

## 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, NumPy, Pandas, TensorFlow, PyTorch, Transformer, Bert, NLP (Segmentation, Pos Tagging, NER), Faiss.
- Docker, Kubernetes, Redis, Elasticsearch.
- C#, .Net, Azure, javascript, jQuery, AngularJS, MVVM, RESTful API, HTML, CSS, VB, VB Script, ASP.
- SQL, ETL(Informatica,SSIS,etc.), Data Warehouse, Data modeling, DBA skills: tuning queries, indexes, and stored procedures.
- 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

<b>Trio.ai</b>	<b>User Intent Recognition - Latest News Recommendation</b>	<b>2019.11 – 2020.06</b>
<i>Background</i>	Analyze the user's query and push relevant latest news when the query is found to have "news intent".	
<i>My outputs</i>	1. Regularly crawl latest news from third parties and use word frequency, tfidf and text rank to extract keywords. <a href="#">crawler.py</a> , line 359~395 2. Use search engine search results to determine whether these keywords or their combinations have "news intent". <a href="#">crawler.py</a> , line 408~460 3. Write a customized <b>Trie</b> (prefix tree) to match keywords and obtain relevant news. Refer to lines 62~210 of <a href="#">Storage.h</a> .	
<b>Trio.ai</b>	<b>Text Classification - Product Recommendation</b>	<b>2019.07 – 2019.10</b>
<i>Background</i>	Classify queries into four categories: "mobile phones, accessories, operators, and others" and recommend related products.	
<i>My outputs</i>	1. Crawl product information from <a href="#">JD.com</a> . Train a <a href="#">Character-level CNN</a> model to infer product category. <a href="#">model script</a> , <a href="#">training script</a> . 2. Create a c++ <a href="#">wrapper</a> for the <a href="#">tensorflow c api</a> that provides interfaces for loading models ( <a href="#">::InitWithCKPT</a> ) and inference ( <a href="#">::Run</a> ). 3. Online service load tf model, <a href="#">line 50~67</a> in Storage.cc; Online inference, <a href="#">line 299~316</a> and <a href="#">line 513~579</a> in Storage.cc.	
<b>Trio.ai</b>	<b>Chinese Spelling Correction - team research project</b>	<b>2019.02 – 2019.03</b>
<i>Background</i>	Try different structures of <b>Transformers</b> to correct errors in queries caused by characters with the same pronunciation but different shapes.	
<i>Team outputs</i>	1. Crawl news from reliable sources to generate positive sample. Refer to <a href="#">raw data</a> . 2. According to the statistical results, generate negative sample from positive sample. Refer to <a href="#">negative generate script</a> , <a href="#">train data</a> . 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 <a href="#">lm_net.py</a> line 67~97, <a href="#">modules.py</a> line 192~280. 5. V3. Negative as Q, positive as K and V, apply corresponding mask. Refer to <a href="#">network.py</a> line 89~105, <a href="#">modules.py</a> line 282~370. 6. V4. Implement a <a href="#">Cloze-driven bi-directional transformer</a> , Refer to <a href="#">network.py</a> line 65~117, modules.py <a href="#">line 254</a> , <a href="#">line 160</a> . 7. Single layer V3 model with 240,000 steps training perform best, accuracy 0.74, recall 0.83. VS Baidu API, accuracy 0.80, recall 0.37.	
<i>My outputs</i>	Provide the V1 baseline model within one day and participate in the discussion and implementation of the V2, V3, and V4 models.	

<b>Trio.ai</b>	<b>Sentence Similarity Calculation Engine</b>	<b>2017.09 – 2019.06</b>
<b>Background</b>	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%."	
<b>My outputs</b>	<ol style="list-style-type: none"> <li>1. Read through google w2v source code. Found suitable training parameters to train word embeddings on a 40G generalized corpus. <b>See "w2v Principle and Source Code Analysis" in the Appendix for details.</b></li> <li>2. The similarity of the two sets of word embeddings was scored using a greedy algorithm. <b>See "Short Sentence Similarity Calculation Engine" in the Appendix for details.</b></li> <li>3. Implement a memory pool to stabilize the memory usage of the Calculation engine. <b>See "Implement a Memory Pool in C++" in the Appendix for details.</b></li> </ol>	
<b>End-user</b>	<ol style="list-style-type: none"> <li>1. Become the engine of the QA module of Trio.ai chatbot. Customers include: China Mobile, etc.</li> <li>2. Other project teams at Trio.ai, such as the "Chat Module" and the "Security Module", also use it to solve their problems.</li> </ol>	

<b>Trio.ai</b>	<b>In-memory data update mechanism for online services</b>	<b>2018.06</b>
<b>Background</b>	It is <b>unacceptable</b> to update the data in memory of an online service by restarting the service.	
<b>My outputs</b>	<ol style="list-style-type: none"> <li>1. Use two mutex locks and two read/write locks to control the switching between the old and new memory.</li> <li>2. The service can still access the data in memory when switching between the old and new memory.</li> <li>3. Packaged into a Apache thrift service for easy reuse and was widely adopted by other online services project team in Trio.ai.</li> </ol>	
<b>Highlights</b>	<b>See "In-memory data update mechanism for online services" in the Appendix for details.</b>	
<b>Microsoft</b>	<b>OS and Devices Division - Refactoring MS short link services</b>	<b>2015.05 – 2016.01</b>
<b>Background</b>	Refactoring the Microsoft Short Links service ( <a href="https://aka.ms">https://aka.ms</a> , <a href="https://go.microsoft.com/fwlink">https://go.microsoft.com/fwlink</a> ) and migrate it from on-premises to Azure.	
<b>My outputs</b>	<ol style="list-style-type: none"> <li>1. Design and implement of high throughput <b>lock-free hash tables</b> as in-memory database. <b>See "Lock-free Hash Tables" in the Appendix for details.</b></li> <li>2. Locate performance bottlenecks in IIS, recommend Nginx + lua instead. Reduce the number of servers and save operating costs.</li> <li>3. Attach "<a href="#">Architecture Design Demo</a>" for reference.</li> </ol>	

<b>Microsoft</b>	<b>OS and Devices Division - Ddistributed data synchronization</b>	<b>2015.11 – 2016.02</b>
<b>Background</b>	<ol style="list-style-type: none"> <li>1. Designing a cross-data center data synchronization mechanism for the content management platform of <a href="http://www.msn.com">www.msn.com</a></li> <li>2. The existing Azure service has high latency and cannot meet the requirement.</li> </ol>	
<b>My outputs</b>	<ol style="list-style-type: none"> <li>1. Read the Paxos papers systematically, understand the ideas.</li> <li>2. Referring to Liskov's paper and implementation demo, I propose a method to synchronize data across data centers. <b>See "Paxos Algorithm Study" in the "Appendix" for details.</b></li> </ol>	

## Working experience

Learning best practices for [recommender system](#) from 2024.03 and [LLM](#) from 2024.10.

<b>Trio.ai</b>	<b>NLP engineer</b>	<b>2017.04 – 2020.06</b>
<b>Microsoft</b>	<b>SDE 2</b>	<b>2011.09 – 2017.04</b>
<b>adSage</b>	<b>DBA</b>	<b>2006.02 – 2011.08</b>
<b>Duties</b>	<ol style="list-style-type: none"> <li>1. Design and implement data projects architecture.</li> <li>2. Design and implement efficient ETL processes.</li> <li>3. Stored Procedures / Query / Index tuning.</li> </ol>	
<b>adSage</b>	<b>Software testing manager</b>	<b>2006.02 – 2007.02</b>
<b>Duties</b>	<ol style="list-style-type: none"> <li>1. Built and managed test teams of up to 24 people for Microsoft outsourcing projects.</li> </ol>	
<b>adSage</b>	<b>Microsoft Ad Lab data program</b>	<b>2006.02 – 2011.08</b>
<b>Duties</b>	<ol style="list-style-type: none"> <li>1. Design, implement and maintain different data, ETL projects.</li> <li>2. Worked in Seattle and LA for 21 months, coordinated the work of the technical teams in Beijing and the United States.</li> </ol>	

<b>Personal business</b>	<b>ERP system for Liu Ning Piano School</b>	<b>2005.05 – 2006.01</b>
<b>Beijing Taihao Qiren Software Technology</b>	<b>Technical support</b>	<b>2003.04 – 2005.04</b>
<b>Beijing Shenzhou Long'an Technology</b>	<b>Website maintenance, development</b>	<b>2001.11 – 2003.03</b>

## Education

Xi'an University of Posts & Telecommunications (In China)	
Computer Science Department	bachelor degree
	<b>1997.09 – 2001.06</b>

# Appendix

ACM/ICPC algorithm contests training.....	3
Implement a Memory Pool in C++.....	4
w2v Principle and Source Code Analysis .....	4
Short Sentence Similarity Calculation Engine.....	5
In-memory data update mechanism for online services .....	5
Lock-Free Hash Table.....	5
Paxos Algorithm Study .....	6
Recommender Systems.....	6
Large Language Model.....	7
Microsoft Separation Certificate .....	8
Microsoft Separation Certificate (English Translation).....	9
Trio.ai Separation Certificate .....	10
Trio.ai Separation Certificate (English Translation).....	11

## 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#>

 <b>daleiyang</b> 杨大磊		24 hours	7 days	30 days	Overall solved	Overall attempted	Detail
		0	0	0	229	230	<a href="#">Toggle</a>
OJ	Solved						Attempted
Gym	<a href="#">100134J</a>						
POJ	<a href="#">3580</a>						
UVA	<a href="#">100</a> <a href="#">101</a> <a href="#">116</a> <a href="#">120</a> <a href="#">122</a> <a href="#">127</a> <a href="#">129</a> <a href="#">133</a> <a href="#">136</a> <a href="#">140</a> <a href="#">156</a> <a href="#">177</a> <a href="#">201</a> <a href="#">202</a> <a href="#">207</a> <a href="#">208</a> <a href="#">210</a> <a href="#">211</a> <a href="#">212</a> <a href="#">213</a> <a href="#">215</a> <a href="#">220</a> <a href="#">221</a> <a href="#">225</a> <a href="#">227</a> <a href="#">230</a> <a href="#">232</a> <a href="#">246</a> <a href="#">253</a> <a href="#">272</a> <a href="#">297</a> <a href="#">307</a> <a href="#">340</a> <a href="#">400</a> <a href="#">401</a> <a href="#">424</a> <a href="#">437</a> <a href="#">439</a> <a href="#">442</a> <a href="#">455</a> <a href="#">465</a> <a href="#">489</a> <a href="#">506</a> <a href="#">508</a> <a href="#">509</a> <a href="#">511</a> <a href="#">512</a> <a href="#">514</a> <a href="#">524</a> <a href="#">536</a> <a href="#">540</a> <a href="#">548</a> <a href="#">572</a> <a href="#">673</a> <a href="#">679</a> <a href="#">690</a> <a href="#">699</a> <a href="#">712</a> <a href="#">714</a> <a href="#">725</a> <a href="#">748</a> <a href="#">804</a> <a href="#">806</a> <a href="#">810</a> <a href="#">814</a> <a href="#">815</a> <a href="#">816</a> <a href="#">817</a> <a href="#">818</a> <a href="#">822</a> <a href="#">839</a> <a href="#">1025</a> <a href="#">1085</a> <a href="#">1103</a> <a href="#">1149</a> <a href="#">1152</a> <a href="#">1153</a> <a href="#">1160</a> <a href="#">1203</a> <a href="#">1225</a> <a href="#">1312</a> <a href="#">1328</a> <a href="#">1329</a> <a href="#">1339</a> <a href="#">1342</a> <a href="#">1343</a> <a href="#">1347</a> <a href="#">1354</a> <a href="#">1368</a> <a href="#">1374</a> <a href="#">1400</a> <a href="#">1401</a> <a href="#">1411</a> <a href="#">1428</a> <a href="#">1442</a> <a href="#">1449</a> <a href="#">1451</a> <a href="#">1471</a> <a href="#">1479</a> <a href="#">1533</a> <a href="#">1572</a> <a href="#">1580</a> <a href="#">1583</a> <a href="#">1584</a> <a href="#">1585</a> <a href="#">1586</a> <a href="#">1587</a> <a href="#">1588</a> <a href="#">1589</a> <a href="#">1590</a> <a href="#">1591</a> <a href="#">1592</a> <a href="#">1593</a> <a href="#">1594</a> <a href="#">1595</a> <a href="#">1596</a> <a href="#">1597</a> <a href="#">1598</a> <a href="#">1599</a> <a href="#">1600</a> <a href="#">1601</a> <a href="#">1602</a> <a href="#">1603</a> <a href="#">1604</a> <a href="#">1605</a> <a href="#">1606</a> <a href="#">1607</a> <a href="#">1608</a> <a href="#">1609</a> <a href="#">1610</a> <a href="#">1611</a> <a href="#">1612</a> <a href="#">1613</a> <a href="#">1614</a> <a href="#">1615</a> <a href="#">1616</a> <a href="#">1617</a> <a href="#">1618</a> <a href="#">1619</a> <a href="#">1620</a> <a href="#">1622</a> <a href="#">1623</a> <a href="#">1624</a> <a href="#">1625</a> <a href="#">1626</a> <a href="#">10003</a> <a href="#">10020</a> <a href="#">10082</a> <a href="#">10106</a> <a href="#">10129</a> <a href="#">10148</a> <a href="#">10305</a> <a href="#">10340</a> <a href="#">10366</a> <a href="#">10384</a> <a href="#">10391</a> <a href="#">10410</a> <a href="#">10474</a> <a href="#">10494</a> <a href="#">10562</a> <a href="#">10570</a> <a href="#">10603</a> <a href="#">10674</a> <a href="#">10679</a> <a href="#">10763</a> <a href="#">10815</a> <a href="#">10838</a> <a href="#">10935</a> <a href="#">10954</a> <a href="#">10976</a> <a href="#">11019</a> <a href="#">11020</a> <a href="#">11054</a> <a href="#">11059</a> <a href="#">11093</a> <a href="#">11107</a> <a href="#">11134</a> <a href="#">11175</a> <a href="#">11178</a> <a href="#">11212</a> <a href="#">11214</a> <a href="#">11235</a> <a href="#">11297</a> <a href="#">11400</a> <a href="#">11468</a> <a href="#">11491</a> <a href="#">11536</a> <a href="#">11572</a> <a href="#">11584</a> <a href="#">11694</a> <a href="#">11732</a> <a href="#">11796</a> <a href="#">11809</a> <a href="#">11846</a> <a href="#">11853</a> <a href="#">11882</a> <a href="#">11922</a> <a href="#">11925</a> <a href="#">11988</a> <a href="#">11991</a> <a href="#">11992</a> <a href="#">11995</a> <a href="#">11996</a> <a href="#">11997</a> <a href="#">12003</a> <a href="#">12096</a> <a href="#">12100</a> <a href="#">12107</a> <a href="#">12108</a> <a href="#">12113</a> <a href="#">12118</a> <a href="#">12166</a> <a href="#">12171</a> <a href="#">12174</a> <a href="#">12206</a> <a href="#">12265</a> <a href="#">12325</a> <a href="#">12333</a> <a href="#">12412</a> <a href="#">12504</a> <a href="#">12545</a> <a href="#">12558</a> <a href="#">12559</a> <a href="#">12563</a> <a href="#">12569</a> <a href="#">12627</a> <a href="#">12657</a>						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 principles 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 programming 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 tutorial 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.

```
64 struct Swap{
65     public:
66         int flag = 1; //1 means A is active, 0 means B is active
67         pthread_mutex_t flagMutex = PTHREAD_MUTEX_INITIALIZER;
68
69         Storage* A = NULL;
70         pthread_rwlock_t rwlockA = PTHREAD_RWLOCK_INITIALIZER;
71
72         Storage* B = NULL;
73         pthread_rwlock_t rwlockB = PTHREAD_RWLOCK_INITIALIZER;
74
75         int updating = 0; // 1 means update in process, 0 means no update happening
76         pthread_mutex_t updatingMutex = PTHREAD_MUTEX_INITIALIZER;
77 }
```

## 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 in C# as a candidate solution for Microsoft's 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

<b>Test Environment:</b> Laptop, CPU i7-1065G7, 4 cores, 16GB RAM							
<b>Data:</b> 3 million key-value pairs. Keys are int64 and values are 265 byte arrays.							
<b>Test Method:</b>							
1. In each execution, all processes target <b>RANDOM</b> records.							
2. 4 process to read. 4 process to add or update. 4 process to delete.							
3. Start 12 processes at the same time and try to finish them at the same time.							
Lock-Free Hash Table			.Net Concurrent Dictionary			RPS Comparison	Success Rate Comparison
Method	RPS	Success Rate	Method	RPS	Success Rate		
TryGet	3,780,242	86.53%	TryGetValue	4,924,242	96.11%	-30.26%	-11.07%
TrySet	3,233,309	100.00%	TryAdd	4,404,597	3.44%	-36.23%	96.56%
TryDelete	7,625,000	5.62%	TryRemove	9,584,365	6.46%	-25.70%	-14.95%

# 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 ([tutorial](#)). 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](#)

## 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，签署无固定期限劳动合同。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





## 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.



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: <i>Zhu Xiunan</i>
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



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.**

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



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*