Group Project
EB-NeRD RecSys Challenge

Recommender Systems

Group 26

Maximilian Ranzinger (12023139)
Pia Schwarzinger (12017370)
Philipp von der Thannen (12014568)
Yahya Jabary (11912007)

Experimental Setup

Overview

Set up environment (locally)

Small dataset

Implementation

NRMS

GRU (assigned)

LSTUR

Optimization

Algorithms

Gated Recurrent Unit

- RNN – simpler than LSTM (reset gate and update gate)
- Computationally less intensive, faster to train
- GRUCell: temporal dependened sequences, general pupose
- Collaborative Filtering

Long- and Short-term User Representations (LSTUR)

- Specifically tailored for news recommendation
- Combines user representations with current session data as embeddings
- Uses GRU layers to generate user-activity representations
- Content-Based Filtering

Expectations vs Reality

- Baseline and LSTUR – partially in ebnerd-benchmark
  - "simply" implement the algorithmis
  - Started with earlier version of repository -> after update adaptions
- GRU - Only quick start in recommender repository (Amazon set)
  - Outdated implementation of GRU model
  - Preprocessing of data
    - Apply Amazon Review preprocessing to ebnerd data (Creation of user/item/category vocabulary)
    - Format of dataframe for sequential iterator

Technical Setup

Systems

Local Workstation

- **CPU:** 12-Core AMD Ryzen 9 7900X
- **RAM:** 32GB DDR5 5600MHz
- **GPU:** NVIDIA RTX 4080 (16GB)

- Used system, but largely limited by memory (e.g. TensorBoard needed 42GB for small dataset)

JupyterHub Issues

- Conda only usable via workaround
- GPU execution runs into JIT compilation error
- CPU obviously takes forever, but is proposed workaround
  (194.035-2024S: EBNeRD - nrms notebook train model error | TUWEL (tuwien.ac.at))

Hyperparameter Optimization

Hyperparameter Optimization

- Done with **Tensorflow Keras Tuner**
- Used algorithm: Hyperband by Li et al. (2018)
  - Less resource hungry than Bayesian Optimization
  - Faster than Grid Search
  - More targeted approach than Random Search
  - Optimization on validation loss
- Runtimes:
  - NRMS: 1h 42min
  - LSTUR: 10h 11min
- Large storage requirement

Hyperparameter Optimization: Findings

- NRMS Hyperparameter Optimzation found no optimization
- LSTUR Hyperparameter Optimization found improvement
- Some Hyperparameters could not be tuned, because of fixed architecture
- Low validation loss does not mean better algorithm

Metric \ Algorithm	NRMS (baseline)	NRMS (HP opt.)	LSTUR (fixed)	LSTUR (HP opt.)
AUC	0.556486	0.555230	0.570433	**0.571175**
MRR	0.349973	0.348463	0.354222	**0.358854**
NDCG@5	0.389757	0.387890	0.396656	**0.401295**
NDCG@10	0.467566	0.465807	0.472802	**0.476100**
validation loss	1.2980	1.2941		**0.0779**

Hyperparameter Optimization: Parameters

Parameter	NRMS (baseline)	NRMS (opt)	LSTUR (fixed)	LSTUR (opt)
title size	30 (Fixed)			
history size	30	20	30	10
head num	20	32	-	-
head dim	20	10	-	-
att. hidden dim	200	200	200	250
dropout	0.2	0.3	0.2	0.3
learning rate	0.0001	0.000129	0.0001	0.000465
optimizer	Adam			
loss	cross entropy loss			
#users	-	-	50000	100000
activation func	-	-	relu	tanh
type	-	-	ini	ini
gru unit	-	-	400 (fixed)	
filter num	-	-	400 (fixed)	
windows size	-	-	3	4

Conclusion

Lessons Learned, Conclusion, Open Questions

- Familiarizing ourselves with content took longer than expected
- Extreme workload overhead with environment set up and GRU implementation
- Biggest technical issue = too little memory
- GPU execution and CONDA do not work together on GPU Jupyter Notebook. What should we do for the final submission?
- Improvement according to beyond accuracy measures not implemented yet
  - Possible Approach – Rerankig the final list by assigning weights
  - Only test set (large = small set x 50) has Is_Beyond_Accuracy