

# NEURAL COLLABORATIVE FILTERING

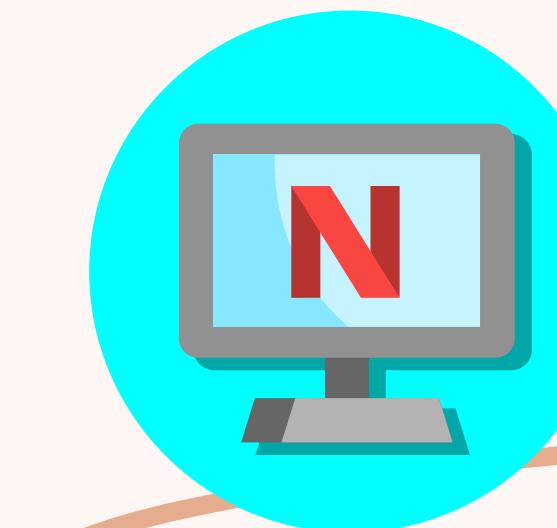
*with MovieLens dataset*

วีระชาติ เรืองสารา 6620422029  
ณกัทธร กัพมี 6620422031  
ธิวากร ชัยวงศ์ 6620422034

# RECOMMENDATION

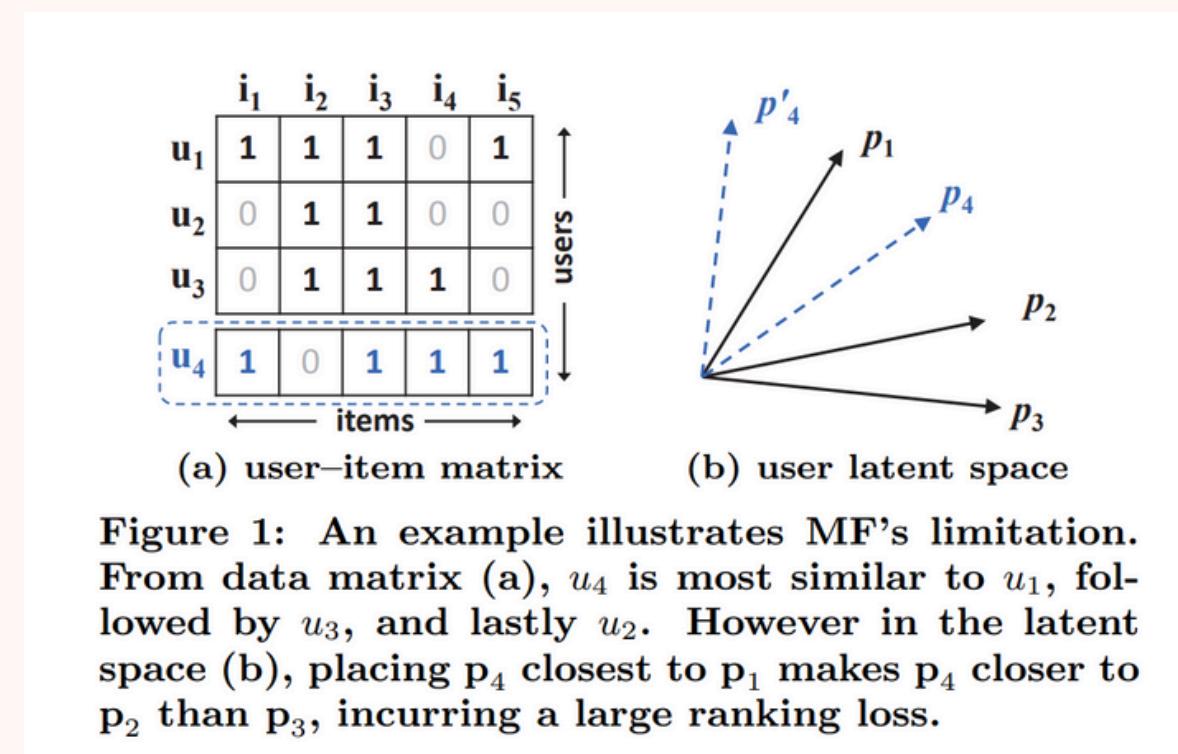


ในยุค “สภาวะข้อมูลก่ำกับ” (Information Overload) ระบบแนะนำ (Recommender Systems) ได้กลายเป็นส่วนสำคัญในการช่วยผู้ใช้ค้นหาข้อมูลที่เกี่ยวข้อง เช่น สินค้า เพลง และภาพยนตร์ โดยระบบแนะนำที่ได้รับความนิยมอย่าง Collaborative Filtering (CF) มีข้อจำกัด ในเรื่องความสามารถในการจับโครงสร้างข้อมูลที่ซับซ้อน เนื่องจากการใช้ Matrix Factorization (MF) ที่อาศัยพังก์ชันคูณแบบ Inner Product เป็นหลัก งานวิจัยนี้มุ่งพัฒนา Neural Collaborative Filtering (NCF) ซึ่งเป็นกรอบงานที่ใช้ Deep Neural Networks (DNNs) เพื่อเพิ่มความสามารถและความยืดหยุ่นในระบบแนะนำ



# WHY NOT MF?

ทำไม Matrix Factorization (MF) ถึงไม่เพียงพอ ?



จากรูป  
ในส่วน (a) user-item matrix

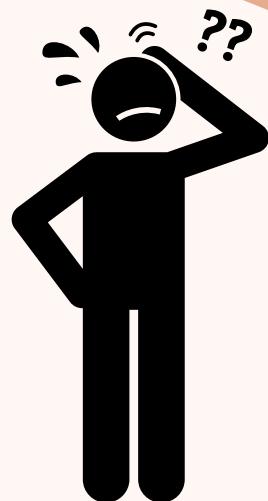
- แสดงความสัมพันธ์ระหว่างผู้ใช้ ( $u_1-u_4$ ) กับสินค้า ( $i_1-i_5$ )
- ค่า 1 คือมีปฏิสัมพันธ์, 0 คือไม่มีปฏิสัมพันธ์

ในส่วน (b) user latent space

- แสดงผู้ใช้ให้อยู่ในรูป vectors ใน latent space
- $p_1-p_4$  แทน vectors ของผู้ใช้  $u_1-u_4$

## ปัญหาที่เกิดขึ้นของ MF

เมื่อพยายามวาง  $p_4$  ให้ใกล้  $p_1$  เนื่องจากมีพหุติกรรมคล้ายกัน ทำให้  $p_4$  ใกล้กับ  $p_2$  มากกว่า  $p_3$  ซึ่งขัดกับข้อมูลจริงที่  $u_4$  ควรคล้าย  $u_3$  มากกว่า  $u_2$  จึงทำให้เกิดปัญหา



## RANKING LOSS

# NCF MODEL

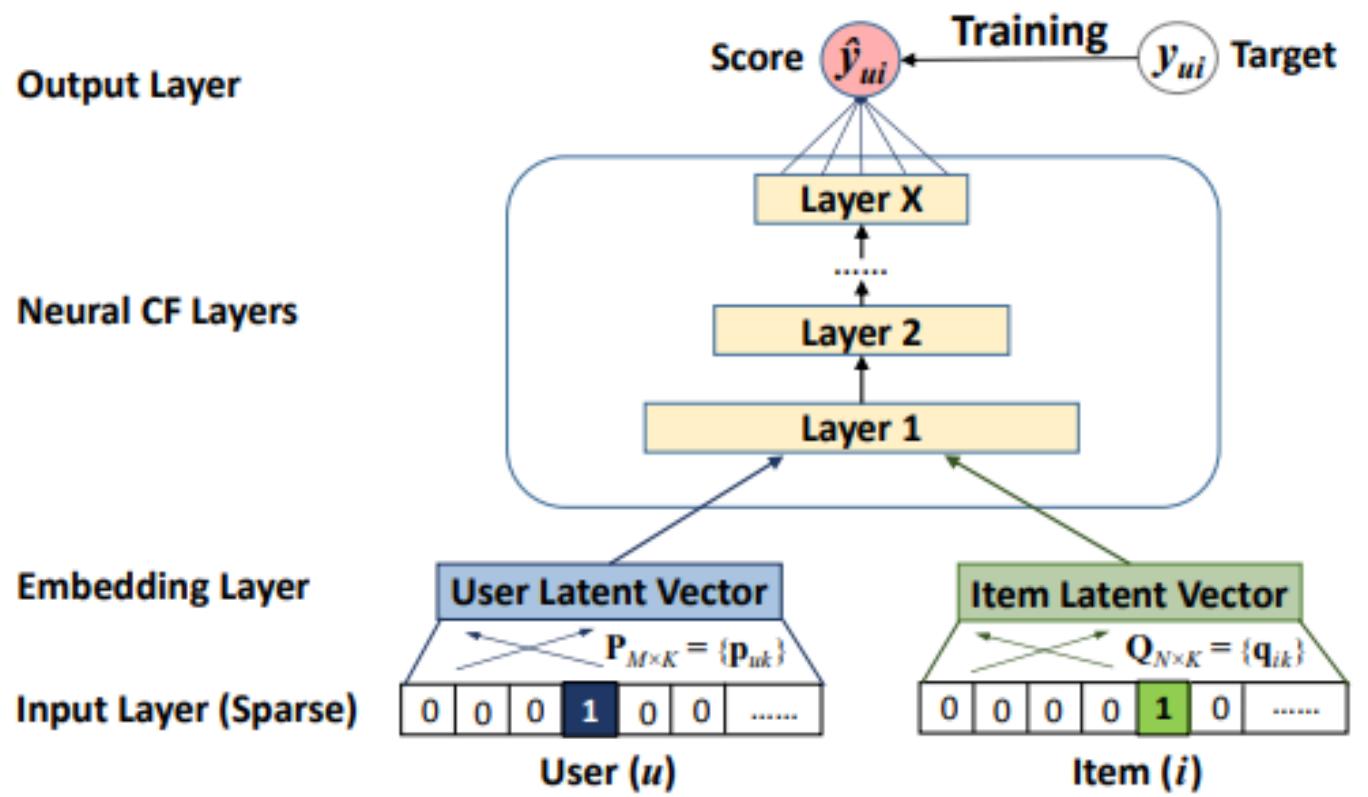


Figure 2: Neural collaborative filtering framework

## Neural Collaborative Filtering (NCF) Framework

### 1. Input Layer (ชั้นนำเข้าข้อมูล)

- เป็นการแทนผู้ใช้และสินค้าด้วย sparse vectors ซึ่งส่วนใหญ่จะเป็น 0 และมี 1 เฉพาะตำแหน่งที่ระบุตัวตนของผู้ใช้หรือสินค้านั้น

### 2. Embedding Layer (ชั้นการแปลงข้อมูล)

- ทำหน้าที่แปลง sparse vectors ให้เป็น latent vectors

### 3. Neural CF Layers (ชั้น Neural Network)

- มีจำนวน Layer 1 ถึง Layer X โดย Layer 1 รับ latent vectors จาก Embedding Layer และส่งต่อไปยัง Layer ถัดไป

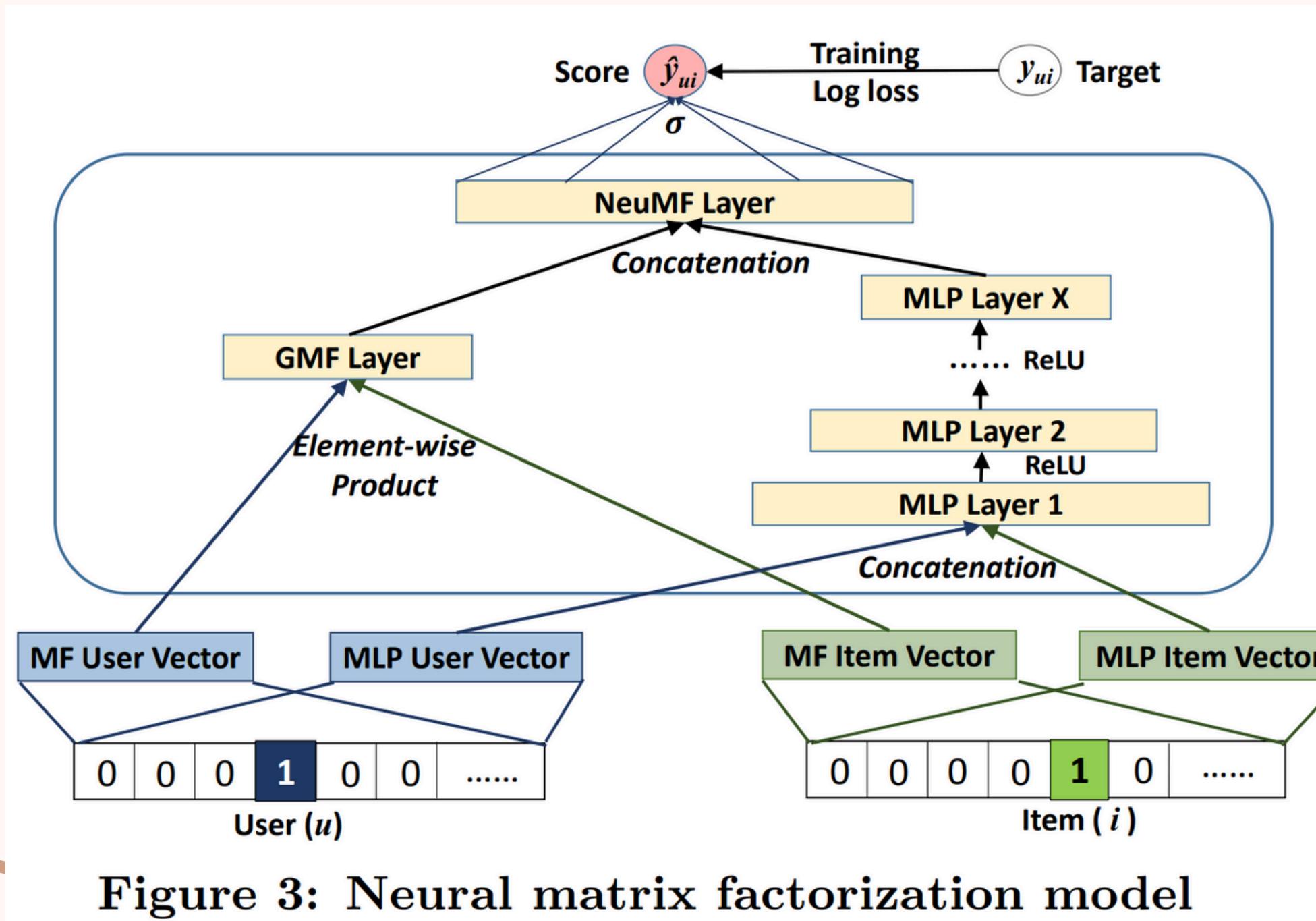
### 4. Output Layer (ชั้นผลลัพธ์)

- คำนวณค่า score ( $\hat{y}_{ui}$ ) ที่แสดงความน่าจะเป็นที่ผู้ใช้  $u$  จะสนใจสินค้า และเปรียบเทียบกับค่าเป้าหมาย ( $y_{ui}$ ) ในระหว่างการ training เพื่อใช้ความแตกต่างระหว่างค่าที่คำนวณและค่าจริงในการปรับปรุงโมเดล

ใน MF แบบดั้งเดิม ใช้การคูณ vectors แบบ inner product ซึ่งเป็นการคำนวณแบบเชิงเส้น (linear) เท่านั้น แต่ NCF แก้ปัญหานี้โดยใช้ neural networks ที่มีหลายชั้น แต่ละชั้นสามารถเรียนรู้รูปแบบความสัมพันธ์ที่ต่างกัน ปรับเปลี่ยนวิธีการเรียนรู้ได้ตามข้อมูล ทำให้ NCF มักให้ผลที่แม่นยำกว่า MF อย่างไรก็ตาม สิ่งสำคัญคือต้องเลือกใช้ให้เหมาะสมกับงาน บางครั้ง MF อาจเพียงพอสำหรับงานที่ไม่ซับซ้อน

# NEUMF MODEL

Neural Matrix Factorization (NeuMF) เป็น Model กี่ใช้ในระบบแนะนำ (Recommender Systems)  
เป็นการผสมผสานของ MF และ NCF เข้าด้วยกัน ประกอบด้วย 3 Layers



1. Generalized Matrix Factorization (**GMF**)
2. Multi-Layer Perceptron (**MLP**)
3. Neural Matrix Factorization (**NeuMF**)

Figure 3: Neural matrix factorization model

# GMF LAYER

## Generalized Matrix Factorization (GMF)

- GMF Layer ทำงานโดยการคำนวณผลคูณแบบรายองค์-ประกอบ (Element-wise Product) ระหว่าง Input vector สองตัว ได้แก่ MF User Vector และ MF Item Vector ที่อยู่ในรูปแบบ One-hot Encoding
- Element-wise Product มีวัตถุประสงค์สำคัญคือ การจับความสัมพันธ์ระหว่างคุณลักษณะแฝง (Latent Features) ของ User และ Item เช่น ในระบบ Recommendation สำหรับหนัง Latent Feature อาจสื่อถึง Genre ที่ชอบ, Preference ต่อบัลลังก์แสดง หรือ Style ของผู้กำกับ

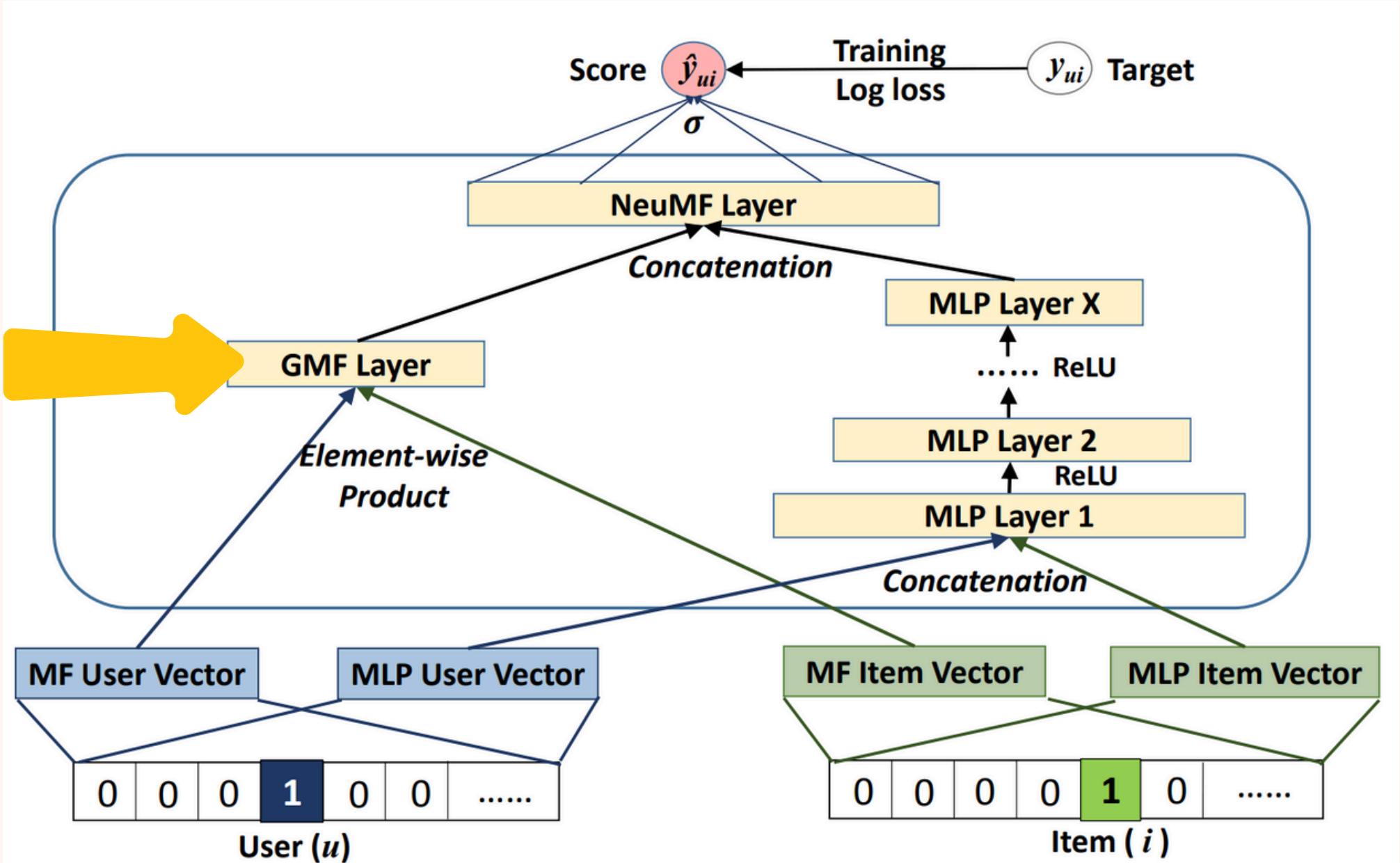


Figure 3: Neural matrix factorization model

# MLP LAYER

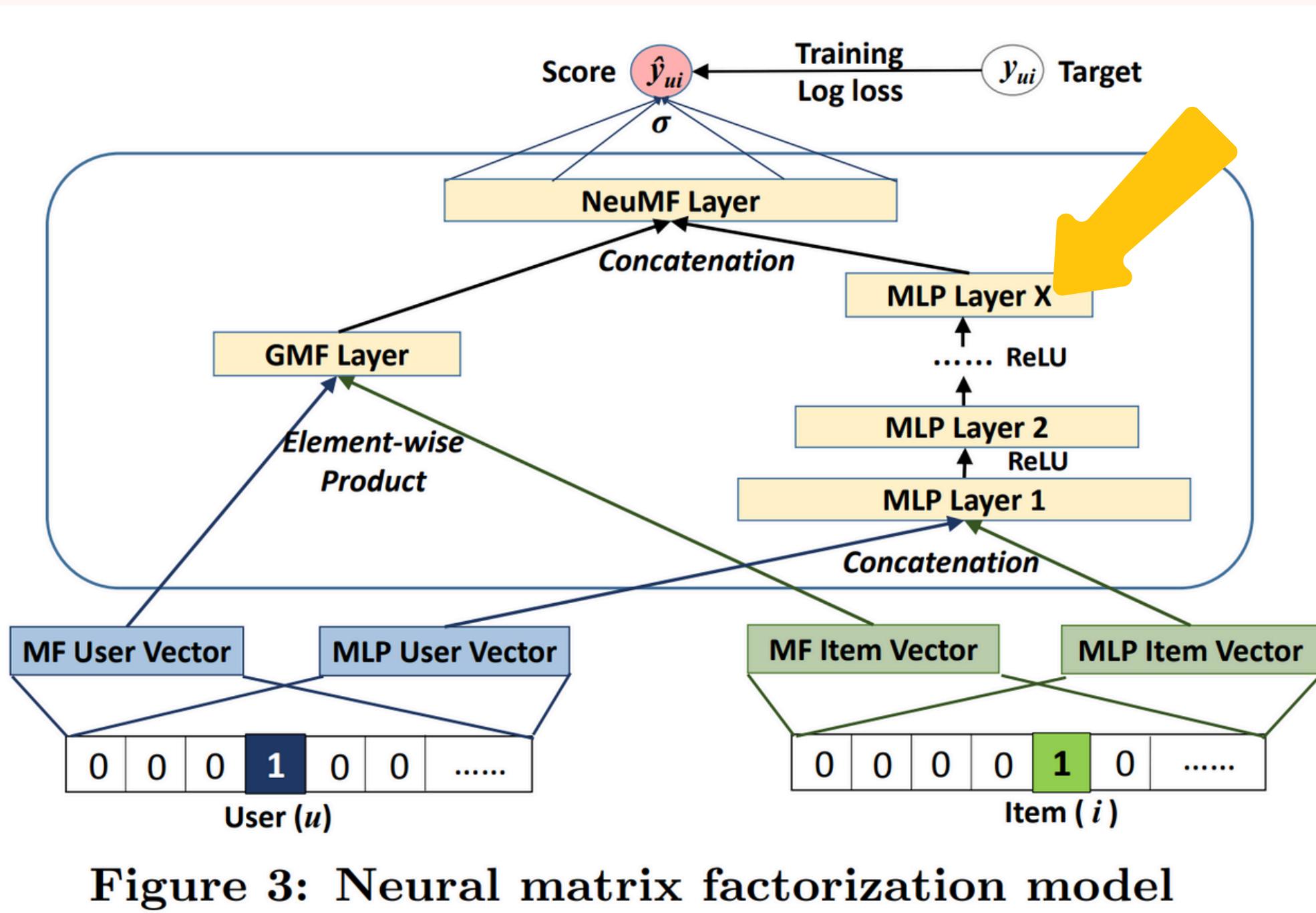


Figure 3: Neural matrix factorization model

## Multi-Layer Perceptron (MLP)

- Input ของ MLP มาจากการ Concatenation ระหว่าง MLP User Vector และ MLP Item Vector ก่อนที่จะถูกส่งเข้า MLP Layer 1 การทำ Concatenation ทำให้ Network สามารถเรียนรู้ความสัมพันธ์ระหว่าง Feature ของ User และ Item ได้อย่างอิสระ ไม่ถูกจำกัดด้วยการคูณแบบ Element-wise เมื่อเป็นใน GMF Layer
- ในแต่ละ MLP Layer จะมีการ Transform ข้อมูลผ่าน Weight Matrix และ Bias รวมถึงการใช้ ReLU เป็น Activation Function เพื่อสร้าง Non-linearity ทำให้ Network สามารถเรียนรู้ Pattern ที่ซับซ้อนได้ เช่น ความสัมพันธ์แบบมีเงื่อนไข (Conditional Relationships) หรือ การ Interact กันระหว่าง Feature หลาย ๆ ตัว
- ได้ Output เป็น MLP Layer X

# NEUMF LAYER

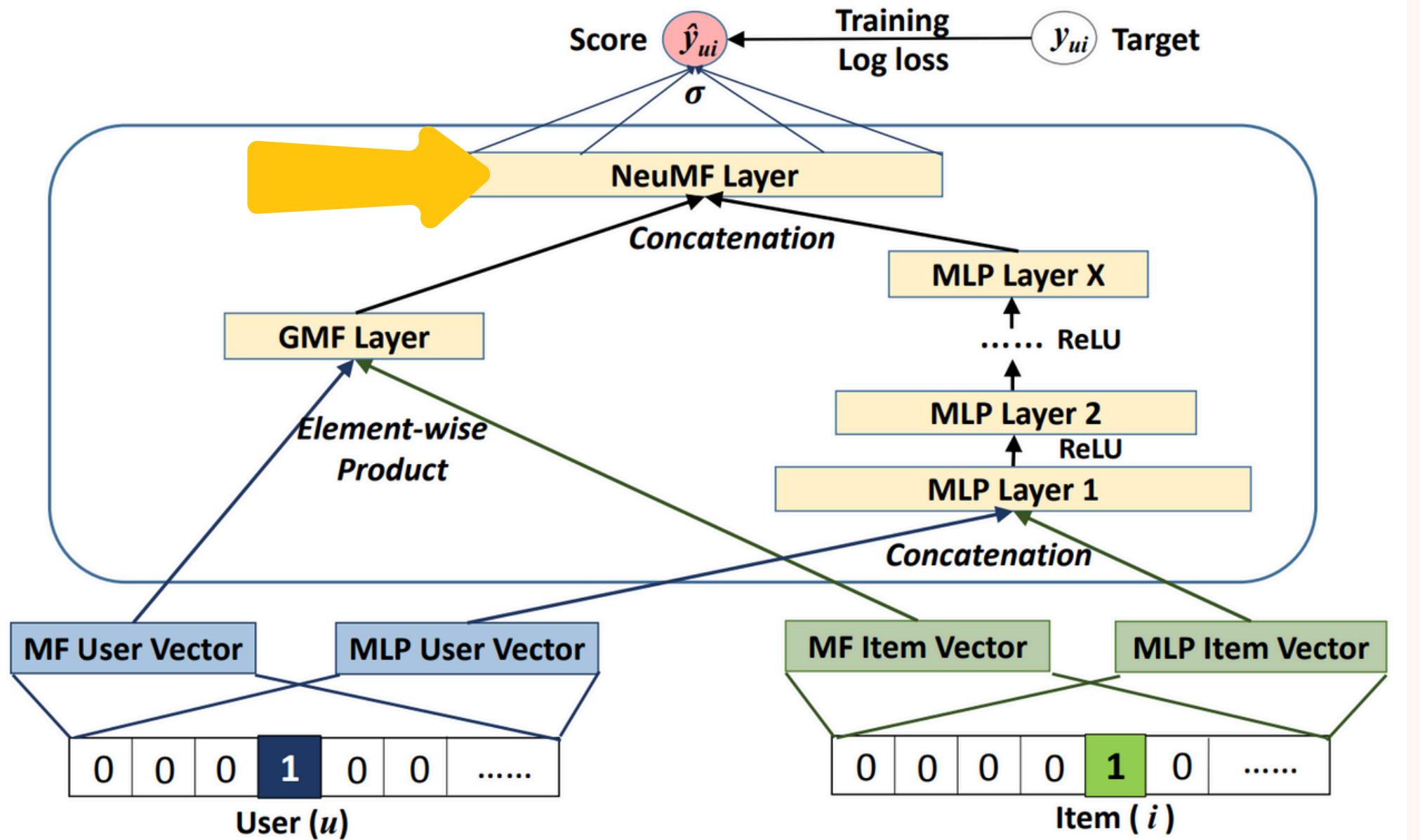


Figure 3: Neural matrix factorization model

## Neural Matrix Factorization (NeuMF)

- NeuMF Layer รับ Input จากการ Concatenate ระหว่าง Output ของ GMF Layer และ MLP Layer X ทำให้ได้ Feature Vector ที่มีความสัมพันธ์ทั้งแบบ Linear และ Non-linear เข้าด้วยกัน จากนั้นจะทำการประมวลผลผ่าน Weight Matrix และ Activation Function เพื่อคำนวณคะแนนสุดท้าย (Score) ที่แสดงถึงความน่าจะเป็นที่ User สนใจ Item
- การใช้ Loss Function ในการ Training ช่วยให้ NeuMF Layer สามารถปรับ Weight ทั้งของตัวเองและของ Layer อื่นๆ ในเครือข่ายเพื่อลดความคลาดเคลื่อนระหว่างคะแนนที่กำหนดกับคะแนนจริง ทำให้ระบบแนะนำมีความแม่นยำมากขึ้น

# NEUMF LAYER

## การเรียนรู้

- ใช้ Log Loss Function ในการจัดการกับข้อมูล Implicit Feedback โดยมองปัญหาเป็นการจำแนกข้อมูลแบบ Binary Classification
- ใช้การสุ่มตัวอย่างลบ (Negative Sampling) เพื่อสร้างตัวอย่างข้อมูลสำหรับการเรียนรู้
- ใช้ Pre-training เพื่อช่วยปรับปรุงประสิทธิภาพของ NeuMF โดยการเริ่มต้นจากโมเดล GMF และ MLP ที่ผ่านการฝึกมาก่อน

# OBJECTIVE



พัฒนาโมเดล Collaborative Filtering ที่สามารถวิเคราะห์โครงสร้างข้อมูลที่ซับซ้อนระหว่างผู้ใช้และสินค้า

ใช้ Deep Neural Networks และการคำนวณแบบ Inner Product ใน MF เพื่อเพิ่มความสามารถในการเรียนรู้ฟังก์ชันปฎิสัมพันธ์ที่ซับซ้อน

มุ่งเน้นการใช้งานกับข้อมูล Implicit Feedback ที่เป็นข้อมูลเชิงพฤติกรรม เช่น การคลิกหรือการซื้อสินค้า

# DATA SET



ข้อมูลที่นำมาใช้ทดสอบ เป็นข้อมูล File .rating ที่เก็บข้อมูลการให้คะแนนหรือการจัดอันดับ (Rating) โดยทั่วไปใช้ในระบบแนะนำ (Recommendation Systems)

ลักษณะของ Dataset:

- **Train Dataset (ml-1m.train.rating):**

- ประกอบด้วย User ID, Movie ID, Rating, Timestamp.
- เป็นข้อมูลการให้คะแนนของผู้ใช้ต่อภาพยนตร์ในระดับคะแนน 1-5
- Timestamp - เวลาที่ให้คะแนน รูปแบบ Unix timestamp
- ลักษณะข้อมูล:
  - Sparse: มีผู้ใช้หลายคน แต่แต่ละคนให้คะแนนภาพยนตร์เพียงบางส่วนเท่านั้น.
  - Explicit Feedback: ใช้คะแนนที่ชัดเจน (Explicit Ratings) เพื่อแสดงความชอบ.

- **Test Dataset (ml-1m.test.rating):**

- โครงสร้างเดียวกับ train dataset ใช้สำหรับการประเมินโมเดล.

- **Negative Dataset (ml-1m.test.negative):**

- ประกอบด้วย Negative Samples (รายการที่ผู้ใช้งานไม่ได้ต้องบ) ซึ่งสำคัญในงาน recommendation.
- ใช้ร่วมกับ test set เพื่อวัดความสามารถของโมเดลในการแยกแยะระหว่าง positive และ negative interactions.

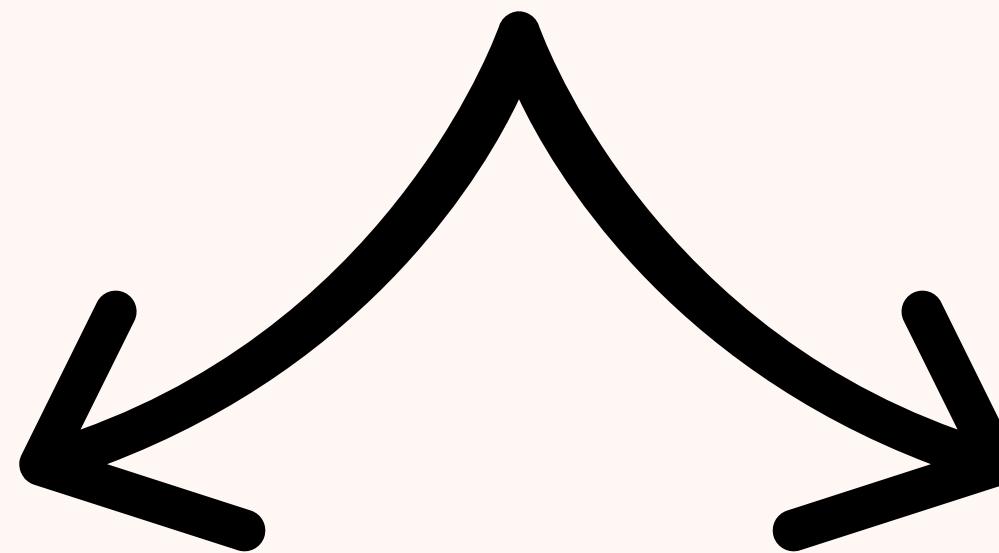
ตัวอย่างข้อมูลใน ml-1m.test.rating

Data > ml-1m.test.rating				
1	0	25	5	978824351
2	1	133	3	978300174
3	2	207	4	978298504
4	3	208	4	978294282
5	4	222	2	978246585
6	5	396	5	978239019
7	6	74	3	978234898
8	7	91	3	978247143
9	8	514	3	978226678
10	9	659	5	980638688

# EVALUATION PROTOCOLS

THE PERFORMANCE OF A RANKED LIST IS JUDGED BY

HIT RATIO  
(HR)



NORMALIZED  
DISCOUNTED  
CUMULATIVE GAIN  
(NDCG)

# EVALUATION PROTOCOLS

## HIT RATIO (HR)

อัตราส่วนระหว่างจำนวนผู้ใช้ที่ได้รับการแนะนำให้กับจำนวนผู้ใช้ทั้งหมดในชุดทดสอบ

$$HR = \frac{|U_{hit}^L|}{|U_{all}|}$$

$|U_{hit}|$  --> จำนวนผู้ใช้ที่ระบบสามารถแนะนำรายการที่ถูกต้องได้ภายใน L อันดับแรกของรายการแนะนำ

$|U_{all}|$  ----> จำนวนผู้ใช้ทั้งหมดที่มีอยู่ในชุดข้อมูลทดสอบ



ค่า HR ที่ได้จะอยู่ระหว่าง 0 ถึง 1 ยิ่งเข้าใกล้ 1 ยิ่งมีประสิทธิภาพ



- HIT RATIO จะสูงขึ้นตามไปด้วย
- ถ้าค่า L สูงเกินไป อาจทำให้คุณภาพของคำแนะนำลดลง เนื่องจากอาจรวมตัวเลือกที่ไม่เกี่ยวข้องมีความแม่นยำต่ำ



- HIT RATIO น้อยลง
- ประสิทธิภาพโดยรวมของระบบแนะนำลดลงผู้ใช้อาจไม่ได้รับคำแนะนำที่ตรงกับความต้องการ



การหาจุดสมดุลของค่า L จึงเป็นสิ่งสำคัญในการออกแบบระบบแนะนำที่มีประสิทธิภาพ

# EVALUATION PROTOCOLS



## NORMALIZED DISCOUNTED CUMULATIVE GAIN (NDCG)

- เป็นตัวชี้วัดที่ใช้ประเมินคุณภาพของระบบแนะนำ โดยพิจารณาทั้งความถูกต้องและลำดับความสำคัญของรายการที่แนะนำ
- หลักการสำคัญของ NDCG คือการให้น้ำหนักกับตำแหน่งของรายการที่แนะนำ โดยรายการที่อยู่ในลำดับต้นๆ จะมีความสำคัญมากกว่ารายการที่อยู่ลำดับท้าย

$$CG(k) = \sum_{i=1}^k G_i$$

$$DCG(k) = \sum_{i=1}^k \frac{G_i}{\log_2(i+1)}$$

$$IDCG(k) = \sum_{i=1}^{|I(k)|} \frac{G_i}{\log_2(i+1)}$$

$$NDCG = \frac{DCG}{IDCG}$$

การรวมค่าความเกี่ยวข้อง (RELEVANCE SCORES) ของรายการทั้งหมดตั้งแต่ตำแหน่งที่ 1 ถึง K โดยไม่คำนึงถึงลำดับความสำคัญของตำแหน่ง เป็นวิธีการพื้นฐานที่สุดในการวัดความแม่นยำของระบบแนะนำ



ข้อเสียของ CUMULATIVE GAIN คือไม่ได้มีการเอาตำแหน่งของ การ RANKING มาคิด

เป็นการปรับปรุงจาก CG โดยเพิ่มตัวหารที่เป็นพังก์ชันส์ของการทึบ เพื่อลดน้ำหนักของรายการที่อยู่ในลำดับท้ายๆ เนื่องจากผู้ใช้มักให้ความสนใจกับรายการในลำดับแรกๆ มากกว่า ทำให้การวัดสะท้อนพฤติกรรมจริงของผู้ใช้มากขึ้น

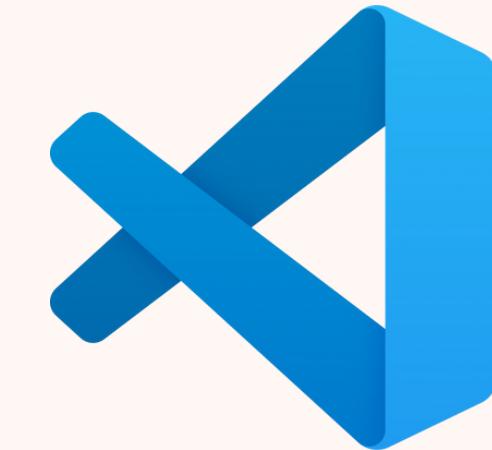
คำนวณเหมือน DCG แต่ใช้กับรายการที่จัดเรียงในลำดับที่ดีที่สุดที่เป็นไปได้ (IDEAL RANKING) โดย  $I(K)$  คือรายการที่เรียงลำดับอย่างสมบูรณ์แบบจนถึงตำแหน่งที่ K และ  $|I(K)| = K$  คือจำนวนรายการทั้งหมด

เป็นการนำค่า DCG มาปรับให้เป็นมาตรฐานโดยการหารด้วย IDCG ทำให้ได้ค่าที่อยู่ในช่วง 0 ถึง 1 ค่า NDCG ที่เข้าใกล้ 1 แสดงว่าระบบสามารถจัดลำดับรายการได้ใกล้เคียงกับลำดับที่ดีที่สุด

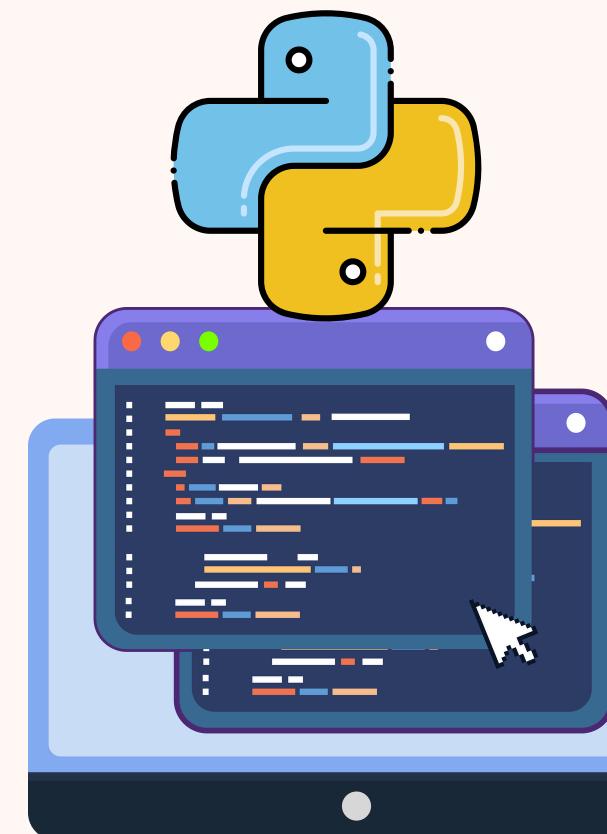
# TOOL



**Docker**  
ใช้ **Docker** เพื่อช่วยเพิ่มประสิทธิภาพ  
ในการ **Run Code**



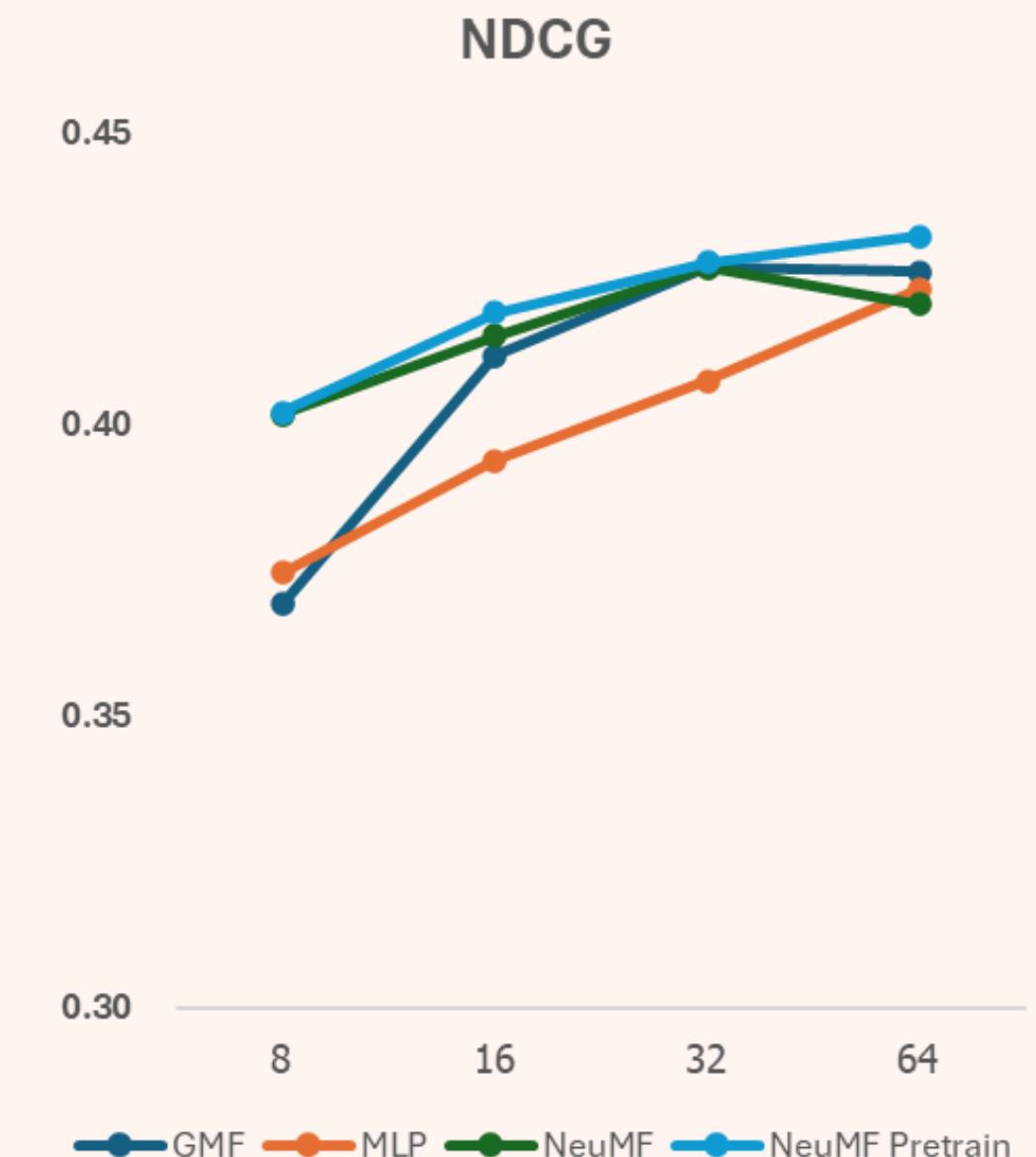
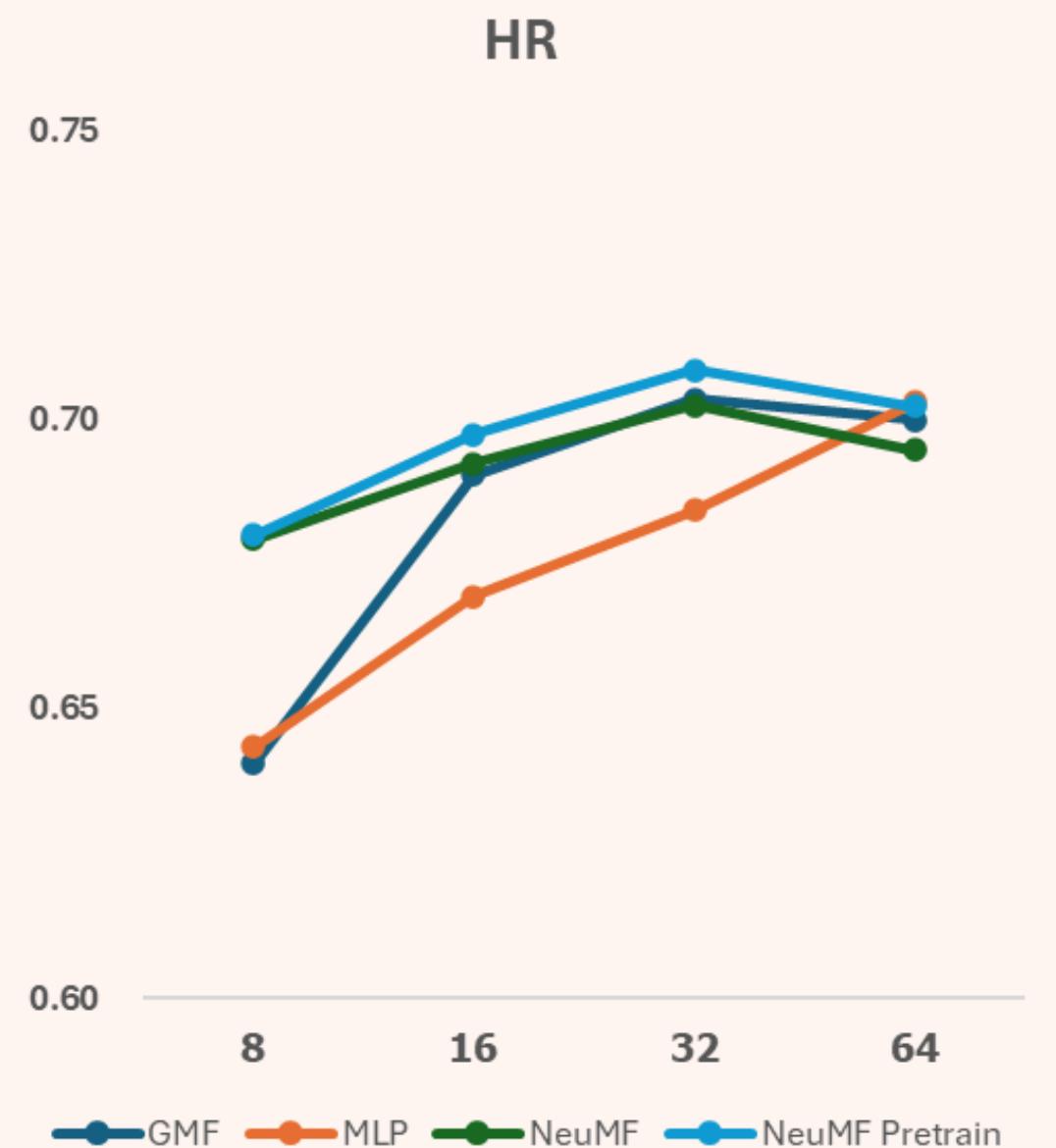
**VS Code**  
• ใช้สำหรับปรับปรุงแก้ไข **Python Code**  
• ใช้อ่าน **File Input .rating** และ  
**Output .h5**



# REPRODUCE

อันดับแรก สามารถเห็นได้ว่า NeuMF Pretrain มีประสิทธิภาพดีที่สุดในชุดข้อมูล MovieLens รองลงมา คือ NeuMF (Without Pretrain)

อันดับที่สอง วิธี NCF อื่น ๆ อีกสองวิธี ได้แก่ GMF และ MLP ก็แสดงประสิทธิภาพที่แข็งแกร่งเช่นกัน โดย MLP มีผลที่ดีอยกว่า GMF พoS ณ ควรอย่างไร ก็ตาม MLP สามารถปรับปรุงได้เพิ่มเติมโดยการเพิ่ม Hidden Layer และในที่นี้แสดงผลลัพธ์ของ MLP ที่มี 3 Layer เท่านั้น

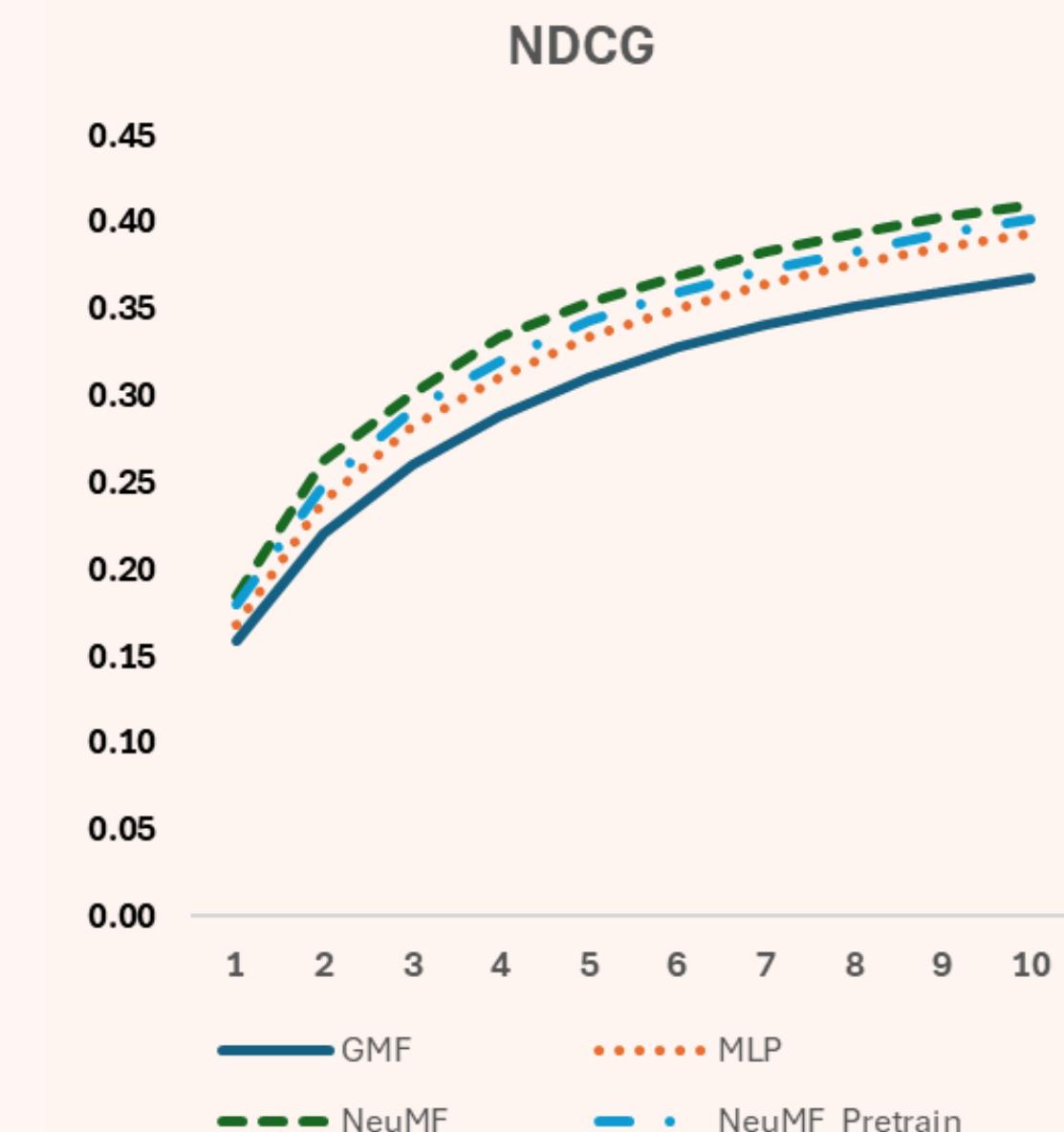
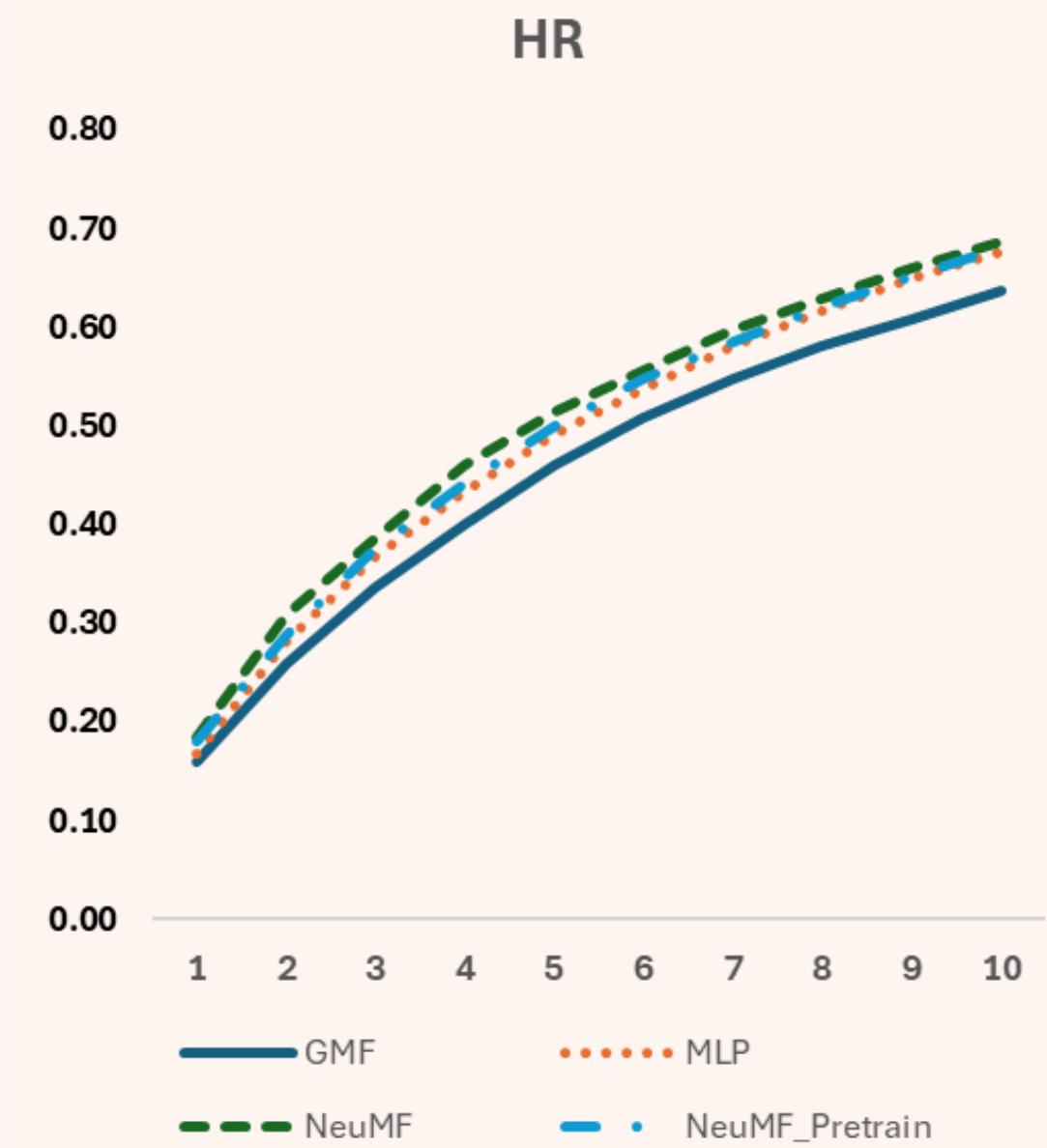


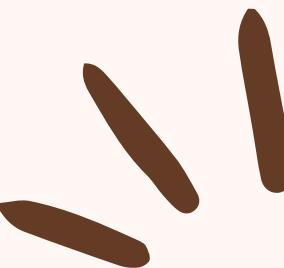


# REPRODUCE

ค่า Top-K ต่อ HR และ NDCG เมื่อค่า Top-K เพิ่มขึ้น (จาก 1 → 10)

- ค่า HR และ NDCG ของทุกโมเดลมีแนวโน้มเพิ่มขึ้นอย่างต่อเนื่อง เมื่อค่า TopK เพิ่มขึ้น
- ค่า Top-K ที่เพิ่มขึ้นช่วยเพิ่มประสิทธิภาพของโมเดล ในการแนะนำไอเท็ม (ทั้ง HR และ NDCG)
- การเติบโตของผลลัพธ์จะชะลอตัว เมื่อ Top-K สูงเกินไป → ต้องเลือกค่า Top-K ให้เหมาะสม

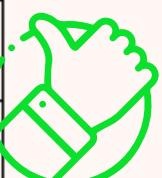




# REPRODUCE

NeuMF Without Pretrain vs NeuMF with Pretrain

HR		
Factor	NeuMF	NeuMF Pretrain
8	0.68	0.68
16	0.69	0.70
32	0.70	0.71
64	0.69	0.70

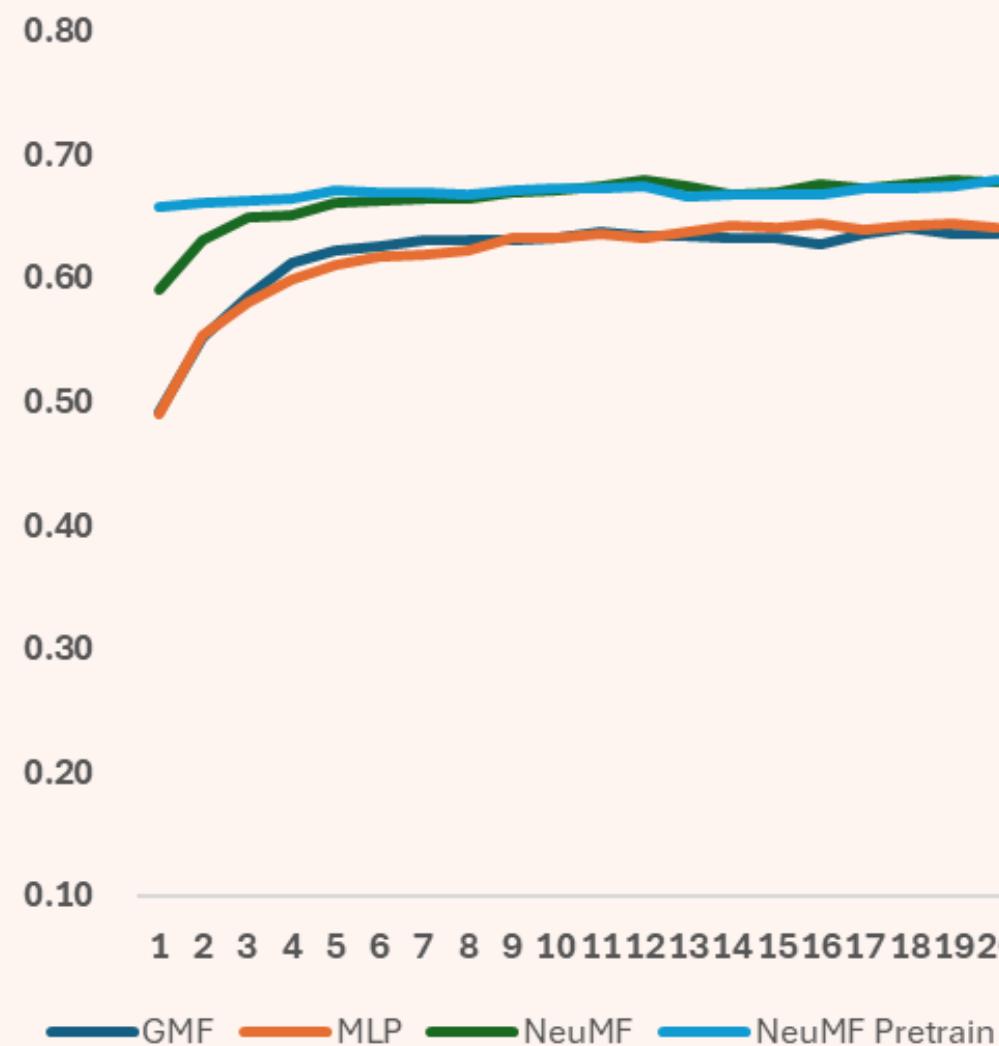


NDCG		
Factor	NeuMF	NeuMF Pretrain
8	0.40	0.40
16	0.42	0.42
32	0.43	0.43
64	0.42	0.43

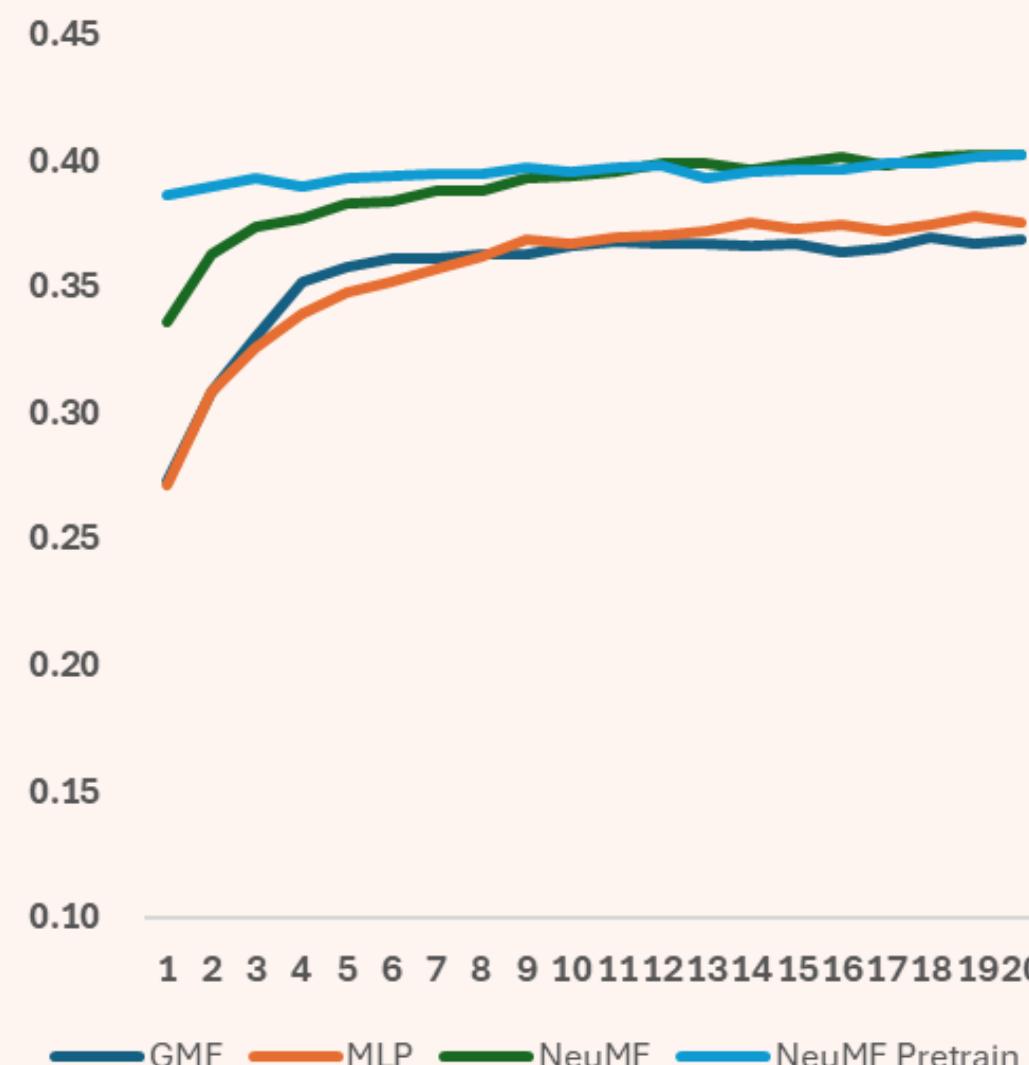


# REPRODUCE

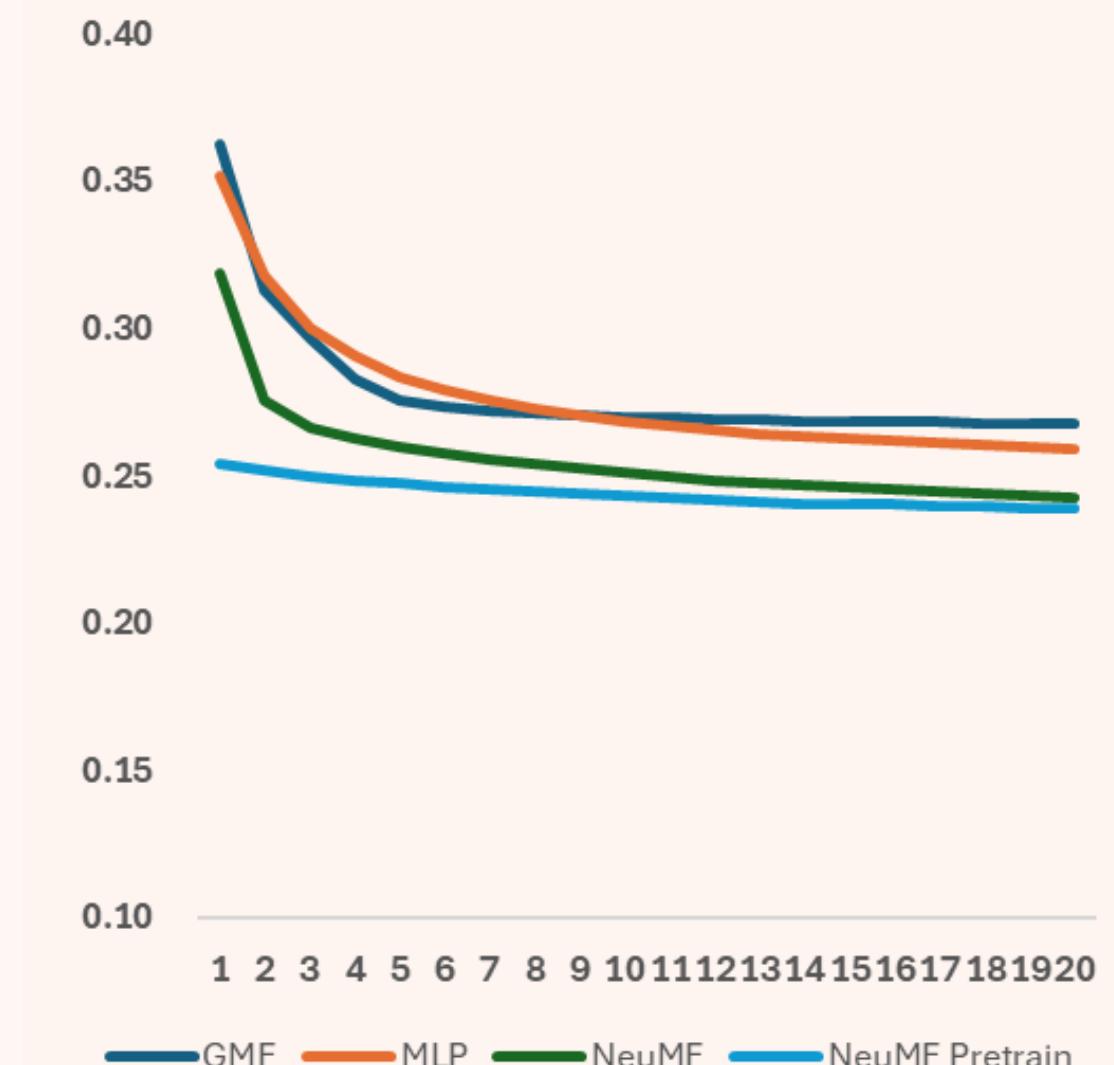
HR



NDCG



Traning loss

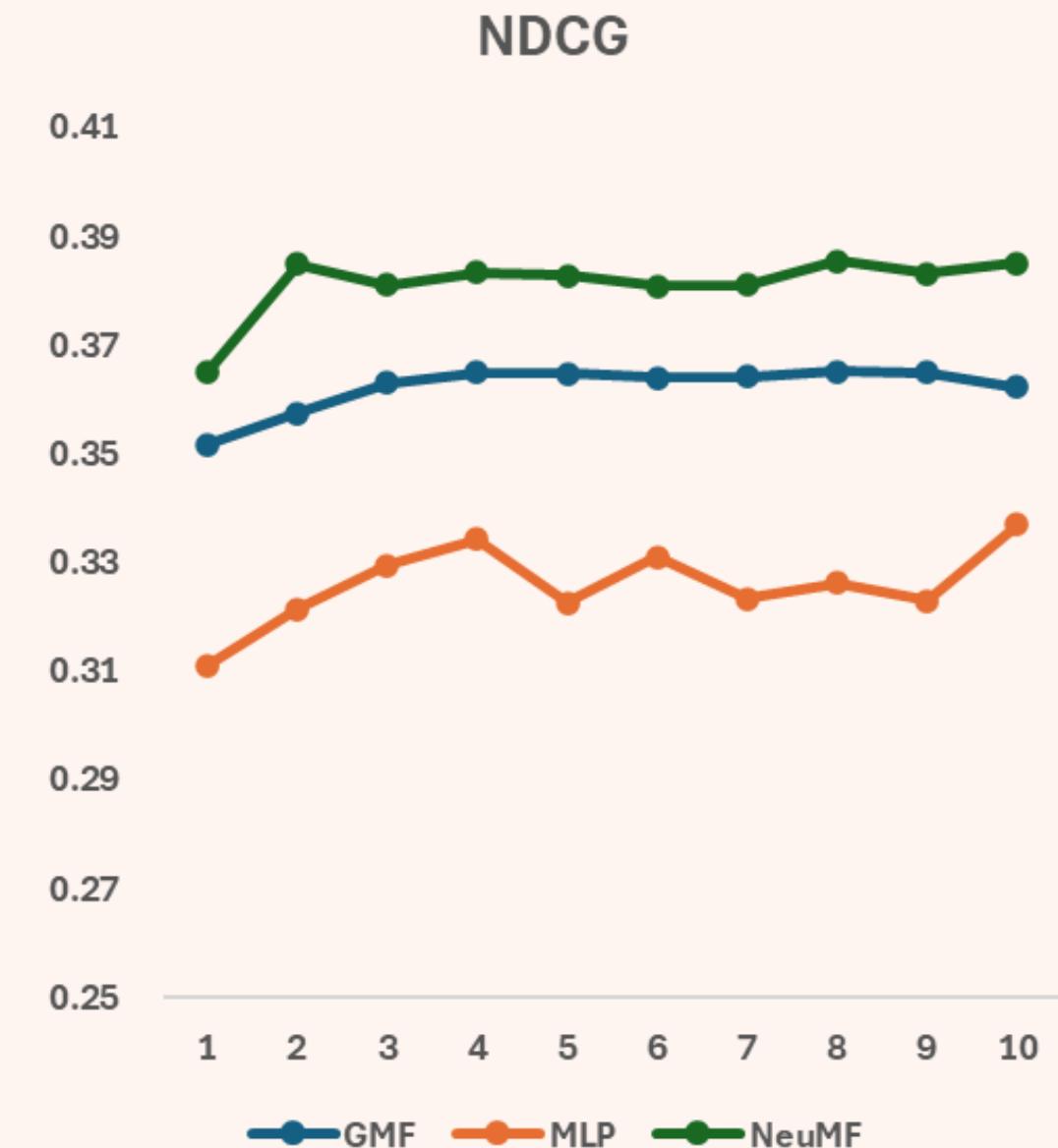
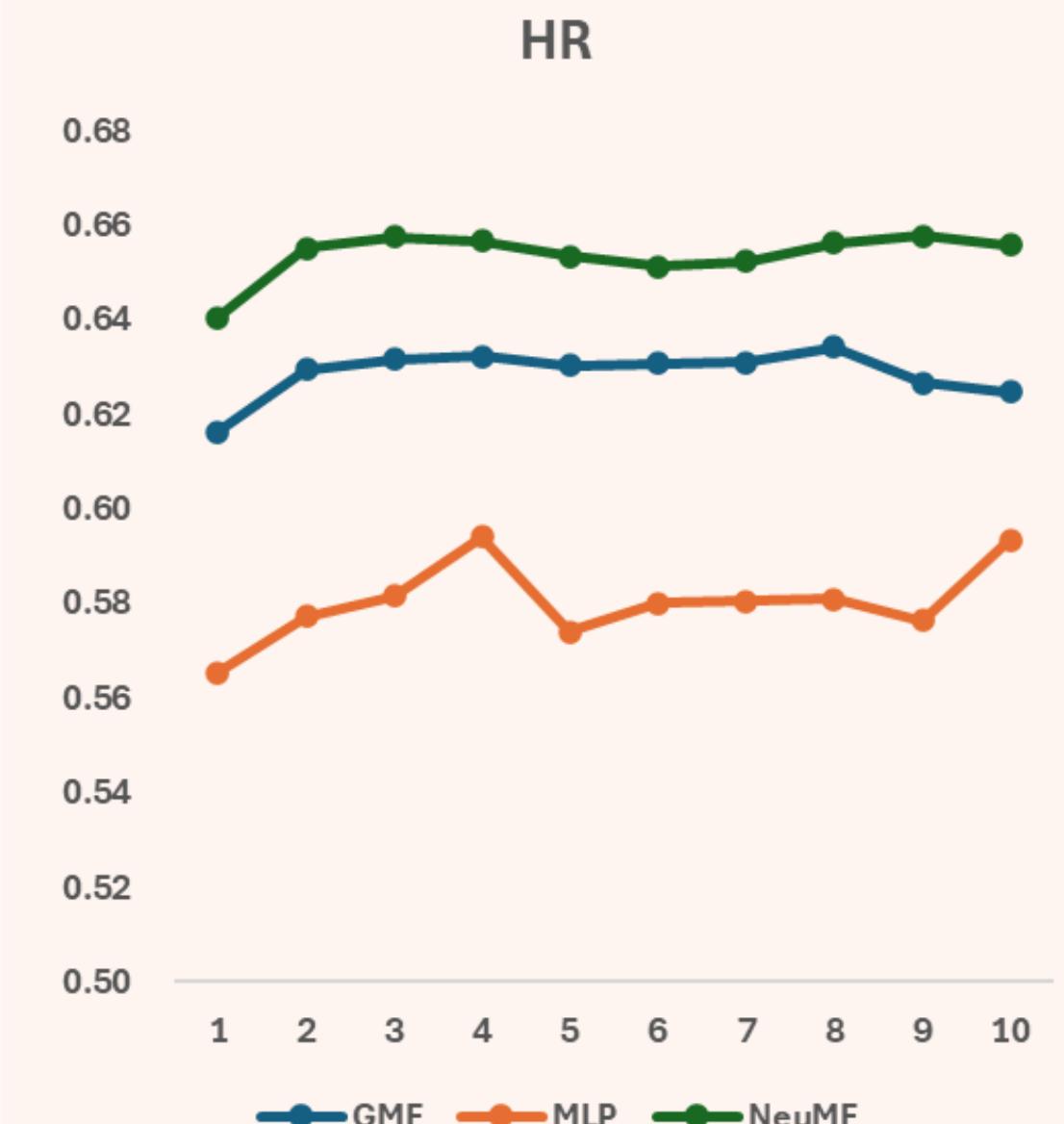


# REPRODUCE

Number of Negative Samples per Positive Instance (factors=8)

สามารถเห็นได้ชัดว่า NeuMF มีประสิทธิภาพดีที่สุดในชุดข้อมูล MovieLens รองลงมา คือ GMF ส่วน MLP มีประสิทธิภาพน้อยที่สุด

อัตราส่วนการสุ่มตัวอย่างที่เหมาะสมจะอยู่ที่ประมาณ 2 ถึง 4 พบว่าเมื่ออัตราส่วนการสุ่มตัวอย่างเกินกว่า 4 ประสิทธิภาพของวิธี NCF เริ่มลดลง ซึ่งแสดงให้เห็นว่าการตั้งอัตราส่วนการสุ่มตัวอย่างที่สูงเกินไปอาจส่งผลเสียต่อประสิทธิภาพของโมเดล



# WITH DIFFERENT LAYER

HR				
Factor	MLP-1	MLP-2	MLP-3	MLP-4
8	0.45	0.59	0.63	0.67
16	0.45	0.63	0.65	0.68
32	0.45	0.66	0.68	0.69
64	0.45	0.68	0.70	0.71

NDCG				
Factor	MLP-1	MLP-2	MLP-3	MLP-4
8	0.25	0.33	0.37	0.39
16	0.25	0.37	0.38	0.41
32	0.25	0.39	0.40	0.42
64	0.25	0.41	0.42	0.42

# PROPOSE AND IMPLEMENT IMPROVEMENTS

## แนวคิด (Idea)

เพื่อเพิ่มประสิทธิภาพของโมเดล NeuMF จึงเสนอแนวคิดในการทดลองเปลี่ยน Activation Function ในส่วนของ MLP จาก ReLU ไปเป็นฟังก์ชันอื่น ๆ ที่มีความสามารถในการเรียนรู้ลักษณะข้อมูลที่ซับซ้อนมากขึ้น

## (Expected Results)

- คาดว่า Swish และ PReLU จะให้ผลลัพธ์ดีกว่า ReLU เนื่องจากมีความต่อเนื่องของ Gradient และความยืดหยุ่นในการเรียนรู้
- Leaky ReLU อาจลดปัญหา Dead Neurons ได้ดีกว่า ReLU
- ELU น่าจะให้ความเสถียรในการเรียนรู้ที่ดีกว่า

### ReLU (Rectified Linear Unit)

- ให้ค่า  $x$  เมื่อ  $x > 0$  และ 0 เมื่อ  $x \leq 0$
- ✓ คำนวณเร็ว, ลดปัญหา Vanishing Gradient
- ✗ อาจเกิด Dead Neurons (หยุดทำงาน)

### Leaky ReLU

- เมื่อ ReLU แต่ต้องมีการกำหนดค่า  $\alpha$
- ✓ ลด Dead Neurons,
- ✗ ต้องปรับค่า  $\alpha$  ให้เหมาะสม
- สม

### PReLU (Parametric ReLU)

- เมื่อ Leaky ReLU แต่ค่า  $\alpha$  สามารถเรียนรู้เองได้ระหว่างการฝึก (Training)
- ✓ ยืดหยุ่นสูง, ลด Dead Neurons
- ✗ เพิ่มพารามิเตอร์ เสี่ยง Overfitting

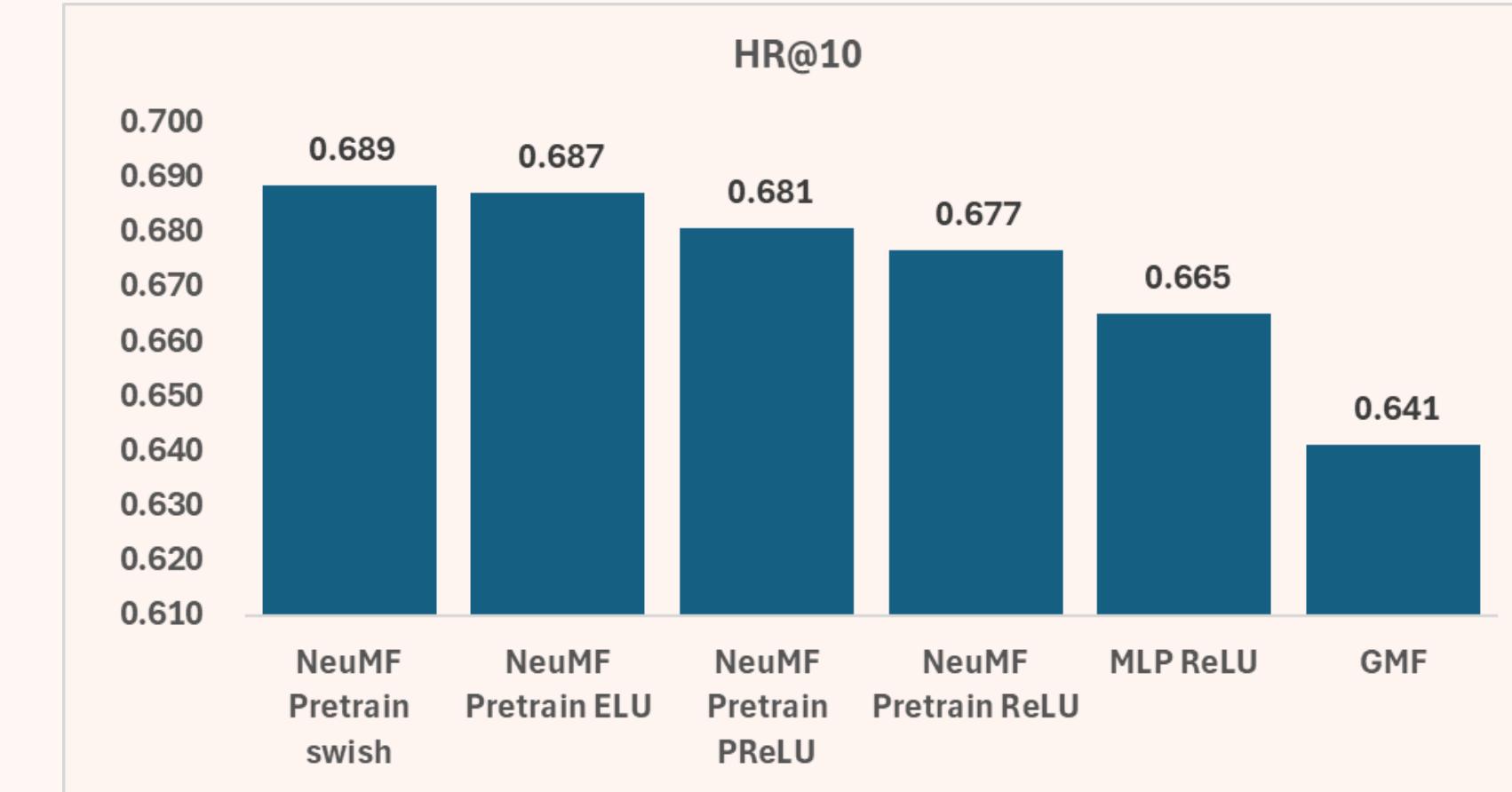
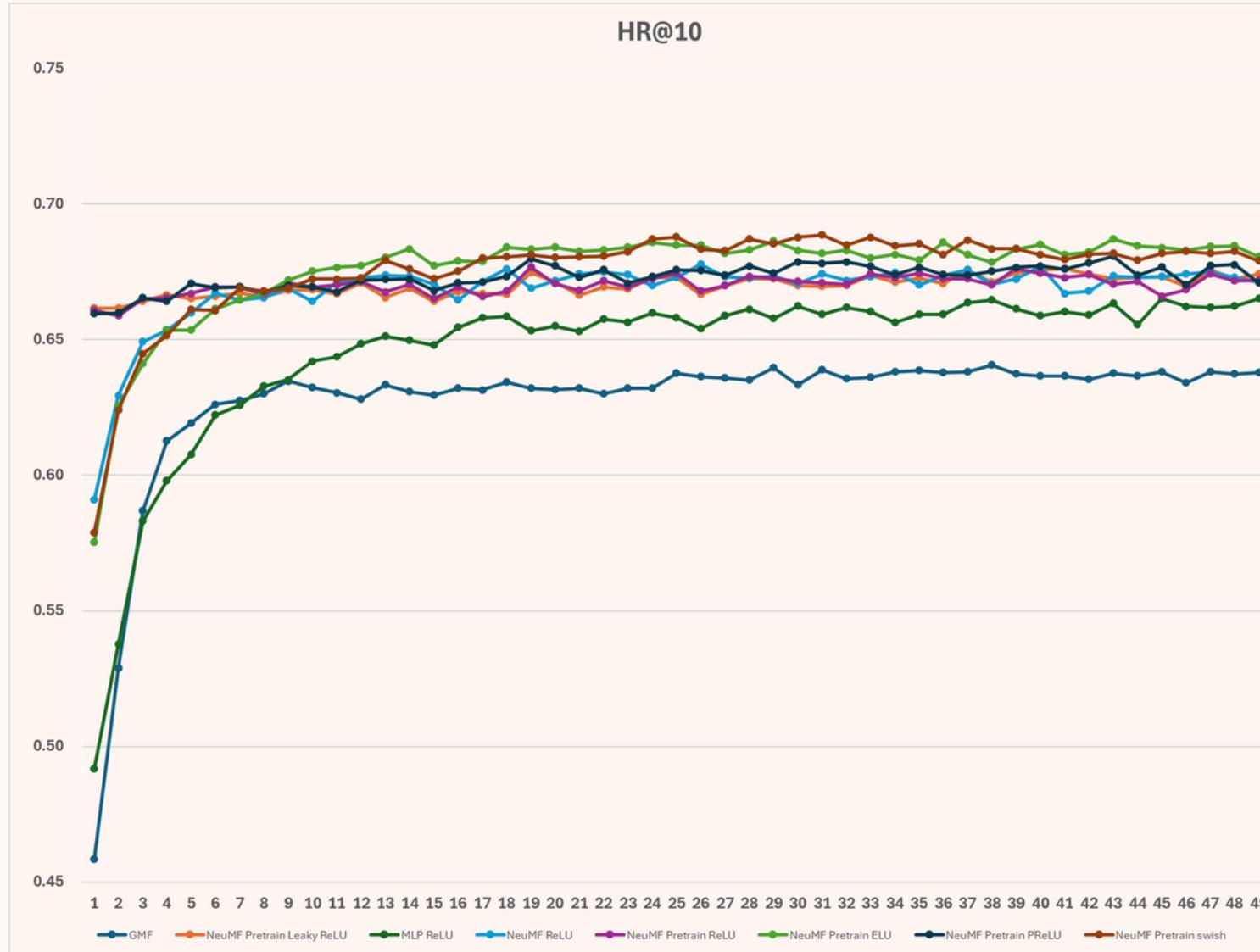
### ELU (Exponential Linear Unit)

- $x > 0$  เป็น  $x$ ,  $x \leq 0$  เป็น  $\alpha(e^x - 1)$
- ✓ แก้ปัญหา Dying ReLU, Robust ต่อ Noise
- ✗ คำนวณช้ากว่า ReLU

### Swish

- $f(x) = x \cdot \sigma(x)$  ( $\sigma(x)$  คือ Sigmoid)
- ✓ Non-monotonic, Smooth
- ✗ คำนวณช้า เพราะต้องคำนวณ Sigmoid

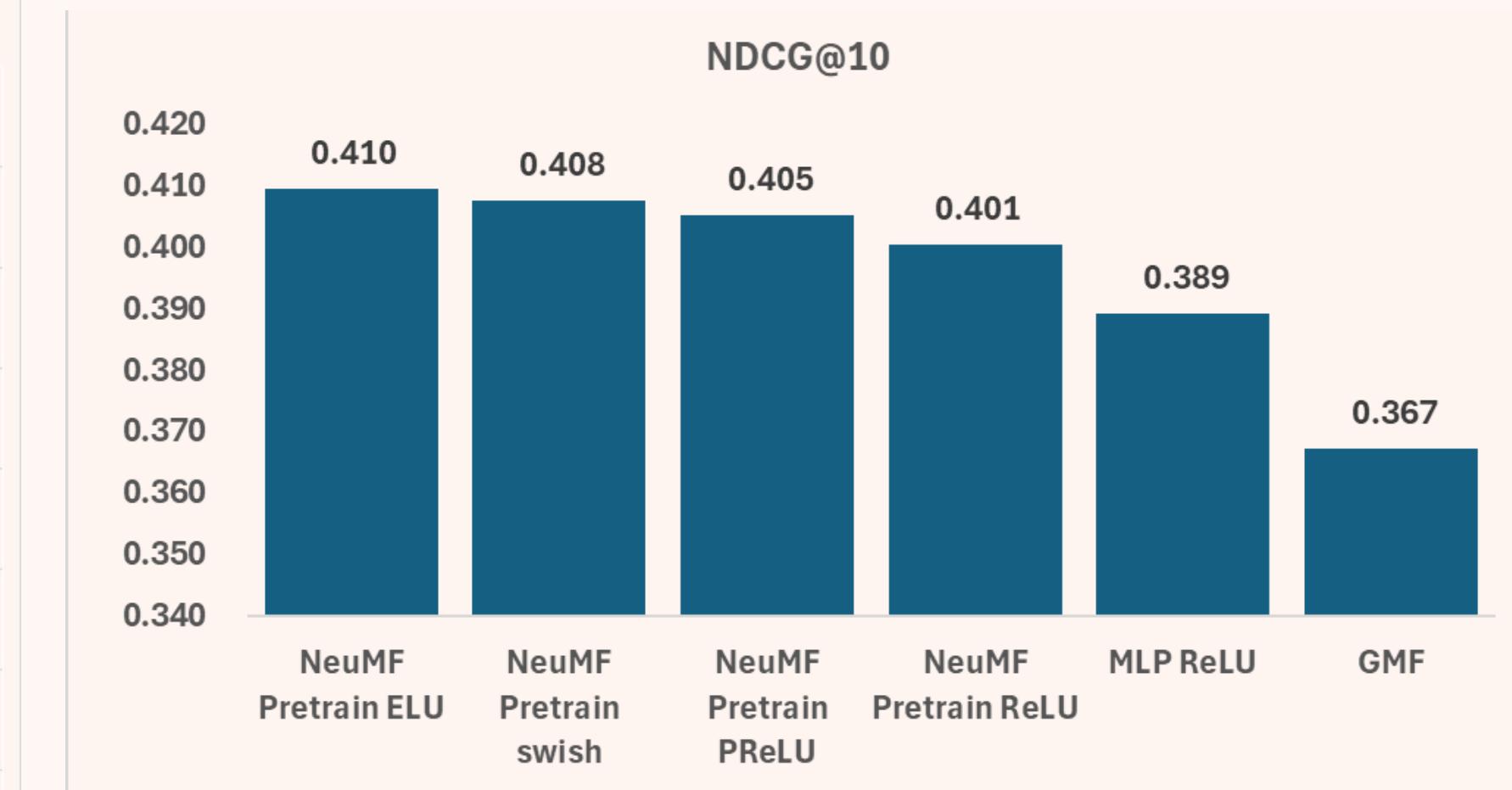
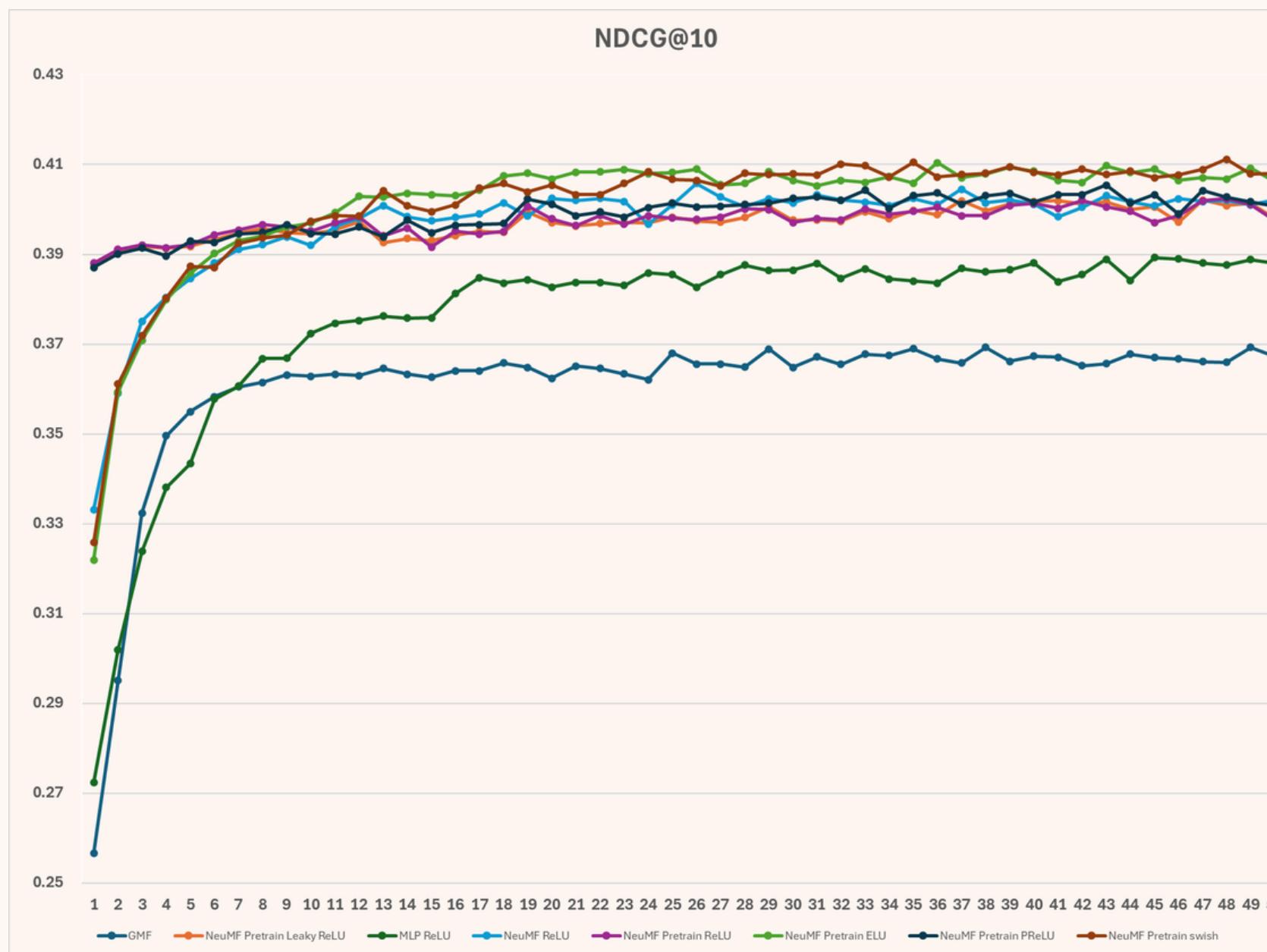
# PROPOSE AND IMPLEMENT IMPROVEMENTS



## ผลทดสอบ HR@10

- Swish เป็น Activation Function กี่ช่วยให้โมเดล NeuMF มีประสิทธิภาพสูงสุด
- ELU และ PReLU มีความสามารถในการเรียนรู้ที่ดี รองจาก Swish
- ReLU แม้จะเป็นฟังก์ชันมาตรฐาน แต่ยังสู้ฟังก์ชันกี่ซับซ้อนไม่ได้
- GMF และ MLP ReLU มีความสามารถในการแบ่งนำ้อยกว่าโมเดลอื่น ๆ อย่างชัดเจน

# PROPOSE AND IMPLEMENT IMPROVEMENTS



## ผลทดสอบ NDCG@10

- ELU เป็น Activation Function ที่ช่วยให้โมเดล NeuMF มีประสิทธิภาพสูงสุด
- Swish และ PReLU มีความสามารถในการเรียนรู้ที่ดี รองจาก Swish
- ReLU แม้จะเป็นฟังก์ชันมาตรฐาน แต่ยังสู้ฟังก์ชันที่ซับซ้อนไม่ได้
- GMF และ MLP ReLU มีความสามารถในการแบนนำอยกว่าโมเดลอื่น ๆ อย่างชัดเจน



# EXAMPLES OF MODEL IMPLEMENTATION AND USAGE

Getting recommendations for user 1:  
1. Movie ID: 1344  
2. Movie ID: 1061  
3. Movie ID: 515  
4. Movie ID: 743  
5. Movie ID: 265

ให้แนะนำภาพยนตร์ที่เหมาะสมกับ  
User ID 1

Finding similar movies to movie 265:  
1. Movie ID: 475, Similarity Score: 0.723  
2. Movie ID: 438, Similarity Score: 0.692  
3. Movie ID: 1915, Similarity Score: 0.691  
4. Movie ID: 593, Similarity Score: 0.649  
5. Movie ID: 460, Similarity Score: 0.629

ให้แนะนำภาพยนตร์  
ที่มีความคล้ายกันกับ  
Movie ID 265

Cluster 0:  
Number of users: 595  
Top 5 recommended movies for this cluster:  
1. Movie ID: 2825  
2. Movie ID: 90  
3. Movie ID: 2824  
4. Movie ID: 78  
5. Movie ID: 576

Cluster 1:  
Number of users: 1542  
Top 5 recommended movies for this cluster:  
1. Movie ID: 2748  
2. Movie ID: 1740  
3. Movie ID: 1602  
4. Movie ID: 2304  
5. Movie ID: 2824

แสดงกลุ่มผู้ใช้และภาพยนตร์ที่  
แนะนำสำหรับกลุ่มนั้น

```
== Model Information ==  
Number of users: 6040  
Number of items: 3706  
Embedding dimension: 32  
=====  
== Movie Recommendation System ==  
1. Get user recommendations  
2. Find similar movies  
3. Cluster users  
4. Exit  
Enter your choice (1-4):
```

# REFERENCE

1. [HTTPS://GITHUB.COM/HEXIANGNAN/NEURAL\\_COLLABORATIVE\\_FILTERING](https://github.com/hexiangnan/neural_collaborative_filtering)
2. [HTTPS://MEDIUM.COM/P/23499E234289](https://medium.com/p/23499e234289)
3. [HTTPS://TOWARDSDATASCIENCE.COM/RANKING-EVALUATION-METRICS-FOR-RECOMMENDER-SYSTEMS-263D0A66EF54](https://towardsdatascience.com/ranking-evaluation-metrics-for-recommender-systems-263d0a66ef54)

# THANK you