Hyeongchan Kim

https://github.com/kozistr, http://kozistr.tech/about

EDUCATION **Korea University of Technology and Education** Mar 2016 –

CHALLENGES & AWARDS

Kaggle Challenges :: Competition Expert

top 1% **Google Brain – Ventilator Pressure Prediction** (20 / 2605), 2021.
top 4% **SIIM-FISABIO-RSNA-COVID-19 Detection** (47 / 1305), 2021.
top 7% **Shopee – Price Match Guarantee** (166 / 2426), 2021.
top 2% **Cornell Birdcall Identification** (24 / 1395), 2020.
top 9% **ALAKSA2 Image Steganalysis** (93 / 1095), 2020.
top 4% **Tweet Sentiment Extraction** (84 / 2227), 2020.
top 4% **Flower Classification with TPUs** (27 / 848), 2020.
top 4% **Bengali.AI Handwritten Grapheme Classification** (67 / 2059), 2020.
top 3%, **Kannada MNIST Challenge** (28 / 1214), 2019.

Domestic Challenges

6th place, **NAVER NLP Challenge**, SRL Task, 2018.
4th / 13th place, **NAVER A.I Hackathon**, 2018.
Final Round (Digital Forensic), **A.I R&D Challenge**, 2018.
9th place (3rd price, A book as an award), **TF-KR MNIST Challenge**, 2017.

PUBLICATIONS

[1] Kim et al, CNN ARCHITECTURE PREDICTING MOVIE RATING FROM AUDIENCE'S REVIEWS WRITTEN IN KOREAN. Jan. 2020.

INDUSTRY EXPERIENCE

Watcha, Seoul, South Korea Jun 2020 – Dec 2021

Machine Learning Researcher

- Developed a new sequential recommendation architecture. (named *Trans4Rec*)
  - Newly proposed transformer architecture to improve the performance in a general manner.
  - Apply proper post-processing logic into the model.
  - In A/B (online) test, *FutureFLAT vs Trans4Rec* (statistically significant p-value < 0.01)

Click Ratio: improved **1.01%+**

- Developed a music recommendation system (prototype)
- Developed a training recipe to train sequential recommendation architecture robustly. (In service), (named *FutureFLAT*)
  - Build a new module to understand better at the time of inference.
  - Apply augmentations to the various features, leads to performance gain and robustness.
  - In A/B (online) test, *FLAT* vs *FutureFLAT* (statistically significant p-value < 0.05)

    - Compared to the previous model, there's been no (statistically significant) change.

    - However**,** it still seems to be better on the **offline metrics** & **training stability**. So, we chose to use it.
  - In A/B (online) test, *Div2Vec* vs *FutureFLAT* (statistically significant p-value < 0.05)

    ***Viewing Days (mean):** improved **1.012%+**

    ***Viewing Minutes (median):** improved **1.015%+**
- Developed the model to predict users' view-time of the contents.
  - Predict how many people going to watch, how much time people going to watch the content before the content is supplied.
  - Find out which features impact users' watch.
- Developed the pipeline to recognize main actors from the poster & still-cut images.
  - Utilize SOTA face detector & recognizer.
  - Optimize pre/post-processing routines for low latency.
- Developed a novel sequential recommendation architecture to recommend what content to watch next. (In service), (named *FLAT*)
  - Achieve SOTA performance compared to previous SOTA architectures (e.g. *BERT4Rec*).
  - In A/B (online) test, *previous algorithm* vs *FLAT* (statistically significant p-value < 0.05)

    Paid Conversion: improved **1.39%p+**

    ***Viewing Days** (mean)**:** improved **0.25%p+**

    ***Viewing Minutes** (median): improved **4.10%p+**

    Click Ratio: improved **4.30%p+**

    Play Ratio: improved **2.32%p+**
- Developed Image Super-Resolution model to upscale movie & tv posters, still-cuts.
  - Optimize the codes for fast inference time & memory-efficiency on CPU.
  - In internal evaluation (qualitative evaluation by the designers), it catches details better & handles higher resolution & takes a little time.

- Working as a full-time.

% ***Viewing Days**: how many days users active on an app each month.
% ***Viewing Minutes**: how many minutes user watched the contents.

Rainist, Seoul, South Korea Nov 2019 – Jun 2020
Machine Learning Engineer

- Developed the category classification model of card transactions, designed lightweight purpose for low latency. (In service)
  - In A/B (online) test (statistically significant p-value < 0.05)
    ***Accuracy:** improved about **25 ~ 30%p**
- Developed the RESTful API server to serve (general purpose) machine learning models.
  - About 1M MAU service, 500K ~ 1M transactions / day. (1 transaction = about 100 samples)
  - Utilized inference-aware framework (onnx) to reduce the latency.
    median 100 ~ 200ms / transaction
  - **zero failure rate** (zero 40x, 50x error)
  - Deployed & managed with Kubernetes.
- Developed the classification model, forecasting the possibility of loan overdue.
- Worked as a full-time.

% ***Accuracy**: how many people don't update/change their transactions' category.

VoyagerX, Seoul, South Korea Jan 2019 – Sep 2019
Machine Learning Engineer

- Developed speaker verification & diarization models to recognize the arbitrary speakers recorded from the noisy environments.
- Developed a semantic image segmentation model to identify a region of hair.
- Developed an image in-paint model to remove hair naturally from the face.
- Worked as an intern.

ELCID, Pangyo, Korea Jun 2016 - Aug 2016
Penetration Tester

- Penetrated some products related to network firewall and anti-virus.
- Worked as a part-time job.

OUTSOURCING

Korea University Course Information Web Parsing, ITL July 2017 – Mar 2018

AWS CloudTrail logger analyzer / formator, ELCID Sep 2019 – Oct 2019

RESEARCH EXPERIENCE

Heterogeneous Parallel Computing Lab, Cheonan, Korea Sep 2018 - Dec 2018

Undergraduate Research

- Wrote a paper about the CNN architecture, which utilizes a channel-attention method to TextCNN model, brings performance gain over the task while keeping its latency, generally.
- Handling un-normalized text with various convolution kernel size and spatial dropout.

TALKS

NAVER NLP Workshop 2018, Pangyo, Korea Dec 2018

- SRL Task, challenging without any domain knowledge. Presented about trials & errors during the competition.

PROJECTS

Generative

Awesome Generative Adversarial Networks (Stars 670+) July 2017 –

Implement lots of Generative Adversarial Networks in TF 1.x. & 2.x. Novelty of this project is implementing lots of GANs in TF 1.x & 2.x based on the papers with some tweaks.

gan-metrics (Stars 5) Mar 2020 –

Implement lots of metrics for evaluating GAN in pytorch.

I2I Translation

Improved Content Disentanglement (Stars 3+) Sep 2019

Re-implement / tune 'Content Disentanglement' paper in pytorch.

Image Inpainting

Improved Edge-Connect (Stars 9) Oct 2019

Re-implement / tune 'Edge-Connect' paper in pytorch.

Style Transfer

Neural Image Style Transfer Mar 2018

Implement a neural image style transfer.

Segmentation

Awesome Segmentation (Stars 65+) Aug 2018

Implement lots of image semantic segmentation and ordered the papers.

Optimizer

pytorch-optimizer (Stars 20+) Sep 2021-

Bunch of optimizer implementations in PyTorch with clean-code, strict types. Also, including useful optimization ideas. Most of the implementations are based on the original paper, but I added some tweaks.

AdaBound Optimizer (Stars 40+) Jan 2019

Implement AdaBound Optimizer (Luo et al. 2019) w/ some tweaks in tensorflow.

RAdam Optimizer (Stars 4+) Sep 2019

Implement RAdam Optimizer (Liu et al. 2019) w/ some tweaks in tensorflow.

Super Resolution

Deep Residual Channel Attention Network (Stars 40+) Sep 2018

Implement a RCAN model in tensorflow.

Enhanced Super Resolution GAN (Stars 30+) Jun 2019

Implement an ESRGAN model in tensorflow.

Natural and Realistic SISR w/ Explicit NMD (Stars 5+) Apr 2020

Implement a NatSR model in pytorch.

NLP

Improved TextCNN (Stars 4+) Dec 2018

Implement an improved TextCNN model (Kim et al. 2020)

Text Tagging Dec 2018

Implement a text category classifier in tensorflow.

R.L

Rosetta Stone (Stars 560+) Sep 2018-

Hearthstone simulator using C++ w/ some R.L.
I contributed to the project by implementing `feature extractor` and `neural network’ in libtorch++.

Speech Synthesis

Tacotron Jan 2019

Implement a google tacotron speech synthesis in tensorflow.

Open Source Contributions

syzkaller :: New Generation of Linux Kernel Fuzzer
#575

simpletransformers :: Transformers made simple with training, evaluating, and prediction possible with one line each
#290

pytorch-image-models :: Pytorch image models, scripts, pretrained weights
#1058, #1069

deit :: DeiT Data-efficient Image Transformers
#140, #147, #148

MADGRAD :: MADGRAD Optimization Method
#11

tensorflow-image-models :: Tensorflow Image Models (tfimm) is a collection of image models with pretrained weights, obtained by porting architectures from timm to Tensorflow
#61