python × PYCHINA

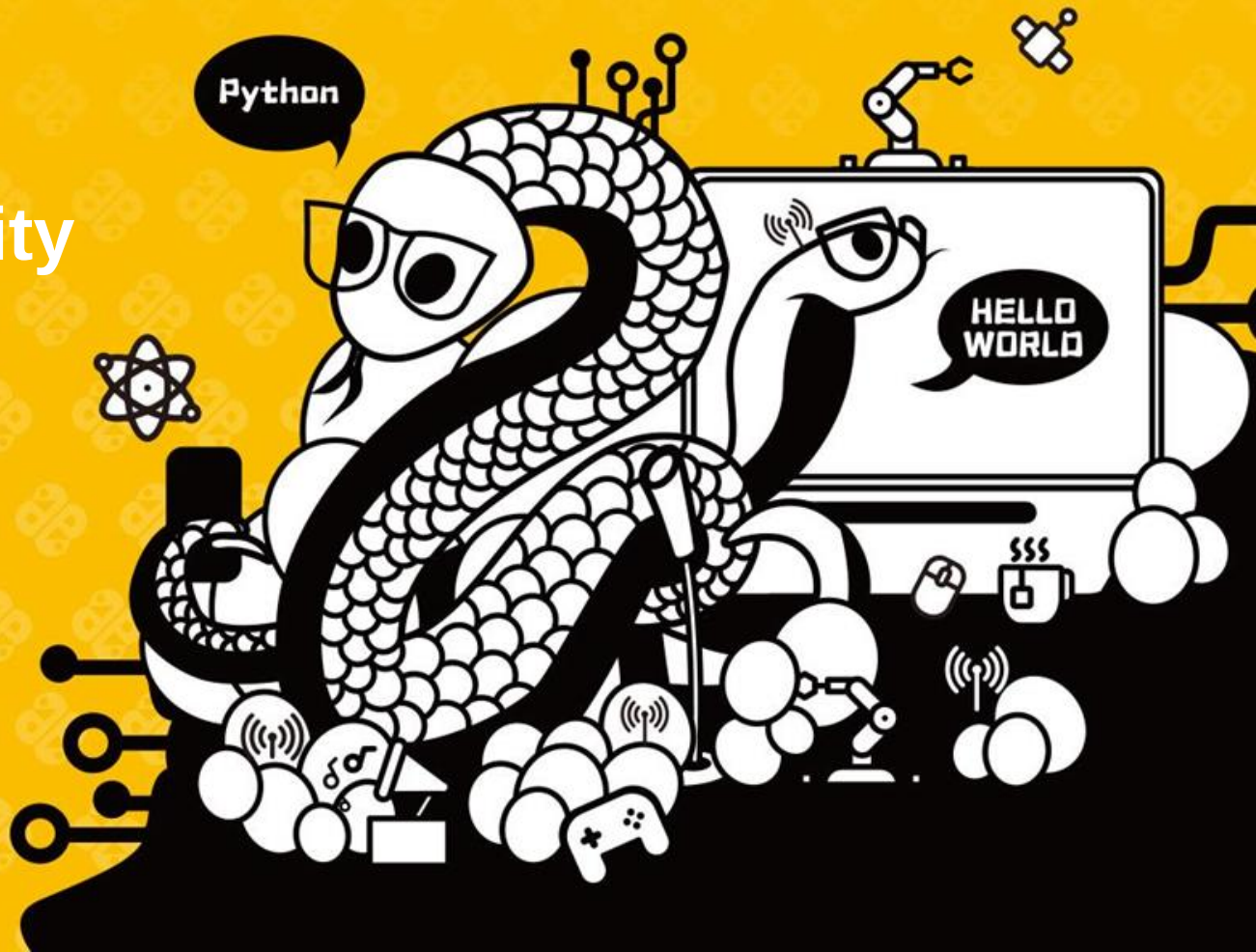
Python for Good

PyCon China 2022

Model Fine-tuning For Similarity Learning

from algorithms to infrastructure

Bo Wang
Engineering Manager@Jina AI | bo.wang@jina.ai



Me

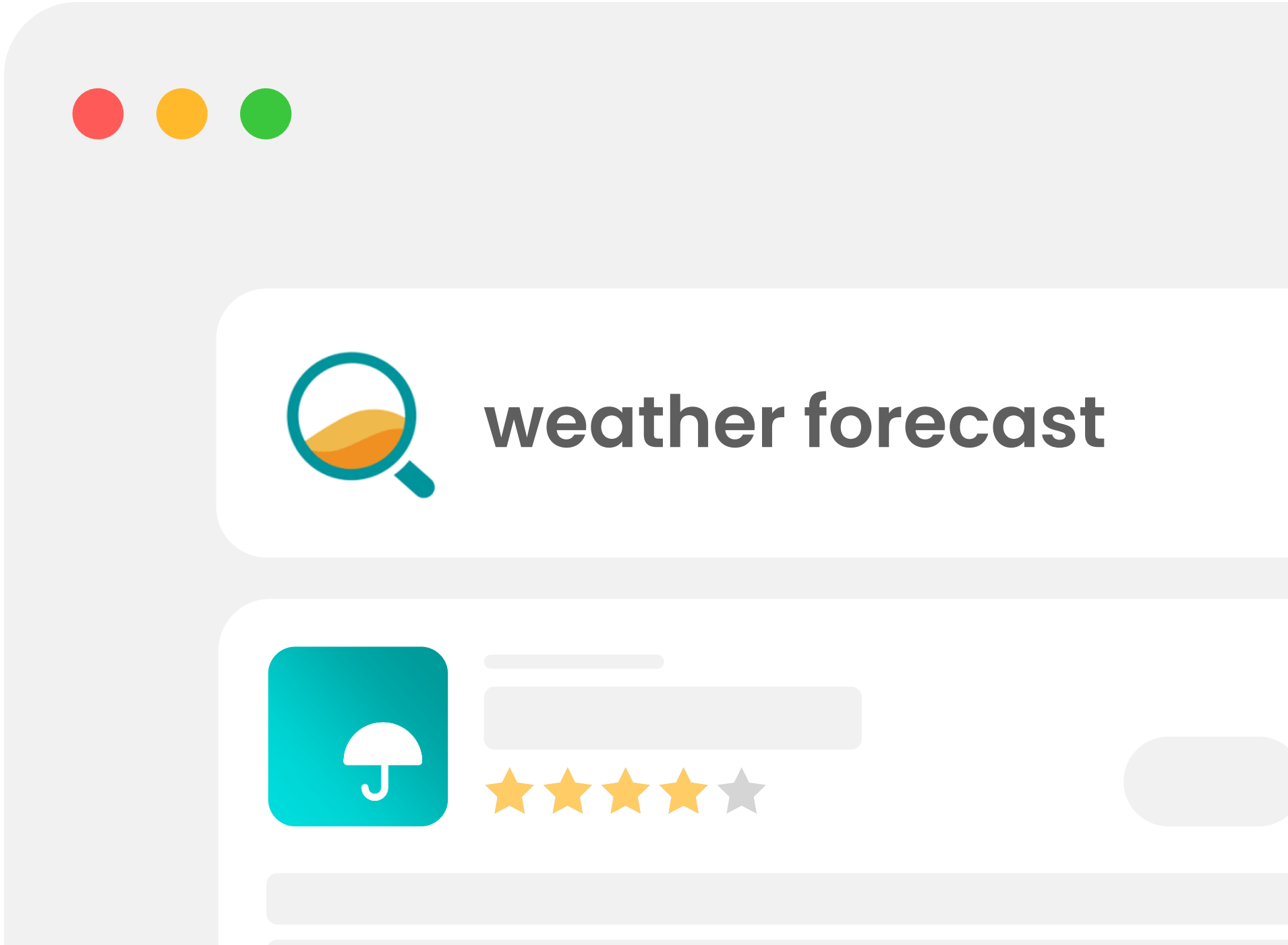
Bachelor: Lanzhou University

Master: Delft University of Technology (Multimedia Computing Group)

Working: IBM (Amsterdam), Sensara (Rotterdam), Jina AI (Berlin, current)

Github: bwanglzu

Twitter: bo_wangbo


weather forecast

sneakers.jpg

black sneakers

Neural Search

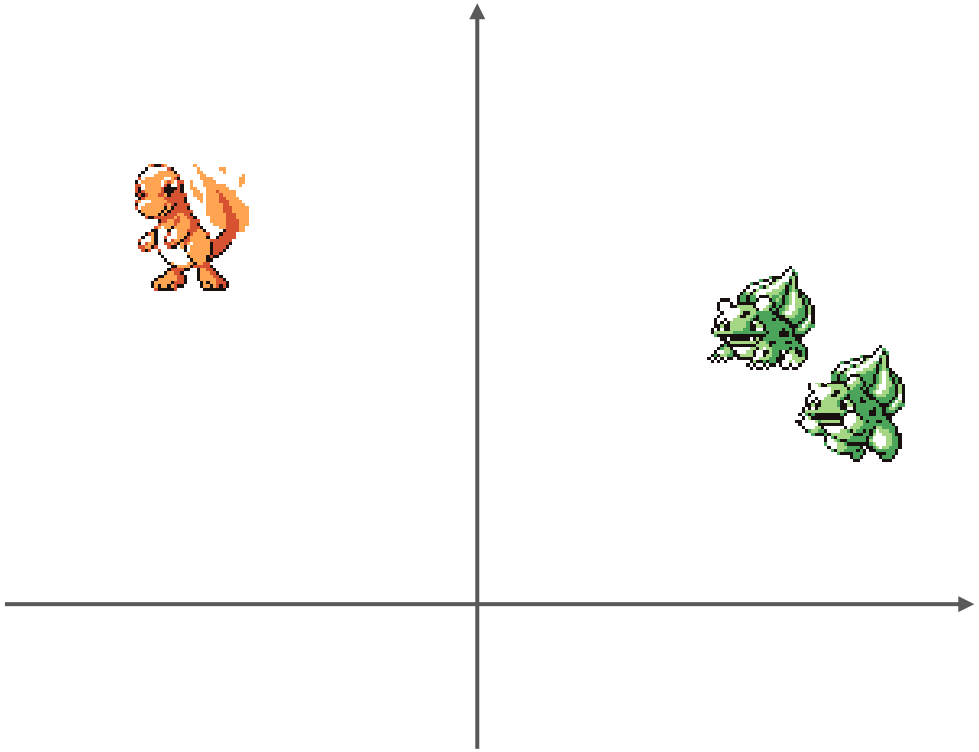
Forget about keyword search

- Basic idea: use embeddings from Deep Learning models for similarity search (cosine, euclidean)

=> [0.6, 0.4]

=> [0.5, 0.5]

=> [-0.5, 0.6]



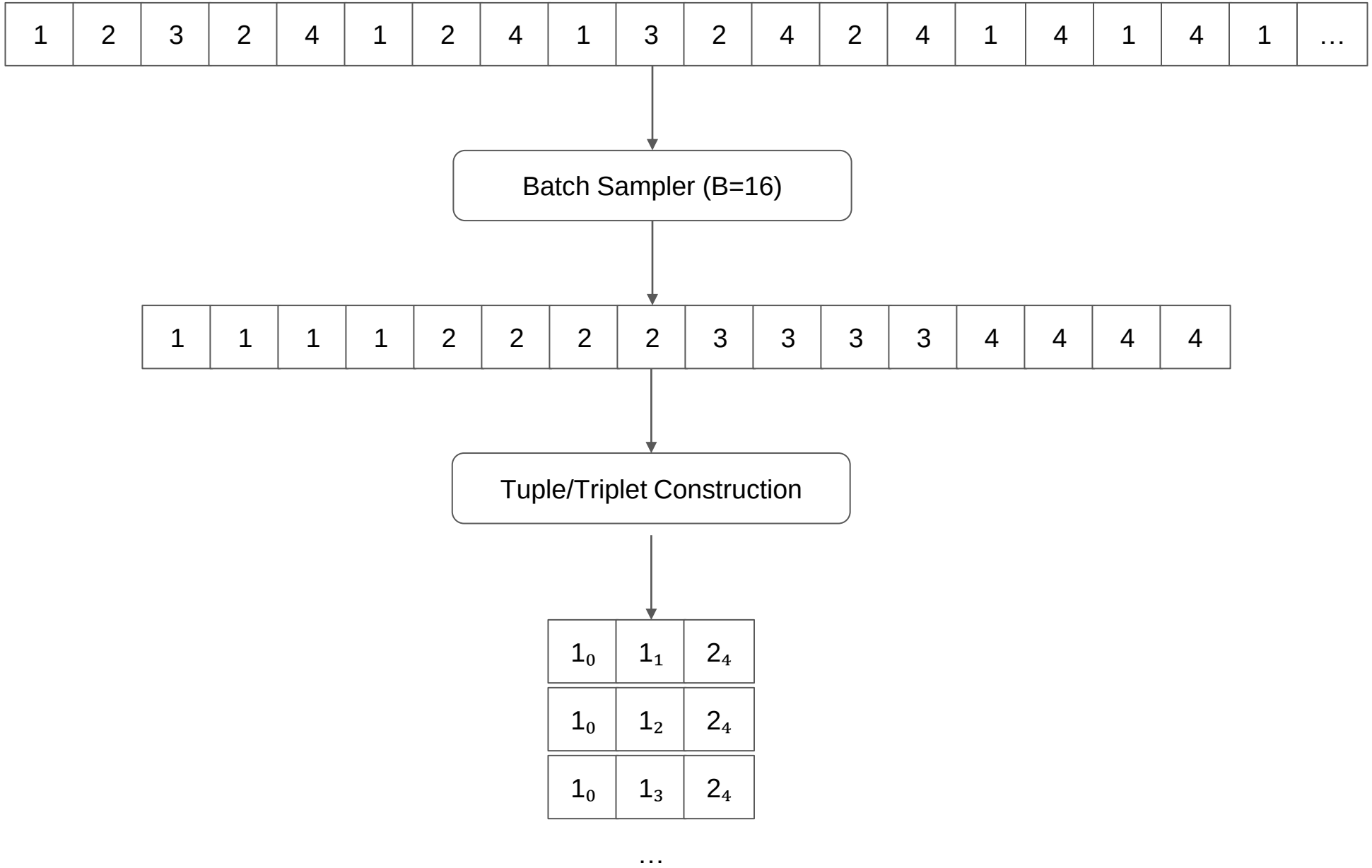
However, pre-trained models do not work out-of-the-box (for search)

Why?

1. **Domain shift**: the training data and the application domain is different.
2. **Task shift**: Most of the models are pre-trained for classification/regression/segmentation, not for similarity matching.
3. **Knowledge shift**: Machine learning engineers do not have enough knowledge on Information Retrieval, vice versa.

Our Approach: Contrastive Learning

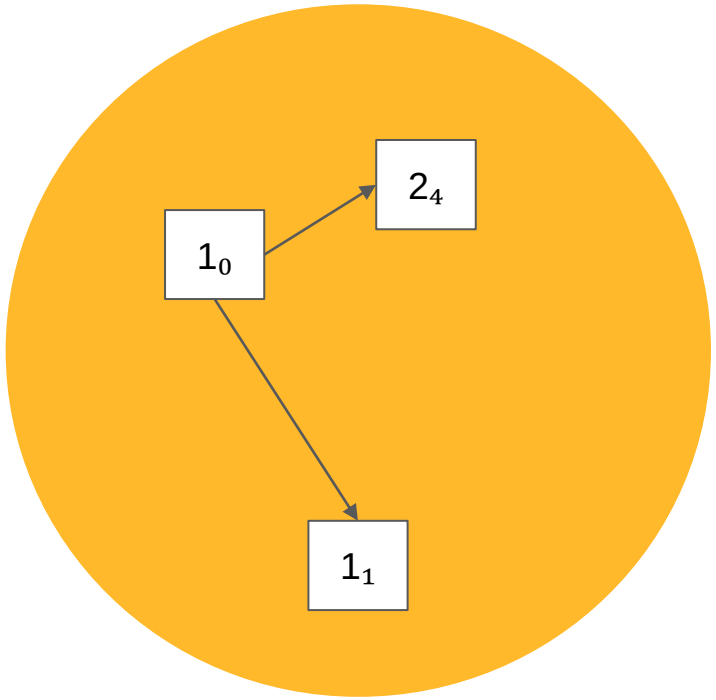
With Labels - 1



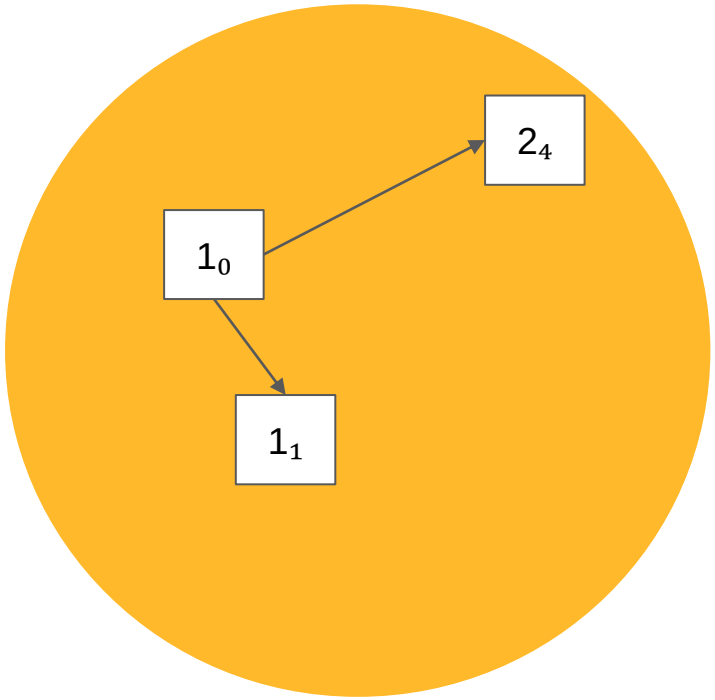
With Labels - 2

1_0	1_1	2_4
1_0	1_2	2_4
1_0	1_3	2_4

...



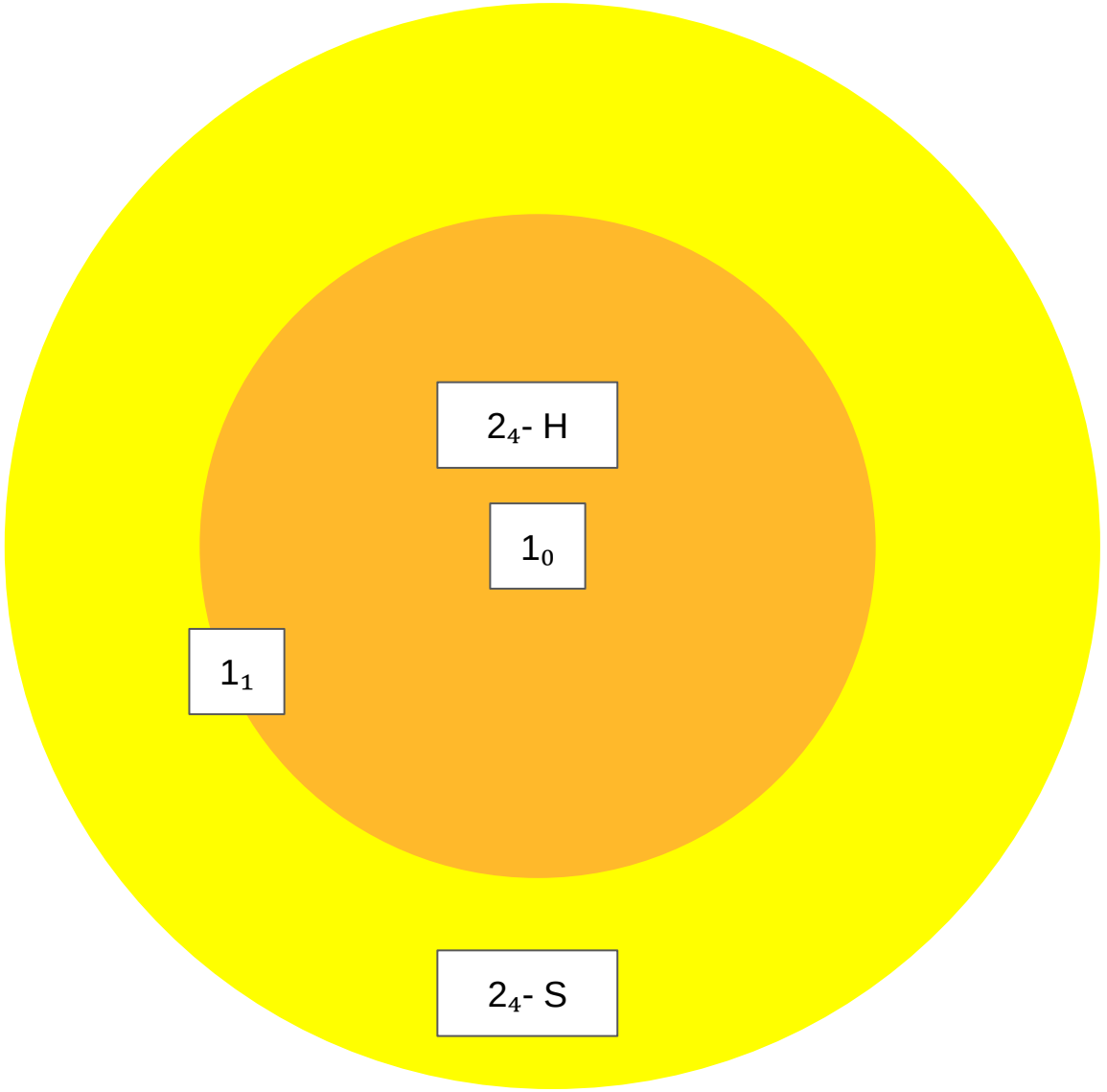
Learning



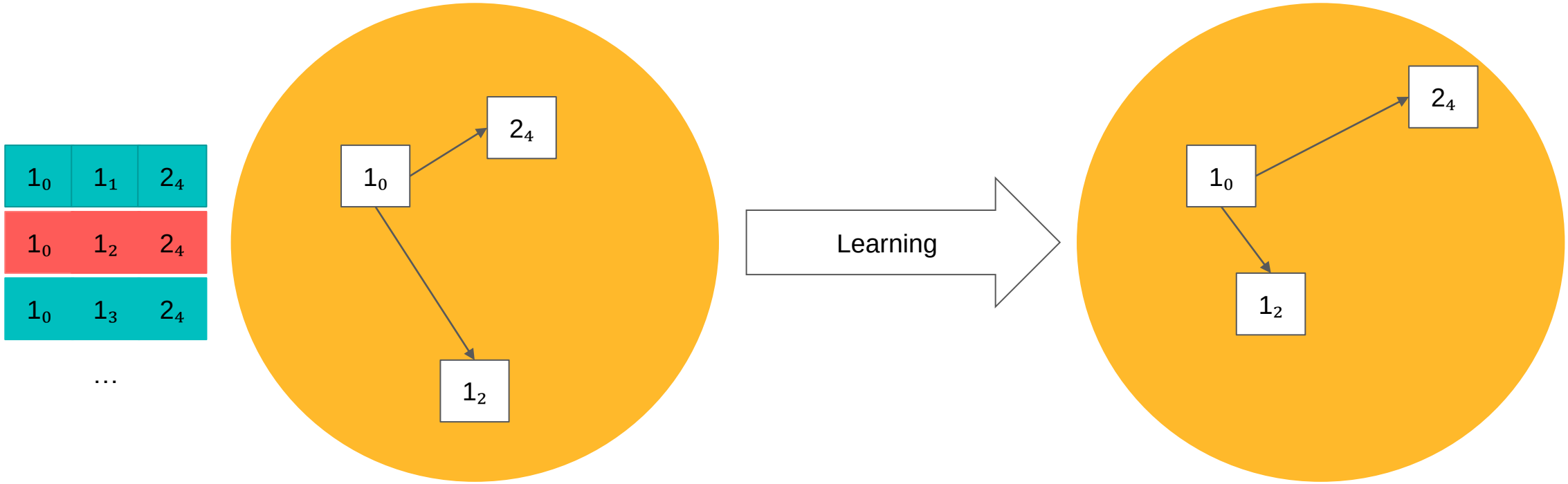
$$L(a, p, n) = \max\{d(a_i, p_i) - d(a_i, n_i) + \text{margin}, 0\}$$

With Labels - 3

2_4- E



With Labels - 4



Xuan, Hong, Abby Stylianou, and Robert Pless. "Improved embeddings with easy positive triplet mining." *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2020.

triplet-example

```
import finetuner

finetuner.login()

finetuner.fit(
    model = 'bert-base-cased',
    train_data = 'your-train-data',
    eval_data = 'your-eval-data',
    loss='TripletMarginLoss',
    learning_rate=1e-4,
    batch_size=16,
    epoch=5,
    device='cuda',
    miner='TripletMarginMiner',
    miner_options={
      'margin': 0.3,
      'type_of_triplets': 'hard'
    },
    ...
)
```

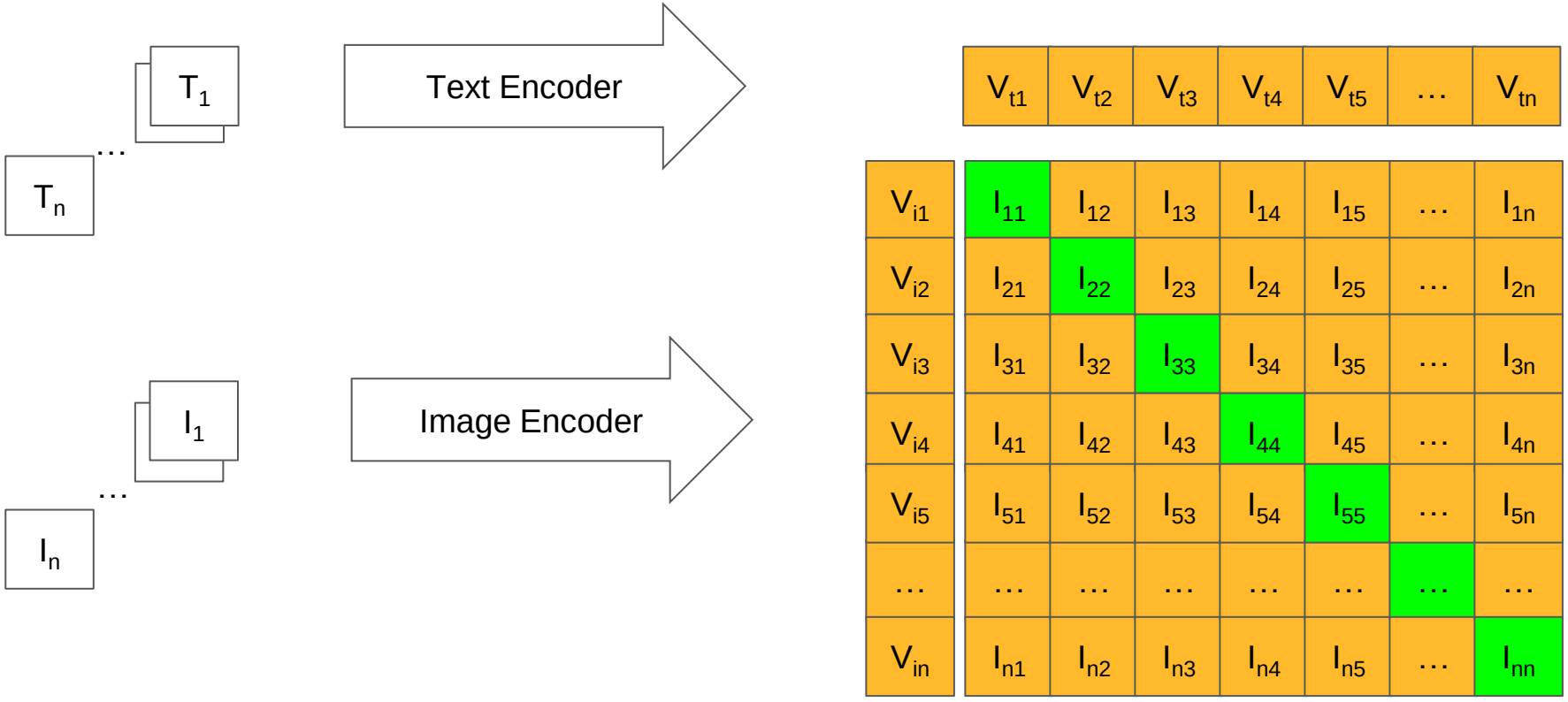
Without Labels - CrossModel

black sneakers comfortable

Text Encoder → feature vector V_t

Image Encoder → feature vector V_i

Without Labels - CrossModel



Radford, Alec, et al. "Learning transferable visual models from natural language supervision." *International Conference on Machine Learning*. PMLR, 2021.

clip-example

```
import finetuner

finetuner.login()

finetuner.fit(
    model = 'openai/clip-vit-base-patch32',
    train_data = 'your-train-data',
    eval_data = 'your-eval-data',
    loss='CLIPLoss',
    learning_rate=1e-6,
    batch_size=128,
    epoch=5,
    device='cuda',
    ...,
    callbacks=[
      EvaluationCallback(
        model='clip-text',
        index_model='clip-vision',
        query_data='your-query-data',
        index_data='your-index-data',
      )
    ],
)
```

Without Labels - UniModel

Contrastive Loss

v_0

v_1

Encoder

Encoder

What do get from it?

Model	Task	Metric	Pretrained	Finetuned	Delta	Run it!
BERT	Quora Question Answering	mRR	0.835	0.967	15.8%	Open in Colab
		Recall	0.915	0.963	5.3%	
ResNet	Visual similarity search on TLL	mAP	0.110	0.196	78.2%	Open in Colab
		Recall	0.249	0.460	84.7%	
CLIP	Deep Fashion text-to-image search	mRR	0.575	0.676	17.4%	Open in Colab
		Recall	0.473	0.564	19.2%	
M-CLIP	Cross market product recommendation (German)	mRR	0.430	0.648	50.7%	Open in Colab
		Recall	0.247	0.340	37.7%	

All metrics were evaluated for k@20 after training for 5 epochs using the Adam optimizer with learning rates of 1e-4 for ResNet, 1e-7 for CLIP and 1e-5 for the BERT models.

What do you pay?

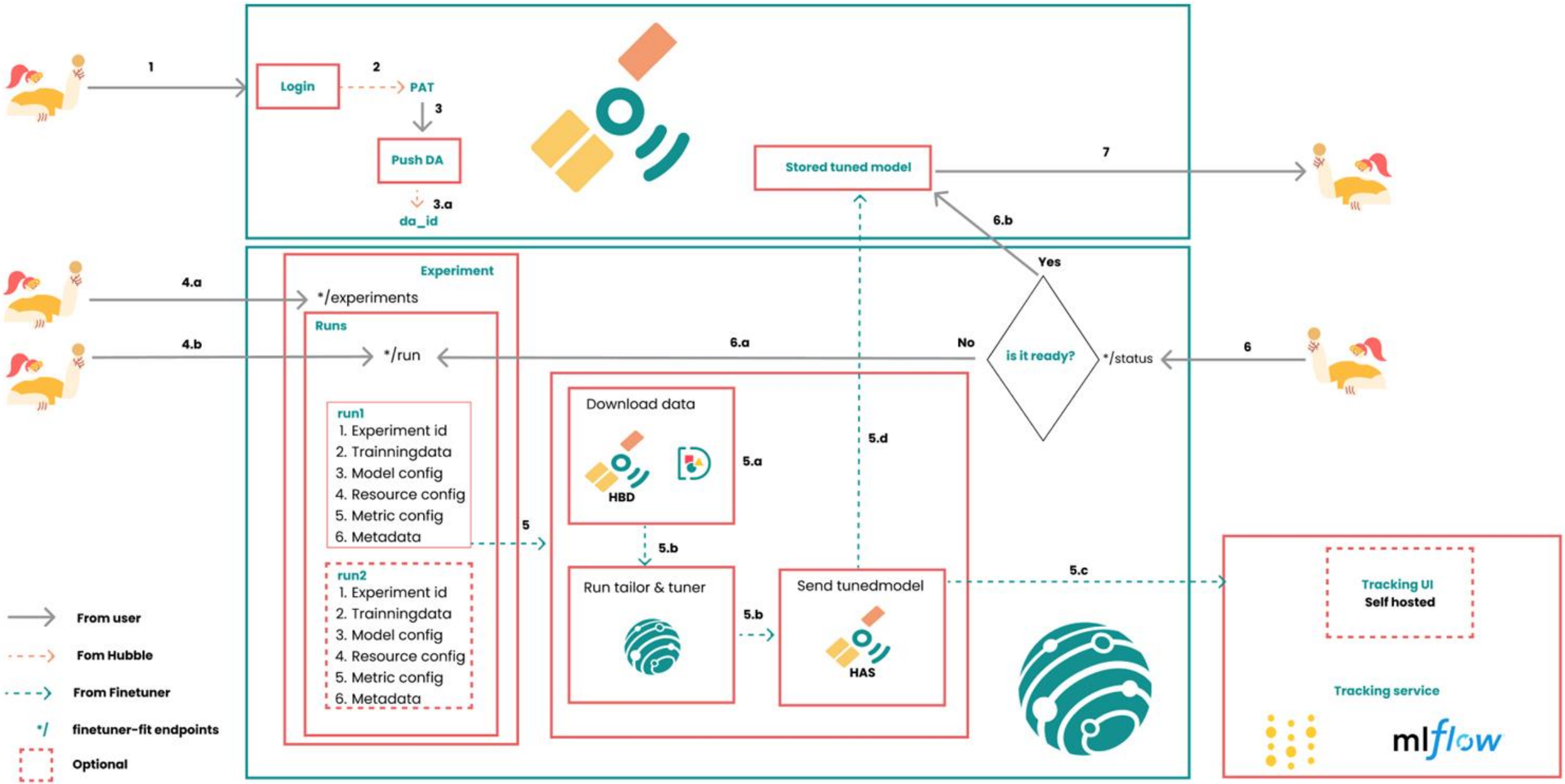
1. Depends, a few handureds to several thousands labeled data is sufficiently enough.
2. Given the effectiveness of GPU utilization, and limited amount of training data, fine-tuning could be finished in 15 minutes to less than half a day.

Fine-tuning with Low Budget and High Expectations:
https://jina.ai/news/fine-tuning-with-low-budget-and-high-expectations/

This is not the end..

Model Fine-tuning as Service

1. For software engineers, still too complex.
2. Lack of computing resources, or lack of experience in GPU utilization.
3. Heavy dependencies (ML Frameworks).


1
Login
2
PAT
3
Push DA
3.a
da_id
Stored tuned model
7
6.b
Yes
4.a
Experiment
*/experiments
Runs
4.b
*/run
6.a
No
is it ready?
*/status
6
run1
1. Experiment id
2. Trainningdata
3. Model config
4. Resource config
5. Metric config
6. Metadata
run2
1. Experiment id
2. Trainningdata
3. Model config
4. Resource config
5. Metric config
6. Metadata
5
Download data
HBD
5.a
5.d
5.b
Run tailor & tuner
5.b
Send tunedmodel
HAS
5.c
Tracking UI
Self hosted
Tracking service
mlflow
From user
Fom Hubble
From Finetuner
*/
finetuner-fit endpoints
Optional

Before/after you call the `fit` function..

1. At each release, we build cpu and cuda docker image to Container Registry.
2. A dedicated computing resource will be applied for you, with X GPUs, Y CPUs and resources such as memory and disk space from our cloud Kubernetes Cluster.
3. We utilize Argo Workflow which is a container-native workflow engine to run your fine-tuning job.
4. We utilize Karpenter for node auto-scaling, to provision nodes based on workload requirements, at scale.
5. In the mean while, all the logs will be centralized in our cloud buckets & logging stack.

With this setup, we can efficiently setup computing resources within minutes with CPU or CUDA jobs (up to 16 Nvidia GPUs)

For open-source users, we always offer one free GPU.

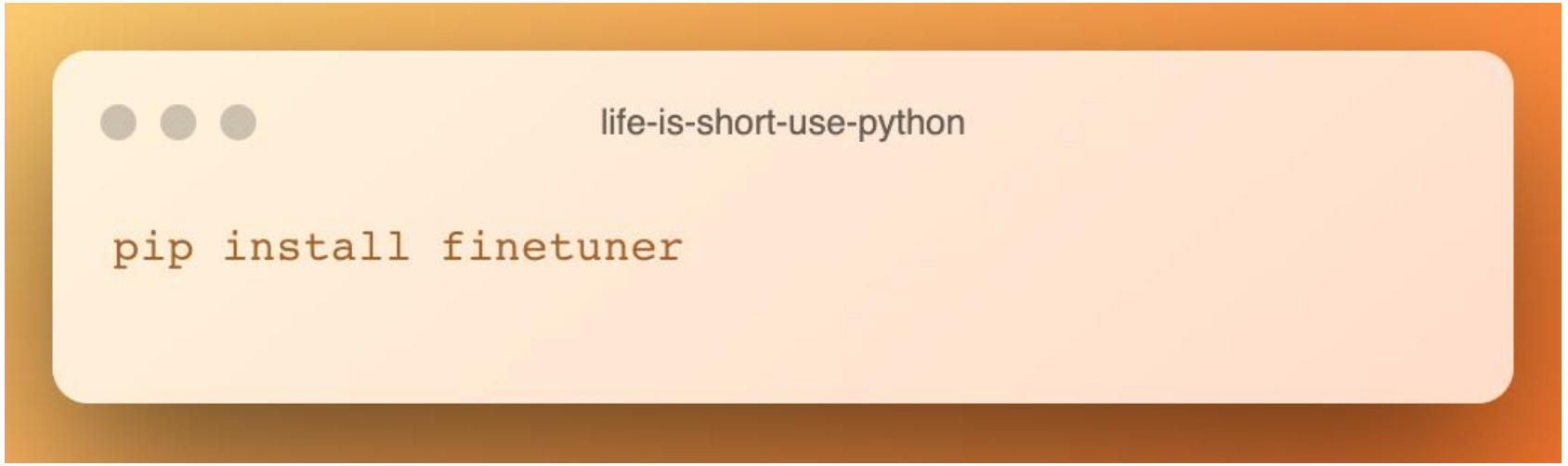
What we achieved

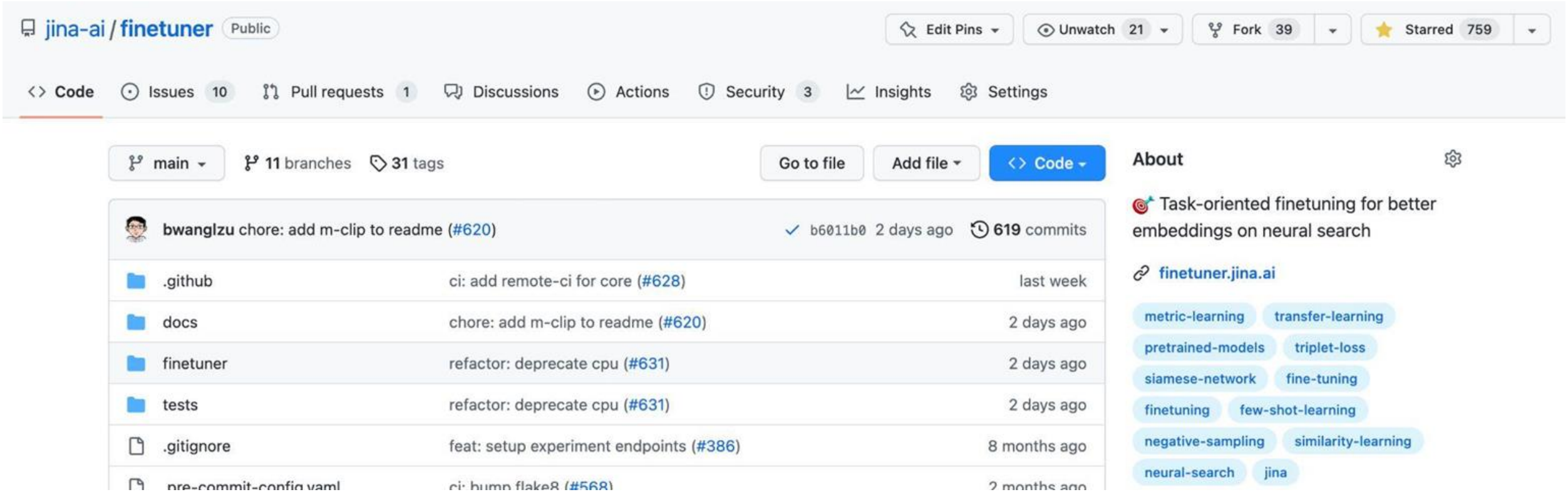
1. Extreme easy user interface, even with no code.
2. Always full (90%+) GPU Utilization in the cloud given sufficient training data.
3. Huge dependency size reduction (1GB +), no need torch, torchvision etc. Since each `fit` is just an api call...
4. July 1st since release up to Dec 15th 2022, we help our internal & external users fine-tuned 1423 models.

Challenges & Next Step

1. Losing certain level of flexibility, for advanced users.
2. Data handling should at users hand, this is crucial for big entities.
3. Not enough ppl, even ML engineers are aware the power of embedddings.
4. Breakthrough embedding model towards cross-modality/mult-modality.
5. Be more open & developer friendly towards to the next step: inference.

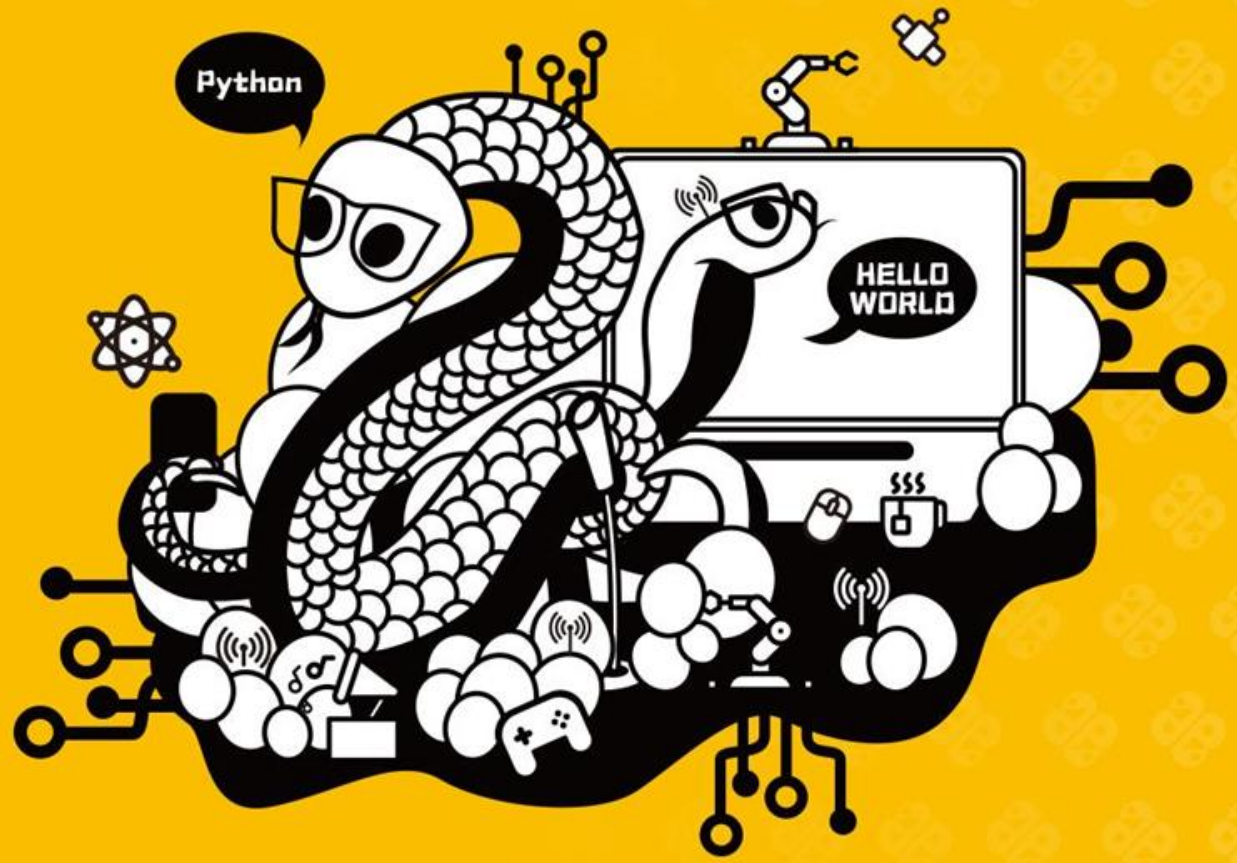
Last but not least..




jina-ai / finetuner
Public
Edit Pins
Unwatch 21
Fork 39
Starred 759
Code
Issues 10
Pull requests 1
Discussions
Actions
Security 3
Insights
Settings
main
11 branches
31 tags
Go to file
Add file
Code
bwanglzu chore: add m-clip to readme (#620)
b6011b0 2 days ago
619 commits
.github
ci: add remote-ci for core (#628)
last week
docs
chore: add m-clip to readme (#620)
2 days ago
finetuner
refactor: deprecate cpu (#631)
2 days ago
tests
refactor: deprecate cpu (#631)
2 days ago
.gitignore
feat: setup experiment endpoints (#386)
8 months ago
About
Task-oriented finetuning for better embeddings on neural search
finetuner.jina.ai
metric-learning
transfer-learning
pretrained-models
triplet-loss
siamese-network
fine-tuning
finetuning
few-shot-learning
negative-sampling
similarity-learning
neural-search
jina

Thank you!

jina.ai slack.jina.ai @JinaAI_ github.com/jina-ai jina.ai/careers/



Thanks!

感谢观看