Rationalizing Neural Predictions Reimplementation

Xu Han
xh852@nyu.edu

Jason Wang
jw7383@nyu.edu

Chloe Zheng
cz1300@nyu.edu

Introduction

Many recent advances in Natural Language Processing (NLP) use neural models that often come with a significant trade-off between accuracy and interpretability, which is a major downside in applications where transparency is necessary. The paper Rationalizing Neural Predictions (Lei et al., 2016) addresses this concern by proposing a method to increase interpretability for rationale analysis. The paper's approach combines two modular components, a generator and an encoder, to predict sentiment while also extracting short, continuous pieces of input text as justification for the prediction. As part of the paper, Lei et. al implements the model in Theano and evaluates the model on the BeerAdvocate dataset, which is composed of multi-aspect reviews. In this work, we recreate the architecture in PyTorch and use a subset of the same dataset to gauge reproducibility. As an extension, we adapt our model to a new dataset, Rotten Tomatoes audience reviews, to gauge flexibility and generalizability of this model. Interpretability is necessary in various domains, particularly those with high-stakes, that require rationales for predictions and decisions; this work aims to validate an existing rationale prediction method which may further contribute to work in transparency and interpretability in NLP.

Experimental Setup

Datasets

The **BeerAdvocate review dataset** consists of approximately 1.5 million multi-aspect beer reviews containing multiple sentences that describe the overall impression, appearance, smell, palate, and taste of beer accompanied by ratings from 1 to 5 stars for each aspect. The authors provide access to a sample of the original dataset of about 250,000 reviews. Each rating is normalized to a value between [0,1]. In addition, Lei provides 1,000 manually annotated rationales that, for each aspect, specify the phrases and sentences that contribute to that particular aspect's rating. [Fig. 1] shows an example review with the associated aspect ratings and highlighted rationales for each aspect.

"Poured into a snifter. ***Produces a small coffee head that reduces quickly. Black as night.*** *Pretty typical imp.* ***Roasted malts*** *hit on the nose. A* ***little sweet chocolate follows.*** *Big toasty character on the taste. In between i'm getting plenty of dark chocolate and some bitter espresso. It finishes with hop bitterness.* ***Nice smooth mouthfeel with perfect carbonation for the*** *style. Overall a nice stout i would love to have again, maybe with some age in it."*

Aspect	Sentiment
Appearance	4
Aroma	3.5
Palate	5

Figure 1: Example Review of the BeerAdvocate dataset

The **Rotten Tomatoes Audience Reviews** consists of 65,347 reviews contains single-aspect reviews on overall impression of a television show accompanied by the show title and an overall rating from 0 to 5. We chose a single aspect review dataset to experiment on whether or not rationale generation could extend well to a different type of review. We normalize ratings to a value between [0,1] and manually annotate 100 reviews for rationale data. We make sure to only annotate reviews that are at least 4 sentences long, and we choose between 25-75% of each review as the target rationale. We also annotate a mixture of positive, negative, and neutral reviews for class balance. An example is shown in [Fig. 2]. Pre-trained word embeddings are also used to train the model. We use provided embeddings from the authors for the BeerAdvocate model, and gloVe Wikipedia 2014 + Gigaword 5 embeddings with 6 billion tokens and 200 features.

"Maybe my favorite TV Series in the last two-plus years. ***Every performance is just perfect, the set dec, the cinematography.. everything is so intentional.*** *Yes, it's a slow burn, and each episode builds ever so slightly on the last but trust me the finale will have you in an hour-long anxiety attack. Perfect season of television."*

Rating	5

Figure 2: Example Review of the Rotten Tomatoes dataset

Model

The paper's architecture utilizes a custom Region-based Convolutional Neural Network (RCNN), which are used in practice to represent the generator and encoder. However, for our implementation, we use a simple CNN architecture to build the generator and encoder. The paper also implements the model in Theano - we chose to implement the architecture with PyTorch as it is a more accessible and flexible library that may be more useful for future adaptations. The paper's approach, shown in [Fig. 3], combines two modular components, a generator and an encoder, to extract short, continuous pieces of input text as justification for the prediction. The generator is a probability distribution over text fragments as candidate rationales, which are then passed through the encoder for predictions.

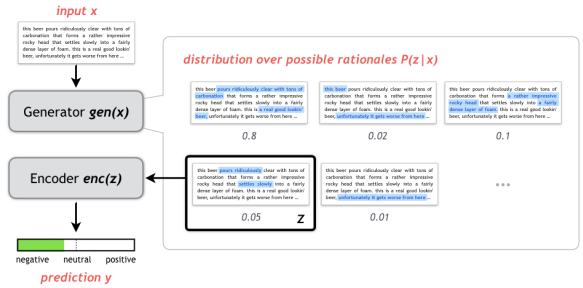

Figure 3: Model Architecture

$$\mathcal{L}(\mathbf{z}, \mathbf{x}, \mathbf{y}) = \|\mathbf{enc}(\mathbf{z}, \mathbf{x}) - \mathbf{y}\|_2^2$$
$$\Omega(\mathbf{z}) = \lambda_1 \|\mathbf{z}\| + \lambda_2 \sum_t |\mathbf{z}_t - \mathbf{z}_{t-1}|$$
$$\text{cost}(\mathbf{z}, \mathbf{x}, \mathbf{y}) = \mathcal{L}(\mathbf{z}, \mathbf{x}, \mathbf{y}) + \Omega(\mathbf{z})$$
$$\min_{\theta_e, \theta_g} \sum_{(\mathbf{x}, \mathbf{y}) \in D} \mathbb{E}_{\mathbf{z} \sim \mathbf{gen}(\mathbf{x})} [\text{cost}(\mathbf{z}, \mathbf{x}, \mathbf{y})]$$

Figure 4: Loss Function

The loss function [Fig. 4] uses mean squared error and regularization that motivates shorter, continuous rationales. So given training instances (x, y) where $x = \{x_t\}_{t=1}^{l}$ is the input text sequence of length l, $y \in [0, 1]_m$ is the target m-dimensional sentiment vector, and binary variables $\{z_1, ..., z_l\}$ where each $z_t \in [0, 1]$ indicates whether word x_t is selected or not, $cost(z, x, y) = \|enc(z, x) - y\|_2^2 + \Omega(z)$. Minimizing the expected cost is computationally infeasible, so the gradient is approximated, and the cost is minimized using a doubly stochastic gradient.

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2$$
$$\textit{Precision} = \frac{TP}{TP + FP}$$

Figure 5: Metrics

To evaluate the model's sentiment and rationale predictions, we use mean squared error (MSE) and precision [Fig. 5] respectively. The labels for both datasets are converted to continuous values between [0,1] such that mean squared error is applied to compare the predicted sentiment to the actual sentiment label. To evaluate the extracted rationales, we use precision to see if the model returns more relevant rationales than irrelevant ones. Recall would be less useful for this task because recall measures if the model returns most of the relevant rationales. However this could vary wildly since the annotations are manually determined by human judgment. In addition, the original paper only reports precision, which is why we chose to only evaluate precision.

Experiments

To train and evaluate our model, we use a $\frac{4}{6}$, $\frac{1}{6}$ and $\frac{1}{6}$ split of our BeerAdvocate Dataset. Note that we have two test sets, the test set from the train-valid-test split of our dataset and a rationale set for reviews that have manual annotations. To tune our model, we vary the regularization hyperparameters of selection lambda (λ_1) and continuity lambda (λ_2). A higher λ_1 encourages shorter sequences and a higher λ_2 encourages more continuous sequences. For other hyperparameters such as learning rate, batch size, hidden dimensions, and number of layers, we set them to the following values for all iterations: Learning rate = 0.001, Batch size = 128, Hidden Dimensions = 200,

and Number of Layers = 2, which are the same hyperparameters used in the paper (excluding learning rate which is not specified).

We test the same set of selection lambdas and continuity lambdas that the original tune on, [0.0002, 0.0004], [0.0003, 0.0006] and [0.0004, 0.0008], with 20 epochs on the appearance aspect. The precision scores are reported in [Table 1.] We select $\lambda_1 = 0.0004$ and $\lambda_2 = 0.0008$, the pair with the highest precision, for further experiments on aspects aroma and palate, completing the three aspects that the original paper evaluates. Because the loss converges quickly, the aroma and palate aspects are trained with only 5 epochs.

Parameters		Scores	
selection	**continuity**	**precision**	**mse**
0.0002	0.0004	0.6966	0.0628
0.0003	0.0006	0.7637	0.0405
0.0004	0.0008	0.8021	0.0257

Table 1: Hyperparameter Tuning $Appearance$

For the Rotten tomatoes extension, we use a 60-20-20 split and train the model with the same hyperparameters as the BeerAdvocate dataset. Precision is calculated on a small test set of 100 reviews with manual annotations that we created ourselves.

Results

We trained the BeerAdvocate on three of the provided aspect datasets. [Fig. 6] shows the convergence of losses for both train and valid for all three datasets - we trained on 20, 5, and 5 epochs for $Appearance$, $Aroma$ and $Palate$ respectively. [Fig 7.] and [Fig 8.] visualizes the final metrics, MSE and Precision, of our models with respect to the original scores reported by the paper. Note that the paper only reports a single MSE and it is unclear which aspect was evaluated.

Metric	**Aspects**	**Original**	**Replicate**
MSE	Appearance	0.0087	0.0248
	Smell	0.0087	0.0351
	Palate	0.0087	0.0345
Precision	Appearance	0.948	0.796
	Smell	0.938	0.498
	Palate	0.793	0.462

Table 2: BeerAdvocate Metrics

Metric	**Score**
MSE	0.0685
Precision	0.5362

Table 3: Rotten Tomatoes Metrics

[Table 2.] reports the exact metrics reported. Finally, [Table 3.] reports the metrics for the Rotten Tomatoes dataset.

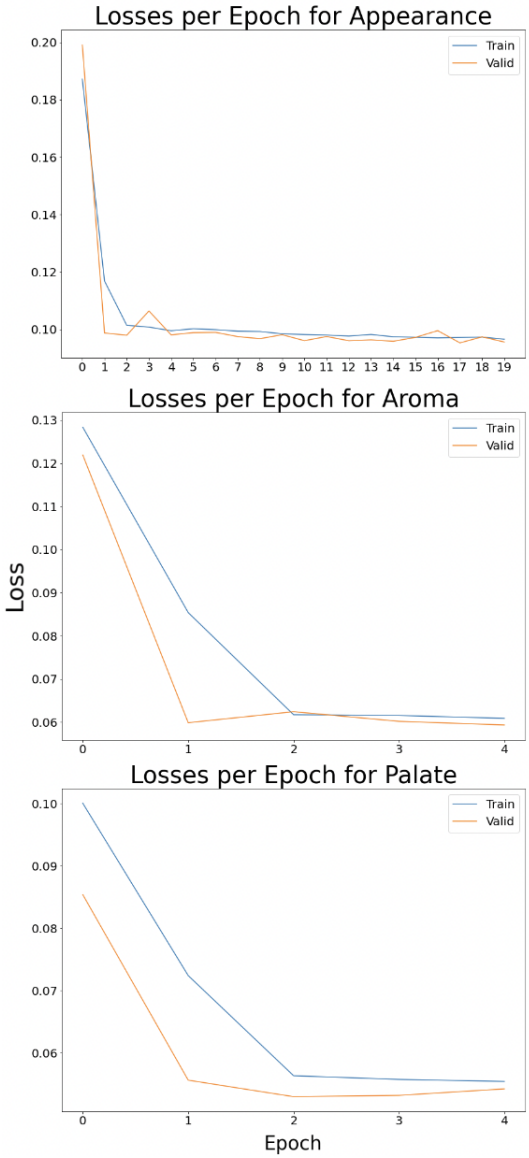

Figure 6: Loss Graphs for $Appearance$, $Aroma$ and $Palate$ datasets

Discussion

Our PyTorch implementation of the Generator-Encoder model can generate rationales for sentiment prediction for the BeerAdvocate dataset. However, it is unable to replicate the exact metrics reported by the original paper for sentiment prediction and rationale generation. Based on our tuned

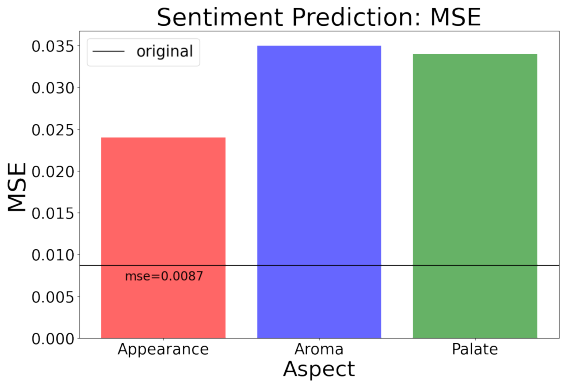

Figure 7: BeerAdvocate results for sentiment prediction

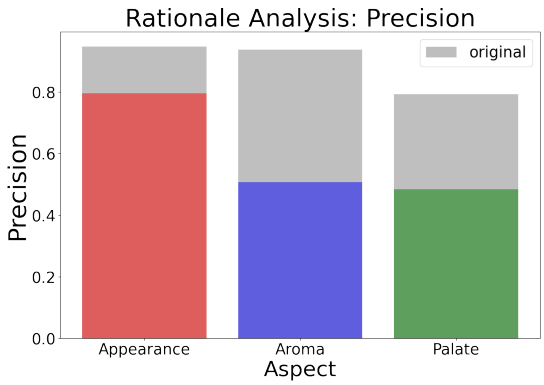

Figure 8: BeerAdvocate results for rationale analysis

results, rationale generation is very sensitive to hyperparameters; the optimal hyperparameters for the "appearance" dataset are not generalizable for the other datasets, "aroma" and "palate". In addition, poor results may be due to the lack of the full training dataset and differences between the Theano and PyTorch implementation, especially since the original paper uses a RCNN model while ours uses a simpler CNN model. On the other hand, we successfully adapt the model on the Rotten Tomatoes dataset and generate rationales that are intuitively reasonable; however, similar to the BeerAdvocate model, we need a much larger dataset to more accurately evaluate the Rotten Tomatoes model.

Although the original paper was able to achieve high precision, additional metrics could be identified to better measure the quality of rationale prediction (accuracy, f1-score). Moreover, the BeerAdvocate dataset has no specifications about how rationale annotations are generated. It is important that annotators identify what makes a quality rationale annotation. These specifications should identify the target audience and what they deem an appropriate and satisfactory rationale, which may differ across language, culture, race, ethnicity, age, and gender. This also poses a challenge when comparing rationale prediction models, such as the BeerAdvocate model to the Rotten Tomatoes model, particularly if the target audiences' rationale expectations greatly differ.

Conclusion

Our team was able to implement the paper's Theano model in PyTorch with both the original dataset and a new dataset. We did not achieve high performance, but the PyTorch library is accessible and our model may be further fine-tuned with additional data and metrics. This model, proposed by Tao Lei, Regina Barzilay and Tommi Jaakkola, has the potential to improve interpretability, which may be useful in many domains where stakeholders find rationales necessary. Machine learning systems have proven to achieve high performance in predictive analysis; however in domains where predictions may have a profound negative or positive affect on society and individuals, interpretability should be a priority. Sentiment Analysis on beer and entertainment reviews may be a low-stake use case for interpretability, but the implemented rationale prediction model has the potential to be adapted for other domains (e.g. insurance, healthcare, employment) where interpretability is vital.

Author contribution statement

The project was broken down into three main tasks:

1. Pytorch implementation
2. Debugging and Hypertuning
3. Model adaptation to Rotten Tomatoes dataset

Jason Wang led task (1), Xu Han led task (2), and Chloe Zheng led task (3). Every team member assisted on all three tasks when necessary. Jason assisted both Xu and Chloe to debug both models and generated visualizations for loss, Xu helped Chloe annotate and load the Rotten Tomatoes dataset and ran iterations for both models, and Chloe helped Jason and Xu run iterations for both models and generated visualizations for metrics. All three authors contributed equally in creating a poster summarization of the work, as well as writing the final paper.

References

Tao Lei, Regina Barzilay, Tommi Jaakkola. 2016. Rationalizing Neural Predictions. arXiv:1606.04155

Andrej Karpathy, Justin Johnson, and Fei-Fei Li. 2015. Visualizing and understanding recurrent networks. arXiv preprint arXiv:1506.02078.

B Kim, JA Shah, and F Doshi-Velez. 2015. Mind the gap: A generative approach to interpretable feature selection and extraction. In Advances in Neural Information Processing Systems

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2015. Molding cnns for text: non-linear, non-consecutive convolutions. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP).

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should i trust you?": Explaining the predictions of any classifier. In ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD).

Omar Zaidan, Jason Eisner, and Christine D. Piatko. 2007. Using "annotator rationales" to improve machine learning for text categorization. In Proceedings of Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, pages 260–267.

Ye Zhang, Iain James Marshall, and Byron C. Wallace. 2016. Rationale-augmented convolutional neural networks for text classification. CoRR, abs/1605.04469.

Michiel Hermans and Benjamin Schrauwen. 2013. Training and analysing deep recurrent neural networks. In Advances in Neural Information Processing Systems, pages 190–198.

Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, Louis-Philippe Morency. 2017. Tensor Fusion Network for Multimodal Sentiment Analysis. arXiv:1707.07250

M.D. Devika, C. Sunitha, Amal Ganesh. 2016. Sentiment Analysis: A Comparative Study on Different Approaches. In Procedia Computer Science, Volume 87, Pages 44-49, ISSN 1877-0509. https://doi.org/10.1016/j.procs.2016.05.124.

Shi, Tian and Rakesh, Vineeth and Wang, Suhang and Reddy, Chandan K. 2019. Document-Level Multi-Aspect Sentiment Classification for Online Reviews of Medical Experts. In Association for Computing Machinery.

L. Stappen, A. Baird, L. Schumann and S. Bjorn. 2021. The Multimodal Sentiment Analysis in Car Reviews (MuSe-CaR) Dataset: Collection, Insights and Improvements. In IEEE Transactions on Affective Computing. doi: 10.1109/TAFFC.2021.3097002.

Finale Doshi-Velez, Been Kim. 2017. Towards A Rigorous Science of Interpretable Machine Learning. arXiv:1702.08608.