Code Documentation

Datasets

Run `pip install -r requirements.txt` to install all the necessary packages required to run `prepareDataset.py` and `prepareDataset2.py`

prepareDataset.py

This script processes chemical data files containing SMILES strings. It standardizes the SMILES strings, converts them to SELFIES strings, and saves the processed data to new files. The script supports various file formats and processes each file in a specified directory.

How to run:

```
python script_name.py --folder_path <path_to_input_folder>
--smiles_column_name <column_name> --output_directory
<path_to_output_directory>
```

prepareDataset2.py

This script processes chemical data files containing SMILES strings and converts them to various chemical representations: DeepSMILES, SELFIES, SAFE, and Group SELFIES. The script can load datasets from Hugging Face or local directories, process the SMILES strings, and save the processed data.

How to run:

```
python script_name.py --dataset <dataset_name>
```

Tokenizers

Run `pip install -r requirements.txt` to install all the necessary packages required to run train_tokenizer.py

train_tokenizer.py

This Python script provides functionality to train various types of tokenizers for SMILES strings, including atom-wise, k-mer, BPE, and SmilesPE tokenizers. It uses the Hugging Face datasets library for loading SMILES datasets and the tokenizers library for tokenizer implementation.

How to run:

```
python smiles_tokenizer.py -dataset_name <dataset_name>
-mol_type <molecule_type> -tokenizer_method
<tokenizer_method> -max_vocab_size <max_vocab_size>
```

Model

Run `pip install -r requirements.txt` to install all the necessary packages required to run the model training and generation scripts

Training

The training can be carried out by running the bash scripts `train_moses.sh` and `train_guacamol.sh` to train using the moses and guacamol datasets respectively. The parameters of the conditioned training can be altered by modifying the bash scripts.

How to run:

```
Run bash ./train_moses.sh
Run bash ./train_guacamol.sh
```

Generate

The generation of new molecules using the trained models can be carried out using the bash scripts `generate_guacamol_prop.sh` and `generate_moses_prop_scaf.sh`. The parameters of the generation can be controlled by modifying these scripts.

How to run:

```
./generate_guacamol_prop.sh
./generate_moses_prop_scaf.sh
```

Fine-Tuning

Run `pip install -r requirements.txt` to install all the necessary packages required to run the files

ReSTforFineTuning.py

This script implements the Reinforced Self-Training (ReST) framework for fine-tuning large language models (LLMs). The ReST framework leverages principles of reinforcement learning (RL) to optimize LLMs, particularly in domains requiring high-quality outputs such as molecular generation. The script integrates dataset generation with policy improvement to align models with specific desired outcomes.

How to run:

```
python ReSTforFineTuning.py
```

visualize_logs.py

This script plots training metrics from a CSV file and saves the plots as an image. It uses pandas for data manipulation, matplotlib and seaborn for plotting, and argparse for parsing command-line arguments.

How to run:

```
python visualize_logs.py --csv_file <path_to_csv_file>
--output_file <path_to_output_image> --metrics <metric1>
<metric2> <metric3> ...
```

visualize_mol.py

This script generates images of molecules from SMILES strings found in a CSV file. It uses pandas for data manipulation and rdkit for molecule generation and visualization. The images are saved in a specified directory.

How to run:

```
python visualize_mol.py --csv_file <path_to_csv_file>
--smiles_column <smiles_column_name> --output_dir
<path_to_output_directory> --num_samples <number_of_samples>
```

plot_training_metrics.py

This script plots training metrics from a CSV file and saves the plots as images. It uses pandas for data manipulation and matplotlib and seaborn for plotting.

The script allows customization of the output directory and font size for the plots.

How to run:

```
python plot_training_metrics.py --csv_file <path_to_csv_file>
--output_dir <path_to_output_directory> --font_size
<font_size>
```

Evaluation

MolEvalMetric

This script calculates various metrics to evaluate the performance of molecular generation models. The metrics help in understanding how well a model produces novel, chemically valid molecules that are relevant to specific research objectives. The script uses several libraries, including rdkit for molecular operations, fcd_torch for Fréchet ChemNet Distance calculation, and tdc for Oracle-based evaluations.
Run `pip install -r requirements.txt` to install all the necessary packages required to run

How to run:

- Load the MolGen/MolEvalMetrics from Hugging Face using:
  `MolEvalMetrics = evaluate.load("MolGen/MolEvalMetrics")`
- Get the evaluation metrics for the generated set of SMILES by calling the compute method:
  `print(MolEvalMetrics.compute(gensmi = ls_gen, trainsmi = ls_train))`

cpp_chem

This C++ library, exposed to Python using Pybind11, provides functions for evaluating molecular properties and diversity based on SMILES strings. The library uses RDKit for molecular operations and is designed to be accessible from Python.

How to run:

```
https://www.matecdev.com/posts/cpp-call-from-python.html
```

rust_chem

This Rust library provides functions for handling molecular data, specifically for converting SMILES strings to molecule objects and retrieving their canonical SMILES representation. The library uses the chemcore crate for molecular operations and is designed to be accessible from Python using the PyO3 library.

How to run:

```
https://pyo3.rs/v0.22.0/
```