CS224u experiment protocol

Lara Thompson
Principle Data Scientist @ Salesforce
lara.thompson@salesforce.com

March 12, 2023

1 Hypotheses

The central hypothesis of my project is that richer recommendations will result if user reviews/comments and item descriptions are included among the user and item features using a large language model (LLM) to encode them semantically.

A few questions I want to answer along the way include:

- How rich can "a simple baseline" sentence embeddings (Arora et al., 2017) be using encodings from an LLM (e.g. BERT) instead of GLoVe embeddings that lack context? If the early layers of BERT can be used, with early exiting a system such a system could be far more lightweight than a full LLM-based system, such as Sentence BERT (Reimers and Gurevych, 2019).
- How much lift does fine-tuning the sentence embeddings give the recommender?
- Can the same sentence embeddings be used to encode item and user features?
- Can user embeddings be optimized separately from item embeddings when text features are included?

2 Data

To evaluate sentence embeddings I will test with the task-specific datasets from GLUE (Wang et al., 2018).

To evaluate the recommender system as a whole, I will choose two from:

- wine reviews dataset from RecoBERT (Malkiel et al., 2020)
- a book reviews dataset that includes the social connections from LibraryThing (Zhao et al., 2015)
- a massive GoodReads dataset (Wan and McAuley, 2018)
- a Beer review dataset with additional features (McAuley et al., 2012)
- a climbing log dataset from sendage.com, a site that allows climbers to log, rate, grade, comment on and offer beta for climbs they've sent[1]; climbs have a description and location

I'll use whichever datasets prove easiest to load and process given the short time frame.

3 Metrics

GLUE tasks each have a standard metric that I'll use (accuracy, F1, and a mix of Pearson, Spearman and Matthews correlations).

Metrics relevant to evaluate a recommender system are precision, recall and coverage. Personalization is important to assess as well: it is often modelled as the average (cosine) distance between user recommendation vectors.

4 Models

To embed the text fields in as frugal a manner possible, I'll try "a simple" sentence embeddings with increasingly complex LLM encodings, and compare them with BERT-base and SBERT-base.

As recommender baselines, I'll use a pure popularity-based recommender ("everyone likes ***"), and simple user-item collaborative filtering ("you liked *, and users who also liked * tend to like ***").

A recommender based solely on sentence embeddings uses content-based filtering ("you liked * which is similar to ***"), as in RecoBert (Malkiel et al., 2020).

User-item interactions can be expressed as an unordered set, or as a sequence if there's a time

[1] "Beta" is the specific sequence of moves to ascend cleanly, aka "send".

ordering, in analogy to bag-of-words vs text sequences in NLP. The transformer architecture was adapted to recommenders in Transformers2Rec (de Souza Pereira Moreira et al., 2021); adding user and item features is easy in this model architecture. These are trained to predict masked items in a set/sequence or to predict the next item in a sequence, similar to large language modelling.

5 General Reasoning

I'll assess sentence embeddings primarily on how well they can be used for sentiment analysis and detecting semantic similarity. Simpler is better as I want a lean yet performant system. If I must use BERT fully I'll try to frame my system to allow as much pre-computation as possible.

As baseline recommender systems, I'll use a "popularity" recommender; a collaborative filtering system with no other user/item features; a content-filtering system with only text features. From there, the system complexity will grow as more features get added. The final system will be set/sequence based learning with all user/item features that improve performance.

6 Summary of Progress

So far, I've gathered all the datasets mentioned in section 2. I've begun testing BERT-base using various layer embeddings on the GLUE SST-2 task. I have "a simple" sentence embeddings based on BERT-base hidden layer states ready to test.

Most importantly, I got my home computer repaired so that my GPU is usable again.

I have yet to develop any recommenders, but I found various python libraries that will help[2].

The biggest unknowns at this point are how large the recommender + dataset can be for my system. I can always subset (by date or region) to iterate faster in the beginning and scale up for final evaluations.

References

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. *5th International Conference on Learning Representations, ICLR 2017*.

Gabriel de Souza Pereira Moreira, Sara Rabhi, Jeong Min Lee, Ronay Ak, and Even Oldridge. 2021. Transformers4rec: Bridging the gap between nlp and sequential / session-based recommendation. In *Proceedings of the 15th ACM Conference on Recommender Systems*, RecSys '21, page 143–153, New York, NY, USA. Association for Computing Machinery. https://github.com/NVIDIA-Merlin/Transformers4Rec.

Itzik Malkiel, Oren Barkan, Avi Caciularu, Noam Razin, and other. 2020. Recobert: A catalog language model for text-based recommendations. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1704–1714.

Julian McAuley, Jure Leskovec, and Dan Jurafsky. 2012. Learning attitudes and attributes from multi-aspect reviews. In *International Conference on Data Mining (ICDM)*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Mengting Wan and Julian J. McAuley. 2018. Item recommendation on monotonic behavior chains. In *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018*, pages 86–94. ACM. https://sites.google.com/eng.ucsd.edu/ucsdbookgraph/home.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Tong Zhao, Julian McAuley, and Irwin King. 2015. Improving latent factor models via personalized feature projection for one class recommendation. In *Conference on Information and Knowledge Management (CIKM)*.

[2] Implicit from a friend of mine; there are many others and I may just want to code the simpler ones from scratch anyway.