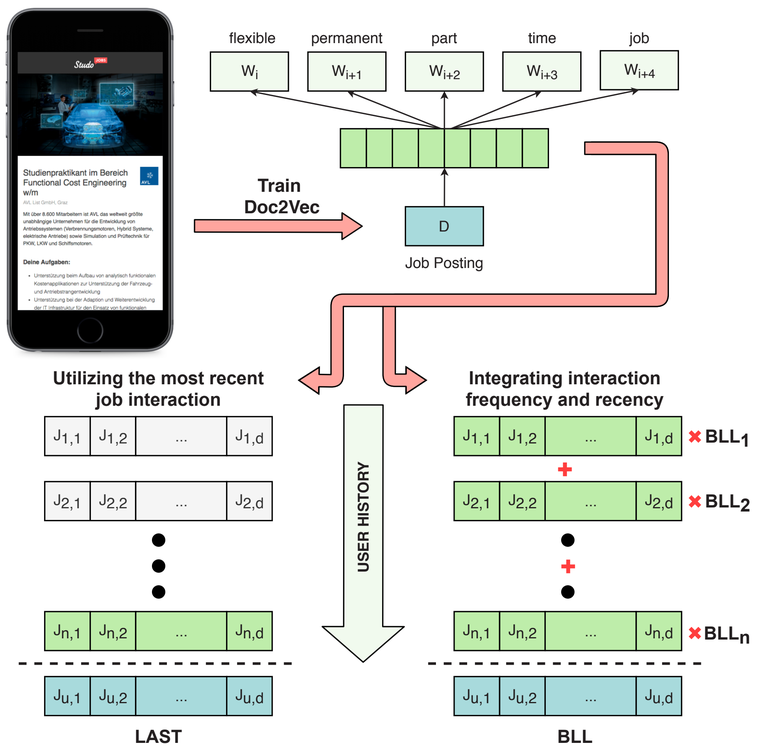
Should we Embed?
A Study on the Online Performance of Utilizing Embeddings for Real-Time Job Recommendations

Emanuel Lacic* — elacic@know-center.at
Markus Reiter-Haas* — markus.reiter-haas@moshbit.com
Tomislav Duricic — tduricic@know-center.at
Valentin Slawicek — valentin.slawicek@moshbit.com
Elisabeth Lex — elisabeth.lex@tugraz.at

* Both authors contributed equally to this work.

Study Setup

Impact of embeddings analysed under real-time constraints on the Austrian job platform Studo Jobs[a].



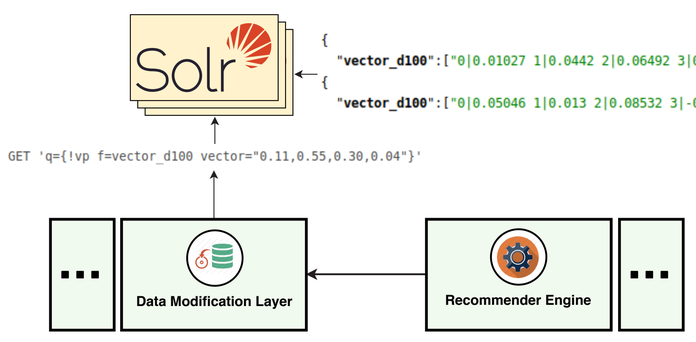
LAST. A natural way to utilize embeddings is to consider the most recent job posting and recommend top-k similar jobs.

BLL. Accounts simultaneously for both **frequency** and **recency**. Browsing behaviour for a reference vector is modelled by:

$$BLL_{u,j} = \ln(\sum_{i=1}^{n} (TS_{ref} - TS_{j,i})^{-d})$$

where d represents the time-dependent decay parameter.

Adapting for real-time job recommendations. Response times need to be **below** 100-200 milliseconds. Embeddings can be stored as payloads in Lucene and Cosine similarity can be used for retrieval.



[a]Superseded by the Talto career platform: https://talto.com

Conclusion

Using embeddings **significantly improves** the CTR and runtime performance for recommending **similar jobs**.

Factors of **recency** seem to be especially **influential** in this setting.

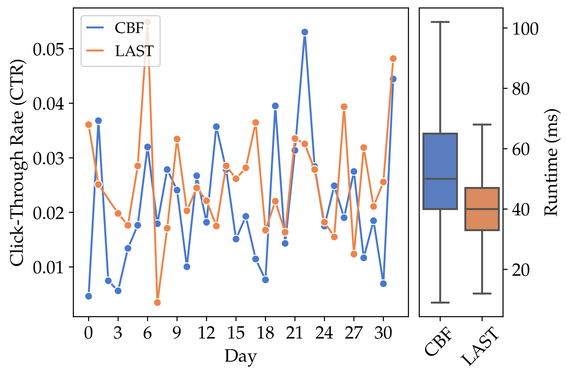
By combining embeddings based on the factors of **frequency** and **recency**, we can further **enhance** the online performance when personalizing the **homepage**.

Similar Jobs

Preliminary Analysis: Embedding size. In the case of Studo, embeddings larger than 100 **did not** contribute to a **higher CTR**, but **did increase** the overall **runtime** performance.

Baseline: Content-Based Filtering (CBF). Popular in many systems for recommending similar items. Easily adaptable when recommendations need to be served in real-time.

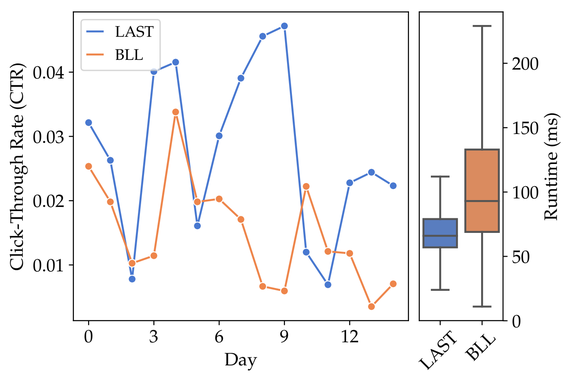
Impact of embeddings



 # Days: 32
 # Distinct Users: 8,576
 # Recommendation Requests: 31,968

Approach	CTR	↗	Runtime (ms)	↘
CBF	0.0194	18.04%	51	23.53%
LAST	**0.0229***		**39****	

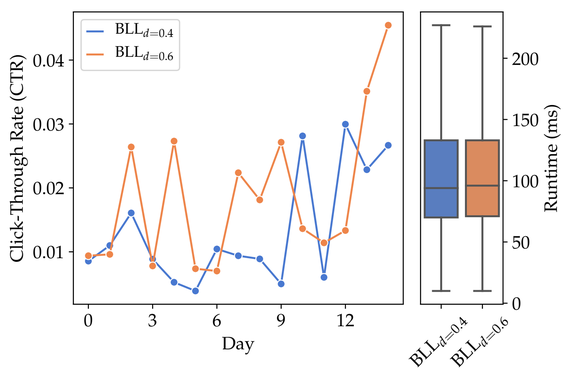
Influence of frequency and recency



 # Days: 15
 # Distinct Users: 4,715
 # Recommendation Requests: 18,464

Approach	CTR	↗	Runtime (ms)	↘
LAST	**0.0249****	75.35%	**67****	28.72%
BLL	0.0142		94	

Merit of recency



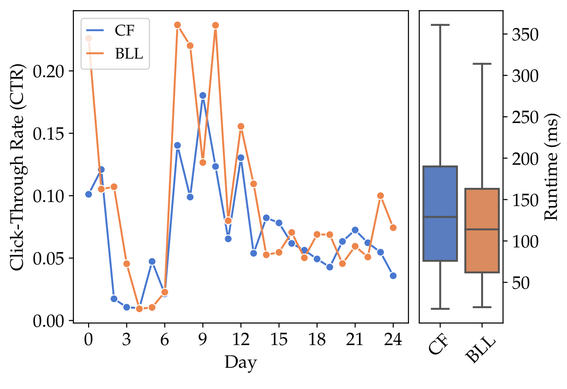
 # Days: 15
 # Distinct Users: 3,375
 # Recommendation Requests: 11,992

Approach	CTR	↗	Runtime (ms)	↘
$BLL_{d=0.6}$	**0.0174***	35.94%	97	2.06%
$BLL_{d=0.4}$	0.0128		**95**	

Homepage

Baseline: Collaborative Filtering (CF). One of the most explored and utilized techniques for personalizing a system in real-time. To account for cold-start users, we recommend the most popular job postings as a fallback. To provide recommendations in real-time, the inverted-index structure available in the Apache Solr search engine is used to find the k-nearest neighbors using the Cosine similarity metric.

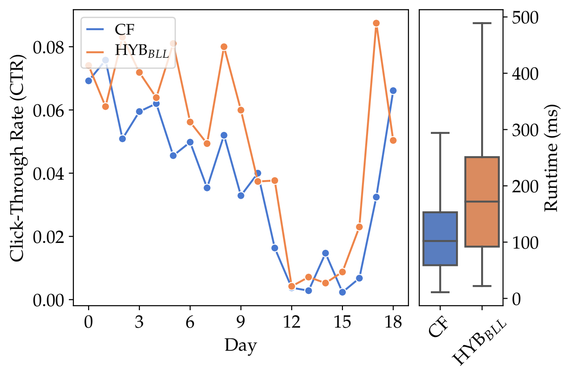
Influence of frequency and recency



 # Days: 25
 # Distinct Users: 9,620
 # Recommendation Requests: 26,334

Approach	CTR	↗	Runtime (ms)	↘
BLL	**0.0671***	15.69%	**114****	13.64%
CF	0.0580		132	

Combining frequency and recency



 # Days: 19
 # Distinct Users: 9,313
 # Recommendation Requests: 24,907

Approach	CTR	↗	Runtime (ms)	↘
HYB_{BLL}	**0.0471****	33.05%	172	38.37%
CF	0.0354		**106****	