YAML_TMI

CheolSeong Park
tjd4987@naver.com

YAML_TMI

- JSON과 같이 데이터 직렬화에 사용되는 포맷
- 딥러닝 프로젝트에서 하이퍼파라미터 등의 변수를 YAML을 이용하여 관리할 수 있다.

YAML_TMI

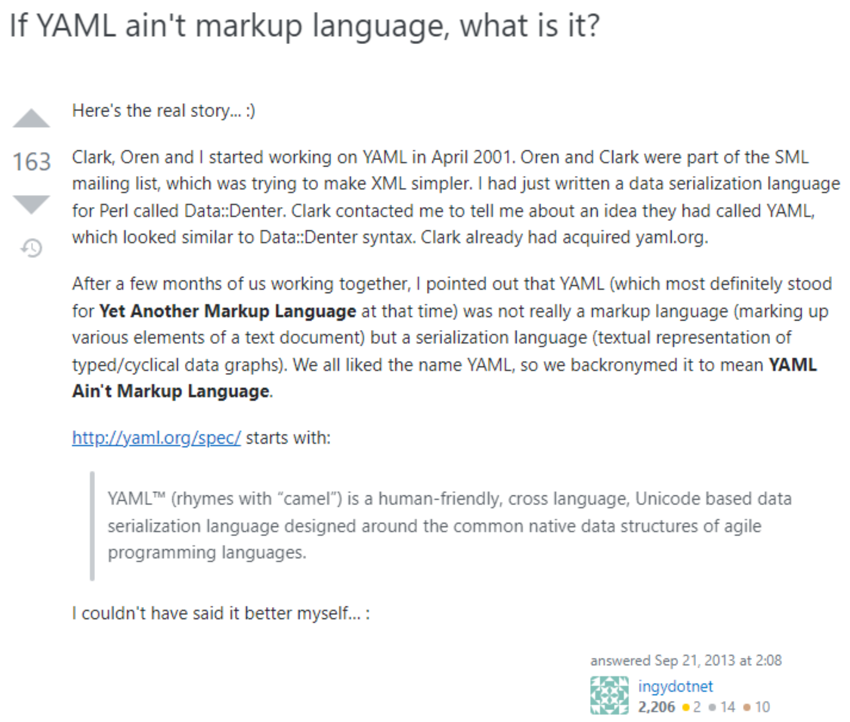
- YAML
  - Yet Another Markup Language
  - **또 다른 마크업 언어** 였으나…

YAML_TMI

- YAML
  - ~~Yet Another Markup Language~~
  - YAML Ain’t Markup Language
    - 데이터 직렬화 핵심을 강조하기 위해

YAML_TMI

- YAML
  - YAML Ain't Markup Language
    - 창립자 중 하나인 Ingy döt Net의 답변



If YAML ain't markup language, what is it?

163

Here's the real story... :)

Clark, Oren and I started working on YAML in April 2001. Oren and Clark were part of the SML mailing list, which was trying to make XML simpler. I had just written a data serialization language for Perl called Data::Denter. Clark contacted me to tell me about an idea they had called YAML, which looked similar to Data::Denter syntax. Clark already had acquired yaml.org.

After a few months of us working together, I pointed out that YAML (which most definitely stood for **Yet Another Markup Language** at that time) was not really a markup language (marking up various elements of a text document) but a serialization language (textual representation of typed/cyclical data graphs). We all liked the name YAML, so we backronymed it to mean **YAML Ain't Markup Language**.

http://yaml.org/spec/ starts with:

> YAML™ (rhymes with "camel") is a human-friendly, cross language, Unicode based data serialization language designed around the common native data structures of agile programming languages.

I couldn't have said it better myself... :

answered Sep 21, 2013 at 2:08
ingydotnet
2,206 ● 2 ● 14 ● 10

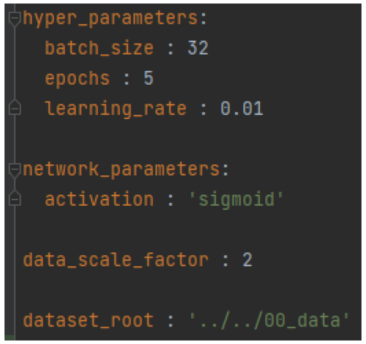
YAML_TMI

- YAML의 특징
  - 가독성 좋음
  - 사용하기 쉬움
  - JSON의 superset
  - python과 유사한 문법
    - → python 기반 딥러닝 프로젝트와 궁합이 좋다

YAML_TMI

- vs another data serilization

YAML



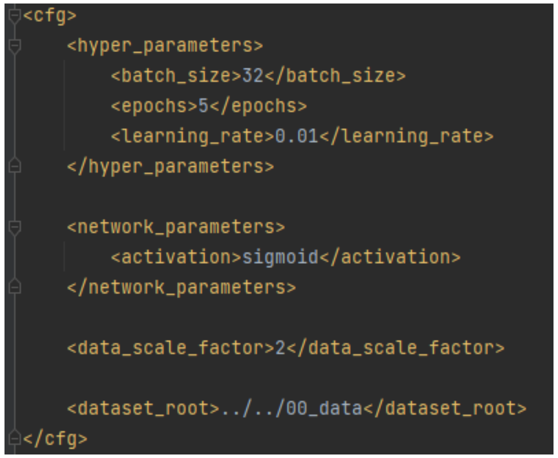
```yaml
hyper_parameters:
  batch_size : 32
  epochs : 5
  learning_rate : 0.01

network_parameters:
  activation : 'sigmoid'

data_scale_factor : 2

dataset_root : '../../00_data'
```

XML



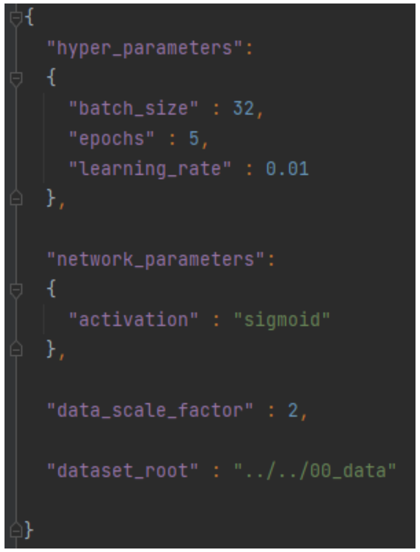
```xml
<cfg>
    <hyper_parameters>
        <batch_size>32</batch_size>
        <epochs>5</epochs>
        <learning_rate>0.01</learning_rate>
    </hyper_parameters>

    <network_parameters>
        <activation>sigmoid</activation>
    </network_parameters>

    <data_scale_factor>2</data_scale_factor>

    <dataset_root>../../00_data</dataset_root>
</cfg>
```

JSON



```json
{
  "hyper_parameters":
  {
    "batch_size" : 32,
    "epochs" : 5,
    "learning_rate" : 0.01
  },

  "network_parameters":
  {
    "activation" : "sigmoid"
  },

  "data_scale_factor" : 2,

  "dataset_root" : "../../00_data"

}
```