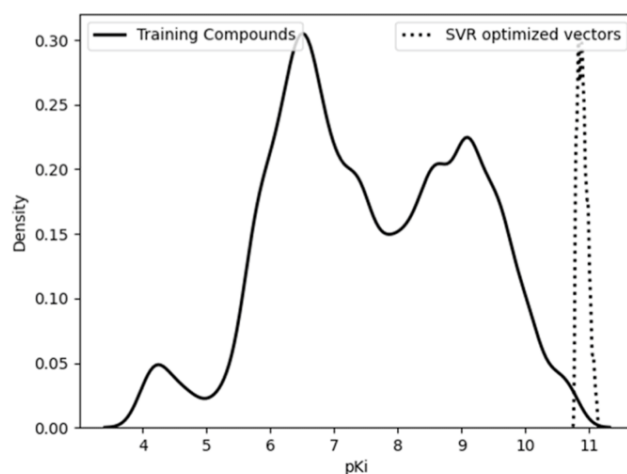


Figure 2. Distribution of pK_i for the compounds used to train the model. The dotted line renders the distribution of predicted pK_i for the vectors of the final population emerging from the evolutionary sampling approach.

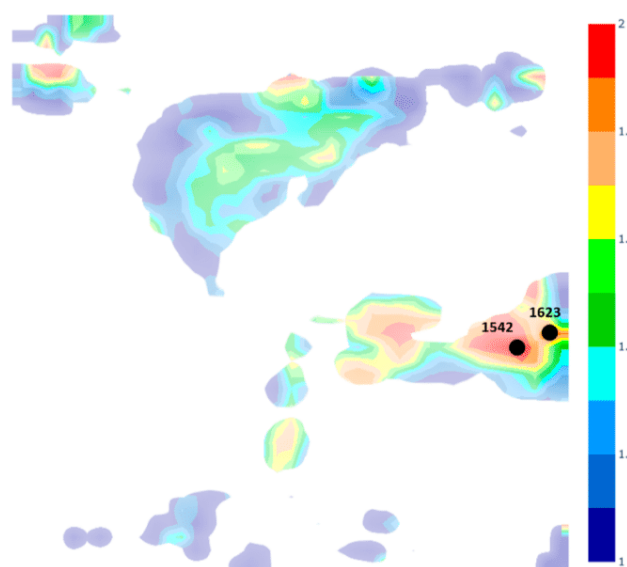


Figure 3. Selected nodes for target ChEMBL1862 on the fuzzy activity class landscape where color encodes the relative populations of actives (class 2, red when pure) vs inactives (class 1, blue when pure). Intermediate color design nodes with residents of both classes in various proportions. Numbers of the node are represented.

(see 2.3). Projection of these seed vectors on the landscapes below unsurprisingly assigns quasi-unitary responsibility values to their “source” nodes, implicitly qualifying them as “probable actives.”

3.2. ACoVAE Calibration Results. Two distinct ACoVAEs were trained—one for each relevant ISIDA descriptor space:⁴⁰ IIRAB-1-3 for the inverse-SVR problem and IA-FF-2-3-FC for the inverse-GTM challenge. Each training set contained the same 1,540,615 compounds from ChEMBL-23, standardized using ChemAxon⁶⁰ standardizer, following the procedure implemented on the VS server of the Laboratory of Chemoinformatics in the University of Strasbourg (<http://infochim.u-strasbg.fr/webserv/VSEngine.html>). The following standardization steps were applied: (i) dearomatization and final aromatization according to the “basic” setup of the

ChemAxon procedure (heterocycles like pyridone are not aromatized), (ii) dealkalization, (iii) conversion to canonical SMILES, (iv) removal of salts and mixtures, (v) neutralization of all species, except nitrogen(IV), and (vi) generation of the major tautomer with ChemAxon. This resulted in 1,540,615 unique, stereochemistry-depleted SMILES strings used for training (stereochemical information was removed because the herein used molecular descriptors do not capture it).

Model training was done for 100 epochs and lasted for about 30 h on a QUADRO RTX 6000 graphic card. The loss function tends to stabilize early during training as shown in Figure 4; however, the model continues to learn as character-

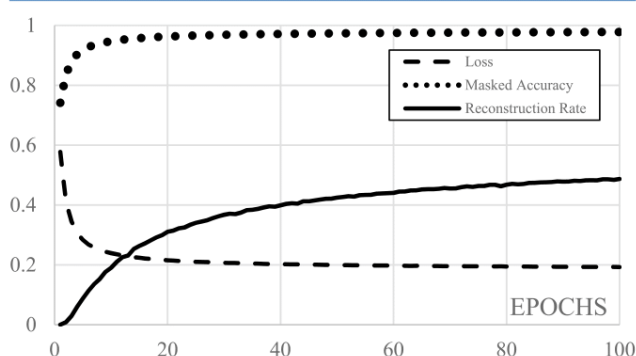


Figure 4. Training metrics for the ACoVAE transformer model based on ISIDA descriptors. “Loss” is the loss function of the model. “Masked accuracy” corresponds to the character-specific reconstruction rate. “Reconstruction rate” corresponds to the full SMILES string reconstruction rate.

specific reconstruction rates and pure reconstruction rates continue to grow. Arguably, the model could be trained for somewhat longer since the reconstruction rate (*val_rec_rate*) has seemingly not reached a plateau at 100 epochs. However, we believed that the achieved accuracy—some 50% reconstruction rate and 98% character-specific reconstruction rate, was sufficient for the model acceptance. Notice that variational autoencoders have a tendency for lower reconstruction rates than their deterministic counterparts because of the element of randomness introduced by sampling latent vectors from a given distribution instead of having deterministic latent vectors.

3.3. Inverse QSAR Results. **3.3.1. Inverse-SVR and Inverse-Lead Compounds.** According to Table 1 displaying various quality criteria of inverse-SVR compounds, the low success rate in the sampling procedure can be mitigated if we consider the time factor. Sampling of 512,000 SMILES strings (using 5 conditional vectors corresponding to the 5 vectors of highest activity predicted by the SVR model) resulting in 6899 valid, unique candidates takes only about 4 to 5 h on a QUADRO RTX 6000 GPU. Comparing lead molecule

sampling to inverse-SVR sampling shows that both perform similarly in terms of unique valid compounds and activity prediction, although lead molecule sampling scores a bit lower on the latter metric.

A descriptor vector marking a position in the chemical space may or may not translate to a chemically meaningful structure, knowing that the initial vector is typically not a slightly perturbed position vector of a real molecule but merely a chemical space point associated with high predicted activity according to a machine-learned, action mechanism-agnostic model. However, the ACoVAE decoder process injecting randomized latent vectors (see Section 2.1) may produce an arbitrary number of SMILES strings based on a given chemical space point. For each of the five considered chemical space points of high predicted affinity, chemically meaningful molecules were obtained (at a low success rate of 1.34%—but this is merely an order of magnitude of the likelihood to draw a random latent vector *i.e.*, “compatible” with the current chemical space position). The complexity of the molecule that the model is trying to generate is implicitly affecting the chance to retrieve a valid structure. Since the model generates SMILES strings, it must conform to a very specific grammar which is intolerant to errors. Any misplaced character in the SMILES sequence can render it incorrect and bring up an error—a well-known problem in chemoinformatics. Without extensive understanding of the chemical meaning behind a SMILES string, it can be very difficult to correctly open and close multiple rings to recreate valid structures with correct aromaticity and stable behavior. This, in part, explains why the model may be very successful in some parts of chemical space and struggle more in other parts. A possible solution to that problem would be the use of DeepSMILES^{61,62} or SELFIES⁶³ which use a simpler syntax eliminating the risks of incorrect ring closures and parenthesis errors.

GTM landscapes identify zones enriched in actives, nevertheless containing some inactives. The sampling is performed using an ensemble of seeds generated from a given GTM node. These seeds can occasionally be located in the vicinity of inactives. In contrast, sampling from the most active compound generates structures similar to this seed. This explains the difference in the proportion active/inactive for different seeds in Table 1.

Generated compounds were filtered to remove both chemically inconsistent species (by CGRtools) and duplicates and were compared to the initial training database (ChEMBL) to compute the “novelty” rate which corresponds to the percentage of valid unique generated compounds not appearing in the training set of the model. Table 1 shows that all generated compounds are novel. The trained SVR model was used to estimate the pK_i values of the generated compounds, which were then classified as actives or inactives

Table 1. Performance of the ACoVAE Transformer Model for the ChEMBL1862 Target When Sampling from Seed Descriptor Vectors from Different Sources

seed vector source	number (percentage) of valid compounds	number (percentage) of unique compounds	novelty compared to ChEMBL (%)	predicted active ^a (%)
SVR	12,432 (2.43%)	6,899 (55.49%)	100	48.6
GTM	70,684 (13.8%)	61,342 (86.78%)	99.98	6.9
lead molecule	23,559 (4.60%)	7,600 (32.26%)	99.95	41.6

^a“Predicted active” implies predicted $pK_i > 7$ by the SVR model. This latter is more stringent than GTM landscape-based predictions, which positions a vast majority of inverse-GTM compounds close to their “source” nodes and herewith classifies them as “actives.”

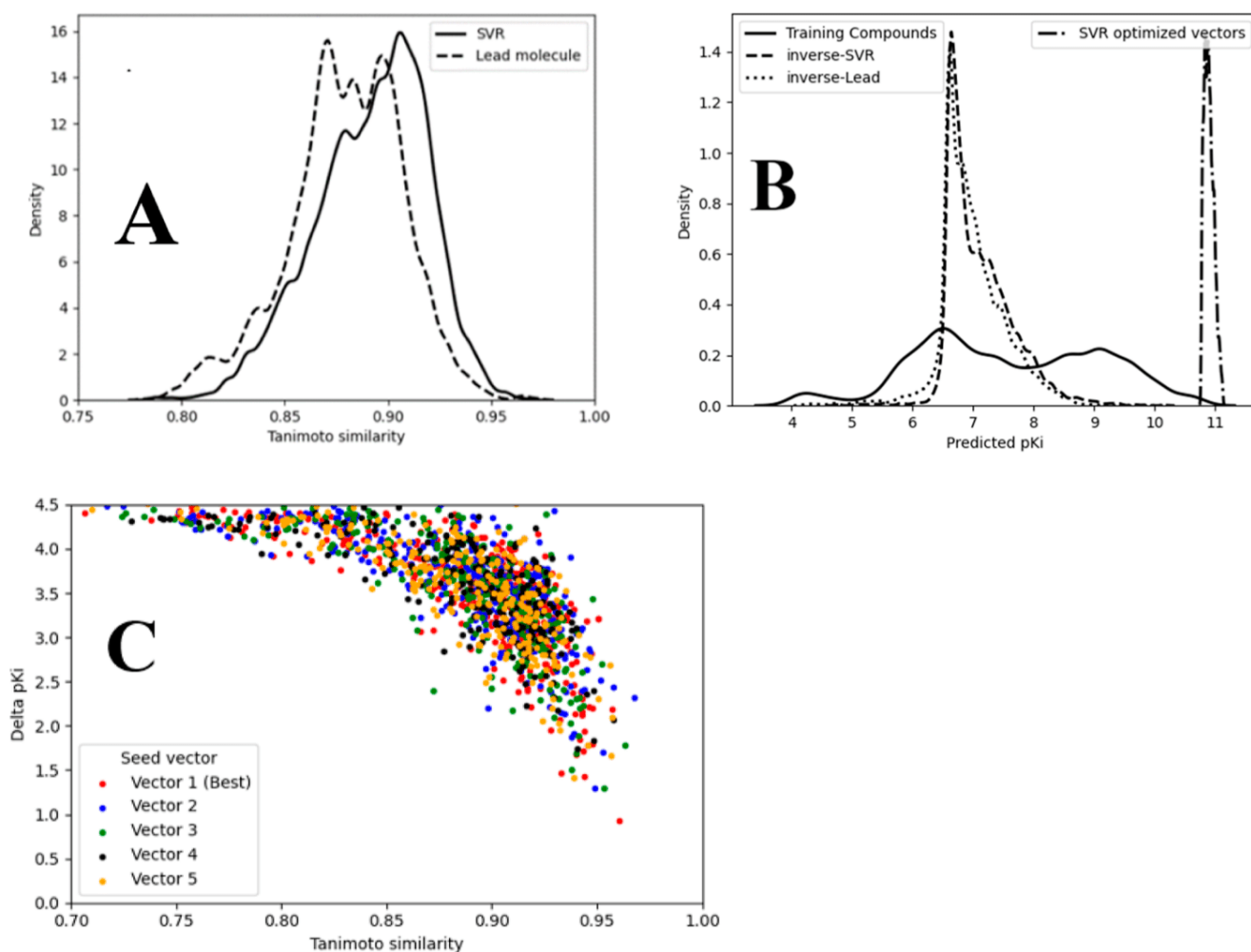


Figure 5. (A) Distribution of Tanimoto similarity calculated between sampled compounds and the ISIDA descriptors used for their sampling (obtained *via* SVR GA and lead molecule). (B) Distribution of predicted activities for inverse-SVR compounds, lead molecule sampled compounds, training compounds, and vectors optimized by GA. (C) Scatter plot with the *x*-axis being the Tanimoto similarity between the sampled compound and the GA vector and the *y*-axis, the difference in (calculated) pK_i between the inverse-SVR compounds and the original GA vector. The different colors correspond to the five different “seed” vectors used for the sampling procedure.

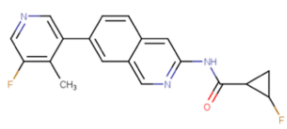
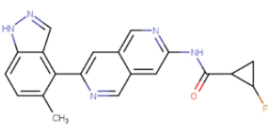
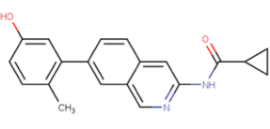
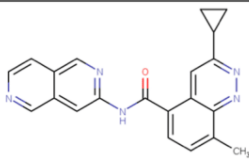
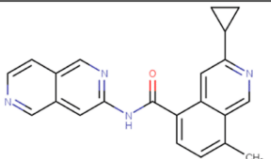
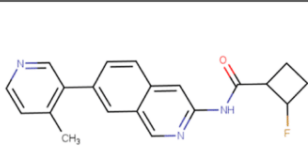
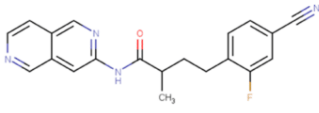
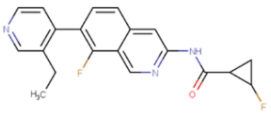
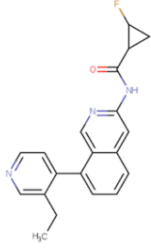
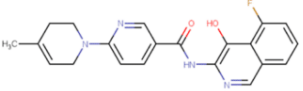
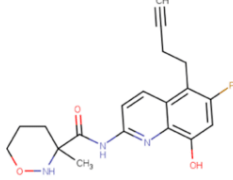
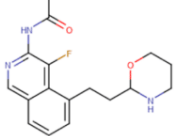
by using a threshold 7. As such, about half of the generated compounds were predicted to be active.

Compounds predicted as inactive by the model were filtered out. Generated compounds were compared to the GA-optimized vectors used as input to the model. Results in Figure 5A show that most compounds are very similar ($T_c > 0.85/0.90$) to their “seed,” meaning the model was able to understand the information contained in the descriptor vector and translate it in terms of SMILES. Given that the value contained in the vectors may not be integers or that some of the descriptor values may be incompatible, an average of $T_c = 0.9$ is a sign that the model was able to extract hidden knowledge from the ISIDA descriptor and adapt it to a chemically feasible structure. Some generated compounds approach the activity values of the GA-optimized vectors as shown in Figure 5B, although all active compounds have lower pK_i . Figure 5C shows the difference in predicted pK_i between the generated compounds (based on their actual \vec{D} vectors) and the “source” GA-optimized vectors ($\langle \vec{D} \rangle$), plotted against the Tanimoto coefficient $T_c(\vec{D}, \langle \vec{D} \rangle)$. Unsurprisingly, the SVR QSAR models are neighborhood-behavior compliant: the

closer the source vector ($\langle \vec{D} \rangle$) remains to the actual compound descriptor, the higher the likelihood to have the latter predicted at high affinity levels—(virtual) activity cliffs notwithstanding (pK_i shifts of 2 orders of magnitude may occasionally happen for 90% similar descriptor vector pairs).

The three most active compounds from ChEMBL, the three inverse-SVR and three inverse-lead molecules predicted that the most active were extracted and compared in terms of structural similarity and pK_i values. The most active inverse-SVR and inverse-lead compounds are structurally very similar in terms of substructure counts but not necessarily in terms of overall topology to the most active ChEMBL compounds, as shown in Table 2. Similar substructures or features like quinoline, cyclopropane, peptide bonds, and fluoride atoms appear in both ChEMBL and generated compounds—but they may be interconnected in a different way. Sampling the neighborhood of a given compound is likely to witness the neural network return typical “building blocks” seen in those compounds, all while recombining them and placing them in original contexts.

Table 2. Most Active ChEMBL-Reported Compounds (A, B, C) against the ChEMBL1862 Target as Well as the Most Potent Structures Generated from the Different Seed Vectors^a

ChEMBL compounds		
<p>A</p>  <p>10.73</p>	<p>B</p>  <p>10.70</p>	<p>C</p>  <p>10.70</p>
inverse-SVM compounds		
 <p>10.20</p>	 <p>9.84</p>	 <p>9.82</p>
inverse-Lead compounds ^b		
 <p>10.08</p>	 <p>9.45</p>	 <p>9.35</p>
inverse-GTM compounds		
 <p>7.88</p>	 <p>7.84</p>	 <p>7.83</p>

^aThe numbers correspond to experimentally measured (for ChEMBL compounds) or predicted with SVR models pK_i values. ^bCompounds generated for the descriptor vector generated for molecule A, which is the highest affinity molecule (inverse-LEAD) with $pK_i = 10.73$.

3.3.2. "Inverse-GTM" Compounds. Inverse-GTM sampling, in this case, gives better results in terms of validity and uniqueness than inverse-SVR compounds.

Compounds generated from a GTM node vector consistently tend to be projected into the same area they were sampled from. This is not true of all compounds, a minority being projected in different areas of chemical space—in inactive-dominated zones (see Figure 6).

In inverse-GTM, random noise is also used to perturb the input descriptor (GTM node vector), whereas inverse-SVR compounds were strictly sampled on hand of the five optimized descriptor vectors. Accordingly, the resulting compounds are more diverse but less prone to score very

high predicted pK_i values as shown in Table 2. Rather than focusing on recombination of fragments maximally contributing to SVR-predicted pK_i values, the model incorporates fragments of all training compounds occupying the vicinity of the chosen "seed" vector.

3.3.3. "Inverse-SVR" and "Inverse-Lead" Versus "Inverse-GTM". Sampling with inverse-SVR and inverse-lead has a chance to return molecules predicted highly active, which is not the case for compounds generated with inverse-GTM. This can be explained by the fact that inverse-SVR (inverse-lead) vectors served as the generation seed correspond to high activity values, which is not the case for the GTM node vectors. Inverse-GTM molecules have lower SVR-predicted

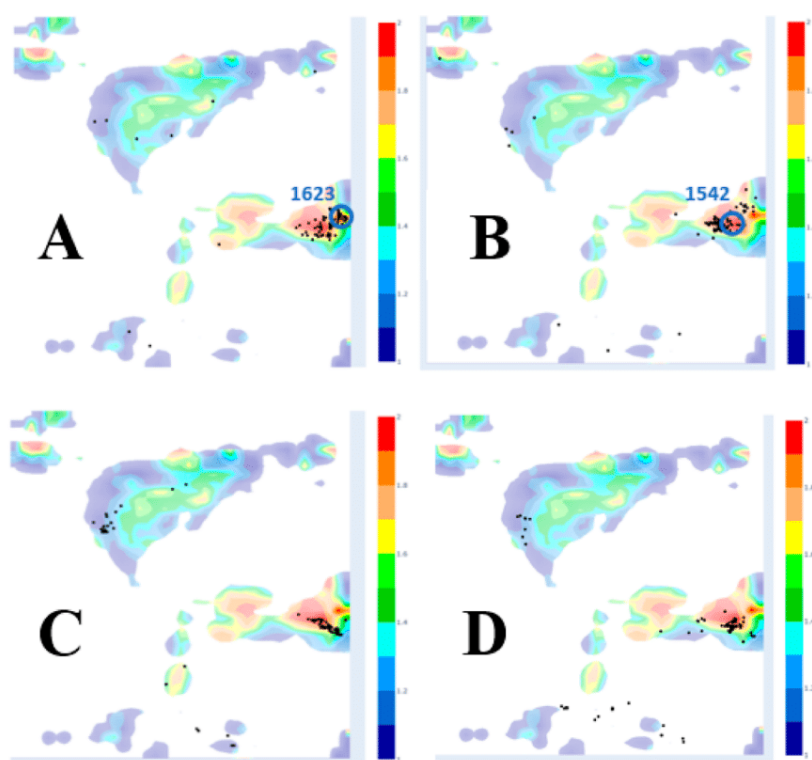


Figure 6. Projection of the 100 most active compounds predicted by the SVR models, generated in different fashions. See caption of Figure 3 for landscape color coding. (A) Compounds were generated from “node” vectors obtained from node 1623. (B) Compounds were generated from “node” vectors obtained from node 1542. (C) Inverse-SVR compounds. (D) Inverse-lead compounds.

pK_i values comparatively because “active” GTM landscape areas were defined to harbor “actives” of $pK_i \geq 7$, and the categorical nature of the landscape makes no further distinction between submicromolars and subnanomolars. The two methods produce active compounds, but molecules generated from inverse-SVR tend to be more focused on specific chemical space zones predicted to stand for very high affinity. Therefore, they reproduce structural features typical to the few top actives—the “originality” mostly consisting in the way in which these features (scaffolds, linkers) are reorganized in the final structures. Inverse-GTM seeds tend by contrast to stem from structurally less specific neighborhoods, generating a more diverse set.

Figure 7 confirms this trend as we see that the distribution of activities of inverse-SVR and inverse-lead compounds has a tail in the very active regions, while the distribution of pK_i for GTM-based compounds has a lower mean and is centered.

Interestingly, most of inverse-SVR compounds are projected in the large active zone where inverse-GTM compounds were sampled—even though the GTM-driven categorical QSAR is based on other descriptors than the SVR approach. This is additional proof that SVR-based and GTM-based models are not fundamentally divergent in terms of prediction but merely conflicting in terms of the specific definition of “actives” as continuous *versus* categorical magnitudes.

As it follows from Figure 8, synthetic accessibility score for the generated compounds (inverse-SVR, inverse-lead, and inverse-GTM) have on average a higher SA score than ChEMBL compounds. According to this score, generated structures are more difficult to synthesize than real ChEMBL molecules. On the other hand, they are still in the range of ChEMBL distribution (which goes up to 4.5–5) meaning that

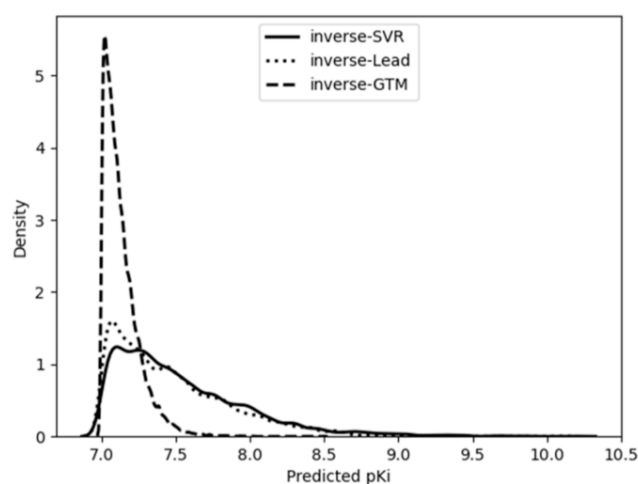


Figure 7. Comparison between the distribution of (SVR-predicted) activities between inverse-SVR, inverse-lead, and inverse-GTM compounds.

generated structures are not synthetically unreachable and therefore viable. The quantitative estimate druglikeness index shows that on average, inverse-SVR and inverse-lead compounds are of more interest for medicinal chemists than inverse-GTM compounds.

3.3.4. Validation of Inverse-SVR and Inverse-Lead Compounds Using Pharmacophore Modeling. Pharmacophore models were trained using LigandScout⁴¹ (4.4) to check whether the generated compounds would also comply to the ligand- and structure-based hypothetical binding patterns that can be inferred on hand of current structure–activity data.

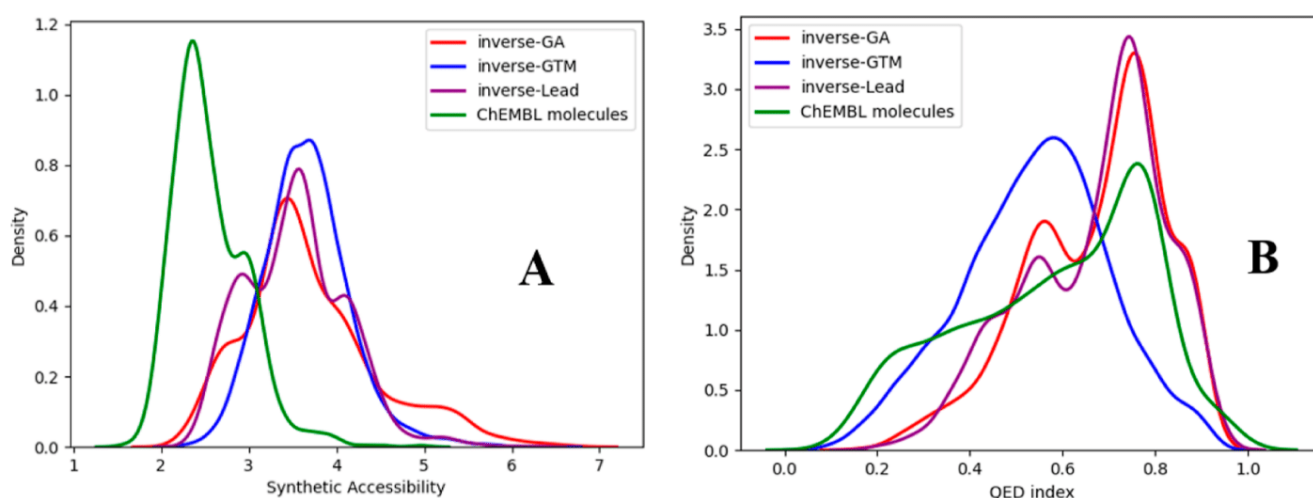
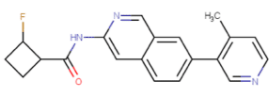
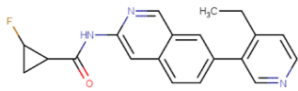
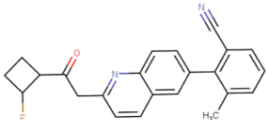
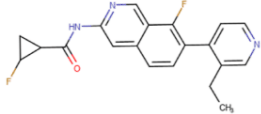


Figure 8. (A) Synthetic accessibility score for the four datasets calculated. (B) Quantitative estimate druglikeness index distribution for the three different datasets.

Table 3. Hits Found with Pharmacophore Models and Their Validation with Docking for Inverse-SVM (I–III) and Inverse-Lead (IV) Compounds

	Hits	calculated pK_i / activity rank	Pharmacophore	Docking score (LeadIT)
I		9.82 3 rd most active	Model 1	-33.2
II		9.34 / 16 th most active	Model 1	-31.4
III		9.18 25 th most active	Model 1	-23.27
IV		9.45 2 nd most active	Model 2	-31.8

Both structure-based and ligand-based approaches were applied in an effort to be as comprehensive as possible. The compounds present in the training set of the SVR model (821 compounds) were used for ligand-based model training. Ligand-based pharmacophores should reflect consensus features in highly active binders. Therefore, a threshold of $pK_i \geq 9$ was considered here to define “actives,” in contrast to the default $pK_i \geq 7$ defining “actives” in other contexts of this work (GTM landscape, docking studies—*vide infra*). In addition, only the inverse-SVR and inverse-lead compounds with predicted $pK_i \geq 9$ were screened. This subset of the initial generated compounds contains 39 inverse-SVR molecules and

8 inverse-lead compounds which makes 47 generated compounds in total.

For ligand-based pharmacophores, conformations for the training set compounds were calculated using the pre-loaded FAST parameters of the software. These settings returned a maximum of 25 conformations by compound. Ligand-based pharmacophores were built and clustered by LigandScout.⁴¹ Pharmacophore models were calculated for two clusters containing 78 and 5% (163 and 9 molecules, respectively) of all training set actives (model 1 and model 2, respectively). Different pharmacophore models were generated for each cluster using sets of 5 to 10 molecules.

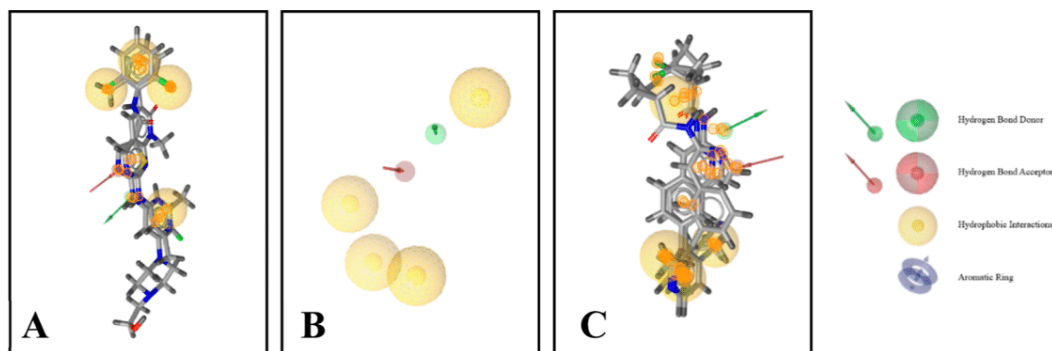


Figure 9. (A) Pharmacophore aligned with both PDB crystal structure ligands. (B) Shared pharmacophore model. (C) Selected inverse-SVR hits aligned with the pharmacophore model.

Structure-based pharmacophores were built based on PDB crystal structures of human proto-oncogene tyrosine-protein kinase ABL1. 2HZI and 2CQG crystal structures were used to generate the shared pharmacophore model which was screened against the 47 generated compounds for which $pK_i > 9$ was predicted.

3.3.4.1. Ligand-Based Pharmacophore. The screening of 47 inverse-SVR and inverse-lead molecules “hidden” in a set of 328 inactive decoys selected from the training set inactives allowed to understand if the two ligand-based pharmacophore models were selective enough to primarily focus on putative actives. If the considered pharmacophore models were observed to be as likely to match inactive decoys, it may be inferred that “matching” the pharmacophore model is no reliable indicator of putative activity against ChEMBL1862 likely because the ligand-based pharmacophore models are too generic (easily matched by random compounds).

Model 1 and model 2 returned, respectively, three and one hits. The hits align well with the pharmacophore model, and most features match as shown in⁴⁰ Figures S4 and S5 in the Supporting Information. Table 3 shows that the four hits have relatively high ranking among the most actives, one of them being the third predicted most active inverse-SVR compound and another the second most active inverse-lead compound.

3.3.4.2. Structure-Based Pharmacophore Screening. The shared pharmacophore model computed for two PDB structures (2HZI and 2GQG) is mostly based on hydrophobic interactions with one hydrogen bond donor and one hydrogen bond acceptor as shown in Figure 9B. The ligands contained in the PDB crystal structures are typically larger than inverse-SVR molecules. However, Figure 9A shows that crystalized ligands may include specific moieties not directly involved in binding. VS with the shared pharmacophore returned eight hits (see Table S2 in Supporting Information), four of which correspond to those found with ligand-based pharmacophores (Table 3). Notice that inverse-SVR compounds nicely match the pharmacophore, all while being smaller than the PDB ligands (see Figure 9C). These results show that the generated compounds are not only predicted active by the SVR models because they were optimized to do so but also fit the activity criteria of external validation methods like pharmacophore models. The fact that these three compounds were found by both methods and predicted highly active by the SVR model indicates that these compounds may be good candidates for further testing.

3.3.5. Validation of Inverse-SVR Compounds Using Ligand-To-Protein Docking. In the docking challenge, both

LeadIT and S4MPLE were able to predict the correct binding geometry of the native ligand of 2E2B (in protein-rigid redocking mode), and both were seen to significantly prioritize “actives” ($pK_i > 7$), for LeadIT, the area under the ROC curve obtained after redocking the 821 training set compounds (out of which only 816 could be docked) was of 0.77. S4MPLE also performed reasonably well (ROC AUC = 0.69 after the docking of 550 of the training set compounds, in random order). At that point, a quantitative correlation of $R^2 = 0.51$ between LeadIT and S4MPLE scores could be observed. Unfortunately, neither the LeadIT score ($R^2 = 0.21$, over 816 redocked compounds) nor S4MPLE ($R^2 = 0.16$ over the 550 ligands) can return docking scores that quantitatively correlate with the experimental pK_i values. We refer the reader to the Supporting Information section for a detailed analysis of the relationships between docking scores and actual, respective predicted pK_i values. It was observed that 76% of the experimentally confirmed training set actives ($pK_i > 7$) dock with LeadIT scores below or equal to -30 , whereas LeadIT score ≤ -25 would retrieve 92% of them. Therefore, the percentage of a library achieving LeadIT scores better (more negative) than this order of magnitude is a first rough estimate of how strongly ChEMBL-1862-focused that library is. Indeed, these percentages are significantly higher within the mixed collection of inverse-GTM and inverse-SVR leads (blue in Figure 10) than within the random subset of ZINC random decoys (orange bars). It should be noticed that only two out of three hits selected by pharmacophore models (molecules I, II, and IV, Table 2) were validated in docking calculations as actives: the LeadIT score for molecule III was larger than the threshold of -25 . The fact that the molecules I, II, and IV were found by both pharmacophore and docking methods as well as predicted highly active by the SVR model indicates that these compounds may be good candidates for further testing. We do not exclude that application of a docking score correlating with studied activity (e.g., that reported by Ahmed *et al.*⁶⁴) may better validate generated molecules.

4. CONCLUSIONS AND PERSPECTIVES

This article introduced a new type of architecture based on state-of-the-art deep learning method which is capable, given a descriptor type and successful training, to generate compounds possessing wanted activity and structural features from “seed” descriptor vectors—where the descriptor vectors are not “latent” vectors themselves produced by some encoder architecture but standard, state-of-the-art descriptors typically used in QSAR (here, ISIDA fragment counts). This provides

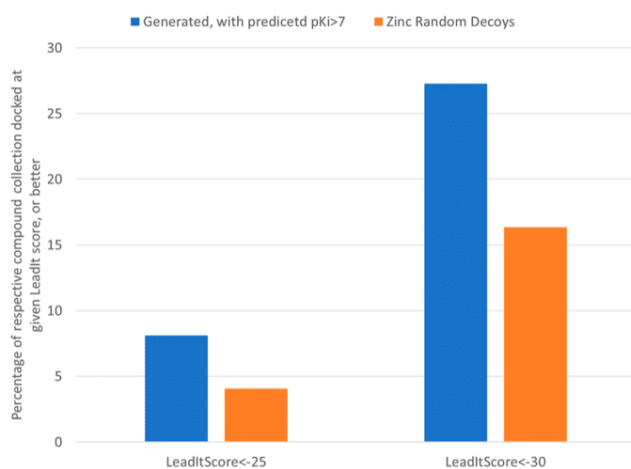


Figure 10. Percentages within the collection of inverse-GTM and inverse-SVR leads (blue) and the set of random ZINC decoys (orange) achieving LeadIt docking scores typical of experimentally validated actives of $pK_i > 7$.

an elegant solution for the inverse QSAR problem—the inference of novel molecular structures matching model-predicted high activity zones of the descriptor space. Finding descriptor “seeds” corresponding to aforementioned interesting zones has been herein addressed in two model-specific ways: evolutionary search for D vectors corresponding to high predicted affinity values (pK_i) according to SVR models or D vectors within the immediate neighborhood of GTM nodes preferentially populated by active compounds. Additionally, the descriptor vector generated for the highest affinity ligand from the training set was also used as a seed. Selecting only descriptor vectors associated with very high predicted affinity values (pK_i) equal or close to the best ever values reported in ChEMBL lead to inverse-SVR and inverse-lead molecules being structurally related to already existing top-active ChEMBL compounds—in the sense that they share significant common substructures, all while preserving their global originality. An external pharmacophore study performed on inverse-SVR compounds shows that several molecules with high predicted activity show good matches with existing active molecules in terms of pharmacophores. Selecting the vectors based on generative topographic mapping is focused on a binary, class-based definition of activity, and inverse-GTM molecules appear more diverse, all while predicted to have remarkable pK_i values by the SVR models (better than 100 nM, but not yet close to the top-active ChEMBL compounds). Original compounds of acceptable synthetic feasibility index could be readily obtained. Therefore, the inverse QSAR problem—fast discovery of original feasible compounds specifically selected for being predicted active by a given QSAR model—can be considered as conveniently solved, at least for the (rather widely used) class of fragment-based molecular descriptor-based QSAR models. Of course, the ultimate promise of prospective discovery of experimentally validated actives may only be kept if the “inversed” model lives up to its promises in terms of prediction—but this is an altogether different problem, which is not covered by the present, purely methodological work. It is clearly not expected to necessarily see inverse-QSAR *de novo* compounds automatically score well in docking if docking scores are decorrelated from the QSAR-predicted affinity estimator. In particular, fragment-count-based QSARs may overrate the importance of

given molecular fragments if the latter happen to appear by chance only within the structures of actives, thus establishing the mechanistically wrong shortcut “presence of key fragments → activity” simply because inactive counterexamples containing the same fragments in a different mutual configuration were not found at the training stage. ACoVAE-based approaches may, as seen in this work, readily suggest structures issued by recombining such key fragments—guaranteed to achieve high ratings by the parent QSAR model but not sure to still feature a global pharmacophore compatible with the target. The goal of this work was to present genuine solutions for the QSAR inversion problem based on “classical” fragment descriptors rather than on DNN-specific latent space vectors. Technically, this was a success, but it also clearly reveals that QSAR inversion *alone* is too risky a path to take in drug design: the actual pursuit of the synthesis efforts of sometimes challenging (but-granted-novel) structures may or may not pay, given the intrinsically incomplete and error-prone nature of QSAR models. However, if inverse QSAR is coupled with orthogonal activity prediction techniques, as done here, it can be observed that many of compounds alleged to be active by the initial QSAR models fail to pass the additional, independent activity assessment tests (pharmacophore matching, docking). This is no surprise because the consensus rate of chemoinformatics predictors based on premises as radically different as 2D-QSAR, pharmacophore screening and docking are typically very low. Nevertheless, we were successful in discovering some *de novo* structures which did pass the latter tests. This shows that the exploration of the initial inverse-QSAR-relevant chemical space is sufficient to visit areas in which not only the original QSAR model but also the alternative approaches indicate that biological activity is likely, pending experimental validation.

■ ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge at <https://pubs.acs.org/doi/10.1021/acs.jcim.2c01086>.

Detailed description of neural network architecture and some complementary results of QSAR and pharmacophore modeling (PDF)

■ AUTHOR INFORMATION

Corresponding Author

Alexandre Varnek – Laboratory of Chemoinformatics, UMR 7140 University of Strasbourg/CNRS, 67000 Strasbourg, France; orcid.org/0000-0003-1886-925X; Email: varnek@unistra.fr

Authors

William Bort – Laboratory of Chemoinformatics, UMR 7140 University of Strasbourg/CNRS, 67000 Strasbourg, France

Daniyar Mazitov – Laboratory of Chemoinformatics and Molecular Modeling, A. M. Butlerov Institute of Chemistry, Kazan Federal University, 420008 Kazan, Russia

Dragos Horvath – Laboratory of Chemoinformatics, UMR 7140 University of Strasbourg/CNRS, 67000 Strasbourg, France; orcid.org/0000-0003-0173-5714

Fanny Bonachera – Laboratory of Chemoinformatics, UMR 7140 University of Strasbourg/CNRS, 67000 Strasbourg, France

Arkadii Lin – Laboratory of Chemoinformatics, UMR 7140
University of Strasbourg/CNRS, 67000 Strasbourg, France
Gilles Marcou – Laboratory of Chemoinformatics, UMR 7140
University of Strasbourg/CNRS, 67000 Strasbourg, France;
orcid.org/0000-0003-1676-6708

Igor Baskin – Department of Material Science and
Engineering, Technion—Israel Institute of Technology,
3200003 Haifa, Israel

Timur Madzhidov – Laboratory of Chemoinformatics and
Molecular Modeling, A. M. Butlerov Institute of Chemistry,
Kazan Federal University, 420008 Kazan, Russia;
orcid.org/0000-0002-3834-6985

Complete contact information is available at:
<https://pubs.acs.org/10.1021/acs.jcim.2c01086>

Notes

The authors declare no competing financial interest.
Data and Software Availability: Developed code is available at the GitHub storage of the Laboratory of Chemoinformatics: <https://github.com/Laboratoire-de-Chemoinformatique/ACoVAE>. The data used for the model training and validation are available at <https://entrepot.recherche.data.gouv.fr/dataset.xhtml?persistentId=doi:10.57745/ILWSLF>.

ACKNOWLEDGMENTS

The High Performance Computing (HPC) Center of the Strasbourg University is acknowledged for technical support.

ABBREVIATIONS

ACoVAE, attention-based conditional variational autoencoder; DNN, deep neural network; GA, genetic algorithm; GTM, generative topographic map; QSA/PR, quantitative structure–activity/property relationships; SVR, support vector regression; VS, virtual screening

REFERENCES

- (1) Dudek, A.; Arodz, T.; Galvez, J. Computational Methods in Developing Quantitative Structure–Activity Relationships (QSAR): A Review. *Comb. Chem. High Throughput Screening* **2006**, *9*, 213–228.
- (2) Hansch, C. H.; Leo, A. J. *Exploring QSAR: Fundamentals and Applications in Chemistry and Biology*; American Chemical Society, 1995; Vol. 1.
- (3) Korotcov, A.; Tkachenko, V.; Russo, D.; Ekins, S. Comparison of Deep Learning With Multiple Machine Learning Methods and Metrics Using Diverse Drug Discovery Data Sets. *Mol. Pharm.* **2017**, *14*, 4462–4475.
- (4) Varnek, A.; Baskin, I. Machine Learning Methods for Property Prediction in Chemoinformatics: Quo Vadis? *J. Chem. Inf. Model.* **2012**, *52*, 1413–1437.
- (5) Jin, Y.; Wang, H.; Sun, C. *Introduction to Machine Learning*; Springer International Publishing, 2021; Vol. 975.
- (6) Smola, A.; Schölkopf, B. A tutorial on support vector regression. *Stat. Comput.* **2004**, *14*, 199–222.
- (7) Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32.
- (8) Skvortsova, M.; Fedyaev, K.; Palyulin, V.; Zefirov, N. Inverse Structure–Property Relationship Problem for the Case of a Correlation Equation Containing the Hosoya Index. *Dokl. Chem.* **2001**, *379*, 191–195.
- (9) Skvortsova, M.; Stankevich, I.; Zefirov, N. Generation of molecular structures of polycondensed benzenoid hydrocarbons using the randic index. *J. Struct. Chem.* **1992**, *33*, 416–422.
- (10) Skvortsova, M.; Baskin, I.; Slovokhotova, O.; Palyulin, V.; Zefirov, N. Inverse Problem in QSAR/QSPR Studies for the Case of Topological Indices Characterizing Molecular Shape (Kier Indices). *J. Chem. Inf. Comput. Sci.* **1993**, *33*, 630–634.
- (11) Lin, A.; Horvath, D.; Marcou, G.; Beck, B.; Varnek, A. Multi-task generative topographic mapping in virtual screening. *J. Comput.-Aided Mol. Des.* **2019**, *33*, 331–343.
- (12) Ripphausen, P.; Nisius, B.; Bajorath, J. State-of-the-art in ligand-based virtual screening. *Drug Discovery Today* **2011**, *16*, 372–376.
- (13) Hartenfeller, M.; Zettl, H.; Walter, M.; Rupp, M.; Reisen, F.; Proschak, E.; Weggen, S.; Stark, H.; Schneider, G. DOGS: Reaction-Driven de novo Design of Bioactive Compounds. *PLoS Comput. Biol.* **2012**, *8*, No. e1002380.
- (14) Mauser, H.; Guba, W. Recent developments in de novo design and scaffold hopping. *Curr. Opin. Drug Discovery Dev.* **2008**, *11*, 365–374.
- (15) Hartenfeller, M.; Proschak, E.; Schüller, A.; Schneider, G. Concept of Combinatorial De Novo Design of Drug-like Molecules by Particle Swarm Optimization. *Chem. Biol. Drug Des.* **2008**, *72*, 16–26.
- (16) Sattarov, B.; Baskin, I.; Horvath, D.; Marcou, G.; Bjerrum, E.; Varnek, A. De Novo Molecular Design by Combining Deep Autoencoder Recurrent Neural Networks with Generative Topographic Mapping. *J. Chem. Inf. Model.* **2019**, *59*, 1182–1196.
- (17) Gómez-Bombarelli, R.; Wei, J.; Duvenaud, D.; Hernández-Lobato, J.; Sánchez-Lengeling, B.; Sheberla, D.; Aguilera-Iparraguirre, J.; Hirzel, T.; Adams, R.; Aspuru-Guzik, A. Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. *ACS Cent. Sci.* **2018**, *4*, 268–276.
- (18) Zhou, Z.; Kearnes, S.; Li, L.; Zare, R.; Riley, P. Optimization of Molecules via Deep Reinforcement Learning. *Sci. Rep.* **2019**, *9*, 10752.
- (19) Segler, M. H. S.; Kogej, T.; Tyrchan, C.; Waller, M. P. Generating Focused Molecule Libraries for Drug Discovery with Recurrent Neural Networks. *ACS Cent. Sci.* **2018**, *4*, 120–131.
- (20) Olivecrona, M.; Blaschke, T.; Engkvist, O.; Chen, H. Molecular De Novo Design through Deep Reinforcement Learning. *J. Cheminf.* **2017**, *9*, 48.
- (21) Prykhodko, O.; Johansson, S.; Kotsias, P.; Arús-Pous, J.; Bjerrum, E.; Engkvist, O.; Hongming, C. A de novo molecular generation method using latent vector based generative adversarial network. *J. Cheminf.* **2019**, *11*, 74.
- (22) Arús-Pous, J.; Patronov, A.; Bjerrum, E.; Tyrchan, C.; Raymond, J.-L.; Hongming, C.; Engkvist, O. SMILES-based deep generative scaffold decorator for de-novo drug design. *J. Cheminf.* **2020**, *12*, 38.
- (23) Cova, T.; Pais, A. Deep Learning for Deep Chemistry: Optimizing the Prediction of Chemical Patterns. *Front. Chem.* **2019**, *7*, 809.
- (24) Gupta, M. K.; Gupta, S.; Rawal, R. Impact of Artificial Neural Networks in QSAR and Computational Modeling. In *Artificial Neural Network for Drug Design, Delivery and Disposition*; Academic Press, 2016; pp 153–179.
- (25) Mitchell, J. B. O. Machine learning methods in chemoinformatics. *Wiley Interdiscip. Rev.: Comput. Mol. Sci.* **2014**, *4*, 468–481.
- (26) Fjodorova, N.; Vračko, M.; Jezierska, A.; Novič, M. Counter propagation artificial neural network categorical models for prediction of carcinogenicity for non-congeneric chemicals. *SAR QSAR Environ. Res.* **2010**, *21*, 57–75.
- (27) Ajmani, S.; Viswanadhan, V. N. A Neural Network-Based QSAR Approach for Exploration of Diverse Multi-Tyrosine Kinase Inhibitors and its Comparison with a Fragment-Based Approach. *Curr. Comput.-Aided Drug Des.* **2013**, *9*, 482–490.
- (28) Myint, K.-Z.; Wang, L.; Tong, Q.; Xie, X. Molecular Fingerprint-Based Artificial Neural Networks QSAR for Ligand Biological Activity Predictions. *Mol. Pharmaceutics* **2012**, *9*, 2912–2923.
- (29) Rana, A.; Rawat, A.; Bijalwan, A.; Bahuguna, H. Application of Multi Layer (Perceptron) Artificial Neural Network in the Diagnosis System: A Systematic Review. In *2018 International Conference on Research in Intelligent and Computing in Engineering (RICE)*; IEEE, 2018; pp 1–6.

- (30) Baskin, I. I.; Palyulin, V. A.; Zefirov, N. S. Neural Networks in Building QSAR Models. In *Artificial Neural Networks: Methods and Applications*; Livingstone, D. J., Ed.; Humana Press, 2009; pp 133–154.
- (31) Sabando, M. V.; Ponzoni, I.; Milios, E.; Soto, A. Using molecular embeddings in QSAR modeling: Does it make a difference? *Briefings Bioinf.* **2022**, *23*, bbab365.
- (32) Muratov, E. N.; Bajorath, J.; Sheridan, R.; Tetko, I.; Filimonov, D.; Poroikov, V.; Oprea, T.; Baskin, I.; Varnek, A.; Roitberg, A.; Isayev, O.; Curtalolo, S.; Fourches, D.; Cohen, Y.; Aspuru-Guzik, A.; Winkler, D.; Agrafiotis, D.; Cherkasov, A.; Tropsha, A. QSAR without borders. *Chem. Soc. Rev.* **2020**, *49*, 3525–3564.
- (33) Sousa, T.; Correia, J.; Pereira, V.; Rocha, M. Generative Deep Learning for Targeted Compound Design. *J. Chem. Inf. Model.* **2021**, *61*, S343–S361.
- (34) Weininger, D.; Weininger, A.; Weininger, J. Algorithm for generation of unique SMILES notation. *J. Chem. Inf. Comput. Sci.* **1989**, *29*, 97–101.
- (35) Kotsias, P.-C.; Arús-Pous, J.; Chen, C.; Engkvist, O.; Tyrchan, C.; Bjerrum, E. Direct steering of de novo molecular generation with descriptor conditional recurrent neural networks. *Nat. Mach. Intell.* **2020**, *2*, 254–265.
- (36) Ucak, U. V.; Ashyrmamatov, I.; Lee, J. Reconstruction of lossless molecular representations, SMILES and SELFIES, from fingerprints. *ChemRxiv* **2022**, DOI: 10.26434/chemrxiv-2022-tqv76-v2.
- (37) Varnek, A.; Fourches, D.; Horvath, D.; Klimchuk, O.; Gaudin, C.; Vayer, P.; Solov'ev, V.; Hoonakker, F.; Tetko, I.; Marcou, G. ISIDA—Platform for Virtual Screening Based on Fragment and Pharmacophoric Descriptors. *Curr. Comput.-Aided Drug Des.* **2008**, *4*, 191–198.
- (38) Varnek, A.; Fourches, D.; Solov'ev, V.; Klimchuk, O.; Ouadi, A.; Billard, I. Successful "In Silico" Design of New Efficient Uranyl Binders. *Solvent Extr. Ion Exch.* **2007**, *25*, 433–462.
- (39) Sidorov, P.; Gaspar, H.; Marcou, G.; Varnek, A.; Horvath, D. Mappability of drug-like space: Towards a polypharmacologically competent map of drug-relevant compounds. *J. Comput.-Aided Mol. Des.* **2015**, *29*, 1087–1108.
- (40) Ruggiu, F.; Marcou, G.; Varnek, A.; Horvath, D. ISIDA Property-labelled fragment descriptors. *Mol. Inf.* **2010**, *29*, 855–868.
- (41) Wolber, G.; Langer, T. LigandScout: 3-D Pharmacophores Derived from Protein-Bound Ligands and Their Use as Virtual Screening Filters. *J. Chem. Inf. Model.* **2005**, *45*, 160–169.
- (42) LeadIT; BioSolveIT GmbH: Sankt Augustin, Germany, 2022. www.biosolveit.de.
- (43) Hoffer, L.; Chira, C.; Marcou, G.; Varnek, A.; Horvath, D. S4MPLE—Sampler for Multiple Protein-Ligand Entities: Methodology and Rigid-Site Docking Benchmarking. *Molecules* **2015**, *20*, 8997–9028.
- (44) Mehta, S.; Ghazvininejad, M.; Iyer, S.; Zettlemoyer, L.; Hajishirzi, H. DeLighT: Deep and Light-weight Transformer. **2020**, arxiv:2008.00623. arXiv preprint.
- (45) Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. *31st Conference on Neural Information Processing Systems 2017*, 2017; pp 5999–6009.
- (46) Lin, Z.; Winata, G. I.; Xu, P.; Liu, Z.; Fung, P. Variational Transformers for Diverse Response Generation. **2020**, arxiv:2003.12738. arXiv preprint.
- (47) Davidson, T. R.; Falorsi, L.; De Cao, N.; Kipf, T.; Tomczak, J. M. Hyperspherical variational auto-encoders. *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*, 2018; Vol. 2, pp 856–865.
- (48) De Cao, N.; Aziz, W. The Power Spherical Distribution. **2020**, arxiv:2006.04437. arXiv preprint.
- (49) Mehta, S.; Ghazvininejad, M.; Iyer, S.; Zettlemoyer, L.; Hajishirzi, H. DeLighT: Deep and Light-weight Transformer. **2021**, arxiv:2008.00623. arXiv preprint.
- (50) Hendrycks, D.; Gimpel, K. Gaussian Error Linear Units (GELUs). **2016**, arxiv:1606.08415. arXiv preprint.
- (51) Chieng, H. H.; Wahid, N.; Pauline, O.; Perla, S. R. K. Flatten-swish: A thresholded relu-swish-like activation function for deep learning. *Int. J. Adv. Intell. Inform.* **2018**, *4*, 76–86.
- (52) Horvath, D.; Brown, J.; Marcou, G.; Varnek, A. An Evolutionary Optimizer of libsvm Models. *Challenges* **2014**, *5*, 450–472.
- (53) Bishop, C. M.; Svensen, M.; Williams, C. GTM: The Generative Topographic Mapping. *Neural Comput.* **1998**, *10*, 215–234.
- (54) Dragos, H.; Gilles, M.; Alexandre, V. Predicting the Predictability: A Unified Approach to the Applicability Domain Problem of QSAR Models. *J. Chem. Inf. Model.* **2009**, *49*, 1762–1776.
- (55) Horvath, D.; Jeandenans, C. Neighborhood behavior of in silico structural spaces with respect to in vitro activity spaces—A novel understanding of the molecular similarity principle in the context of multiple receptor binding profiles. *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 680–690.
- (56) Casciuc, I.; Zabolotna, Y.; Horvath, D.; Marcou, G.; Bajorath, J.; Varnek, A. Virtual Screening with Generative Topographic Maps: How Many Maps Are Required? *J. Chem. Inf. Model.* **2019**, *59*, S64–S72.
- (57) Nugmanov, R. I.; Mukhametgaleev, R.; Akhmetshin, T.; Gimadiev, T.; Afonina, V.; Madzhidov, T.; Varnek, A. CGRtools: Python Library for Molecule, Reaction, and Condensed Graph of Reaction Processing. *J. Chem. Inf. Model.* **2019**, *59*, 2516–2521.
- (58) Ertl, P.; Schuffenhauer, A. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *J. Cheminf.* **2009**, *1*, 8.
- (59) Kosugi, T.; Ohue, M. Quantitative Estimate Index for Early-Stage Screening of Compounds Targeting Protein-Protein Interactions. *Int. J. Mol. Sci.* **2021**, *22*, 10925.
- (60) ChemAxon Standardizer, Version 5.12; ChemAxon, Ltd.: Budapest, Hungary, 2012.
- (61) O'Boyle, N.; Dalke, A. DeepSMILES: An Adaptation of SMILES for Use in Machine-Learning of Chemical Structures. *ChemRxiv* **2018**, DOI: 10.26434/chemrxiv.7097960.v1.
- (62) Berenger, F.; Tsuda, K. Molecular generation by Fast Assembly of (Deep)SMILES fragments. *J. Cheminf.* **2021**, *13*, 88.
- (63) Krenn, M.; Häse, F.; Nigam, A.; Friederich, P.; Aspuru-Guzik, A. Self-referencing embedded strings (SELFIES): A 100% robust molecular string representation. *Mach. Learn.: Sci. Technol.* **2020**, *1*, 045024.
- (64) Ahmed, S.; Prabakar, A. E.; Saxena, A. K. Molecular docking-based interactions in QSAR studies on Mycobacterium tuberculosis ATP synthase inhibitors. *SAR QSAR Environ. Res.* **2022**, *33*, 289–305.

5.5.1 Summary

A novel Conditional Variational Autoencoder (ACoVAE) was successfully developed and specially adapted to ISIDA descriptor vectors. The combination of a GRU-based Variational Encoder with a state-of-the-art Attention-based decoder was trained and used for the generation of molecules with preferred structural features. The model achieved great Tanimoto Similarity when generating from known and unknown ISIDA descriptor vectors.

In order to generate compounds active against the ChEMBL1862 target, different condition selection methods were employed, GTM-based, GA/SVR-based and active-based. Active-based and SVR-based sampling resulted in the generation of molecules predicted highly active by the activity models (about 40-50% of them had $pK_i \geq 7$), with some of them going up to $pK_i \geq 10$, with very specific and similar structures, which could be considered a very focused dataset. GTM-based sampling resulted in more varied structural features with on average a lower pK_i value (only 7% had a $pK_i \geq 7$).

The differences between the two generated sets can be explained by the two condition selection methods: SVR-based and active-based select only the best vectors which are naturally close to the actual active. Small differences in the descriptor vector imply small changes in the predicted pK_i and structure which creates a focused dataset. The GTM-based selection method selects indiscriminately any vector with an activity ≥ 7 which is considered active. The nodes are then a lot more varied in terms of structural features and less oriented on pure activity. Both methods were found to generate potential actives however.

So-called Inverse-SVR and Inverse-Lead compounds were validated using a ligand-based and structure-based pharmacophore study as well as docking. Out of the 47 potential hits, 4 were found to be a match by both pharmacophore methods and had reasonable docking scores hinting at a high potential for activity. The model was therefore able to invert the QSAR process and generate active compounds from desirable structural vectors. The model could also be adapted to accept property vectors in combination with structural descriptors to specify conditions even further.

6 General Conclusion & Perspectives

Towards a better understanding of Deep Neural Network Latent Spaces

The introduction and complexification of Deep Learning methods created a new branch of drug design. The generative capabilities and efficiency of neural networks in terms of de novo design have made deep architectures very popular. The ability of these models to create and navigate latent spaces in search of compounds of interest have allowed the discovery of active compounds. However, the complex mathematical equations and the nature of the exact workings of these models remains blurry in the field of Chemistry. There is a clear understanding of latent spaces based on structural descriptors, since they are interpretable and readable, and the rules were carefully hand-crafted and designed. However, latent spaces that neural networks create using their interpretation of character-based or graph-based representations of molecular structures do not follow the same rules.

Therefore, one of the objectives of this thesis was to obtain a better grasp on the construction and organization of neural network latent spaces, and in particular latent spaces of Autoencoders which remain one of the most popular architectures in terms of molecular generation. A LSTM-based Vanilla Autoencoder, based on SMILES strings obtained from the ChEMBL database was trained and its latent space mapped using GTM. Its generative capabilities were tested by sampling compounds in every point of latent space mapped by said GTM. The AE was able to create molecules for each node which were similar in terms of structure and properties to the already existing ChEMBL molecules, showing the already great learning power and adaptability of one of the most basic deep architectures of seq2seq models. The comparison of property landscapes built on ChEMBL and generated compounds showed that the models are able to reproduce the general outlook and organization of its latent space when mapped by GTM. These models could therefore be used to fill gaps and holes, or less dense areas of chemical space which is an important aspect in the constant search for interesting new compounds.

However, the latent space construction depends on the interpretation of the SMILES string by the model which is order-dependent and can lead to some inconsistencies and activity cliffs. The generative capabilities also depend greatly on the complexity of the structural features and the density of the training data, making these latent spaces quite different from structural spaces. The multiple failed attempts to link latent space with ISIDA structural space shows that

they are constructed differently, and even though both work in terms of active separations, they are in fact incompatible, as proven by the Hilbert-Schmidt Criterion.

In the future, it would be interesting to perform the same kind of analysis on more complex architectures, like VAEs, CVAEs, Attention-based VAEs and Transformers to compare the results. Small differences in parameters, model architecture, input form result in different chemical spaces so the task of finding generalized rules that govern the organization of these latent spaces is complex. However, by having a deeper understanding of how these algorithms encode and decode chemical information and how this information is interpreted, it could be feasible to find general tendencies like the ones shown in this project to facilitate the exploration of said chemical spaces.

Still, the capabilities of this model could be harnessed by modifying it to accept Condensed Graph of Reactions in order to map the latent space of reactions from the USPTO database and generate potentially new and feasible transformations. Methods of novelty detection and reaction classification were developed using Reaction Centres and Reaction Environments.

Inverting the classical QSAR algorithm

Entering needed properties and structural features and obtaining several potent molecules corresponding to the given restrictions would simplify and accelerate the drug design process significantly. A combination of the generative power of neural networks and the efficient construction and organization of structural descriptor spaces could provide with great candidate features selection and an ability to generate molecules corresponding to these features which would amount to inverse-QSAR.

To that extent, several methods based on Neural Networks and GTM were tested to try and link the latent space of a generative Autoencoder to the structural space of ISIDA descriptors. A basic LSTM-based translator from SMI2ISIDA was tested as a building block to the bigger model aiming to directly translate ISIDA vectors to corresponding SMILES but failed due to the incapacity of the model to accurately enumerate the ISIDA fragments with high deviation. A Multimodal Boltzmann Machine was developed as a more complex solution to the previous issue, however showed similar weaknesses during training: ISIDA vectors could not be precisely reproduced which is capital for this task. Stargate-GTM and direct GTM links were applied as an alternative not requiring the processing of ISIDA vectors by sequence-based

algorithms but failed due to the incompatibility of the two spaces. Indeed, the evaluation of the Hilbert-Schmidt Criterion between ISIDA and latent spaces showed that those spaces are completely independent meaning no statistical link can be made between them, rendering machine learning methods like GTM essentially useless.

The many attempts did confirm the difference between classical structural spaces and SMILES-based latent spaces and triggered the development of a novel CVAE architecture, which applied ISIDA vectors as condition to VAE latent vectors, eliminating the need for an external link. The model, based on the latest developments in Deep Learning like Multi-Head Attention and a GRU-based encoder was successfully used to generate compounds from selected “seeds”. These seeds resulted from the exploration of chemical space using different methods, Genetic Algorithm coupled with activity prediction, and Generative Topographic Mapping. Both methods produced different but equally interesting results, with SVR-based generation giving the most potent generated molecules, as confirmed by pharmacophore, and docking studies.

The ACoVAE model, coupled with chemical space exploration techniques allowed the reversing of the classical QSAR method and the generation of active molecules from selected structural features. The model could quite simply be adapted to work with any other structural descriptor and/or property vectors and could even swap SMILES to CGR to function with reactions. Coupling this with broad, versatile chemical space visualization tools^[164] could be a powerful method of chemical space exploration allowing the discovery of new compounds in charted zones, and even the discovery of new uncharted zones of chemical space. Taking a step back and looking at the entire drug discovery process, this tool could also be a part of a larger drug discovery “machine” where every step could be automated, from the setting of structural and physico-chemical properties to fit a given target, to chemical space exploration, synthesis planning and even chemical synthesis using the developments in chemical automation.

7 List of Abbreviations

AAE	Adversarial AutoEncoder
AE	Autoencoder
AI	Artificial Intelligence
ANN	Artificial Neural Network
CGR	Condensed Graph of Reaction
CRNN	Conditional Recurrent Neural Network
CVAE	Conditional Variational Autoencoder
DL	Deep Learning
ECFP	Extended Connectivity FingerPrint
GAN	Generative Adversarial Network
GRU	Gated Recurrent Unit
GTM	Generative Topographic Mapping
HSIC	Hilbert-Schmidt Independence Criterion
InChI	International Chemical Identifier
IUPAC	International Union of Pure and Applied Chemistry
LDA	Linear Discriminant Analysis
LLh	Log Likelihood
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MDBM	Multimodal Deep Boltzmann Machine
MHA	Multi-Head Attention
ML	Machine Learning

MSE	Mean Squared Error
NLP	Natural Language Processing
NMR	Nuclear Magnetic Resonance
PCA	Principal Component Analysis
PSO	Particle Swarm Optimization
QSAR	Quantitative Structure-Activity Relationship
QSPR	Quantitative Structure-Property Relationship
RBF	Radial Basis Function
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
RNN	Recurrent Neural Network
SE	Shannon Entropy
SELFIES	SELF-referencing Embedded String
Seq2Seq	Sequence-to-Sequence
S-GTM	Stargate-GTM
SMILES	Simplified Molecular-Input Line-Entry System
SOM	Self-Organizing Map
SVR	Support Vector Regression
SVM	Support Vector Machine
TF	Teacher Forcing
TL	Transfer Learning
t-SNE	t-distributed Stochastic Neighbour Embedding
UM	Universal Map
USPTO	United States Patent and Trademark Office
VAE	Variational Autoencoder
VS	Virtual Screening

8 References

1. Polishchuk, P. G., Madzhidov, T. I. & Varnek, A. Estimation of the size of drug-like chemical space based on GDB-17 data. *J. Comput. Aided. Mol. Des.* **27**, 675–679 (2013).
2. Tong, X., Liu, X., Tan, X., Li, X., Jiang, J., Xiong, Z., Xu, T., Jiang, H., Qiao, N. & Zheng, M. Generative Models for de Novo Drug Design. *J. Med. Chem.* **64**, 14011–14027 (2021).
3. Lopez Pinaya, W. H., Vieira, S., Garcia-Dias, R. & Mechelli, A. Autoencoders. *Mach. Learn. Methods Appl. to Brain Disord.* 193–208 (2020).
4. Lim, J., Ryu, S., Kim, J. W. & Kim, W. Y. Molecular generative model based on conditional variational autoencoder for de novo molecular design. *J. Cheminform.* **10**, 31 (2018).
5. Polykovskiy, D., Zhebrak, A., Vetrov, D., Ivanenkov, Y., Aladinskiy, V., Mamoshina, P., Bozdaganyan, M., Aliper, A., Zhavoronkov, A. & Kadurin, A. Entangled Conditional Adversarial Autoencoder for de Novo Drug Discovery. *Mol. Pharm.* **15**, 4398–4405 (2018).
6. Prykhodko, O., Johansson, S. V., Kotsias, P. C., Arús-Pous, J., Bjerrum, E. J., Engkvist, O. & Chen, H. A de novo molecular generation method using latent vector based generative adversarial network. *J. Cheminform.* **11**, 74 (2019).
7. Bishop, C. M., Svensén, M. & Williams, C. K. I. GTM: The Generative Topographic Mapping. *Neural Comput.* **10**, 215–234 (1998).
8. Ruggiu, F., Marcou, G., Varnek, A. & Horvath, D. ISIDA Property-Labelled Fragment Descriptors. *Mol. Inform.* **29**, 855–68 (2010).
9. Nugmanov, R. I., Mukhametgaleev, R. N., Akhmetshin, T., Gimadiev, T. R., Afonina, V. A., Madzhidov, T. I. & Varnek, A. CGRtools: Python Library for Molecule, Reaction, and Condensed Graph of Reaction Processing. *J. Chem. Inf. Model.* **59**, 2516–2521 (2019).
10. Lowe, D. M. (Thesis) Extraction of chemical structures and reactions from the literature. (University of Cambridge, 2012).
11. Hinton, G. E. *A practical guide to training restricted boltzmann machines. Lecture Notes in Computer Science* (2012).
12. Gaspar, H. A., Baskin, I. I., Marcou, G., Horvath, D. & Varnek, A. Stargate GTM: Bridging Descriptor and Activity Spaces. *J. Chem. Inf. Model.* **55**, 2403–2410 (2015).
13. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. & Polosukhin, I. Attention Is All You Need. *Adv. Neural Inf. Process. Syst.* **2017**, 5999–6009 (2017).
14. Casciuc, I., Zabolotna, Y., Horvath, D., Marcou, G., Bajorath, J. & Varnek, A. Virtual Screening with Generative Topographic Maps: How Many Maps Are Required? *J. Chem. Inf. Model.* **59**, 564–572 (2019).

15. Wetzel, S., Klein, K., Renner, S., Rauh, D., Oprea, T. I., Mutzel, P. & Waldmann, H. Interactive exploration of chemical space with Scaffold Hunter. *Nat. Chem. Biol.* **5**, 581–583 (2009).
16. Van Deursen, R. & Reymond, J. L. Chemical space travel. *ChemMedChem* **2**, 636–640 (2007).
17. Oprea, T. I. Chemical space navigation in lead discovery. *Curr. Opin. Chem. Biol.* **6**, 384–389 (2002).
18. Reymond, J.-L. The Chemical Space Project. *Acc. Chem. Res.* **48**, 722–730 (2015).
19. Coley, C. W. Defining and Exploring Chemical Spaces. *Trends Chem.* **3**, 133–145 (2021).
20. Varnek, A. & Baskin, I. I. Chemoinformatics as a Theoretical Chemistry Discipline. *Mol. Inform.* **30**, 20–32 (2011).
21. Llanos, E. J., Leal, W., Luu, D. H., Jost, J., Stadler, P. F. & Restrepo, G. Exploration of the chemical space and its three historical regimes. *Proc. Natl. Acad. Sci. U. S. A.* **116**, 12660–12665 (2019).
22. Hogan, J. C. Combinatorial chemistry in drug discovery. *Nat. Biotechnol.* **15**, 328–330 (1997).
23. Pereira, D. A. & Williams, J. A. Origin and evolution of high throughput screening. *Br. J. Pharmacol.* **152**, 53–61 (2007).
24. Seidel, T., Ibis, G., Bendix, F. & Wolber, G. Strategies for 3D pharmacophore-based virtual screening. *Drug Discov. Today Technol.* **7**, (2010).
25. Sun, H. Pharmacophore-based virtual screening. *Curr. Med. Chem.* **15**, 1018–1024 (2008).
26. Kontoyianni, M. Docking and virtual screening in drug discovery. *Methods Mol. Biol.* **1647**, 255–266 (2017).
27. Kitchen, D. B., Decornez, H., Furr, J. R. & Bajorath, J. Docking and scoring in virtual screening for drug discovery: Methods and applications. *Nat. Rev. Drug Discov.* **3**, 935–949 (2004).
28. Neves, B. J., Braga, R. C., Melo-Filho, C. C., Moreira-Filho, J. T., Muratov, E. N. & Andrade, C. H. QSAR-based virtual screening: Advances and applications in drug discovery. *Front. Pharmacol.* **9**, (2018).
29. Achary, P. G. R. Applications of Quantitative Structure-Activity Relationships (QSAR) based Virtual Screening in Drug Design: A Review. *Mini-Reviews Med. Chem.* **20**, 1375–1388 (2020).
30. Ferreira, L. T., Borba, J. V. B., Moreira-Filho, J. T., Rimoldi, A., Andrade, C. H. & Costa, F. T. M. Qsar-based virtual screening of natural products database for identification of potent antimalarial hits. *Biomolecules* **11**, 1–12 (2021).
31. Menchon, G., Maveyraud, L. & Czaplicki, G. Molecular dynamics as a tool for virtual ligand screening. *Methods Mol. Biol.* **1762**, 145–178 (2018).

32. Nichols, S. E., Baron, R. & McCammon, J. A. On the use of molecular dynamics receptor conformations for virtual screening. *Methods Mol. Biol.* **819**, 93–103 (2012).
33. Ertl, P. Cheminformatics Analysis of Organic Substituents: Identification of the Most Common Substituents, Calculation of Substituent Properties, and Automatic Identification of Drug-like Bioisosteric Groups. *J. Chem. Inf. Comput. Sci.* **43**, 374–380 (2002).
34. Drew, K. L. M., Baiman, H., Khwaounjoo, P., Yu, B. & Reynisson, J. Size estimation of chemical space: How big is it? *J. Pharm. Pharmacol.* **64**, 490–495 (2012).
35. Shivanyuk, A. N., Ryabukhin, S. V., Bogolyubsky, A. V., Mykytenko, D. M., Chupryna, A. A., Heilman, W., Kostyuk, A. N. & Tolmachev, A. A. Enamine real database: making chemical diversity real. *Chim. Oggi* **25**, 58–59 (2007).
36. Lipinski, C. & Hopkins, A. Navigating chemical space for biology and medicine. *Nature* **432**, 855–861 (2004).
37. Reymond, J. L., Van Deursen, R., Blum, L. C. & Ruddigkeit, L. Chemical space as a source for new drugs. *Medchemcomm* **1**, 30–38 (2010).
38. Schneider, G. & Fechner, U. Computer-based de novo design of drug-like molecules. *Nat. Rev. Drug Discov.* **4**, 649–663 (2005).
39. Mauser, H. & Guba, W. Recent developments in de novo design and scaffold hopping. *Curr. Opin. Drug Discov. Devel.* **11**, 365–374 (2008).
40. Hartenfeller, M. & Schneider, G. De novo drug design. *Methods Mol. Biol.* **672**, 299–323 (2011).
41. Danziger, D. J. & Dean, P. M. Automated site-directed drug design: a general algorithm for knowledge acquisition about hydrogen-bonding regions at protein surfaces. *Proc. R. Soc. London. Ser. B, Biol. Sci.* **236**, 101–113 (1989).
42. Lewis, R. A., Roe, D. C., Huang, C., Ferrin, T. E., Langridge, R. & Kuntz, I. D. Automated site-directed drug design using molecular lattices. *J. Mol. Graph.* **10**, 66–78 (1992).
43. Lewis, R. A. Automated site-directed drug design: Approaches to the formation of 3D molecular graphs. *J. Comput. Aided. Mol. Des.* **4**, 205–210 (1990).
44. Bohacek, R. S. & McMartin, C. Multiple Highly Diverse Structures Complementary to Enzyme Binding Sites: Results of Extensive Application of a de Novo Design Method Incorporating Combinatorial Growth. *J. Am. Chem. Soc.* **116**, 5560–5571 (1994).
45. Loving, K., Alberts, I. & Sherman, W. Computational Approaches for Fragment-Based and De Novo Design. *Curr. Top. Med. Chem.* **10**, 14–32 (2010).
46. Maveyraud, L. & Mourey, L. Protein X-ray crystallography and drug discovery. *Molecules* **25**, (2020).
47. Kendrew, J. C., Bodo, G., Dintzis, H. M., Parrish, R. G., Wyckoff, H. & Phillips, D. C. A Three-Dimensional Model of the Myoglobin Molecule Obtained by X-Ray Analysis. *Nature* **181**, 662–666 (1958).

48. Pickford, A. R. & Campbell, I. D. NMR studies of modular protein structures and their interactions. *Chem. Rev.* **104**, 3557–3565 (2004).
49. Benjin, X. & Ling, L. Developments, applications, and prospects of cryo-electron microscopy. *Protein Sci.* **29**, 872–882 (2020).
50. Boland, A., Chang, L. & Barford, D. The potential of cryo-electron microscopy for structure-based drug design. *Essays Biochem.* **61**, 543–560 (2017).
51. Waszkowycz, B., Clark, D. E., Frenkel, D., Li, J., Murray, C. W., Robson, B. & Westhead, D. R. PRO_LIGAND: An Approach to de Novo Molecular Design. 2. Design of Novel Molecules from Molecular Field Analysis (MFA) Models and Pharmacophores. *J. Med. Chem.* **37**, 3994–4002 (1994).
52. Gugerty, L. Newell and Simon’s logic theorist: Historical background and impact on cognitive modeling. *Proc. Hum. Factors Ergon. Soc.* **50**, 880–884 (2006).
53. Crevier, D. *AI: The Tumultuous History of the Search for Artificial Intelligence*. (BasicBooks, 1993).
54. Nordhaus, W. D. The Progress of Computing. *Cowles Found. Discuss. Pap.* (2001).
55. Chan, H. C. S., Shan, H., Dahoun, T., Vogel, H. & Yuan, S. Advancing Drug Discovery via Artificial Intelligence. *Trends Pharmacol. Sci.* **40**, 592–604 (2019).
56. Zhu, H. Big Data and Artificial Intelligence Modeling for Drug Discovery. *Annu. Rev. Pharmacol. Toxicol.* **60**, 573–589 (2020).
57. Yang, X., Wang, Y., Byrne, R., Schneider, G. & Yang, S. Concepts of Artificial Intelligence for Computer-Assisted Drug Discovery. *Chem. Rev.* **119**, 10520–10594 (2019).
58. Otter, D. W., Medina, J. R. & Kalita, J. K. A Survey of the Usages of Deep Learning for Natural Language Processing. *IEEE Trans. Neural Networks Learn. Syst.* **32**, 604–624 (2021).
59. Liu, L., Wang, Y. & Chi, W. Image Recognition Technology Based on Machine Learning. *IEEE Access* (2021).
60. Ruff, K. M. & Pappu, R. V. AlphaFold and Implications for Intrinsically Disordered Proteins. *J. Mol. Biol.* **433**, (2021).
61. Chen, H., Engkvist, O., Wang, Y., Olivecrona, M. & Blaschke, T. The rise of deep learning in drug discovery. *Drug Discov. Today* **23**, 1241–1250 (2018).
62. Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. & Bengio, Y. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* 1724–1734 (Association for Computational Linguistics, 2014).
63. Serban, I. V., Sordoni, A., Bengio, Y., Courville, A. & Pineau, J. Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models. in *30th AAAI Conference on Artificial Intelligence* 3776–3783 (AAAI press, 2015).

64. Weininger, D. SMILES, a Chemical Language and Information System: 1: Introduction to Methodology and Encoding Rules. *J. Chem. Inf. Comput. Sci.* **28**, 31–36 (1988).
65. Segler, M. H. S., Kogej, T., Tyrchan, C. & Waller, M. P. Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS Cent. Sci.* **4**, 120–131 (2018).
66. Olivecrona, M., Blaschke, T., Engkvist, O. & Chen, H. Molecular de-novo design through deep reinforcement learning. *J. Cheminform.* **9**, 48 (2017).
67. Gupta, A., Müller, A. T., Huisman, B. J. H., Fuchs, J. A., Schneider, P. & Schneider, G. Generative Recurrent Networks for De Novo Drug Design. *Mol. Inform.* **37**, (2018).
68. Blaschke, T., Olivecrona, M., Engkvist, O., Bajorath, J. & Chen, H. Application of Generative Autoencoder in De Novo Molecular Design. *Mol. Inform.* **37**, (2018).
69. Putin, E., Asadulaev, A., Ivanenkov, Y., Aladinskiy, V., Sanchez-Lengeling, B., Aspuru-Guzik, A. & Zhavoronkov, A. Reinforced Adversarial Neural Computer for de Novo Molecular Design. *J. Chem. Inf. Model.* **58**, 1194–1204 (2018).
70. Ertl, P., Lewis, R., Martin, E. & Polyakov, V. In silico generation of novel, drug-like chemical matter using the LSTM neural network. *ArXiv* (2017).
71. Chen, J. H. & Baldi, P. No electron left behind: A rule-based expert system to predict chemical reactions and reaction mechanisms. *J. Chem. Inf. Model.* **49**, 2034–2043 (2009).
72. Blurock, E. S. Reaction: System for Modeling Chemical Reactions. *J. Chem. Inf. Comput. Sci.* **35**, 607–616 (1995).
73. Baylon, J. L., Cilfone, N. A., Gulcher, J. R. & Chittenden, T. W. Enhancing Retrosynthetic Reaction Prediction with Deep Learning Using Multiscale Reaction Classification. *J. Chem. Inf. Model.* **59**, 673–688 (2019).
74. Segler, M. H. S., Preuss, M. & Waller, M. P. Planning chemical syntheses with deep neural networks and symbolic AI. *Nature* **555**, 604–610 (2018).
75. Zhou, Z., Kearnes, S., Li, L., Zare, R. N. & Riley, P. Optimization of Molecules via Deep Reinforcement Learning. *Sci. Rep.* **9**, 10752 (2019).
76. Arús-Pous, J., Blaschke, T., Ulander, S., Reymond, J. L., Chen, H. & Engkvist, O. Exploring the GDB-13 chemical space using deep generative models. *J. Cheminform.* **11**, 1–14 (2019).
77. Oprea, T. I. & Gottfries, J. Chemography: The art of navigating in chemical space. *J. Comb. Chem.* **3**, 157–166 (2001).
78. Jolliffe, I. T. & Cadima, J. Principal component analysis: a review and recent developments. *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.* **374**, (2016).
79. Pearson, K. LIII. On lines and planes of closest fit to systems of points in space. *London, Edinburgh, Dublin Philos. Mag. J. Sci.* **2**, 559–572 (1901).
80. Hotelling, H. Analysis of a complex of statistical variables into principal components. *J. Educ. Psychol.* **24**, 417–441 (1933).

81. Cohen, J. Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences. *Appl. Mult. Regression/Correlation Anal. Behav. Sci.* **1**, (2013).
82. Van Der Maaten, L. & Hinton, G. Visualizing Data using t-SNE. *J. Mach. Learn. Res.* **9**, 2579–2605 (2008).
83. Kohonen, T. Self-organized formation of topologically correct feature maps. *Biol. Cybern.* **43**, 59–69 (1982).
84. Lin, A., Horvath, D., Marcou, G., Beck, B. & Varnek, A. Multi-task generative topographic mapping in virtual screening. *J. Comput. Aided. Mol. Des.* **33**, 331–343 (2019).
85. Kireeva, N., Baskin, I. I., Gaspar, H. A., Horvath, D., Marcou, G. & Varnek, A. Generative Topographic Mapping (GTM): Universal Tool for Data Visualization, Structure-Activity Modeling and Dataset Comparison. **31**, 301–312 (2012).
86. Gaspar, H. A., Baskin, I. I., Marcou, G., Horvath, D. & Varnek, A. GTM-Based QSAR Models and Their Applicability Domains. *Mol. Inform.* **34**, 348–356 (2015).
87. Sattarov, B., Baskin, I. I., Horvath, D., Marcou, G., Bjerrum, E. J. & Varnek, A. De Novo Molecular Design by Combining Deep Autoencoder Recurrent Neural Networks with Generative Topographic Mapping. *J. Chem. Inf. Model.* **59**, 1182–1196 (2019).
88. Tealab, A. Time series forecasting using artificial neural networks methodologies: A systematic review. *Futur. Comput. Informatics J.* **3**, 334–340 (2018).
89. Sutskever, I., Vinyals, O. & Le, Q. V. Sequence to sequence learning with neural networks. *Adv. Neural Inf. Process. Syst.* **4**, 3104–3112 (2014).
90. Nallapati, R., Zhou, B., dos Santos, C., Gulçehre, Ç. & Xiang, B. Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond. *20th SIGNLL Conf. Comput. Nat. Lang. Learn. Proc.* 280–290 (2016).
91. Zhang, Y., Li, D., Wang, Y., Fang, Y. & Xiao, W. Abstract Text Summarization with a Convolutional Seq2seq Model. *Appl. Sci.* **9**, 1665 (2019).
92. Ghandi, T., Pourreza, H. & Mahyar, H. Deep Learning Approaches on Image Captioning: A Review. *ArXiv* (2022).
93. Mnasri, M. Recent advances in conversational NLP : Towards the standardization of Chatbot building. *ArXiv* (2019).
94. Patidar, M., Agarwal, P., Vig, L. & Shroff, G. Automatic conversational helpdesk solution using Seq2Seq and slot-filling models. *Int. Conf. Inf. Knowl. Manag. Proc.* 1967–1976 (2018).
95. Shih, Y. J., Wu, S. L., Zalkow, F., Muller, M. & Yang, Y. H. Theme Transformer: Symbolic Music Generation with Theme-Conditioned Transformer. *IEEE Trans. Multimed.* (2022).
96. Pascanu, R., Mikolov, T. & Bengio, Y. On the difficulty of training Recurrent Neural Networks. in *30th International Conference on Machine Learning* 2347–2355 (International Machine Learning Society (IMLS), 2012).

97. Hochreiter, S. & Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **9**, 1735–1780 (1997).
98. Yang, S., Yu, X. & Zhou, Y. LSTM and GRU Neural Network Performance Comparison Study: Taking Yelp Review Dataset as an Example. in *2020 International Workshop on Electronic Communication and Artificial Intelligence* 98–101 (Institute of Electrical and Electronics Engineers Inc., 2020).
99. Makhzani, A. & Frey, B. k-Sparse Autoencoders. in *2nd International Conference on Learning Representations* (International Conference on Learning Representations, ICLR, 2013).
100. Vincent, P., Larochelle, H., Bengio, Y. & Manzagol, P. A. Extracting and composing robust features with denoising autoencoders. *Proc. 25th Int. Conf. Mach. Learn.* 1096–1103 (2008).
101. Rifai, S., Mesnil, G., Vincent, P., Muller, X., Bengio, Y., Dauphin, Y. & Glorot, X. *Higher order contractive auto-encoder. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* vol. 6912 LNAI (2011).
102. Cho, K., Van Merriënboer, B., Bahdanau, D. & Bengio, Y. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. *ArXiv* (2004).
103. Bahdanau, D., Cho, K. & Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. *ArXiv* (2014).
104. Luong, M. T., Pham, H. & Manning, C. D. Effective Approaches to Attention-based Neural Machine Translation. in *Conference on Empirical Methods in Natural Language Processing* 1412–1421 (Association for Computational Linguistics (ACL), 2015).
105. Sherstinsky, A. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network. *Phys. D Nonlinear Phenom.* **404**, (2020).
106. Schuster, M. & Paliwal, K. K. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.* **45**, 2673–2681 (1997).
107. Xu, K., Ba, J. L., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R. S. & Bengio, Y. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. in *32nd International Conference on Machine Learning* vol. 3 2048–2057 (International Machine Learning Society (IMLS), 2015).
108. Lin, T., Wang, Y., Liu, X. & Qiu, X. A Survey of Transformers; A Survey of Transformers. *ArXiv* (2021).
109. Akhmetshin, T., Lin, A., Mazitov, D., Zabolotna, Y., Ziaikin, E., Madzhidov, T. & Varnek, A. HyFactor: A Novel Open-Source, Graph-Based Architecture for Chemical Structure Generation. *J. Chem. Inf. Model.* **62**, 3524–3534 (2022).
110. Abate, C., Decherchi, S. & Cavalli, A. Graph neural networks for conditional de novo drug design. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* (2023).
111. O’Boyle, N. M. Towards a Universal SMILES representation - A standard method to generate canonical SMILES based on the InChI. *J. Cheminform.* **4**, 1–14 (2012).

112. Weininger, D., Weininger, A. & Weininger, J. L. SMILES. 2. Algorithm for Generation of Unique SMILES Notation. *J. Chem. Inf. Comput. Sci.* **29**, 97–101 (1989).
113. Heller, S. R., McNaught, A., Pletnev, I., Stein, S. & Tchekhovskoi, D. InChI, the IUPAC International Chemical Identifier. *J. Cheminform.* **7**, 1–34 (2015).
114. Gómez-Bombarelli, R., Wei, J. N., Duvenaud, D., Hernández-Lobato, J. M., Sánchez-Lengeling, B., Sheberla, D., Aguilera-Iparraguirre, J., Hirzel, T. D., Adams, R. P. & Aspuru-Guzik, A. Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. *ACS Cent. Sci.* **4**, 268–276 (2018).
115. Winter, R., Montanari, F., Noé, F. & Clevert, D. A. Learning continuous and data-driven molecular descriptors by translating equivalent chemical representations. *Chem. Sci.* **10**, 1692–1701 (2019).
116. O’boyle, N. M. & Dalke, A. DeepSMILES: An Adaptation of SMILES for Use in Machine-Learning of Chemical Structures. *ChemRxiv* (2018).
117. Krenn, M., Häse, F., Nigam, A. K., Friederich, P. & Aspuru-Guzik, A. Self-referencing embedded strings (SELFIES): A 100% robust molecular string representation. *Mach. Learn. Sci. Technol.* **1**, (2020).
118. Literature Statistics. <https://www.tylervigen.com/literature-statistics>.
119. Goyal, A., Lamb, A., Zhang, Y., Zhang, S., Courville, A. & Bengio, Y. Professor Forcing: A New Algorithm for Training Recurrent Networks. *Adv. Neural Inf. Process. Syst.* 4608–4616 (2016).
120. Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H. & He, Q. A Comprehensive Survey on Transfer Learning. *Proc. IEEE* **109**, 43–76 (2019).
121. Moret, M., Friedrich, L., Grisoni, F., Merk, D. & Schneider, G. Generative molecular design in low data regimes. *Nat. Mach. Intell.* **2**, 171–180 (2020).
122. Merk, D., Friedrich, L., Grisoni, F. & Schneider, G. De Novo Design of Bioactive Small Molecules by Artificial Intelligence. *Mol. Inform.* **37**, (2018).
123. Merk, D., Grisoni, F., Friedrich, L. & Schneider, G. Tuning artificial intelligence on the de novo design of natural-product-inspired retinoid X receptor modulators. *Commun. Chem.* **1**, 1–9 (2018).
124. Blaschke, T., Arús-Pous, J., Chen, H., Margreitter, C., Tyrchan, C., Engkvist, O., Papadopoulos, K. & Patronov, A. REINVENT 2.0: An AI Tool for De Novo Drug Design. *J. Chem. Inf. Model.* **60**, 5918–5922 (2020).
125. Arulkumaran, K., Deisenroth, M. P., Brundage, M. & Bharath, A. A. A Brief Survey of Deep Reinforcement Learning. *ArXiv* (2017).
126. Blaschke, T., Engkvist, O., Bajorath, J. & Chen, H. Memory-assisted reinforcement learning for diverse molecular de novo design. *J. Cheminform.* **12**, 1–17 (2020).
127. Kotsias, P.-C. C., Arús-Pous, J., Chen, H., Engkvist, O., Tyrchan, C. & Bjerrum, E. J. Direct steering of de novo molecular generation with descriptor conditional recurrent neural networks. *Nat. Mach. Intell.* **2**, 254–265 (2020).

128. Winter, R., Montanari, F., Steffen, A., Briem, H., Noé, F. & Clevert, D. A. Efficient multi-objective molecular optimization in a continuous latent space. *Chem. Sci.* **10**, 8016–8024 (2019).
129. Griffiths, R.-R. R. & Hernández-Lobato, J. Constrained Bayesian optimization for automatic chemical design using variational autoencoders. *Chem. Sci.* **11**, 577–586 (2020).
130. Kang, S. & Cho, K. Conditional Molecular Design with Deep Generative Models. *J. Chem. Inf. Model.* **59**, 43–52 (2019).
131. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. & Bengio, Y. Generative Adversarial Networks. *Commun. ACM* **63**, 139–144 (2014).
132. Putin, E., Asadulaev, A., Vanhaelen, Q., Ivanenkov, Y., Aladinskaya, A. V., Aliper, A. & Zhavoronkov, A. Adversarial Threshold Neural Computer for Molecular de Novo Design. *Mol. Pharm.* **15**, 4386–4397 (2018).
133. Sanchez-Lengeling, B., Outeiral, C., Guimaraes, G. L. & Aspuru-Guzik, A. Optimizing distributions over molecular space. An Objective-Reinforced Generative Adversarial Network for Inverse-design Chemistry (ORGANIC). *ChemRxiv* (2017).
134. Guimaraes, G., Sanchez-Lengeling, B., Outeiral, C., Luis, P., Farias, C. & Aspuru-Guzik, A. Objective-Reinforced Generative Adversarial Networks (ORGAN) for Sequence Generation Models. *ChemRxiv* (2017).
135. Nam, J. & Kim, J. Linking the Neural Machine Translation and the Prediction of Organic Chemistry Reactions. *ArXiv* (2016).
136. Schwaller, P., Gaudin, T., Lányi, D., Bekas, C. & Laino, T. “Found in Translation”: predicting outcomes of complex organic chemistry reactions using neural sequence-to-sequence models. *Chem. Sci.* **9**, 6091–6098 (2018).
137. Liu, B., Ramsundar, B., Kawthekar, P., Shi, J., Gomes, J., Luu Nguyen, Q., Ho, S., Sloane, J., Wender, P. & Pande, V. Retrosynthetic Reaction Prediction Using Neural Sequence-to-Sequence Models. *ACS Cent. Sci.* **3**, 1103–1113 (2017).
138. Karpov, P., Godin, G. & Tetko, I. V. A Transformer Model for Retrosynthesis. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)* 817–830 (2019).
139. Schwaller, P., Petraglia, R., Zullo, V., Nair, V. H., Haeuselmann, R. A., Pisoni, R., Bekas, C., Iuliano, A. & Laino, T. Predicting Retrosynthetic Pathways Using a Combined Linguistic Model and Hyper-Graph Exploration Strategy. *Chem. Sci.* **11**, 3316–3325 (2020).
140. Kohonen, T., Kaski, S., Somervuo, P., Lagus, K., Oja, M. & Paatero, V. Self-organizing map. in *Proceedings of the IEEE* 1464–1480 (1990).
141. Gaspar, H. A., Sidorov, P., Horvath, D., Baskin, I. I., Marcou, G. & Varnek, A. Generative topographic mapping approach to chemical space analysis. *ACS Symp. Ser.* **1222**, 211–241 (2016).
142. Gaspar, H. A., Baskin, I. I. & Varnek, A. Visualization of a multidimensional descriptor

- space. *ACS Symp. Ser.* **1222**, 243–267 (2016).
143. Gaspar, H. A., Baskin, I. I., Marcou, G., Horvath, D. & Varnek, A. Chemical data visualization and analysis with incremental generative topographic mapping: big data challenge. *J. Chem. Inf. Model.* **55**, 84–94 (2015).
 144. Bengio, Y. Learning Deep Architectures for AI. *Found. Trends Mach. Learn.* **2**, 1–55 (2009).
 145. Bengio, Y., Courville, A. & Vincent, P. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**, 1798–1828 (2013).
 146. Karpov, P. V., Osolodkin, D. I., Baskin, I. I., Palyulin, V. A. & Zefirov, N. S. One-class classification as a novel method of ligand-based virtual screening: the case of glycogen synthase kinase 3 β inhibitors. *Bioorg. Med. Chem. Lett.* **21**, 6728–6731 (2011).
 147. Xu, Z., Zhu, F., Wang, S. & Huang, J. Seq2seq fingerprint: An unsupervised deep molecular embedding for drug discovery. in *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics* 285–294 (Association for Computing Machinery, Inc, 2017).
 148. Rogers, D. & Hahn, M. Extended-connectivity fingerprints. *J. Chem. Inf. Model.* **50**, 742–754 (2010).
 149. Zhavoronkov, A. *et al.* Deep learning enables rapid identification of potent DDR1 kinase inhibitors. *Nat. Biotechnol.* **37**, 1038–1040 (2019).
 150. Jin, W., Barzilay, R. & Jaakkola, T. Junction Tree Variational Autoencoder for Molecular Graph Generation. in *35th International Conference on Machine Learning* vol. 5 3632–3648 (2018).
 151. Liu, Q., Allamanis, M., Brockschmidt, M. & Gaunt, A. L. Constrained Graph Variational Autoencoders for Molecule Design. *Adv. Neural Inf. Process. Syst.* 7795–7804 (2018).
 152. Simonovsky, M. & Komodakis, N. GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)* 412–422 (2018).
 153. Samanta, B., De, A., Jana, G., Chattaraj, P. K., Ganguly, N. & Rodriguez, M. G. NeVAE: A Deep Generative Model for Molecular Graphs. *Proc. AAAI Conf. Artif. Intell.* **33**, 1110–1117 (2019).
 154. Jin, W., Barzilay, R. & Jaakkola, T. S. Multi-Resolution Autoregressive Graph-to-Graph Translation for Molecules. *ChemRxiv* (2019).
 155. Kusner, M. J., Paige, B. & Hernández-Lobato, J. M. Grammar Variational Autoencoder. in *34th International Conference on Machine Learning* 1945–1954 (PMLR, 2017).
 156. RDKit. <http://www.rdkit.org/>.
 157. Lim, H., Kim, T. H. & Kang, S. Prediction-based error correction for gpu reliability with low overhead. *Electron.* **9**, 1–18 (2020).
 158. Chemaxon. <https://chemaxon.com/>.
 159. Hopfield, J. J. Neural networks and physical systems with emergent collective

- computational abilities. *Proc. Natl. Acad. Sci. U. S. A.* **79**, 2554–2558 (1982).
160. Montúfar, G. Restricted Boltzmann Machines: Introduction and Review. *Springer Proc. Math. Stat.* **252**, 75–115 (2018).
 161. Salakhutdinov, R. & Hinton, G. Deep Boltzmann machines. in *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics* vol. 5 448–455 (PMLR, 2009).
 162. Srivastava, N. & Salakhutdinov, R. Multimodal learning with Deep Boltzmann Machines. *J. Mach. Learn. Res.* **15**, 2949–2980 (2014).
 163. Wang, T., Dai, X. & Liu, Y. Learning with Hilbert–Schmidt independence criterion: A review and new perspectives. *Knowledge-Based Syst.* **234**, 107567 (2021).
 164. Zabolotna, Y., Bonachera, F., Horvath, D., Lin, A., Marcou, G., Klimchuk, O. & Varnek, A. Chemspace Atlas: Multiscale Chemography of Ultralarge Libraries for Drug Discovery. *J. Chem. Inf. Model.* **62**, 4537–4548 (2022).

Résumé

Cette thèse est dédiée à l'exploration et à la compréhension des espaces latents des réseaux de neurones, dans le but de créer un lien entre ces derniers et des descripteurs structuraux classiques afin de réaliser du QSAR inverse. Le potentiel génératif des architectures seq2seq est souvent accompagné d'une compréhension partielle des règles qui définissent leurs espaces latents. Une étude de la construction l'espace chimique d'un Autoencodeur a montré son habilité à recréer les propriétés et caractéristiques structurales de molécules existantes avec différents niveaux de réussite selon la complexité des structures et leur densité dans l'espace. Le modèle a même été modifié pour générer des nouvelles réactions atteignables chimiquement.

Cependant, l'interprétation séquentielle des structures chimiques à travers les chaînes SMILES ont tendance à créer des faiblesses dans les espaces chimiques résultants. De ce fait, les descripteurs ISIDA, qui sont plus robustes, sont généralement préférés lors de la cartographie et l'identification de zones d'intérêt lors de la recherche d'actifs. Plusieurs méthodes pour combiner l'efficacité des vecteurs ISIDA avec le pouvoir génératif d'un espace latent d'Autoencodeur ont abouti au développement d'une nouvelle architecture basée sur les Autoencodeurs Variationnels Conditionnels et le mécanisme d'Attention qui a permis la génération ciblée de nouvelles molécules potentiellement actives contre une cible biologique.

Résumé en anglais

This thesis is dedicated to the exploration and understanding of neural network latent spaces, to allow the creation of a link between the latter and classical structural descriptors to perform inverse QSAR. The generative potential of seq2seq architectures often comes with a blurry understanding of the rules governing its chemical spaces. A study of an Autoencoder's chemical space construction showed its ability to recreate existing property distributions and molecular structures with varying degrees of success depending on complexity and density factors. The model was even successfully modified to generate feasible and novel reactions.

However, the sequential interpretation of chemical structures through SMILES strings tend to create weaknesses in the resulting chemical spaces. As such, structural descriptors like ISIDA, which are more robust, are usually preferred to map and identify zones of interest when searching for active compounds. Several methods to harness the efficiency of ISIDA descriptors and combine it with the generative power of an Autoencoder latent space resulted in the development of a new architecture based on Conditional Variational Autoencoders and the Attention Mechanism to generate potent molecules against biological targets.