tspDB

tspDB enables predictive query functionality in PostgreSQL by building an additional "prediction index" for a collection of time-series of interest.

A prediction index will allow for fast data retrieval, but for entries that are:

1- At a future time step (i.e. forecasting);

2- Missing/corrupted by noise (i.e. imputation)

Our paper here (https://arxiv.org/abs/1903.07097) provides more information about how tspDB works and its performance.

Installation

For installation instruction, go to the installation page here (installation).

Getting Started

The main functionalities of tspDB are enabling predictive queries, which are enabled via creating a prediction index on your time series table. The index is created via the function `create_pindex()`. which you can use as follow:

```
select create_pindex('tablename','time_column_name','{"value_column_name"}','index_nam
e')
```

To get you familiar with tspDB capabilities, we provided a testing function that will create a set of time series tables in your database. The test function will also create several prediction indices. Run the function from your Postgres terminal

```
SELECT test_tspdb();
```

if you get at the last line

```
NOTICE:  Pindices successfully created
```

then the test has passed. Now we can check the predictive indices the test has created through

```
SELECT * FROM list_pindices();
```

You will see the three predictive indices created by the test. Now let's create our own predictive index, which we will call 'pindex1' on the time series table `mixturets2` . The prediction index is created on the time column `time` and the value column `ts_7` :

```
SELECT create_pindex('mixturets2','time','{"ts_7"}','pindex1');
```

we can see our newly created index by running `list_pindices` again:

```
SELECT * FROM list_pindices();
```

Let's now use that prediction index to prodice some *predictions*! let's for example predict at a time t that exists in the database. Effectively, we are *denoising* the existing observation or *imputing* a null observation. For example, at time 1, `ts_7` has a null value as you can see by running:

```
SELECT ts_7 FROM mixturets2 WHERE time = 1;
```

Let's *impute* this point by running:

```
SELECT * FROM predict('mixturets2','ts_7',1,'pindex1');
```

Which will return predictions as well as upper an lower bound for a 95% confidence interval. We can get a tighter bound with lower confidence by changing the confidence interval to, say 80%:

```
SELECT * FROM predict('mixturets2','ts_7',1,'pindex1', c=> 80);
```

The prediction index also support forecasting queries using the same function. For example, you can forecast the value of column `ts_7` at time 100010, ten points ahead of what exists in the database by running:

```
SELECT * FROM predict('mixturets2','ts_7',100010,'pindex1');
```

In a similar fashion, you can execute range predictive queries using `predict()`. for example, we can *impute* the first hundered points of `ts_7` using:

```
SELECT * FROM predict('mixturets2','ts_7',0,100,'pindex1');
```

or *forecast* the next 10 points using:

```
SELECT * FROM predict('mixturets2','ts_7',100001,100010,'pindex1');
```

For more information about the functionalities of tspDB, visit the API Reference (API) page.

Examples

For further examples, check the python notebook examples here (https://github.com/AbdullahO/tspdb/blob/master\notebook_examples).

Demo

Take a look at our 2020 NeurIPS demo here (https://colab.research.google.com/drive/1XqiahxFolSDnTkS8O-E0uwgIGNXcNQo0?usp=sharing).

Contributing

Please visit our Github (https://github.com/AbdullahO/tspdb/blob/master/CONTRIBUTING.md) page for more information.

License
