This CVPR workshop paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Prioritised Moderation for Online Advertising

Phanideep Gampa [*]
Amazon
India
phanide@amazon.com

Akash Anil Valsangkar [*]
Amazon
India
valsangk@amazon.com

Shailesh Choubey
Amazon
India
choubey@amazon.com

Pooja A.
Amazon
India
apooja@amazon.com

Abstract

Online advertisement industry aims to build a preference for a product over its competitors by making consumers aware of the product at internet scale. However, the ads that violate the applicable laws and location specific regulations can have serious business impact with legal implications. At the same time, customers are at risk of getting exposed to egregious ads resulting in a bad user experience. Due to the limited and costly human bandwidth, moderating ads at the industry scale is a challenging task. Typically at Amazon Advertising, we deal with ad moderation workflows where the ad distributions are skewed by non defective ads. It is desirable to increase the review time that the human moderators spend on moderating genuine defective ads. Hence prioritisation of deemed defective ads for human moderation is crucial for the effective utilisation of human bandwidth in the ad moderation workflow. To incorporate the business knowledge and to better deal with the possible overlaps between the policies, we formulate this as a policy gradient ranking algorithm with custom scalar rewards. Our extensive experiments demonstrate that these techniques show a substantial gain in number of defective ads caught against various tabular classification algorithms, resulting in effective utilisation of human moderation bandwidth.

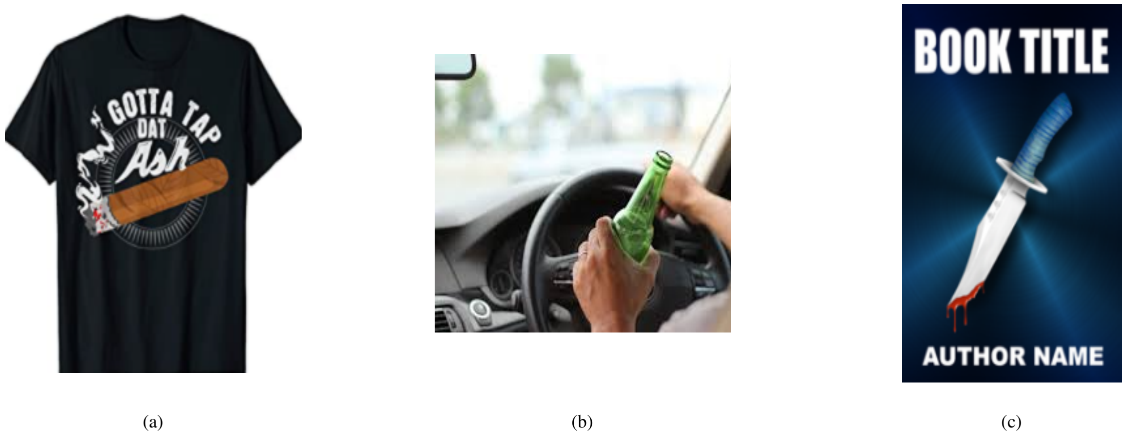
1. Introduction

The number of internet users are increasing at a rapid scale along with the information they interact with in their daily lives in various social sites and e-commerce websites. At the same time, the risk of users interacting with harmful, illegal, controversial, egregious contents is going to increase if unmoderated (refer Figure 1). Bad user experiences are going to affect the reputation and business of the content websites. This would mean that there is a requirement for the content websites to maintain an efficient system to moderate or filter out harmful contents – social media posts, ads, blogs etc – so that the end users do not feel unsafe while accessing their sites. We use the words *ads* and *contents* interchangeably depicting that although the method which we are proposing was applied in an ad moderation setting, it can be extended to any content moderation setting in the industry. To handle the content volumes at industry scale, various systems tend to use multi-tiered workflows containing a mix of manual, semi-automated and automated components for moderating the contents.

The manual component would generally comprise of expert humans moderating the contents. Because expert human-rater capacity is expensive and limited, it would be good to maximize the impact by focusing on contents that have high chance of being unsafe. At the same time, if the moderation bandwidth of the manual moderation component cannot grow at the same scale as the contents to be moderated, it would be desirable to focus on highly deemed unsafe contents.

Typically, most of the content moderation workflows deal with highly skewed data [12, 13] which affects the performance of ML based semi-automated and automated techniques. Therefore, any increase in the number of unsafe contents moderated by the manual moderation component as a whole would help improve the performance of the existing semi-automated or automated components. In workflows where there is a tight service level agreement (SLA) constraint for the contents to get moderated like that of an online advertising portal, the time the human raters spend on mod-

[*] equal contribution



(a)

(b)

(c)

Figure 1. Sample ad contents that are inappropriate and prohibited from advertising on Amazon website. (a) The primary focus of the ad image is to promote or glorify smoking. Smoking is addictive and has serious health consequences. Hence the ads need moderation to filter out smoking and tobacco related ads. (b) The ad image encourages drinking and driving. Alcohol slows down the activity of central nervous system that can lead to reduced reaction time. Hence it is illegal to drive under the influence of alcohol. (c) The ad image displays blood on the weapon that encourages aggressive, violent or threatening behaviour.

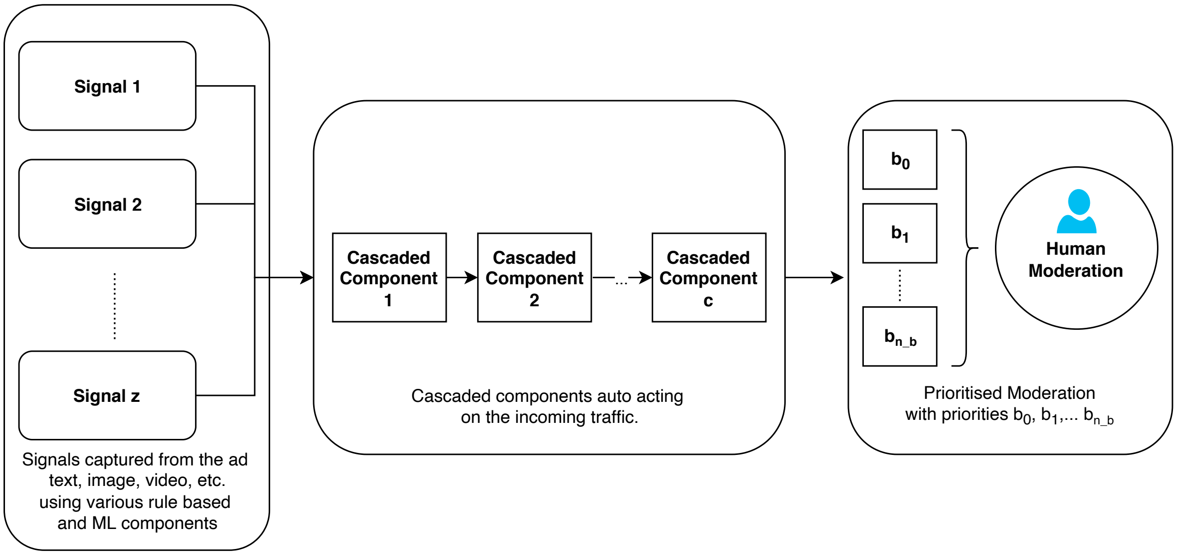

Signal 1
Signal 2
Signal z
Signals captured from the ad text, image, video, etc. using various rule based and ML components
Cascaded Component 1
Cascaded Component 2
...
Cascaded Component c
Cascaded components auto acting on the incoming traffic.
b_0
b_1
b_{n_b}
Human Moderation
Prioritised Moderation with priorities b_0, b_1,... b_{n_b}

Figure 2. A typical automated moderation workflow.

erating unsafe or defective ads would become more crucial. In this paper, we discuss about assigning priorities to the contents using Machine learning techniques to maximize the impact of human raters.

To handle large volumes of ads, moderation workflows tend to deploy various automated components in cascaded [13] mode. In cascaded mode, a series of components act on the incoming traffic in successions resulting in only a few uncertain contents reaching the manual moderation component. Thus, the final number of ads reaching the manual moderation component is unpredictable and varying in nature, making it difficult to control. Hence, priority assignment would be crucial for effective utilisation of human bandwidth.

In such workflows depicted in Figure 2, ML based components have to be trained/retrained on the limited annotated data available and deployed in the workflow where they can also encounter unknown unannotated data distribution. So, the expected performance of the models can be different than offline analysis when measured on unannotated data distribution. To tackle this, we apply unsupervised techniques to learn the interactions between different variables involved from both annotated and unannotated data available in the workflow.

Here are the key contributions from our paper:

- We formulate the priority allocation problem as classification and observed around 5% increase in the number of defective ads caught along with 3.5% reduction in the human moderation hours spent on non-defective ads.
- We also formulate the problem as a policy gradient ranking which resulted in 5.2% reduction in the human moderation hours spent on non-defective ads.
- The ranking objective further increased true high priority defective ads caught by at least 7% as given in confusion matrix.
- We utilise unsupervised techniques to incorporate the knowledge from large chunks of unannotated data and show improvement in F1 score by 5.9% and 1.9% for classification and ranking task respectively, based on a human audit.

2. Background

2.1. Priorities in Ad Moderation

While moderating an ad, it is rejected and sent back to the advertiser if it does not comply to the Amazon's Ad Guidelines and Acceptance Policies [1]. Due to the industry scale volume of ads per day, it is difficult and not cost effective to employ humans to moderate each and every ad. Having an automation engine to identify the policy violations per ad is also challenging since the policies keep on evolving and adversaries come up with creative ways of escaping the policies. Hence, we would want to not only automate the moderation task but also assign a priority to the rejected ad. For example, an ad that is deemed to contain weapon related information might be less severe than an ad containing egregious contents involving children. The business defines p such ordered policy violations and associates a weight w_p to each policy based on the severity. Since the number of policies are large, we map the ordered collection of policies to a relatively small set of n_b buckets. The mapping ensures that most egregious ads go to the top bucket, followed by medium egregious ads in second bucket and so on. At the same time, it maintains comparable distribution of volume across the buckets. This bucket priority is used by human moderation workflow for prioritising the ad moderation subject to the SLA. This makes an efficient use of the limited costly human bandwidth available at our disposal. In all the below experiments, we compare all the models against a *baseline model* used in the production system. The baseline model is a simple rule-based model that allocates a priority bucket to an ad based on the ad policy specific weights w_p. These weights are multiplied with the respective ad policy specific probabilities provided by inhouse models to obtain the ad to bucket mapping.

2.2. Data

We collected around three Million human annotated ads from the Amazon ad workflows to train and validate our model. Each ad belongs to at least one of the n_b different severity bucket (refer the previous section 2.1). We maintain a separate annotated dataset for validation and hyper parameter tuning. All the final metrics are reported on 8 Million annotated contents which is referred as benchmark data. Since the annotated data is limited, we use 10 Million unannotated ads for unsupervised learning. Each ad has fixed set of following features:

- Decisions and probability scores from various ad policy specific image and text based ML models present in the workflow. The image and text models are trained using the classification loss to detect ad policy violations.
- Relevant keywords extracted by ad policy specific text based rules applied on the various text attributes available for the product in the ad.
- Hierarchical catalogue information of the product in the ad. For example, jeans is a subset of clothing set.
- Ad specific geography, marketplace and other metadata information.

2.3. Related Work

A good overview of a multi-tiered ad moderation workflow is given in [13], and the importance of AI in content moderation domain is outlined in [20]. The component which we are proposing typically comes just before the human moderation component and after the various cascaded automated components have taken their decisions. While the method given in [13] suggests stratified sampling from various components, our method tries to assign priorities for all new ads coming into the workflow but also can be expanded to the post moderation workflows after the ads go live.

A deep learning based model to classify the ads into sensitive vs non sensitive categories based on the ad images is described in [12]. The method which we are proposing assumes that there are different input signals available from such models as described in section 2.2. Moreover, the use of various text based or image based ML models in any automated audit workflows may be common which would provide sufficient information for assigning priorities.

[1] For more information: https://advertising.amazon.com/resources/ad-policy/creative-acceptance

Model	Catboost	Xgboost	MLP	TabTransformer	Random Classifier	MLP Ranking
Overall precision % improvement	**10.74**	9.81	10.29	8.94	-6.69	9.70
High priority bucket precision % improvement	6.59	5.56	6.33	5.12	-10.34	**6.76**
Overall F1 % improvement	**6.72**	6.24	3.70	5.07	-13.05	5.39
% increase in moderated defects	**5.01**	4.62	1.99	3.02	-12.58	4.24
% change in human moderation hours wasted on non-defective ads	-3.48	-3.42	-3.61	-4.15	3.33	**-5.22**

Table 1. Comparison of the performance of the classifier (refer section 3.1) and ranking (refer section 3.2) models on the prioritised moderation task. The model task is, given an ad push it into one of the n_b priority buckets. The figures are not absolute numbers and are relative to a simple rule-based baseline model (refer section 2.1). For the last row, lower value is better and for the rest, higher value is better. Best metrics in the respective rows are in bold.

The usage of deep learning for click through rate (CTR) estimation has been increased in recent times as outlined in [22]. And also, reinforcement learning is being used for various post moderation steps such as online biddings of ads, CTR prediction, ad placement based on implicit feedback etc. as given in [6, 7, 21, 23]. While these papers outline use of Bandits and MDP related models in various post moderation stages, we describe the use of policy gradient ranking in pre-moderation phase, to incorporate business driven scalar rewards for assigning the priorities.

The data in our current formulation as described in section 2.2 falls under the regime of tabular data. In the last two decades, end to end learning using gradient descent based techniques via Multi-layer perceptron (MLP) has been a breakthrough for most of problems in the text, image classification domain. However, tabular data modeling is still difficult due to the required robustness in the noisy data and assumption of independence of the features [5]. Hence traditional tree-based ensemble methods like Xgboost [2], Catboost [11] are still relevant for modelling tabular data. At the same time, recent developments in the field have allowed deep learning models like TabTransformer [5] and Tab-net [1] to achieve comparable performance to that of boosting based ensemble models. We compare these relevant models in the section 3.1 where we formulate the priority assignment problem as a classification problem.

While this being said, it is always possible to build an end-to-end deep learning system that can fuse knowledge from different modalities of image, text, video, tabular etc for content moderation. We touch upon the same in section 3.3 by learning tabular contextual embeddings in an unsupervised way which can be used in downstream tasks.

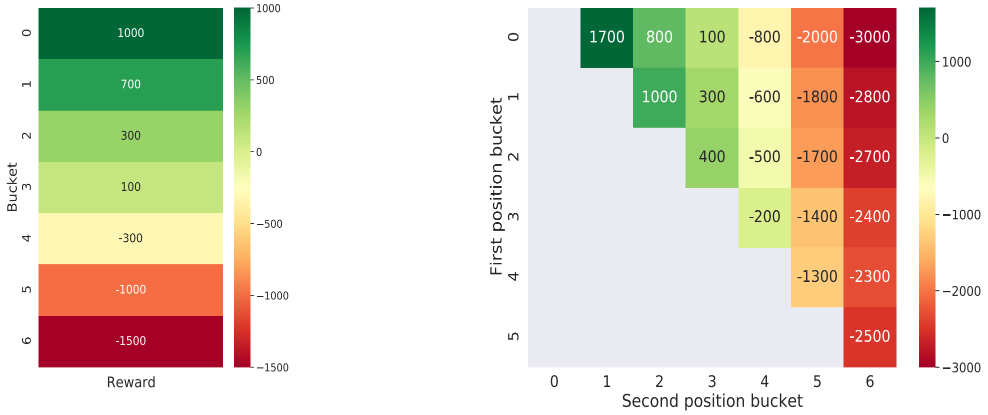
3. Prioritised Ad Moderation

As described in the section 2.1, an annotated ad belongs to at least one of the p policy violation which is mapped to one of the n_b buckets . The goal of Prioritised Ad Moderation is given an ad x with collection of continuous, binary and categorical features, label it in one of the n_b buckets. We approach this problem from two different perspectives, Classification and Policy gradient ranking. We also explore unsupervised techniques to further boost the expected test time performance on unannotated data distributions which the model may encounter when deployed.

3.1. Classification

The intuitive way to solve the ad moderation problem is to view it from the classification lens. Given an ad, the idea is to classify the ad in one of the n_b buckets. We use standard algorithms like (1) Xgboost (2) Catboost (3) MLP (4) TabTransformer (state of the art for tabular data) for solving this as a classification problem. Since this is a multi-class classification problem, the models minimize cross entropy loss. We employ standard data pre-processing techniques to process the ad data. For the categorical features with high cardinality, frequency based selection is performed. The selected labels in the categorical features are then encoded using one-hot encoding. The missing values in the continuous columns are replaced with the feature mean and are scaled using min-max feature scaling.

The results of classification experiments are listed in table 1. The results are corresponding to the best models obtained after extensive hyper parameter tuning as given in Appendix section A.1. In our experiments, we have observed that MLP implemented with Highway layers [15] show around 1% improvement in precision over traditional linear layers. We report overall precision and overall F1 score calculated as the weighted average of all the priority buckets. The high priority bucket precision explicitly captures the precision of the buckets with most egregious contents. As part of business metrics, we report increase in the number of moderated defects and reduction in human moderation hours spent on non-defective ads. The increase in the number of moderated defects captures additional number of defects caught. The



(a) A sample reward vector rv_k

(b) A sample reward matrix rm_k

Figure 3. A sample reward assignment for all the ads belonging to bucket 0 is shown in 3a. For all such ads, when the predicted bucket is close to 0 the reward is high. The reward matrix derived using equation (6) is shown in 3b. The desired bucket sequence prediction when true bucket is 0 is (0,1). Hence $rm_{k=0}$(0,1) gets the highest reward and the reward decreases gradually as the predicted sequence moves away from (0,1).

last metric, explicitly tracks the man-hours wasted while moderating non-defective ads. For efficient utilisation of the human bandwidth, this should be as low as possible. The reported numbers are not absolute and are percentage deviation from a baseline model (refer section 2.1). For comparison, we have also reported the numbers on a random classifier that uses random stratified sampling based on the input bucket distribution to do the classification. We can observe that Catboost and Xgboost models are performing better in classification related metrics when compared with the neural network based models. The TabTransformer model performs better with respect to the reduction in human moderation hours spent, but at the cost of decrease in F1-score when compared with tree based models. While the MLP classifier achieves relatively low performance, MLP trained with a ranking objective performs well as described in section 3.2.

3.2. Policy Gradient Ranking

To incorporate the business knowledge and to better deal with the class imbalance problem, we formulated the priority allocation problem as a policy gradient learning to rank problem across buckets. Policy gradient ranking algorithms allow us to directly maximise custom non differentiable scalar rewards because of the REINFORCE trick [16, 19].

3.2.1 Data

Given a vectorised ad $x_i \in R^d$, and bucket $b_k, k \in \{1, 2...n_b\}$ we use ranking model that generates bucket sequences of size $s, s \leq n_b$ [2]. To simplify the computational complexity, we select s=2. The model parameters are estimated using the objective given below to predict both the exact priority and the next possible priority for the ads. During training time, the model is trained to maximise the probabilities of the top two relevant priority buckets. During the inference time, we take the bucket with top one probability.

3.2.2 Objective

Following the structure given in [4, 14] we formulate the policies π to comprise of two components: a scoring model that defines a distribution over rankings, and its associated sampling method. Starting with the scoring model p_θ, which can be any differentiable machine learning model with parameters θ, for example a linear model or a neural network. Given an input c representing the content vector, the scoring model outputs a vector of scores over n_b buckets as $p_\theta(c) = (p_\theta(c)_1, p_\theta(c)_2, \ldots, p_\theta(c)_{n_b})$. Given this score vector, the probability $\pi(r|c)$ of a ranking $r = (r(1), r(2), \ldots, r(n_b))$ is given as:

$$\pi_\theta(r|c) = \prod_{i=1}^{s} \left(\frac{p_\theta(c)_{r(i)}}{z(c) - \sum_{l=1}^{i-1} p_\theta(c)_{r(l)}} \right), \qquad (1)$$

where $z(c) = \sum_{m=1}^{n_b} p_\theta(c)_m$. In our case, we sample top two possible bucket sequences which implies $s = 2$.

Given the above policy structure and the reward definition, we try to maximise the expected reward over ranking

[2] A dummy bucket b_{n_b+1} is added to generate a dummy sequence from the last bucket of length s=2.

buckets given an ad. Given a content-true priority pair (c, g_c) sampled from the dataset $D(c, g_c)$, and the policy $\pi_\theta(.|c)$, the objective function and its gradient using REINFORCE trick [16, 19] is given as:

$$J(\theta) = E_{(c,g_c)\sim D(c,g_c), r_c \sim \pi_\theta(.|c)}[R(r_c, g_c)] \quad (2)$$

$$\nabla_\theta J(\theta) = E[\nabla_\theta \log \pi_\theta(r_c|c) R(r_c, g_c)] \quad (3)$$

where $R(r_c, g_c)$ is a scalar reward depicting closeness of predicted sequence r_c from true sequence g_c as defined in equation 7. In our case, the true ranking sequence $g_c = (b_k, b_{k+1})$ is also of length two as we are working on length $s = 2$ sequences. The expectation evaluated using Monte Carlo estimate on all possible length two sequences for a given content c is given below, where $B = {}^{n_b}C_2$.

$$\nabla_\theta J_c(\theta) \approx \frac{\sum_{i=1}^{B} \nabla_\theta \log \pi_\theta(r_c^i|c) R(r_c^i, g_c)}{B} \quad (4)$$

3.2.3 Reward

The overall goal is to given an ad, maximise the true bucket mapping and minimise the movement of true positive ads across other buckets. So for an ad where true bucket is b_k, we want the probability of sequence (b_k, b_{k+1}) to be high. For every true bucket $b_k, k \in \{1, 2...n_b\}$, we define a reward vector $rv_k \in R^{n_d}$. Intuitively, for a given ad with true bucket label=b_k, $rv_k(i = k)$ is always maximum and the reward monotonically decreases as the index i moves away from k. Next, as a part of sequence generation, with s=2, for every bucket, the reward matrices are constructed to generate set of all the possible combinations of (i, j) using $rv_k, k \in 1, 2, ...b_n$ as given below.

$$rm_k(i, j) = rv_k(i) + rv_k(j) \quad (5)$$

$$rm_k(i, j) = rm_k(i, j) + ((i - j + 1) * distancePenalty) \quad (6)$$

$$s.t. 1 \leq i \leq n_b, \quad 2 \leq j \leq n_b + 1, \quad i < j$$

Here, $distancePenalty$ is a scalar used to minimise the distance between the predicted buckets in the sequence. Given rm_k, $R(r_c = (i, j), g_c)$ used in equation 2 is defined as:

$$R((i, j), g_c) = rm_{g_c}(i, j) \quad (7)$$

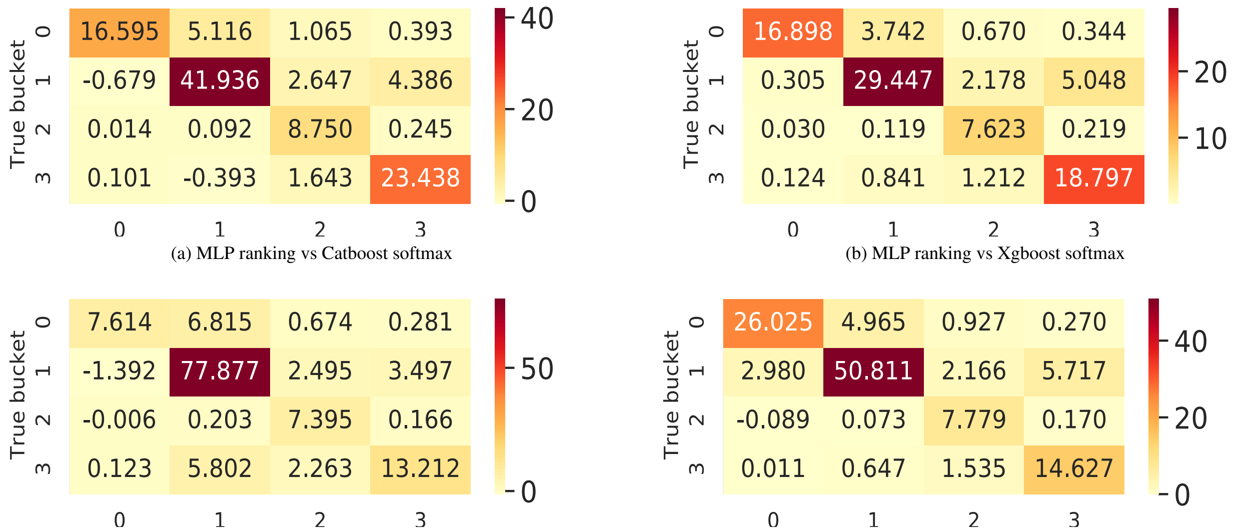
The motivation behind the reward matrix is given an ad, belonging to the bucket b_k, the true sequence should be (b_k, b_{k+1}) in order. For example, consider a possible assignment of rewards as shown in the figure 3a for an ad belonging to bucket, $b_{k=0}$ with $n_b = 5, distancePenalty = 500$. This reward generates the corresponding reward matrix for $rm_{k=0}(i, j)$ as shown in figure 3b. Clearly, the combination $rm_{k=0}(i = 0, j = 1)$ has the highest reward and the reward reduces as i increases. Notice that we penalise the combination $rm_{k=0}(i = 0, j = 6)$. This is because as a part of sequence generation, the sequence indices should be as close as possible. This effectively gives a harder problem for the model to solve thereby improving the defect classification and minimising the movements of true positives. The contextual rewards defined in this way not only try to minimise the distance between the predicted buckets in the sequence s, but they also factor in the penalty of predicting a wrong bucket. Because of this formulation during training time, we can expect the true positive buckets to stay closer to the original bucket during inference. The exact reward mapping is domain specific and we use the aggregate policy violation weights in the respective buckets to generate the reward vectors and the reward matrices.

We train an MLP network [3] with the above ranking objective 3.2.2 with the associated business reward. Refer Appendix section A.2 for details about hyperparameter tuning. As shown in Table 1 MLP model with ranking objective gives the best performance on business metrics such as high priority bucket precision and reduction in human moderation hours on non-defective ads. It gives comparable performance with tree based techniques in the other metrics. The comparative analysis of the percentage improvement in the confusion matrix of the top four buckets is shown in Figure 4. This explains the better performance of MLP ranking model when compared with the classification based models on high priority defect precision. The confusion matrix plots show improved concentration of the true positives in and around the diagonal elements by at least 7%. This improved behaviour over classification based models is what makes ranking based model an ideal candidate for the inference on stream when deployed.

3.3. Unsupervised learning

In a typical advertisement moderation setting, the number of egregious ads are sparse compared to the total number of ads. This coupled with limited human bandwidth and SLA's may lead to the shortage of annotated data to train a model. Moreover, in a workflow containing automated components cascaded after each other, ad volumes which were auto rejected by an automated/ semi-automated component would contain additional information of defective ads. These unannotated volumes that have the potential to improve performance of ML models cannot be included in the training directly. Training a model in an unsupervised way on both annotated and unannotated data would also help bridge the gap between training time and deployment time data distributions. The prioritised moderation component is expected to be deployed right before the human modera-

[3] It is flexible to train a neural network on custom defined gradients with the help of existing autograd packages. It may be possible to train boosting based models using Differentiable Decision Trees (DDT).



(a) MLP ranking vs Catboost softmax

(b) MLP ranking vs Xgboost softmax

(c) MLP ranking vs MLP softmax

(d) MLP ranking vs Tab-tranformer softmax

Figure 4. Each subfigure shows the percentage improvement in the confusion matrix of the top four buckets by the MLP using ranking loss v/s each of the classification based models. All the subfigures show clear improvement in and around the diagonal elements which means ranking is able to gather more number of the true positives near the respective buckets.

With Embeddings Vs Without Embeddings (% improvement)			
Data Type	Overall precision	High priority bucket precision	Overall F1
Classification			
Annotated Data	0.381	0.177	0.205
Audit on 3000 Unannotated samples	9.726	4.995	5.900
Ranking			
Annotated Data	0.276	0.065	0.108
Audit on 3000 Unannotated samples	3.484	0.967	1.975

Table 2. The above table shows the % improvement of a model trained using embeddings over a model which was not trained using embeddings for the prioritised moderation task. We experiment with downstream objectives, classification and ranking. While the model trained using embeddings not only shows slight improvement on the annotated data distribution, it shows significant improvement in the unannotated data distribution based on an human audit for both downstream tasks.

tion component, where it works on volume influenced by all the other components. It has to be robust enough to handle the drifts in the influenced volumes and changes in the ad policies.

To tackle the above mentioned problems, we first train a model in an unsupervised way on both the annotated and unannotated data available in the workflow i.e. the entire data influenced by the automated components. More specifically we train a tab transformer architecture [5] in an unsupervised way for Replaced Token Detection (RTD) task using the sample efficient method described in ELECTRA [3]. The ELECTRA method suggests to train a combination of a generator and a discriminator together. The generator is a masked language model and has roughly a quarter size of the discriminator. The discriminator is then trained to detect the corrupted input tokens which are corrupted by the output softmax distribution of the generator. At the end of the training, generator is discarded and discriminator is used for the downstream tasks. The embeddings learned in this way capture the complex interaction between the categorical columns that can be used by the downstream models on their specific tasks. The same is evident in a 3-dimensional t-SNE [17] plot of the mean pooled contextual embeddings of few ads given in Figure 5.

We train the discriminator with 128 dimensional categorical token embeddings. Please refer to the Appendix section A.3 for the exact hyperparameters used for the unsupervised training phase. Next, we use these embeddings in a Catboost based classifier and an MLP with ranking loss for the prioritised moderation task. We concatenate the mean pooled categorical contextual embeddings obtained from the pre trained discriminator with the continuous columns for

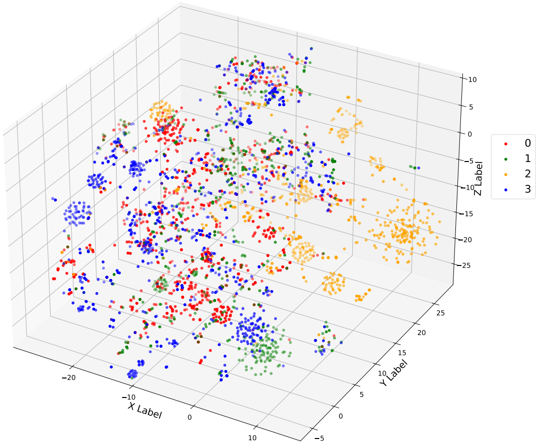


Figure 5. A 3d t-sne plot of the 128 dimensional mean pooled contextual embeddings for a certain set of ads belonging to top 4 priority buckets. We could observe clear separable local clouds when the ads are coloured by the priority labels which were not known to the model during the unsupervised training time.

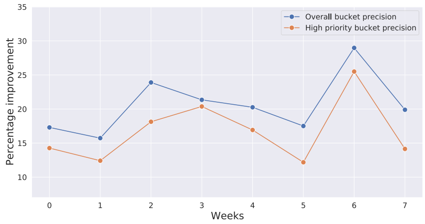


Figure 6. The week-over-week shadow mode model statistics compared to the rule-based model as baseline. The shadow mode model is consistently outperforming the rule-based model in terms of overall bucket precision and high priority bucket precision.

training the models. We compare these models with their respective best models where embeddings were not used for training. To understand the explicit improvements by unsupervised embeddings on the unannotated data distribution, we have performed a human audit. As part of this audit, we sampled 3000 representative contents from unannotated data distribution to compare the models trained with and without embeddings. As given in table 2, the downstream models trained using the embeddings improve upon the models trained without embeddings on the classification metrics. While the improvement in annotated data distribution is close, we could observe a significant improvement on the unannotated data distribution that was not seen by the downstream models during training. We have noticed that the embeddings are adding significant value when used with the classification objective than the ranking objective. This can be because, ranking objective comes with an extra overhead of working over sequences when compared with a simple classification objective.

4. Deployment

We build, train, and prototype all the experiments on AWS Sagemaker. For execution of experiments and hyperparameter tuning at scale, we use Ray [8] library. For all the experiments, bayesian optimization is used to speed up the hyperparameter search. Scikit-learn [10], Pandas [18] and Pytorch [9] libraries were used for all the research prototyping. The model is deployed in production in the "shadow mode" state. In the shadow mode, all the possible decisions taken by the model are captured and stored for analysis. The business decision is still driven by the original model in production. The flexibility of this deployment strategy is that the switch to make it live is relatively simple once the collected results results are convincing.

We deployed a MLP model trained with ranking objective using the embeddings learnt from unsupervised learning in the shadow mode and tracked its performance for few weeks. The week over week comparison of the shadow mode model against the rule-based model in production is shown in Figure 6. We track two metrics, the overall bucket precision and the high priority bucket precision. As seen from the plot, the shadow mode model is consistently outperforming the baseline model in production.

5. Conclusion

In this paper, we presented multiple techniques to tackle the prioritised moderation problem on industry scale. Our method works well even in the realm of data scarcity and class imbalance setting. Initially, the prioritised moderation problem is formulated as a classification task. Here, the tree based models gives the best results due to the tabular setting of the data. However, since each class has priorities, the business demands explicit improvement in the individual class allocation. Modelling prioritised moderation as a ranking task, which gives implicit ordering to the predicted buckets helps in resolving the class overlap problem. Moreover, the explicit reward formulation in policy gradient ranking is an effective way to deal with the class imbalance. We also explored how learning embedding from the un-annotated data can give serious booster in solving the moderation problem. Overall, the methods we discussed are relevant and generic enough for any content moderation workflows. In future, we would like to experiment with the higher sequence lengths of $s > 2$ and different reward formulations in ranking.

References

[1] Sercan O Arık and Tomas Pfister. Tabnet: Attentive interpretable tabular learning. *arXiv*, 2020. 4

[2] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd*

international conference on knowledge discovery and data mining, pages 785–794, 2016. 4

[3] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*, 2020. 7

[4] Phanideep Gampa and Sumio Fujita. Banditrank: Learning to rank using contextual bandits. In *Advances in Knowledge Discovery and Data Mining*, pages 259–271, Cham, 2021. Springer International Publishing. 5

[5] Xin Huang, Ashish Khetan, Milan Cvitkovic, and Zohar Karnin. Tabtransformer: Tabular data modeling using contextual embeddings, 2020. 4, 7

[6] Branislav Kveton, Csaba Szepesvari, Zheng Wen, and Azin Ashkan. Cascading bandits: Learning to rank in the cascade model. In *International Conference on Machine Learning*, pages 767–776, 2015. 4

[7] Chang Li, Artem Grotov, Ilya Markov, Bryan Eikema, and Maarten de Rijke. Online learning to rank with list-level feedback for image filtering. *CoRR*, abs/1812.04910, 2018. 4

[8] Richard Liaw, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E Gonzalez, and Ion Stoica. Tune: A research platform for distributed model selection and training. *arXiv preprint arXiv:1807.05118*, 2018. 8

[9] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dÁlché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 8

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. 8

[11] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. *arXiv preprint arXiv:1706.09516*, 2017. 4

[12] Ashutosh Sanzgiri, Daniel Austin, Kannan Sankaran, Ryan Woodard, Amit Lissack, and Sam Seljan. Classifying sensitive content in online advertisements with deep learning. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 434–441, 2018. 1, 3

[13] D. Sculley, Matthew Eric Otey, Michael Pohl, Bridget Spitznagel, John Hainsworth, and Yunkai Zhou. Detecting adversarial advertisements in the wild. In *Proceedings of the 17th ACM SIGKDD International Conference on Data Mining and Knowledge Discovery*, 2011. 1, 2, 3

[14] Ashudeep Singh and Thorsten Joachims. Policy learning for fairness in ranking, 2019. 5

[15] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015. 4

[16] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000. 5, 6

[17] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. 7

[18] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010. 8

[19] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992. 5, 6

[20] Tim Winchomb. Use of ai in online content moderation. White paper. 3

[21] Wei Zeng, Jun Xu, Yanyan Lan, Jiafeng Guo, and Xueqi Cheng. Multi page search with reinforcement learning to rank. In *Proceedings of the 2018 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 175–178. ACM, 2018. 4

[22] Weinan Zhang, Jiarui Qin, Wei Guo, Ruiming Tang, and Xiuqiang He. Deep learning for click-through rate estimation, 2021. 4

[23] Xiangyu Zhao, Long Xia, Jiliang Tang, and Dawei Yin. “deep reinforcement learning for search, recommendation, and online advertising: a survey” by xiangyu zhao, long xia, jiliang tang, and dawei yin with martin vesely as coordinator. *ACM SIGWEB Newsletter*, (Spring):1–15, Jul 2019. 4