Drug Discovery through Machine Learning and Deep Learning

Yuan Chang
Cornell Tech
New York, NY, USA
yc2378@cornell.edu

Bradley Wise
Cornell Tech
New York, NY, USA
bmw246@cornell.edu

Neel Madhukar
Cornell Tech
New York, NY, USA
neelmadhukar@gmail.com

Olivier Elemento
Weill Cornell School of Medicine
New York, NY, USA
ole2001@med.cornell.edu

Abstract

One of the biggest challenges in healthcare is the discovery of new drugs for untreated diseases. The current drug discovery process is costly, slow, and failure prone ($2.6B, 10 years, and 98% failure rate). One factor leading to this high cost is the effort needed to create new drugs and test them for various biological activities. This project aims to target a facet of the cost of creating and testing these drugs by predicting the optimal drugs for a given property prior to synthesizing any compounds. One big problem in neuroscience research is identifying drugs that will cross the blood-brain barrier for treatment. We built two models using Random Forest Regression and Convolutional Neural Net for predicting whether or not a drug can cross the blood brain barrier based on its chemical structures, features, and descriptors. Additionally, we generated new compounds from old ones that did not cross our blood-brain barrier classifier. Finally, we built a local interface to do single prediction on the chemical formula, or SMILES, of a specific compound.

Introduction

One of the biggest challenges in healthcare is the discovery of new drugs for untreated diseases. According to the Tufts Center for the Study of Drug Development, the current drug discovery process is costly, slow, and failure prone ($2.6B, 10 years, and 98% failure rate). One factor leading to this high cost is the effort needed to create new drugs and test them for various biological activities. For example, one big problem in neuroscience research is identifying which drugs will cross the blood-brain barrier. This is the basis of our project. We wanted to explore how we could train a machine learning model that correctly classifies whether a drug can cross the blood-brain barrier. Currently the best way to know whether a drug crosses the blood-brain barrier is to test the compound on non-human and human populations. The importance of using machine learning is to mitigate the cost of synthesizing drugs and testing with a robust and confident set of predictions.

We had three objectives in formulating this project. First, we wanted to research models and find features that were relevant to testing whether a drug can cross the blood-brain barrier. We chose to explore features and descriptors that were related solely to chemical structure of the drug itself rather than it's protein targets, side effects, usage, etc. Using machine learning and deep learning approaches, we created a model that predicts whether or not a drug crosses the blood-brain barrier. Many labs have tried used various classification techniques on drug information using molecular dynamic simulations [1] or other machine learning techniques in various capacities [2]. Some labs have also come up with deep learning architectures pre-trained on thousands of drugs that have the potential to be used for blood-brain classification [3] [4]. We used some of the ideas from these deep learning models to help inform our own model.

Second, after creating models to predict the probability of a drug crossing the blood-brain barrier, we wanted to research state-of-the-art of deep learning approaches that could "generate" new compounds. The goal of compound generation was to take drugs that did not pass our classifier and create structurally similar compounds that did cross the blood-brain barrier. We found and utilized a model that can "generate" new drugs and then adapted it to generate drugs that have the potential to cross the blood-brain barrier. By generating new drugs that pass classification, we hoped to get one step closer to reducing the cost of creating and testing new drugs prior to synthesizing them.

Finally, we wanted to host and open source our solution on Github. We wanted to create a user-friendly interface that would allow researchers/scientists to simply upload a molecule of interest and get the probability of whether or not the drug crosses the blood-brain barrier.

Related work

There have been many attempts to accurately predict whether or not a drug crosses the blood-brain barrier. For example, researchers out of Stanford have their own dataset for blood-brain barrier classification and have tried using various models for classification like logistic regression and other types of neural networks [2]. Other labs have tried creating prediction models using molecular dynamics simulations [1].

There have also been attempts at creating classification models for drugs using convolutional neural nets. Garret Goh and his lab created a model called Chemception [3] and ChemNet [4], where they pre-trained a deep learning architecture on thousands of images of the chemical structures of drugs. We wanted to use a different approach, combining both classifical machine learning techniques and transfer learning to get predictions on whether or not a drug crossed the blood-brain barrier.

There have also recently been attempts to discover or generate new drugs using various deep learning techniques. Some projects are generating new drug images using recurrent neural networks [5], [6], variational autencoders, or general adversarial networks [7]. Other projects are using techniques in natural language processing to generate new SMILES (simplified molecular-input line-entry system) data from drugs [8]. These various methods for drug generation were explored in our project as well.

Methods

Overview

Here is a brief overview of how we tackled the project. First, we found a open-source dataset with labels associated with whether or not a drug crossed the blood-brain barrier. We then extracted features from the dataset. After extracting features, we built two types of classification models for passage of the blood-brain barrier using a random forest classifier and convolutional neural net. Second, for drug generation, we explored state-of-the-art deep learning techniques, including variational autoencoder (VAE) and general adversarial network (GAN) on images of compounds. The goal was to create newly generated chemical structures that could be recognized and used by our classification algorithms. Finally, we built a web interface using Flask that would allow researchers to input a string-like representation of a molecule of their choosing and get a probability score of blood-brain barrier passage.

Data and Feature Engineering

We found blood-brain barrier data from MoleculeNet's BBBP dataset, a dataset generated by researchers from University of Lisbon, who used a Bayesian approach for classification [2]. Each molecule was associated with a label (0 or 1) and SMILES (Simplified molecular-input line-entry system). We then extracted chemical descriptors of each SMILES using a molecular descriptor calculator API called mordred [10] and RDKIT's molecular transformation of SMILES data [13]. With this, we were able to get 1613 chemical descriptors as features for 2039 drugs from the MoleculeNet dataset.

Classification Model

Our classification models used two distinct feature sets. Our random forest model used mordred's 1613 chemical descriptors. Our neural net model used image data that were extracted from RDKIT using the SMILES from the orginal dataset.

Random Forest Model

We used two techniques for feature engineering for the random forest model. One idea was to run a Kolmogorov Smirnov (KS) test on all 1613 descriptors, a test which is used to find the distance between two distributions (those that pass and those that do not pass). These differences were then ranked, and the top 50 features were used as features in the first iteration of the random forest model. We used 5-fold cross validation and balanced the dataset such that we trained on 500 compounds that did pass and 450 that did not pass. We had 25 trees in the forest, a max depth of 10 for each tree, 1 sample required to be at the leaf node, and the max number of features to consider when looking for the best split to be the square root of the total number of features. This acted a baseline model for our random forest regression.

After analysis of our original approach (see Current Outcomes section), we considered a different approach to feature selection. We first performed a grid search to identify the most important hyperparameters to use for our random forest. Given our dataset, and with all other parameters kept the same, the only changes to the baseline were as follows: 75 trees and a max depth of 20.

Based on our results from the baseline, we found that one of the most important features in our feature set for explaining blood-brain barrier passage was TopoPSA, molecular surface area (see outcomes for more information). So, for our final model, we took out all features correlated with TopoPSA above a threshold of 0.6 and used them as features for our random forest.

Convolutional Neural Net

We also wanted to test a model with transfer learning. After researching various deep learning API's (Keras, Tensorflow, and PyTorch), we decided to use fastai (V1), a library built on top of PyTorch [11]. The fastai library simplifies training fast and accurate neural nets using modern best practices, including various regularization methods like data augmentation, dropout, and use of pretrained models. It can also be easily implemented for training and single prediction.

Using fastai, we loaded a pretrained deep learning architecture called resnet34, a 34-layer architecture that is known to do a very good job at classifying images like dogs, cats, and objects. After getting these pretrained weights, we applied random rotation augmentation on images for additional training, set the batch size to 150, set a learning rate of 0.01, and trained over 5 epochs of the data using stratified 5-fold cross validation without early stopping on a cpu.

Drug Generation

For compound generation, we researched various models that had the possibility of generating new images or structures, including generative adversarial networks and variational autoencoders. After exploring options and some light testing, we ultimately chose to pursue a variational autoencoder called ChemVAE [9], a library that allows for generation of new SMILES strings.

GAN

The first approach we tried was to implement a generative adversarial network [12]. At a high level, a GAN has two models, a discriminator and a generator . The discriminator tries to discriminate between both real and fake images while the generator tries to "fool" the discriminator into thinking the generated images are real. This approach, while interesting, had a lot drawbacks. It took a long time to train, as the generator initially starts forming images from random noise. Computational requirements and a pre-test failure put us in search for new options.

Variational Autoencoder

Next, we began researching various autoencoder methods on images. This was also not a feasible approach [12] and made us think of different approaches. One such approach we tested was whether or not we could produce new strings. Much like projects in NLP, where new strings could be generated using some training document or corpus of words, we thought we could do the same with SMILES data. We found one such library called ChemVAE, which is used to encode molecular SMILES into a code vector representation (or latent space) based on a corpus of symbols that are one-hot encoded and are solely related to SMILES relevant symbols [9]. These vector representations are then decoded back to molecular SMILES that are slightly different from the original representation of the SMILES. The autoencoder may also be jointly trained with property prediction to help shape the latent space. The new latent space can then be optimized upon to find the molecules with the most optimized properties of interest.

Interface

A single drug prediction interface was built with front-end using HTML/Javascript and backend using Flask. This simple interface was built so that researchers or scientists can simply put in a string without having to implement the models themselves. The backend uses both saved models of the random forest from sklearn and the CNN (set of weights), as well as the ChemVAE library for generating new drugs.

Current Outcomes

Results of Classification Models

For baseline, using 49 Descriptors only, we got an accuracy of 75%, with a specificity of 67.3%, sensitivity of 82%, both precision/recall of 75%, and an ROC/AUC score of 89.6%. After model evaluation, we found that TopoPSA (topological surface area) was one of the primary predictors of whether or not a drug crossed the blood brain barrier. However, we did not believe it painted the whole picture. To validate this we used TopoPSA as a single feature on our previously cross-validated set. Using TopoPSA only, we got a classification accuracy of 67%, with an overal precision/recall of 67%, and an ROC/AUC score of 81.9%. Please refer to Figure 7 for more information.

Because of these results, we wanted to base our final model on features that were not related to TopoPSA. We therefore performed a grid search using parameters described in our methods section, and we chose the best correlation cutoff based on these results. Please refer to Figure 8, which maps both the accuracies and number of features selected as a function of the correlation cutoff applied to TopoPSA. Our best and final model used a correlation cutoff of 0.6 on TopoPSA, and we got an accuracy of 77%, with a specificity of 69.3%, sensitivity of 84%, both precision/recall of 77%, and an ROC/AUC score of 90.3%. Please refer to Figure 9 for more information.

For our CNN, we used the same metrics as above for validating our model. We were happy to find that by using the fastai library with minimal training and applying transfer learning, we were able to get an accuracy of 70%, with an overall precision/recall of 71%, and an ROC/AUC of 79.9%. We found in the minimally trained model, the CNN generally learned that molecules with higher polar surface or size did not pass the blood brain barrier, while molecules that were smaller generally passed the barrier. To see more details of the way the CNN classified images, please refer to Figures 10, 11, 12, and 13.

We then saved both models and used them for single classification prediction on new SMILES.

Results of Drug Generation Model

We used the auto-encoder model to generate new drugs based on the given SMILES. The model encodes the input into latent space and decodes it back to the original molecule within a certain standard deviation of that latent space. While ChemVAE offered a metric for measuring similarity to the original drug according to some of their own default properties, we used Tanimoto Similarity from RDKIT to compare similarities between the original compound and the generated compound.

Additionally, we wanted to make sure the generated structures were real and feasible structures so we generated 500 hundred new structures with ChemVAE from the BBBP dataset as input and compared these generated structures with 2,000 existing compounds in the DrugBank. We found 25 structures that had more than 99% similarity with at lease one structure from DrugBank dataset. For this reason, we believed the drug generation model was valid and used it for our final interface.

Interface

To use the web interface, users need to give a drug structure (SMILES) as the input which should ideally be a molecule string as shown in figure 1. If the input received is a valid molecule structure, a single prediction model will start computing the probability of whether or not it crosses the blood-brain barrier.

The output will ensemble results from the random forest and the CNN. A spinning icon will show up to indicate the computing is still running as shown in figure 2. If the input passes the blood-brain barrier, the interface will simply tell the user we predict the drug to pass along with its associated probability (Figure 3). If the drug is not likely to pass, which means our model predicts the possibility is lower than 0.5, the auto-encoder generation model will start computing new drug structures based on the given structure.

Figure 4 shows recommended structures that will pass blood-brain barrier based on the given structure.

To test the server and follow the code of the single prediction model, please go to https://github.com/bmwise14/drug-discovery-localweb. Our saved models are located here as well.

Future Work

1. *Improving the Random Forest Model and CNN - Ensemble Approaches*

   The regression model we finally chose is not entirely satisfactory. While we used a model that did not include surface area, it is generally unwise to use so many features (approximately 1200) in the final model, as it may increase the likelihood of overfitting the model. In addition, we are currently checking to see if the random forest is classifying better with certain types of features. This will help us nail down which features of the 1200 can help generalize the model more.

   One of the drawbacks of training a deep learning model on a CPU is the length of time it takes to train such a model. The most surprising result was that the training and validation losses continued to go down during training. Given CPU constraints, it is ideal to train a model with a GPU. With a GPU, additional next steps for this model would be to unfreeze the resnet34 layers and fine-tune their weights, as these weights are not classically used for molecular structures.

   Finally, one component of the project we were not able to finish was ensembling the models into one larger model by using the outputs of the predictions of both. Our single predictor in Flask predicts from both models instead of one full model that combines the outputs of both. Additionally, we are also interested in finding what specific drugs are doing better with the random forest and which molecules are being classified better with the CNN.

2. *Refine the Auto-Encoder Model*

   One issue with the autoencoder is that a compound that doesn't pass will likely generate similar drugs that still do not pass. How to keep the balance between maintaining a threshold similarity and generating a new compound that could pass the blood-brain barrier will be meaningful. It would be nice to know what changes to a molecule contribute to it's passing or not passing.

3. *Improvement of the Interface*

   Currently the metadata showed by the interface is just text information. We would like to add more graphic metadata in the web page such as images of the molecule itself or a drug network of generated compounds.

4. *Deployment on a server*

   Currently, we have a local version our application running. We would like to deploy it on a cloud server such as AWS so that we do not need to run a virtual environment and run the flask app locally.

Acknowledgements

We would like to thank Olivier Elemento and Neel Madhukar for helping with us our project. Additionally, we would like to thank Coryandar Gilvary for her insights on how to improve our random forest classifier and helping work through the updated fastai library.

References

[1] Timothy S. Carpenter, Daniel A. Kirshner, Edmond Y. Lau, Sergio E. Wong, Jerome P. Nilmeier, Felice C. Lightstone A Method to Predict Blood-Brain Barrier Permeability of Drug-Like Compounds Using Molecular Dynamics Simulations. In *Biophysical Journal*. DOI: 10.2174/1573406413666170209124302

[2] Zhenqin Wu, Bharath Ramsundar, Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, Vijay Pande. MoleculeNet: A Benchmark for Molecular Machine Learning. In *Chemical Science*. DOI: 10.1039/C7SC02664A

[3] Garrett B. Goh, Charles Siegel, Abhinav Vishnu, Nathan O. Hodas, Nathan Baker. Chemception: A Deep Neural Network with Minimal Chemistry Knowledge Matches the Performance of Expert-developed QSAR/QSPR Models. In *Biophysical Journal* DOI: arXiv:1706.06689

[4] Garrett B. Goh, Charles Siegel, Abhinav Vishnu, Nathan O. Hodas. Using Rule-Based Labels for Weak Supervised Learning: A ChemNet for Transferable Chemical Property Prediction. In *Proceedings of ACM SIGKDD Conference, London, UK,* ***Aug 2018*** *(KDD 2018), 9 pages.* DOI: 10.475/1234

[5] Mariya Popova, Olexandr Isayev, Alexander Tropsha1. Deep Reinforcement Learning for de-novo Drug Design. In *SCIENCE ADVANCES 25* ***Jul 2018****: Vol. 4, no. 7, eaap7885* DOI: 10.1126/sciadv.aap7885

[6] Daniel Neil, Marwin Segler, Laura Guasch, Mohamed Ahmed, Dean Plumbley, Matthew Sellwood, Nathan Brown. Exploring Deep Recurrent Models with Reinforcement Learning for Molecule Design. DOI: https://openreview.net/forum?id=HkcTe-bR-

[7] Artur Kadurin, Sergey Nikolenko, Kuzma Khrabrov, Alex Aliper, Alex Zhavoronkov. druGAN: An Advanced Generative Adversarial Autoencoder Model for de Novo Generation of New Molecules with Desired Molecular Properties in Silico. In *Mol. Pharmaceutics* ***2017****, 14, 3098âĹŠ3104* DOI: 10.1021/acs.molpharmaceut.7b00346

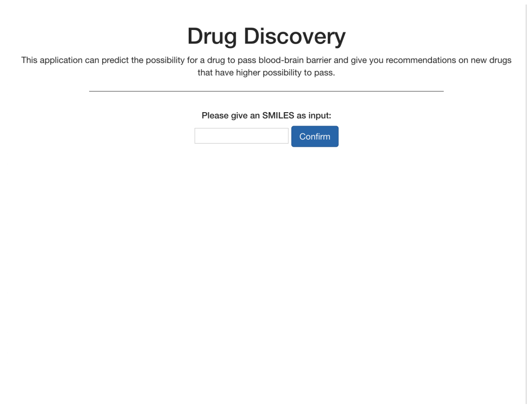
[8] Naruki Yoshikawa, Kei Terayama, Teruki Honma, Kenta Oono, Koji Tsudaa. Population-based de novo molecule generation, using grammatical evolution. In *Chemistry Letters* ***2018****, 47(11), 1431-1434*. DOI: arXiv:1804.02134v1

[9] Rafael Gomez-Bombarelli , Jennifer N. Wei, David Duvenaud, Jose Miguel Hernandez-Lobato, Benjamin Sanchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D. Hirzel, Ryan P. Adams, and Alan Aspuru-Guzik. Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. In *ACS Cent. Sci.* ***2018****, 4 (2), pp 268-276*. DOI: 10.1021/acscentsci.7b00572

[10] Hirotomo Moriwaki, Yu-Shi Tian, Norihito Kawashita, Tatsuya Takagi. Mordred: a molecular descriptor calculator. In *Journal of Cheminformatics **2018** 10:4*. DOI: https://doi.org/10.1186/s13321-018-0258-y
[11] Jeremy Howard. fastai library. https://docs.fast.ai/index.html.
[12] Erik Linder-Noren. Open Source Code for GANs and Autoencoder. https://github.com/eriklindernoren/Keras-GAN.
[13] Greg Landrum. RDKIT Library. http://rdkit.org/docs/api-docs.html.

Appendix

1. Figure 1: Web interface screenshot1

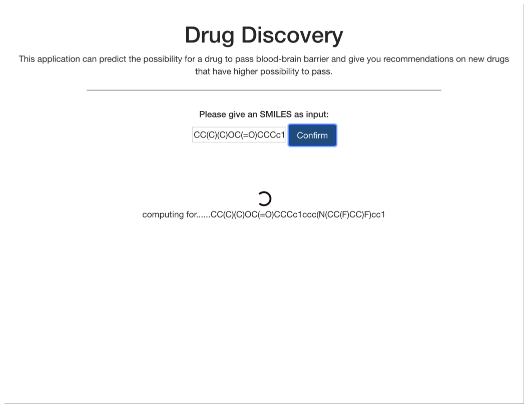


Drug Discovery

This application can predict the possibility for a drug to pass blood-brain barrier and give you recommendations on new drugs that have higher possibility to pass.

Please give an SMILES as input:

2. Figure 2: Web interface screenshot2



Drug Discovery

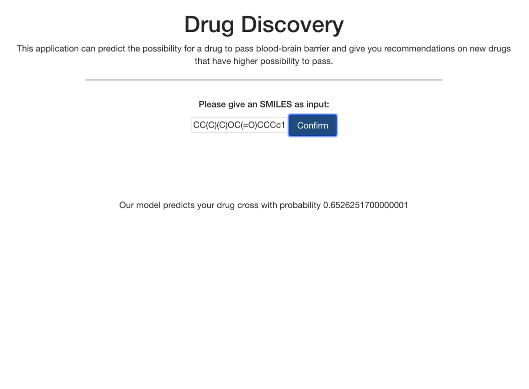
This application can predict the possibility for a drug to pass blood-brain barrier and give you recommendations on new drugs that have higher possibility to pass.

Please give an SMILES as input:

CC(C)(C)OC(=O)CCCc1

computing for......CC(C)(C)OC(=O)CCCc1ccc(N(CC(F)CC)F)cc1

3. Figure 3: Web interface screenshot3



Drug Discovery

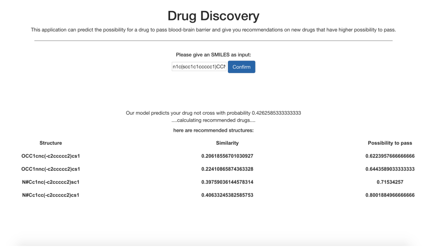
This application can predict the possibility for a drug to pass blood-brain barrier and give you recommendations on new drugs that have higher possibility to pass.

Please give an SMILES as input:

CC(C)(C)OC(=O)CCCc1

Our model predicts your drug cross with probability 0.6526251700000001

4. Figure 4: Web interface screenshot4



Drug Discovery

This application can predict the possibility for a drug to pass blood-brain barrier and give you recommendations on new drugs that have higher possibility to pass.

Please give an SMILES as input:

n1c(scc1c1ccccc1)CCN

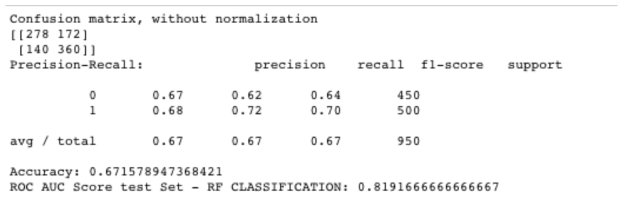
Our model predicts your drug not cross with probability 0.4262585333333333
....calculating recommended drugs....
here are recommended structures:

Structure	Similarity	Possibility to pass
OCC1cnc(-c2ccccc2)cs1	0.20618556701030927	0.6223957666666666
OCC1nnc(-c2ccccc2)cs1	0.22410865874363328	0.6443589033333333
N#Cc1nc(-c2ccccc2)sc1	0.39759036144578314	0.71534257
N#Cc1cc(-c2ccccc2)cs1	0.40633245382585753	0.8001884966666666

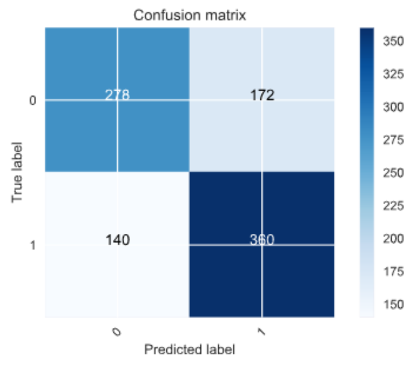
5. Video Demo for drugs predicted to pass
   https://drive.google.com/file/d/1GmAfAbMG6WCpAA_UjWjEsz8wko6_xENd/view?usp=sharing

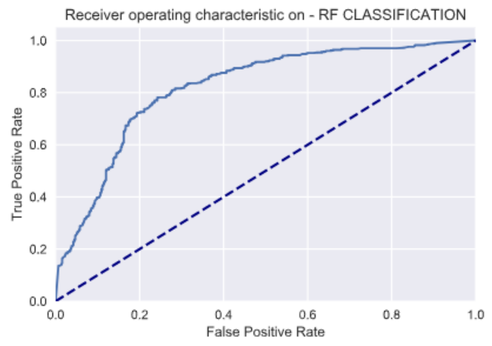
6. Video Demo for drugs predicted to not pass
   https://drive.google.com/file/d/1CJpUoCSfr4lNbsf2300rikhwd9Wvn4_m/view?usp=sharing

7. Figure 7: ROC-AUC Curve and Confusion Matrix for Random Forest Classifier TopoPSA only

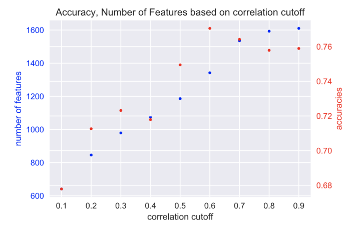


```
Confusion matrix, without normalization
[[278 172]
 [140 360]]
Precision-Recall:              precision    recall  f1-score   support

          0       0.67      0.62      0.64       450
          1       0.68      0.72      0.70       500

avg / total       0.67      0.67      0.67       950

Accuracy: 0.671578947368421
ROC AUC Score test Set - RF CLASSIFICATION: 0.8191666666666667
```



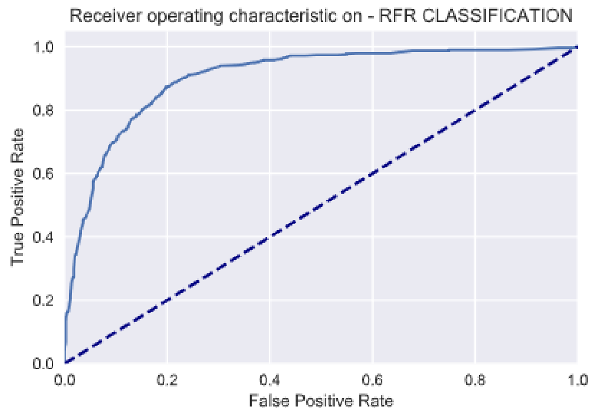


8. Figure 8: Visual of accuracies and number of features selected as a function of correlation cutoff to TopoPSA

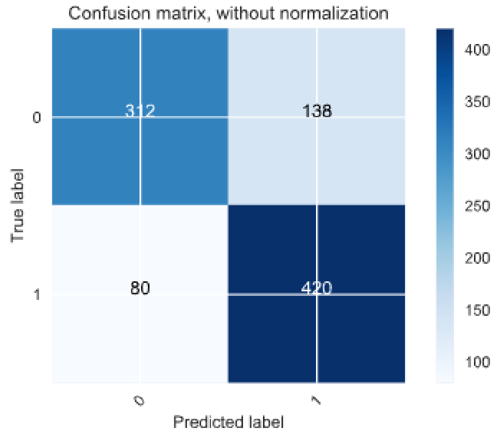


9. Figure 9: ROC-AUC Curve and Confusion Matrix for Final Random Forest Classifier

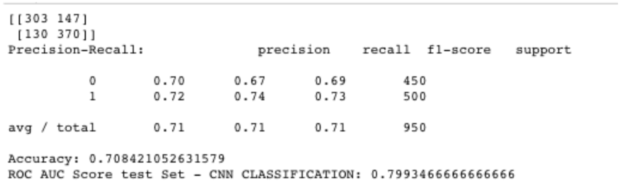
```
accuracies: 0.7705263157894737
specificity: 0.6933333333333334
sensitivity: 0.84
precision: 0.7526881720430108
ROC AUC Score test Set - RFR CLASSIFICATION: 0.9039355555555555
```



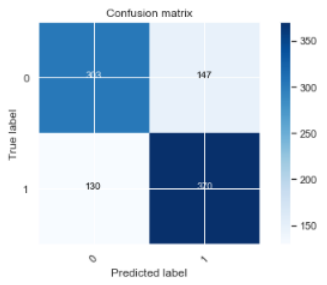
```
Confusion matrix, without normalization
[[312 138]
 [ 80 420]]
```

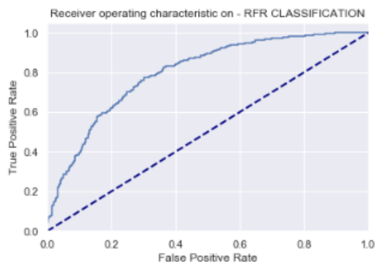


10. Figure 10: ROC-AUC Curve and Confusion Matrix for CNN

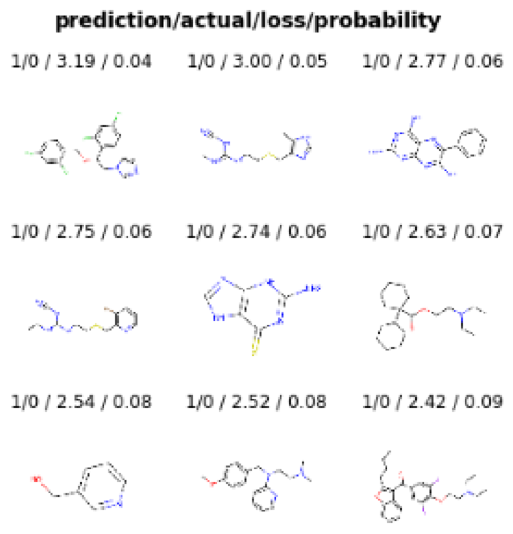


```
[[303 147]
 [130 370]]
Precision-Recall:             precision    recall  f1-score   support

          0       0.70      0.67      0.69       450
          1       0.72      0.74      0.73       500

avg / total       0.71      0.71      0.71       950

Accuracy: 0.708421052631579
ROC AUC Score test Set - CNN CLASSIFICATION: 0.7993466666666666
```

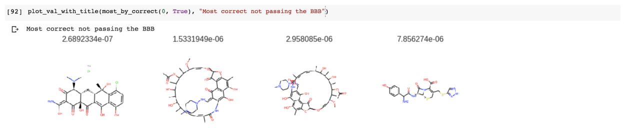




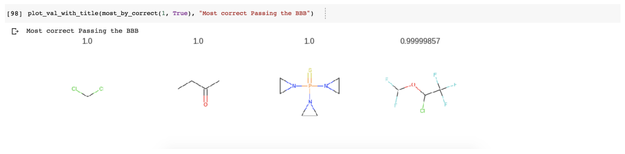
11. Figure 11: Images That Were Classified Most Incorrectly



12. Figure 12: Images That were Classified Most Correctly



```
[92] plot_val_with_title(most_by_correct(0, True), "Most correct not passing the BBB")

Most correct not passing the BBB
     2.6892334e-07     1.5331949e-06     2.958085e-06     7.856274e-06
```



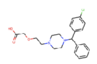
```
[98] plot_val_with_title(most_by_correct(1, True), "Most correct Passing the BBB")

Most correct Passing the BBB
     1.0     1.0     1.0     0.99999857
```

13. Figure 13: Images Where Classifier Most Uncertain

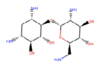
```
[101] most_uncertain = np.argsort(np.abs(probs -0.5))[:4]
      plot_val_with_title(most_uncertain, "Most uncertain predictions")
```

```
Most uncertain predictions
```

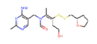
0.48866215



0.5201168



0.47306815



0.4698914