


1차년도 주요 결과물

(과제명) 대규모 분산 에너지 저장장치 인프라의 안전한 자율운영
및 성능 평가를 위한 지능형 SW 프레임워크 개발
(과제번호) 2021-0-00077

- 결과물명 : 물성 진단 지표 후보(SW)
- 작성일자 : 2021년 12월 30일

과학기술정보통신부 SW컴퓨팅산업원천기술개발사업
“1차년도 주요 결과물” 로 제출합니다.

수행기관	성명/직위	확인
서울대학교 산학협력단	강명주/교수	

정보통신기획평가원장 귀하

<목차>

1. 개요	1
가. 목적	1
나. 범위	1
2. 기법	2
3. 소프트웨어	2
가. 주요 기능	2
나. 사용 환경	4
다. 사용 방법	4
라. 코드	4

1. 개요

가. 목적

- 본 문서는 “대규모 분산 에너지 저장장치 인프라의 안전한 자율운영 및 성능 평가를 위한 지능형 SW 프레임워크 개발” 사업의 주요 결과물에 대한 보고서이다.
- 안전 AI 분석 엔진 중 “물성 진단 지표 후보” 소프트웨어에 대한 설명을 한다

[안전 AI 분석 엔진 SW]

구분	1차년도	2차년도	3차년도	4차년도	5차년도
안전 AI 분석 엔진	물성 진단 지표 후보(SW)	실험 데이터로 학습한 이상 탐지 알고리즘(SW)	전이학습을 완료한 이상 탐지 알고리즘(SW)	최적화된 이상 탐지, 성능 진단 알고리즘(SW)	이상 탐지 알고리즘 및 사고 대응 알고리즘(SW)
	데이터 전처리 알고리즘(SW)	실험 데이터로 학습한 성능 진단 알고리즘(SW)	전이 학습을 완료한 성능 진단 알고리즘(SW)	개선된 이상 탐지, 성능 진단 알고리즘(SW)	상태 진단 알고리즘 및 효율적 운영 분석 결과(SW)

나. 범위

- 배터리의 물성을 진단할 수 있는 다양한 지표를 정의
- 이를 구현하는 소프트웨어를 개발
- 구현 소프트웨어의 특징 설명

2. 기법

- 2차 전지의 효율적이고 안전한 사용을 위해서 잔량(State of Charge, SOC)나 노화상태(State of Health, SOH)와 같은 배터리의 물성 지표가 이용되고 있다. 이러한 배터리의 지표를 추정하기 위하여 배터리의 충전과 방전 시의 전압과 전류 데이터가 사용되고 있다.
- 본 프로그램은 이러한 배터리의 지표를 보다 정확하게 추정하기 위해, 다양한 지표의 수치 계산 과정에서 정상범위를 벗어나는 수치 불안정 문제를 수치계산 방식의 개선을 통하여 개선함.
- 이러한 과정을 통해 용량(Capacity), 전하량(Q), 잔량(SOC), 노화상태(SOH) 등의 기본적인 배터리 상태 지표를 추정함.

3. 소프트웨어

가. 주요 기능

- 용량(Capacity) 추정

```
# Estimate capacity at each moment using reference discharge
# These values are also used for estimating the charge-discharge rate
# during the unknown interval.
df.loc[:, 'capacity'] = np.nan
temp = df.loc[df['charge_type'] == 'reference discharge']
ref_D_start_idx = list(temp.loc[(temp['t'] - temp['t'].shift(1)) > 100, :].index)
ref_D_end_idx = list(temp.loc[(temp['t'].shift(-1) - temp['t']) > 100, :].index)
ref_D_start_idx.insert(0, temp.index[0])
ref_D_end_idx.append(temp.index[-1])
assert len(ref_D_start_idx) == len(ref_D_end_idx)

for idx in range(0, len(ref_D_start_idx), 1): # Note: this does not take long time !
    start = ref_D_start_idx[idx]
    end = ref_D_end_idx[idx]
    discharge_I = temp.loc[start:end, 'I']
    discharge_dt = temp.loc[start:end, 'dt']
    exact_discharge_capa = -(discharge_I * discharge_dt).sum()
    df.loc[end, 'capacity'] = exact_discharge_capa
```

- 전하량(Q) 추정

```

# Define 'Q' (and also modify 'I' during 'unknown charge')
# Note : Integration of 'I' to obtain 'Q' is numerically unstable
# Note : Reset SOC as 100% (i.e, 'Q' = 'capa') when one of the following:
#   (1) 'charge (after random walk discharge)' ends
#   (2) reference type A starts
#   (3) reference type C starts
df.loc[:, 'Q'] = np.nan
first_capa = df.loc[:, 'capacity'].iloc[0]
no_unk_df = df.loc[df['UNK'] == False, :].copy()
no_unk_df.loc[:, 'Q'] = no_unk_df.loc[:, 'I'].cumsum() + first_capa

```

- 잔량(SOC) 추정

```

# Reset SOC as 100% at particular points
idx1 = (no_unk_df['RW'] == True) & (no_unk_df['CDR'] != 'C') & (no_unk_df['CDR'].shift(1) == 'C')
idx2 = (no_unk_df['ref_type'] == 'A') & (no_unk_df['ref_type'].shift(1) != 'A')
idx3 = (no_unk_df['ref_type'] == 'C') & (no_unk_df['ref_type'].shift(1) != 'C')
idx = idx1 | idx2 | idx3
err_Q = no_unk_df.loc[idx, 'capacity'] - no_unk_df.loc[idx, 'Q']
temp = no_unk_df['Q'].copy()
temp.loc[:] = np.nan
temp.loc[idx] = err_Q
temp = temp.fillna(method='ffill').fillna(0.)
no_unk_df.loc[:, 'Q'] += temp # Modify the error

# Define 'SOC' by 'Q' / 'capacity'
df.loc[:, 'SOC'] = df['Q'] / df['capacity']

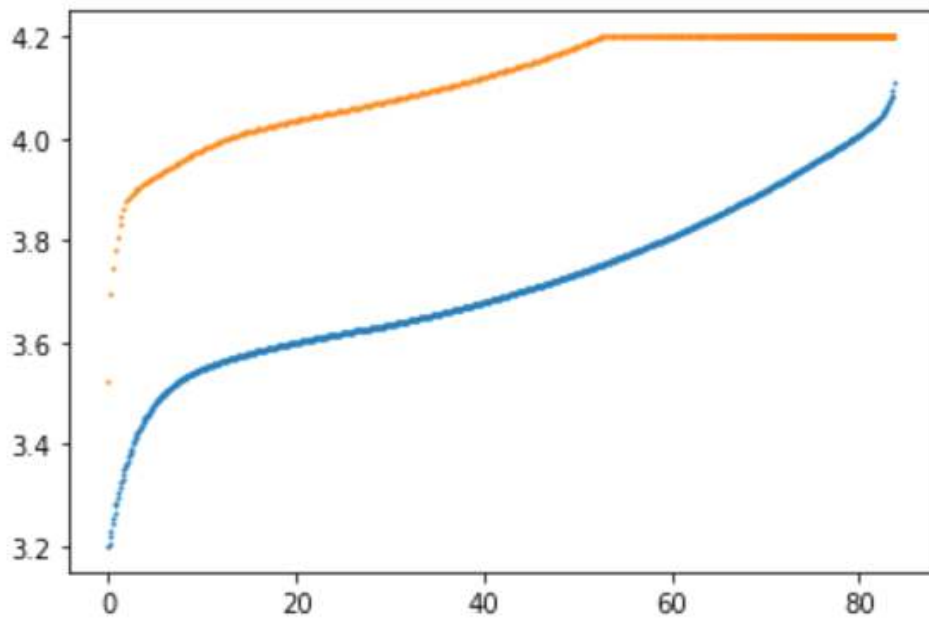
```

- 노화상태(SOH) 추정

```

# Define "SOH"
df.loc[:, 'SOH'] = np.nan
df.loc[:, 'SOH'] = df.loc[:, 'capacity'] / df.loc[:, 'capacity'].max()

```



Q-V Plot

나. 사용 환경

- 본 프로그램은 python 코드로 구현되었음
- Python이 설치된 어떠한 OS 환경에서도 사용이 가능함
- 코드의 구동에는 python의 OS, numpy, pandas, scipy 라이브러리가 필요함

다. 사용 방법

- 입력과 출력 데이터는 parquet 형식을 사용함
- data_file에 입력 데이터 path, basic_file과 advanced_file에 출력 데이터 path를 적어준 뒤
- python preprocessing.py 를 실행해줌

라. 코드

- OpenESS/SNU_AI/data_preprocess/preprocessing.py 코드 중 preprocess_advanced 모듈