Yelp Recommendation:
Comparisons of Different BERT Models

A. Rane, J. Mizuno and S. Chikne

Northeastern University, Khoury School of Computer Sciences Boston
rane.am@northeastern.edu mizuno.j@northeastern.edu chikne.s@northeastern.edu

Github Repository

Abstract

Collaborative filtering using ratings has been consistently used to build recommender systems based on user ratings. However, these ratings are often 1-Dimensional representation that often resulting in a flattening of more multifaceted information that could potentially gained from user reviews. This paper aims to explore the feasibility of using BERT transformer based models to replace user ratings and directly predict user feature vectors from the text content of user reviews. The ALS algorithm has been used as a baseline to measure the accuracy of generated user feature vectors. We have also explored four variations of the Bert base model: RoBERTa, DistilBERT, ALBERT, DeBERTa and performed a comparative analysis based on error values.

Introduction*1*

The modern culinary landscape is filled with the cuisine of various cultures, styles and cost. For many consumers, it can be hard to try out new dishes due to the sheer volume of choices, and many restaurant owners can find it very difficult to differentiate themselves and attract new customers with such competition. To help both restaurants and customers, we decided to set up a recommendation system based upon Yelp reviews left by patrons across the nation.

The problem with many traditional, ranking based recommender systems is that modeling the user experience as a ranking inevitably results in a data loss as a rich experience is summarized into a single value. Textual reviews can in comparison provide much more multifaceted information such as the following aspects:

1. User preferences (e.g., “I prefer spicy food”)
2. Traits of the discussed item (“cheap and delivered quickly”)
3. Comparisons to others (“the curry has the best value in the city”)
4. The context of the interaction (“I went to this Burger joint as part of my friend’s birthday party”)

It is important to infer not only to what extent a user enjoys a dish, but also why and what factors led to such a judgment. We aim to integrate this unstructured data with traditional collaborative filtering data using outputs from different transformer models.

Background

We must be careful to make sure sentiments expressed in review texts are accounted for in their context. The primary weakness of many traditional methods was that they operated by topic-modeling and using bag-of-words methods that simply made recommendations based on lexical similarity. An effective recommendation system must consider the review and user holistically to get a more accurate perspective.

Traditional Approaches

1. Reviews topics are extracted using review topics extracted using topic models such as LDA. The limitation of this method is that the only information extracted from each user is a single topic and that topics are not associated with star ratings [Julian McAuley and Jure Leskovec. 2013]
2. Review topics and star ratings are mapped together but the sentiment of the review is not considered [Guang Ling, Michael R. Lyu, and Irwin King. 2014]

Copyright © 2020, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

3. Combines user sentiments, review topics but uses an interest variable instead of an explicit numeric rating [Qiming Diao, Minghui Qiu, Chao-Yuan Wu, Alexander J. Smola, Jing Jiang, and Chong Wang. 2014]

Limitations of Traditional Approaches

1. Overlook context. These approaches take a bag-of-words approach, and process individual tokens or n-grams while ignoring their order. In contrast, deep learning approaches can process each token with respect to its context tokens.
2. Operate solely based on lexical similarity. This is a pitfall because semantically similar reviews may have a low lexical overlap. On the other hand, deep learning approaches are based on distributed representations, which capture the semantics of the words.

In this project, we would aim to predict the matching vectors for collaborative filtering for each user using four different algorithms based on the BERT architecture

The dataset that we are using for this project will be provided by the Yelp dataset. The dataset would be available at: https://www.yelp.com/dataset. We are using three datasets, specifically the reviews, business and user data. The json files contain nearly seven million reviews from nearly two million users spread across eleven metropolitan areas. The review files contain the star rating, text and review reactions. Business data contain their location, star rating, cuisine tags, business attributes such as parking and more. User files contain user profile info along with their review count, friends, rating history.

Related Work*2*

In discussions prior to our testing and analysis we have considered several different methods to optimize our recommendation system. Firstly, for the before any model of deep learning was chosen, we considered the method of feature extraction from traditional approaches such as the less sophisticated bag of words and the more nuanced tf-idf or term-frequency and inverse-document-frequency methods. We explicitly rejected the bag of words method as it failed to consider the user's usage of words in specific context, only relying on the pure frequency of words. Furthermore, tf-idf was also rejected as although it offered a more comprehensive way of understanding a word's frequency based on its perceived significance and presence in multiple documents, it too was rejected due its lack of nuance in word positioning and semantics.

For models, we considered several methods including but not limited to RNNs, LSTMs and Attention Mechanisms. The first, known as Recurrent Neural Networks, is one the first neural networks that could learn and understand sequence-based data instead of instance-based data. However, due to its structure, it is computationally expensive and as sequences get larger, it gets harder to train on such data. Gradients may "explode" or "vanish" depending on data causing an unstable model. Given that we are using moderately long sequences in the form of yelp reviews this model was rejected.

Moreover, the second, known as Long Short Term are like the previous RNNs as this one is also a recurrent network, but also applies input, output and forget gates to the data to solve the "exploding/vanishing" problem RNNs faced. However, LSTMs require an incredible amount of memory and time to train and can be easy to overfit. As this project is in a limited timeframe with limited computational power this was rejected as well.

Finally, Attention Mechanisms is a transformer that is based on an attention mechanism utilizing an encoder and decoder. Through the use of an encoder, the mechanism encodes contextual information about a given word vector to allow subsequent data to utilize such context to interpret more accurately. However, because of its intricacy it can produce incredible amounts of parameters, and thus, it is extremely computationally and time expensive especially with long sequences. Again, just like LSTMs we rejected Attention Mechanisms for the similar reason.

We eventually decided upon the BERT [Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019)] series of models as they comprise the most advanced and effective models in the field so far. BERT is a transformer with multi-headed attention mechanisms, effectively producing multiple sets of representation of the input sequence, each encoding different characteristics of the input. Using a deep bidirectional language model, it tries to understand the context of before and after each input. Although this is also computationally expensive, the effectiveness of BERT cannot be overstated. When first released it surpassed human-level performances on the SQuAD dataset and inspired multiple different versions in the following years. Thus, if all other models are computationally expensive, we decided to use the best at interpretation and prolific with its versions.

Dateset*3*

From the Yelp dataset, we extract the columns for UserId, BusinessId, Star Rating and User Review Text. The reviews

Copyright © 2020, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

which had the most 'useful' ratings given by other users were selected to prevent spam reviews and inaccurate or poorly written reviews from affected the quality of the data. The UserId and BusinessId were converted to a series of unique integers to work with the Implicit package. The User review text was preprocessed to add separator and end tokens wherever necessary. The user ratings were used to generate the user feature vectors using the Alternating Least Squares algorithm with an Alpha of 40, 20 factors and was run for 20 iterations with regularization of 0.1.

Approach*4*

For this project, we utilized various Bert models to determine the best model for our Yelp dataset and as such, we started by pre-processing the Yelp review data via Bert guidelines. All BERT type algorithms are based upon Google's original BERT design and aim to decrease computation and memory usage to improve upon performance.

For each review, we utilized the user id, the business id the review was written for, the star rating and finally the review itself. The Adams optimizer [Kingma, Diederik & Ba, Jimmy. (2014)] has been used to train the models, represented by the given weights update operation:

$$w_t = w_t - w_{t-1} - \alpha\,(\widehat{m_t})/\left(\sqrt{\widehat{v_t}} + \epsilon\right)$$
$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$$
$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$$

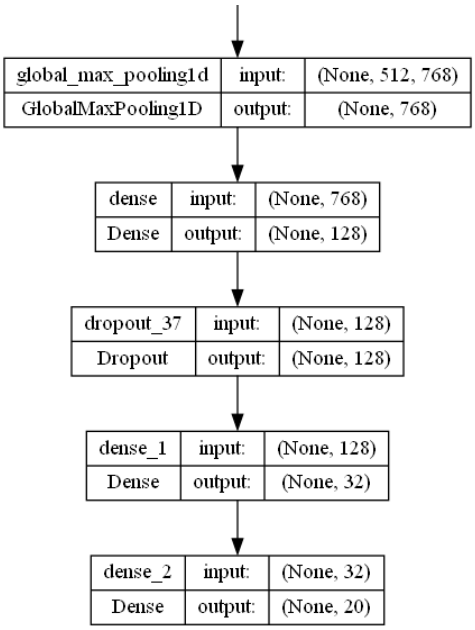
Where w_t is the weights at time step t, m_t is the moving average for the gradients, v_t is the moving average for the square of the gradients and g_t is the gradient vector at time step t.

To accommodate the review for the models, we fed the review through the base Bert tokenizer. As bert models only take up to 512 tokens, any review with not enough tokens are padded, while those that have too many are truncated from the middle. The middle truncation was chosen as recent research (C. Sun, X. Qiu, Y. Xu and X. Huang, 2019) has indicated that better results were found when truncating in the middle compared to those from the beginning or the end. The Bert tokenizer outputs the input_ids, which are tokenized representations of the words in the input data and the attention mask, which tells the model which tokens contain vital data and which can be ignored.

Subsequently, we aimed to predict the matching vectors(latent features) for each user using different algorithms based on the following BERT based architectures: BERT, roBERTa, alBERT, distilBERT and deBERTa.

The feature vectors for each user were calculated by the using the Alternating Least Squares [Y. Hu, Y. Koren and C. Volinsky 2008] Collaborative Filtering Approach implemented by the implicit python package

For each user a vector of size 20 was calculated as their user feature vector. In order to predict this value with the BERT based transformers, we have added an additional set of fully connected layers and 1D max pooling layers to obtain an output vector of the appropriate size.



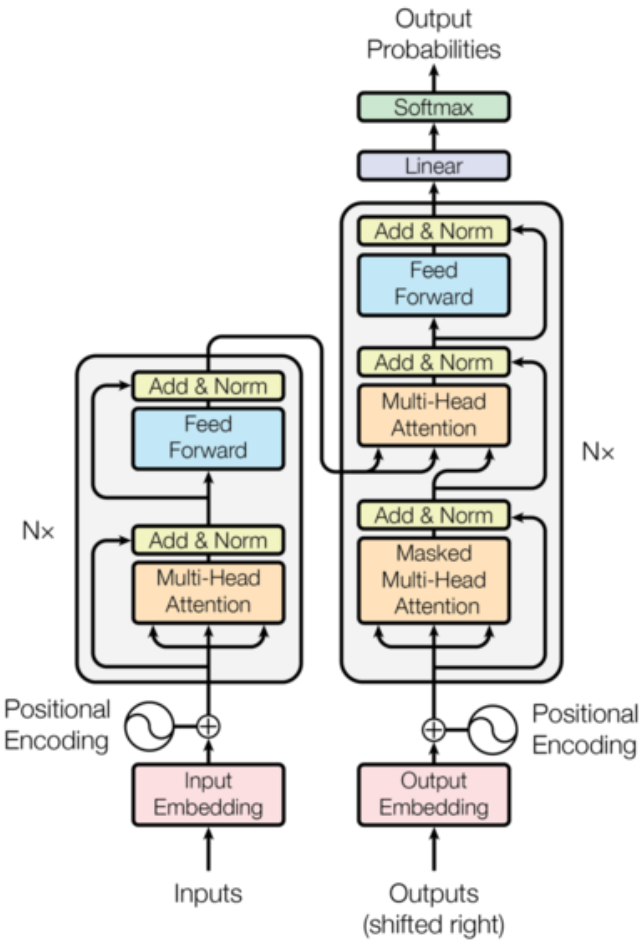
Custom Layers diagram added to the base transformer output

Each model was also trained to predict user ratings given the review as well, to ensure that it was able to extract the necessary information from the text reviews. Each model additionally operated with the same set of hyper-parameters as listing below:

- Tokenizer: BertTokenizer using 'bert-base-uncased' weights
- Weights from pre-trained 'roberta-base' model
- Max token length: 512 with truncation and padding
- Optimizer: Adam Optimizer
- Learning rate: 5e-05
- Epsilon: 1e-08
- Decay: 0.01
- Clip norm: 1.0
- Loss Function: Mean Squared Error

- Epochs: 2
- Batch Size: 36

The BERT model was used as a baseline to measure the effectiveness of other transformer variations



Original BERT architecture

In our work, we have directly accessed the output hidden state and added our own head layers to predict a vector output that would be uniform across all variations. We have kept the training data, pretrained weights, encoders and hyperparameters all constant to reduce the variations in each bert variant.

The Roberta Model [Liu, Yinhan & Ott, Myle & Goyal, Naman & Du, Jingfei & Joshi, Mandar & Chen, Danqi & Levy, Omer & Lewis, Mike & Zettlemoyer, Luke & Stoyanov, Veselin. (2019)] was included to measure the effectiveness of the dynamic masking while runtime. Although the original paper used a larger corpus of training data for Roberta, we have maintained the same pretrained weights for all models.

The ALBERT model [Lan, Zhenzhong & Chen, Mingda & Goodman, Sebastian & Gimpel, Kevin & Sharma, Piyush & Soricut, Radu. (2019)] was included to test whether the cross layer parameter sharing would negatively impact the performance compared to the BERT baselines

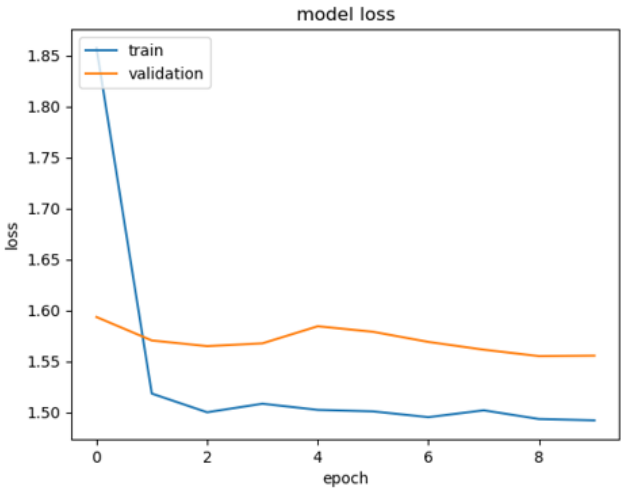
The DistilBERT model [Sanh, Victor & Debut, Lysandre & Chaumond, Julien & Wolf, Thomas. (2019)] aims to achieve the same accuracy as the baseline BERT model using the Knowledge Distillation Technique. We aimed to observe whether the model compression techniques used would be able to accurately replicate the BERT Performance.

The DeBERTA model [He, Pengcheng & Liu, Xiaodong & Gao, Jianfeng & Chen, Weizhu. (2020)] is a state of the art model which uses disentangled attention mechanism, an enhanced mask decoder, and a virtual adversarial training method for fine-tuning and has surpassed human accuracy on certain datasets.

Experiments and Results5

The following tables represent the MSE values for the test data on models trained on user rating and user feature vectors respectively. The BERT model have been used a baseline to compare with its modified variants.

The user feature vectors were generated by using the Alternatin least squares algorithm implemented by the Implicit package, using a configuration of 20 factors



Loss visualization across epochs

Since there was no significant changes in the metrics after the second epoch, all the models were trained for a maximum of 2 epochs on 10k data points. The first 10K and last 10K records of the dataset were chosen to avoid involving any position bias in the data.

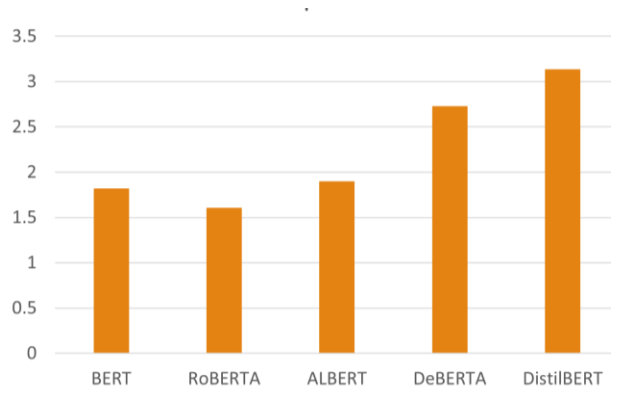
Data	BERT	roB-ERTa	alBERT	deBERTa	distilBERT
First 10K rows	1.6892	1.7164	1.9013	2.7278	3.1380
Last 10K rows	1.8788	1.9348	1.9831	1.4523	3.6799
Parameters	124,748,852	124,748,852	14,380,852	124,149,044	124,157,492

Mean Square Error values for the models trained on user rating

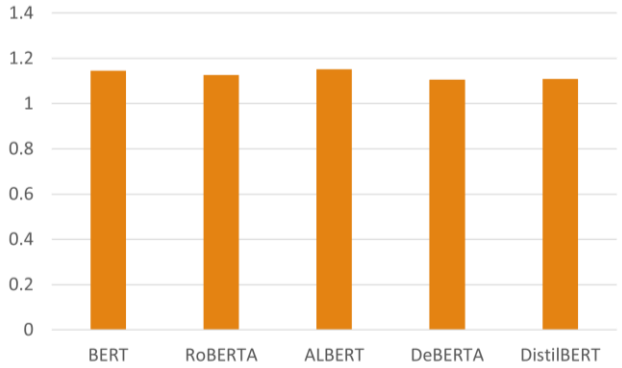
Data	BERT	roB-ERTa	alBERT	deBERTa	distilBERT
First 10K rows	1.8189	1.1214	1.1522	1.1059	1.1097
Last 10K rows	1.1171	1.1235	1.1389	1.1187	1.1102
Parameters	124,748,225	124,748,225	14,380,225	124,148,417	124,156,865

Mean Square Error values for the models trained on user factors

Conclusions*6*



MSE error values for the models trained on user star rating



MSE error values for the models trained on user star rating

While the BERT, RoBERTa and albert models were able to predict user ratings to a reasonable accuracy, none of the models performed particularly well at predicted user feature vectors. The error was consistent between the test and train data and also plateaued after the second epoch of training in most models, indicating the model was probably not overfitting or underfitting, Training for more epochs or incorporating more data both did not have a significant impact on the accuracy. This implies that the most likely method to improve accuracy would to be improve the method of taking reviews and ask guided questions (what did you like most about this establishment) prompting the user to enter more multifaceted responses.

The DeBERTA and DistilBERT models were quite surprisingly not able to perform as well on the user ratings prediction task, which might be due the lack of structured content present in the review text compared to the text corpus they were optimized for.

The roBERTa model performed the most consistently across both task, most likely owing to its dynamic masking during training across epochs.

However, the accuracy improvement is not as significant, considering that the model using nearly 10 times as many parameters as the ALBERT model. This would suggest that considering the targets of both time and accuracy, the ALBERT model would be a good fit for this dataset.

Future Scope*7*

While all the models were able to perform reasonably well predicting user ratings, prediction of user feature vectors was limited by several impeding factors, including

Copyright © 2020, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

computing limitation on reviews that were able to processed, and presence of brief user reviews that did not accurately capture the user sentiments in the entirety. A survey that is able to capture different aspects of user sentiment, such as 'What did you like most?' would perhaps provided better data to generate user feature factors.

References[8]

Sun, C., Qiu, X., Xu, Y., Huang, X. (2019). How to Fine-Tune BERT for Text Classification?. In: Sun, M., Huang, X., Ji, H., Liu, Z., Liu, Y. (eds) Chinese Computational Linguistics. CCL 2019. Lecture Notes in Computer Science(), vol 11856. Springer, Cham. https://doi.org/10.1007/978-3-030-32381-3_16

Guang Ling, Michael R. Lyu, and Irwin King. 2014. Ratings meet reviews, a combined approach to recommend. In Proceedings of the 8th ACM Conference on Recommender systems (RecSys '14). Association for Computing Machinery, New York, NY, USA, 105–112. https://doi.org/10.1145/2645710.2645728

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In Proceedings of the 7th ACM conference on Recommender systems (RecSys '13). Association for Computing Machinery, New York, NY, USA, 165–172. https://doi.org/10.1145/2507157.2507163

Qiming Diao, Minghui Qiu, Chao-Yuan Wu, Alexander J. Smola, Jing Jiang, and Chong Wang. 2014. Jointly modeling aspects, ratings and sentiments for movie recommendation (JMARS). In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '14). Association for Computing Machinery, New York, NY, USA, 193–202. https://doi.org/10.1145/2623330.2623758

Shalom, O.S., Roitman, H., Kouki, P. (2022). Natural Language Processing for Recommender Systems. In: Ricci, F., Rokach, L., Shapira, B. (eds) Recommender Systems Handbook. Springer, New York, NY. https://doi.org/10.1007/978-1-0716-2197-4_12

Y. Hu, Y. Koren and C. Volinsky, "Collaborative Filtering for Implicit Feedback Datasets," *2008 Eighth IEEE International Conference on Data Mining*, 2008, pp. 263-272, doi: 10.1109/ICDM.2008.22.

Kingma, Diederik & Ba, Jimmy. (2014). Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations.*

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. ArXiv, abs/1810.04805.

Liu, Yinhan & Ott, Myle & Goyal, Naman & Du, Jingfei & Joshi, Mandar & Chen, Danqi & Levy, Omer & Lewis, Mike & Zettlemoyer, Luke & Stoyanov, Veselin. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Lan, Zhenzhong & Chen, Mingda & Goodman, Sebastian & Gimpel, Kevin & Sharma, Piyush & Soricut, Radu. (2019). ALBERT: A Lite BERT for Self-supervised Learning of Language Representations.

Sanh, Victor & Debut, Lysandre & Chaumond, Julien & Wolf, Thomas. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter.

He, Pengcheng & Liu, Xiaodong & Gao, Jianfeng & Chen, Weizhu. (2020). DeBERTa: Decoding-enhanced BERT with Disentangled Attention.