Graph-based Recommendation Systems

Barbara Rychalska

SYNERISE

Our base paper:

An efficient manifold density estimator for all recommendation systems

J.Dąbrowski, B. Rychalska, M.Daniluk, D.Basaj, K. Gołuchowski, P.Bąbel, A.Michałowski

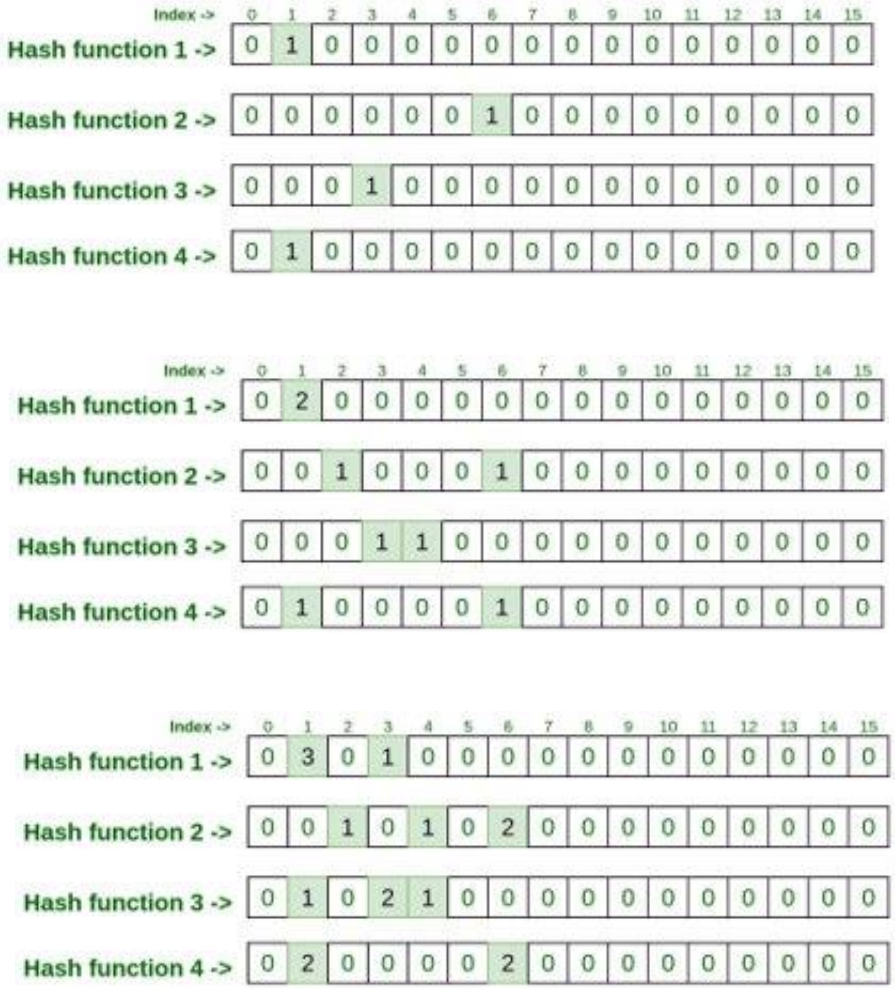
https://arxiv.org/pdf/2006.01894.pdf

- A one pass algorithm for efficient, approximate representation of smooth probability densities on Riemannian manifolds.
- EMDE includes intuitions from LSH (Locality Sensitive Hashing) and Count-Min Sketch algorithm.
- We establish new state-of-the-art results on several recommendation datasets for session-based and top-k scenarios.

Count-Min Sketch

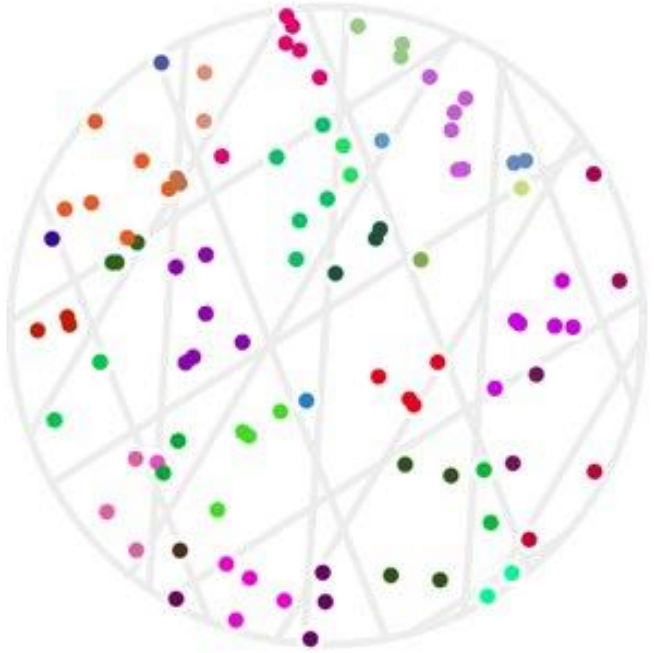
Probabilistic data structure that serves as a frequency table of events in data.

- we only have limited space availability; it would help if we could get away with not storing elements themselves but just their counts
- With the help of a few hash functions *h*, we can implement a counting mechanism. To increment the count for element *a*, we hash it to get an index into the array. The cell at the respective index *h*(*a*) is then incremented by 1.



Density-dependent Local Sensitive Hashing

Johnson-Lindenstrauss lemma: [for high dimensional data] if you pick a random subspace and project onto it, the scaled pairwise distances between points will *probably* be preserved.



Locality Sensitive Hashing (LSH)

- Hashes similar input items into the same "buckets" with high probability
- Utilizes vector representation by multiple partitioning of the embedding manifold

Density-dependent LSH (our contribution)

- Modified version of LSH
- Cuts the the manifold into non-empty parts, avoiding unutilized regions

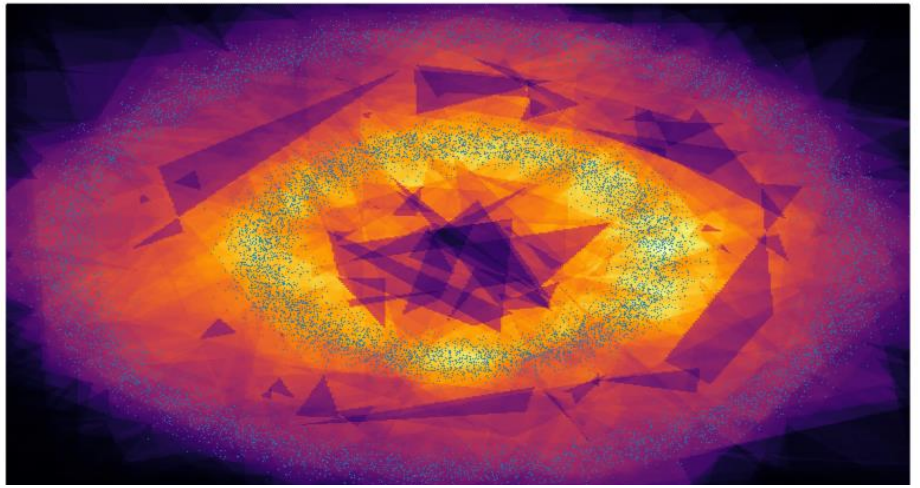
Density-dependent Local Sensitive Hashing

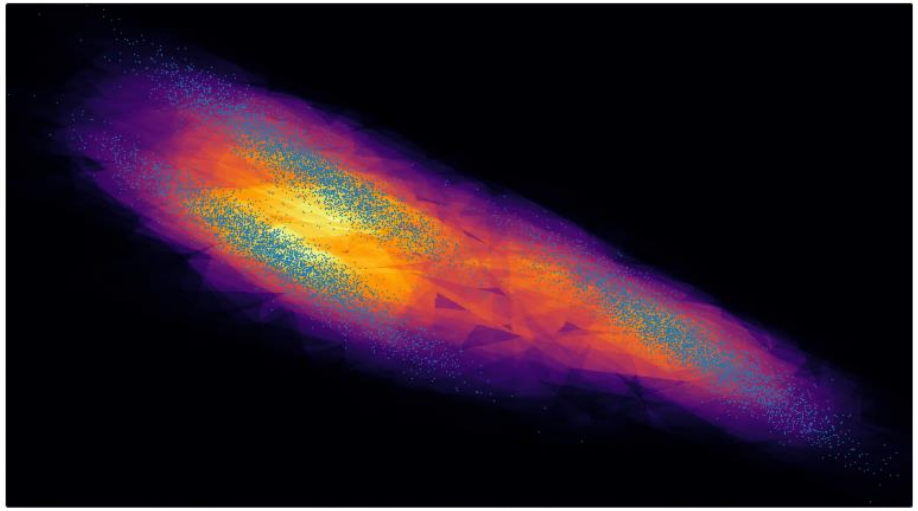
1. Draw K random vectors *r*
2. Draw biases *b* from $Q_i(U \sim \mathrm{Unif}[0,1])$ where *Q* is a data-dependent quantile function of $v \cdot r_i$

3. Compute hash codes
   $hash_i(v) = sgn(v \cdot r_i - b_i)$

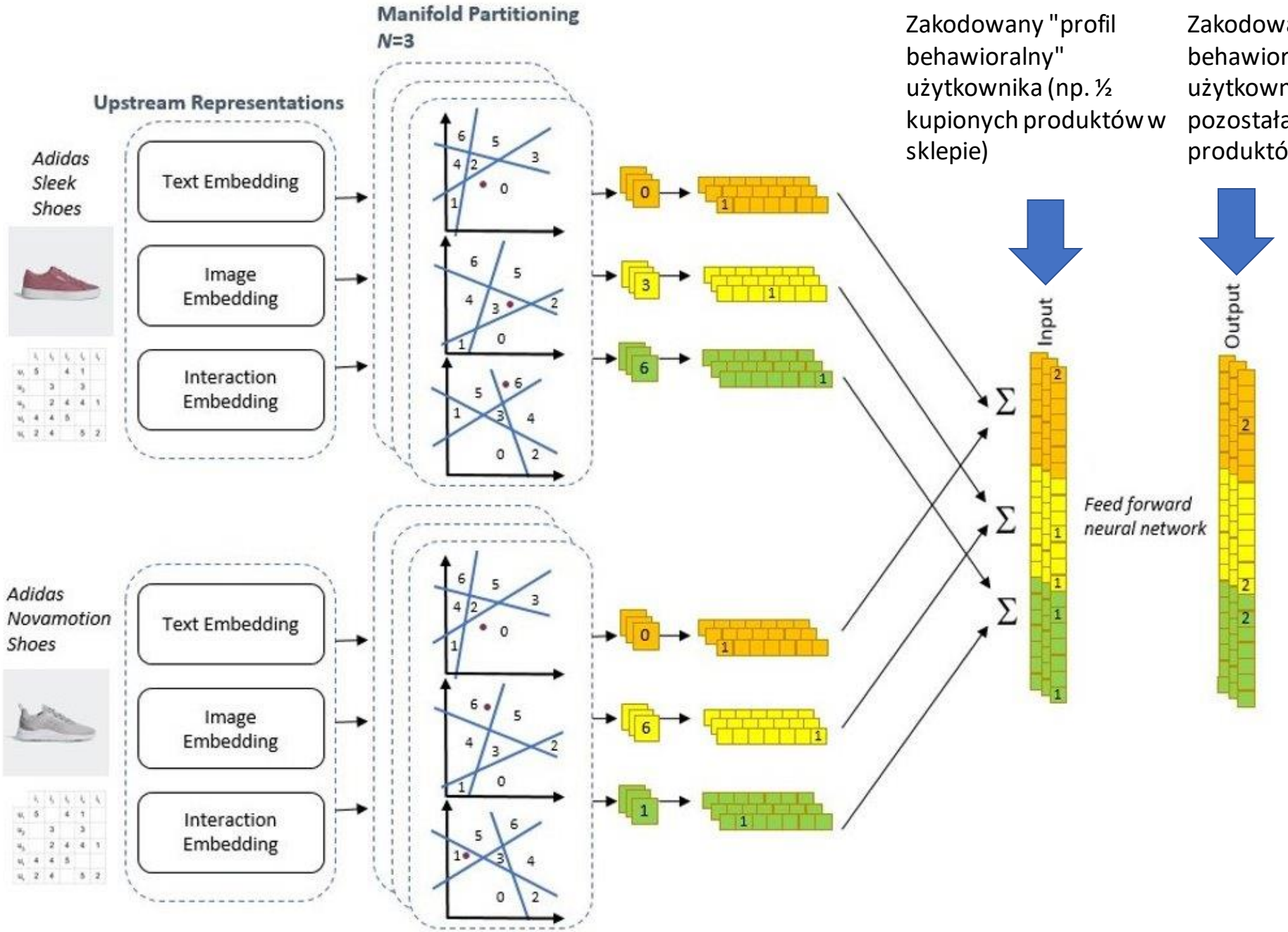
4. The procedure is parametrized by 2 parameters: *K* and *N*
   - *N* determines the number of independent manifold divisions
   - Each manifold is split into 2^*K* regions

   The parameter *b* which specifies the cutoff point of a linear function on the *Y* axis, is picked based on data density, ensuring that space divisions are created in the regions of high density.





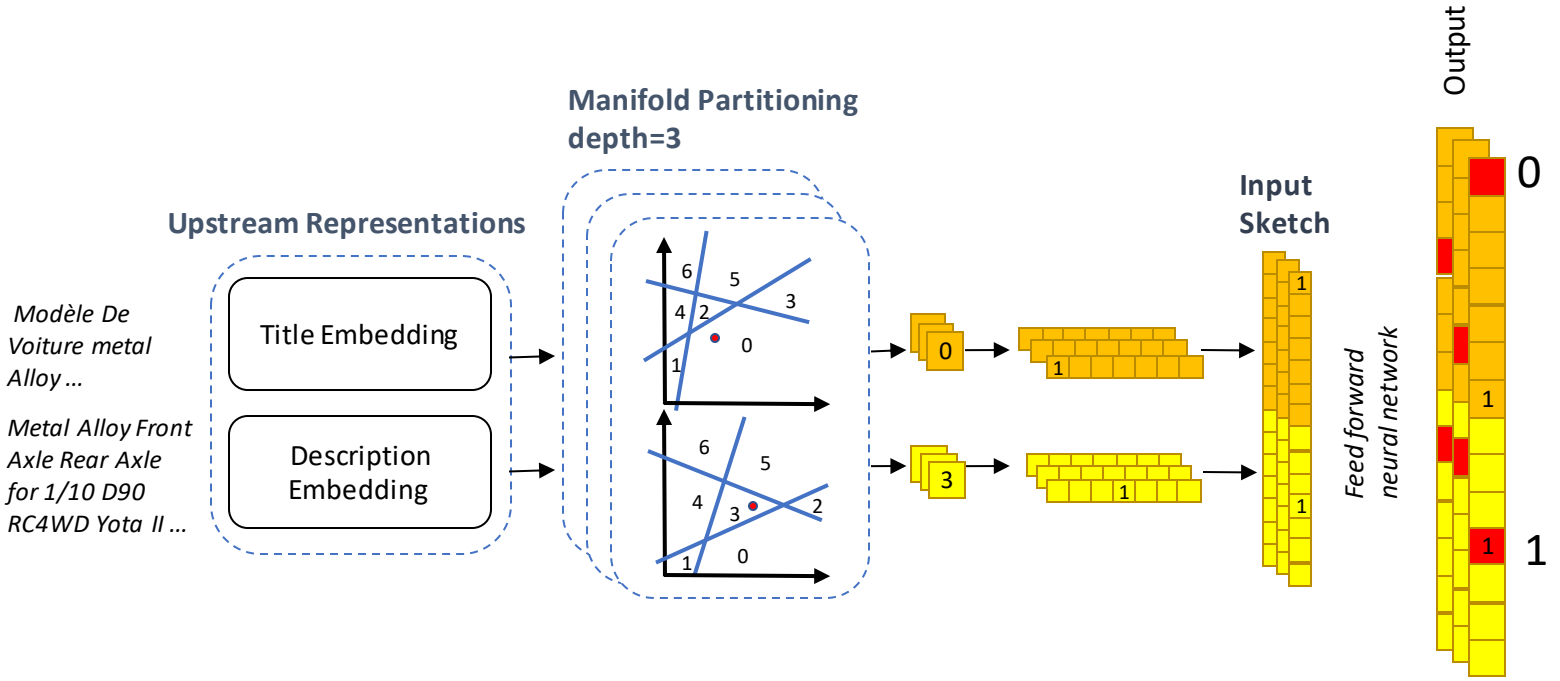
SYNERISE


Manifold Partitioning
N=3
Upstream Representations
Adidas Sleek Shoes
Text Embedding
Image Embedding
Interaction Embedding
Adidas Novamotion Shoes
Text Embedding
Image Embedding
Interaction Embedding
Zakodowany "profil behawioralny" użytkownika (np. ½ kupionych produktów w sklepie)
Zakodowany "profil behawioralny" użytkownika (np. pozostała ½ kupionych produktów w sklepie)
Input
Output
Feed forward neural network

SYNERISE

Prediction

1. Encode all image representations into sketches (this is done just once).
2. Following Count-Min Sketch retrieval procedure, identify relevant buckets for each item in the output.
3. Select the values from the output which correspond to the relevant buckets.
4. Count the score as a *geometric mean* of the values from relevant buckets instead of *min* in Count-Min Sketch.
5. Pick the item with the highest score as the best recommendation.



score = geom_mean([0,1,...])

EMDE as a Session-Based Recommender System

On top of EMDE, we use a simple 3-layer feed forward network with leaky ReLU activations, batch normalization and residual connections between layers. Whenever possible, we incorporate multimodal item features such as additional metadata or data from different sources such as search queries, playlists, purchases etc.

EMDE - no multimodal data, only basic user-item interactions

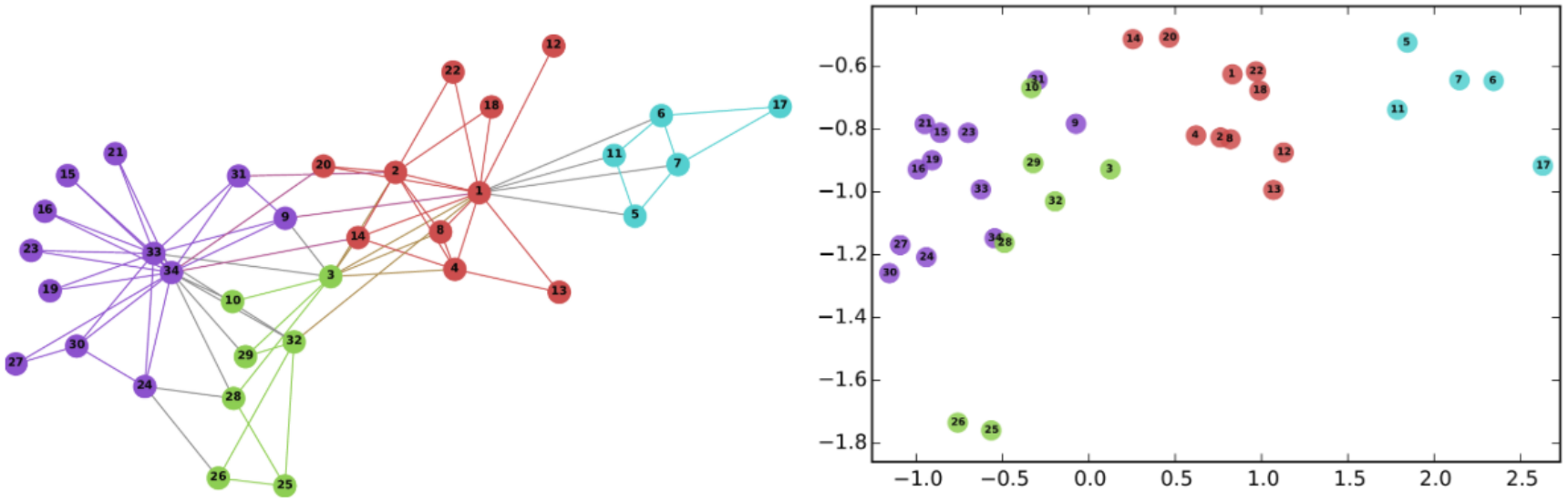
EMDE MM - configurations where selected multimodal channels are present

RETAIL	MRR	P@20	R@20	HR@20	MAP@20
EMDE MM	0.3649	0.0556	0.4974	0.6202	0.0302
EMDE	0.3522	0.0512	0.4643	0.5774	0.0277
VS-KNN	0.3395	0.0531	0.4632	0.5745	0.0278
S-KNN	0.337	0.0532	0.4707	0.5788	0.0283
GRU4REC	0.3237	0.0502	0.4559	0.5669	0.0272

DIGI	MRR	P@20	R@20	HR@20	MAP@20
EMDE MM	0.1731	0.0620	0.3849	0.4908	0.0268
EMDE	0.1714	0.0595	0.3697	0.4691	0.0254
VS-KNN	0.1784	0.0584	0.3668	0.4729	0.0249
S-KNN	0.1714	0.0596	0.3715	0.4748	0.0255
GRU4REC	0.1644	0.0577	0.3617	0.4639	0.0247

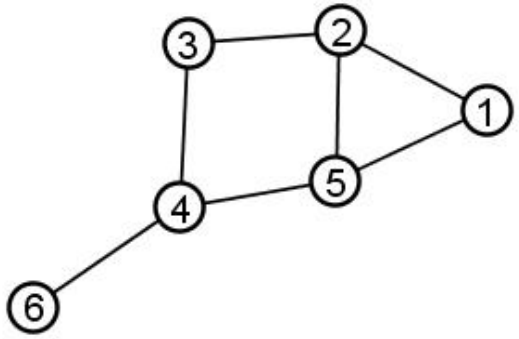
Cleora

A system computing graph node embeddings.



Cleora

Extremely simple procedure.



1) Compute transition matrix M

	1	2	3	4	5	6
1		1/3			1/3	
2	1/2		1/2		1/3	
3		1/3		1/3		
4			1/2		1/3	1
5	1/2	1/3		1/3		
6				1/3		

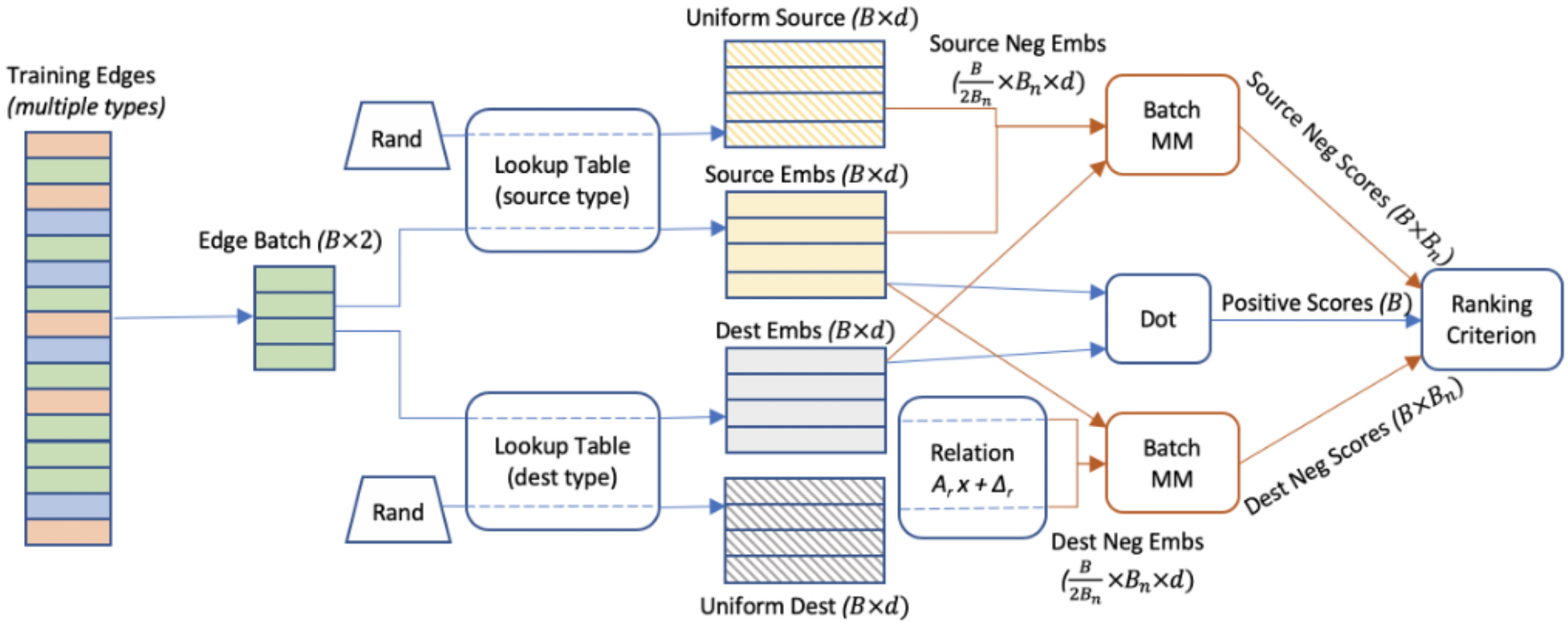
2) Init embedding matrix T0 uniformly

3) For i in iter_num:

1. Ti = M * T(i-1)
2. Normalize embedding vectors in Ti

Each multiplication extends visible neighborhood by 1. Iteration number of e.g. 4 means that nodes with similar 4-hop neighborhoods will be similar.

PyTorch BigGraph



Some Results

Mean Reciprocal Rank and Hit Rate@10 for link prediction:

	Cleora	PyTorch BigGraph
MRR	0.586	0.565
HR@10	0.627	0.672

LiveJournal Dataset

	Cleora	PyTorch BigGraph
MRR	0.929	0.824
HR@10	0.942	0.837

RoadNet Dataset

Training speed:

	RoadNet	LiveJournal
Cleora	00:21:59 h	00:53:44 h
PBG	00:31:11 h	07:10:00 h

Name	Facebook	YouTube	RoadNet	LiveJournal	Twitter
Nodes	22,470	1,134,890	1,965,206	4,847,571	41,652,230
Edges	171,002	2,987,624	2,766,607	68,993,773	1,468,365,182
Average Degree	12	5	3	16	
Density	6×10^{-4}	4.7×10^{-6}	1.4×10^{-6}	3.4×10^{-6}	
Classes	4	47	-	-	-
Directed	No	No	No	Yes	Yes

Table 1: Dataset statistics.

Trends in the field

Graph embeddings:

- Computed on a GPU for parallelization
- Efforts at speedup: coarsening
- 2 tracks:
  - Embeddings can be either more elaborate
  - … or faster.

Recommenders:

- Utilizing additional knowledge:
  - Multimodal (item description, item image, item movie)
  - hetero-interactional (e.g. predicting purchase from clicks or page views)
  - Cross-domain (recommend clothes based on the electronics a user bought)
- A lot of directions: graph neural networks, VAEs, KNN-based approaches... usually one line of research compares just to itself
- A lot of crossovers with economy/business approaches
- Underdeveloped field: adversarial attacks
- Major venue: RecSys
- Important work: https://arxiv.org/abs/1907.06902

[it has] become difficult to keep track of what represents the state-of-the-art at the moment, e.g., for top-n recommendation tasks. At the same time, several recent publications point out problems in today's research practice in applied machine learning, e.g., in terms of the reproducibility of the results or the choice of the baselines when proposing new models. In this work, we report the results of a systematic analysis of algorithmic proposals for top-n recommendation tasks. Specifically, we considered 18 algorithms that were presented at top-level research conferences in the last years. Only 7 of them could be reproduced with reasonable effort. For these methods, it however turned out that 6 of them can often be outperformed with comparably simple heuristic methods, e.g., based on nearest-neighbor or graph-based techniques. The remaining one clearly outperformed the baselines but did not consistently outperform a well-tuned non-neural linear ranking method.