# Screening Workflows And Nanomaterials Documentation

***Release 0.3.1***

**Felipe Zapata**

**Aug 21, 2020**

# CONTENTS:

CHAPTER

ONE

# SCREENING WORKFLOWS AND NANOMATERIALS

**Swan** is a Python pacakge to create statistical models to predict molecular properties. See Documentation.

## 1.1 Installation

- Download miniconda for python3: miniconda (also you can install the complete anaconda version).
- Install according to: installConda.
- Create a new virtual environment using the following commands:
  - `conda create -n swan`
- Activate the new virtual environment
  - `source activate swan`

To exit the virtual environment type `source deactivate`.

### 1.1.1 Dependencies installation

- Type in your terminal:

  `conda activate swan`

Using the conda environment the following packages should be installed:

- install RDKit and H5PY:
  - *conda install -y -q -c conda-forge h5py rdkit*
- install Pytorch according to this recipe

### 1.1.2 Package installation

Finally install the package:

- Install **swan** using pip: - `pip install git+https://github.com/nlesc-nano/swan@master`

Now you are ready to use *swan*.

**Notes:**

- Once the libraries and the virtual environment are installed, you only need to type `conda activate swan` each time that you want to use the software.

CHAPTER

TWO

# TUTORIAL

In this tutorial we explore how to create and train statistical models to predict molecular properties using the Pytorch library. We will use smiles to represent the molecules and use the csv file format to manipulate the molecules and their properties.

As an example, we will predict the *activity coefficient_* for a subset of carboxylic acids taken from the *GDB-13 database_*. Firstly, We randomly takes a 1000 smiles from the database and compute the *activity coefficient_* using the *COSMO approach_*. We store the values in the *thousand.csv_* file.

A peek into the file will show you something like:

```
,smiles,E_solv,gammas
808780,OC(=O)C1OC(C#C)C2NC1C=C2,-11.05439751550119,8.816417146193844
593047,OC(=O)C1C2NC3C(=O)C2CC13O,-8.98188869016993,52.806217658944995
21701,OC(=O)C=C(C#C)C1NC1C1CN1,-11.386853547889574,6.413128231164093
768877,OC(=O)C1=CCCCC2CC2C#C1,-10.578966144649726,1.426566948888662
```

Where the first column contains the index of the row, the second the solvation energy and finally the *activity coefficients_* denoted as *gammas*. Once we have the data we can start exploring different statistical methods.

*swan* offers a thin interface to Pytorch. It takes yaml file as input and either train an statistical model or generates a prediction using a previously trained model. Let's briefly explore the *swan* input.

## 2.1 Simulation input

A typical *swan* input file looks like:

```
dataset_file:
  tests/test_files/thousand.csv
property: gammas

use_cuda: True

featurizer:
  fingerprint: atompair

model:
  input_cells: 2048
  hidden_cells: 1000

torch_config:
  epochs: 100
  batch_size: 100
```

(continues on next page)

(continued from previous page)

```
  optimizer:
    name: sgd
    lr: 0.2
```

**dataset_file**: Could be either a csv file with the smiles and other molecular properties or a *joblib* file that is binary format to load a previous used dataset (see the *save_dataset* keyword).

**property**: the columns names of hte csv file representing the molecular properties to fit.

**featurizer**: The type of transformation to apply to the smiles to generates the features. Could be either **fingerprint** or **molecular_graph**.

## 2.2 Training a model

In order to run the training, run the following command:

```
modeller --mode train -i input.yml
```

*swan* will generate a log file called *output.log* with a timestamp for the different steps during the training. Finally, you can see in your *cwd* a folder called *swan_models* containing the parameters of your statistical model.

## 2.3 Predicting new data

To predict new data you need to provide some smiles for which you want to compute the properties of interest, in this case the *activity coefficient_*. For doing so, you need to provide in the *dataset_file* entry of the *input.yml* file the path to a csv file containing the smiles, like the *smiles.csv_*:

```
,smiles
0,OC(=O)C1CNC2C3C4CC2C1N34
1,OC(=O)C1CNC2COC1(C2)C#C
2,OC(=O)CN1CC(=C)C(C=C)C1=N
```

Then run the command:

```
modeler --mode predict -i input.yml
```

*swan* will look for a *swan_model.pt* file with the previously trained model and will load it.

Finally, you will find a file called "predicted.csv" with the predicted values for the activity coefficients.

CHAPTER

# THREE

# API

Class to train and predict statistical models.

Deep Feedforward Network

Molecular Graph Convolutional Network

CHAPTER

FOUR

# INDICES AND TABLES

- genindex
- modindex
- search