**Enhancing Query-based Workload Forecasting Framework for Self-Driving DBMS**

CSE 15 김승욱

2019학년도 봄 학기 과제연구

과제연구 진행 Github Repository:  
<https://github.com/wookiekim/CSED499II_POSTECH>

정돈된 주요 코드 (tf\_transformer.py) 주석 추가  
코드 링크

**개요:**

기존 [논문](https://www.cs.cmu.edu/~dvanaken/papers/forecasting-sigmod18.pdf)의 DBMS 워크로드 예측 프레임워크를 개선하고자 하는 연구이다.

기존 모델에서는 LSTM RNN을 사용하여 워크로드 데이터 (Time-series)를 예측하였는데, 이번 연구에서는 이를 Attention Is All You Need라는 논문에서 소개된 Transformer라는 모델로 대체하여 성능을 비교하고자 한다.

**코드 빌드 및 사용 방법:**

1. **기존 논문/코드 (QueryBot5000)**

기존 논문의 코드는 [github repository](https://github.com/malin1993ml/QueryBot5000)를 통해 공개되었으며, 사용 방법은 해당 repository의 README에 정리가 되어있다.

기존 논문의 코드에 대해 간략하게 설명하자면 Python으로 작성된 코드이기 때문에 별다른 빌드 과정이 없으며, run.sh 스크립트 파일을 실행하면 데이터 다운로드, 전처리, 클러스터링, 그리고 예측까지 한 번에 이루어지도록 작성이 되어있다. Python 버전과 요구되는 library은 다음과 같다.

python>=3.5   
scikit-learn>=0.18.1   
sortedcontainers>=1.5.7   
statsmodels>=0.8.0   
scipy>=0.19.0   
numpy>=1.14.2   
matplotlib>=2.0.2   
pytorch>=0.2.0\_1 (you need to install the GPU version if you want to use GPU)

그러나 Python3.5와, 위에 기재되어있는 pytorch0.2.0\_1은 현재 시점에서 이미 outdated 되었으며, pytorch0.2.0\_1과 같은 경우 매뉴얼 설치를 통해서 구축하기도 쉽지 않다. 이에 따라 Python > 3.6, 그리고 최신 pytorch 버전에서 돌아가는 버전의 코드를 과제연구 진행 repository에 구현 해두었다.

<https://github.com/wookiekim/CSED499II_POSTECH>

[해당 커밋기록](https://github.com/wookiekim/CSED499II_POSTECH/commit/5c67e87aaca32fca9c8462da1ed853c14bef9fa7#diff-64a021f04731559c5fb9f97006278b87)을 확인하면 어떤 방식으로 PyTorch 버전에 따라 코드를 수정했는지 확인할 수 있다.

1. **Transformer로 LSTM RNN을 대체한 코드**

기존 코드에서 입력단과 출력단을 유지한 상태로 model만 transformer로 바꾸고자 하였으나,

* 기존 트레이닝 코드와, 사용된 기법이 Transformer를 적용하기 어렵다고 판단되었으며
* Time series forecasting 문제는 regression으로, Embedding Layer를 대체하고 최종 출력단에 있는 Softmax를 취하지 않는 정도의 차이를 줘야하는데 PyTorch 특성상 디버깅이 어려웠다.

이에 따라 Tensorflow를 사용하여 Transformer를 이용한 Forecaster 역할을 하는 스크립트를 구축하였다. 논문상에서 Forecasting 전에 이뤄져야 할Preprocessing이나 Clustering은 기존 코드에서 변화를 주지 않았다.

기존 코드와 같이 제공되는 데이터셋에 대해서 돌아가도록 코드를 작성하였기에 parameter를 받지 않고 하드코딩 되어있다. 이 또한 코드 내부에 참조하는 클러스터 파일 (1.csv, 7.csv과 같이 되어있다)명만 바꾼 뒤 Python으로 다음과 같이 실행하면 된다.

>>python3 tf\_transformer.py

혹은 제공되어있는 run\_tf.sh 를 통해 “>>./run\_tf.sh” 와 같은 커맨드로 실행할 수 있으나, 위의 커맨드에 time 커맨드를 붙여 트레이닝 시간을 측정하기 위한 용도로 만들었기 때문에 필수적이지 않다.

**보고서에 있는 실험의 재현 방법:**

실험은 일부분 본인 컴퓨터에 VM을 띄워서 진행하였으며, 이는 속도가 너무 느려 후반부의 실험은 제 2공학관, 컴퓨터공학과 휴게실의 컴퓨터에서 진행되었다. 이에 따라서 Transformer를 적용한 방식이 Training Time이나 Inference Time에 있어서 얼마나 우수해졌는지 판단하기는 어려웠기에, 단순히 정확도만 두고 비교를 하였다.

실험결과를 보면 우선 기준이 되는 기존 LSTM RNN의 결과와, Transformer를 적용한 모델에 다양한 hyperparameter를 다르게 적용해본 뒤 얻은 결과를 두가지 방법으로 비교한다. 이는 각각 정확도 메트릭인 Log MSE와, 실제 Workload값과 예측 Workload값을 비교한 그래프이다. 굳이 그래프까지 그린 이유는, DBMS Workload 예측의 경우 정확한 값을 예측하는 것 보다는 전반적인 워크로드의 경향 (Peak, Cyclic pattern 등)을 파악하는 것이 더욱 중요하기에, 그래프를 통하여 비교하는 것 또한 중요하게 다뤘다.

각 실험은 tf\_transformer.py의 hyperparameter들을 직접 바꿔주며 진행되었다. 코드를 보면 꽤나 자명하며, 주석을 통해 어디에서 hyperparameter를 수정할 수 있는지 명시해두었다.

Tf\_transformer.py 코드를 실행하면 Log MSE의 계산과 그래프에 필요한 데이터가 csv 파일로 생성되며, 이를 Microsoft Excel로 통해 마무리 계산과 그래프 작성을 진행하였다.

그 외 부분 (GPU 사용 여부 등)은 최종 보고서에 기재되어 있다.