Journal of Research High School

Volume 2022 (2) 159 - 167
ISSN 2688-3651 [online]

Deep Transformer based Intent Classifier (DTIC)

Connor Lee[1*], Albert Wang[2]

[1]Saratoga High School, Saratoga, CA USA, [2]Monta Vista High School, Cupertino, CA USA

Received October 9, 2022; Revised November 7, 2022; Accepted, November 15, 2022

Abstract

With the large-scale data availability and increasing popularity of digital service platforms, more and more users have begun reading reviews of listed items before deciding to perform a transaction. To determine the intention of these reviews, we will use a type of machine learning technique called sentiment analysis or intent mining to analyze the texts. A vital task of using Natural Language Processing (NLP) is to classify the opinion or sentiment from text into different sentiment categories. In this work, we propose to use a deep, fine-tuned BERT (bidirectional encoder representations from transformers) intent classifier model to analyze the texts. A xgboost classifier and a classical logistic regression model are used as baselines to compare with the BERT based model. These models are applied to one category of the open public Amazon book review data (Amazon book review data) to perform intent mining and survey the results. From our experiments, we observed the deep learning BERT model showed a higher performance than the other two baseline models. The accuracy of using a deep learning model can be boosted by up to 20%.

Keywords: BERT Language Model, Classification, Natural Language Processing

1. Introduction

As online retailers attempt to reign the digital market by satisfying customer needs, they rely on customer reviews and communication channels for a rating on their products. These reviews not only help retailers but also for customers to gauge the quality of their products. Reading reviews about an item gives users an idea of quality from other customers. By analyzing the reviews, businesses can survey the customers' opinions. To leverage public opinion, companies can build recommendation systems or better-targeted marketing campaigns. These recommendation systems can have distinct classifications: negative, positive or neutral. This process is called sentiment analysis or intent mining in the machine learning domain. Sentiment analysis uses techniques like natural language processing (NLP) and computational linguistics to identify subjective information.

In this paper, we create a TF-IDF (term frequency-inverse document frequency) feature vector (Cox, 1958) for traditional xgboost classifier and logistic regression models. We also use pre-trained BERT (Pedregosa, et al., 2011) to generate encoded vectors for fine-tuning classification. These classifiers are used to classify book reviews into multi-class intent categories in our experiments. The main objective of our work is to present a comparative study on different classifiers based on confusion matrix and accuracy while analyzing the features correlated to sentiment categories to identify the most efficient model in the text domain.

Through study, we expect to see that the latest fine-tuned BERT model should outperform classical logistic regression and xgboost classifier. Moreover, xgboost classifier should have better performance

* Corresponding Author
copanglee@gmail.com

Advisor: Dr. Wei Liu
weiliu.au@gmail.com

than logistic regression. The reason behind this is logistic regression and xgboost classifier cannot handle large numbers of sparse and high-dimensional features. The BERT-pretrained model which was pre-trained on a big corpus and the fine-tuned deep learning architecture can learn non-linearity and high-dimensional vectors very well without overfitting. Furthermore, xgboost classifier will perform better than logistic regression since it can learn not only linear boundaries, but also complex non-linear features. Our experiments validate these reasoning and our insights are derived by analyzing the results. On the other hand, although BERT-pretained plus fine-tuning models demonstrate its superiority over the rest traditional models, the large number of model parameters and large size of model is still a bottleneck for small-scale applications. We plan to investigate a distilled model to reduce the complexity of the BERT model and have the similar performance as what we have in our current study. We provide our summary and future work at the end.

1.1 Feature Engineering

TF-IDF feature vector

TF-IDF stands for term frequency-inverse document frequency and it is a measure to quantify the importance or relevance of string representations (words, phrases, lemmas, etc.) in a document amongst a collection of documents (also known as a corpus). TF-IDF is composed of two parts: TF (term frequency) and IDF (inverse document frequency). Term frequency refers to the frequency of a given term relative to the total terms occurrence in the document. Inverse document frequency (IDF) measures how common (or uncommon) a word/term is amongst the corpus. IDF is calculated below where t is the term (word) frequency measuring the commonness of, and N is the number of documents (d) in the corpus (D). The denominator is the number of documents in which the term (t) appears in.

$$idf(t, D) = log(\frac{N}{count(d \in D : t \in d)})$$

In our data, all the words in review text are used as corpus and each review text is treated as a document. For our baseline models, logistic regression and xgboost classifier model, we use TF-IDF as input features.

BERT Pretrained Embedding vector

BERT is a deep bidirectional pre-trained encoder model (Jacob, et al., 2019). It is available in two sizes: BERT-BASE, with 12 transformer layers and 768 hidden layers, and BERT-LARGE, containing 24 transformer layers and 1024 hidden layers. These two models are trained on the same datasets: BookCorpus, which contains 800M words, and the English Wikipedia, which holds 2500M words. Using BERT, we can extract features, namely word and sentence embedding vectors, from text data. These embeddings are useful for keyword search, semantic search, and information retrieval. They are also used as high-quality feature inputs to downstream tasks like clustering or semantic search. In our fine-tuned BERT classification model, we used embedding of the sentence-transformer model (Lee, et al, 2022) as initial input features. It maps sentences & paragraphs to a 384-dimensional dense vector space for multi-class classification.

1.2. Baseline Models and Proposed Model

In this section, we provide the details of two baseline models: xgboost classifier and logistic regression model and our proposed model fine-tuned BERT multi-class classifier.

Logistic Regression

Logistic regression is a classification model used in machine learning (Addanki, et al., 2019) (see Figure 1). The model predicts the probability of a dependent (*i.e* target variable) as a function of independent variables (*i.e.* input features). The dependent variable is the class that the model attempts to predict for a data point and the independent variables are input features. Logistic regression assumes that there exists a linear relationship between each independent variable and the logit of the dependent variable. The sigmoid function is used for mapping predicted value to probability (see Figure 1 (2)). The objective function is:

$$J(w) = \sum_{i=1}^{m} - y^{(i)} log\left(\phi(z^{(i)})\right) - (1 - y^{(i)}) log\left(1 - \phi(z^{(i)})\right) \quad (1)$$

where y is label , $\phi = 1/(1 + e^{-z})$ is the sigmoid function and $z = w1 \cdot x1 + w2 \cdot x2 + \ldots + wn$ is the linear function for mapping features to logit of label value.

Using logistic regression is straightforward to implement, interpret, and efficiently train. However, non-linear problems can't be solved with logistic regression because it has linear decision boundaries. In real-world scenarios, linearly separable data are rarely found. Furthermore, for intent classification in NLP, the TF-IDF feature vector is sparse and high-dimensional, leading to overfitting. Our experiment in the following validates this claim.

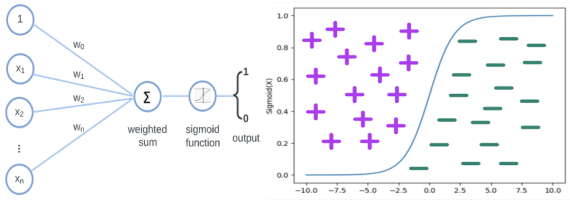


Figure 1. (1) Logistic regression function visualization (2) Sigmoid function for mapping probability (Addanki, et al., 2019)

Xgboost Classifier

Xgboost classifier is a tree-based sequential ensemble machine learning algorithm (Chen, & Guestrin, 2016). This model predicts a target variable by building a strong classifier from the number of weak classifiers in sequence. Firstly, a tree model is built from the training data. Then a second tree model is built, targeting to correct the prediction errors learned from the first model. As this step is continued, the models are sequentially added until either the prediction error achieves a certain threshold or the maximum number of models is added.

The training process of the xgboost classifier is shown in Figure 2. In the training dataset, there are a total 4 positive (+) and 6 negative (-) data points. At the beginning, the data is trained by the classifier 1 ($F_1(x)$). The classifier 1 incorrectly predicts two negative (-) and three positive (+) data points, which are highlighted with a circle. The weights of these incorrectly predicted data points are increased and provided to the next classifier 2 ($F_2(x)$) to fit for the error 1 (*i.e.* r_1) . The classifier 2 incorrectly predicts the one negative (-) and one positive (+) data point. The error 2 (*i.e.* r_2) is learned by next classifier 3. The classifiers are sequentially added until all the items in the training dataset are predicted correctly or a maximum number of classifier models are added. The optimal maximum number of classifier models to train can be determined using hyperparameter tuning. A combined final classifier is built to predict all the data points correctly.

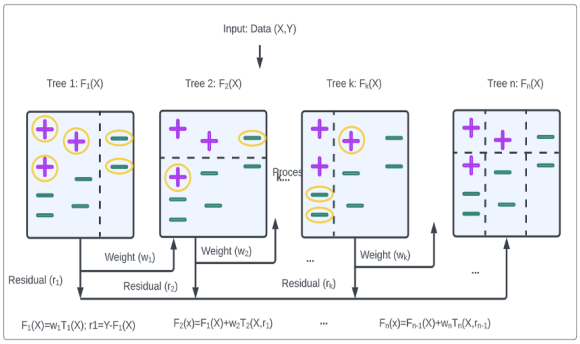


Figure 2. Gradient Boost Model

The objective function of the XGBoost model on the t'th iteration:

$$L^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t) \quad (2)$$

where $L^{(t)}$ is the cost function to be minimized for *n* training data points, the first term on the right side is the error generated by the previous classifier and the new fitted function. The second term is regularizer (Kingma, D. and Ba, J. 2015), which is used to control the model complexity.

Pretrained Language Modeling (BERT) and Transfer Learning

Recently, transformer-based pretrained language models have demonstrated state-of-the-art performance in the natural language processing

(NLP) domain. For example, BERT (Vaswani, et al., 2017) have achieved outstanding performance through masked self-supervised pre-training and transformer-based modeling. Transformer language modeling with pretraining and word representations combined with transfer learning (Lisa, et al., 2010), has significantly improved many natural language understanding (NLU) tasks, such as text classification and natural language inference. In the following, more details of the BERT model and transfer learning are introduced.

BERT is designed for training using self-supervised learning, wherein the model learns the context of a sentence during training by having a masked language model. The masked language model randomly masks tokens from the input text to predict the original masked word based on its context. This method allows BERT to consider the meaning of words based on the context in the training phase, thereby enabling the BERT language model to achieve performances comparable to those supervised learning with the help of human labeling cost. Furthermore, the next sentence prediction task allows the relationships between sentences to be learned based on text-pair representations. By applying next sentence prediction, BERT can be utilized in many downstream tasks such as question and answering as well as natural language inference by understanding the relationship between two sentences. In our approach, we will use it for sentiment multi-class classification tasks.

Transfer learning (Martín, et al., 2015) improves performance by sharing the parameters of a model that has been trained on similar task data in advance, unlike conventional machine learning that performs individual and single-task learning. In transfer learning, previously learned tasks are used when learning a new task; therefore, it is more accurate, and requires less training data. Moreover, its learning process is faster than that of a single-trained machine learning model. BERT can be applied to downstream tasks by fine-tuning all pretrained parameters.

BERT Fine Tuning Model Architecture (our approach)

In this work, we implemented a fine-tuned deep learning model for multiclass intent classification downstream task by utilizing BERT-based pretrained encoder and adding an output layer for classification.

The BERT-based model contains an encoder with 12 transformer blocks, 12 self-attention heads, and the hidden size of 768. BERT takes an input of a sequence of no more than 512 tokens and outputs the representation of the sequence. The sequence has one or two segments that the first token of the sequence is always [CLS] which contains the special classification embedding and another special token [SEP] is used for separating segments. The final hidden state h of the first token [CLS] is taken as the representation of the whole sequence. A softmax classifier is added to the top of BERT to predict the probability of label C: $p(c|h) = \text{softmax}(W_h)$, (1) where W is the task-specific parameter matrix. We fine-tune all the parameters from BERT as well as W jointly by maximizing the log-probability of the true label. We applied ADAM (adaptive moment estimation) as an optimizer and categorical cross-entropy loss as a loss function. The fine-tuned model architecture is as follows (Figure 3): tokenizer, encoding; output layer; softmax.

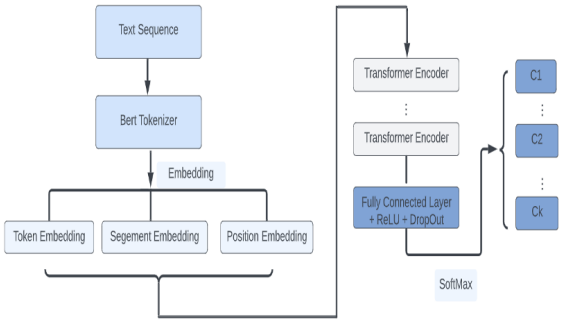


Figure 3. Fine-tuned Bert multi-classification model

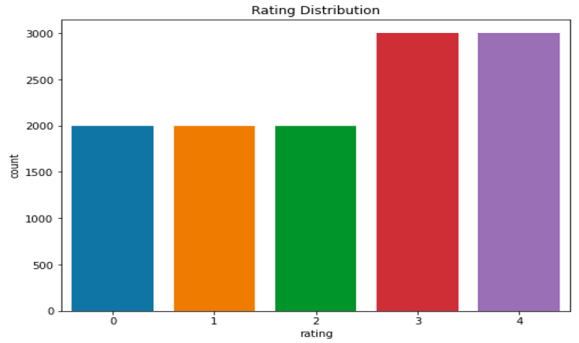
2. Materials and Methods

We evaluated the performance of the BERT fine-tuned classification models on an amazon open public book review dataset (Amazon review data, 2018) and compared it to the logistic regression and xgboost classifier. The experiments demonstrate the classification performance depending on the model architecture. The results show the BERT classifier performs much better than logistic regression and xgboost classifier.

2.1 Book Review Classification and Summary Dataset

The amazon book review dataset contains product reviews from Amazon, including 142.8 million reviews spanning May 1996 - July 2014. This dataset includes reviews (ratings, text, helpfulness votes), product metadata (descriptions, category information, price, brand, and image features), and links (also viewed/also bought graphs). Full dataset has 12,000 records and is randomly split for training and validation. Target variable is the review rating value(total 5 levels of rating: 1,2,3,4,5) and its distribution is shown in Figure 4 (1). A detailed description of an example from the dataset is presented in Figure 4 (2). Each data point has 3 columns: rating (0-5), reviewText and Summary. These three features contain the most relevant information to the rating classes.

- reviewText - text of the review (heading).
- rating - rating of the product.
- summary - summary of the review (description).



	rating	reviewText	summary
0	5	This book was the very first bookmobile book I bought when I was in the school book club. I loved the story then and I bet a dollar to a donut I will love it again. If my memory serves, I bought this book in 5th grade. That would have been about 1961. I am looking forward to reliving the memories.	50 + years ago...
1	1	When I read the description for this book, I couldn't wait to read it. Once I downloaded it to my Kindle, I found it extremely hard to keep reading it. To be honest, I stopped reading halfway through the book. It began slow and remained a slow, uninteresting read. It lacked passion; not making love passion, but passion for life. Neither Jada or Aaron were interesting characters and the story was too, too 'everything is perfect'. Everybody is just so understanding and accommodating--the bit of drama with his father and her grandmother was blah. To give an example of what I mean (and there are MANY), Aaron finally finds out almost half way into the book that he fathered a child 5 years ago--keep in mind that Aaron and Jada kept in contact for a few months after he left and she never mentioned to him that she was pregnant. When she finally tells him he has a son, Aaron becomes overwhelmed with emotions (misty eyed) knowing that he's a father. WTH! I think most men would be upset/angry to know that they had a child and the woman they love never told them! Not in this book; it's all good--all is well; NO PASSION. OMG, don't let me get started on the scene when father and son finally meet. It was so over the top, it made me want to barf. Maybe if this scene had been towards the beginning of the book, it would have been touching. In the middle of the book after just too much blah it made me thrown down my kindle. It was like eating a sweet dessert that was so sweet it made your teeth hurt. This book had a good theme, but no follow-through. Also, if we're to believe that Jada graduated from Harvard, then let it be reflected in her speech and her job. We're told that she has this wonderful personality which caused Aaron to fall in love with her--never saw it. If the author had described her as depressed and foolish--Yes! I would believe it in a heartbeat. She was boring. This book was so frustrating to read. I truly struggled to read it as far as I did. I finally stopped torturing myself because I realized it wasn't going to get any better.	Boring! Boring! Boring!

Figure 4. Overview of review data. (top) Rating Distribution, (bottom) one sample of Review text

2.2 Evaluation Metric

The accuracy score and confusion matrix are used as the evaluation metrics in this study.

Accuracy is one metric for evaluating classification models. Informally, accuracy is the fraction of correct predictions from our model. The calculation is:

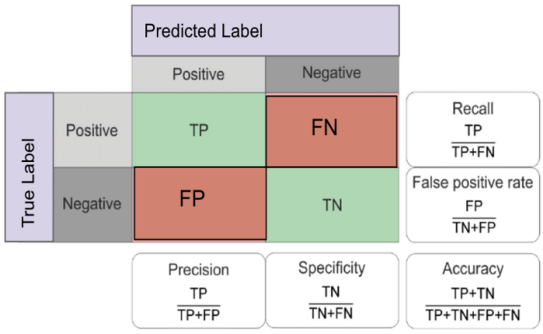
$$\text{Accuracy} = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions} \qquad (3)$$

For binary classification, accuracy can also be calculated in terms of positives and negatives:

Accuracy= (TP+TN)/((TP+TN)+F+FN) (4)
where (TP: true positive, FP: false positive, and FN: false negative).

Confusion matrix is also often used to measure the performance of a classification algorithm, for instance: recall, specificity, accuracy, and precision classification metrics are easily obtained from confusion matrix, which is a table with 4 different combinations of predicted and actual values (Table1).

Table 1. Performance Metrics: Confusion Matrix . It is a table that is used to define the performance of a classification algorithm. It visualizes and summarizes the performance of a classification algorithm. The detailed explanation of each term in the table is listed in the "Note" below.



Note:
True positive (TP): Observation is predicted positive and is actually positive.
False positive (FP): Observation is predicted positive and is actually negative.
True negative (TN): Observation is predicted negative and is actually negative.
False negative (FN): Observation is predicted negative and is actually positive.
Precision: The fraction of positive values out of the total predicted positive instances.
Recall: The fraction of positive values out of the total actual positive instances
Accuracy: Accuracy gives the proportion of the total number of predictions that are correct.

Compared to the machine learning classification metrics like "accuracy" give less useful information, which is simply the difference between correct predictions divided by the total number of predictions. Confusion matrix is useful for deep analysis of model performance since it directly shows true positives, false positives, true negatives and false negatives.

3. Results

3.1 Logistic Regression

We utilized the sklearn package to build pipeline and grid search to perform hyper-parameter tuning (Pedregosa, F. et al 2011). The TF-IDF frequency features are created by pipeline. The responsible variable is the rating values={1,2,3,4,5}. Since it is a multi-class prediction, the training algorithm uses the one-*v.s.*-rest (OvR) scheme and the regularization is applied. After tuning, the best learning rate is 2 and L2 regularizer is recommended. Figure 5 shows the confusion matrix on training and validation set. Table 2 shows the accuracy for each class.

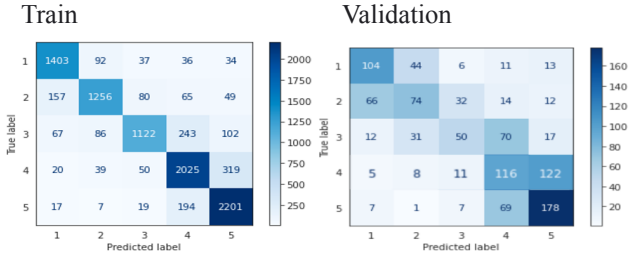


Figure 5. logistic regression output

Table 2. Accuracy of each class for Logistic Regression

Accuracy	Rating 1	Rating 2	Rating 3	Rating 4	Rating 5	Accuracy
Training	87.46	78.15	69.25	82.55	90.27	82.30
Validation	58.42	37.37	27.78	44.27	67.94	47.62

3.2 XGBClassifier

Same as logistic regression, we also utilized the sklearn package to build pipeline and grid search to perform hyper-parameter tuning (Pedregosa, et al., 2011). After gridsearch, the best parameter are: {'clf__max_iter': 20, 'clf__penalty': 'l2', 'vect__max_df': 1.0, 'vect__ngram_range': (1, 2)}, where clf__max_iter is the max number of training epochs, clf__penalty is L2 regularizer, vect__max_df is the maximum document frequency and vect__ngram_range includes unigrams and bigrams. Figure 6 shows the confusion matrix on training and validation set. Table 3 shows the accuracy for each class.

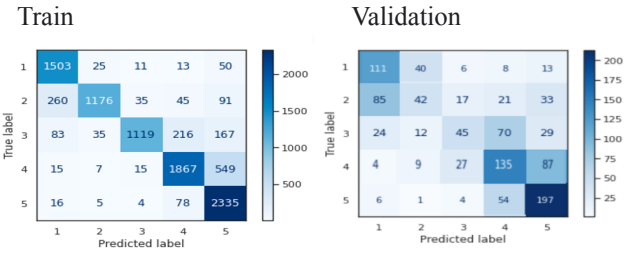


Figure 6. xgboost classifier output

Table 3. Accuracy of each class for XGBClassifier

Accuracy	Rating 1	Rating 2	Rating 3	Rating 4	Rating 5	Accuracy
Training	93.82	73.17	79.07	76.11	95.77	88.28
Validation	62.35	42.92	25.00	51.53	75.19	49.81

3.3 BertClassifier

In our implementation,we used the transformer library of Hugging Face (Transformers, https://huggingface.co/docs/transformers/index). It contains PyTorch implementation of state-of-the-art NLP models including BERT and pre-trained model weights. We created a BertClassifier class with a BERT model to extract the last hidden layer of the [CLS] token and a single-hidden-layer feed-forward neural network as our classifier. Figure 7 shows the confusion matrix on training and validation set. Table 4 shows the accuracy for each class. Figure 8 (1) shows the training loss and validation loss with the number of epochs. Figure 8 (2) visualizes and summarizes the accuracy of logistic regression *vs.* xgboost classifier *vs.* BertModel on validation sets.

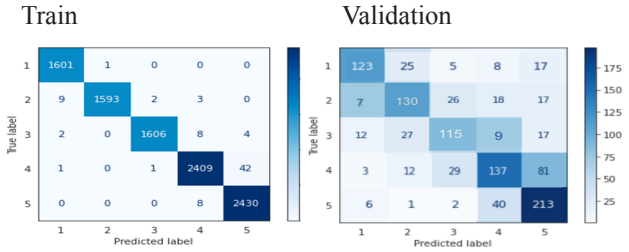


Figure 7. Bert classifier output

Table 4. Accuracy of each class for BERTClassifier

Accuracy	Rating 1	Rating 2	Rating 3	Rating 4	Rating 5	Accuracy
Training	99.93	99.13	99.14	98.20	99.67	99.16
Validation	69.10	65.65	63.88	52.29	81.29	65.45

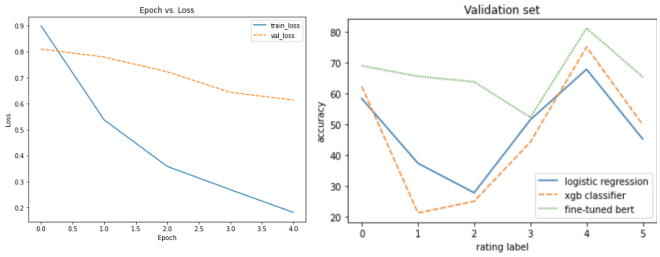


Figure 8. (left) Bert classifier Train and Validation loss vs. Epoch, (right) Bert vs. xgboost vs . logistic regression validation set performance

To summarize, the experiments above show the confusion matrix, accuracy of three models as well as the training/validation loss of the BERT model. We can see that a fine-tuned BERT model outperforms classical logistic regression and xgboost classifier. In the training phase, the accuracy goes up from 82.30% to 88.28% to 99.16%. During the validation phase, the accuracy increases from 47.62% to 49.81% to 65.45%. For each class prediction, the accuracy also increases from the logistic model to xgboost classifier to BertModel. The result validates the limitation of logistic regression and xgboost classifier which cannot handle large numbers of sparse and high-dimensional features. Furthermore, logistic regression can only learn linear boundaries. The BERT-pretrained model was pre-trained on a big corpus and the fine-tuned deep learning architecture can learn non-linearity and high-dimensional vectors very well without overfitting.

4. Discussion

4.1 Transformer Interpreter

To further support the deep learning model's performance, we ran a transformer interpreter (Charles, n.d.), which is a model explainability tool designed to work exclusively with the transformers package. It produces a dictionary of word attributions mapping labels to a list of tuples for each word and its attribution score. These explainer attribution scores are highlighted and display the visualization in-line with weights.

In the examples below (Table 5), each rating data point with the true class is displayed and corresponding attribute with higher-weight related to target variable is highlighted. For instance, for the first review text: *"The premise of this book was totally disgusting and I 'm sorry I read it. I deleted it. Don 't bother if you value your sanity". "disgusting"* has more weights related to our target rating value (=1, very low raing). It is highlighted to reflect its corresponding low rating and the summary *"Disgusting premise"* also validates *"disgusting"* consistent with the content from our ground-truth data. The rest of the examples are the same as this interpretation.

4.2 Embedding Visualization

Furthermore, to have a good understanding of the review text, we generate the embeddings from trained models and visualize them in tensorboard (Martín, et al., 2015) shown from Figure 9. Figure 9 (1) shows 3-D visualization, each data point is the embedding vector learned by model and represents corresponding review text. If the mouse hovers over one particular data point, it also shows its content (*i.e.* raw review text), as you can see from Figure 9 (2). In the rest of this section, we list one review text with high rating and its nearest neighbor (shown in Figure 10); same for one review text with low rating and its nearest neighbor (shown in Figure 11). These two examples demonstrate the embedding vectors learned from our deep learning models can capture the content and sentiment very well.

Table 5. Bert classifier attribute visualization

Negative Neutral Positive

Rating	ReviewText	Summary
1	The premise of this book was totally disgusting and I 'm sorry I read it. I deleted it. Don 't bother if you value your sanity.	Disgusting premise
2	This is not a good ghost story , nor is it a romance. To me it was a poorly written Har le quin romance with a lot of sm ut thrown in. The author doesn't appear to know what genre she's trying to capture. There were numerous typos which I found annoying. The author also incorporated Elvis ' name, but did it in a way that I found disrespectful as well as inaccurate .	Lousy Read
3	I really thought this book was going to be better from the reviews. Maybe it was more of a teenage type book than an adult one. I didn't read the second book in the series was not that interested.	An okay read
4	Way too short. It is a good, fun story. It pulls you in and keeps you reading. It 's not that it was really too short, just that it was so fun , I tore through it too fast. It 's well written with very few errors. What few there were did not stop the flow.	Good... but ;)
5	This was a book that I thoroughly enjoyed from the beginning to the end. The story line was full of details and kept me involved and entertained. I loved the characters and their quirks. The story was smoothly written. Secret agents and art. Who would have believed how much fun and add a twist of fate and love. I will read this author again. Enjoy this book and you won 't be disappointed.	Wow and wonderful read with a twist

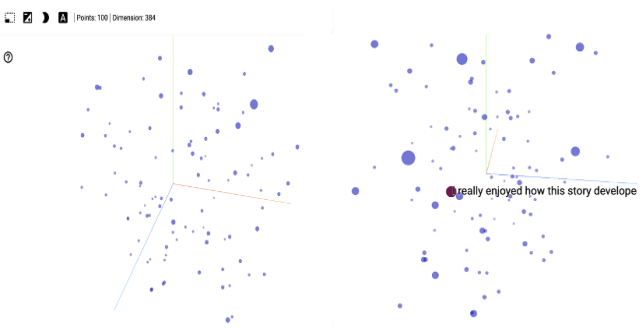
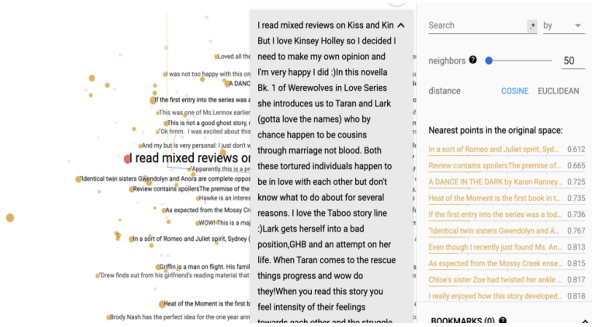


Figure 9. (left) 3-D TensorBoard UI, (right) Data Point and Label visualization

Review Text Visualization (Positive):



Nearest neighbor Text Visualization (Positive):

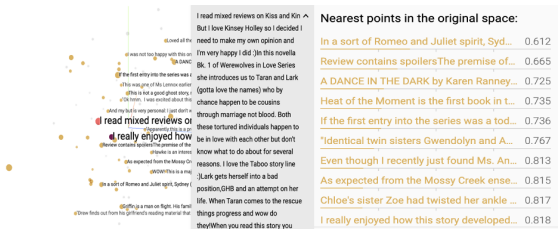
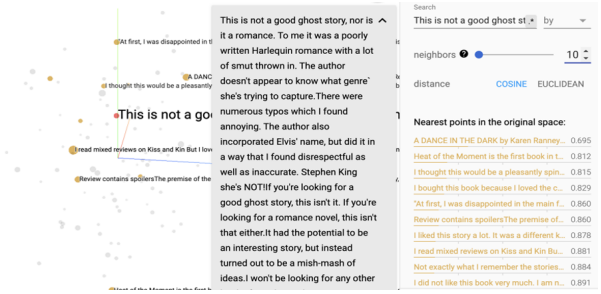


Figure 10 (1). Top: positive review text (2). Bottom: The neighbors of positive review text . On the right panel, it shows the top closest texts near to the given positive review text. The last text has the highest closest score. We can tell the texts close to each other have similar positive sentiments.

Review Text Visualization (Negative) :



Nearest neighbor Text Visualization (Negative):

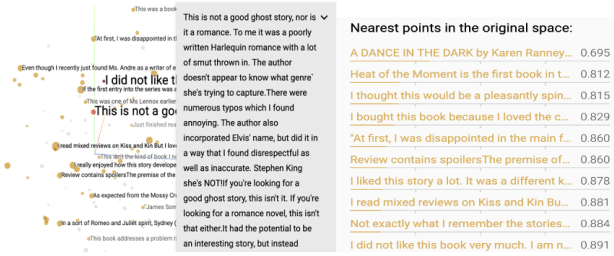


Figure 11 (1). Top: negative review text (2). Bottom: The neighbor of negative review text . On the right panel, it shows the top closest texts near to the given negative review text. The last text has the highest closest score. We can tell the texts close to each other have similar negative sentiments.

5. Conclusion

In this study, we investigated the applicability of fine-tuned bidirectional encoder representations from transformers (BERT) deep learning models in book review text datasets and evaluated them through multiclass classification. We compared the performance of BERT-based encoder models with the classical logistic regression and xgboost classifier. We verified and validated that the fine-tuned BERT model outperformed the other models in terms of classification. The better result can be achieved because BERT was trained on the huge amount and already encodes a lot of information about our human language. It also validates that BERT is one of the most powerful natural language processing (NLP) models available at present. In the future work, we will apply other fine-tuned models and multi-tasks models to different data sets for real-world applications. Furthermore, due to the large size of pre-trained model, we will also investigate the distilled model to reduce model size and keep the most important information on a small model.

Acknowledgements

We would like to thank our mentor Professor Wei Liu for the guidance of literature research, brainstorming, open source data research and model building and results discussion.

References

Abadi M., et al. (2015), TensorFlow: Large-scale machine learning on heterogeneous systems, Software available from tensorflow.org.

Addanki M., et al. (2019), "Classification of Book Reviews based on Sentiment Analysis: A Survey", 10.13140/RG.2.2.11576.29447

Amazon product data, (2018), https://jmcauley.ucsd.edu/data/amazon/

Charles, P., (n.d.) Transformers Interpret, https://github.com/cdpierse/transformers-interpret

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785–794). New York, NY, USA: ACM. https://doi.org/10.1145/2939672.2939785

Cox, D. R. (1958). The regression analysis of binary sequences. Journal of the Royal Statistical Society: Series B (Methodological), 20(2), 215–232

Devlin J., et al. (2019) BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", ArXiv, abs/1810.04805

Kingma, D. and Ba, J. (2015) Adam: A Method for Stochastic Optimization. Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015).

Lee, E., et al. (2022) Comparative Study of Multiclass Text Classification in Research Proposals Using Pre-trained Language Models. Appl. Sci. 2022, 12, 4522. https://doi.org/10.3390/app12094522

Pedregosa, F. et al (2011), "Scikit-learn: Machine Learning in python",Journal of Machine Learning Research, page 2825--2830

Transformers, https://huggingface.co/docs/transformers/index

Torrey, L. and Shavlik, J. (2010) Transfer Learning. In: Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques, IGI Global, Hershey, 242-264. https://doi.org/10.4018/978-1-60566-766-9.ch011

Vaswani, A, et al. (2017), "Attention is all you need", Advances in Neural Information Processing Systems, page 5998--6008.