in_silico

October 20, 2024

— Author: James Aaron Erang Affiliation: Biotechnology and Analytical Laboratory, Department of Biological Sciences, College of Science —

[1]:
```
from os import path
from os import mkdir
from itertools import zip_longest

import pandas as pd
import matplotlib.pyplot as plt
from PIL import Image
from chembl_webresource_client.new_client import new_client
from rdkit.Chem import Draw
from rdkit import Chem
```

[2]:
```
type Mol = Chem.Mol
type MHash = dict[str, dict[str, str]]
type Df = pd.DataFrame
```

[3]:
```
# where the mol figs will be stored
if not path.exists(fig_path := "./chem_figs"):
    mkdir(fig_path)
```

[4]:
```
# initiate the client to fetch
# molecule information
mol_target = new_client.molecule
```

[5]:
```
chembl_data: Df = pd.DataFrame.from_dict(
        mol_target.search("magainin")
    )
```

[ ]:
```
chembl_data
```

[7]:
```
# hashmap for chembl_id: canonical smiles
chembl_mol: MHash = {}

for mol_name, id_, mol_type, smiles in zip(
        chembl_data["pref_name"],
        chembl_data["molecule_chembl_id"],
        chembl_data["molecule_type"],
        chembl_data["molecule_structures"],
    ):
    # save some data in a hashmap
    chembl_mol[id_] = {
            "mol_name": mol_name,
            "mol_id": id_,
            "mol_type": mol_type,
            "smiles": smiles["canonical_smiles"]
        }
```

[ ]:
```
chembl_mol_pd: Df = pd.DataFrame(
        chembl_mol
    )
chembl_mol_pd
```

[9]:
```
# draw the molecular structure using the
# canonical smiles fetched and rdkit
for mol_key in chembl_mol.keys():
    mol = Chem.MolFromSmiles(
            chembl_mol[mol_key]["smiles"]
        )
    mol_drawn = Draw.MolDraw2DCairo(400, 400)
    mol_drawn.DrawMolecule(mol)
    mol_drawn.FinishDrawing()
    png_data = mol_drawn.GetDrawingText()

    with open(
            f"./{fig_path}/{mol_key}.png", "wb"
        ) as mol_png:
        mol_png.write(png_data)
```

[10]:
```
ulimit: int = len(
        chembl_data["molecule_chembl_id"].keys()
    )

while ulimit%3 != 0:
    ulimit += 1
```

[11]:
```
COLS: int = 3
ROWS: int = round(ulimit/3)
MOL_COUNT: int = len(chembl_mol.keys())
```

[12]:
```
axes: list = []
```

[13]:
```
plt.rcParams["font.family"] = "serif"
plt.rcParams["font.serif"] = ["MLMRoman12"]
```

[ ]:
```
fig = plt.figure(figsize=(8,8))
for i, id_ in zip_longest(
        range(ROWS*COLS),
        chembl_mol.keys()
    ):

    axes.append(
        fig.add_subplot(
            ROWS,
            COLS,
            i+1
        )
    )

    mol_data: dict[str, str] = chembl_mol[id_]
    mname: str = mol_data["mol_name"]
    mtype: str = mol_data["mol_type"]
    mid: str = mol_data["mol_id"]

    if not mname:
        mname: str = "---"

    img_path = f"{id_}.png"
    axes[-1].set_title(
        f"{mid}\n{mname}\n{mtype}"
    )
    plt.axis("off")
    plt.imshow(
        Image.open(
            f"./{fig_path}/{img_path}"
        )
    )

fig.tight_layout()
plt.savefig(
    f"{fig_path}/molecules.png",
    bbox_inches="tight",

)
plt.show()
```

[ ]:
```
fig = plt.figure(figsize=(8,8))
for i, id_ in zip_longest(
        range(ROWS*COLS),
        chembl_mol.keys()
    ):
```

```
    axes.append(
        fig.add_subplot(
            ROWS,
            COLS,
            i+1
        )
    )

    img_path = f"{id_}.png"
    plt.axis("off")
    plt.imshow(
        Image.open(
            f"./{fig_path}/{img_path}"
        )
    )

fig.tight_layout()
plt.savefig(
    f"{fig_path}/molecules_strip.png",
    bbox_inches="tight",

)
plt.show()
```

[ ]: