Dalei Yang

Location: Stockholm
Mobile +46-724451057
daleiyang@gmail.com

Overview of capabilities

- Worked in NLP, backend, data engineer, DBA and software testing. Have a certain understanding of recommender systems and LLM.
- Worked as a **Research & Development Engineer** at Trio.ai, a startup focusing on NLP, for **3 years**.(**Employment Certificate**) (**English Translation**).

  Worked at **Microsoft** for **5.6 years**, last position was **SDE 2**, focused on C# backend, data engineer.(**Employment Certificate**)(**English Translation**)

  Worked at adSage as a **DBA** and **data engineer** for **5.5 years** and as a **software testing manager** for 1 year.
- Worked in Seattle and Los Angeles for adSage for 21 months.
- In 2016, spent 8 months of spare time studying "ACM/ICPC" algorithmic contest topics. Solved 229 ICPC regional and world finals problems.

  See "**ACM/ICPC Algorithm Contests Training**" in "Appendix" for details.

Technical capabilities

- C, C++, python, Elasticsearch, C#, Azure.
- NLP (Segmentation, Pos Tagging, NER, etc.), NumPy, Pandas, TensorFlow, PyTorch, faiss.
- SQL, ETL(Informatica,SSIS,etc.), Data Warehouse, DBA (query/index/stored procedures tuning).
- Demonstration of the basics knowledge of machine learning:
  1. **LR** mathematical principles derived by hand, numpy implementation. **source code**.
  2. **XGBoost**, understanding the mathematical principles, **tutorial**. Financial risk control project practice. **source code**.
  3. **Transformer**, follow the **tutorial** to understand "The Annotated Transformer " **source code**.
- Learn best practices for "recommender system" in industry from 2024.03. See "**Recommender Systems**" in "Appendix" for details.
- Learn Large Language Model from 2024.10 See "**Large Language Model**" in "Appendix" for details.

Project experience

Trio.ai User Intent Recognition - Latest News Recommendation 2019.11 – 2020.06

Background Analyze the user's query and push relevant latest news when the query is found to have "news intent".

My outputs

1. Regulary crawl latest news from third parties and use word frequency, tfidf and text rank to extract keywords. **crawler.py**,line 359~395
2. Use search engine search results to determine whether these keywords or their combinations have "news intent". **crawler.py**,line 408~460
3. Write a customized **Trie** (prefix tree) to match keywords and obtain relevant news. Refer to lines 62~210 of **Storage.h**.

Trio.ai Text Classification - Product Recommendation 2019.07 – 2019.10

Background Classify queries into four categories: "mobile phones, accessories, operators, and others" and recommend related products.

My outputs

1. Crawl product information from JD.com. Train a **Character-level CNN** model to infer product category. **model script**, **training script**.
2. Create a c++ **wrapper** for the **tensorflow c api** that provides interfaces for loading models (**::InitWithCKPT**) and inference (**::Run**).
3. Online service load tf model, **line 50~67** in Storage.cc; Online inference, **line 299~316** and **line 513~579** in Storage.cc.

Trio.ai Chinese Spelling Correction - team research project 2019.02 – 2019.03

Background Try different structures of **Transformer**s to correct errors in queries caused by characters with the same pronunciation but different shapes.

Team outputs

1. Crawl news from reliable sources to generate positive sample. Refer to **raw data**.
2. According to the statistical results, generate negative sample from positive sample. Refer to **negative generate script**, **train_data**.
3. V1. Baseline model: write a HMM model in Python, calculate parameters offline, and use the Viterbi for predictions.
4. V2. Bi-directional transformer, Refer to **lm_net.py** line 67~97, **modules.py** line 192~280.
5. V3. Negtive as Q, positive as K and V, apple corresponding mask. Refer to **network.py** line 89~105, **modules.py** line 282~370.
6. V4. Implement a **Cloze-driven bi-directional transformer**, Refer to **network.py** line 65~117, modules.py **line 254**, **line 160**.
7. Single layer V3 model with 240,000 steps trainning perform best,accuray 0.74, recall 0.83. VS Baidu API, accuray 0.80, recall 0.37.

My outputs Provide the V1 baseline model within one day and participate in the discussion and implementation of the V2, V3, and V4 models.

Trio.ai Sentence Similarity Calculation Engine 2017.09 – 2019.06

Background Calculate the similarity between the query and all entries in the knowledge base.

e.g. Query: "What is the interest rate on a one-year deposit?". Best Match in knowledge base: " The interest rate on a one-year deposit is 3.5%."

My outputs 1. Read through google w2v source code. Found suitable training parameters to train word embeddings on a 40G generalized corpus.

See "w2v Principle and Source Code Analysis" in the Appendix for details.

2. The similarity of the two sets of word embeddings was scored using a greedy algorithm.

See "Short Sentence Similarity Calculation Engine" in the Appendix for details.

3. Implement a memory pool to stabilize the memory usage of the Calculation engine.

See "Implement a Memory Pool in C++" in the Appendix for details.

End-user 1. Become the engine of the QA module of Trio.ai chatbot. Customers include: China Mobile, etc.

2. Other project teams at Trio.ai, such as the "Chat Module" and the "Security Module", also use it to solve their problems.

Trio.ai In-memory data update mechanism for online services 2018.06

Background It is **unacceptable** to update the data in memory of an online service by restarting the service.

My outputs 1. Use two mutex locks and two read/write locks to control the switching between the old and new memory.

2. The service can still access the data in memory when switching between the old and new memory.

3. Packaged into a Apache thrift service for easy reuse and was widely adopted by other online services project team in Trio.ai.

Highlights **See "In-memory data update mechanism for online services" in the Appendix for details.**

Microsoft OS and Devices Division - Refactoring MS short link services 2015.05 – 2016.01

Background Refactoring the Microsoft Short Links service (https://aka.ms, https://go.microsoft.com/fwlink) and migrate it from on-premises to Azure.

My outputs 1. Design and implement of high throughput **lock-free hash tables** as in-memory database.

See "Lock-free Hash Tables" in the Appendix for details.

2. Locate performance bottlenecks in IIS, recommend Nginx + lua instead. Reduce the number of servers and save operating costs.

3. Attach "Architecture Design Demo" for reference.

Microsoft OS and Devices Division - Ddistributed data synchronization 2015.11 – 2016.02

Background 1. Designing a cross-data center data synchronization mechanism for the content management platform of www.msn.com

2. The existing Azure service has high latency and cannot meet the requirement.

My outputs 1. Read the Paxos papers systematically, understand the ideas.

2. Referring to Liskov's paper and implementation demo, I propose a method to synchronize data across data centers.

See "Paxos Algorithm Study" in the "Appendix" for details.

Working experience

Learning best practices for recommender system from 2024.03 and LLM from 2024.10.

Trio.ai NLP engineer 2017.04 – 2020.06

Microsoft SDE 2 2011.09 – 2017.04

adSage DBA 2006.02 – 2011.08

Duties **1**. Design and implement data projects architecture. **2**. Design and implement efficient ETL processes. **3**. Stored Procedures / Query / Index tuning.

adSage Software testing manager 2006.02 – 2007.02

Duties **1**. Built and managed test teams of up to 24 people for Microsoft outsourcing projects.

adSage Microsoft Ad Lab data program 2006.02 – 2011.08

Duties **1**. Design, implement and maintain different data, ETL projects.

2. Worked in Seattle and LA for 21 months, coordinated the work of the technical teams in Beijing and the United States.

Personal business ERP system for Liu Ning Piano School 2005.05 – 2006.01

Beijing Taihao Qiren Software Technology Technical support 2003.04 – 2005.04

Beijing Shenzhou Long'an Technology Website maintenance, development 2001.11 – 2003.03

Education

Xi'an University of Posts & Telecommunications (In China)

Computer Science Department bachelor degree **1997.09 – 2001.06**

Appendix

ACM/ICPC algorithm contests training

2016.05 – 2017.01

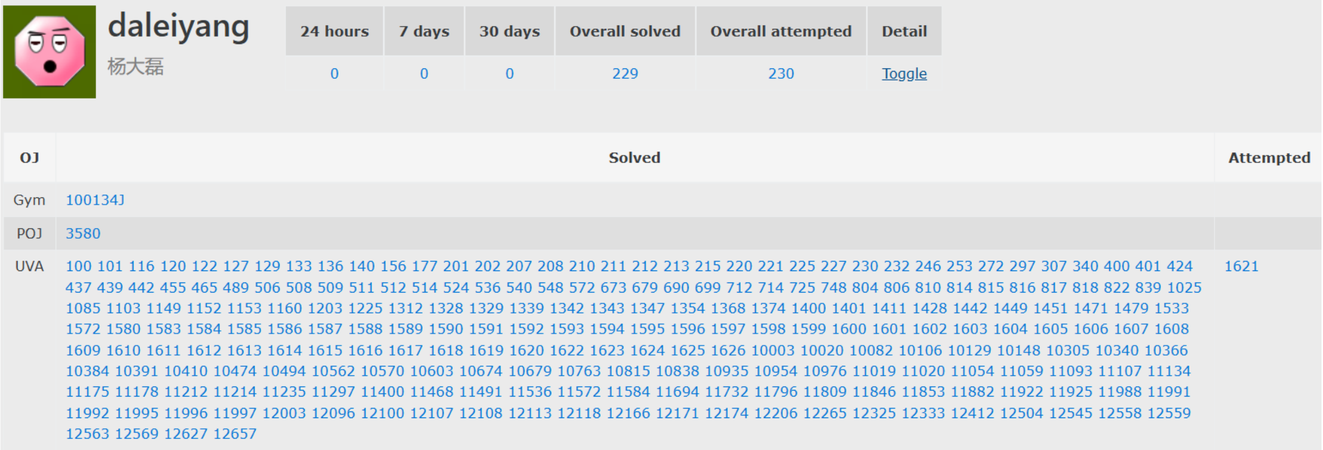
Background Looking to change career paths to algorithms.

My outputs

1. The training process took eight months of spare time.
2. AC (Accepted) 229 ICPC regional and world finals problems.
3. Most of the AC code is based on my own thinking and having experienced many failed submissions.
4. For some of the topics that were too difficult, I referred to the code of the masters, understood it, and then submitted it.

Resource

1. Follow this book to train: "AOAPC II: Beginning Algorithm Contests (Second Edition) (Rujia Liu)"
2. The code was submitted at ***Virtual Judge*** https://vjudge.net/user/daleiyang#

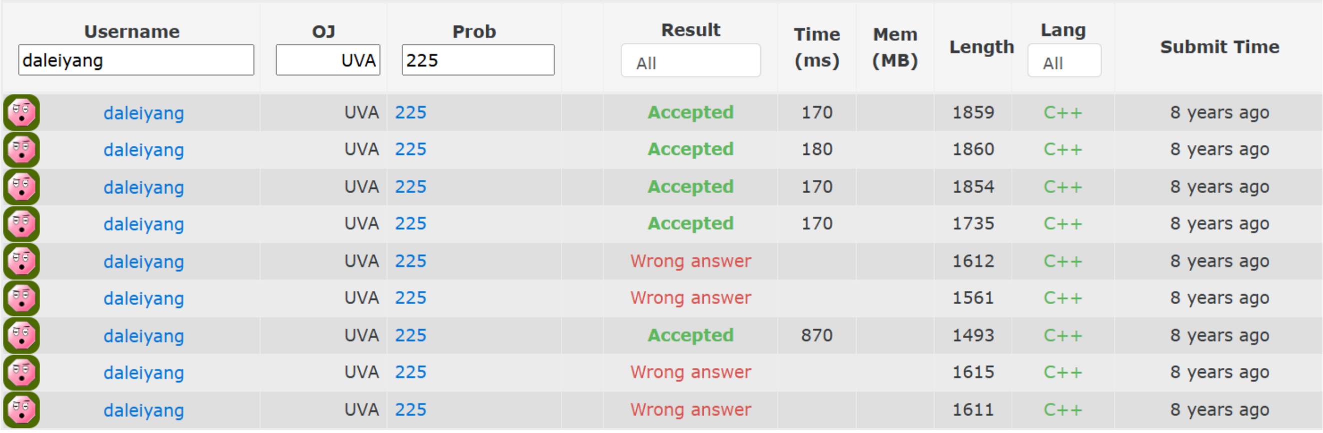


daleiyang
杨大磊

24 hours	7 days	30 days	Overall solved	Overall attempted	Detail
0	0	0	229	230	Toggle

OJ	Solved	Attempted
Gym	100134J	
POJ	3580	
UVA	100 101 116 120 122 127 129 133 136 140 156 177 201 202 207 208 210 211 212 213 215 220 221 225 227 230 232 246 253 272 297 307 340 400 401 424 437 439 442 455 465 489 506 508 509 511 512 514 524 536 540 548 572 673 679 690 699 712 714 725 748 804 806 810 814 815 816 817 818 822 839 1025 1085 1103 1149 1152 1153 1160 1203 1225 1312 1328 1329 1339 1342 1343 1347 1354 1368 1374 1400 1401 1411 1428 1442 1449 1451 1471 1479 1533 1572 1580 1583 1584 1585 1586 1587 1588 1589 1590 1591 1592 1593 1594 1595 1596 1597 1598 1599 1600 1601 1602 1603 1604 1605 1606 1607 1608 1609 1610 1611 1612 1613 1614 1615 1616 1617 1618 1619 1620 1622 1623 1624 1625 1626 10003 10020 10082 10106 10129 10148 10305 10340 10366 10384 10391 10410 10474 10494 10562 10570 10603 10674 10679 10763 10815 10838 10935 10954 10976 11019 11020 11054 11059 11093 11107 11134 11175 11178 11212 11214 11235 11297 11400 11468 11491 11536 11572 11584 11694 11732 11796 11809 11846 11853 11882 11922 11925 11988 11991 11992 11995 11996 11997 12003 12096 12100 12107 12108 12113 12118 12166 12171 12174 12206 12265 12325 12333 12412 12504 12545 12558 12559 12563 12569 12627 12657	1621

You can see the submission process and source code for each problem.



Username	OJ	Prob	Result	Time (ms)	Mem (MB)	Length	Lang	Submit Time
daleiyang	UVA	225	All				All	
daleiyang	UVA	225	Accepted	170		1859	C++	8 years ago
daleiyang	UVA	225	Accepted	180		1860	C++	8 years ago
daleiyang	UVA	225	Accepted	170		1854	C++	8 years ago
daleiyang	UVA	225	Accepted	170		1735	C++	8 years ago
daleiyang	UVA	225	Wrong answer			1612	C++	8 years ago
daleiyang	UVA	225	Wrong answer			1561	C++	8 years ago
daleiyang	UVA	225	Accepted	870		1493	C++	8 years ago
daleiyang	UVA	225	Wrong answer			1615	C++	8 years ago
daleiyang	UVA	225	Wrong answer			1611	C++	8 years ago

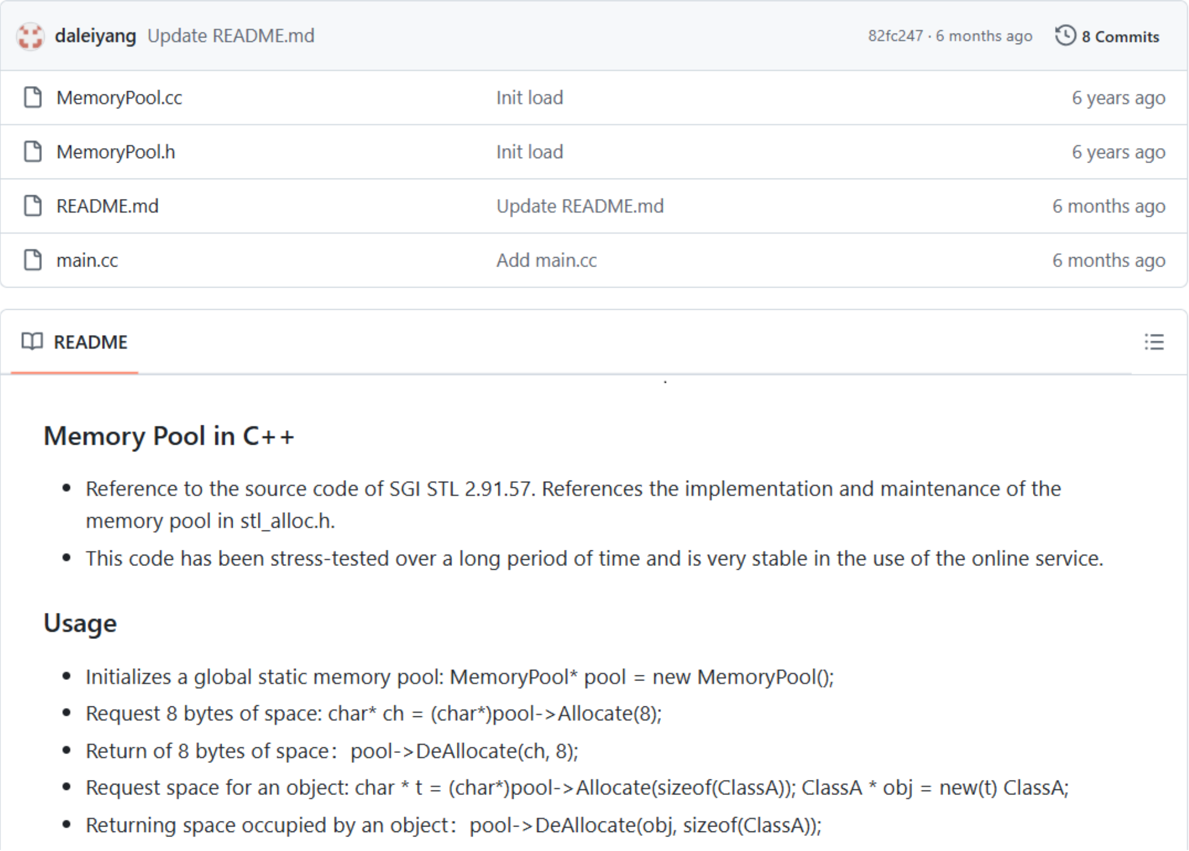
Implement a Memory Pool in C++ 2018.05

Background In “China Mobile” production environment, the "Similarity Engine" needs to satisfy a high RPS. Sometimes memory usage is unstable and debugging is difficult. So, I decided to implement my own memory pool.

My outputs 1. Simplified and rewritten the memory pool implementation method of "stl_alloc.h" in SGI STL v 2.91.57.

2. The "Similarity Engine" using this memory pool became very stable in “China Mobile” production environment.

Resource Source code and instructions https://github.com/daleiyang/MemoryPool



daleiyang Update README.md 82fc247 · 6 months ago 8 Commits

MemoryPool.cc	Init load	6 years ago
MemoryPool.h	Init load	6 years ago
README.md	Update README.md	6 months ago
main.cc	Add main.cc	6 months ago

README

Memory Pool in C++

- Reference to the source code of SGI STL 2.91.57. References the implementation and maintenance of the memory pool in stl_alloc.h.
- This code has been stress-tested over a long period of time and is very stable in the use of the online service.

Usage

- Initializes a global static memory pool: MemoryPool* pool = new MemoryPool();
- Request 8 bytes of space: char* ch = (char*)pool->Allocate(8);
- Return of 8 bytes of space： pool->DeAllocate(ch, 8);
- Request space for an object: char * t = (char*)pool->Allocate(sizeof(ClassA)); ClassA * obj = new(t) ClassA;
- Returning space occupied by an object： pool->DeAllocate(obj, sizeof(ClassA));

w2v Principle and Source Code Analysis 2017.09

Background Analyze the principle and source code of google's w2v to design and implement the "Sentence Similarity Computation Engine".

Resource Follow the tutorial "Mathematical priciples in word2vec" and study the source code "word2vec.c".

What I learned from the "word2vec.c" and the tutorial:

1. **word2vec.c** is a short 700 lines of code, but the depth of the programing skill is breathtaking. The author, Tomas Mikolov, is supposed to be an algorithm contest player, judging by his code style. Programming skills that can be learned from this code are: Handling of input parameters, Accelerate sigmoid function calculations with a segmented lookup table, Weighted sampling, Tips for reading files, Dynamically control the size of the hash table, Building Huffman trees, Splitting files by thread, Random number generation methods, Randomizing window size, Adaptive learning rate.

2. This **tutorial** explains in detail the machine learning principles used in this code: Logistic regression, language models, optimization goals, formula derivation, gradient update, a variety of detailed techniques are available, formula derivation and coding consistency is very high. Understand w2v, read this turorial is enough.

Short Sentence Similarity Calculation Engine

2017.09 – 2019.06

Resource For implementation details, see the source code on GitHub. https://github.com/daleiyang/Similarity

In-memory data update mechanism for online services

2018.06

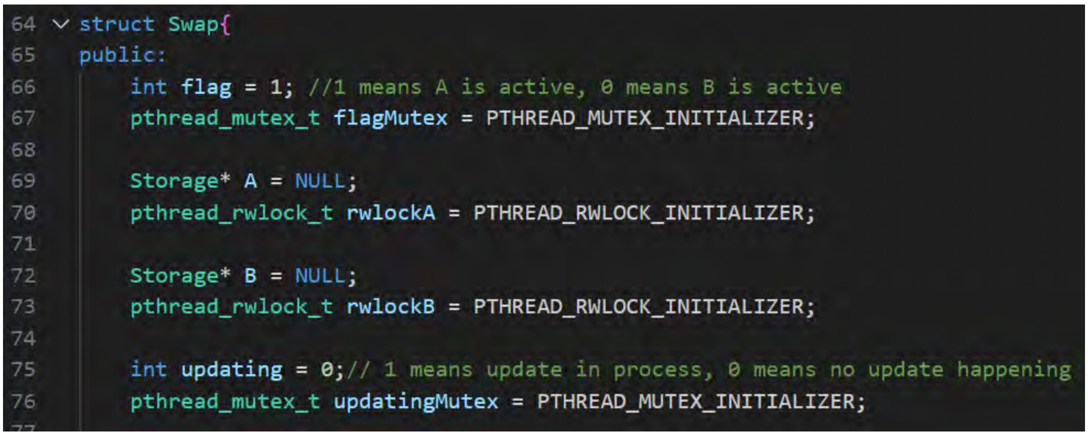
Background Online services need to be able to reload data in memory while still being able to access it.

My outputs

1. Accurately control the use of all types of "locks" to ensure correct coding.
2. Minimize the impact on user queries during memory swapping.
3. Memory updates can be triggered by timing or manually.
4. Packaged into a generic Apache thrift service template and widely adopted by other online services project team in Trio.ai.
5. At an internal sharing meeting, a former Baidu.com engineer found that my implementation was almost the same as the principle of Baidu.com (The No. 1 search engine in China) online service memory swap. It's really "heroes see eye to eye" :)

Resource

1. See the source code and comments for more information on the memory swapping process and minimizing the impact on user queries. GitHub: https://github.com/daleiyang/MemorySwap
2. Code example: The Swap structure controls the memory switching process and contains two mutex locks and two read/write locks.



```
struct Swap{
public:
    int flag = 1; //1 means A is active, 0 means B is active
    pthread_mutex_t flagMutex = PTHREAD_MUTEX_INITIALIZER;

    Storage* A = NULL;
    pthread_rwlock_t rwlockA = PTHREAD_RWLOCK_INITIALIZER;

    Storage* B = NULL;
    pthread_rwlock_t rwlockB = PTHREAD_RWLOCK_INITIALIZER;

    int updating = 0;// 1 means update in process, 0 means no update happening
    pthread_mutex_t updatingMutex = PTHREAD_MUTEX_INITIALIZER;
```

Lock-Free Hash Table

2015.08

Background Saw an article outlining the core data structures and algorithms used in the Shanghai Stock Exchange's securities trading system. During the bull market in 2015, this solution smoothly supported **daily** trading volumes of over **one trillion** chinese Yuan.

My outputs

1. Implemented it in C# and applied it to a pre-production environment for the Microsoft short link service.
2. Ensure that the logic of the code based on the **Compare-And-Swap** API (**CAS primitive**) is correct.

Resource Design details, source code, and test methods can be found on my GitHub: https://github.com/daleiyang/LockFreeHashTable

Performance

1. With 3 million key-value pairs, 30 processes in parallel to read, update and delete operations, 17-minute stress test.

	Lock-Free Hash Table	**.Net Concurrent Dictionary**	**Comparison**
Operation	operations per second	operations per second	
Get	7,113,400	1,681,929	4.2X
Add/Update	8,927,004	240,321	37.1X
Delete	13,566,043	245,884	55.2X

2. **See "Perf testing report" for performance testing report.**

Paxos Algorithm Study 2015.11 – 2016.02

Background Designing a cross-data center data synchronization mechanism for the content management platform of www.msn.com, the existing Microsoft cloud service has high latency and cannot meet the requirement.

My outputs

1. Reading papers on related topics eventually revealed that adopting the approach described below was a viable path:
   2.1 Miguel Castro from MSR Practical Byzantine Fault Tolerance
   2.2 Barbara Liskov from MIT BFT - Practical Byzantine Fault Tolerance
   2.3 Google engineering team Paxos Made Live - An Engineering Perspective
2. Drafting system design documentation. Highly-Available Distributed In-Memory Cache with Byzantine Paxos
3. Drafting resource estimation report. Replication with Byzantine Paxos - Current Status and Estimation
4. The project eventually failed to get off the ground due to the lack of a skilled engineering team and inadequate budget :)

Recommender System 2024.03 –

My outputs

1. Characteristics of recommendation systems (tutorial). Common issues and methods of **feature engineering** (tutorial).
2. **Embedding** and **Parameter Server**. ps-lite source code analysis. Distributed LR With ps-lite (xflow) .
3. **Fine-grained Ranking (**tutorial)
   6.1 **FM** formula derivation (turotial). alphaFM, multi-thread implementation of FM with FTRL （source code).
   6.2 Handcrafting Wide & Deep with NumPy and FTRL online learning optimizer. tutorial. source code.
   6.3 DeepFM with multi-valued, sparse, shared weight support using TensorFlow. tutorial. source code.
   6.4 Long time and short User Interest Embedding with Deep Interest Network and Search-based Interest Model.
4. **Recall (**tutorial)
   7.1 Inverted index. Item CF. MF. Merge results from different recall methods.
   7.2 Define positive sample in I2I, U2I, U2U2I scenarios, Key is random negative sampling.
   7.3 Understand NEC loss, NEG loss, Sampled softmax loss, Pairwise loss.
   ......*(Summary not completed)*
5. **Pre-ranking** and **Re-ranking** (Not started yet)
6. **Multi-Task** and **Multi-scenario** (Not started yet)
7. **Cold start** (Not started yet)
8. **Evaluation** and **Debug** (Not started yet)
9. Interview FAQ

Large Language Model 2024.10 –

A summary of the information to get an overall picture of LLM. **Highlight** the article I have already read.

Overview

State of GPT

Current Best Practices for Training LLMs from Scratch

An Overview on Language Models: Recent Developments and Outlook

Harnessing the Power of LLMs in Practice: A Survey on ChatGPT...

Cutting Edge Model

GPT-4 Technical Report

Spark of AGI

PALM2 Technical Report

Pretraining - Overall

Language Models are Few-Shot Learners

LLaMA: Open and Efficient Foundation Language Models

BloombergGPT: A Large Language Model for Finance

Transformer Math 101

Pretraining - Data

Processing Data for Large Language Models

The BigScience ROOTS Corpus: A 1.6TB Composite...

The RefinedWeb Dataset for Falcon LLM:...

Scaling Data-Constrained Language Models

A Pretrainer's Guide to Training Data: Measuring the...

Pretraining - Training

Using DeepSpeed and Megatron to Train Megatron-Turing...

ZeRO Memory Optimizations Toward Training Trillion...

Performance and Scalability: How To Fit a Bigger Model and ...

How to Train Really Large Models on Many GPUs?

Tokenization

Normalization and pre-tokenization

Byte-Pair Encoding tokenization

Building a tokenizer, block by block

Supervised Fine Tuning

Finetuned Language Models Are Zero-Shot Learners

Exploring the Impact of Instruction Data Scaling on LLM

Towards Better Instruction Following Language Models for Chinese

Instruction Tuning with GPT-4

LIMA: Less Is More for Alignment

RLHF

Fine-Tuning Language Models from Human Preferences

Learning to summarize from human feedback

Recursively Summarizing Books with Human Feedback

WebGPT: Browser-assisted question-answering with human feedback

Training language models to follow instructions with human feedback

Prompt Engineering

Chain-of-Thought Prompting Elicits Reasoning in LLM

Self-Consistency Improves Chain of Thought Reasoning in LM

Tree of Thoughts: Deliberate Problem Solving with LLM

Reflexion: Language Agents with Verbal Reinforcement Learning

AutoGPT

Generative Agents: Interactive Simulacra of Human Behavior

Large Language Models as Tool Makers

Tools

Toolformer: Language Models Can Teach Themselves to Use Tools

TaskMatrix.AI: Completing Tasks by Connecting Foundation Mod...

REPLUG: Retrieval-Augmented Black-Box Language Models

Code Related

Evaluating Large Language Models Trained on Code

Math Related

Training Verifiers to Solve Math Word Problems

Loss Function

Efficient Training of Language Models to Fill in the Middle

UL2: Unifying Language Learning Paradigms

Optional

Building Systems with the ChatGPT API

OpenAI CookBook

Microsoft | 微软(中国)有限公司

离职证明

兹证明微软（中国）有限公司于 2017 年 4 月 17 日与杨大磊（身份证号：11010819780908491X）终止了劳动关系。

其于 2011 年 9 月 1 日正式入职微软（中国）有限公司, 签署无固定期劳动合同。其在我公司的最后工作日为 2017 年 4 月 17 日, 离职前担任 SOFTWARE ENGINEER 2 职位, 在我公司的工作年限为 5.6 年。

微软(中国)有限公司
人力资源部
2017 年 4 月 17 日

MICROSOFT (CHINA) CO. LTD.
微软(中国)有限公司人事专用章

Microsoft MICROSOFT (CHINA) CO., LTD.

Employment Separation Certificate

This is to certify that Microsoft (China) Co., Ltd. terminated its labor relations with Yang Dalei (ID card number: 11010819780908491X) in April 17, 2017.

He formally joined Microsoft (China) Co., Ltd. on September 1, 2011, and signed a non-fixed-term labor contract. His last working day in our company is April 17, 2017. He held the position of SOFTWARE ENGINEER 2 before leaving his job and has worked in our company for 5.6 years.

LANGUAGES HUB TRANSLATION SERVICE (NANTONG) CO., LTD.
吉栢翻译服务（南通）有限公司
翻译专用章
CERTIFIED TRANSLATION
3206020047448

LANGUAGES HUB TRANSLATION SERVICE (NANTONG) CO., LTD.
吉栢翻译服务（南通）有限公司
3206020122166

Human Resources Department

Microsoft (China) Co., Ltd.

April 17, 2017

Seal: Special Seal for Human Resources Department of Microsoft (China) Co., Ltd. (sealed)

I confirm it is an accurate translation of the original document
Translator: Zhu Xiunan Signature: Zhu Xiunan
Qualification: TEM 8 (TEST for English Major-Band 8)
Certificate No.: EVIII 1710021188
Company: Languages Hub Translation Service (Nantong) Co., Ltd.
Add: No 83, Chongchuan Road, Nantong City, Jiangsu, China
Private Number: +8615862748936 Dated on: 2021.8.1

trio.ai 三角兽（北京）科技有限公司

离职证明

兹证明 杨大磊 先生（身份证号：11010819780908491X ）自 2017 年 04 月 18 起加入三角兽（北京）科技有限公司 技术中心 部门 研发工程师 职务，至 2020 年 06 月 30 日，与本公司解除劳动关系，双方已办理完成全部离职手续。

该人员在离职后，无需履行竞业限制业务；未经我司书面许可，不得向任何单位和个人透露我司商业秘密和其他经营秘密。

三角兽（北京）科技有限公司

离职人员签字：杨大磊

2020 年 06 月 30 日

兽（北京）科技有限公司

1101051021

兽（北京）科技有限公

trio.ai Tendrillion (Beijing) Science and Technology Co., Ltd.

Employment Separation Certificate

This is to certify that Mr. Yang Dalei (ID No.: 11010819780908491X) has joined the Technology Center of Tendrillion (Beijing) Science and Technology Co., Ltd. since April 18, 2017 as a Research and Development Engineer, and discharged the employment relationship with the Company on June 30, 2020. Both parties have completed all resignation procedures.

After leaving office, the person is not required to perform non-competition business; Without the written permission of our Company, the person shall not disclose our business secrets and other business information to any organization or individual.

Tendrillion (Beijing) Science and Technology Co., Ltd.

Signature of separating employee: **Yang Dalei (signature)**

June 30, 2020

Tendrillion (Beijing) Science and Technology Co., Ltd. (seal)

Tendrillion (Beijing) Science and Technology Co., Ltd. (seal)

LANGUAGES HUB TRANSLATION SERVICE(NANTONG) CO.,LTD.
吉航翻译服务（南通）有限公司
翻译专用章
CERTIFIED TRANSLATION
3206020047448

I confirm it is an accurate translation of the original document
Translator: Zhu Xiunan Signature: Zhu Xiunan
Qualification: TEM 8 (TEST for English Major-Band 8)
Certificate No.: EVIII 1710021188
Company: Languages Hub Translation Service (Nantong) Co., Ltd.
Add: No 83, Chongchuan Road, Nantong City, Jiangsu, China
Private Number: +8615862748936 Dated on: 2021.8.3