Travel Recommendation System

Booking.com & NVIDIA

Sources:

NVIDIA's winning solution: ← diagrams

https://developer.nvidia.com/blog/how-to-build-a-winning-deep-learning-powered-recommender-system-part-3/

Competition webpage (WSDM 21)

https://www.bookingchallenge.com

APPLIED ROOTS

Problem definition:

4 suggestions

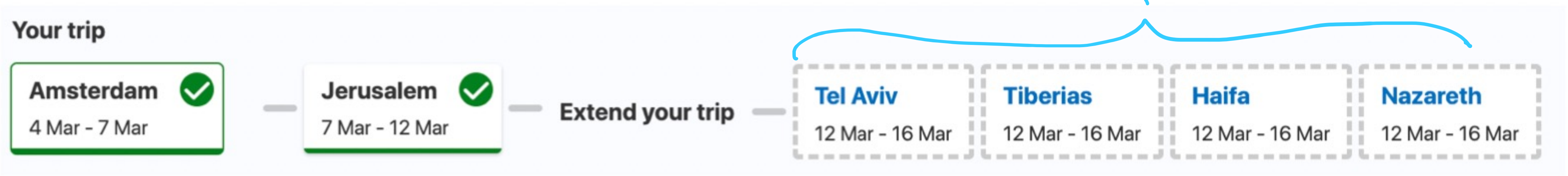


Figure 1: Multi-Destinations trip recommendation bar on Booking.com website

Multi Destination Trip.

Pr{Milan→ Venice→ Florence→ Rome→ Naples→ Reggio Calabria→ Palermo}

> Pr{Milan→ Florence→ Palermo→ Rome→ Venice→ Naples→Reggio Calabria}

Similar:

Pr{**next_dest**=Palermo | **so_far** = (Venice, Florence, Rome)}

> Pr{**next_dest**=Milan | **so_far** = (Venice, Florence, Rome)}

APPLIED ROOTS

Problem:

Predict the final city (city_id) of each trip (utrip_id)

Metric:

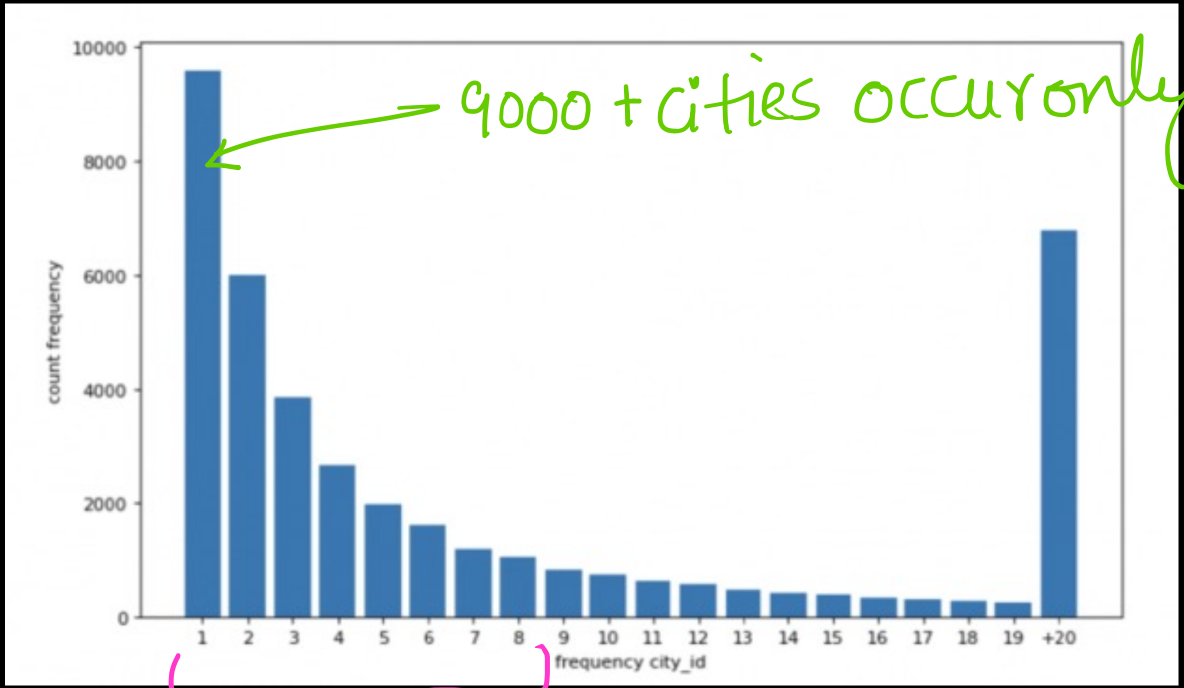
Precision@4 (4 suggestion slots)

Dataset:

- 269K trips; ~1.5MM reservations;
- each trip has atleast 4 cities/reservations

- User_id – User ID
- Check-in – Reservation check-in date
- Checkout – Reservation check-out date
- Affiliate_id – An anonymized ID of affiliate channels where the booker came from (e.g. direct, some third-party referrals, paid search engine, etc.)
- Device_class – desktop/mobile
- Booker_country – Country from which the reservation was made (anonymized)
- Hotel_country – Country of the hotel (anonymized)
- City_id – city_id of the hotel's city (anonymized)
- Utrip_id – Unique identification of user's trip (a group of multi-destination bookings within the same trip)

EDA + featurization:



9000 + cities occur only once in the dataset

↳ all cities which occur < 9 are grouped into one ID

~ 10,277 cities

Additional engineered features:

- **Trip context date, time features**: day-of-week, week-of-year, month, weekend, season, stay length (checkout – check-in), days since the last booking (check-in – previous checkout).
- **Trip context sequence features**: the first city in the trip, lagged (previous 5) cities and countries from the trip.
- **Trip context statistics:** trip length (number of reservations), trip duration (days), reservation order (in ascending and descending orders).
- **Past user trip statistics:** number of user's past reservations, number of user's past trips, number of user's past visited cities, number of user's past visited countries.
- **Geographic seasonal city popularity:** features based on the conditional probabilities of a city c from a country co, being visited at a month m or at a week-of-year w, as follows: $P(c \mid m)$, $P(c \mid m, co)$, $P(c \mid w)$, $P(c \mid w, co)$.

APPLIED ROOTS

Data Augmentation : 1.5MM is "small" data.

epxerimentation

Boston->New York->Princeton->Philadelphia->Washington DC

reverse

(2x data)

Boston<-New York<-Princeton<-Philadelphia<-Washington DC

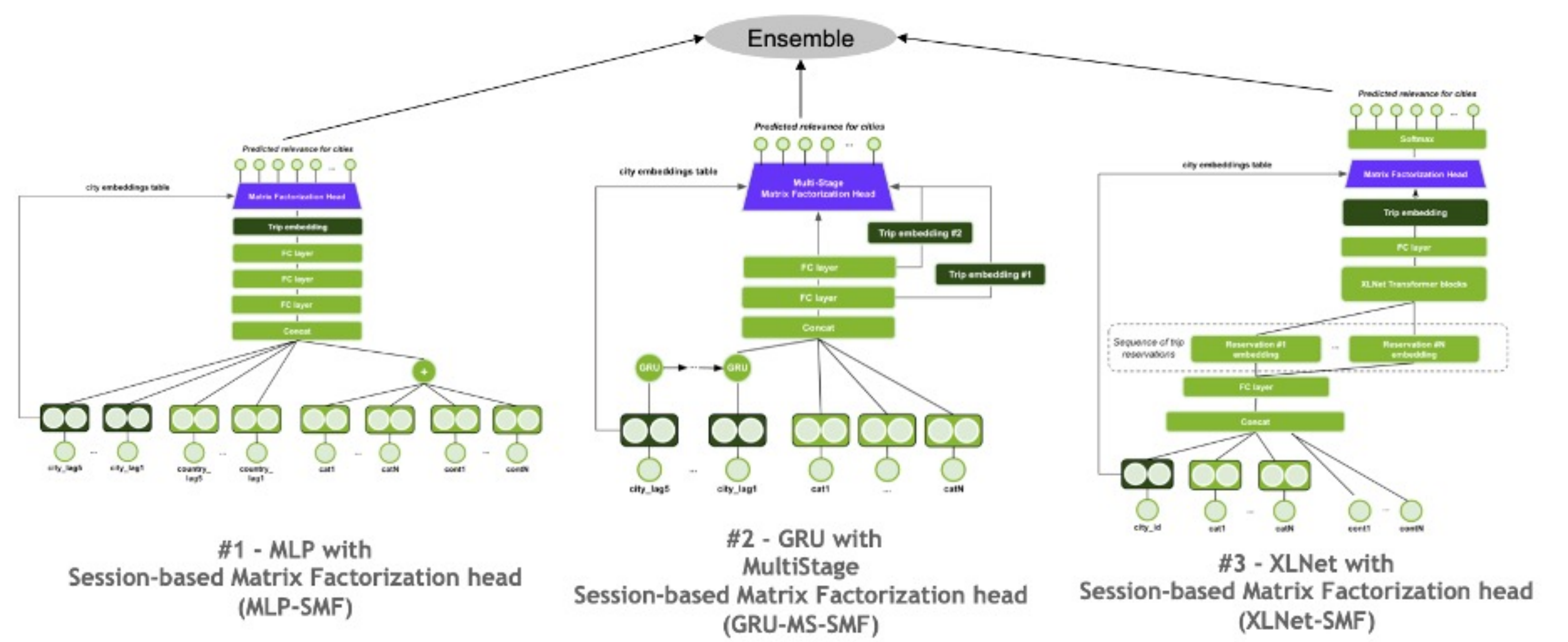
Boston->Princeton->Washington DC->New York->Philadelphia

random order

X

Ensembles:

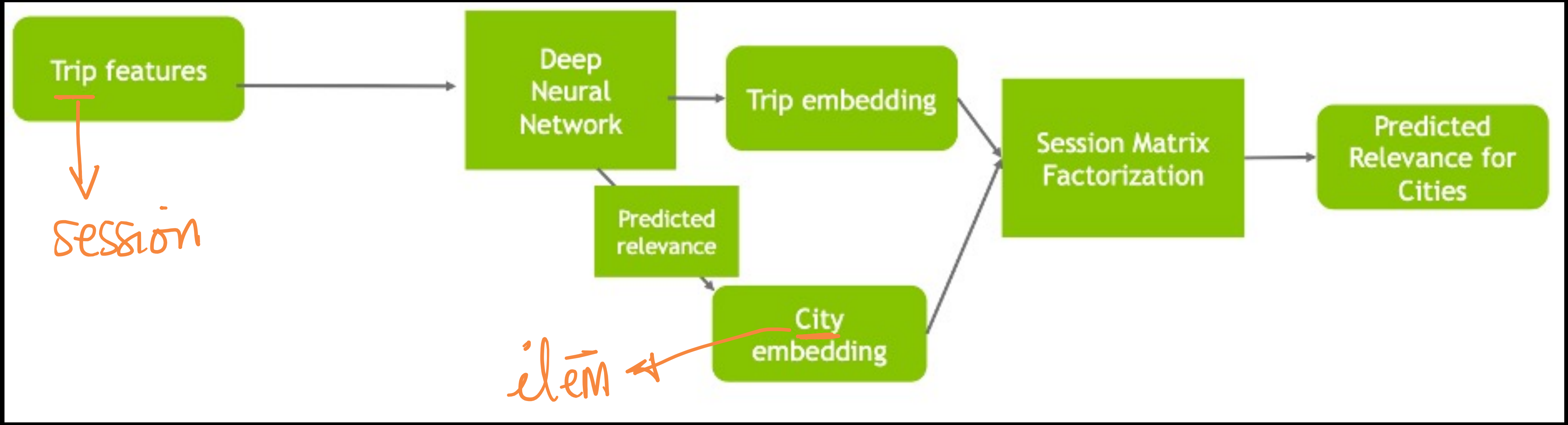
- Popular in competitions
- expensive for deployment
- weighted average of predictions from 3 DL-models
- Recommendation as a multi-class classification (#cities is not too large)



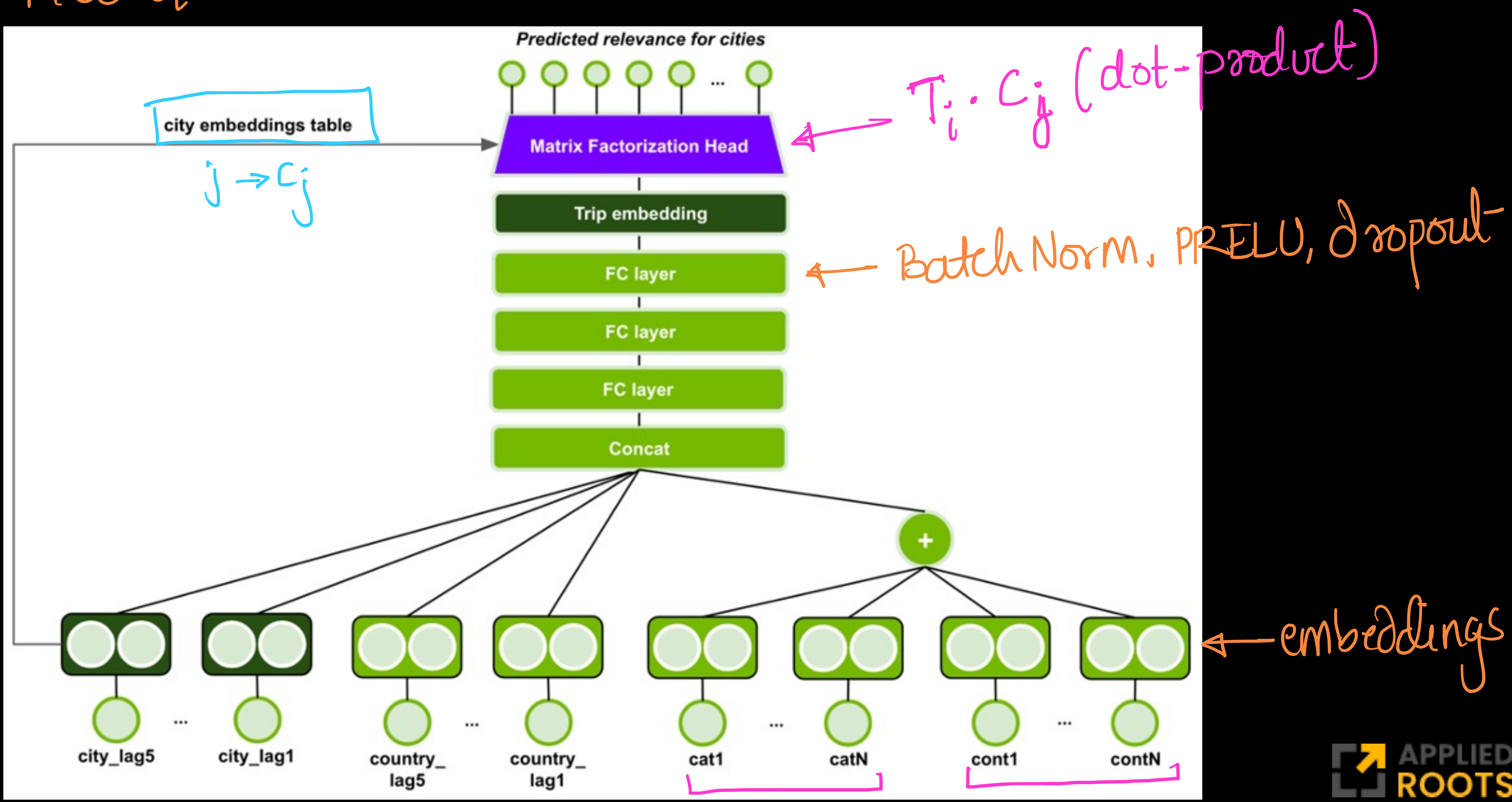
APPLIED ROOTS

Common component : SMF

↳ Session-based MF



1st Model: MLP-SMF



APPLIED ROOTS

(*) sequence info of cities preserved via position

- Classical MF : $\underline{P_u} \cdot \underline{Q_i} \approx r_{ui}$

- computed by decomposing R matrix

- T_i & C_j vectors from DL-model

→ loss: categorical cross-entropy

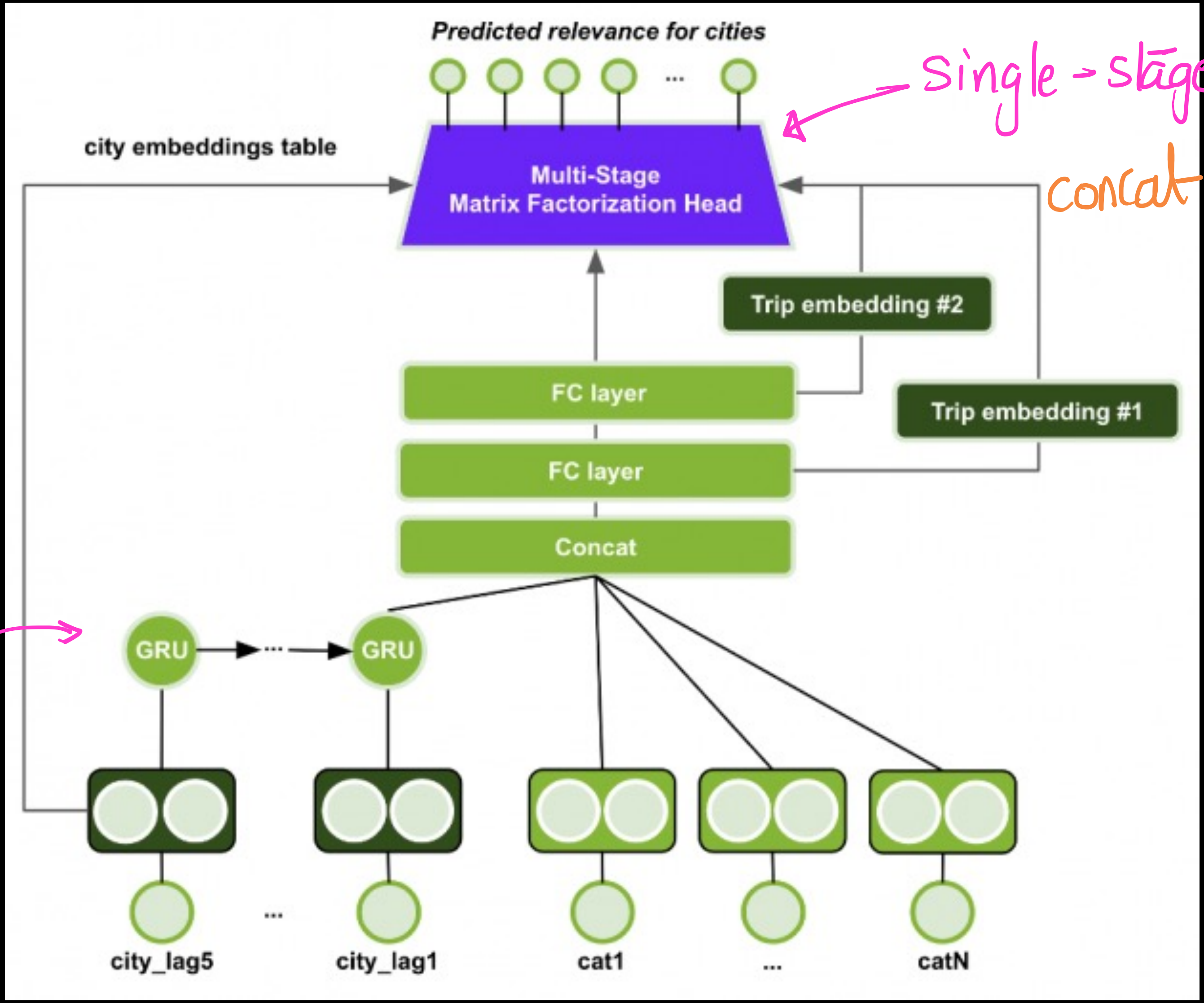
- cat1 - catN : categorical features

- cont1 - contN : continous-valued features

Model 2: GRU-SMF

single-stage is actually sufficient here.

concat

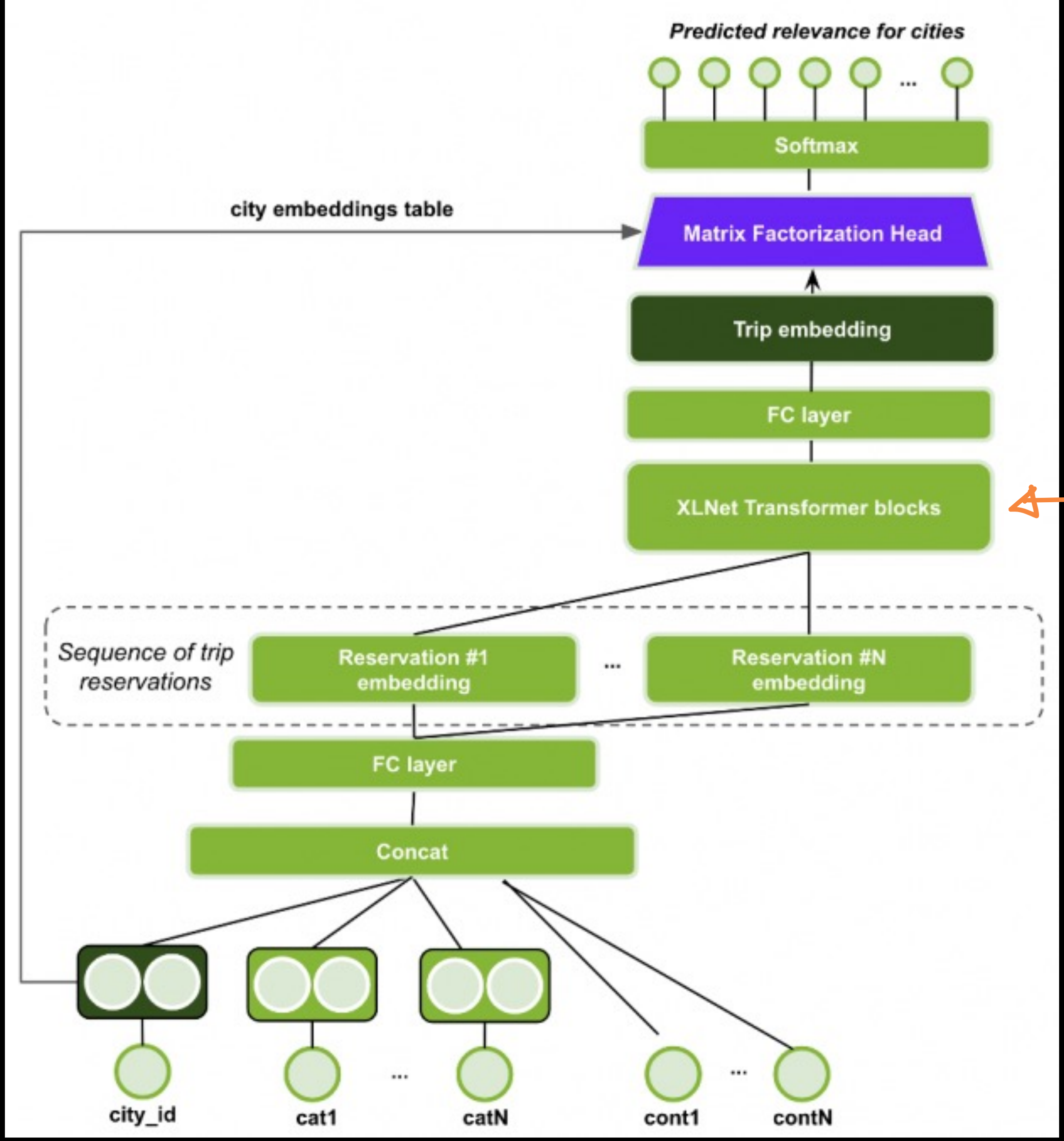


Sequence

(padding if needed)

APPLIED ROOTS

Model3: XLNet - SMF

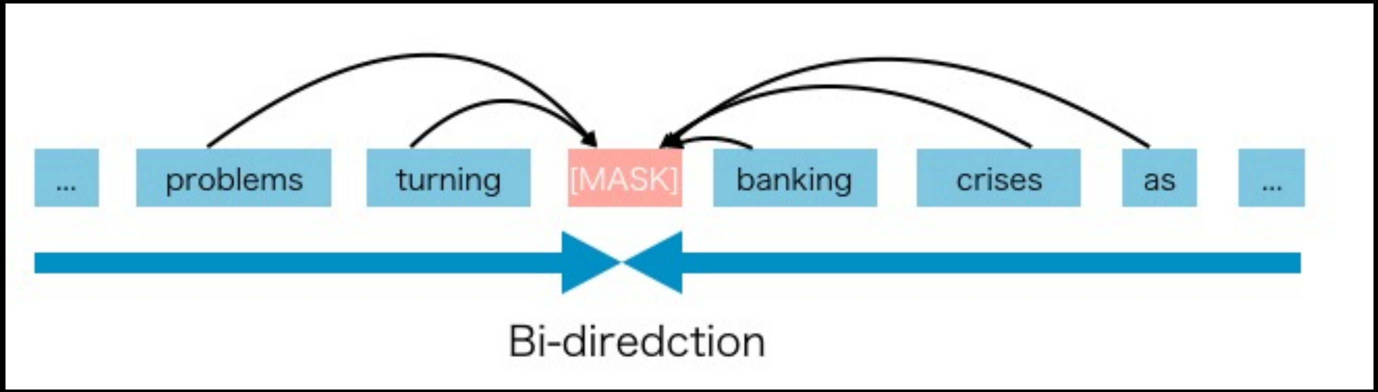


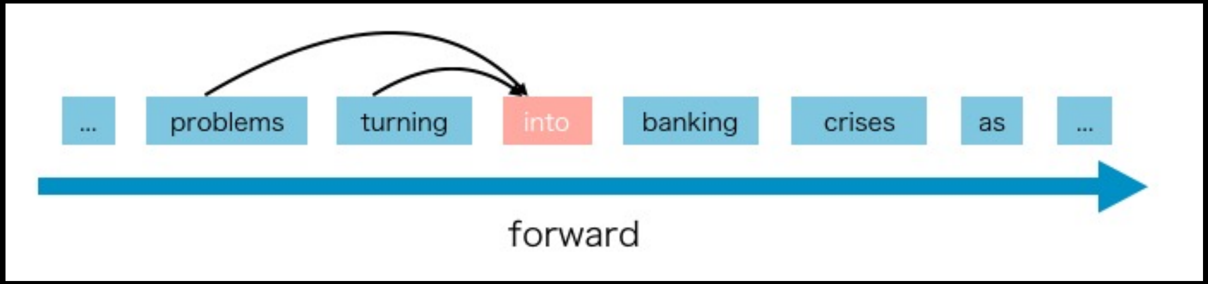
XLNet

sequence of cities

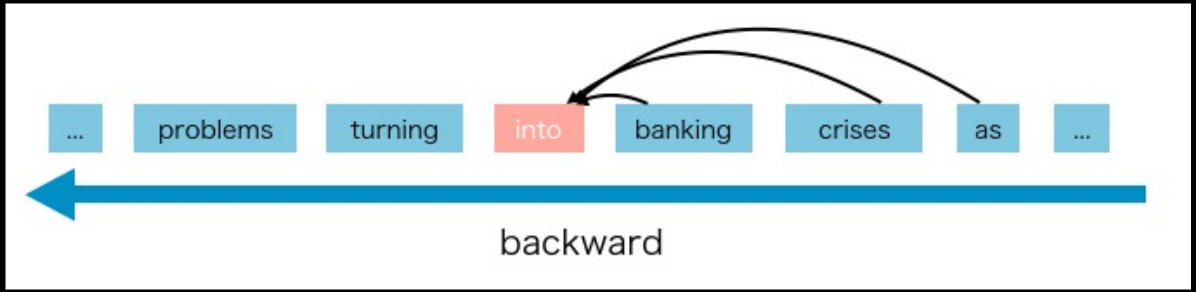
APPLIED ROOTS

XLNet vs BERT:



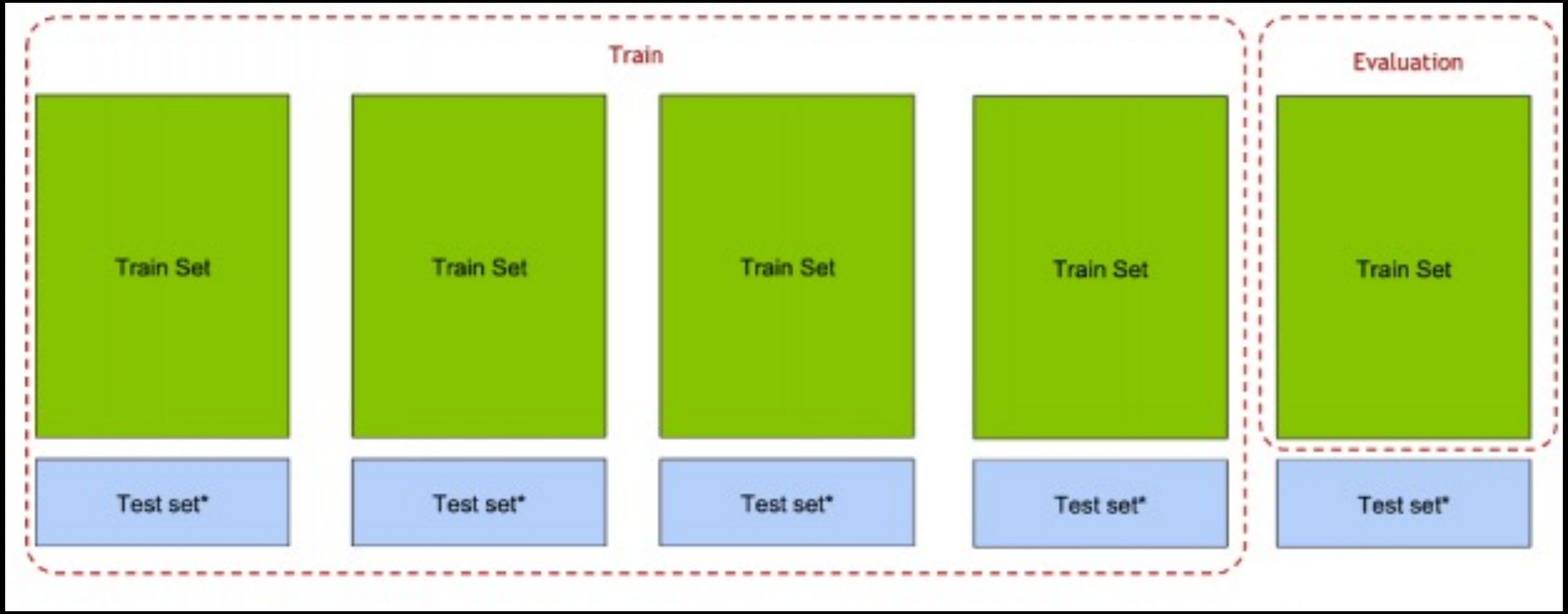


→ we use this to avoid leakage



APPLIED ROOTS

5-fold CV:



Train
Evaluation
Train Set
Train Set
Train Set
Train Set
Train Set
Test set*
Test set*
Test set*
Test set*
Test set*

Hyper-param tuning:

http://ceur-ws.org/Vol-2855/challenge_short_2.pdf

Ensembling:

```
split data in k-folds;
for each architecture do
    j = number of bags for this architecture;
    for each fold in folds do
        for i in [1, ..., j] do
            concatenate train and test out-of-fold data
              (OOF);
            train model on OOF;
            evaluate model based on last city of in-fold train
              data;
            predict last city of all folds of test data;
        end
    end
    ensemble by averaging over each fold and each bag;
end
ensemble by averaging over architectures;
```

bagging
+
3-architectures

simple average.

* ~80% matching across 3-architectures scores

⇓

improvement via ensemble

Scores:

	Single bag CV	Ensemble CV	Final LB
MLP-SMF	0.5667	0.5756	
GRU-MS-SMF	0.5664	0.5762	
XLNet-SMF	**0.5681**	0.5751	
Final Ensemble		**0.5825**	**0.5939**

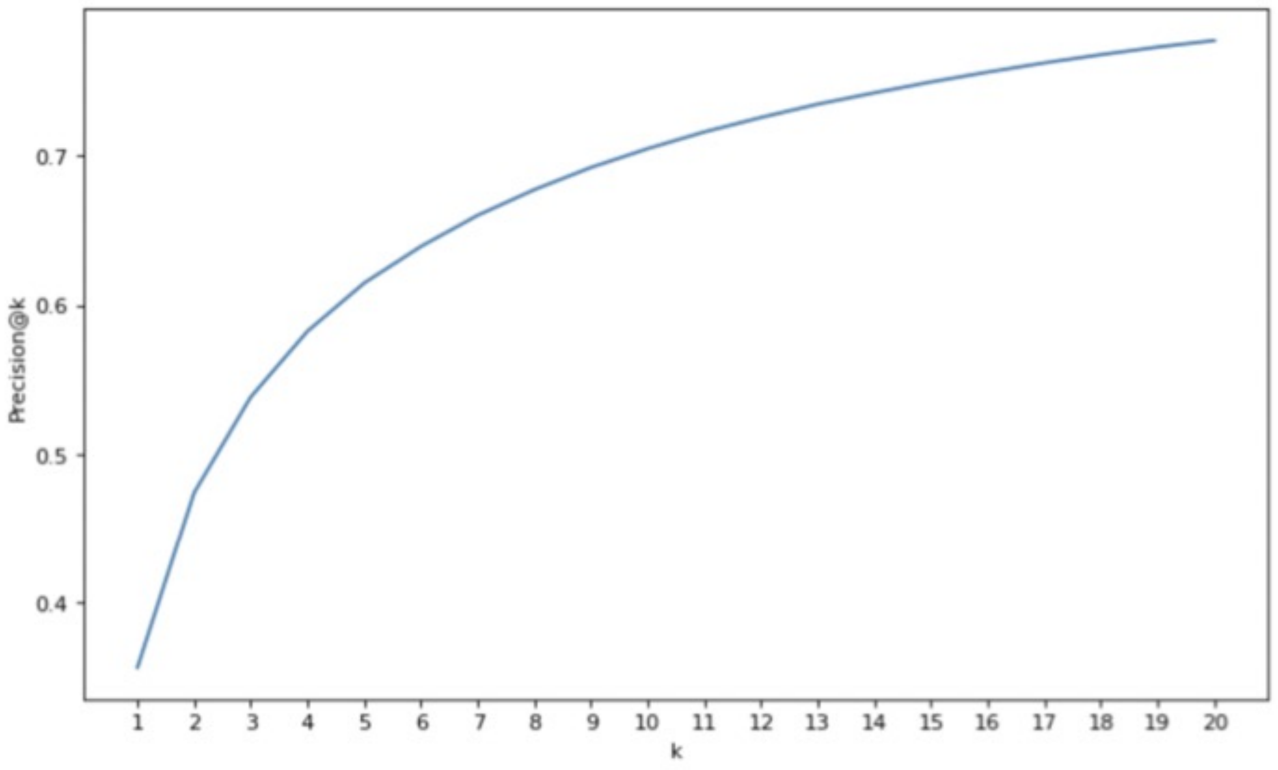


Figure 5: Precision@k for top20 recommendations by ensemble model.

F ABLATION STUDY

	1 model CV
MLP	0.5550
MLP-SMF	0.5667 (+2.1%)

Table 3: Cross validation score for training MLP architecture with and without the Session-based Matrix Factorization head (MLP-SMF).

Q & A