

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 3% **American Express – Default Prediction** (135 / 4875), 2022.
- 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

Toss core, Seoul, South Korea

Dec 2021 – Present

Data Scientist

- Personal CSS model for the CB.
 - Developed a more accurate & robust CSS model for more general targets like thin-filer, thick-filer.
 - Outperformed about **15%** (on the primary metric) compared with the previous method.

- Classify the category of the user review for the NPS (Net Performer Score) product.
 - Developed the RESTful API server to infer the deep learning model for the batch job.
 - Saved analysis time and labor of the NPS team a lot.
- OCR model to break captchas for the automation product.
 - Developed the lightweight models (text detector & captcha classifier) for inference in real-time (about **1000 TPS** for a batch transaction, **80 ~ 100 TPS** for a sample on CPU) and built the RESTful API server to serve the mode in real-time on the CPU.
 - In the A/B test, the **new captcha model outperforms** the Google Vision OCR
Accuracy (top-1): improved **50%p+** (45% to **95%**)
Latency (p95): reduced by **80x** (about >1000ms to **12ms**)
Revenue: reduced cost by about **\$7,000 ~ / year**
- User consumption forecasting model for *CDP product.
 - Developed the Transformer based sequential model that predicts what users will consume in next month.
 - Built an efficient pipeline to process and train lots of tabular data (about 500GB).
 - In the A/B test, a new model achieved...
- CSS model for BNPL (Buy Now Pay Later) service.
 - Developed **the CSS model** (default prediction), mainly targeted to the **thin filer**. The new model achieved the targeted **default rate of about 1%**.
 - Developed **the explainer** to describe which factors affect the rejection.
- Transaction category classification model to boost the advertisement.
 - Developed the **ads category classifier** that **increases revenue** in a roundabout way.
- Internal product, Slack bot that summarizes the long threads
 - Help people to understand the context quickly with minimum effort.
- Working as a full-time

% *CDP: Customer Data Platform. Lots of user segments generated by machine learning models.

Watcha, Seoul, South Korea

Jun 2020 – Dec 2021

Machine Learning Researcher

- **Watcha recommendation system** to offer a better user experience and increase paid conversion.
 - Developed the advanced training recipe & architecture to improve training stability and offline performance. Also, worked on post-processing to

recommend unseen content to users. In the A/B test, the new model boosts the **Click ratio online metric** by **about 1.01%+**.

- Developed the network to capture the time the user watches while the augmentations bring the training stability and performance gain. In the A/B test, **the new model wins the online metrics** by the followings. (compared with [Div2Vec](#) and the new model, the previous deep learning model beats the current new model)

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

***Viewing minutes** (median): improved **1.015%+**

- Developed the sequential recommendation architecture to recommend what content to watch next. **It achieved SOTA performance** compared to the previous SOTA architecture like BERT4Rec. In the A/B test, **the new model outperforms** by the following metrics.

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+**

- Face recognition model to find actors from the poster & still-cut images for the Watcha Pedia product.
 - Developed the pipeline to identify & recognizing actor faces from the images with the face detection & identification deep learning models (similarity-based searching).
 - Built a daily job that runs on the CPU. Also, optimize CPU-intensive operations to run fast.
- The internal product, to predict expected users' view-time of the content.
 - Before importing the content, the model offers an insight into the valuation of the content like expected view-time affecting the cost of the content.
- The internal product that helps the designer's work
 - Developed the image super-resolution model to upscale the image more accurately and faster than the public methods (e.g., waifu).
- Watcha Music sequential recommendation system (prototype).
- Worked as a full-time

% ***Viewing days**: how many days users are active on the app each month.

% ***Viewing minutes**: how many minutes the user watched the content.

Machine Learning Engineer

- Transaction category classification application to identify the category for the convenience of user experience.
 - Developed the lightweight transaction category classification model. In the A/B test, the new model **achieved 25 ~ 30%p+ *Accuracy improvement**.
 - Developed the backends (e.g., model serving, business logic microservices) in Python.
 1. Utilized inference-aware framework (ONNX) to achieve stable and low latency.
 2. Achieved a target latency of about 7 ~ 10 TPS (p50) while handling 1M transactions/day (1 transaction = 100 samples).
- CSS model to forecast the possibility of loan overdue.
- Worked as a full-time

% ***Accuracy**: how many users don't update their transactions' category.

VoyagerX, Seoul, South Korea

Jan 2019 – Sep 2019

Machine Learning Engineer

- 'Proceedings' deep learning application which automatically recognizes speakers & speeches (speaker diarization).
 - Developed the backend to diarize the conversation.
 - Developed the lightweight speaker verification model (served at AWS Lambda)
 - Developed the on/offline speaker diarization based on the clustering & E2E methods
- 'Hair Salon' project to swap the hair with what the user wants naturally.
 - Developed a hair/face image segmentation model to identify segments accurately.
 - Developed image in-painting model to detach a hair.
 - Developed an I2I translation model to change the hairstyle.
- Worked as an intern

ELCID, Pangyo, Korea

Jun 2016 - Aug 2016

Penetration Tester

- Penetrated the network firewall and anti-virus products.
- 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 sizes 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 730+)

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 70+)

Aug 2018

Implement lots of image semantic segmentation and ordered the papers.

Optimizer

pytorch-optimizer (Stars 65+)

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](#)