Learning to Read Like a Human: Reinforcement Learning for Natural Language Processing

Jeremy Curuksu
People Experience and Technology
curukj@amazon.com

Abstract

Making complex decisions when processing natural language contents, such as a conversation transcribed in real time, often involves a balancing act between different sources of feedback. Reinforcement learning is designed to learn optimal tradeoffs between different sources of potentially delayed feedback, in sequence and in real time, by trial-and-error. Instead of replacing a supervised system by a reinforcement system, we combine the two together. We demonstrate an improvement of the combined system compared to baseline supervised learning. The method can be used to fine tune any pre-trained supervised learning system. When reading Amazon product reviews, the reinforcement learning agent is able to learn a nuanced meaning for each sentence relative to the baseline, leading to an improvement of 10% in predicting the final star rating across 1,000 reviews. The improvement happens at sentence level, thus the RL agent learns to track customer satisfaction *as the narrative progresses* in each review, which we monitor by looking at the specific decisions made by the agent in each review.

1 Introduction

Reading, processing and making decisions based on contents written in English or other natural languages (e.g., transcribed conversation) is challenging because the information is sequential in nature and the author's intents can rapidly evolve in the course of the conversation. Take sentiment analysis as an example. Typical natural language supervised learning (SL) models such as Word2Vec [1] trained at sentence level cannot track the evolving emotional state in a conversation because it ignores other sentences in the conversation. If such models are trained on entire blocks of conversations, such as Doc2Vec or Objet2Vec [2], they are not trained, by definition, to track specific individual sentences. So how could a computer model learn an evolving mental and emotional state during a conversation? Reinforcement Learning (RL) has been successfully applied in recent years for autonomous and real-time decision making in video games and strategic board games [3], robotics [4], vehicle driving [5], algorithmic trading [6], and many others. An RL agent can learn to make decisions in sequence to reach a predetermined long-term goal (e.g., maximize profit) while tracking a measure of this objective function in real time while interacting with its environment and learning from the experience accumulated so far [7]. Since text contents (e.g., transcribed conversation, product review) written in natural languages are sequential in nature, RL can help process text contents to better categorize and understand intents.

In this paper, we describe how RL can be combined with existing Natural Language Processing (NLP) machine learning methods (both supervised and unsupervised) to better understand text contents. We adopt the view that RL, instead of being used as *substitute* to supervised learning methods, is best leveraged to *improve* supervised learning methods. We demonstrate that RL leads

to better performance when it assists a SL method in NLP context, compared to when using this same SL method without RL. The generality of the proposed method is immediate because it will always be at least as good as the underlying SL model, and in addition it offers the possibility to fine tune the agent behavior to mitigate additional goals even under imperfect supervision. The only drawbacks are the added conceptual and algorithmic complexity, which we describe in this paper, and potentially longer model training time.

The proposed method is applied to assess customer satisfaction in real time, using Amazon product reviews as training data. The RL agent learns to fine tune a supervised learning NLP sentiment analyzer to better track the evolving sentiment of the customer as the narrative progresses. Results indicate the RL agent learns to optimally mitigate feedback provided by the pre-trained SL model and feedback provided by actual customer ratings. This reinforcement learning process improves the performance of the baseline SL model at rating each review by over 10% in most test samples. Most importantly, the improved rating happens at sentence level, thus enabling to better track customer satisfaction as it evolves in a product review or conversation.

2 Reinforcement learning for natural language processing

In a RL process, a goal is defined for an *agent* which makes decisions in an *environment*. This goal is translated into a mathematical formula called a reward function, which rewards or penalizes the agent when it takes an action, helping the agent reach the predefined goal. At each step t, the agent receives a representation of the environment state s_t, takes an action a_t which brings the agent to a new state s_{t+1}, and receives a numerical reward r_{t+1} for having taken a_t in s_t, as shown in Fig. 1.

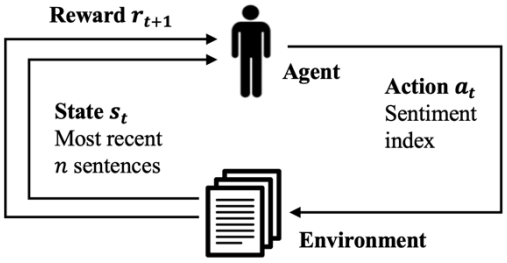

Figure 1: Components of the reinforcement learning decision process

A sequence of actions is called a policy and noted π. The purpose of reinforcement learning is to identify an optimal policy π^* to reach the predefined goal. The NLP method presented in this paper relies on the ability of a RL framework to learn delayed rewards and also to learn an optimal tradeoff between different sources of reward feedback i.e., different sources of rewards and penalties reflecting complex and nuanced goals. This is possible because in a RL framework the objective function used to identify optimal policies can be defined over an entire sequence of past and future states and actions, as long as sufficiently many actions and states are explored. The sequences of states and actions can be sampled by interacting with and/or simulating the real world.

The RL objective function is typically defined as the expected value of all accumulated rewards in in a policy π, from a given state s_t in a given sequence up to the final state s_T of this sequence:

$$v_\pi(s_t) = \mathbb{E}_\pi\left(\sum_k^T r_{t+1+k} \mid s_t\right)$$

An RL agent selects the sequence of actions a_t that correspond to the highest $v_\pi(s_t)$. This defines the optimal policy π^*. The *true* value of $v_\pi(s_t)$ can be learned and approximated in real time as the agent explores the environment by recursively estimating $v_\pi(s_t)$ as a function of current estimate

of $v_\pi(s_{t+1})$ (this method is called temporal difference, see [7] for details). The index t represents a moment in time, or more generally a position is a sequence of states.

In this paper, the goal of the agent is to process text contents and track the author sentiment as it evolves in real time. A state s_t will be defined as a sentence in Amazon product reviews (details in section 3.2), thus the environment in Fig. 1 is made of Amazon reviews and the index t indexes a particular sentence in a given review. We use Amazon product reviews to test our method because Amazon reviews are readily available and systematically annotated with customer ratings, providing a reliable ground truth for measuring the performance of the agent. But the method could readily be applied to any other text contents, such as a conversation transcribed in real time.

An unsupervised learning NLP encoder can transform each utterance into an embedding numerical vector [1, 2] to define a state s_t in the environment, as described in section 3.2. Upon observing such state, the agent takes a single action i.e., it selects a sentiment index between 0 and 5.

To assess customer satisfaction as it evolves during a conversation (or product review), the agent needs to reconcile the intrinsic sentiment of each sentence with the more nuanced meaning acquired given other sentences that accumulate in sequence as the conversation proceeds. To this aim, the RL reward function detailed in section 3.4 contains a term to help emulate a sentiment analyzer trained by supervised learning, and another term to fine tune the sentiment index of each sentence based on the specific sequential context and final star rating in a given review.

To demonstrate the impact of a reading agent trained by RL compared to a sentiment analyzer trained only by SL, we measure separately the learning made by each component in the RL reward function (section 4.1, Fig. 2) to confirm that the RL agent effectively learns from both components. Then, we compare the tracked sentiment index at the end of the review relative to the sentiment index predicted by the SL model (section 4.2, Fig. 3), to confirm that RL rates attributed to each sentence s_t lead to similar or improved prediction of customer satisfaction in s_T. Note that the final customer state of satisfaction in each review is the only available ground truth. The actual, evolving state tracked by RL (Fig. 4) is not known and can only be assessed in the final state s_T.

3 Methodology: Components of the RL reading process

3.1 Reinforcement Learning Goals

The goal of the agent is to process text contents and track the author sentiment as it evolves in real time. In the case of Amazon product reviews, the ground truth for the final state s_T is available (star rating given by the customer) so the goal of the agent is also to end each review with this star rating.

In contrast to supervised learning, with reinforcement learning it is possible to mitigate multiple learning goals simultaneously, by adding multiple terms in the reward function, each addressing a particular type of behavior [7]. To track an evolving sentiment index, a sentiment analyzer trained by supervised learning would rate each sentence or section independently and add up their scores. This represents one useful feedback for the RL agent to learn from (section 3.4.1). But often, even just one negative sentence in a review may contribute more to the overall product rating than many other positive sentences. Inversely, when a customer makes peculiar positive comments such as *the price is right*, satisfaction often remain high regardless of other, more negative comments in the review. An RL agent explicitly models entire sequences of actions (policies) to approximate the value of any given action, thus an RL agent can learn from the specific customer ratings of product reviews to fine tune the sentiment index attributed to each sentence, recursively as described in section 2, based on specific sequential contexts in the reviews. This represents a second source of feedback in the reward function (section 3.4.2).

Thus, RL goals are to reconcile (1) the baseline sentiment of each sentence provided by a pre-trained SL model with (2) a more nuanced meaning acquired given other sentences that accumulate in sequence and given the rate actually provided by the customer in each review.

3.2 State Space

To define the RL environment, we used Amazon Customer Reviews [8], which is a public dataset

released by Amazon and constructed to represent a sample of customer evaluations and opinions, variation in the perception of a product across geographical regions, and promotional intent or bias in reviews. In this paper, we used the Amazon furniture review dataset and selected a random sample of 1000 reviews that contains at least 50 words.

Each sentence in the Amazon furniture review dataset represents an observation in the RL environment. Any NLP encoder can be used to represent a sentence as a numerical embedding vector, which in turn defines a state $\boldsymbol{s_t}$. The Amazon SageMaker BlazingText algorithm [1] was used to transform each sentence into a numerical vector. The BlazingText Word2Vec unsupervised learning model was pre-trained on the entire corpus of Amazon furniture reviews to represent each word as a numerical embedding vector. The embedding vectors computed in real time for each word in a sentence were then averaged to define a sentence-level embedding vector. We used an embedding space with 100 dimensions thus each sentence was encoded by a vector of 100 dimensions after averaging word vectors over all words in a given sentence.

The Word2Vec encoding implemented with SageMaker represents a simplified version of more advanced NLP models such as Object2Vec [2] which leverages CNN and RNN, and BERT [9] which leverages Attention mechanisms. To our knowledge neither have been combined with RL in the manner described in this paper and as mentioned above, our method does not serve as substitute for supervised learning: it can be used *with* any supervised learning model and aims to fine tune its performance by adding additional ground truth feedback in the RL reward function.

3.3 Action Space

For each sentence, the RL agent estimates the effective contribution of this sentence to the star rating of the entire product review by attributing a rate between 0 and 5. The RL action space is thus an integer between 0 and 5. At the end of a review, the agent takes the average of the rates given to each sentence observed in the review to define the overall star rating of the review.

3.4 Reward Function

The reward function aims at rewarding or penalizing the agent when it takes an action, and can contain multiple components to help the agent mitigate multiple learning goals. It can also be 0 for some (or all) states. Each goal is encoded by a specific component in the reward function:

$$r_t = \lambda_1 r_t^1 + \lambda_2 r_t^2$$

where λ_1, λ_2 are weights (hyperparameters) which can be fine-tuned to boost the impact of each term relative to one another.

As described in section 3.1, the goal of the agent is to identify a sentiment index for each sentence that both reflects an ongoing intrinsic sentiment evolving in the review, and also an effective contribution to the final star rating given other sentences in the review. To this aim, we defined a reward function that contributed a baseline feedback on the sentiment of each sentence based on predictions made by a SL sentiment analyzer (r_t^1), and a ground truth feedback on the rating available in the final state $\boldsymbol{s_T}$ of each review (r_t^2).

3.4.1 Sentence-level reward component ($\boldsymbol{r_t^1}$)

A shaping reward component is often used in RL by providing feedback at every state to help guide exploration toward regions of the state-action space that matter the most. This frequent, baseline feedback can help increase the speed and convergence of learning via transfer learning from a pre-trained sentiment analyzer. To define a reward for every sentence in Amazon reviews, the Amazon SageMaker Blazingtext classification algorithm, an NLP SL model, was trained on the UCLI dataset of Amazon review sentences labelled for sentiment polarity [10]. During RL simulations, the agent

rates each sentence in the review between 0 and 5 and receives a reward based on the difference between this rate and the probability p_t that the sentence has positive sentiment according to the pre-trained SL model. The Blazingtext algorithm outputs the probability p_t that the sentiment polarity is 1 and so was used to compute the reward component r_t^1 as shown below:

$$r_t^1 = 5 - |5p_t - a_t|$$

The component r_t^1 is thus a function of the difference between a_t and the customer satisfaction predicted by a pre-trained SL sentiment analyzer. It is available for every sentence in every review.

3.4.2 Final reward component (r_t^2)

The star rating provided by the customer who wrote each sentence in a review is an actual, ground truth feedback which measures the satisfaction of this particular customer at the end of the review i.e., in the final state s_T. But this feedback is only available once at the end of the review. To learn the customer's evolving state of satisfaction at each sentence in the review, the agent computes the average of the rates a_T attributed to each sentence and receives a final reward r_t^2 based on the difference between this average rate and the star rating r_{review} actually observed:

$$r_t^2 = 5 - \left| r_{review} - \frac{1}{T}\sum_{k=1}^{T} a_k \right| \quad \text{if} \quad t = T$$

$$r_t^2 = 0 \quad \text{if} \quad t \neq T$$

The component r_t^2 is thus a function of the difference between the policy (sequence of actions a_t) followed in a given review and the star rating provided by the customer who wrote the review. It is available only once per review.

Thus, the agent is guided by the pre-trained SL sentiment analyzer when rating a given sentence, but also fines tune this rate based on its effective contribution to the star rating of the entire review. This contribution is learned by reinforcement, recursively from the sequence of other sentences in each review by learning and approximating $v_\pi(s_t)$, as described in section 2.

4 Analysis of RL-based natural language processing

All RL simulations were carried out using actor-critic with the proximal policy optimization (PPO) algorithm available in Amazon SageMaker RL Intel Coach v11.0, Python 3.5 and TensorFlow, for a total number of 1,000 product reviews containing at least 50 words, which represent approximately 5,000 sentences. Each simulation was repeated 40 times (40,000 reviews) for a total of 200,000 steps, which was sufficient for the agent to explore and converge toward an optimal behavior as reported in section 4.1. All simulations ran on 36 CPUs using C5 9XL Amazon EC2 instances. An efficient exploration of the action space was ensured by applying an ε-greedy exploration schedule that linearly switched ε from 10% to 1% over the first 50,000 steps.

4.1 Analysis of RL convergence when processing Amazon reviews

To assess the learning performance of the RL agent for processing Amazon reviews, the total accumulated reward was computed for each episode. The accumulated reward in a given episode measures how good the policy followed in this episode was. Thus, when compared between episodes (Fig. 2) it measures the relative value of the policies learned to optimally mitigate feedback provided by the pre-trained SL model and by customer ratings. We measure separately the learning made by each component in the RL reward function (Fig. 2, left panels) as well as the total accumulated reward (Fig. 2, right panel) to measure how much the RL agent learns from each component.

Fig. 2 demonstrates that the RL agent effectively learns from both components. For each reward component, the RL agent transitions from a phase of random exploration in the first 10 episodes, to a phase (episode 10 and beyond) where a plateau is observed: the total accumulated reward has reached convergence and revolves around a higher and relatively stable value. So the agent explored then identified policies that mitigate feedback provided by the pre-trained SL model and feedback provided by actual customer ratings.

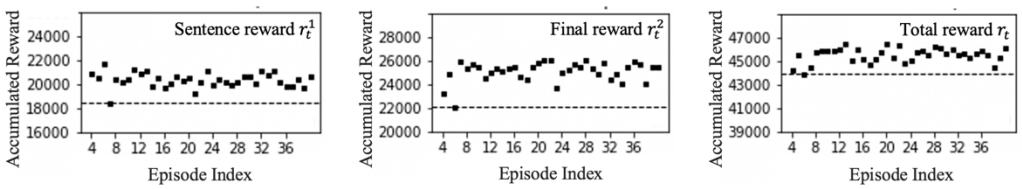

Figure 2: Accumulated rewards across sentences in corpus of 1,000 Amazon reviews

These results indicate that the RL agent has learned to infer customer satisfaction and converged toward a stable behavior with reproducible policies to track customer satisfaction in the Amazon product reviews.

4.2 Analysis of RL performance when processing Amazon reviews

To further demonstrate the RL agent's ability to track the evolving state of customer satisfaction as the narrative progresses in product reviews, we benchmarked its performance at predicting the star rating provided by the customer compared to the baseline pre-trained SL model. In Fig. 3, an orange circle is the mean absolute error obtained across the 1,000 Amazon reviews between the average of the rates attributed to each sentence by the RL agent and the star rating actually observed in each review. The RL simulation was repeated 40 times and compared to the mean absolute error obtained when using the baseline SL model trained at sentence level (green line).

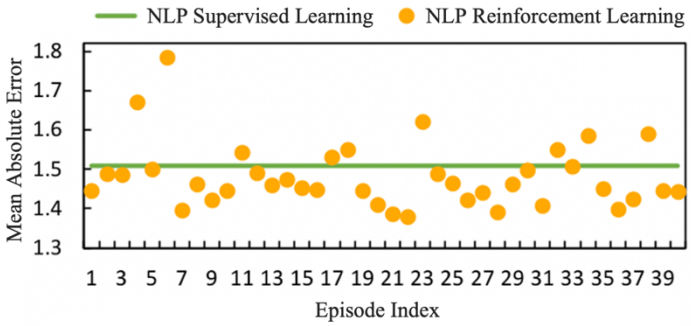

Figure 3: Mean absolute error between star rating derived from tracking evolving state of satisfaction and ground truth in 1,000 product reviews

Fig. 3 confirms the RL agent learned reproducible policies to track customer satisfaction as it evolves in a product review. The mean absolute error across all reviews is smaller in 78% (31 out of 40) of trials when using the RL agent compared to when using the pre-trained SL model. In other words, in 78% of RL simulations, the agent learned to rate each sentence more consistently with the star rating actually observed in each review. Once converged (episode 10 and beyond), the error is frequently below 1.4 while it is 1.51 when using the baseline SL model, representing approximately 10% of improvement in performance at rating product reviews.

Most importantly, the improved rating happens at sentence level i.e., the RL agent effectively rates each sentence based on its effective contribution to the rate of the entire review by learning a more nuanced meaning for each sentence from combining the feedback provided by the SL model trained at sentence level and the feedback provided by ground truth star ratings. Thus, the RL agent learned to better track customer satisfaction as it evolved in the product reviews.

An example of Amazon review is provided in Fig. 4, where the star rating provided by the customer for this review was 4 stars, and the state of customer satisfaction predicted by the RL agent after reading each sentence in the review was 3.5 stars. The two columns on the right-hand side of Fig. 4 indicate the effective rate assigned to each sentence by the RL agent, and the corresponding evolving state of customer satisfaction along this sequence (average of accumulated rates from top to bottom) i.e., as the agent read the review in real time.

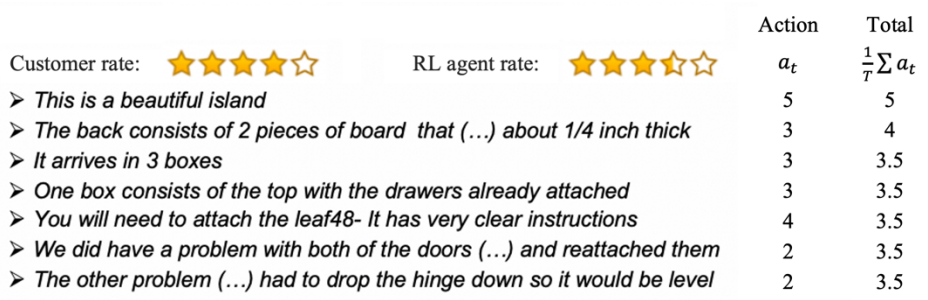

Figure 4: Example of an Amazon product review and decisions made by the RL agent

By tracking the rates provided by the RL agent for each sentence, it is easy to interpret the agent behavior. The customer started her review with high satisfaction (*This is a beautiful island*), then made mostly neutral (*It arrived in 3 boxes*) or positive comments (*It has very clear instructions*), and finished with two relatively negative comments. Each of these steps are quantitatively and accurately measured by the RL agent, in such a way that by the end of the review, the average of accumulated rates is very close to the rate actually provided by the customer

5 Conclusion

Reinforcement learning was combined with supervised and unsupervised natural language processing machine learning methods to better understand text contents, and more specifically to better track customer satisfaction in real time. An RL agent can learn delayed rewards and optimal tradeoffs between different sources of feedback. When reading Amazon product reviews, it was able to learn a nuanced meaning for each sentence relative to a baseline SL sentiment analyzer leading to an improvement of 10% in predicting the final star rating across 1,000 reviews.

The improved rating happens at sentence level, thus the RL agent learned to better track customer satisfaction as the narrative progresses in each review relative to the baseline SL model. This can be monitored by visualizing the specific decisions made by the agent in the reviews.

The RL reading agent developed in this paper can be used to fine tune the performance of *any* pre-trained supervised learning NLP model, using *any* sources of feedback as long as they can be introduced as numerical components in the RL reward function e.g., to better track customer satisfaction as it evolves in a conversation transcribed in real time.

Further research is needed to assess the performance of the RL agent with more complex corpora of text contents and more diverse sources of feedback, such as transcribed conversations in call centers where the ground truth is not as readily available as when using Amazon product reviews.

The current method was successfully applied with an AWS customer in financial services to help

audit compliance levels of items in legal claims. RL was used to fine tune SL models pre-trained on large corpora of claims with different compliance levels, using additional ground-truth feedback on peculiar fraudulent patterns e.g., a same author who submits an identical item twice in two different claims using a different description. By applying a downstream unsupervised clustering analysis on the set of fraudulent items detected by the RL agent, it was possible to identify entirely new *patterns* of fraudulent behaviors not previously detected in past audits.

Our results demonstrate that RL can be combined with SL models to improve the ability to make autonomous decisions in real time as the narrative progresses in text contents.

Customer Problem Statement

Customers regularly make decisions based on reading contents written in English or other natural languages. With the broad adoption of machine learning, most organizations are trying to automate such decision making, even more so given the possibility to transcribe conversations (e.g., phone calls) happening in real time. But typical supervised learning natural language processing models are challenged by the complex and sequential nature of information in text contents, where key feedback can be sparse and delayed. Reinforcement learning is designed to learn optimal tradeoffs between different sources of potentially delayed feedback, in sequence and in real time, by trial-and-error. Moreover, RL can be combined with any supervised learning system by using this system as main source of feedback, and fine-tuned using other sources of custom feedback. The method presented in this paper is applied to assess customer satisfaction in Amazon product reviews because it makes it easy to demonstrate its impact, and because the data is readily and publicly available. The generalization to other customer problems is immediate, such as tracking the customer sentiment during a phone call transcribed in real time to efficiently manage and redirect requests in customer call centers. The proposed method was also successfully applied with one of our AWS customers in financial services to help audit compliance levels of items in legal claims. RL was used to fine tune SL models pre-trained on large corpora of claims with different compliance levels, using additional ground-truth feedback on peculiar fraudulent patterns e.g., a same author who submits an identical item twice in two different claims using a different description. For this customer problem, the RL reward function included a baseline feedback (r_t^1) on the compliance of each item based on predictions made by a pretrained SL model, and a ground-truth feedback (r_t^2) on the fraudulent pattern available only for some (but not all) fraudulent items. By applying a downstream unsupervised clustering analysis on the set of fraudulent items detected by the RL agent, it was then possible to identify entirely new *patterns* of fraudulent behaviors not previously audited by our AWS customer. More generally, the proposed RL method can be combined with any SL model to improve the ability to make autonomous decisions in real time in NLP context.

References

[1] Gupta S, Khare V (2017) Blazingtext: Scaling and accelerating word2vec using multiple GPUs. Proceedings of the Machine Learning on HPC Environments: 1-5

[2] Ping D, Xiang B et al. (2018) Introduction to Amazon SageMaker Object2Vec. AWS Machine Learning

[3] Silver D, Hubert T, Schrittwieser J et al. (2018) A general reinforcement learning algorithm that masters Chess, Shogi and Go through self-play. Science 362: 1140–1144

[4] Ibarz J, Tan J, Finn C et al. (2021) How to Train Your Robot with Deep Reinforcement Learning: Lessons We've Learned. CoRR abs/2102.02915

[5] Sallab AE, Abdou M, Perot E, Yogamani S (2017) Deep reinforcement learning framework for autonomous driving. Electronic Imaging 19: 70-76

[6] Wu X, Chen H, Wang J et al. (2020) Adaptive stock trading strategies with deep reinforcement learning methods. Information Sciences 538: 142-158

[7] Sutton RS, Barto AG (2018) Reinforcement learning: An introduction. MIT press: 47-141

[8] Amazon Customer Reviews Dataset (2015) available at https://s3.amazonaws.com/amazon-reviews-pds/readme.html

[9] Devlin J, Chang MW et al. (2018) BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805

[10] Ni J, Li J et al. (2019) Justifying recommendations using distantly-labeled reviews and fine-grained aspects. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing: 188-197