TwHIN: Embedding the Twitter Heterogeneous Information Network for Personalized Recommendation

Leah Lu, Sr. Data Scientist

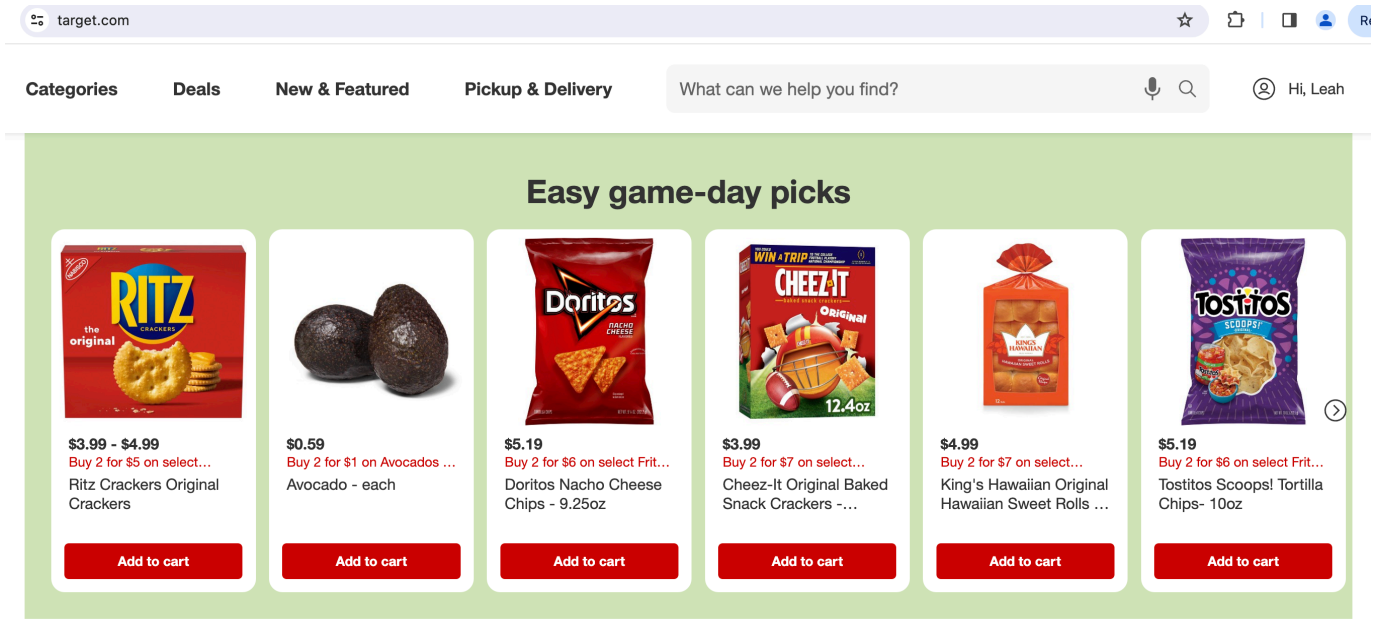
2024 – 02 - 06

KDD '22: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, August 2022, Pages 2842–2850 https://doi.org/10.1145/3534678.3539080

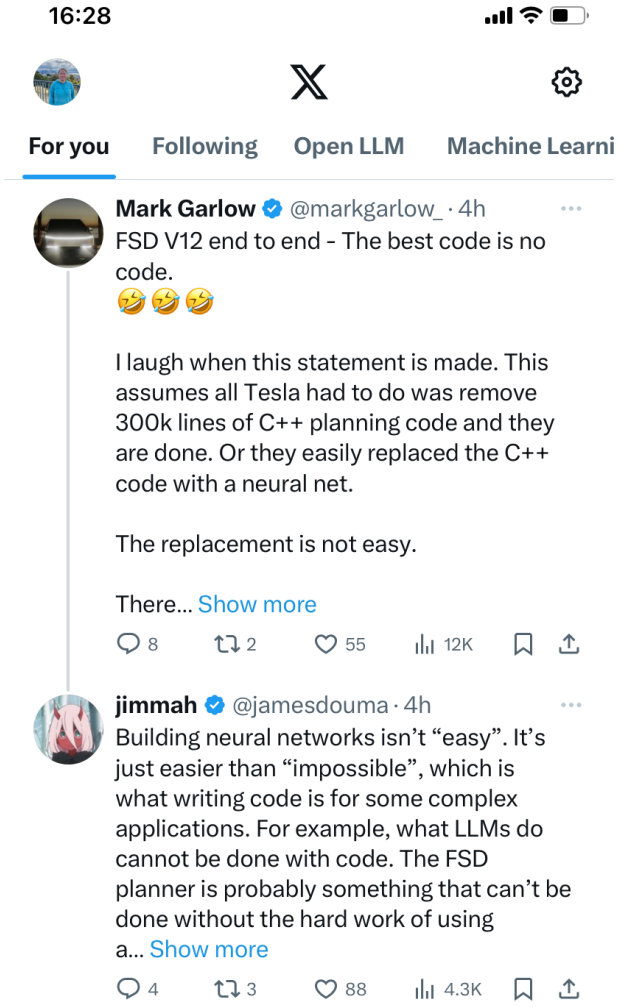
Outline

- Background
- TwHIN Embeddings Development
- Experiments and Results for Recommendation
- Practical Considerations
- Miscellaneous:
  - Twitter's Recommendation Algo Open Source
  - Twitter Architecture 2022
- Q&A
- References

Personalized Recommendation



Target Recommendation items

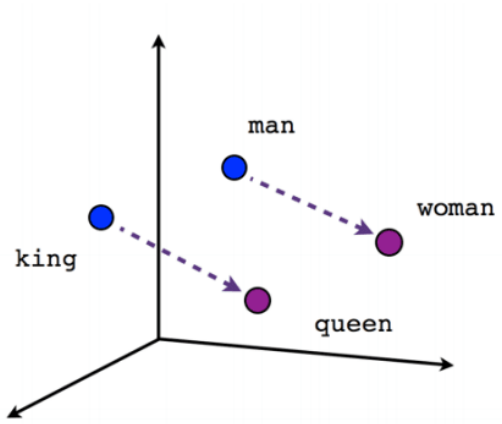


'For You' at X, also called Twitter

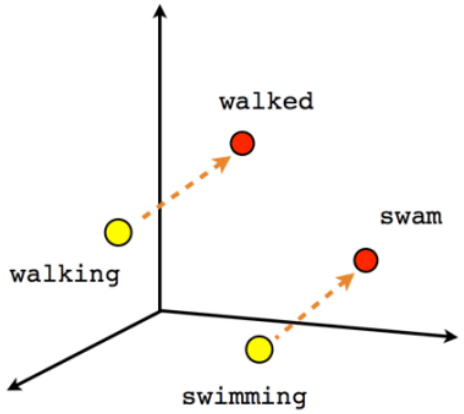
- Algorithms-driven to suggest a small number of interesting items out of millions for each user and critical to effectively explore user' interest.

Embeddings

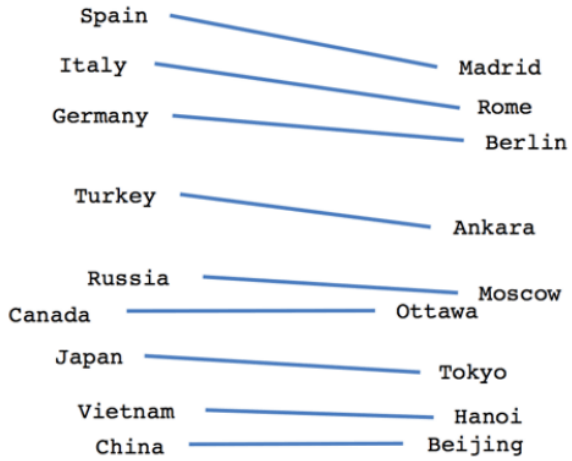
- Embedding started with Word2Vec (2013, Google), credited the Test of Time Award at the NeurIPS Conference 2023.



Male-Female



Verb tense

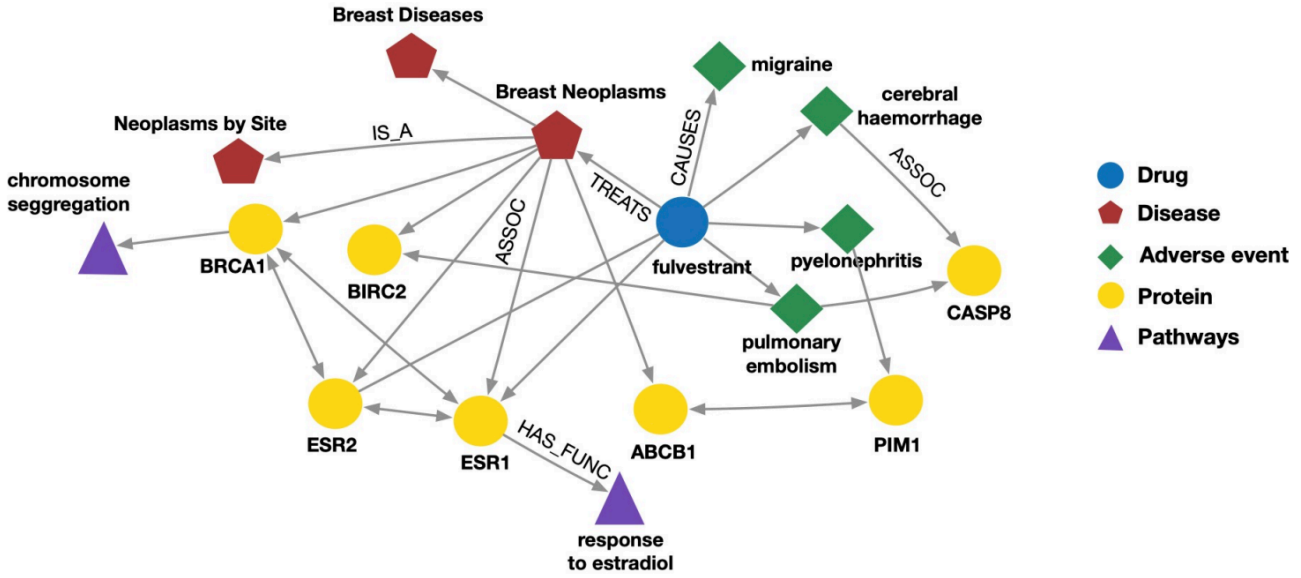


Country-Capital

e.g. king = [-0.74501 , -0.11992 , 0.37329 , 0.36847, ….];
queen = [-1.1266 , -0.52064 , 0.45565 , 0.21079 , ….]

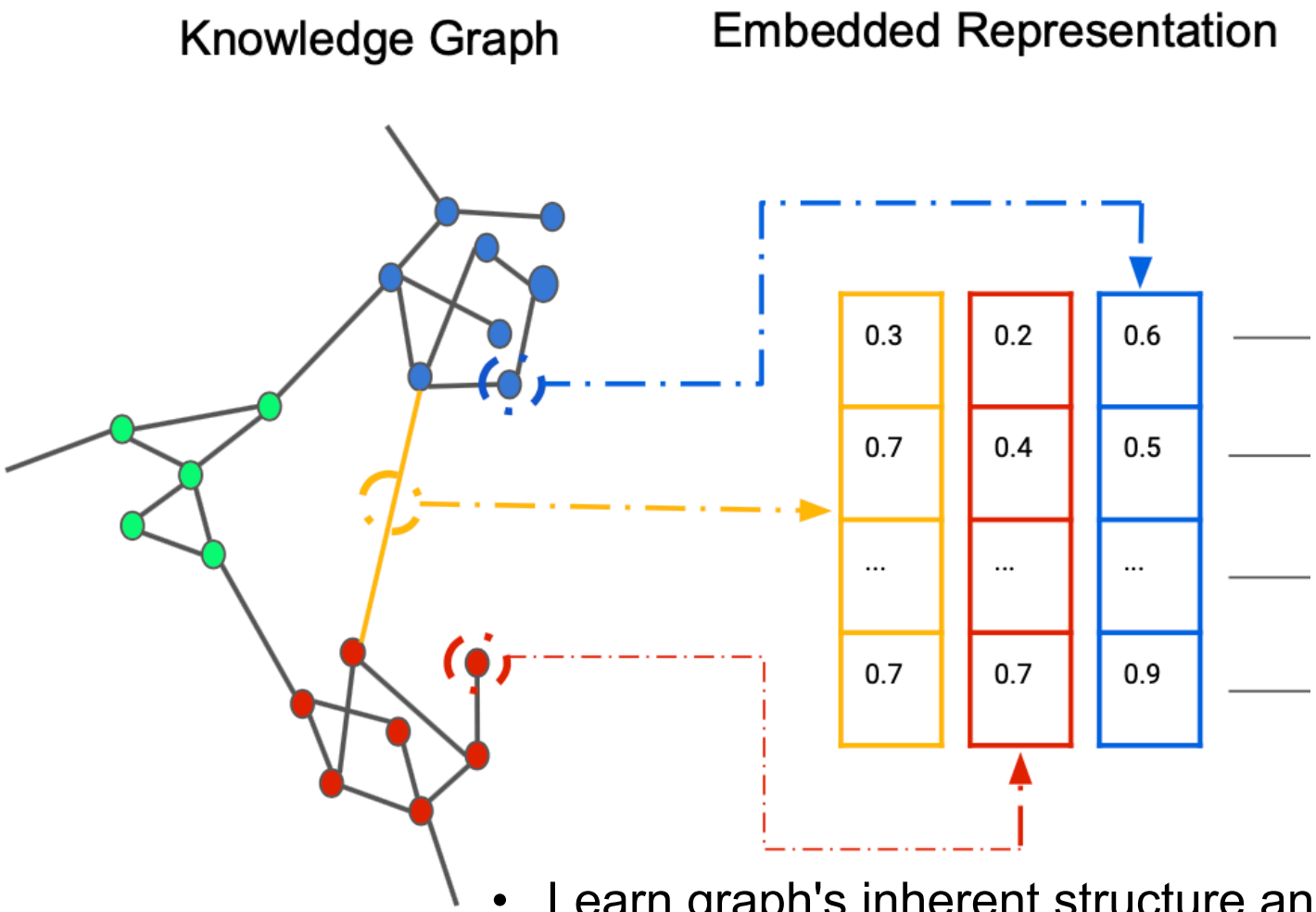
Graph: entities (nodes) with relations

Social graph (Credit: Medium)



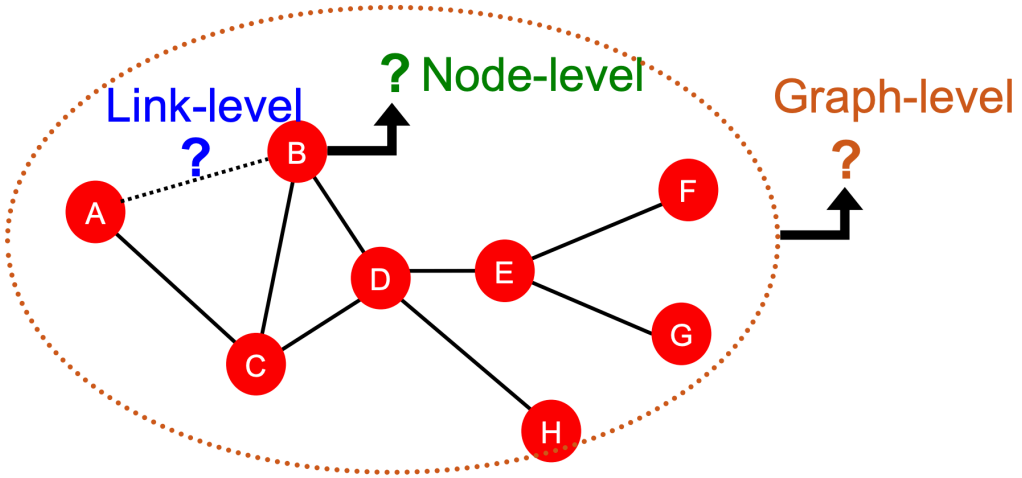
Knowledge graph: nodes, edges, relationship representing real-world knowledge, G = {entity, relation, entity'}

Knowledge Graph Embedding



Machine Learning Task

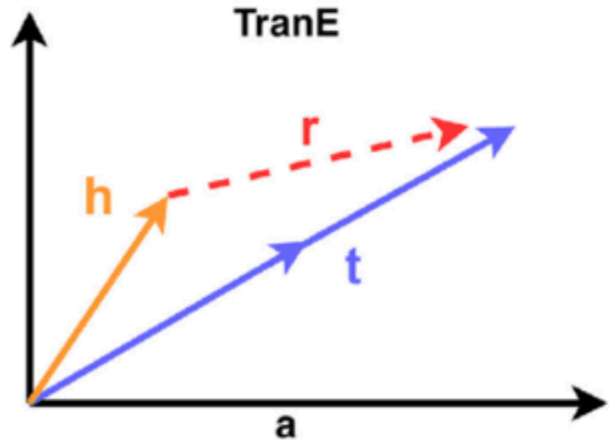
- Node-level prediction
- Link-level prediction
- Graph-level prediction

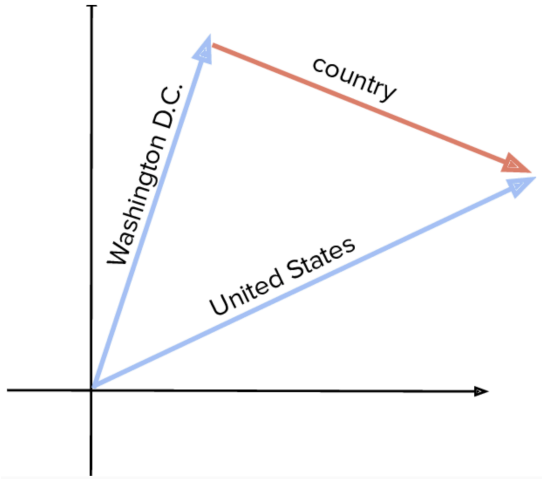


- Learn graph's inherent structure and the semantics of its relationships and present nodes/edges as vectors

TransE (Translation Embeddings)

- **Illustration for TransE**:
  - For {**h**ead, **r**elation, **t**ail} with embedding vectors
  - h+r≈t if a link exists
  - Entity scoring function: f(h, t) = ||h + r – t||
- **Loss Function**: maximize difference between positive triples (true facts) and negative triples (false facts)
- **Negative Sampling**: (*h*′,*r*,*t*) or (*h*,*r*,*t*′) vs (*h*,*r*,*t*) (Positive)





Twitter Heterogeneous Information Network

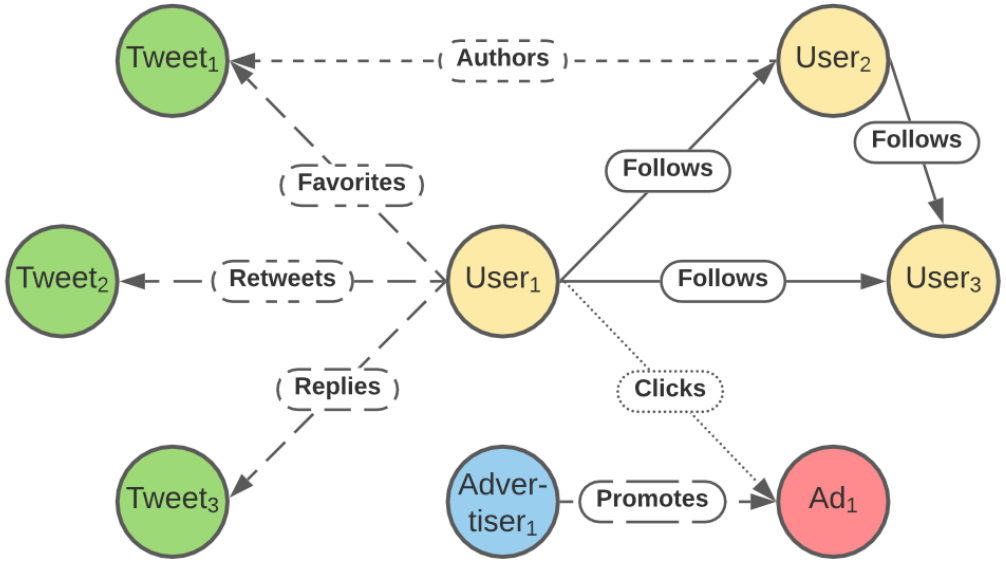


Figure 2: An example heterogeneous information network (HIN) where $|V| = 8$ and $|E| = 9$. There are four entity types ($\mathcal{T}$): 'User', 'Tweet', 'Advertiser', and 'Ad'. There are seven types of relationship ($\mathcal{R}$): 'Follows', 'Authors', 'Favorites', 'Replies', 'Retweets', 'Promotes', and 'Clicks'. See Section 3 for more details.

Characteristics:

- multi-type and multi-relational data, encode valuable information about social network entities not fully captured by a single relation
- 4 Entities: User, Tweet, Advertiser, Ad
- 7 types of relationship

Innovations:

- Data Supplementation
- Task Reusability

TwHIN Embedding Development

Input data: 10^9 nodes and 10^11 edges
Relationship: social signals (follow-graph), content engagement signals (Tweet, image, and video engagement), advertisement engagements

TransE and Loss function

Computational Implementation

Algorithm 1: HIN Embedding

Input: $G = (V, E, \phi, \psi)$, $epochs$, P
Output: θ (learned embeddings)

let θ: initialized embedding vectors entities in V relations in $\mathcal{R}$
for *each* $\{1 \ldots epochs\}$ **do**
  for *each bucket* $(i, j) : 0 \leq i < P;\ 0 \leq j < P$) **do**
    load bucket $B_{i,j}$ edges onto memory
    load $\{\theta_v : \pi(v) = i \vee \pi(v) = j\}$ onto GPU
    train embeddings on edges using Equation 1
  end
end

TwHIN: deriving two high-coverage networks (1) follow-graph and (2) User-Tweet engagement graph.

Inductive multi-modal embeddings

- Shortcomings in considerations
  - **Single Embedding Limitation**: Entities are limited to single embedding vector when "multi-modal interests.”
  - **Transductive Learning Limitation**: embeddings for entities are transductive from knowledge graph, but new entities (referred to as "out-of-vocabulary nodes") have no embeddings

- Solutions for post-processing
  - **Inductive Embedding of New or Cold-Start Nodes**: infer embeddings for new nodes from the entities in the graph
  - **Cluster Existing Unimodal Embeddings and Compute Multiple Embeddings for a Node**: generate ‘multi-modal’ embeddings from the clusters

Evaluation on TwHIN embeddings

- Advancement in methodologies
- Cold-start for new nodes
- Optimization in practice
- Computational Consideration and Scaling

TwHIN Application for Recommendation

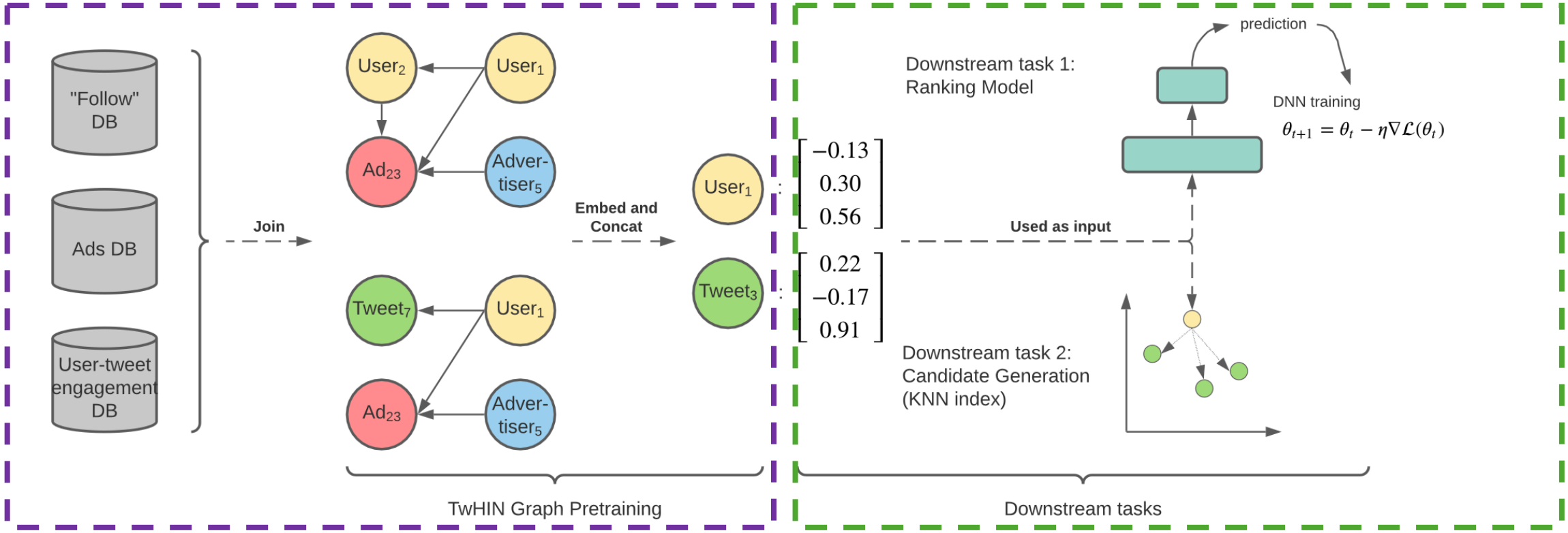


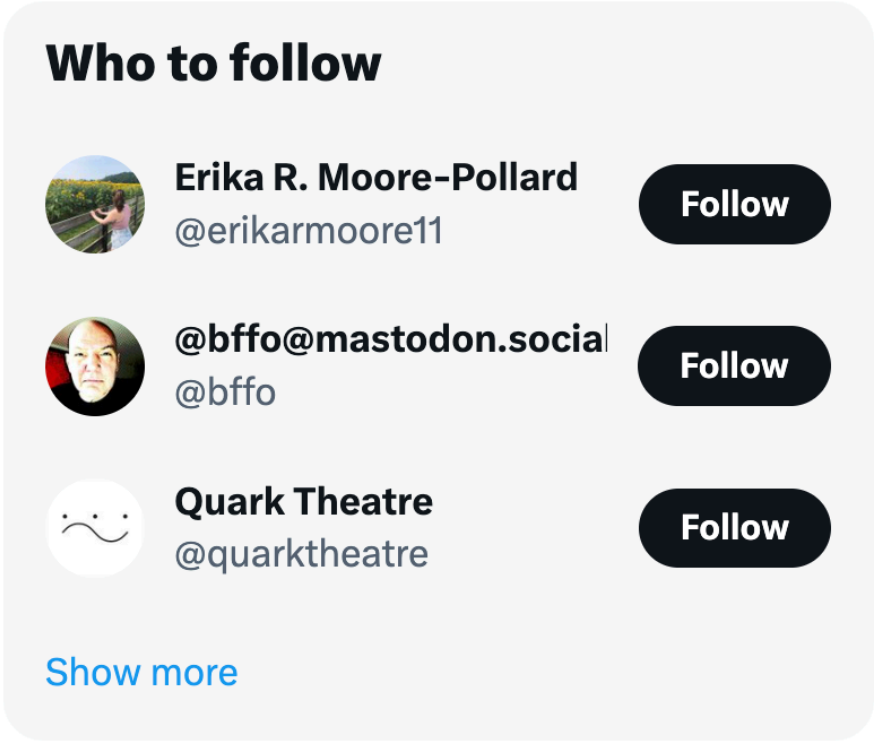
Figure 3: The end-to-end framework aggregates disparate network data to construct TwHIN, joint-embedding is performed and embeddings are consumed in downstream tasks and ML models.

‘Who to Follow’ use case

Experiment groups
- Unimodal (Control): basic feature + **graph embedding**
- Mixture (Test): basic feature + **TwHIN Embeddings**

Models
- Nearest Neighbor: retrieve Twitter accounts to follow

Evaluation Metrics:
- Recall: ability of a model to find all the relevant cases within a data set



Approach	R@10	R@20	R@50
Unimodal	0.58%	1.02%	2.06%
Mixture	3.70%	5.53%	8.79%

Ads Ranking

Experiment groups

- Baseline / Control : Basic feature
- TwHIN (Test):
  - Basic feature + **U (User)**
  - Basic feature + **U+A (User + Advertiser)**
  - Basic feature + **U+T (User + Entity)**
  - Basic feature + **U+A+T (User + Advertiser + Entity)**

Models

- Supervised Classification

Evaluation Metrics:

- Relative Cross Entropy (RCE), derived from probability, the higher the better

Model	Baseline	U	U+A	U+T	U+A+T
Ads_1	21.23	21.32	21.43	21.46	**21.48**
Ads_2	13.53	13.61	13.54	**13.63**	13.59
Ads_3	17.11	17.26	17.27	**17.27**	17.26

Table 2: Offline RCE for ads models with feature ablation. We investigate performance when using TwHIN embeddings for User (U), Advertiser (A), and Target entity (T) such as app to install, video to watch, or advertisement to click.

Relative Cross Entropy (RCE)

$$\text{RCE} = 100 \times \left(1 - \frac{\text{Cross Entropy of Model}}{\text{Cross Entropy of Baseline Model}}\right)$$

Example Dataset

Suppose we have a small dataset of predicted probabilities from our model and the actual labels for ad clicks:

User	Predicted Probability (p)	Actual Label (y)
1	0.9	1
2	0.2	0
3	0.6	1
4	0.1	0
5	0.8	1

Calculating Cross Entropy (CE) Loss

The CE loss for each observation is calculated as:

$$-\left(y \log(p) + (1 - y) \log(1 - p)\right)$$

Applying this to User 1:

$$-\left(1 \log(0.9) + (1 - 1) \log(1 - 0.9)\right) = -\log(0.9) \approx 0.105$$

Calculating Relative Cross Entropy (RCE)

With the average CE loss of our model and the baseline, we can calculate RCE:

$$\text{RCE} = 100 \times \left(1 - \frac{\text{Model CE Loss}}{\text{Baseline CE Loss}}\right)$$

Substituting the values:

$$\text{RCE} = 100 \times \left(1 - \frac{0.25}{0.693}\right) \approx 63.92$$

Search Ranking

Experiment groups

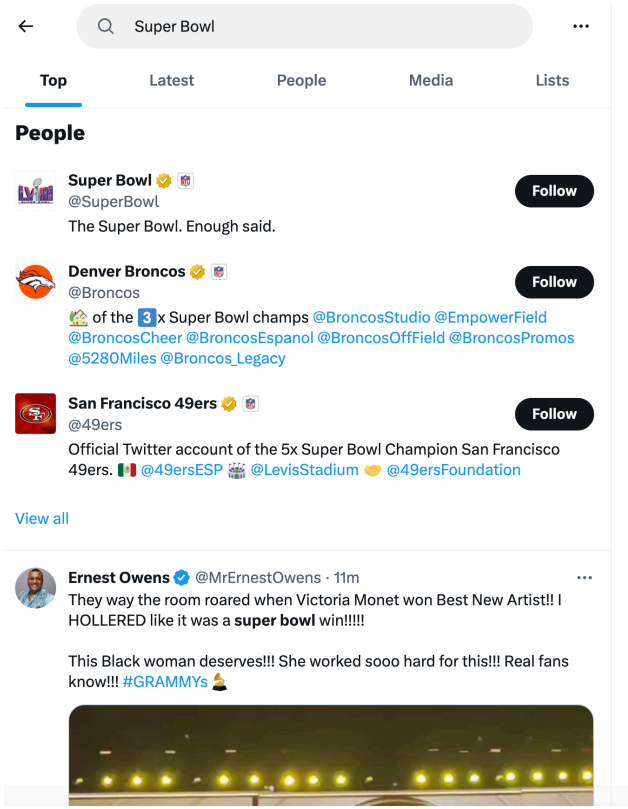
- Baseline / Control: Hand-Crafted Features + mBERT Outputs
- TwHIN (Test): Hand-Crafted Features + mBERT Outputs + **TwHIN Embeddings**

Models

- Supervised classification -- MLP (Multilayer Perceptron) for engagement prediction

Evaluation Metrics:

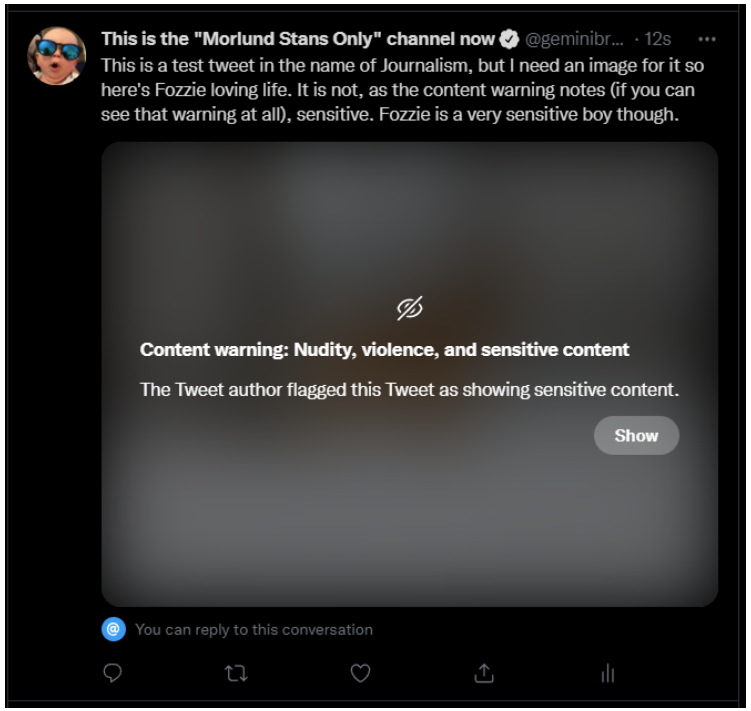
- MAP (Mean Average Precision)
- ROC (Receiver Operating Characteristic)



Metric	Baseline	$+U_f$	$+U_e$	$+A$	$+U_f + U_e$	$+U_f + U_e + A$
MAP	55.7	56.2	56.6	55.8	56.6	**57.0**
ROC	57.9	58.6	59.0	57.9	59.0	**59.6**

Table 3: Search engagement-based ranking with TwHIN embeddings: TwHIN user embeddings, both follow-base (U_f) and engagement-base (U_e), and TwHIN author embeddings (A)

Detecting Offensive Content



Experiment groups

- Baseline / Control : RoBERTa and BERTweet, most 'state of the art' approaches for contextual understanding
- Test: Content + **TwHIN-Author embeddings**

Models

- Supervised classification -- (e.g., neural network, logistic regression)

Evaluation Metrics:

- AUC (Area Under the ROC Curve)

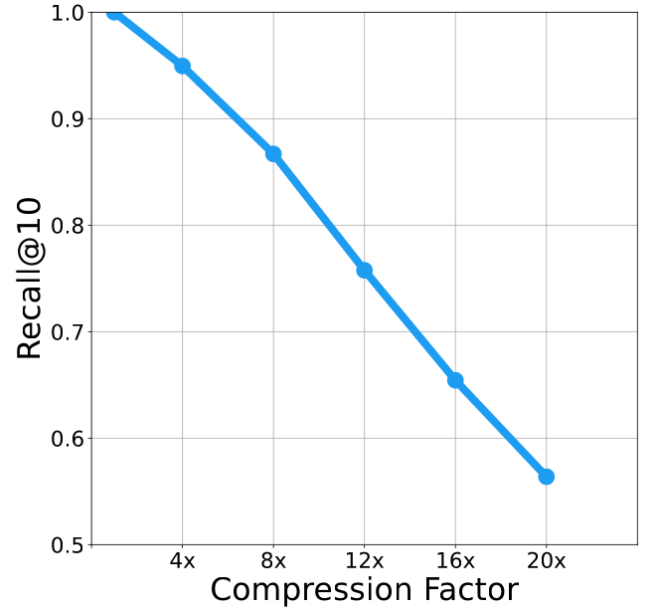
	RoBERTa	BERTweet	+TwHIN-Author
Collection_1	0.4123	0.4692	**0.5161**
Collection_2	0.688	**0.7274**	0.7174

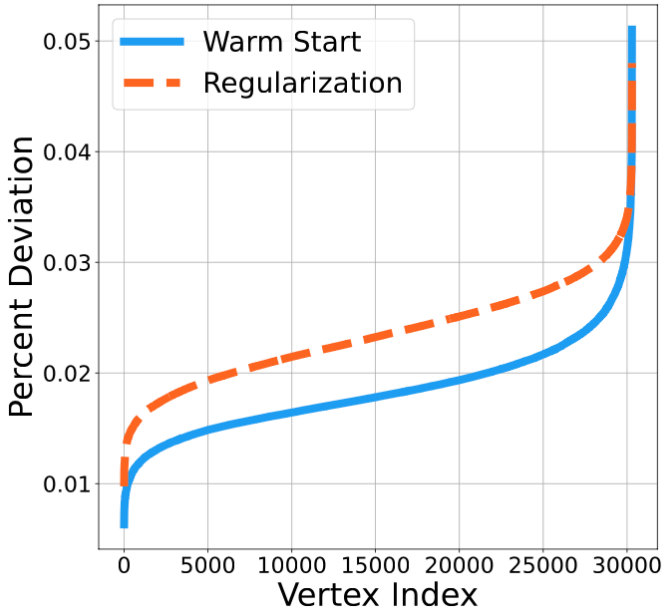
Table 4: PR-AUC results for detecting offensive content. We compare the performance of content-based models to leveraging a TwHIN author embedding in addition to content.

Practical Consideration

- Compression for low latency
  - Quantization of vectors for downstream use cases

- Addressing parameter drift across versions
  - Continue training as warm start
  - Apply regularization

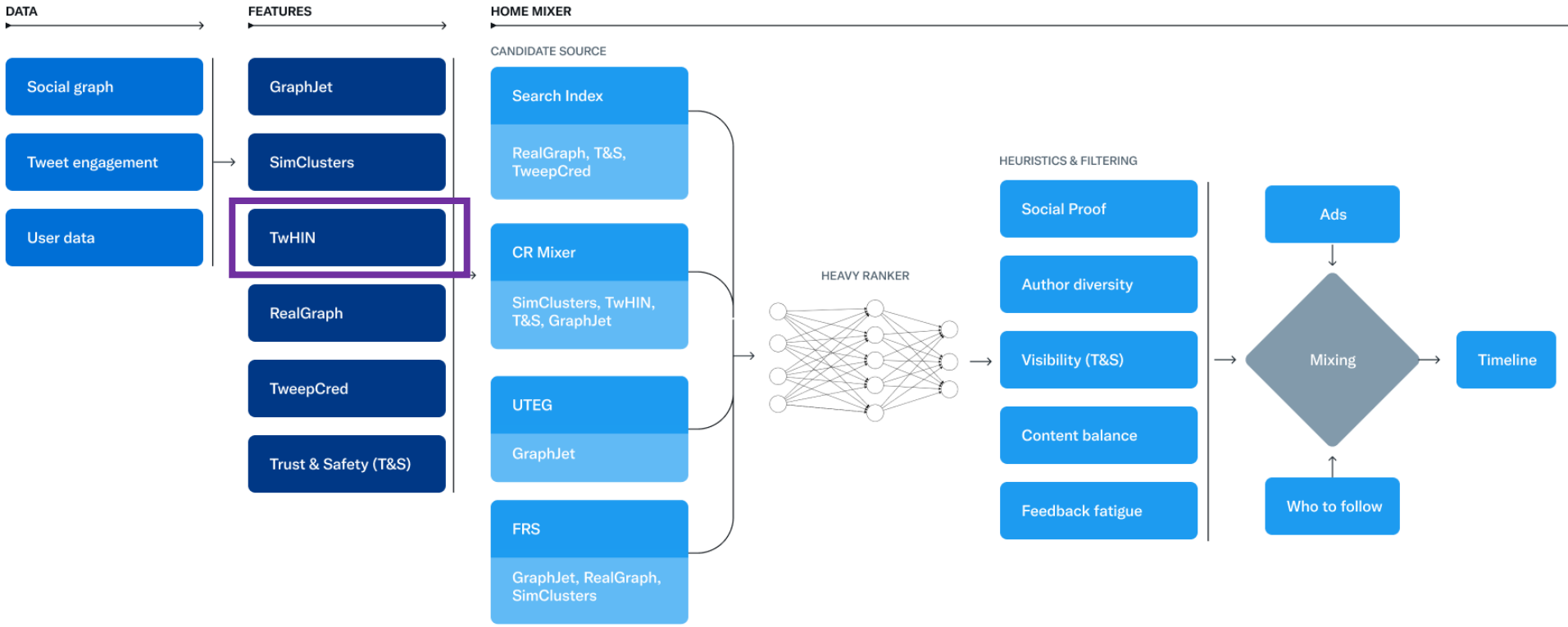




Summaries

- **TwHIN**: embeddings of a multi-type, multi-relation network, capturing the complexity of relationships, addressing data sparsity and improving the model's ability to generalize.
- **Experimental Validation**: improvements with TwHIN embeddings for recommendation use cases
- **Implementation Insights**: Practical tips and methodologies for effectively implementing, deploying, and leveraging large-scale heterogeneous graph embeddings

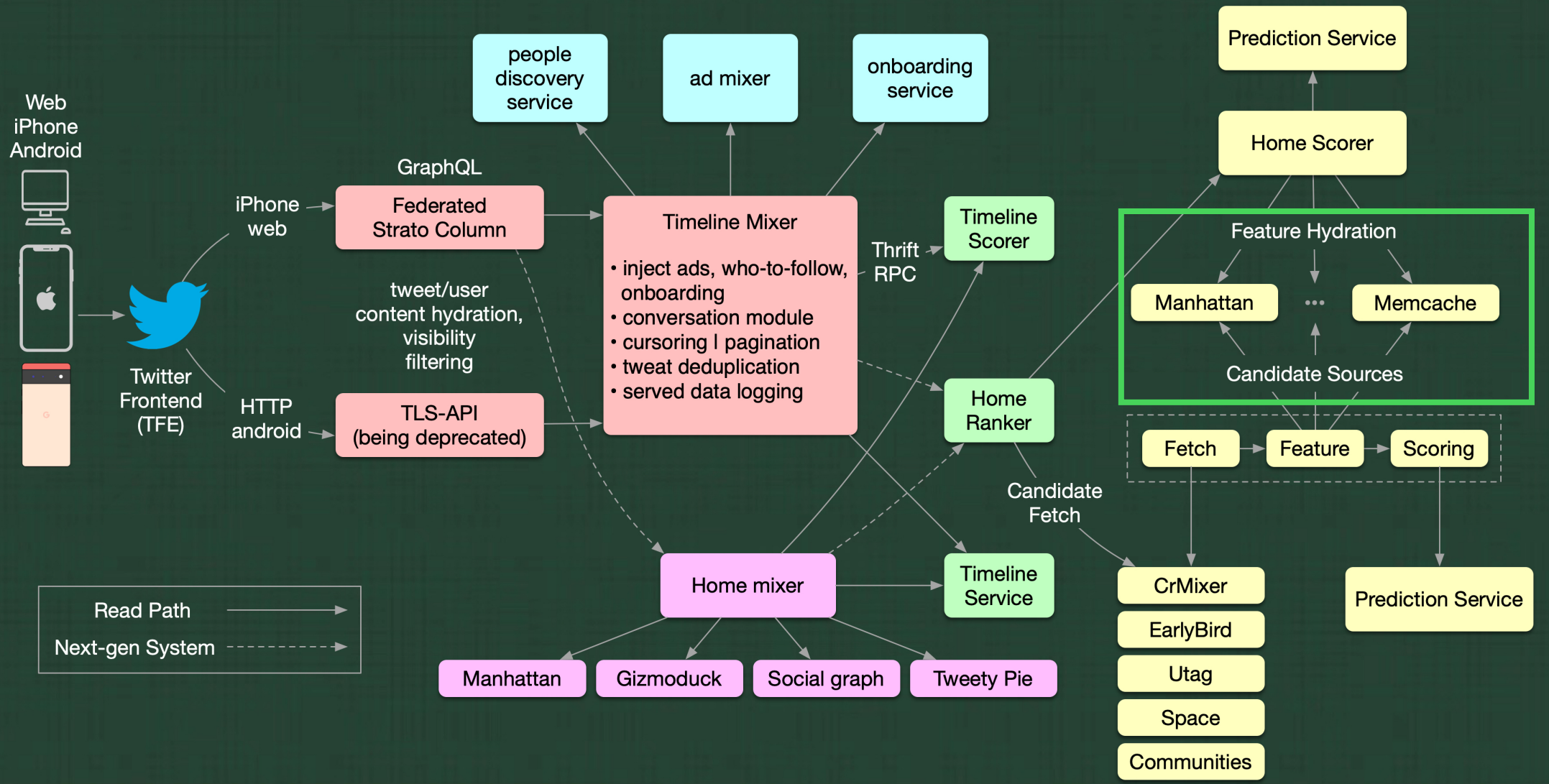
Twitter's Algo Open Source at github



- TwHIN is an important feature
- Home Mixer / CR Mixer: retrieve in-network (750) and out-of-network tweets (750)
- Heavy Ranker: MaskNet multiple objective learning model to rank the tweets

Twitter Architecture 2022

Posted by Elon Musk
Redraw by bytebytego.com



Thank you for your attention!

References

- Arxiv Pdf
- Twitter's Recommendation Algo Open Source
- Twitter Architecture 2022
- Open source Blog
- Open source machine learning algo
- CS224W: Machine Learning with Graphs