AI CRYPTO

AI BlockChain for Decentralized Economy

The Business Report, the first quarter of 2019

개발

개발 보고

AI Crypto 2nd ver. Dapp 출시

1) 주요기능
   - 마이닝풀 서버를 이용한 딥러닝 트레이닝 및 배포 수행
   - 분산된 환경에서 딥러닝 수행
   - 메타마스크 로그인
   - AIC 과금 결제

2) 예상 수요
   - 개인 개발자
   - 관련 전공자
   - 연구 단체
   - 관련 기업

작년 10 월 첫 번째 버전의 AI Crypto Dapp(Decentralized Application)을 출시한 후 6 개월의 개발 기간을 거쳐 두 번째 버전의 Dapp 을 출시하였습니다. 고성능 하드웨어를 모아 암호화폐 채굴에 사용하는 마이닝풀은 현재 투자 비용 대비 이윤이 낮은 상황입니다. AI Crypto 는 마이닝풀의 고성능 하드웨어를 인공지능을 포함한 여러 분야의 개발에 사용함으로써 고성능 GPU 자원이 더 많은 사람들에게 도움을 주는 미래를 꿈꾸었습니다. 여기서 착안한 AI Mining 과 AI Deep Learning 의 기술을 구체화하고 프로그램으로 구현한 것이 이번 Dapp 입니다.

이번 Dapp 은 마이닝풀 서버를 이용한 분산 딥러닝 트레이닝(distributed deep learning training), 마이닝풀 서버에 트레이닝이 완료된 딥러닝 모델 배포(deployment)를 핵심 기능으로 하여, 이들 핵심 기능을 컨트롤하기 위한 CLI (command line interface)와 메타마스크 연동을 통한 AIC 과금 결제 기능을 포함하고 있습니다.

예상 수요는 분산 환경에서의 저렴하고 빠른 트레이닝을 원하는 딥러닝 연구자와, 딥러닝을 활용한 서비스 개발 및 제공을 원하는 기업 등이 될 것입니다. 특히 비싼 장비 구입이 부담되는 딥러닝에 입문하는 학생부터 대규모 서비스를 계획 또는 제공하고 있는 기업까지, 활용 범위가 매우 넓을 것으로 기대합니다.

AI Crypto 2nd ver. Dapp 의 사용방법은 [별첨 1]에서 확인하실 수 있습니다.

로드맵 대비 개발 달성도

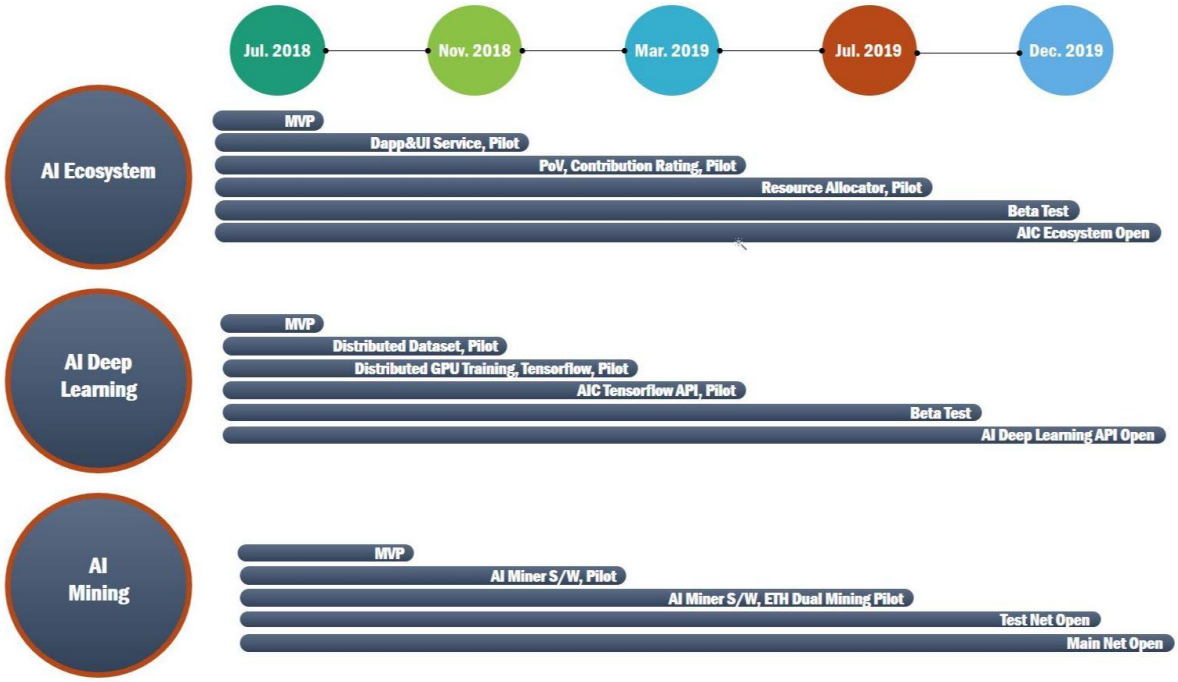

그림 1. Road Map

AI Deep Learning

그림 2. AI Deep Learning 개발 달성도

AI Mining

그림 3. AI Mining 개발 달성도

AI Ecosystem

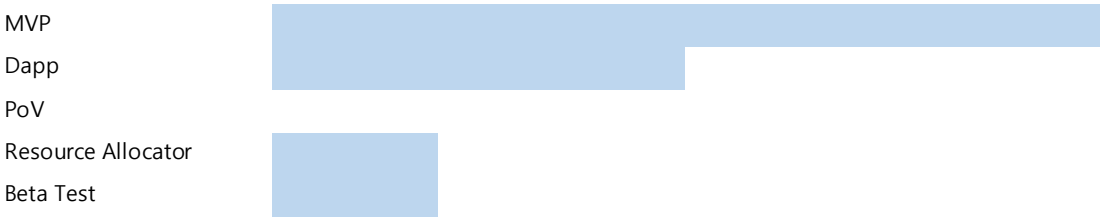

그림 4. AI Mining 개발 달성도

운영

홍보

AI Crypto 2nd ver. Dapp 언론발표

두 번째 버전의 Dapp 를 공개하며 아래 기사로 글로벌 언론과 한국 언론에 소개되었습니다.

AI Crypto, AI Blockchain Platform 2nd version released

AI CRYPTO released the second version of Dapp (Decentralized Application) on March 14th. It has added the key practical functions that users need after launching the world's first Dapp for artificial intelligence(AI) development in October last year.

The newly developed key functions of AI Crypto Dapp are AICrypto Deeplearning Service(ADS) and payment mechanism with AIC(AI Crypto's Coin). ADS is consisted of Learning Service; AI model learns by virtual GPU bound with multiple high performance hardware in Mining Pool and Distribution Service, which then distributes the learned model to Mining Pool servers.

Users can utilize its deep learning technology with ADS after logging in to AI Crypto Dapp via Metamask by paying the expenses in AIC. The current service is for general developers and planned to be expanded for the business related customers in the future.

AI CRYPTO is a team of experts in artificial intelligence, developing an AI resource distribution platform based on Blockchain to provide individuals to develop and use AI. In 2018, the team launched Dapp for AI development and made partnership with Shinhan Data System in recognition of its own technology expanding its business customers related in AI, VR, and FinTech.

[글로벌 채널 2019.03.16]

http://www.koreaittimes.com/news/articleView.html?idxno=89725

에이아이크립토, 인공지능 블록체인 플랫폼 두 번째 버전 출시

에이아이크립토(AI CRYPTO)가 지난 14 일 두번째 버전의 Dapp(Decentralized Application)을 출시했다고 발표했다. 작년 10 월, 세계 최초로 인공지능 개발을 위한 Dapp 을 오픈한 이후 사용자에게 필요한 실제적인 기능을 추가한 것이다.

이번에 공개된 AI Crypto Dapp 의 주요 기능은 에이아이크립토 딥러닝 서비스(Aicrypto Deeplearning Service)와 에이아이크립토의 토큰(AIC)을 이용한 서비스 결제이다. 에이아이크립토 딥러닝 서비스는 암호화폐 채굴에 사용되는 마이닝풀의 고성능 하드웨어 여러 대를 묶어 생성된 가상 GPU 로 인공지능 모델을 학습시키는 학습서비스와 학습된 모델을 다시 마이닝풀 서버에 배포하는 배포서비스를 포함한다.

딥러닝을 필요로 하는 이용자는 메타마스크를 통해 Ai Crypto Dapp 에 로그인 후 에이아이크립토 딥러닝 서비스를 이용하고 비용을 AIC 로 결제할 수 있다. 현재 공개된 서비스는 일반 개발자를 위한 서비스이며, 향후 기업을 대상으로 사용자 층을 확대할 예정이다.

에이아이크립토는 인공지능 분야 전문가로 구성된 팀으로 누구나 인공지능 기술을 개발하고 이용할 수 있도록 블록체인 기반의 인공지능 자원 유통 플랫폼을 개발하고 있다. 지난 2018 년에는 인공지능 개발을 위한 Dapp 를 출시하며 독자적인 기술을 인정받아 신한데이터시스템과 파트너쉽을 체결하였고 AI, VR, 핀테크 등 관련기업을 고객사로 확보하며 사용자 층을 넓혀가고 있다.

[한국 채널, 2019.03.20]

http://www.itdaily.kr/news/articleView.html?idxno=93690

다국어 커뮤니티 운영

에이아이크립토는 ICO 종료 후, 한국어, 영어 그리고 중국어의 총 3 개 국어 사용자 커뮤니티를 관리하고 있습니다. 각 언어 사용자에게 공통으로 제공되는 서비스 채널은 웹페이지, 블로그, 이메일이며 보다 신속하고 정확한 답변을 위해 각 언어의 사용자들에게 친숙한 채팅 채널을 운영 중입니다(한국어:카카오톡, 영어/중국어:텔레그램).

[별첨 1] AI Crypto 2nd ver. Dapp Tutorial

```
# Tutorial
In this tutorial, we will train and deploy deep learning by using cluster of
cryptocurrency mining sytems (aka miner).

### 0. Background
Open source [Horovod](https://github.com/horovod/horovod) is used for deep learning
distributed training.

In this tutorial, we will distribute train MNIST Classification Example that is based
on [Keras](https://keras.io), [Horovod](https://github.com/horovod/horovod) and
deploy the result of training as a  [Flask](http://flask.pocoo.org/) App.

### 1. Requirements

  Let's start off by installing requirements.

 * **Metamask**
  Metamask Chrome Extension and Etherium Wallet Address are required for login.
  Open Chrome browser, and after installing [Metamask Chrome
Extension](https://chrome.google.com/webstore/detail/metamask/nkbihfbeogaeaoehlefnkod
befgpgknn?hl=ko), create a Etherium wallet.

 * **Docker**
  User CLI is provided through Docker Image. Docker is installed differently by each
OS, so check out the [Docker Installation Guide](https://docs.docker.com/install) for
more information.<br>
  In Ubuntu 16.04, you can install Docker by using the command below.

   ```
   sudo apt-get update && sudo apt install docker.io
   ```


2. Download User Docker Image
```
sudo docker pull aicrypto/user
```


3. Run User Docker Container
```
sudo docker run -it aicrypto/user
```
```

4. Login
```
aicctrl login
```

5. Create Deep Learning Cluster
We will create a Deep Learning Cluster to distributed tran deep learning.
```
aicctrl create
```

Run the command above, and the screen will change like below, telling you to choose a Gateway.

```
[aicctrl] gateway lists below.
   1. name: test_gateway, available miners: 8
[aicctrl] choose gateway :
```

Gateway is a network access point that is consisted of Miners. Distribute training is impossible for miners that have different gateways. This is beta mode, and we provide only 1 Gateway at the moment, so write 1 and press enter to select Gateway 1.

```
[aicctrl] Input cluster name :
```

Choose a cluster name.

```
[aicctrl] Input miner counts(max == 12) :
```

Enter the number of miners you wish to include in your cluster. max == 12 means that yu can select up to 12 miners. Let's choose 8 for now.

```
[aicctrl] No ssh key pairs.
[aicctrl] Input ssh key name to be created :
```

Then, deep learning cluster need SSH Key Pair. When you enter your desired SSH Key Pair name, it will automatically be created and be connected. Input a name.

```
[aicctrl] No data storage.
[aicctrl] Input data storage name to be created :
```

Then, enter any name to create a data storage that you will mount to your cluster.

```
[aicctrl] Now 8 miners are creating.
[aicctrl] Successfully created 8 miners
```

Then, you are finished. You have now created a DL Cluster that has 8 miners.
(It will take approximately 1 min for the real miner to be ready.)

6. Let's connect to miner on DL Cluster

```
aicctrl connect
```

A screen like below will show up.

```
[aicctrl] cluster lists below.
   1. name: test-cluster, storage: test-storage, minerCount: 8
[aicctrl] choose cluster :
```

When you input the number 1 and tap enter,

```
index, hostname
1 m6
2 m16
3 m4
4 m13
5 m17
6 m5
7 m8
8 m9
```

```
[aicctrl] enter miner index to connect :
```

a screen like above will pop up. Let's input any number and connect. We will input number 4 and connect to m13 miner.

```
Warning: Permanently added '[125.188.51.150]:43531' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 16.04.5 LTS (GNU/Linux 4.15.0-43-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

root@m13:~#
```

A screen like above will show up. You havve successfully connected to m13 miner SSH.

```
exit
```

When you enter the command above, your ssh connection will be over.

Let's try connecting SSH directly without using CLI. For manual SSH connection, you need an SSH Key. By using the command below, you can download an SSH Key from the server.

```
aickey save
```

A screen like below will show up.

```
```

```
[aickey] ssh key pair lists below.
   1. name: test-key
[aickey] choose ssh key pair :
```

Down load a test-key by inputting number 1.

```
[aickey] Successfully saved ssh key pair, filename .test-key on current folder.
```

When a screen shows up like above, you are finished. On the route where you have ran the current CLI, it is downloaded as `.keyname`.

```
aicctrl list
```

View the miner's IP and Port by using the command above, and connect like below.

```
ssh root@125.188.51.150 -p 43531 -i .test-key
```

7. Upload Horovod examples to the remote storage

Download an Horovod example.

```
apt-get install -y --no-install-recommends subversion
svn checkout https://github.com/uber/horovod/trunk/examples
rm -rf examples/.svn
```

On the `examples` directory of the current path, your example will be downloaded. Among the examples that you have downloaded, we will upload keras_mnist.py to the remote data storage. Input the command below.

```
aicdata upload
```

Then, select `1 + Enter`.

```
[aicdata] ssh key pair lists below.
    1. name: test-storage
[aicdata] choose data storage :
```

Then, enter examples/keras_mnist.py.

```
[aicdata] Input source path on your system :
```

Then,

```
Warning: Permanently added '[125.188.51.150]:5500' (ECDSA) to the list of known hosts.
[aicdata] Uploaded.
```

Let's check if it is correctly uploaded. Input command below.

```
aicdata download
```

Then, Select 1.

```
[aicdata] ssh key pair lists below.
    1. name: test-storage
[aicdata] choose data storage :
```

When you input `. + Enter` , it will download to the current working directory.

```
[aicdata] Input download path on your system :
```

Then, input `. + Enter` to download all the files that are on the root path of Data Stroage.

```
[aicdata] Input source path on your storage :
```

Then,

```
[aicdata] Downloaded.
```

`ls` 명령을 통해 확인해보면 `keras_mnist.py`가 다운로드 된 것을 확인할 수 있을 것입니다.

8. Train
Discribed train can be executed through the command below.

```
aicrun keras_mnist.py
```

(aicrun takes the script argument. Script argument must be the script path on the Data Storage.)

Then, press 1 to select the Cluster created before.

```
[aicrun] Start aicrun
[aicrun] cluster lists below.
    1. name: test-cluster, storage: test-storage, minerCount: 8
[aicrun] choose cluster :
```

Then

```
[aicrun] How many miners that you use on this training? (all == 0)(max == 8) :
```

If you input, then all the minres in the cluster will be used for training. If you input 5, only 5 will be used to train. For now, we will input 0 to train on all miners.

Input `0 + Enter`, then

```
[aicrun] Input gpu counts that you use per miner :
```

On the current beta, all the miners will have 6 GPUs. When you input 3, each miners will use 3 GPUs for training. We will input `3 + Enter`.
We have previously selected 8 miners, so total of 24 GPUs will be used.

```
[aicrun] Start machine learning
```

Then, training will begin.

```
[aicrun] End machine learning
[aicrun] Deeplearning execution time -> 0:5:11 seconds.

[aicrun] creating training
[aicrun] done
[aicrun] Finished
```

When finished, it will show the total amount of execution time.

9. Download the result
Despite finishing training above, nothing has changed on the current system. This is because keras_mnist.py saves the result on the first miner. Let's connect to the miner to copy the result file to the remote storage and download it to the local system.

Let's connect to the first miner by the command below.

```
aicctrl connect
```

Then,

```
[aicctrl] cluster lists below.
    1. name: test-cluster, storage: test-storage, minerCount: 8
[aicctrl] choose cluster :
```

Select 1,

```
index, hostname
1 m6
2 m16
3 m4
4 m13
5 m17
6 m5
7 m8
8 m9
[aicctrl] enter miner index to connect :
```

and select first miner. Then,

```
Welcome to Ubuntu 16.04.5 LTS (GNU/Linux 4.15.0-43-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
root@m6:~#
```

upon successful connection, screen will show up like above. Let's check the files through with the below command.

```
ls
```

Results will show like below.

```
checkpoint-1.h5  storage
```

By the command below, you can move checkpoint-1.h5 file to the storage directory.

```
mv checkpoing-1.h5 storage/
```

Then, use `exit` command to terminate ssh connection.

Use the command below to download the result file.

```
aicdata download
```

Select 1 to choose test-storage.

```
[aicdata] ssh key pair lists below.
    1. name: test-storage
[aicdata] choose data storage :
```

Then, input `. + Enter` to download all the files on the remote data storage.

```
[aicdata] Input download path on your system :
```

Then, input . which means the current directory.

```
[aicdata] Input source path on your storage :
```

A message like below will show up when the file is successfully downloaded.

```
[aicdata] Downloaded.
```

이제 `ls` 를 통해 현재 폴더 파일을 조회하면 결과 파일을 확인 할 수 있습니다.

10. Deploy

We will first remove the cluster that we have created above.

```
aicctrl remove
```

Then, select `1 + Enter`.

```
[aicctrl] cluster lists below.
    1. name: test-cluster, storage: test-storage, minerCount: 8
[aicctrl] choose cluster :
```

aicdeploy automatically deploys Flask Project that has structure like below via Nginx, Uwsgi.

```
project/
    app.py              (Flask app file must have a name app.py.)
    requirements.txt    (Write the required Python Package according to the style of requirements.txt)
```

Let's make the Flask App to deploy the trained model above.

First, create a directory named deploy_project, and move the result file named checkpoint-1.h5 to the directory.

```
mkdir deploy_project
mv checkpoint-1.h5 deploy_project/
```

Then, create app.py and paste the below codes.

```
import tempfile

import numpy as np

from flask import Flask, jsonify, request, Response
from PIL import Image
from keras.models import load_model

model = load_model('checkpoint-1.h5')
model._make_predict_function()

app = Flask(__name__)
```

```
@app.route('/', methods=['POST'])
def predict():
    if request.method == 'POST':
        if 'image' not in request.files:
            return Response('image file not attached.', status=400)

        with tempfile.TemporaryFile() as tmp:
            tmp.write(request.files['image'].read())
            image = Image.open(tmp)
            image_arr = np.array(image) ₩
                .reshape(1, 28, 28, 1) ₩
                .astype('float32')
            image_arr /= 255

        return jsonify({
            'result': int(model.predict(image_arr)[0].argmax(axis=0))
        })

    else:
        return 'only post is permitted'

if __name__ == '__main__':
    app.run()
```

Let's deploy. Enter the command below.

```
aicdeploy create
```

Then, select 1.

```
[aicdeploy] gateway lists below.
    1. name: test_gateway, available miners: 11
[aicdeploy] choose gateway :
```

Now, input the name of the cluster. We will use test-deploy.

```
[aicdeploy] Input cluster name :
```

Enter the number of miners. We will choose 3.

```
[aicdeploy] Input miner counts(max == 11) :
```

Then, input the path of the Flask Project. The path can be absolute or relative.
For now, we will enter deploy_project.

```
[aicdeploy] Input deploy project path on your system :
```

You have succeeded when the screen shows up like below.

```
[aicdeploy] Now compressing project on /root/deploy_project
[aicdeploy] Now 3 miners are creating.
[aicdeploy] Successfully created 3 miners
[aicdeploy] API address : http://125.188.51.150:31782
```

Let's figure out if the deploy was successful.

First, download the test image.

```
wget https://pool.aicrypto.ai/images/test.jpg
```

Then, install requests python package.

```
pip install requests
```

Input the following code below on test.py.

```
import requests

r = requests.post('http://125.188.51.150:31782', files={'image': open('test.jpg'), })
```

```
if not r.ok:
   if r.status_code == 502:
      print('Miner is now activating. Please try again 10 secs later.')
   else:
      r.raise_for_status()

print(r.json())
```

When you execute test.py, you can check that the deploy was successful.

If an error occurs, you can print uwsgi log via the command below.

```
aicdeploy logs
```

Contact

Website

http://aicrypto.ai/

Blog

https://medium.com/aicrypto

E-mail

hello@aicrypto.ai

Kakao Talk(Kr)

https://open.kakao.com/o/sww61RQ

Telegram(Eng)

https://t.me/aicryptoai

Telegram(Cn)

https://t.me/aicryptoai_ch