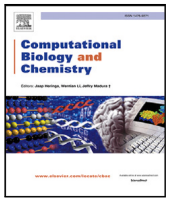
Computational Biology and Chemistry 110 (2024) 108056

Contents lists available at ScienceDirect

Computational Biology and Chemistry

journal homepage: www.elsevier.com/locate/cbac



Research Article

Feature engineered embeddings for classification of molecular data

Claudio Jardim [a,*], Alta de Waal [a], Inger Fabris-Rotelli [a], Najmeh Nakhaei Rad [a], Jocelyn Mazarura [a], Dean Sherry [b]

[a] University of Pretoria, Pretoria, South Africa
[b] SilicoGenesis, Johannesburg, South Africa

ARTICLE INFO

Keywords:
Property prediction
Latent Dirichlet Allocation

ABSTRACT

The classification of molecules is of particular importance to the drug discovery process and several other use cases. Data in this domain can be partitioned into structural and sequence/text data. Several techniques such as deep learning are able to classify molecules and predict their functions using both types of data. Molecular structure and encoded chemical information are sufficient to classify a characteristic of a molecule. However, the use of a molecule's structural information typically requires large amounts of computational power with deep learning models that take a long time to train. In this study, we present an alternative approach to molecule classification that addresses the limitations of other techniques. This approach uses natural language processing techniques in the form of count vectorisation, term frequency-inverse document frequency, word2vec and Latent Dirichlet Allocation to feature engineer molecular text data. Through this approach, we aim to make a robust and easily reproducible embedding that is fast to implement and solely dependent on chemical (text) data such as the sequence of a protein. Further, we investigate the usefulness of these embeddings for machine learning models. We apply the techniques to two different types of molecular text data: FASTA sequence data and Simplified Molecular Input Line Entry Specification data. We show that these embeddings provide excellent performance for classification.

1. Introduction

The process of drug discovery is complex and consists of several stages from target identification through to clinical trials. Machine learning (ML) techniques have proven useful in several stages of drug discovery and in the process of designing novel therapeutics (Deng et al., 2022; Rifaioglu et al., 2019). An exemplary use of these techniques is in molecular property prediction for use in high throughput virtual screening (HTVS) (Oliveira et al., 2023). The goal of HTVS is to filter through large libraries of molecules to find a reduced set of candidates that exhibit favourable properties. Performing experimental validation on a smaller set of candidate molecules can save time and money.

Some of the most commonly used techniques in this space are natural language processing (NLP) and deep learning (DL) (Arabi, 2021; Jarada et al., 2020; Deng et al., 2022; Rifaioglu et al., 2019; Wu et al., 2018). Many of the current methods rely on deep learning models and three-dimensional structural data which results in increased computational cost, compute time and overall complexity. Our aim is to provide a readily reproducible and computationally efficient alternative. Our approach prioritises simple and rapid implementation while minimising resource-intensive calculations, making reproducibility a top priority. Moreover, our method requires fewer software dependencies, hardware requirements, and simplifies the setup of the environment. While we acknowledge the invaluable contributions of existing deep learning models utilising structural data, our goal is to provide an inherently reproducible method that will provide a solid foundation for future work and independent verification.

In this study, we predict the biophysical and physiological behaviour of small molecules, which typically consist of 20 to 100 atoms, which can be expressed in the Simplified Molecular Input Line Entry Specification (SMILES) data type (Weininger, 1988). In particular, we predict blood–brain barrier penetration (permeability) (BBBP) and qualitative binary binding interactions for a set of inhibitors of human β-secretase 1 (BACE) obtained from Moleculenet (Wu et al., 2018), a benchmark created for evaluating ML techniques on molecular property prediction. Additionally, to illustrate the versatility of our method, we

* Corresponding author.
E-mail addresses: claudiomj8@gmail.com, u17029008@tuks.co.za (C. Jardim).

[1] Yang Zhang, What is FASTA format?, 2023, https://zhanggroup.org/FASTA/.
[2] RCSB Protein Data Bank, Homepage, 2023, https://www.rcsb.org/.

https://doi.org/10.1016/j.compbiolchem.2024.108056
Received 17 November 2023; Received in revised form 19 March 2024; Accepted 20 March 2024
Available online 25 March 2024

focus on a protein classification task. Proteins are macromolecules, consisting of 1000s of atoms, formed by one or more chains of monomers termed amino acids. These biological molecules are vital to almost every biological activity and are capable of a myriad of functions. We predict the functional classification of proteins using their residue sequence data in the form of FASTA data[1] on the Structural Protein Sequences (SPS) data set retrieved from the Research Collaboratory for Structural Bioinformatics (RCSB) Protein Data Bank (PDB).[2] We solely make use of the available text data obtained from the FASTA and SMILES files which represent the primary amino acid sequence of protein molecules or the atoms comprising a small molecule, respectively. We create embeddings of this molecular text data using count vectorisation (CVec), term frequency-inverse document frequency (TF-IDF), word2vec (Mikolov et al., 2013) and latent Dirichlet Allocation (LDA) (Blei et al., 2003). While it is true that these methods have been well explored within the field of NLP, we believe their application to molecular text data, presents a novel approach. Particularly, using the LDA topic probability vector as a feature vector. These embeddings reduce the dimensionality of the data, transforming molecular text data to the vector space, whilst keeping relative similarity between observations in their transformed state.

The embeddings are then used for the protein classification and molecular property prediction tasks, making use of three models: a support vector machine classifier (SVM) (Boser et al., 1992), a naïve Bayes classifier (NBC) and a neural network (NN) classifier. We evaluate the usefulness and limitations of the embedding techniques and the ML classification techniques applied to the molecular text data. These models can be improved through hyperparameter tuning in the form of grid search and other techniques, which may be addressed in future studies. The objective of this study is to demonstrate that feature engineering can enhance baseline performance on these tasks or achieve comparable results with significantly reduced complexity. It is important to note that our study is not intended to compare our approach with the latest achievements on the Moleculenet benchmark. We want to make it clear that we do not utilise any structural data or other molecular information. We also do not prioritise optimising our models for classification tasks. Further, we do not pre-train our embedding models or our classification models on data other than the training data available for the dataset in question. Our main focus is to evaluate reliable and reproducible embedding techniques for machine learning classification tasks. To ensure fairness, we use identical models across all embedding techniques for consistent comparison. Our evaluation is based on SMILES and FASTA data to demonstrate the robustness of our techniques. Additionally, we compare our techniques with simpler benchmarks on the Moleculenet benchmark, without considering structural information. Our aim is to show that straightforward embeddings such as ours can perform reasonably well on the benchmark when paired with simpler and more explainable models. We did not deem it fair to compare to techniques that were pre-trained on larger datasets further, we simply could not replicate all of the alternative techniques due to lack of reproducibility or lack of resources to pre-train on these large datasets.

We provide a brief literature review (Section 2) which contextualises these techniques applied to biological data and molecular text data. A concise description of each technique that was utilised is provided (Section 3) with a theoretical and mathematical formalisation that describes the methodology used in this study. A description of the data preprocessing (Section 4) and the details of the experiments conducted for evaluating the use of the embeddings as ML feature vectors are provided (Section 5). Section 6 provides conclusions relating to the study.

2. Literature review

In general, DL and NLP are the main techniques that have been applied specifically to molecular text data (Arabi, 2021; Jarada et al., 2020; Deng et al., 2022; Rifaioglu et al., 2019; Wu et al., 2018). As mentioned previously, most of the approaches implemented to date utilise either structural data[3] or text-based molecular data (Bhatnagar et al., 2022). Structural approaches use information and calculated features based on the components of the three-dimensional structure and chemistry of the molecule and, often, make use of geometric deep learning models (Isert et al., 2023). Molecular text approaches use data such as the SMILES representation of a molecule or the FASTA representation of a protein/DNA/RNA sequence. Given that these data are represented as text it is necessary to convert these representations into numerical formats compatible with ML models. Several molecular text-based embedding methods (Yang et al., 2018; Ibtehaz and Kihara, 2023) have shown promising results for protein classification and molecular property prediction based tasks (Wu et al., 2003; Sarac et al., 2008). Pre-trained transformer models such as ProtAlbert have been used to predict protein sequence profiles (Behjati et al., 2022). Given that embedding techniques for semantic text similarity measurement have been effectively implemented for classical NLP tasks (Shahmirzadi et al., 2019), it follows that NLP techniques can be extended to embed molecular text data. Indeed, text-based embeddings have proven to be useful when using both SMILES data (Wu et al., 2018; Kim et al., 2021; Jaeger et al., 2018) and protein sequence data (Ibtehaz and Kihara, 2023; Yang et al., 2018). Interestingly, the embedding method that is implemented can drastically affect the performance of a model given a specific task (Shahmirzadi et al., 2019).

Commonly implemented global embedding techniques include CVec and TF-IDF (Elkan, 2005). Furthermore, LDA is a widely employed technique in NLP and can be employed as an embedding technique. LDA, initially proposed for population genetics (Pritchard et al., 2000), characterises each topic by a distribution over words, and documents are represented as random mixtures over these latent topics (Blei et al., 2003). To date, LDA has been used in protein-related prediction tasks based on structural data such as protein function prediction and protein folding prediction (Xiao et al., 2017; Schneider et al., 2017; Singh et al., 2012). Using substructures of molecules as words (van Der Hooft et al., 2016; Kikuchi and Kikuchi, 2021; Jaeger et al., 2018), LDA has been successfully applied to structural data for protein structure comparison (Shivashankar et al., 2011). Hence, it is evident that, LDA applied to structural data can efficiently represent protein structures and be utilised to compare these representations based on similarity. LDA has also been utilised in the analysis of genetic sequence data relating to gene expression (Yalamanchili et al., 2017). As such, this observation suggests that LDA could be effectively applied to protein sequence data for a number of tasks. Notably, support vector machines have been shown to work well in conjunction with LDA (Chen and Li, 2016). Additionally local embedding techniques (Jurafsky and Martin, 2021) using self-supervised learning have been used to learn a semantic embedding of molecular text data (Kim et al., 2021; Jaeger et al., 2018). Many of such self-supervised techniques are examples of word2vec (Mikolov et al., 2013), a group of related NN models used to compute continuous vector representation of a word. This approach has been successfully applied to SMILES data to acquire embeddings of molecular substructures (Jaeger et al., 2018). Both global and local embeddings serve as a feature for many ML models, ranging from classical linear models to modern deep learning techniques (Wu et al., 2018; Li et al., 2023). These embeddings find diverse applications including classification tasks such as property prediction (Kim et al., 2021; Ibtehaz and Kihara, 2023; Wu et al., 2018).

[3] Papers with Code, Molecular Property Prediction, 2023, https://paperswithcode.com/task/molecular-property-prediction.

3. Materials and methods

Biological molecules vary significantly in size, leading to data with varying dimensions. In order to address variability in the lengths of the molecular text data, we create embedding vectors of equal dimension for every molecule. Word embeddings generate numerical vectors of uniform dimensionality, encoding the meaning of data such that similar text data are situated closer to each other within the vector space. Consequently, each molecule is assigned a numerical vector representation of the same dimension, and the distance between these vector representations indicates the degree of similarity between molecules. CVec, TF-IDF, LDA and word2vec are all techniques capable of generating such uniform vector representations. Furthermore, embedding techniques can be grouped into global and local embeddings which we discuss briefly[4] (Liang et al., 2018; Jurafsky and Martin, 2021), as well as an overview of SVMs, NBCs and the NN architecture utilised in this study.

3.1. Global embeddings

Global embeddings (Jurafsky and Martin, 2021) typically featurise the entire composition of a document or molecule by utilising counts or statistics at the document or molecule level.

3.1.1. Count vectorisation

CVec is a technique that transforms text data into numerical data (Wallach, 2006; Zhang et al., 2010). A method of doing this is by counting the occurrence of words in the document. For a molecule $\mathbf{x}$, the count vector can be expressed as $[x_1, \dots, x_R]$, where R is the total number of residues or text items in the molecule (vocabulary size) and x_r is the number of times molecular text item or residue r appears in molecule $\mathbf{x}$. An n-gram model can also be used for CVec (Kondrak, 2005). An n-gram is a contiguous set of n words from a sample of text data. The use of n-grams on molecular text data may be very beneficial as it can introduce relations such as combining an atom with the amino acid residue it is a part of. Beyond this n-grams may also be beneficial when applied to amino acid residues alone.

3.1.2. Term frequency-inverse document frequency

The TF-IDF is a statistic that measures the importance of a word or item to a document or file that is in a text corpus (Elkan, 2005; Salton and Buckley, 1988; Luhn, 1957), and it is one of the most widely used weighting schemes for text data. The TF-IDF is the combination of two statistics namely, term frequency and inverse document frequency. Term frequency is how many times a term (residue or molecular text item represented by t) appears in the document or molecule m (Luhn, 1957). A log normalised or logarithmically scaled version of the term frequency, $\log tf$, is given by:

$$\log tf(t,m) = \frac{\log(1 + \text{count of } t \text{ in } m)}{\text{number of terms in } m}.$$

Inverse document frequency is the amount of information the term provides (Robertson, 2004). The inverse document frequency, idf, can be obtained by logarithmically scaling the total number of molecules divided by the number of molecules that contain the term.

$$idf(t,M) = \log\left(\frac{|M|}{m \in M \,:\, t \in m}\right),$$

where M is the corpus of molecules and $|M|$ is the total number of molecules in the corpus. Using the definitions and equations for term frequency and inverse document frequency one can obtain the formula of term frequency-inverse document frequency (Elkan, 2005; Salton and Buckley, 1988; Luhn, 1957):

$$TF\text{-}IDF(t,m,M) = \log tf(t,m) \cdot idf(t,M).$$

TF-IDF proves valuable in eliminating common terms by assigning a higher value or weight to terms with a lower document frequency across the corpus and a higher term frequency within the document. This TF-IDF value serves as a weight, quantifying the significance of a term within a document or molecule.

In the case of molecular text data atoms or amino acid residue represent the terms of the molecule. As such, each molecule file represents a document and the full dataset of molecules files represents the corpus.

3.1.3. Latent Dirichlet allocation

As mentioned previously, in the context of LDA (Blei et al., 2003) a molecule is equivalent to a document and a residue or molecular text item is equivalent to a word or term. In the context of a corpus of molecules C, LDA can be described as a generative statistical topic model where topics are represented by a distribution over molecular text items and random mixtures of these latent topics represent molecules in the corpus. To uncover the topics within a corpus, it is most straightforward to approach the problem through reverse engineering. As such, we employ a generative process for each molecule, aiming to identify the corpus' topics. First, however, we must generate the molecules within the corpus. To do this, we can use a generative process for each molecule $\mathbf{x}_m$ in the corpus C with M the number of molecules in the corpus.

- Let $\{1, \dots, T\}$ be the vocabulary of molecular text items. A molecular text item $x_1, \dots, x_T$ is the basic unit of a molecule.
- A molecule $\mathbf{x}_m$ is a combination of molecular text items denoted by $\mathbf{x_m} = [x_1, \dots, x_{N_m}]$, where x_n is the nth molecular text item in the molecule and N_m is the number of molecular text items in molecule $\mathbf{x_m}$ for $m \in \{1, \dots, M\}$.
- A corpus C is a collection of M molecules denoted as $M = [\mathbf{x_1}, \dots, \mathbf{x_M}]$.

For each molecule $\mathbf{x}_m$ in a corpus C a generative process can be assumed (Blei et al., 2003; Pritchard et al., 2000) with latent variables from probability distributions with parameters equal to the known variables:

1. Draw a topic–molecule distribution $\boldsymbol{\theta_m}$ from a Dirichlet distribution. $\boldsymbol{\theta_m} \backsim Dir(\boldsymbol{\alpha})$, where $m \in \{1, \dots, M\}$ with $\boldsymbol{\alpha}$ a vector of dimension equal to the number of topics K, where, $\sum_{k=1}^{K} \theta_{m,k} = 1$ and $\theta_{m,k} \in [0,1]$ for all $k \in [1, \dots, K]$.
2. For each of the molecular text items $x_{m,n}$, where $m \in \{1, \dots, M\}$, and $n \in \{1, \dots, N_M\}$:
   (a) Generate a topic $z_{m,n} \backsim$ multinomial $(\boldsymbol{\theta_m})$.
   (b) Generate a molecular text item $x_{m,n} \backsim p(x_{m,n}|z_{m,n}, \boldsymbol{\beta})$ a multinomial probability conditioned on the topic $z_{m,n}$, where $\boldsymbol{\beta}$ is the parameter of the Dirichlet prior on the per-topic molecular text item distribution.

This equates to solving the following equation:

$$p(\boldsymbol{\theta}, \mathbf{z}|\mathbf{x}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{p(\boldsymbol{\theta}, \mathbf{z}, \mathbf{x}|\boldsymbol{\alpha}, \boldsymbol{\beta})}{p(\mathbf{x}|\boldsymbol{\alpha}, \boldsymbol{\beta})}.$$

Approximate inference techniques such as variational inference (Blei et al., 2003) need to be applied to the problem since the normalisation factor $p(\mathbf{x}|\boldsymbol{\alpha}, \boldsymbol{\beta})$, cannot be exactly computed and so the distribution is intractable. Using plate notation we provide an intuitive explanation of LDA (Anastasiu et al., 2013). Here the boxes are referred to as "plates". The outer plate represents molecules, while the inner plate represents the repeated positions of the molecular text items which are associated with a choice of topic and molecular text item.

[4] Turing, Word embeddings in NLP: A complete guide, 2022, https://www.turing.com/kb/guide-on-word-embeddings-in-nlp.

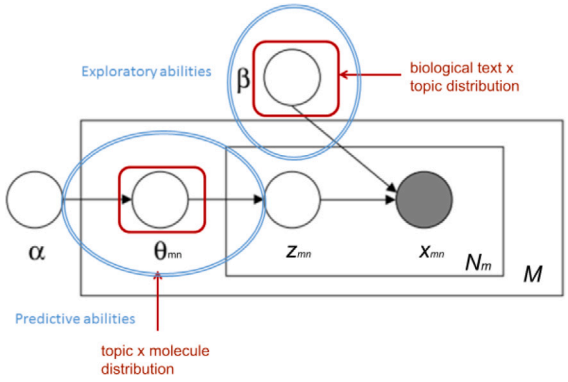


Fig. 1. Plate notation for the LDA model where, M is the number of molecules in the corpus C, N_m is the number of molecular text items in a given molecule m, $\boldsymbol{\alpha}$ is a corpus-level parameter of the Dirichlet prior on the per-molecule topic distribution, $\boldsymbol{\beta}$ is a corpus-level parameter of the Dirichlet prior on the per-topic molecular text item distributions, $\boldsymbol{\theta_m}$ is the topic distribution for a given molecule, z_{mn} is the topic for the n-th molecular text item in molecule m and x_{mn} is the specific molecular text item.

In the plate notation provided in Fig. 1, the only observed variable is x, hence it is coloured grey. All of the other variables are latent variables. Using the view provided in Fig. 1, the molecular text–topic distributions is a matrix with topics as rows and columns defined by molecular text items. The variable $\boldsymbol{\theta}$ can be viewed as a matrix of topic–molecule distributions, where the rows would be individual molecules and the columns would be topics. Thus, each row of $\boldsymbol{\theta}$ is a distribution over topics and each row of the molecular text–topic distributions is a distribution over molecular topics (Blei et al., 2003; De Waal and Barnard, 2008) as can be seen in Fig. 1. The molecular text–topic distribution matrix and topic–molecule distribution matrix can be viewed as the decomposition of the original molecule-molecular text item matrix that represents the corpus of documents being modelled. Thus, LDA can also be thought of as a dimensionality reduction technique (Blei et al., 2003; Crain et al., 2012) where the corpus is represented as an embedding in a lower dimensional form using the topic–molecule distribution matrix. The advantage of this approach is that the interpretable topics should create a semantic embedding of the molecules. Under the LDA model, a molecule can be associated with more than one topic (Blei et al., 2003). Thus, the topic probability vector for a molecule will be of dimension K. For a molecule x_m the topic probability vector can be represented as:

$$\boldsymbol{\theta_m} = [p(\alpha_1), \ldots, p(\alpha_K)],$$

where $p(\alpha_k)$ is the probability of the molecule being associated with topic k. The LDA embedding will reduce the dimension of the molecular text items in a molecule to a vector with dimension equal to the number of topics chosen. This molecule–topic matrix will be a semantic embedding for the corpus of molecules. Initial results saw a decrease in performance for more than 100 topics. For this reason, we capped the number of topics to 100.

3.2. Local embeddings

Local embeddings take the local features of a molecular text item into account (Jurafsky and Martin, 2021), often making use of a window that summarises the local neighbourhood of a molecular text item. As such, in contrast to global embeddings, local embeddings generate a local descriptor at the molecular text item or word level.

Word2vec uses a group of related neural network models to compute a continuous vector representation of words (Mikolov et al., 2013). These models are shallow networks, consisting of only two layers. The continuous vector of each word is chosen such that the cosine similarity between vectors is an indicator of the true semantic similarity between the words. In this way, a corpus of molecules can be transformed into a vector space, where each word in the corpus is associated with a distinct embedding vector within this feature space. Word2vec can either make use of continuous bag-of-words (CBOW) or continuous skip-gram model architectures (Mikolov et al., 2013). The CBOW model takes into account the local window of words for the current word to create its embedding. In this way word2vec takes local features into account to create a local feature embedding vector for a word in contrast to a global embedding at the level of a document. The skip-gram model does the opposite of the CBOW model. It takes the current word and uses it to predict the local window of words. Again, the word2vec model can be applied and trained on a corpus of molecules each of which are comprised of molecular text items.

There are a few considerations to be taken into account with regard to local embedding. For instance, it is often helpful to subsample words that have a frequency above a certain threshold. Subsampling frequent words such as “a” and “the” allows a model to focus on words that occur less frequently and have more information. Additionally, the dimension of the embedding vector is directly related to the quality of the embedding. Thus it is beneficial to increase the dimension of the embedding vector until the performance metrics stop improving or the cost of increasing the dimension is too high (Mikolov et al., 2013). Lastly, the size of the context window can also be adjusted to improve the quality of the embeddings (Mikolov et al., 2013). The context window is the number of words around the target word that are used by the model to make predictions for the target word representation (Mikolov et al., 2013).

3.3. Classifiers

We make use of Support vector machines (SVM), Naïve Bayes classifier (NBC) and a simple neural network (NN) classifier. Here, we provide a brief overview of each classifier and describe the architecture of the NN. Since the focus of this study involves the feature-engineered embeddings, and not the models themselves, we assume default parameters (Pedregosa et al., 2011) for all models unless otherwise stated.

3.3.1. Support vector machine

SVMs are classification techniques (Boser et al., 1992) which map training data points such that the distance between classes is maximised. The SVM classifier identifies an optimal hyperplane that effectively separates the training data points, thereby enabling accurate classification. To make predictions, new data points are projected into the same mapped space and assigned the class that they fall onto. SVMs are also capable of handling non-linear classification tasks through the utilisation of the kernel trick. Moreover, SVMs are memory efficient, versatile and effective in high-dimensional spaces. We make use of the scikit-learn[5](Pedregosa et al., 2011) implementation of an SVM classifier with a radial basis function. For all other parameters the default *scikit-learn* parameters were used.

3.3.2. Naïve Bayes classifier

An NBC is a family of probabilistic classifiers based on Bayes' theorem. They are examples of a simple Bayesian network where all features or embeddings are assumed to be independent and are thought to contribute equally to the outcome. We use the scikit-learn *MultinomialNB*[6] (Pedregosa et al., 2011) classifier for the CVec embeddings and the *GaussianNB* for all other embeddings tested. We use a multinomial NBC for the CVec embeddings since these embeddings reflect the frequency with which terms occur. Given the continuous nature of the remaining embeddings, we employ a Gaussian Naïve Bayes Classifier under the assumption that the embeddings for each class follow a normal distribution.

[5] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
[6] https://scikit-learn.org/stable/modules/naive_bayes.html

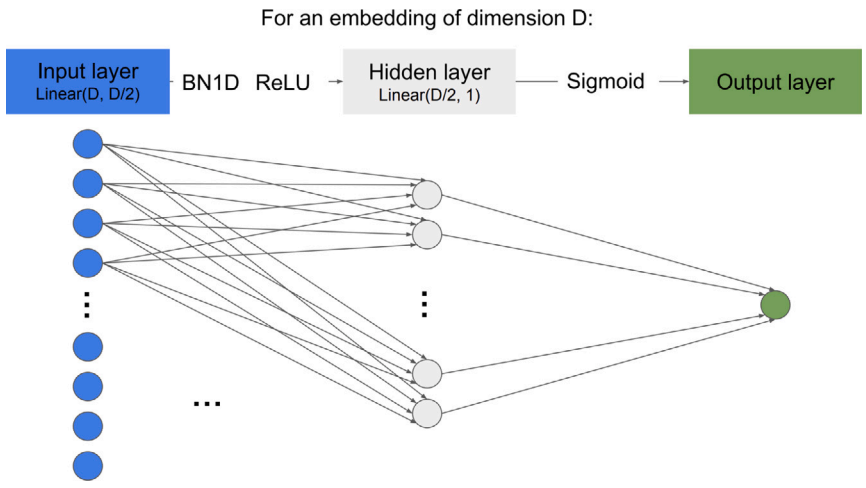


Fig. 2. NN classifier architecture.

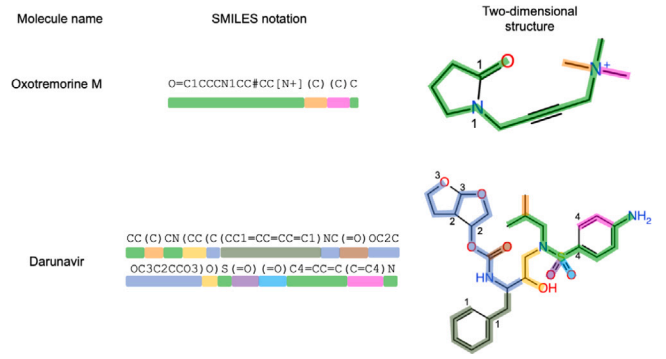


Fig. 3. A representation of two molecules using SMILES notation and their corresponding two-dimensional structures. The continuous backbone of connected atoms is coloured green, while branched groups are coloured differently for clarity.

3.3.3. Neural network architecture

We train a simple NN classifier for each embedding type on each dataset using the PyTorch Python library (Paszke et al., 2019). If the embedding has dimension D then the classifier has an input layer of dimension D, a hidden layer of dimension $D/2$ and an output layer with the final predictions. An illustration of the architecture is provided in Fig. 2. We use batch normalisation and rectified linear unit (ReLU) activation between the input layer and hidden layer and a sigmoid activation for all predictions so as to ensure that the output values are confined within the range of 0 to 1, a convention commonly adhered to in binary classification tasks. Each model was trained for 50 epochs with a batch size of 200. The epoch with the smallest loss was used for the evaluation metrics. In the NN architecture provided in Fig. 2: *BN1D* represents a batch normalisation layer, *ReLU* shows that rectified linear unit activation is applied, *Sigmoid* shows that the element-wise sigmoid function is applied and D represents the dimension or size of the input. This will usually be the dimension of the embedding.

4. Data and data preprocessing

During data preprocessing, duplicates were systematically identified and removed from each dataset, ensuring data consistency and integrity. Furthermore, any null or missing values within the datasets were effectively addressed and rectified. Following these initial quality control measures, each dataset underwent specific processing steps tailored to the characteristics of the data type used.

4.1. SMILES data

The BBBP and BACE datasets are both comprised of SMILES data and were obtained from Moleculenet (Wu et al., 2018). Small molecule are often represented as a three-dimensional structures, as a two-dimensional projection or using SMILES string notation. Fig. 3 illustrates the relationship between a two-dimensional representation of a molecule and its corresponding SMILES string notation. Put simply, a SMILES generation algorithm begins by breaking each cyclic ring at an arbitrary point, making an acyclic structure. The addition of numerical ring closure labels to show connectivity between non-adjacent atoms in the SMILES notation. Following this, a continuous backbone of connected atoms is selected as the basis of the molecule and branches are defined where groups of atoms extend off of the main backbone. Branches are described with parentheses where the first atom within the parentheses, and the first atom after the parenthesised group are both bonded to the same branch point atom.

When considering the SMILES datasets we make use of the rdkit[7] (Landrum et al., 2016) and mol2vec[8] (Jaeger et al., 2018) Python packages. Using these two packages we can divide each molecule into distinct substructures by applying a specified fixed radius. Typically, a substructure comprises the group of atoms closest to a given heavy atom within the specified radius. Subsequently, each substructure is then encoded using a Morgan fingerprint which is a bit vector representation of a molecule (Ding et al., 2021; Glem et al., 2006). The presence or absence of a substructure is represented by each bit. In this manner, the substructures effectively become the words of a molecular sentence, with each word distinguished by its unique Morgan fingerprint. Morgan fingerprints can be used to measure the chemical similarity between molecules (Glem et al., 2006) and have properties that benefit our approach. Consequently, each molecular sentence corresponds to a molecule, and these sentences are now amenable to being inputted into the embedding techniques utilised in this study in order to get a vector representation of each molecule.

4.2. FASTA data

The FASTA SPS dataset is a dataset of extracted FASTA sequences. The FASTA format is a text-based format in which represents the primary structure of a protein which is the linear sequence of its amino acids. By convention, the amino acid sequence of a protein is read and written from the amino terminal to the carboxyl terminal with

[7] https://www.rdkit.org/
[8] https://github.com/samoturk/mol2vec

Table 1
Finding the best embedding for the SVM.

Data type	Dataset	Technique	Accuracy ± (SD)%	AUC ± (SD)%	F1 ± (SD)%
Smiles	BACE	CVec	81.85 ± 1.12×10^{-14}	81.62 ± 0.00	80.14 ± 2.24×10^{-14}
		TF-IDF	**85.48 ± 1.1** × 10^{-14}	**85.41 ± 1.12** × 10^{-14}	**84.62 ± 1.12** × 10^{-14}
		LDA	72.00 ± 2.86	71.67 ± 2.91	68.71 ± 3.72
		word2vec	66.67 ± 1.12×10^{-14}	66.01 ± 1.12×10^{-14}	60.08 ± 1.12×10^{-14}
	BBBP	CVec	87.75 ± 2.24×10^{-14}	76.21 ± 1.12×10^{-14}	92.49 ± 1.12×10^{-14}
		TF-IDF	**91.18 ± 1.12** × 10^{-14}	**82.39 ± 0.00**	**94.55 ± 0.00**
		LDA	85.55 ± 1.18	73.11 ± 2.20	91.14 ± 0.72
		word2vec	80.64 ± 0.00	56.91 ± 0.00	88.89 ± 0.00
FASTA	SPS	**CVec**	**74.01 ± 1.12** × 10^{-16}	**73.05 ± 0.00**	68.66 ± 0.00
		TF-IDF	73.18 ± 0.00	72.68 ± 1.12×10^{-16}	68.91 ± 2.24×10^{-16}
		LDA	72.70 ± 0.50	72.47 ± 0.52	**69.07 ± 0.61**
		word2vec	70.92 ± 0.15	69.86 ± 0.17	64.81 ± 0.24

amino acids represented using their single letter codes for example "G" for Glycine. A description line, or header, precedes the sequence information and begins with ">". The header gives a name or a unique identifier for the sequence, and may also contain additional information about the sequence such as the source organism. Fig. 4 illustrates the relationship between the three-dimensional structure of a protein and its corresponding FASTA sequence.

For the FASTA SPS dataset we first join the labels to the protein sequences using the raw datasets obtained from Kaggle.[9] Following this, we proceed to filter the molecules to select only those that refer to protein sequences. Subsequently, we further refine the dataset by filtering it to contain sequences categorised within the top two classes with the highest sample counts. We perform this step to transform the problem into a binary classification task in order to simplify the architecture of our models. To facilitate subsequent processing, we insert a space between each amino acid residue in the sequence. Additionally, we omit the letter "X" from the sequences since it can potentially represent any amino acid residue, therefore, rendering it non-informative for the classification task.

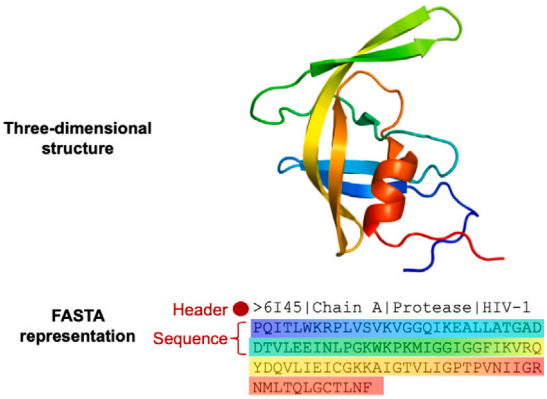


Fig. 4. A representation of a molecule using FASTA notation and its corresponding three-dimensional structure.

5. Results

We run 5 experiments to compare the combination of embeddings (as features) with NN, NBC and SVM models. Unless stated otherwise, we make use of an 80%:20% train–test split of the data for all of the models. Additionally, we set the random state to 0 for the training and test splits for all of the experiments. Due to the large combination of models and datasets, we opted not to perform hyper-parameter tuning. Our focus was on employing straightforward model architectures with default parameters in an effort to maintain simplicity and consistency across our experiments. LDA embeddings of dimension 10, 20, 50 and 100 were considered for each dataset and model combination while embedding dimensions of 100, 200 and 300 are considered for word2vec embeddings. The dimension that provided the best results for the dataset and model combination were used unless stated otherwise. Higher dimensions (greater than 300) were not considered due to initial experiments showing a decrease in performance for larger dimensions and compute limitations for both LDA and word2vec embeddings. Future work could focus on hyper-parameter tuning and larger embedding dimensions. All experiments and coding have been done in Python (Python 3.10). The experiments were run on a Lenovo machine using Ubuntu 20.04 with an Intel i7-8565U processor, 2 GB NVIDIA GeForce MX230 and 12 GB of RAM.[10] Each experiment was run 50 times. The mean and standard deviation of the accuracy,[11] the area under the receiver operating characteristic curve (AUC) and F1 score (Sammut and Webb, 2017) are calculated for each experiment across the 50 runs.[12] We find the best embedding technique for each dataset for the SVM, NBC and NN. We then compare the best embedding and model combination for each dataset. Lastly, we compare our best embedding and model combination to the Moleculenet benchmark metrics.

5.1. Finding the best embedding for the SVM

We compare the different embeddings in terms of accuracy, AUC and F1 score for the SVM. We use the optimal embedding dimension for the word2vec and LDA embeddings that achieved the best results in previous experiments. We use LDA embeddings of dimension 50 for the FASTA SPS dataset and dimension 100 for the BACE and BBBP dataset. We use word2vec dimensions of 100 for the FASTA SPS dataset and for the BACE and BBBP datasets we use a pre-trained model (Jaeger et al., 2018) that outputs word2vec embeddings of dimension 300. The word2vec and LDA embeddings are not the best embedding technique for any dataset for the SVM according to Table 1. It appears that the best embeddings for the SVM are TF-IDF for the BACE and BBBP datasets (85.48% and 91.18% average accuracy) and CVec for the FASTA SPS dataset (74.01% average accuracy). It is interesting to note that TF-IDF seems to perform better for SMILES datasets whilst CVec performs better for the FASTA datasets for the SVM.

5.2. Finding the best embedding for the NBC

We compare the different embeddings in terms of accuracy, AUC and F1 score for the NBC. We use the optimal embedding dimension

[9] https://www.kaggle.com/datasets/shahir/protein-data-set?resource=download&select=pdb_data_seq.csv

[10] It took less than 10 min to train and run predictions for the longest single run on the largest dataset on the machine described in Section 5.

[11] Accuracy can be calculated as the number of correct predictions divided by the total number of predictions (Sammut and Webb, 2017).

[12] The code is available at https://github.com/Claudmj/Feature-engineered-embeddings-for-classification-of-molecular-data.

Table 2
Finding the best embedding for the NBC.

Data type	Dataset	Technique	Accuracy ± (SD)%	AUC ± (SD)%	F1 ± (SD)%
Smiles	BACE	**CVec**	**68.65 ± 1.12 × 10^{-16}**	**69.80 ± 0.00**	**73.83 ± 0.00**
		TF-IDF	67.66 ± 0.00	68.85 ± 1.12 × 10^{-16}	73.22 ± 1.12 × 10^{-16}
		LDA	60.77 ± 4.50	62.10 ± 3.96	68.23 ± 2.89
		word2vec	68.65 ± 1.12 × 10^{-16}	68.39 ± 1.12 × 10^{-16}	65.70 ± 1.12 × 10^{-16}
	BBBP	CVec	65.69 ± 0.00	**72.41 ± 0.00**	73.28 ± 0.00
		TF-IDF	65.44 ± 0.00	71.06 ± 1.12 × 10^{-16}	73.35 ± 1.12 × 10^{-16}
		LDA	59.22 ± 18.07	64.33 ± 6.47	62.80 ± 25.29
		word2vec	**75.25 ± 1.12 × 10^{-16}**	61.42 ± 0.00	**84.44 ± 1.12 × 10^{-16}**
FASTA	SPS	CVec	43.71 ± 5.61 × 10^{-17}	50.50 ± 0.00	60.29 ± 1.12 × 10^{-16}
		TF-IDF	45.57 ± 0.00	51.94 ± 1.12 × 10^{-16}	60.70 ± 0.00
		LDA	64.05 ± 0.99	**66.00 ± 1.01**	**65.67 ± 1.28**
		word2vec	**65.30 ± 1.31**	65.02 ± 1.47	60.93 ± 2.20

Table 3
Finding the best embedding for the NN.

Data type	Dataset	Technique	Accuracy ± (SD)%	AUC ± (SD)%	F1 ± (SD)%
Smiles	BACE	**CVec**	**88.00 ± 1.22**	**93.57 ± 0.48**	**87.29 ± 1.49**
		TF-IDF	83.58 ± 1.33	91.04 ± 0.76	82.62 ± 1.32
		LDA	74.30 ± 2.24	81.83 ± 2.26	71.92 ± 2.61
		word2vec	69.34 ± 1.16	76.01 ± 0.84	67.10 ± 2.48
		CVec	81.00 ± 16.08	83.22 ± 16.10	83.48 ± 16.09
		TF-IDF	**84.93 ± 13.25**	**86.77 ± 13.39**	**87.88 ± 13.30**
	BBBP	LDA	78.56 ± 16.15	79.91 ± 15.98	82.31 ± 16.08
		word2vec	58.26 ± 12.35	54.46 ± 12.28	64.09 ± 12.31
FASTA	SPS	**CVec**	**73.55 ± 0.54**	**81.05 ± 0.34**	**68.68 ± 0.91**
		TF-IDF	72.43 ± 0.51	79.71 ± 0.37	67.48 ± 0.95
		LDA	71.36 ± 0.72	78.83 ± 0.66	66.66 ± 1.14
		word2vec	71.40 ± 0.57	79.25 ± 0.42	66.48 ± 1.33

for the word2vec and LDA embeddings and compare it with count vectorisation (CVec) and TF-IDF. We use LDA embeddings of dimension 100 for the BACE dataset, dimension 10 for the FASTA SPS dataset and BBBP dataset. We use word2vec dimensions of 100 for the FASTA SPS dataset and for the BACE and BBBP datasets we use a pre-trained model (Jaeger et al., 2018) that outputs word2vec embeddings of dimension 300. CVec is the best embedding technique for the NBC for the BACE dataset as seen in Table 2 with an average accuracy of 68.65%. Word2vec embeddings achieve 75.25% average accuracy on the BBBP dataset and 65.30% average accuracy on the FASTA SPS dataset.

5.3. Finding the best embedding for the NN

We compare the different embeddings in terms of accuracy, AUC and F1 score for the NN. We use the optimal embedding dimension for the word2vec and LDA embeddings. We use LDA embeddings of dimension 100 for the BACE dataset, BBBP dataset and FASTA SPS dataset. We use word2vec dimensions of 200 for the FASTA SPS dataset and for the BACE and BBBP datasets we use a pre-trained model (Jaeger et al., 2018) that outputs word2vec embeddings of dimension 300.

CVec is the best embedding technique for the NN on the BACE and FASTA SPS datasets as seen in Table 3. CVec embeddings achieve 88.00% average accuracy on the BACE dataset and 73.55% average accuracy on the FASTA SPS dataset. TF-IDF is the best embedding technique for the NN on the BBBP dataset with 84.93% average accuracy.

5.4. Finding the best embedding and model combination for each dataset

We find and compare the best embedding and model combination for each dataset in terms of accuracy, AUC and F1 score. The model and embedding combinations used are shown in Table 4. CVec and the NN model were the best combination for the BACE dataset (average accuracy 88.25% Table 5), TF-IDF and a SVM was the best for the BBBP dataset (average accuracy 91.18% Table 5) and CVec and a SVM was the best for the FASTA SPS dataset (average accuracy 74.01% Table 5).

Table 4
The model and embedding combination used to find the best model and embedding combination for each dataset.

Model	Dataset		
	BACE	BBBP	SPS
SVM	TF-IDF	TF-IDF	CVec
NBC	CVec	Word2vec	LDA
NN	CVec	TF-IDF	CVec

5.5. Comparison with Moleculenet

Lastly, we compare our approach to the Moleculenet (Wu et al., 2018) using a "scaffold" split[13] from the deepchem package to compare to the Moleculenet results in Table 6. A scaffold split is more "difficult" than a random split as it splits the molecules according to the core structure or scaffold of a molecule resulting in structurally different training and test sets (Ramsundar et al., 2019). We compare the best embedding and model combination for the BACE and BBBP datasets, in terms of AUC, to the best model presented by Moleculenet (Wu et al., 2018). Our best model and embedding combination for both of these datasets is CVec with a NN model.

Our method scores a higher AUC (83.67% compared to 72.90% Table 6) for the BBBP dataset and a slightly lower AUC (85.13% compared to 86.70% Table 6) for the BACE dataset Table 6. We only use the SMILES strings, yet our method scores higher or only slightly lower than alternative approaches that use more features.

5.6. Discussion

It should be noted that there is little to no variation between the 50 runs for CVec and TF-IDF embeddings in Tables 2 and 1.

[13] A scaffold split attempts to separate molecules according to their structure such that structurally different molecules will be in different subsets (Bemis and Murcko, 1996).

Table 5
Finding the best embedding and model combination for each dataset.

Dataset	Model	Embedding	Accuracy ± (SD)%	AUC ± (SD)%	F1 ± (SD)%
BACE	SVM	TF-IDF	85.48 ± 1.12×10^{-14}	85.41 ± 1.12×10^{-14}	84.62 ± 1.12×10^{-14}
	NBC	CVec	68.65 ± 1.12×10^{-16}	69.80 ± 0.00	73.83 ± 0.00
	NN	**CVec**	**88.00 ± 1.22**	**93.57 ± 0.48**	**87.29 ± 1.49**
BBBP	**SVM**	**TF-IDF**	**91.18 ± 1.12** $\times 10^{-14}$	82.39 ± 0.00	**94.55 ± 0.00**
	NBC	word2vec	75.25 ± 1.12×10^{-16}	61.42 ± 0.00	84.44 ± 1.12×10^{-16}
	NN	TF-IDF	84.93 ± 13.25	**86.77 ± 13.39**	87.88 ± 13.30
SPS	**SVM**	**CVec**	**74.01 ± 1.12** $\times 10^{-16}$	73.05 ± 0.00	**68.66 ± 0.00**
	NBC	LDA	64.05 ± 0.99	66.00 ± 1.01	65.67 ± 1.28
	NN	CVec	73.55 ± 0.54	**81.05 ± 0.34**	68.68 ± 0.91

Table 6
Comparison with Moleculenet.

Dataset	Method	Auc%
BBBP	Moleculenet	72.90
	Ours (TF-IDF and NN)	**83.67**
BACE	**Moleculenet**	**86.70**
	Ours (CVec and NN)	85.13

This is because these embeddings will be the same between runs due to the deterministic nature of CVec and TF-IDF. The classification models exhibit minimal variability due to the consistent use of the same random state for data splitting across multiple experimental runs. However, more significant variability was observed in runs involving LDA and word2vec embeddings. This divergence is due to the necessity of training new LDA or word2vec models for each run. Consequently, the embeddings generated in these runs display slight discrepancies, leading to variation in the embeddings across different runs. This coupled with the variation in the trained classifiers results in greater variation than for the CVec and TF-IDF embedding runs Tables 2 and 1. In particular, there is greater variation for the LDA embeddings and the NBC and NN model for the BBBP dataset as can be seen in Tables 2 and 3. We hypothesise that this is due to the large vocabulary size of 2115 which is almost equal to the number of data points for the BBBP dataset as well as the general sensitivity of NNs to the initialisation of weights. It seems that this dataset is particularly difficult to separate between the two classes when using LDA embeddings. Future work could focus on further investigation of the causes for the observed variability between runs. CVec and TF-IDF appear to be the best embedding techniques (see Table 5). Notably, CVec embeddings consistently outperformed TF-IDF, LDA and word2vec when considering the FASTA SPS dataset for the NN and SVM. We hypothesise that this is because FASTA sequences consist of a finite set of amino acids, which means there is no ambiguity in the vocabulary and CVec works well in such scenarios where the vocabulary is well-defined and limited. Additionally, in biological sequences, the frequency of specific amino acids or motifs can be biologically meaningful and CVec is able to capture this information effectively by considering word frequency. Among the models assessed across all datasets, CVec consistently emerged as the top-performing embedding method. In contrast, TF-IDF embeddings, which involve normalisation, exhibited slightly lower performance metrics compared to CVec due to the loss of crucial information. Interestingly, the global embedding techniques outperformed the local embeddings, such as word2vec. Overall, our approach yielded superior metrics, particularly excelling with the BBBP dataset. However, it achieved a slightly lower AUC for the BACE dataset.

6. Conclusion

In this research, we derived latent embeddings for molecular text data and evaluated the use of these embeddings as features for molecular property prediction and protein family classification. CVec, TF-IDF, LDA and word2vec embeddings provide a useful transformation of molecular text data to a vector space. These embeddings reduce the dimensionality of the data whilst keeping relative similarity between observations in their transformed state. In this way, a small amount of data can be used to understand a dataset.

For use as feature vectors for ML models, CVec and TF-IDF were the best performing embedding techniques of those evaluated in this study. It is likely that the normalisation utilised in the TF-IDF embeddings results in the lost of some information and, thus, TF-IDF exhibited slightly poorer performance in comparison to CVec. The LDA embeddings achieved lower metrics than those obtained through TF-IDF which could be attributed to the topic–molecule distribution of the LDA embeddings which may not have captured enough relative information to be useful as a feature vector for the ML models. Furthermore, it can be argued that the increased variation in training the LDA embeddings may have further contributed to the average lower performance of the LDA embedding technique. However, we believe that the LDA embeddings could add robustness and versatility due to their ability to represent corpus information effectively. Interestingly, the embedding of local features using word2vec did not improve classification performance and it was observed that the local embedding techniques performed worse than all of the global embedding techniques except for the NBC on the BBBP dataset. This finding suggests that it may be more useful for future studies to utilise and further investigate the use of global embeddings. Overall, in comparison to the MoleculeNet benchmark (Wu et al., 2018), our approach achieved better performance for the BBBP dataset and a slightly lower AUC for the BACE dataset. These results are particularly encouraging since our methodology achieves similar or better results whilst being computationally efficient. Future work should investigate fine-tuning the classifiers to further improve performance since, in this study, the default arguments were utilised for all classifiers tested.

Additionally, the use of learnt embeddings in machine learning tasks holds significant potential for improving classifier and regressor architectures and enhancing prediction outcomes. When considering proteins, evolutionary information such as protein superfamilies, families and homologues is likely to be captured in a global embedding. Our study has shown that these embedding techniques are useful in specific tasks and have promising applications. However, it is worth noting that we did not compare to the latest result on the Moleculenet benchmark. Instead, we prioritise the evaluation of reliable and reproducible embedding techniques for machine learning classification tasks. We illustrated the robustness of our method using SMILES and FASTA data. To ensure fair comparisons, we employ consistent models across embedding techniques. We did not deem it fair to compare to techniques that were pre-trained on larger datasets. As such, we compare our results with simpler benchmarks on Moleculenet to highlight the effectiveness of straightforward embeddings when paired with interpretable models. This emphasises the importance of accessibility and transparency in machine learning approaches, demonstrating that even simpler techniques can yield competitive and explainable results when deployed thoughtfully.

CRediT authorship contribution statement

Claudio Jardim: Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Alta de Waal:** Supervision, Conceptualization. **Inger Fabris-Rotelli:** Supervision. **Najmeh Nakhaei Rad:** Supervision. **Jocelyn Mazarura:** Supervision. **Dean Sherry:** Writing – review & editing, Writing – original draft, Validation, Investigation.

Declaration of competing interest

The authors declare that there are no known competing interests associated with the publication of this manuscript. We affirm that neither financial nor personal relationships with other people or organisations have influenced the work described in this paper. No competing interests exist, and this work is conducted in an unbiased and impartial manner.

Acknowledgements

The work of the fourth author was based upon research supported in part by the RDP grant at the University of Pretoria, and the National Research Foundation (NRF) of South Africa, Ref.: RA210106581084, grant No. 150170. The opinions expressed and conclusions arrived at are those of the authors and are not necessarily to be attributed to the NRF.

References

Anastasiu, D., Tagarelli, A., Karypis, G., 2013. Document Clustering: The Next Frontier. Chapman and Hall/CRC, pp. 305–338.
Arabi, A.A., 2021. Artificial intelligence in drug design: Algorithms, applications, challenges and ethics. Future Drug Discov. 3 (2), FDD59.
Behjati, A., Zare-Mirakabad, F., Arab, S.S., Nowzari-Dalini, A., 2022. Protein sequence profile prediction using ProtAlbert transformer. Comput. Biol. Chem. 99, 107717.
Bemis, G.W., Murcko, M.A., 1996. The properties of known drugs. 1. Molecular frameworks. J. Med. Chem. 39 (15), 2887–2893.
Bhatnagar, R., Sardar, S., Beheshti, M., Podichetty, J.T., 2022. How can natural language processing help model informed drug development?: A review. JAMIA Open 5 (2), ooac043.
Blei, D.M., Ng, A.Y., Jordan, M.I., 2003. Latent Dirichlet allocation. J. Mach. Learn. Res. 3 (Jan), 993–1022.
Boser, B.E., Guyon, I.M., Vapnik, V.N., 1992. A training algorithm for optimal margin classifiers. In: Proceedings of the Fifth Annual Workshop on Computational Learning Theory. pp. 144–152.
Chen, Y.-H., Li, S.-F., 2016. Using latent Dirichlet allocation to improve text classification performance of support vector machine. In: 2016 IEEE Congress on Evolutionary Computation. CEC, IEEE, pp. 1280–1286.
Crain, S.P., Zhou, K., Yang, S.-H., Zha, H., 2012. Dimensionality reduction and topic modeling: From latent semantic indexing to latent Dirichlet allocation and beyond. In: Mining Text Data. Springer, pp. 129–161.
De Waal, A., Barnard, E., 2008. Evaluating topic models with stability. In: Nineteenth Annual Symposium of the Pattern Recognition Association of South Africa. PRASA 2008, vol. 5221, pp. 79–84.
Deng, J., Yang, Z., Ojima, I., Samaras, D., Wang, F., 2022. Artificial intelligence in drug discovery: Applications and techniques. Brief. Bioinform. 23 (1), bbab430.
Ding, Y., Chen, M., Guo, C., Zhang, P., Wang, J., 2021. Molecular fingerprint-based machine learning assisted QSAR model development for prediction of ionic liquid properties. J. Mol. Liq. 326, 115212.
Elkan, C., 2005. Deriving TF-IDF as a Fisher kernel. In: International Symposium on String Processing and Information Retrieval. Springer, pp. 295–300.
Glem, R., Bender, A., Hasselgren, C., Carlsson, L., Boyer, S., Smith, J., 2006. Circular fingerprints: Flexible molecular descriptors with applications from physical chemistry to ADME. IDrugs: Investig. Drugs J. 9, 199–204.
Ibtehaz, N., Kihara, D., 2023. Application of sequence embedding in protein sequence-based predictions. In: Machine Learning in Bioinformatics of Protein Sequences: Algorithms, Databases and Resources for Modern Protein Bioinformatics. World Scientific, pp. 31–55.
Isert, C., Atz, K., Schneider, G., 2023. Structure-based drug design with geometric deep learning. Curr. Opin. Struct. Biol. 79, 102548.
Jaeger, S., Fulle, S., Turk, S., 2018. Mol2vec: Unsupervised machine learning approach with chemical intuition. J. Chem. Inform. Model. 58 (1), 27–35.
Jarada, T.N., Rokne, J.G., Alhajj, R., 2020. A review of computational drug repositioning: Strategies, approaches, opportunities, challenges, and directions. J. Cheminformatics 12 (1), 1–23.
Jurafsky, D., Martin, J.H., 2021. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Pearson Prentice Hall, pp. 102–133.
Kikuchi, I., Kikuchi, A., 2021. Latent Dirichlet allocation and objective functions to explore chemical space. IRCQE.
Kim, H., Lee, J., Ahn, S., Lee, J.R., 2021. A merged molecular representation learning for molecular properties prediction with a web-based service. Sci. Rep. 11 (1), 1–9.
Kondrak, G., 2005. N-gram similarity and distance. In: International Symposium on String Processing and Information Retrieval. Springer, pp. 115–126.
Landrum, G., Tosco, P., Kelley, B., Ric, Cosgrove, D., sriniker, gedeck, Vianello, R., Schneider, N., Kawashima, E., N, D., Jones, G., Dalke, A., Cole, B., Swain, M., Turk, S., Savelyev, A., Vaucher, A., Wójcikowski, M., Take, I., Probst, D., Ujihara, K., F, V., Scalfani, Godin, G., Lehtivarjo, J., Walker, R., Pahl, A., Berenger, F., Biggs, J., strets123, 2016. Rdkit: Open-source cheminformatics software. 2016. Rdkit 149 (150), 650.
Li, G., Yuan, Y., Zhang, R., 2023. Ensemble of local and global information for protein-ligand binding affinity prediction. Comput. Biol. Chem. 107972.
Liang, W., Feng, R., Liu, X., Li, Y., Zhang, X., 2018. GLTM: A global and local word embedding-based topic model for short texts. IEEE Access 6, 43612–43621.
Luhn, H.P., 1957. A statistical approach to mechanized encoding and searching of literary information. IBM J. Res. Dev. 1 (4), 309–317.
Mikolov, T., Chen, K., Corrado, G., Dean, J., 2013. Efficient estimation of word representations in vector space. In: Proceedings of the International Conference on Learning Representations. ICLR.
Oliveira, T.A.d., Silva, M.P.d., Maia, E.H.B., Silva, A.M.d., Taranto, A.G., 2023. Virtual screening algorithms in drug discovery: A review focused on machine and deep learning methods. Drugs Drug Candidates 2 (2), 311–334.
Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S., 2019. PyTorch: An imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems, vol. 32, Curran Associates, Inc., pp. 8024–8035.
Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. 12, 2825–2830.
Pritchard, J.K., Stephens, M., Donnelly, P., 2000. Inference of population structure using multilocus genotype data. Genetics 155 (2), 945–959.
Ramsundar, B., Eastman, P., Walters, P., Pande, V., Leswing, K., Wu, Z., 2019. Deep Learning for the Life Sciences. O'Reilly Media.
Rifaioglu, A.S., Atas, H., Martin, M.J., Cetin-Atalay, R., Atalay, V., Doğan, T., 2019. Recent applications of deep learning and machine intelligence on in silico drug discovery: Methods, tools and databases. Brief. Bioinform. 20 (5), 1878–1912.
Robertson, S., 2004. Understanding inverse document frequency: on theoretical arguments for IDF. J. Document. 60, 503–520.
Salton, G., Buckley, C., 1988. Term-weighting approaches in automatic text retrieval. Inf. Process. Manage. 24 (5), 513–523.
Sammut, C., Webb, G.I. (Eds.), 2017. Encyclopedia of Machine Learning and Data Mining. Springer US, Boston, MA, p. 8, 75, 497.
Sarac, O.S., Gürsoy-Yüzügüllü, Ö., Cetin-Atalay, R., Atalay, V., 2008. Subsequence-based feature map for protein function classification. Comput. Biol. Chem. 32 (2), 122–130.
Schneider, N., Fechner, N., Landrum, G.A., Stiefl, N., 2017. Chemical topic modeling: Exploring molecular data sets using a common text-mining approach. J. Chem. Inform. Model. 57 (8), 1816–1831.
Shahmirzadi, O., Lugowski, A., Younge, K., 2019. Text similarity in vector space models: A comparative study. In: 2019 18th IEEE International Conference on Machine Learning and Applications. ICMLA, IEEE, pp. 659–666.
Shivashankar, S., Srivathsan, S., Ravindran, B., Tendulkar, A.V., 2011. Multi-view methods for protein structure comparison using latent Dirichlet allocation. Bioinformatics 27 (13), i61–i68.
Singh, L., Chetty, G., Sharma, D., 2012. A novel approach to protein structure prediction using PCA or LDA based extreme learning machines. In: International Conference on Neural Information Processing. Springer, pp. 492–499.
van Der Hooft, J.J.J., Wandy, J., Barrett, M.P., Burgess, K.E., Rogers, S., 2016. Topic modeling for untargeted substructure exploration in metabolomics. Proc. Natl. Acad. Sci. 113 (48), 13738–13743.
Wallach, H.M., 2006. Topic modeling: Beyond bag-of-words. In: Proceedings of the 23rd International Conference on Machine Learning. pp. 977–984.
Weininger, D., 1988. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. J. Chem. Inform. Comput. Sci. 28 (1), 31–36.
Wu, C.H., Huang, H., Yeh, L.-S.L., Barker, W.C., 2003. Protein family classification and functional annotation. Comput. Biol. Chem. 27 (1), 37–47.
Wu, Z., Ramsundar, B., Feinberg, E.N., Gomes, J., Geniesse, C., Pappu, A.S., Leswing, K., Pande, V., 2018. MoleculeNet: A benchmark for molecular machine learning. Chem. Sci. 9 (2), 513–530.
Xiao, C., Zhang, P., Chaovalitwongse, W., Hu, J., Wang, F., 2017. Adverse drug reaction prediction with symbolic latent Dirichlet allocation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31, (no. 1).

Yalamanchili, H.B., Kho, S.J., Raymer, M.L., 2017. Latent Dirichlet allocation for classification using gene expression data. In: 2017 IEEE 17th International Conference on Bioinformatics and Bioengineering. BIBE, IEEE, pp. 39–44.

Yang, K.K., Wu, Z., Bedbrook, C.N., Arnold, F.H., 2018. Learned protein embeddings for machine learning. Bioinformatics 34 (15), 2642–2648.

Zhang, Y., Jin, R., Zhou, Z.-H., 2010. Understanding bag-of-words model: A statistical framework. Int. J. Mach. Learn. Cybern. 1 (1), 43–52.