Semantic Encoding of Review Sentences for Memory-Based Recommenders

Léo Laugier[1], Thomas Bonald[1],Lucas Dixon[2]

[1]Télécom Paris, Institut Polytechnique de Paris
[2]Google

DIG Seminar - 2/22/22 (!)

Contents

Contents

Introduction (1/3)

Sunday afternoon, you're at home and your partner/roommate/friend is out and texts you to know whether you would like to watch "Split" tonight, a movie you've never heard of.

What would you do?

Introduction (1/3)

Sunday afternoon, you're at home and your partner/roommate/friend is out and texts you to know whether you would like to watch "Split" tonight, a movie you've never heard of.

What would you do?

Use a search engine to obtain information about it.

- Have you seen this movie? 🙋

Introduction (2/3): Various levels of interaction **feedback**

- Have you seen this movie? 🙋
- How would you rate this movie? ☝, ✌, $\cdots$, 🖐

Introduction (2/3): Various levels of interaction **feedback**

- Have you seen this movie? 🙋
- How would you rate this movie? ☝️, ✌️, $\cdots$, ✋
- Can you say a word on what you liked or disliked? 🗣️

Introduction (2/3): Various levels of interaction **feedback**

- Have you seen this movie? 🙋
  →**Implicit feedback**
- How would you rate this movie?
  ☝, ✌, $\cdots$, ✋
- Can you say a word on what you liked or disliked? 🗣

Introduction (2/3): Various levels of interaction **feedback**

- Have you seen this movie? 🙋
  →**Implicit feedback**
- How would you rate this movie?
  ☝️, ✌️, . . . , 🖐️
  →**Explicit feedback**
- Can you say a word on what you liked or disliked? 🗣️

Introduction (2/3): Various levels of interaction **feedback**

- Have you seen this movie? 🙋
  →**Implicit feedback**
- How would you rate this movie?
  ☝, ✌, . . . , 🖐
  →**Explicit feedback**
- Can you say a word on what you liked or disliked? 🗣
  →**Review feedback**

Introduction (3/3): Motivations

- Show **personalized prediction scores** to users of a search engine
- Use **what users expressed** in past reviews to make predictions
- Extract "**rationales**" for predictions

Contents

Datasets (1/2): 2014 **Amazon** Product Reviews

- Narrowed down to the 20-**core movie** subset.

Datasets (1/2): 2014 **Amazon** Product Reviews

- Narrowed down to the 20-**core movie** subset.
- 212K reviews broke down into 3**M sentences**

Datasets (1/2): 2014 **Amazon** Product Reviews

- Narrowed down to the 20-**core movie** subset.
- 212K reviews broke down into 3**M sentences**
- Example:

User id	A10HHM2684NZD2 $\in U$
Item name	Spirited away $\in I$
Date	June 28, 2005

Datasets (1/2): 2014 **Amazon** Product Reviews

- Narrowed down to the 20-**core movie** subset.
- 212K reviews broke down into 3**M sentences**
- Example:

User id	A10HHM2684NZD2 $\in U$
Item name	Spirited away $\in I$
Date	June 28, 2005
Rating	5/5

Datasets (1/2): 2014 **Amazon** Product Reviews

- Narrowed down to the 20-**core movie** subset.
- 212K reviews broke down into 3**M sentences**
- Example:

User id	A10HHM2684NZD2 $\in U$
Item name	Spirited away $\in I$
Date	June 28, 2005
Rating	5/5
Review	This is one of the most well told tales i ' ve ever had the pleasure of experienceing . Essensially it's a story of how this little girl learns how to overcome adversity , at the same time learns how to care for someone . This movie will tug at your heart strings . A masterpiece for all ages to enjoy .

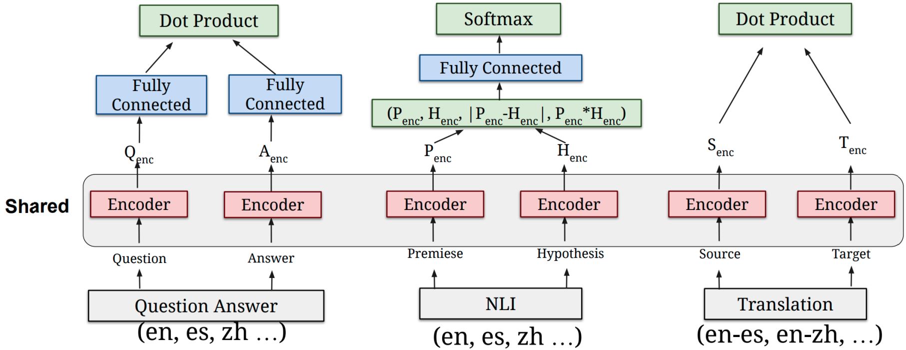
Datasets (2/2): Yelp business Reviews

- Narrowed down to the 20-**core** subset.
- 2M reviews broke down into 20**M sentences**
- Example:

User id	KvLseJGLjDXa4oqJf7KBGA $\in U$
Item name	The Cheescake Factory (Boston) $\in I$
Date	May 12, 2012
Rating	3/5
Review	The portions are RIDICULOUS and the menu gives me vertigo. When I eat off the regular menu I feel like I need to spend the next 24 hours at the gym ... BUT their seasonal pumpkin pecan cheesecake is amazing and, in my heart, I believe it is zero calories.

Contents

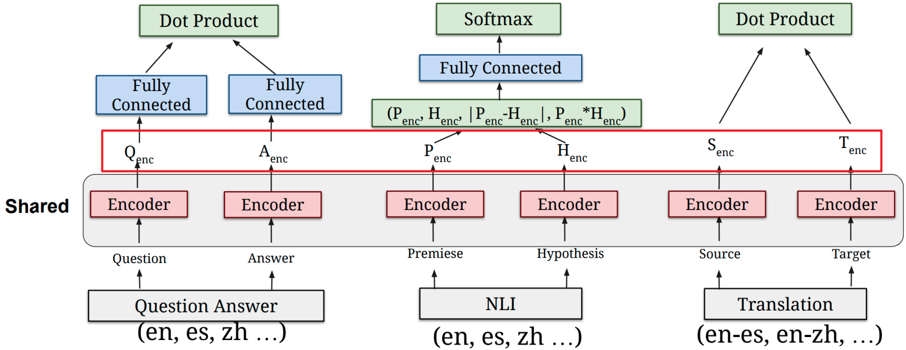
Semantic matching (1/5): **Universal Sentence Encoder** is trained to produce semantic embeddings [3]



Transformer [1] or **CNN** [2] based sentence embedding models provide a shared **encoder** across all tasks.

Pre-trained and available **on-the-shelf**!

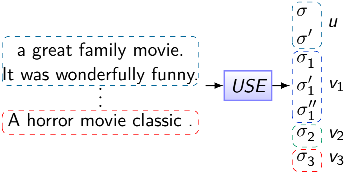
Semantic matching (1/5): **Universal Sentence Encoder** is trained to produce semantic embeddings [3]



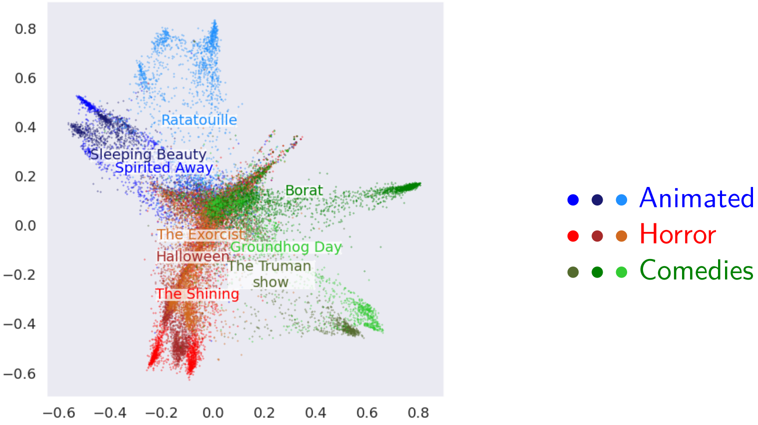
Transformer [1] or **CNN** [2] based sentence embedding models provide a shared **encoder** across all tasks.

Pre-trained and available **on-the-shelf**!

Semantic matching (2/5): Users are represented by their **sentence embeddings**

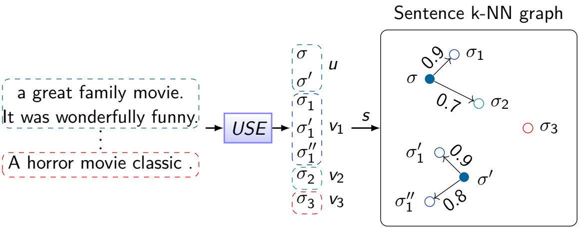


Semantic matching (3/5): Sentences are somehow **clustered** on categories



2D t-SNE projection of sentence embeddings.

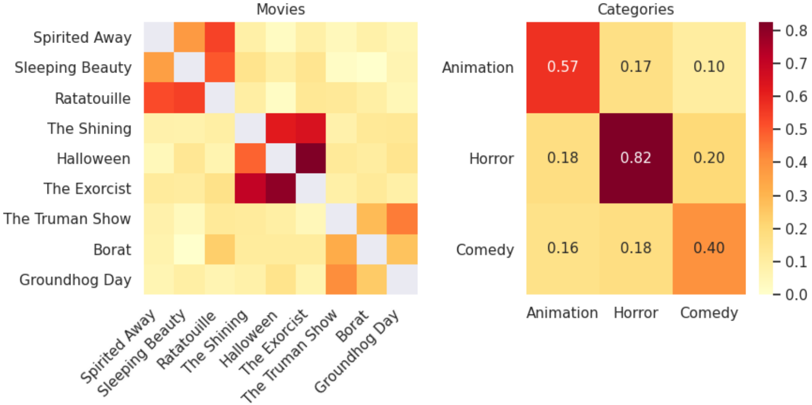
Semantic matching (4/5): We computed the **k-NN directed graph** of the **sentence embeddings**



Semantic match

A pair of sentences connected in the graph: (σ, σ_1) is a match while (σ, σ_3) is not.

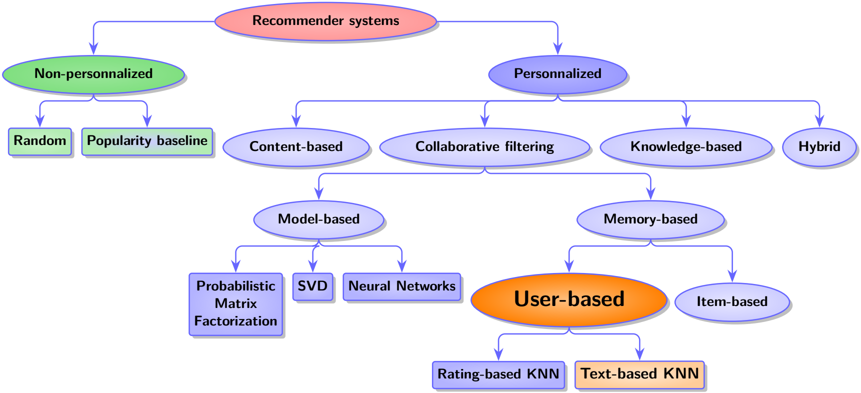
Semantic matching (5/5): **Sentence matching** as proxy for close semantics and preferences



Heatmaps of k-NN sentence embedding matches where $k = 10$. Rows and columns respectively represent head and tail vertices.

Contents

Recommender system (1/3): Hierarchy of systems



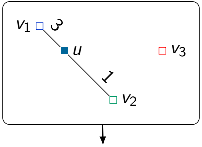
Recommender system (2/3): **Collaborative filtering** makes predictions from **preferences shared by users**.

Moshanin, CC BY-SA 3.0, via Wikimedia Commons

Recommender system (3/3): Memory-based systems are **k-NN regressors** on users

$$\hat{r}_{uj} = \frac{\sum_{v \in N_j^{k'}(u)} w(u,v) \cdot r_{vj}}{\sum_{v \in N_j^{k'}(u)} w(u,v)}$$

User k-NN regressor



$\hat{r}_{uj} = \frac{3}{4} \cdot r_{v_1 j} + \frac{1}{4} \cdot r_{v_2 j}$

r_{vj}: Ground-truth rating of user v on item j.
$N_j^{k'}(u)$: Set of k' nearest neighbor users of user u who have rated item i.
$w(u,v)$: **Weight of v for predicting u's rating.**

Pros:

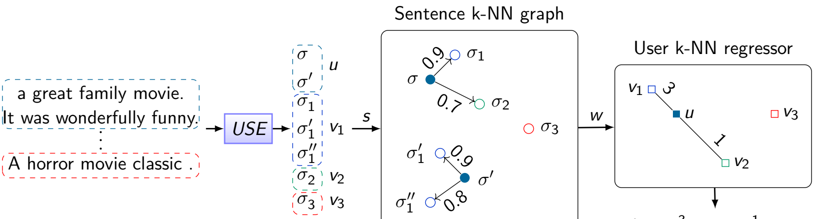
- Simplicity of implementation and understanding.
- Ease of creating explanations.

Cons:

- Less accurate than SOTA model-based systems.

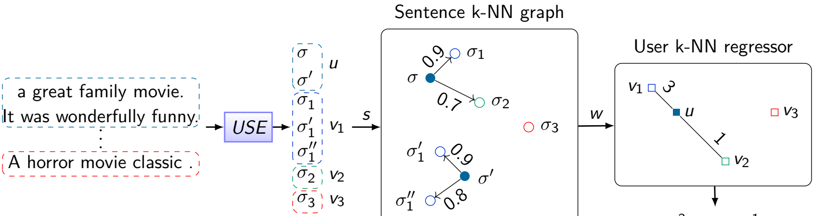
Contents

Text-KNN (1/1): We propose a **k-NN regressor** based on **counting semantic matches**



$$\hat{r}_{uj} = \frac{3}{4} \cdot r_{v_1 j} + \frac{1}{4} \cdot r_{v_2 j}$$

Text-KNN (1/1): We propose a **k-NN regressor** based on **counting semantic matches**

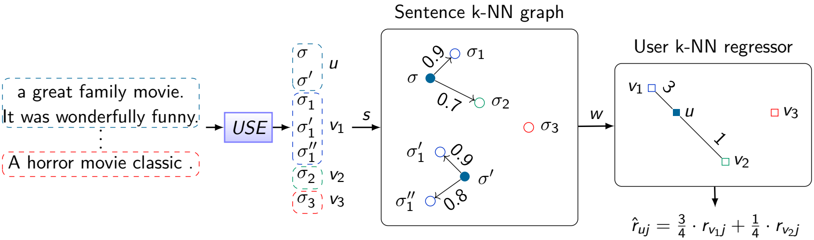


$$\hat{r}_{uj} = \frac{3}{4} \cdot r_{v_1 j} + \frac{1}{4} \cdot r_{v_2 j}$$

3 ways of computing the user weights $w(u, v)$

1. **One-to-One matching**: u and v have expressed similar preferences if v wrote at least one sentence in the semantic neighborhood of at least one sentence written by u.

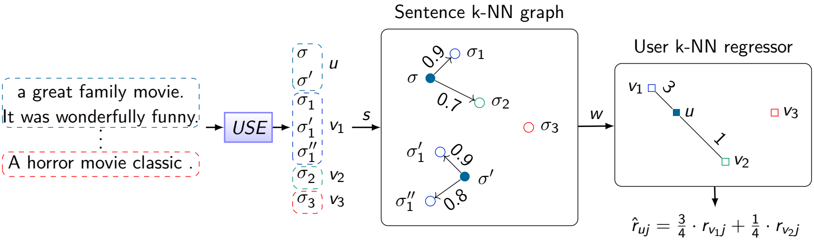
Text-KNN (1/1): We propose a **k-NN regressor** based on **counting semantic matches**



3 ways of computing the user weights $w(u, v)$

1. **One-to-One matching**: u and v have expressed similar preferences if v wrote at least one sentence in the semantic neighborhood of at least one sentence written by u.
2. **Many-to-One matching**: $w(u, v)$ counts the occurrence of v's sentences in the neighborhood of u's sentences.

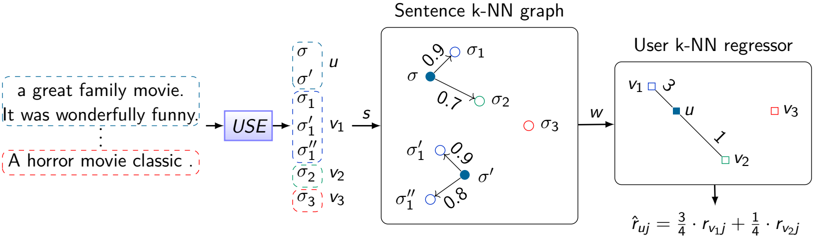
Text-KNN (1/1): We propose a **k-NN regressor** based on **counting semantic matches**



3 ways of computing the user weights $w(u, v)$

1. **One-to-One matching**: u and v have expressed similar preferences if v wrote at least one sentence in the semantic neighborhood of at least one sentence written by u.
2. **Many-to-One matching**: $w(u, v)$ counts the occurrence of v's sentences in the neighborhood of u's sentences.
3. **Many-to-Many matching**: $w(u, v)$ counts the number of sentence matches when considering all pairs of sentences written by u and v.

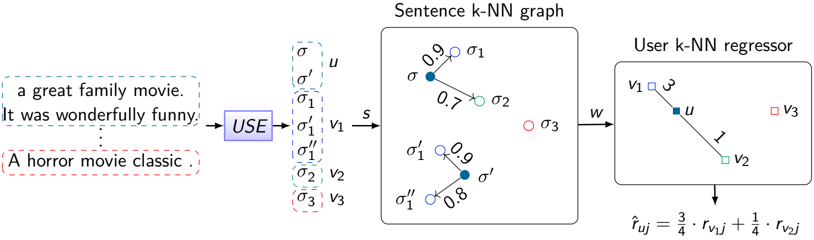
Text-KNN (1/1): We propose a **k-NN regressor** based on **counting semantic matches**



Options to compute the sentence weights $s(\sigma_1, \sigma_2)$

1. **Binary count**: $s(\sigma_1, \sigma_2) = 1$ iif (σ_1, σ_2) is a match and 0 otherwise.

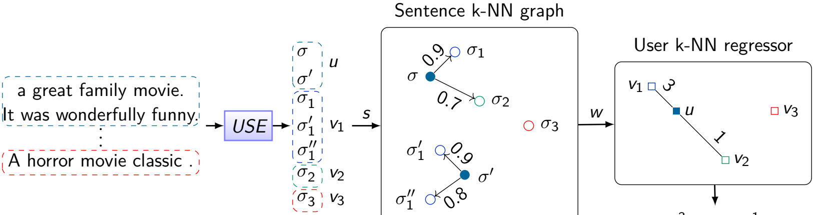
Text-KNN (1/1): We propose a **k-NN regressor** based on **counting semantic matches**



Options to compute the sentence weights $s\left(\sigma_1, \sigma_2\right)$

1. **Binary count**: $s\left(\sigma_1, \sigma_2\right) = 1$ iif $\left(\sigma_1, \sigma_2\right)$ is a match and 0 otherwise.
2. **Continuous similarity scores**: $s\left(\sigma_1, \sigma_2\right) = \frac{1+\text{cosine similarity}(\sigma_1,\sigma_2)}{2}$

Text-KNN (1/1): We propose a **k-NN regressor** based on **counting semantic matches**

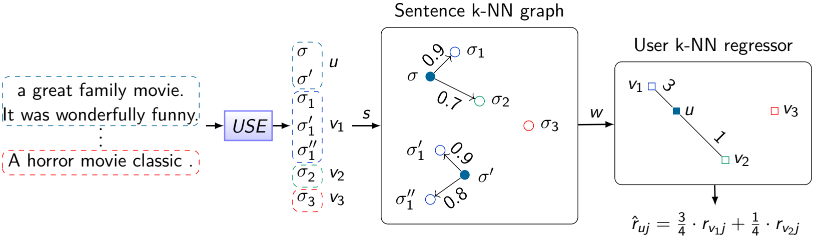


$$\hat{r}_{uj} = \frac{3}{4} \cdot r_{v_1 j} + \frac{1}{4} \cdot r_{v_2 j}$$

Options to compute the sentence weights $s(\sigma_1, \sigma_2)$

1. **Binary count**: $s(\sigma_1, \sigma_2) = 1$ iif (σ_1, σ_2) is a match and 0 otherwise.
2. **Continuous similarity scores**: $s(\sigma_1, \sigma_2) = \frac{1+\text{cosine similarity}(\sigma_1,\sigma_2)}{2}$
3. **Polarization**: cosine similarity(“I love DiCaprio”, “I hate DiCaprio”)$= 0.94$. Polarization multiplies by -1 the matches where the pair is made of sentences from opposite-sentiment reviews.

Text-KNN (1/1): We propose a **k-NN regressor** based on **counting semantic matches**



Options to compute the sentence weights $s(\sigma_1, \sigma_2)$

1. **Binary count**: $s(\sigma_1, \sigma_2) = 1$ iif (σ_1, σ_2) is a match and 0 otherwise.
2. **Continuous similarity scores**: $s(\sigma_1, \sigma_2) = \frac{1+\text{cosine similarity}(\sigma_1,\sigma_2)}{2}$
3. **Polarization**: cosine similarity("I love DiCaprio", "I hate DiCaprio")= 0.94. Polarization multiplies by -1 the matches where the pair is made of sentences from opposite-sentiment reviews.
4. **Per-item graph**: instead of the graph of all sentences, we focused on per-item graphs before aggregating.

Contents

Evaluation (1/2): Existing accuracy **metrics**

<table>
<tr><th>Task</th><th>Rating prediciton</th><th>Item recommendation</th></tr>
<tr><td>Predictive accuracy</td><td>RMSE, MAE, MSE</td><td></td></tr>
<tr><td>Classification</td><td colspan="2">Precision, Recall, ROC, ROC-AUC</td></tr>
<tr><td>Rank correlation</td><td colspan="2">Spearman ρ, Kendall τ, FCP</td></tr>
<tr><td>Next-interaction rank</td><td></td><td>HR@N, nDCG, MR, MRR</td></tr>
</table>

Evaluation (1/2): Existing accuracy **metrics**

Task	**Rating prediciton**	Item recommendation
Predictive accuracy	**RMSE**, MAE, MSE	
Classification	Precision, Recall, ROC, ROC-AUC	Precision, Recall, ROC, ROC-AUC
Rank correlation	Spearman ρ, Kendall τ, **FCP**	Spearman ρ, Kendall τ, **FCP**
Next-interaction rank		HR@N, nDCG, MR, MRR

Root-**M**ean-**S**quare **E**rror

$$\text{RMSE} = \sqrt{\frac{\sum\limits_{(u,i)\in \text{Test Set}} (\hat{r}_{ui} - r_{ui})^2}{|\text{Test Set}|}}$$

Evaluation (1/2): Existing accuracy **metrics**

Task	**Rating prediciton**	Item recommendation
Predictive accuracy	**RMSE**, MAE, MSE	
Classification	Precision, Recall, ROC, ROC-AUC	
Rank correlation	Spearman ρ, Kendall τ, **FCP**	
Next-interaction rank		HR@N, nDCG, MR, MRR

Fraction of Concordant Pairs

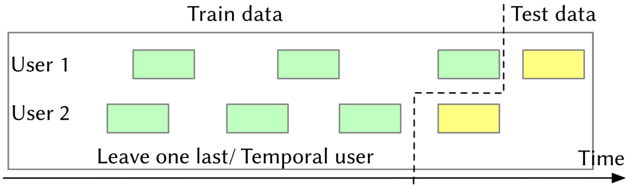
For two items i and j rated by the same user u, i.e. (u, i) and (u, j) in the Test Set, a pair $((r_{ui}, \hat{r}_{ui}), (r_{uj}, \hat{r}_{uj}))$ is:

- Concordant (c) iif $r_{ui} \neq r_{uj}$ and $\operatorname{sgn}(r_{ui} - r_{uj}) = \operatorname{sgn}(\hat{r}_{ui} - \hat{r}_{uj})$,
- Discordant (d) iif $r_{ui} \neq r_{uj}$ and $\operatorname{sgn}(r_{ui} - r_{uj}) \neq \operatorname{sgn}(\hat{r}_{ui} - \hat{r}_{uj})$,
- Ignored if $r_{ui} = r_{uj}$

$\text{FCP} = \frac{n_c}{n_c + n_d}$ ❗ Assumes several test items per user in the test set.

Evaluation (2/2): **T**ime-based **F**raction of **C**oncordant **P**airs

Leave One Last Item strategy splits the set in train/test [4].



For a user u, each pair $\left(\left(r_{ui}, \hat{r}_{ui}\right), \left(r_{uj}, \hat{r}_{uj}\right)\right)$ is made of *a* train item i and *the* test item j.

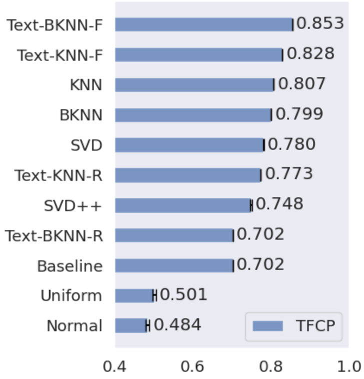
$\text{TFCP}(u) = \frac{n_c(u)}{n_c(u)+n_d(u)}$ indicates how well the system ranks the test item compared to the train items, according to u.

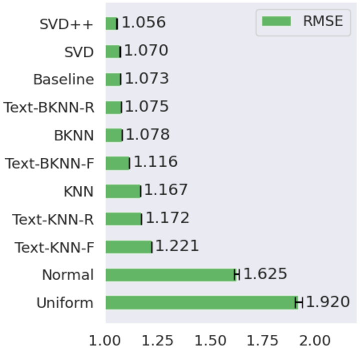
$\text{TFCP} = \sum_{u \in U} \text{TFCP}(u)$ macro-averages $\text{TFCP}(u)$ on all users.

TFCP generalizes AUC for non-binary variables.

Contents

Results (1/3): **Amazon** dataset



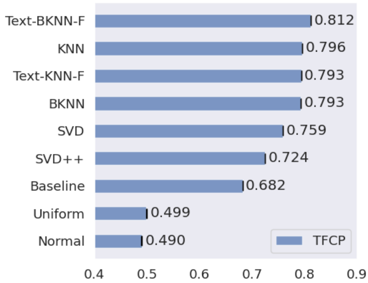


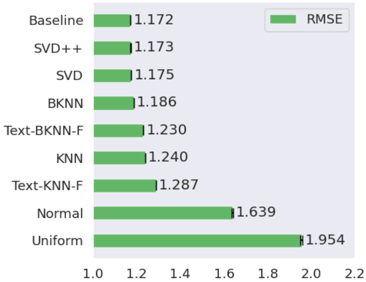
Baseline: $b_{ui} = \mu + b_u + b_i$

KNN: $$\hat{r}_{uj} = \frac{\sum_{v \in N_j^{k'}(u)} w(u,v) \cdot r_{vj}}{\sum_{v \in N_j^{k'}(u)} w(u,v)}$$

BKNN: $$\hat{r}_{uj} = b_{ui} + \frac{\sum_{v \in N_j^{k'}(u)} w(u,v) \cdot (r_{vj} - b_{vi})}{\sum_{v \in N_j^{k'}(u)} w(u,v)}$$

Results (2/3): **Yelp** dataset

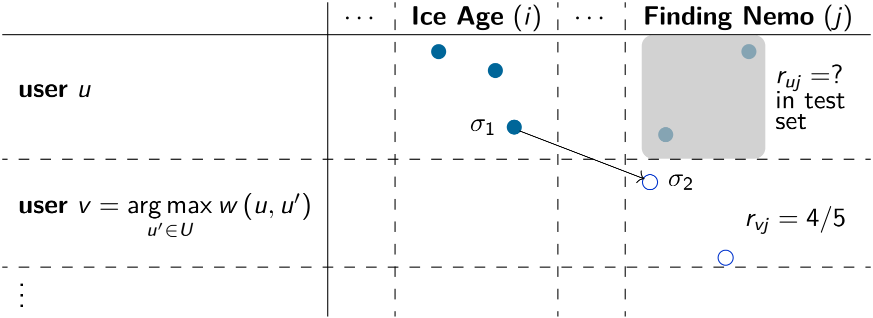




Note: Hyperparameters are not tuned on the Yelp dataset.

Results (3/3): **Scrutability** of recommender systems

Semantic matches could be used for endorsement of recommendations.



Text-KNN predicts that you will rate *Finding Nemo* with $\hat{r}_{uj} \approx r_{vj} = \mathbf{4/5}$ because you expressed "$\boldsymbol{\sigma_1}$" on *Ice Age* and it resembles what the user closest to you expressed when they reviewed *Finding Nemo*: "$\boldsymbol{\sigma_2}$".

Results (3/3): **Scrutability** of recommender systems

Head sentence σ_1 written by u on some item i	Tail sentence σ_2 written by v on u's test item j and matching σ_1
Cemetery Man – This highly entertaining little zombie movie from Italy has all the elements that make it a wonderfully dark horror - comedy in the same vein asEvil Dead 2 : Dead by DawnandAn American Werewolf in London .	**The Horde** – one of the best horror zombie movies of all the times , this movie is equal than 28 days later , in these days European horror movies are the best of the best . . good for Friday at night
Charlie's Angels – I like drew barrymore , she is the best angel out of the three .	**Fever Pitch** – Drew Barrymore , as always , is phenomenal .
Ice Age – While not perfect , it is full of laughs and beautiful computer animation .	**Finding Nemo** – Highlights : Spectacular computer animation ; hilarious , well - developed characters ; original plot .
Monsters, Inc. – Great for parents and kids (or people without kids) .	**Toy Story 3** – Great for kids and adults alike.
Island in the Sky – The Duke[1] does a great job in his role as Dooley - the plane's captain.	**The Alamo** – The Duke turns out one of his best performances , as well as putting together this film .

[1] A nickname for the American actor John Wayne

Contents

Conclusion

- Text-KNN integrates **review feedback** in traditionnal memory-based systems.
- It achieves **competitive results** when compared with baselines and SOTA systems.
- **Evaluation** of recommender systems is an **open-research** topic.
- Work submitted to **KDD'22**.

Future work

- Supervised or unsupervised **pruning** of irrelevant sentences in reviews, e.g. "I like this movie"
- **Better semantic embedding**, suited to recommendation
- Theory on why **ranking-based metrics** evaluate recommenders better than error-based metrics.

References I

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin.
Attention is all you need.
In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Yoon Kim.
Convolutional neural networks for sentence classification.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics.

References II

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil.
Multilingual universal sentence encoder for semantic retrieval.
In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 87–94, Online, July 2020. Association for Computational Linguistics.

Zaiqiao Meng, Richard McCreadie, Craig Macdonald, and Iadh Ounis.
Exploring data splitting strategies for the evaluation of recommendation models.
In *Fourteenth ACM Conference on Recommender Systems*, RecSys '20, page 681–686, New York, NY, USA, 2020. Association for Computing Machinery.

Semantic Encoding of Review Sentences for Memory-Based Recommenders

Léo Laugier[1], Thomas Bonald[1],Lucas Dixon[2]

[1]Télécom Paris, Institut Polytechnique de Paris
[2]Google

DIG Seminar - 2/22/22 (!)