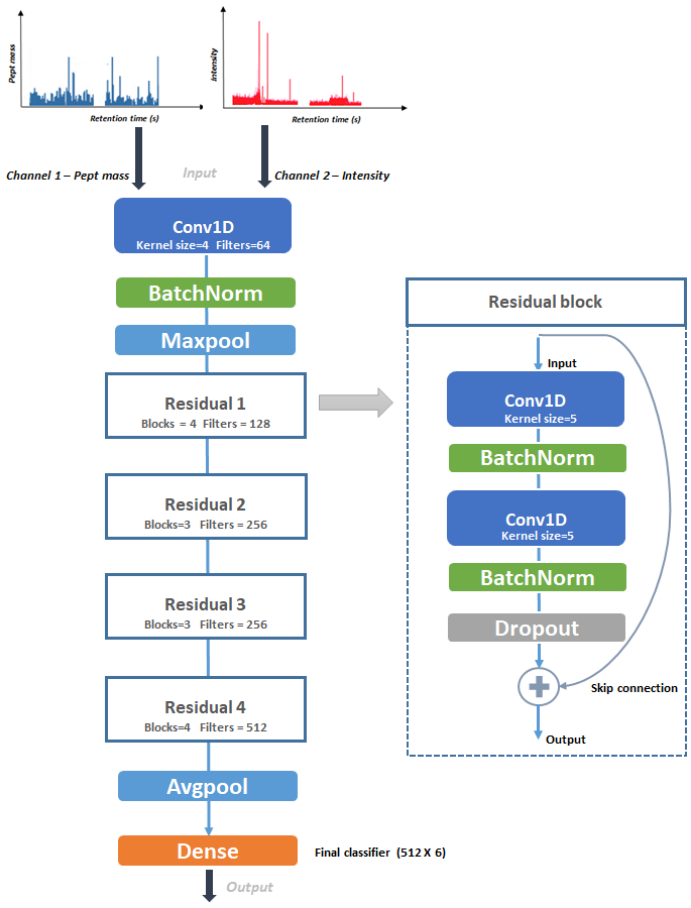
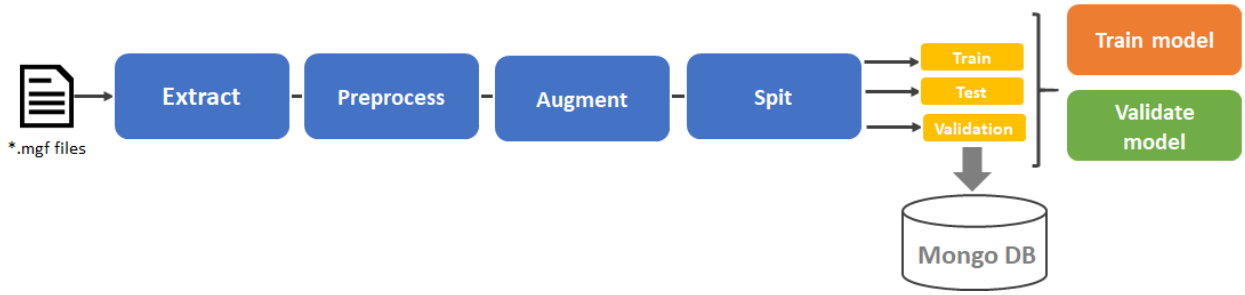
IMPLENTAION OF RESIDUAL CNN MODEL FOR DETERMINATION OF COMORBIITY BETWEEN CANCER AND SCHIZOPHRENIA IN PYTORCH

MODEL



MAIN PIPELINE



USAGE

This pipeline can only be used with the Mongo DB installed. Follow the instructions to install Mongo DB:

https://docs.mongodb.com/manual/installation/

The_config.json_file is used to configure the data loading, training and validation processes.

1. **Configure db connection**

Mongo DB is used to load and store data extracted from mgf files. To set up the database configuration, set the value *"db_type"*: "mongo" in the **config.json** file and fill in the connection parameters block:

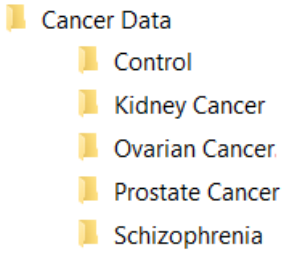
```
"db_type": "mongo"
"mongo_params": {
   "host": "localhost",
   "port": 27017,
   "db_name": "cancer_data",
   "col_name": "data_col"
  },
```

2. Initial loading

For the initial loading of data into, place the mgf files in folders corresponding to the name of the class (pathology) and specify the path to the root folder in the parameters

```
"load from folder": {
  "data folder": <Path to data>,
  "validation": 20
}
```

Example:



Specify the number of files for each class that will be allocated for model validation.

Specify the label of data for each class:

```
"labels": {
   "0": "Control",
   "1": "Ovarian Cancer",
   "2": "Prostate Cancer",
   "3": "Kidney Cancer",
   "4": "Schizophrenia"
  },
```

Define parameters for data preprocessing:

```
"preprocessing": {
   "elliptic": true,
   "contamination": 0.2,
   "min-max scale": true,
```

"rti_align": true,

"align_bound": 5000,

"align_step": 0.1,

"log": true,

"save_models": true,

"mm_model_path": *<path to save minmax scaler model>*,

"outlier_model_path": *<path to save outlier model>*

},

Parameter	Description
elliptic	If set to True the "Elliptic outlier detector" will be used
contamination	Contamination factor for the "Elliptic outlier detector"
min-max scale	If set to True the "Minmax scaler" will be used
rti_align	If set to True the MS signal will be aligned to the RTI axis
align_bound	The maximum value of the signal length along the RTI axis (total length / align_step)
align_step	MS signal alignment step for resampling along the RTI axis
log	If set to True, the intensity value in the MS signal will be logarithm
• save_models • mm_model_path • outlier_model_path	Optional If you want to save the models used for preprocessing, set the save_model parameter to True and specify the path to save the models.

Determine the values for the pipeline_steps parameter:

"pipeline steps": ["load from folder", "clean-create", "scale-create", "align", "upload to db"]

Parameter	Description
load from folder	Indicates that the download will be made from the folder specified in the *data_folder* parameter
clean-create	Indicates that the Elliptic outlier detector model will be created and applied.
scale-create	Indicates that the Elliptic outlier detector model will be created and applied.
upload to db	Indicates that data will be loaded into Mongo DB.

Start loading and transforming data

Execute the code contained in script core_app.py

```
$python core_app.py
```

The results of the data loading and transformation process will be displayed in the console

You can also check the upload results directly in Mongo database.

3. Train model

To start the model training mode, set the value of the *pipeline_steps* parameter

"pipeline steps": ["load from db", "train"]

Execute the code contained in script core_app.py

```
$python core_app.py
```

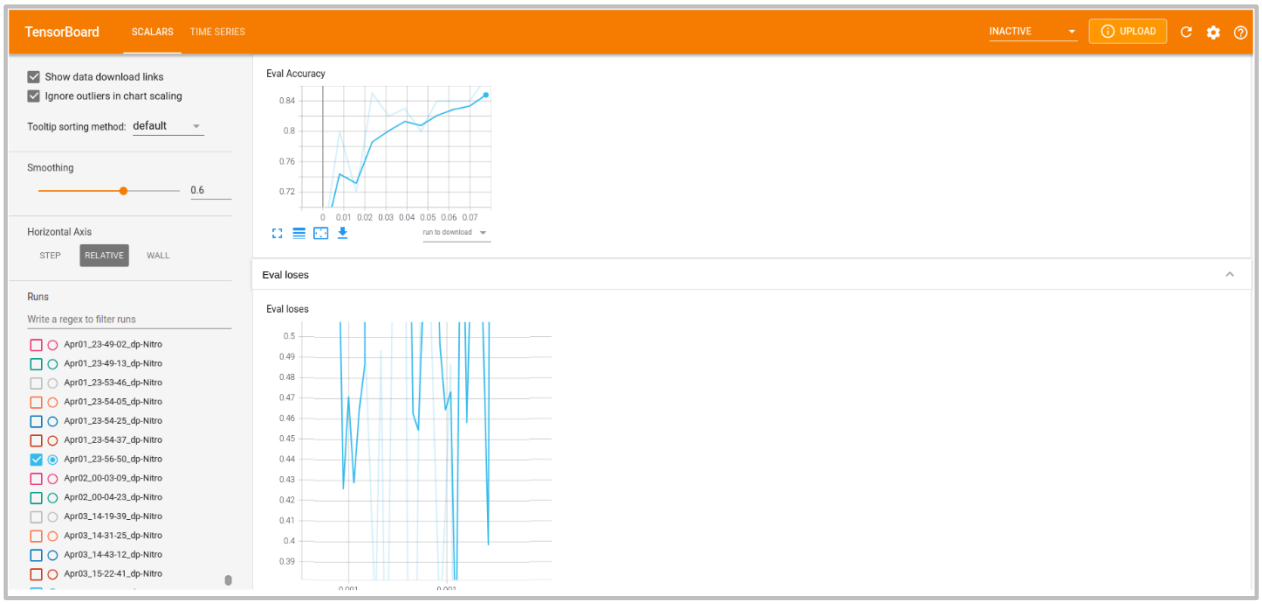
Model training results and metrics for each training epoch will be displayed in the console.

The tensorboard package is used to visualize and track the training process of the model.

Model metrics are logged to the *runs* folder.

To launch tensorboard and view the metrics, run the command:

```
$tensorboard  --logdir runs
```



4. Validate model

To start the model validation mode, set the value of the *pipeline_steps* parameter

```
"pipeline steps": ["load from db", "validate"]
```

Execute the code contained in script core_app.py

```
$python core_app.py
```

Model validation results and metrics for each training epoch will be displayed in the console and also logged in tensorboard.

5. Data augmentation (optional)

If you want to augment data by randomly shuffling MS signal elements execute the following script:

```
$python augmentaion.py
```