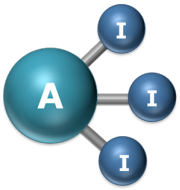

Artificial Intelligence and Augmented Intelligence for Automated Investigations for Scientific Discovery

Optimising Flatland: Inverse design of desalination membranes
Final Report
Project Dates: 01/05/2020 - 15/12/2020
The University of Sheffield

Adam N. Hill and Dr J. Grant Hill
University of Sheffield

Report Date: 03/08/2021

AI3SD-Project-Series:Report5_Hill_Final

Optimising Flatland: Inverse design of desalination membranes
AI3SD-Project-Series:Report5_Hill_Final
Report Date: 03/08/2021
DOI: 10.5258/SOTON/P0040

Network: Artificial Intelligence and Augmented Intelligence for Automated Investigations for Scientific Discovery
This Network+ is EPSRC Funded under Grant No: EP/S000356/1

Principal Investigator: *Professor Jeremy Frey*
Co-Investigator: *Professor Mahesan Niranjan*
Network+ Coordinator: *Dr Samantha Kanza*

Contents

1 Project Details

Title	Optimising Flatland: Inverse design of desalination membranes
Funding reference	AI3SD-FundingCall2_008
Lead Institution	University of Sheffield
Project Dates	01/05/2020 - 15/12/2020
Website	N/A
Keywords	Machine learning; Generative models; Inverse design

2 Project Team

2.1 Principal Investigator

Name and Title	Dr J. Grant Hill
Employer name / University Department Name	University of Sheffield / Department of Chemistry
Work Email	grant.hill@sheffield.ac.uk
Website Link (if available)	http://www.grant-hill.group.shef.ac.uk

2.2 Co-Investigators

Name and Title	Prof. Patrick W. Fowler
Employer name / University Department Name	University of Sheffield / Department of Chemistry
Work Email	p.w.fowler@sheffield.ac.uk
Website Link (if available)	https://www.sheffield.ac.uk/chemistry/people/academic/patrick-w-fowler

Name and Title	Dr Jonathan A. Foster
Employer name / University Department Name	University of Sheffield / Department of Chemistry
Work Email	jona.foster@sheffield.ac.uk
Website Link (if available)	https://foster.group.shef.ac.uk

Name and Title	Dr Peyman Z. Moghadam
Employer name / University Department Name	University of Sheffield / Department of Chemical and Biological Engineering
Work Email	p.moghadam@sheffield.ac.uk
Website Link (if available)	https://www.sheffield.ac.uk/cbe/people/academic-staff/peyman-z-moghadam

Name and Title	Dr Kim Jelfs
Employer name / University Department Name	Imperial College London / Department of Chemistry
Work Email	k.jelfs@imperial.ac.uk
Website Link (if available)	http://www.jelfs-group.org

2.3 Researchers & Collaborators

Adam N. Hill was seconded full-time from his PhD studies to work on this project. Although the project was unrelated to his PhD research, Adam has experience of writing scientific code and the application of machine learning in chemistry.

3 Publicity Summary

Access to clean drinking water for all is part of the UN sustainable development goals and the desalination of sea water is a likely candidate for achieving this. The aim of this project was to establish an integrated discovery pipeline for developing new materials for water desalination. This was partially achieved by creating a software pipeline that generates a large dataset of ultrathin membrane materials and applying artificial intelligence (AI) design paradigms to begin to predict the ideal materials for this application. The consumables required for synthesising and testing the membranes, once their prediction has been refined, have also been obtained and will allow for a closed-loop approach to materials design.

4 Executive Summary

AI has the potential to act as a smart navigation system in the quest for new functional materials that hold the key to solving societal problems, but obtaining large data sets encapsulating chemical information in a way computer algorithms can understand is a significant challenge. This project has explored the use of AI generative design to predict new 2D materials for desalination, building towards an eventual goal of integrated discovery where experimental results are used to refine AI predictions. By creating a pipeline that combines existing open software packages and tools with our own code, we are able to quickly and efficiently construct a large database of hypothetical desalination membranes, including a unique string-based representation for each structure that can be used in machine learning / AI algorithms. We also develop an AI generative model for the prediction of new membrane materials, which learns from this data set and has already picked up many of the important features for a valid membrane structure. We further outline how the database can be easily extended in a semi-automated fashion, how the AI predictions may be improved in future work, and how these predictions will be synthesised and tested. Once the generative model predicts materials that prove to be effective for

desalination, this would make a strong case for further investment from bodies such as EPSRC towards an automated, integrated discovery pipeline for 2D materials.

5 Aims and Objectives

Over 1.2 billion people around the world currently lack access to safe drinking water, a number the World Water Council estimate will rise to 3.9 billion in coming decades as a result of continued industrialisation and climate change. Three quarters of the Earth's surface is covered in water, but current desalination technology is prohibitively expensive and energy intensive. Two-dimensional materials, in particular metal-organic nanosheets (MONs), have shown great potential for creating a new generation of ultrathin membranes with exceptionally high flux rates and selectivity for desalination. The overall goal of the research project was to discover metal-organic nanosheets for desalination using a combination of AI methods and experimental validation. This was sub-divided into the following aims:

- Use generative AI methods for the inverse design of MONs with ideal properties for desalination and exfoliation.
  STATUS: The groundwork has been laid for the generative design of new MONs. With the pipeline and AI generative model developed in this work, future research should be able to build on and utilise the programs and scripts provided to generate new materials.

- Produce a sustainable software pipeline/framework that can be adapted for the inverse design of other 2D materials.
  STATUS: We have developed a full pipeline for data generation: from the initial selection of building blocks to the generation of a database of 2D materials, and through to the training of a sequence to sequence generative algorithm.

- Establish an effective collaboration across the interdisciplinary research team.
  STATUS: Despite the Covid-19 pandemic the research team have worked well together and are in a position to continue collaborating on integrated discovery of 2D materials.

- Obtain preliminary experimental results that will act as a springboard for larger research proposals targeting closed-loop 2D materials discovery.
  STATUS: The experimental team have purchased the required consumables and are ready to synthesise and test new AI generated MONs and evaluate them in separation applications in order to validate the AI membrane design.

6 Methodology

6.1 Scientific Methodology

To train AI models a large dataset is typically required to achieve reasonable results. However, for 2D materials there is insufficient experimental data for training a generative model – by their very nature 2D nanomaterials are poorly crystalline and hence difficult to characterise. In this work we have adopted a "synthetic data" approach, in which a computer algorithm is used to construct the dataset.

To generate these new MONs we build them up from their composite "building blocks": the topological net, giving the overall shape and symmetry of the crystal; the nodes, commonly organometallic structures; and the edges, organic links between each node. Figure 1 shows two options for the node and edge building blocks. In one case we can choose to break the structure down into building blocks based upon chemical intuition and synthesis, which would result in four points of connection in coordination. Or we can break the structure along the carbon-carbon bond resulting in only two points of connection and simplifying the generative

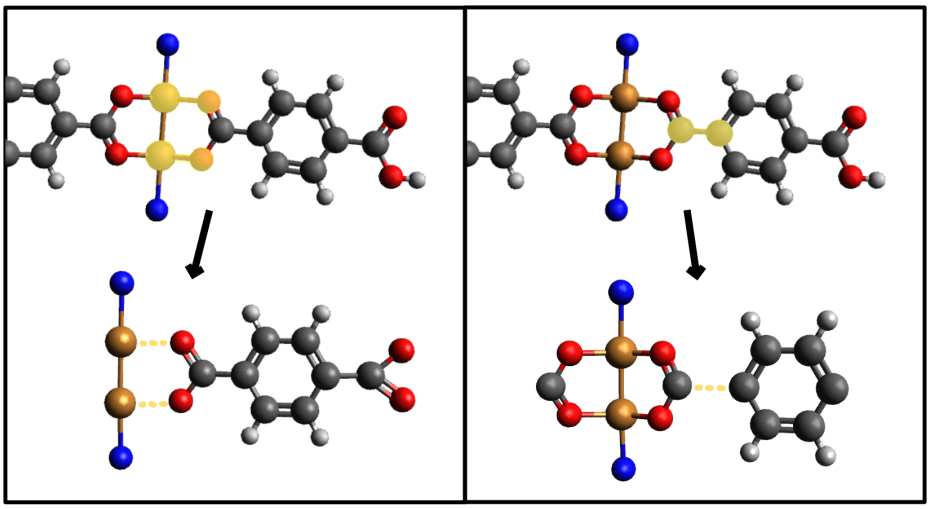

Figure 1: Two options for breaking the overall structure down into building blocks. The left shows the structure being broken along the two oxygen-metal bonds (highlighted in yellow) leading to four points of coordination, which matches how the structure would be synthesised experimentally. The right shows the structure being broken along the carbon-carbon bond (highlighted in yellow) leading to two points of coordination. Using the option on the right for constructing building blocks simplifies the computational problem.

process. This is a common method of describing 3D metal-organic frameworks (MOFs), and these extended node building blocks are typically referred to as secondary building units (SBUs).

After defining a large number of these relevant building blocks, the construction of the nanosheets from these blocks is automated using the Topologically Based Crystal Constructor (*ToBaCCo*) code,[1,2] which was originally designed for MOFs. The resulting large set of hypothetical nanosheets can then be evaluated in terms of relevant properties such as pore size, before the data set is used to train generative AI methods.

6.2 AI Methodology

Two of the most popular generative models are: the generative adversarial network (GAN) which learns through the competitive training of two networks, a generator and a discriminator; and the variational autoencoder (VAE) which generates a probability space to describe the dataset. This project uses a variational autoencoder due to its recent successful application to nanoporous crystalline reticular materials.[3]

Variational autoencoders are composed of an encoder and a decoder. The encoder takes an input, x, and generates two values in a latent space, μ_x and σ_x. Points, z, are randomly sampled from a latent normal distribution that is expected to generate the data and a decoder network regenerates the input data x. To train a VAE on character string sequences we have implemented a long short-term memory recurrent neural network (LSTM) as part of the encoding process.[4] This allows us to maintain information about the string structure in the encoding process.

We have generated a unique string-based identifier, known as a MOFid,[5] for each entry in our dataset, and these will be used to train the LSTM-VAE to discover new MONs. The network has been implemented in Python, using sustainable software practices and existing machine learning / AI frameworks.

7 Results

7.1 Pipeline for a database of hypothetical MONs

The main output of this work is a software pipeline designed for the fast production of a database of hypothetical MONs. The code and examples can be found in our git repository. The pipeline can be split up into sections:

- SBU production (nodes).
- Organic linker production (edges).
- Topologically driven MON production (through *ToBaCCo*).
- Unique ID production (MOFids).
- Structure optimisation and property calculation.

Figure 2 shows the overall structure and data flow throughout the pipeline. This section outlines the structure and logic behind this pipeline, while subsequent sections will show an example of this pipeline's use and the training of an LSTM-VAE with the generated data.

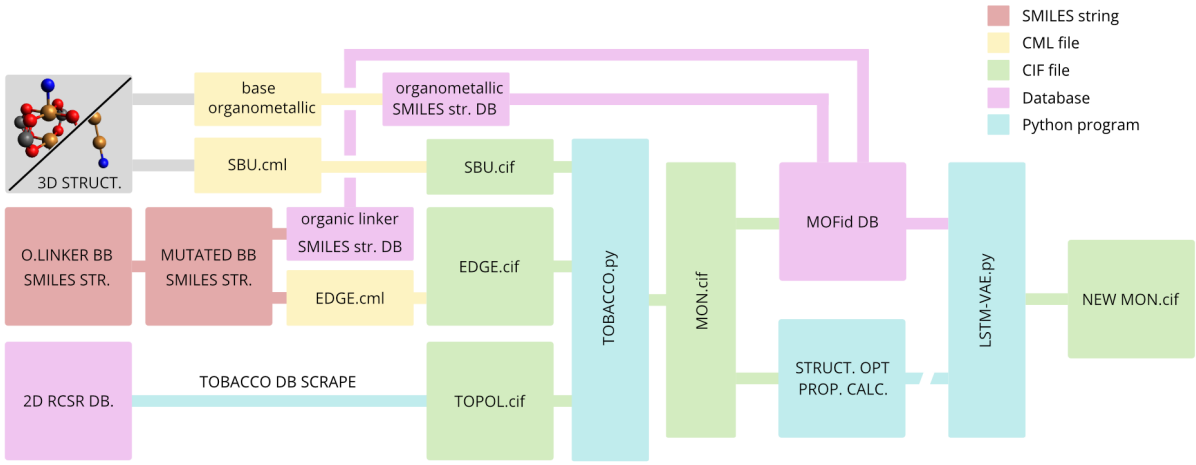

Figure 2: An overview of the full data pipeline, showing data flow and pipeline structure. Each file type is colour coded to assist with visualising each step in the process and how each part leads into the next. Red are SMILES strings [6] for backbones provided by the user and mutated molecules generated through a Python script. Yellow are cml files produced by a 3D structure program in the case of the SBUs and by a Python script utilising *Open Babel* for the edges. Green are *ToBaCCo* specific cif files produced by a Python script in all cases. Purple are databases of data linking the cif files to SMILES strings and MOFids. Blue are Python programs.

SBU production

Due to the complex nature of SBU geometry, specifying their 3D structure manually is preferred over trying to provide a SMILES string for the structure. Using a 3D structure package, a collection of SBUs is generated and their geometry optimised using the universal force field (UFF).[7] After geometry optimisation a dummy/indicator atom (xenon in this case) is added to the structure to highlight where the connections to linkers should be made. Then the structure is saved as a chemical markup language (cml) file with geometry information included.

It is important to also produce a list of file names and their SMILES strings for later use in constructing the MOFid. However, MOFids require the node to be represented as an organometallic building block rather than an SBU, which contains part of the organic linker. The difference between the building block and SBU is shown in Figure 3. Thus a version of the

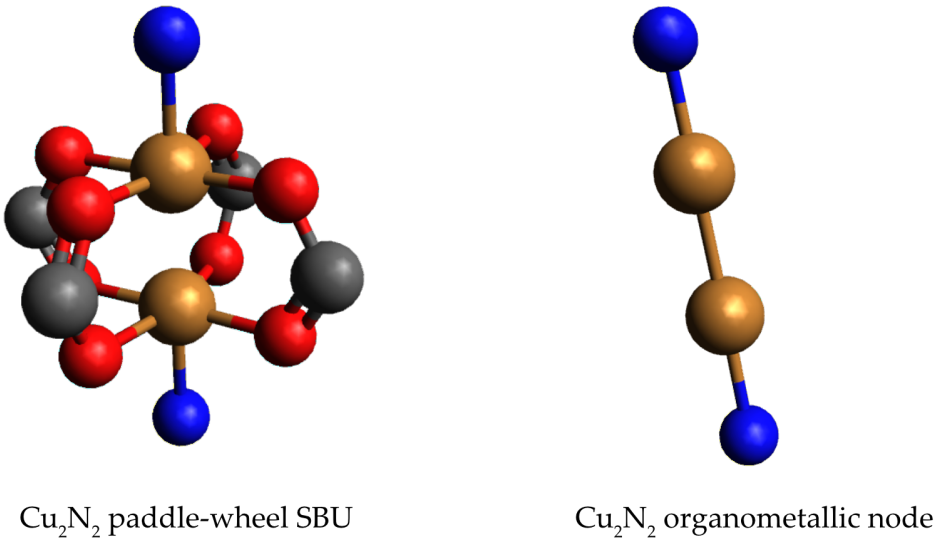

Figure 3: Comparison of the 3D structure of an SBU and the organometallic building block it is constructed from. The use of MOFids as a representation of MONs in a VAE requires the building block / node form.

`node.cml` representing the base organometallic building block is also provided for the pipeline and the `node_extract_smiles.py` script is run to extract the SMILES string associated with the file.

Finally, `node_cml_to_cif.py` is run on the collection of SBUs to produce *ToBaCCo* compatible crystallographic information file (cif) files for the nodes. These files specifically include bonding information that is necessary for *ToBaCCo* to run.

Organic linker production

Although organic linkers can seem incredibly varied due to their large number of functional groups, they often share similar or identical carbon backbones, thus a mutative method of generation is used for large scale production of organic linkers. The process is as follows:

- Common backbones are identified and listed as SMILES strings.
- The desired mutating functional groups are also listed as SMILES strings and are passed to the mutator script.
- If every permutation of mutation is required for each backbone then `permutative_linker_mutation.py` is run (**Caution:** this can generate extremely large numbers of molecules if backbones with many mutation sites are used).
- If only select sites require mutation then these positions are highlighted with an indicator atom (in this case: rubidium) in the SMILES string and `selective_linker_mutation.py` is run instead.

Once a list of mutated SMILES strings has been developed, `smiles_to_cml.py` is run to produce 3D structures for the SMILES strings using *Open Babel*. This step also outputs a database of linker file names and the relevant SMILES string for later use in generating MOFids. The SMILES string here has the placeholder indicator atom replaced for the full connector (in this case it is a carbonyl group, but this could in theory be any connecting group). Finally `edge_cml_to_cif.py` is run to produce *ToBaCCo* compatible cif files for the linkers.

Topologically driven MON production

The construction of the MON from the individual building blocks is achieved through the use of *ToBaCCo*. The desired topologies are extracted from the database provided with the program and placed in the working directory. *ToBaCCo* is designed to take all input edges and generate every permutation of the edges with the provided nodes. As the number of permutations can become very large with even a small number of members, limitations are applied on how many linkers are available to the program for each run. The `randomiser.py` selects a random assortment of edges from the main database and moves them into the working directory for *ToBaCCo* to use. Using the cif files produced for all nodes and topologies, and the edges selected in the randomisation, `tobacco.py` is run to completion. This randomisation process can then be repeated and *ToBaCCo* run again until a sufficiently large and varied database has been constructed.

Unique ID production

A MOFid consists of three main parts: the metal node SMILES string; the organic linker SMILES strings; and the topology information.[5] *ToBaCCo* outputs the cif files with unique names based upon the SBU and linkers used to produce it. This file name is compared with the two node and edge SMILES databases produced earlier and a MOFid is constructed manually using this information. Each cif file is then linked to a valid MOFid.

The MOFid program can automatically construct this string from a cif file. However, this automatic process was found to be unreliable for unoptimised structures. Since optimisation of a MON can be costly, and ultimately not all MONs will be optimised in this pipeline, these unique IDs are produced before any optimisation has occurred and hence many fail to automatically generate valid MOFids.

Structure optimisation and property calculation

It has been shown that it is possible to train a property predictor from latent space using a smaller subset of labelled training data,[3] and since structure optimisation and property prediction are costly compared to constructing the MON, not all structures will undergo this process.

A random subset of the main database is selected and run through *Open Babel's*[8,9] `obminimize` to optimise the structure using using molecular mechanics with UFF.[7] The resulting structure can then be run through Zeo++[10] to calculate pore size, which is then added to the main database.

7.2 An example use of the pipeline

We have constructed a large database of hypothetical, computationally predicted 2D nanosheet structures based on a set of predetermined building blocks: topographies, secondary building units (SBUs), and organic linkers. The process for producing the nanosheets is outlined below, and the code is provided in a git repository.

- A total of 144 2D topography files were extracted from the full topography database contained within *ToBaCCo*. These topologies had been extracted from the Reticular Chemistry Structure Resource (RCSR).[11] These 144 topographies were chosen such that the number of different organic linkers in the structure is limited to a maximum of five to avoid the problem of combinatorial explosion.

- 65 SBUs suitable for the creation of 2D materials were identified from both the 131 units identified in MOF chemistry by Tranchemontagne *et al.*[12], and the SBUs provided with *ToBaCCo*. These SBUs were created in molecular modelling software and their conversion into the cif file format was automated.

- Using existing expertise within the Foster group, 10 unique and common organic linker backbones were identified that can be used in the production of 2D nanosheets.

- Using RDKit and code developed in this project, the selected organic linker backbones were then mutated with 18 different functional groups in specified positions to produce a database of 747 varied organic linkers, which were also converted into cif format.

- The first pass of the randomiser selected 10 organic linkers that were passed to *ToBaCCo* along with the SBUs and topologies. *ToBaCCo* was then run, producing 45,329 new MONs in about 3 hours.

- The resulting MONs then underwent MOFid construction to produce a machine readable string-based identifier for each structure that will represent the MON in AI algorithms.

Flexibility in terms of future use and development has been a key element in the design of this data set, and its mostly automated construction. For example, extending the current database with nanosheets containing any new SBUs discovered, or including additional organic linker backbones that provide promising experimental results would be straightforward. The current choices of topology, SBU and linker building blocks were motivated by relatively constrained mutations of experimentally verified structures, with the aim of constructing nanosheets that are feasible to synthesise. Greater variation within this mutation process, or extension to 3D MOF construction, would require only minor modifications to specific parts of the pipeline.

7.3 AI training

The LSTM-VAE used as a generative model was developed using Tensorflow 2.0[13] with Keras[14], in Python. To acquire the training dataset for the LSTM-VAE, the MOFid database generated in the data pipeline is copied and each entry cleaned to remove extraneous data such as the filename and leaving purely the node-edge-topography sequence. These sequences are encoded using a dictionary to create a sequence of indexes, then zero padded to fix the length of the input generating a tensor of shape [*batch-size*, *max-seq-length*]. The general structure of the LSTM-VAE network is shown in Figure 4.

Firstly the sequences are passed through an embedding layer which one-hot encodes the inputs to generate a 3D tensors of shape [*batch-size*, *max-seq-length*, *number-of-unique-chars*] ready for the LSTM. This layer also applies a feature mask to the zero-padding. These masked fixed length sequences are then fed into a bidirectional LSTM layer. The output tensor of an LSTM contains information about the whole sequence which helps the network learn the string structure. A bidirectional LSTM improves upon a regular LSTM by training two LSTMs on the input sequence. The first trains on the input sequence, and the second trains on the reverse of the input sequence. This can help the network learn more about the context of the input and can lead to faster and fuller learning.

The output of the LSTM is then encoded into two parameters μ_x and σ_x. A decoder LSTM is trained on sampled points z selected from the latent normal distribution: $z = \mu_x + exp(\sigma_x) + \epsilon$,

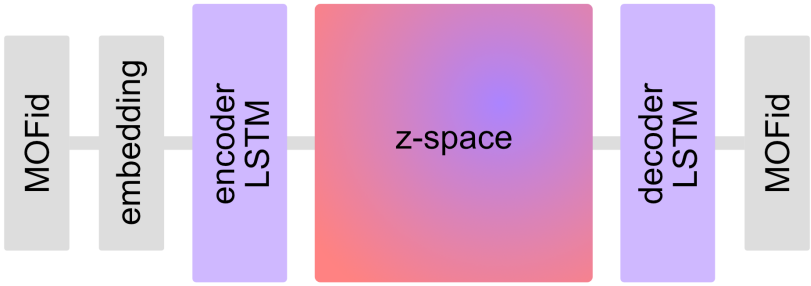

Figure 4: The general structure of the LSTM-VAE showing the flow of the data through the network. The MOFid is extracted from the database and first passed through an embedding layer to mask and one hot encode the sequence. The embedded sequence is then passed through an encoding LSTM and the outputs encoded into the z-space through the two parameters μ_x and σ_x. Outputs from the z-space are decoded using the decoder LSTM and the results converted back into a readable MOFid.

where ϵ is a random normal tensor. The parameters of this model are trained by two loss functions, a reconstruction loss that forces the decoded outputs from the second LSTM to match the initial inputs and the Kullback-Leibler (KL) divergence between latent distributions which acts as a regularisation term and helps reduce overfitting to the training data.[15] The learning rate of the network is also reduced when the loss has not improved for 10 epochs. Figure 5 shows an example output generated by the network after training for 100 epochs on a dataset of 10500 points (8500 in the training set, 2000 in the validation set).

```
Reconstructed MOFid:

[Zn](=))[]O========CCCCCCCCCCCCC((((()))))))))))))))Fcccc
(((((()))))))))ccccccccccccccccccc((((((((((((((((((((((
((((((((((((((OOOOOOOO(((cccc((((OOOO))qq

Input MOFid:

[Zn](=O)[Zn](=O)=O.C1(=CC=C(C=C1)(C(O)=O))(C(O)=O).Fc1cc
((C(O)=O))c(F)cc1(C(O)=O).Clc1cc((C(O)=O))c(Cl)cc1(C(O)=
O).COc1cc((C(O)=O))c(OC)cc1(C(O)=O).cqi
```

Figure 5: An example of the input from the validation set and the output of the trained network. The test sequence is first encoded into the latent space and then decoded from the space back into a sequence vector. If the network is successfully trained the reconstructed sequence should match the initial test sequence, which in this case it clearly does not.

It is clear from this example that the network is not yet usable for generation of new materials, however the resulting sequence is promising and shows that the model has begun to learn some of the string structure.

- It has begun to learn the concept of pairs of brackets.
- Most MOFids start with a metal contained within square brackets, from the example we can see that the sequence correctly starts with "[Zn]".
- Chains of carbons are common in the organic linkers and are present in the reconstructed MOFid.

- The model has correctly placed a carbon ring within the sequence, although the structure of the ring is incomplete.
- The topography is always found at the end of the sequence and the model has successfully identified this, yet it fails to correctly form the topography ("qq" vs. "cqi").
- It correctly zero-pads the outputs (not shown in Figure for clarity).
- Separation of the individual building blocks within the MOFid is yet to be seen.

This output suggests that the input data needs to be a much larger sample, with higher variance to allow the network to start picking up on lower frequency occurrences of sequences. The MOFids that are the inputs to this network are dominated by pairs of brackets and carbon, and with a relatively small dataset the network only learns the common features shared among all training examples.

These results are reflected in the loss values for the model (shown in Figure 6). Overall loss values are very high, and those for the validation set are roughly twice those of the training set, but they have improved over the training cycles. The relatively low training set loss compared to the validation set suggests overfitting of the model to an extent. One would expect that the model will be better at predicting the form of these sequences as the loss value is improved.

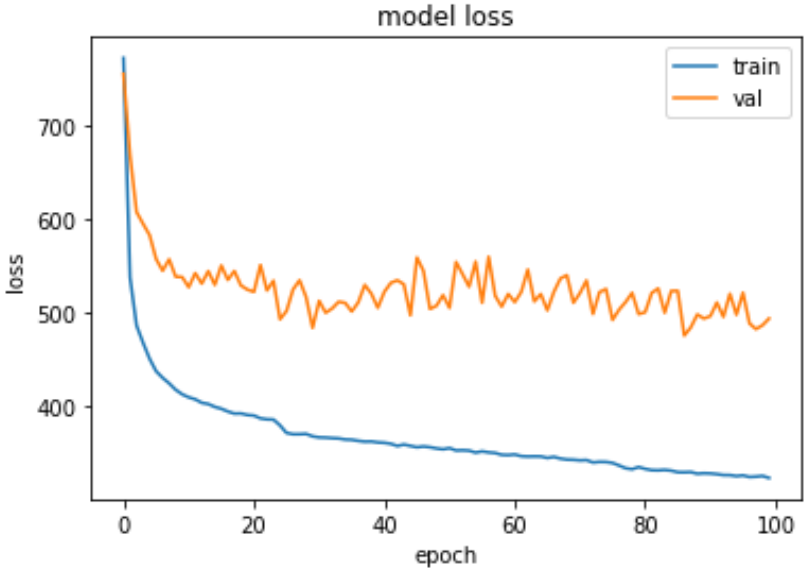

Figure 6: The loss value over time as the model trains. The loss on the training set (blue) continues to improve as the validation loss (orange) plateaus, which suggests that the model is overfitting to the training set. Overall the loss values are very high which shows that the model still needs improving before it will perform well on not only the training set but also new data.

8 Outputs

- A new computational 2D nanosheet database that includes the 3D structure and connectivity (in cif format), along with a unique identifier (MOFid). After further refinement, the initial release of this database will be made freely available via ORDA, the University of Sheffield's online research data hub, providing an open and citeable dataset to the community.

- The software pipeline for construction of the dataset will be made available via GitHub, for hosted version control.
- The LSTM-VAE generative model will also be made available via GitHub.
- A manuscript on the construction of a database of hypothetical 2D nanosheets is in preparation.

9 Conclusions

We have successfully developed a software pipeline that allows for fast construction of a dataset of MONs from their composite building blocks and used this semi-automated approach to produce a database of more than 45000 hypothetical MONs. Each of these structures has an associated MOFid that can be used as a representation in ML/AI models, meaning that the database can be used by other researchers working in this area. The database can be easily extended by selecting different organic linkers and re-running the last few steps of the workflow to construct structures and MOFids. It can also be augmented with property information, such as pore size, depending upon the user's intended application.

We have also constructed the architecture of an LSTM-VAE AI algorithm that can be trained on MOFids to generate new outputs. It has been initially trained on a small database to show that reconstruction is possible, but the progress on the development of this network has been slower than anticipated. It requires further training and fine-tuning before it is coupled with property prediction for the structures generated. This work has allowed us to build a network of collaborators who work well together, even though much of this had to be conducted remotely, who are eager to continue using AI methods to discover new molecules and materials, and to then synthesise them. The project has also allowed us to explore each of the main theoretical phases of automating chemical discovery: generating and curating a suitable dataset with appropriate representations; building a generative model architecture; training the model to produce new structures that can then be refined for experimental synthesis. All of the tools and data produced during the project will be made openly available, both for the continuation of this work, and for use by the molecular/materials discovery community.

10 Future Plans

Improvements to the performance of the LSTM-VAE in generating MON structures are the most immediate steps to be taken in future research, which will continue beyond the scope of this initial project. One route to improvement would be to increase the size and variation of the training database; the software pipeline developed during this work allows for this data to be "synthesised" rapidly in a semi-automated fashion and such database extensions should be easily achievable. Determining how to split structures from the resulting dataset into training and validation sets in an effective way will require further research, although a number of well-established techniques from the machine learning field, such as k-fold cross-validation, are possible candidates. The performance of the generative network will also benefit from further general hyperparameter tuning, including the layer structures and batch sizes.

The subsequent step in this programme of research will be to label the training data with property information (pore size). An additional network will then be added on top of the LSTM-VAE to predict pore size from the latent space, allowing the space to be explored for materials with desirable properties and then inform and direct experiment toward promising new structures. Once the generative model begins to predict MOFids not seen in the training or validation sets, and can generate hypothetical MONs with desired pore sizes, these prediction will be communicated to the experimental team for synthesis and verification. A longer-term

vision is to establish this workflow as part of a "closed-loop" approach, where the resulting experimental data is fed back into the generative model to improve the results.

11 References

[1] Y. J. Colón, D. A. Gómez-Gualdrón and R. Q. Snurr, *Cryst. Growth Des.*, 2017, **17**, 5801–5810.

[2] R. Anderson and D. A. Gómez-Gualdrón, *Cryst. Eng. Comm.*, 2019, **21**, 1653–1665.

[3] Z. Yao, B. Sánchez-Lengeling, N. S. Bobbitt, B. J. Bucior, S. G. H. Kumar, S. P. Collins, T. Burns, T. K. Woo, O. Farha, R. Q. Snurr and A. Aspuru-Guzik, *Nat. Mach. Intell.*, 2021, **3**, 76–86.

[4] S. Hochreiter and J. Schmidhuber, *Neural Computation*, 1997, **9**, 1735–1780.

[5] B. J. Bucior, A. S. Rosen, M. Haranczyk, Z. Yao, M. E. Ziebel, O. K. Farha, J. T. Hupp, J. I. Siepmann, A. Aspuru-Guzik and R. Q. Snurr, *Cryst. Growth Des.*, 2019, **19**, 6682–6697.

[6] D. Weininger, *Journal of Chemical Information and Computer Sciences*, 1988, **28**, 31–36.

[7] A. K. Rappe, C. J. Casewit, K. S. Colwell, W. A. Goddard and W. M. Skiff, *J. Am. Chem. Soc.*, 1992, **114**, 10024–10035.

[8] N. M. O'Boyle, M. Banck, C. A. James, C. Morley, T. Vandermeersch and G. R. Hutchison, *J. Cheminformatics*, 2011, **3**, 33.

[9] N. M. O'Boyle, M. Banck, C. A. James, C. Morley, T. Vandermeersch and G. R. Hutchison, *Open Babel*, https://openbabel.org/, 2020.

[10] T. F. Willems, C. H. Rycroft, M. Kazi, J. C. Meza and M. Haranczyk, *Micropor. Mesopor. Mat.*, 2012, **149**, 134–141.

[11] M. O'Keeffe, M. A. Peskov, S. J. Ramsden and O. M. Yaghi, *Acc. Chem. Res.*, 2008, **41**, 1782–1789.

[12] D. J. Tranchemontagne, J. L. Mendoza-Cortés, M. O'Keeffe and O. M. Yaghi, *Chem. Soc. Rev.*, 2009, **38**, 1257–1283.

[13] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu and X. Zheng, *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*, 2015, http://tensorflow.org/, Software available from tensorflow.org.

[14] F. Chollet and others, *Keras*, https://github.com/keras-team/keras, 2020.

[15] S. Kullback and R. A. Leibler, *Ann. Math. Statist.*, 1951, **22**, 79–86.

12 Data & Software Links

Software/database developed in this work:

- Git repository containing the generative data pipeline and LSTM-VAE architecture

Existing programs/libraries used in this work:

- HDF5 Database
- Keras
- MOFid
- Open Babel
- RDKit
- Tensorflow
- ToBaCCo 3.0
- Zeo++