Evolution of content moderation approaches for online classifieds: From action recommendations to automation

Ivan Guz IGUZ@AVITO.RU
Avito.ru, 125047, Moscow, Lesnaya st. 7, RUSSIA

Vasily Leksin VLEKSIN@AVITO.RU
Avito.ru, 125047, Moscow, Lesnaya st. 7, RUSSIA

Mikhail Trofimov MTROFIMOV@AVITO.RU
Avito.ru, 125047, Moscow, Lesnaya st. 7, RUSSIA

Alexandra Fenster AFENSTER@AVITO.RU
Avito.ru, 125047, Moscow, Lesnaya st. 7, RUSSIA

Abstract

We propose a complex approach how to use machine learning algorithms to ensure content quality for online classified advertising platforms. We show how various text mining, deep learning and regression models can be trained and combined together to dramatically enhance capabilities of human moderation and partially automate their job. We show that it is crucial for models to output class probabilities to be able to start with weak predictors that only give action recommendations and then switch to partial automation when models become more accurate. We also provide accuracy comparison of several methods that we tested on real data. Finally we propose how to design such a human computation system to prevent its degeneration and how to test its efficiency. Our approach was successfully implemented at Avito.ru, one of the largest online classifieds in the world, which led to automation of 80% of all moderation actions.

1. Introduction

As online classified advertising platforms become more and more popular it becomes harder to ensure quality of the available content. The more buyers they attract the more attractive site also becomes for swindlers to upload prohibited content. Another challenge is to deal with ordinary sellers who are tying to promote their ads by creating multiple copies of each ad. Those duplicate ads have different descriptions and might have different images but their meaning is the same. The more sellers there are the more ads they create and edit and all that content needs to be verified to comply with the rules of the classified (here and later on we will refer to online classified advertising platform as classified for brevity). At certain point in time sellers start to create automated tools for massive uploading and generation of ad duplicates. Text morphing algorithms are used to change visual ad description and anticapcha crowdsourcing platforms are used to bypass any captcha. Validating all that inflow by scaling moderation team is becoming unrealistic and new approaches are required to address this challenge. One of the potential solution is to start charging fees for each uploaded ad. This approach solves the problem but it also limits number of good ads on site since not all good sellers are willing to pay.

In this paper we propose complex approach that is based on machine learning methods. It allows to automate content verification actions of human moderation starting from most simple ones and then gradually move to more complex cases. In this approach algorithms are trained on actions of humans. In the beginning algorithms only provide certain recommendations with corresponding confidence levels which allows to estimate and get feedback on their accuracy. When their predictive accuracy becomes verified algorithms are allowed to start taking actions on their own in situations when their confidence is high. In the end human moderation focus only on the most complex verification cases while majority of general cases are moderated automatically. This is how this human computation system evolves over time.

This paper consists of three parts. In the first part we formalize requirements for the system that would allow to gradually shift from action recommendations to their automatic execution. Several machine learning methods are proposed to predict actions. In the second part we benchmark proposed methods with other methods that we have tried on the real data from Avito.ru. In the third part we discuss degeneration risks of the proposed system and we provide efficient design of the moderation process that allows to significantly mitigate those risks.

2. Content Inspection System

The content moderation process is a verification process where is each individual ad is checked to comply with a set of rules. If an ad does not comply with a specific rule then it must be rejected with the corresponding reason. Some rules can be easily formalized (E.g. it is prohibited to specify any contact information inside ads description because special fields should be used for that) and some can not (E.g. it is prohibited to place ads in the incorrect categories. However there are many types of ads for which its hard to choose right category). In order to automate moderation process we need to develop predictive model for each reject reason that would specialize on verification of the corresponding rule. It is required that the model should not only predict action (to reject or to allow) but should also estimate its confidence in this action. That would allow us to trust model to take decision when its confidence is high enough and to recommend decision when it not. Because we are studying binary classification problem it is required for each model to predict one number - reject probability $p \in [0, 1]$ for corresponding reason. In this case when p is small the ad should be allowed, when p is large the ad should be rejected, in other cases the ad should be sent to manual verification by human moderation team. We will refer to the system that can predict reject probabilities for various reject reason as Content Inspection System. Lets us now formalize problem definition and describe how this system should work.

2.1. Decision logic

We have historical collection of ads $D = (d_1, \ldots, d_L)$, which have been uploaded to classified. There are K different categories (Cars, Real Estate, Personal belongings, etc.) and each ad d_i is classified (belongs) to a single category c_i. For each ad d_i we know decision vector $\vec{y_i} = (y_1^i, \ldots, y_r^i), y_j^i \in \{0, 1\}$ created by human moderation team. y_j^i is a binary variable which is equal to 1 if ad was rejected for reason j. Each ad d_i is described by 6 groups of data: 1) title and description texts 2) placement of an ad in catalog - category and additional attributes 3) geographic location - region, city, district 4) requested ad price 5) provided images 6) contact information of the seller. Based on this description a vector of numeric features $\vec{f} = (f_1, \ldots, f_N)$ is constructed. Feature preparation logic is unique for each group of data and some details on which methods we used will be shared further.

For each reject reason $j \in 1, \ldots, r$ we need to train a model m_j that should predict reject probability p_j^i for each ad d_i. Also for each reason j we need to define $\delta_j^a \in [0, 1)$ - automatic allow threshold and $\delta_j^r \in (\delta_j^a, 1]$ - automatic reject threshold. Based on these definitions final automatic verification decision $M(d_i)$ should be taken using following logic:

$$M(d_i) = \begin{cases} \forall j : p_j^i < \delta_j^a & \Rightarrow \quad \text{Allow} \\ \exists j : p_j^i > \delta_j^r & \Rightarrow \quad \text{Reject} \\ else & \Rightarrow \quad \begin{array}{l}\text{Recommend to reject} \\ \text{for reason } j = \underset{j}{\operatorname{argmax}}\, p_j^i\end{array} \end{cases}$$

This logic means that we should automatically allow an ad when all models are confident enough to automatically allow it. And we should automatically reject an ad when there is at least one model that is confident enough to automatically reject it. In other cases we should route and ad to manual verification by human moderation team with recommendation to reject it by reason which has highest probability. Because there might be several reject reasons with high probability it is also useful to recommend ad verification for all reasons with high reject probability.

Below we describe which models we have tried training for the most common reject reasons and which worked better.

2.2. Illicit content text models

Ads are considered to be illicit if their public distribution is prohibited by law or by specific rules of the classified. We created two different models for illicit content – one that analyses only text and another that analyses only images. Then we combined them together to predict final probability by selecting maximum of their predicted probabilities. For text models we used various techniques to transform text into numerical features. Mainly we stemmed individual words, extracted word bigrams and replaced them with various frequencies calculated on a large historical collection of texts extracted from D. This feature preparation resulted in a very sparse feature vector $\vec{f}$. Because of the sparse nature of feature vector and large volumes of training cases (10 Mln+) we tried training models that can effectively handle these data volumes: logistic regression (Yu et al., 2011), naive bayes (Manning et al., 2008) and SVM with linear kernel (Boser et al., 1992). Since SVM does not output probabilities by default we fitted logistic regression on margins that it outputs (Platt, 1999).

We also tried creating text models for each individual category by training only on ads that belong to corresponding categories. Category models had higher accuracy on average but failed to detect some obvious cases. Finally we ended up with a linear composition of 2 models: one trained on all available data (predicts probability p^{total}) and one for each individual category (predicts probability $p^{category}$ depending on the category ad is classified into).

$$p_i^{illicit} = a_{total} * p_i^{total} + a_{category} * p_i^{category}$$

, where $a_{total} > 0$ and $a_{category} > 0$ are constants and their sum equals 1.

We trained linear models because they also have a very useful property - we could extract feature weights from them. When a text model predicts high reject probability we show with our recommendation which individual words or bigrams have highest impact on this decision. This functionality was crucial during adoption of content inspection system and building trust in it among moderation team.

2.3. Illicit content image models

Users frequently post items without detailed text description, with only several images present. Without keywords in item description it's almost impossible to detect and filter items with illicit content using text model. So it is crucial to use all the visual information from posted images as it is the only way to detect illicit items in such case. Deep convolutional neural networks are known to be very successful in image classification tasks hence they were used and trained on the image datasets that are full of unique and very specific user content.

Image-based model of illicit content detection is a set of deep convolutional neural networks (Krizhevsky et al., 2012) that were separately trained for each category of illicit content such as weapon, medicine, alcohol and tobacco. For each category train dataset was made so that it is full of various specific positive examples. More specifically, lots of user images with different types of weapon were present as positive examples of images with prohibited content in the train set for corresponding category. All neural networks were of the same architecture which is very similar to the architecture of the BVLC Reference CaffeNet model in Caffe framework (Jia et al., 2014) thus being AlexNet with several modifications. They were trained as a two-class classifiers with the softmax layer at the top using multiple GPUs and reaching the level of 90% accuracy during the training process.

2.4. Incorrect category models

Another frequent issue with content quality is incorrect classification of uploaded ads. It leads to poor buyer experience, when some irrelevant ads are returned while browsing specific category and also poor seller experience when uploaded ad does not generate leads. Although we had at our disposal collection of ads that were identified by moderation to have incorrect category it turned out that training binary classifier on incorrect category reject reason is not the best way to solve this problem. First of all because we did not have representative enough sample and second when predicting incorrect category it is crucial to recommend which category should an ad belong to.

We used different approach. We took large collection of ads that were allowed by human moderation (meaning they were correctly classified) and trained multiclass classifier on this collection. In fact we used one-against-all schema to train binary classifiers for each category and then normalized their output probabilities (Zadrozny & Elkan, 2002). We used same features as in illicit content text models. We tried training SVM with linear kernel and logistic regression since they can handle sparse feature vectors and large number of training objects.

Using trained multiclass category classifier we can predict for each ad d probabilities $p(c_1), \dots, p(c_k)$. Lets sort those probabilities in descending order: $p(c_{i_1}) >= p(c_{i_2}) >= \dots >= p(c_{i_k})$, where class c_{i_1} has the highest probability. Also lets denote c_d as a category of ad d that it has been classified into. Using those definitions we defined incorrect category probability using following formula:

$$p = \begin{cases} 0, \text{ if } c_d \in \{c_{i_1}, \dots, c_{i_t}, c_{i_{t+1}}\} : \sum_{j=1}^{t} p(c_{i_j}) < \delta \\ 1 - p(c_d), \text{ else} \end{cases}$$

Here δ is constant that we estimated experimentally. Given formula results in a model that cannot reject an ad for incorrect category if its specified category is among several most probable categories. By increasing δ we can increase specificity which is more important then recall for this model. If reject probability $p > 0$ then model will also recommend to classify ad into category c_{i_1}. We tried using obvious formula $p = 1 - p(c_d)$ in all cases but it turned out that on our collection of ads it produced too many rejects for incorrect category reason.

2.5. Price models

Underestimated price is usually a good predictor that an ad is suspicious and should be thoroughly verified. Ads with significantly overestimated prices are in most cases just input errors. They are ignored by buyers but lead to poor selling experience. Price models are used to estimate actual price of an ad and its distribution based on category and attributes of an ad. For different categories different models are trained. For all models it is crucial for their predictions to be explainable.

For real estate objects price is normalized by its area to get price per m^2. Then KNN regression with euclidean distance between location coordinates is used to estimate m^2 price based on prices of k=35 neighbors. In our approach neighbor weights are declining proportionate to their distance and multiplicative correction coefficients are applied based on object building type, floor number, number of rooms, etc. Those coefficients were fitted to minimize Mean Squared Error.

For used cars price model is based on car characteristics like manufacturer, model, mileage, etc. To make model easily explainable our task was to find for each specific car such a slice of car characteristics in which cars would still be considered almost the same and number of cars in this slice should not be less then $M = 20$ to be able to estimate price distribution. To satisfy those requirements we trained decision tree regressor for each car model with minimum leaf size equals to M. Cars that fall into the same tree leaf are similar because they have similar price and each leaf is defined by a set of rules on car characteristics which we identified as a slice we were looking for. We selected best decision tree training method that minimized RMSLE on the training data. It could not overfit because we had restriction on a minimum leaf size.

All price models were trained only on ads with correct prices that have been explicitly accepted by human moderation. Using price distributions simple heuristics were used to transform distribution into reject probability estimates for incorrect price. The further specified price is from median price in the distribution - the larger is reject probability.

2.6. Duplicate models

Purpose of duplicate model is to find for each individual ad similar ads and predict probability for each candidate that it is a duplicate. Duplicate models are much more technologically sophisticated then other models since its impossible to compare ad with all tens of millions of other active ads in realtime. An algorithm consisting of two steps is used to find duplicates and estimate their probability.

1) On the first step duplicate candidates are selected by exact matches of seller contact information or by fuzzy matches of images or texts. To quickly find fuzzy matches on images various distortion-tolerant hash functions computed on images are used as indexes (pixel average hashes, pixel difference hashes). To quickly find fuzzy matches on texts Local Sensitivity Hashes (Gionis et al., 1999) are used as indexes on text fragments.

2) On the second step logistic regression model is applied to each candidate to estimate duplicate probability. This logistic regression uses features that are numeric representations of text similarity, images similarity, price similarity, equality of attributes, etc. We trained it on a large collection of different ad pairs that have been labeled to be duplicate or not.

Table 1. Illicit content text models testing

MODEL	AUC
LOGREG	0.9951
SVM	0.9948
MULTINOMIALNB	0.9759

3. Experiments

This section presents the results of testing of some of the models described above.

3.1. Testing illicit content text models

Full dataset included 3 months of sampled real data from Avito.ru. The training dataset included two months (5.2 Mln ads) and test dataset included one month (2.7 Mln ads). Ads from test dataset were uploaded chronologically after ads from training dataset to estimate how good classifier would generalize for future ads. Area Under Curve (AUC) was used as a quality metric. For text classifiers were tested three models: SVM with linear kernel, Logistic Regression and Multinomial Naive Bayes. Because our data sample was unbalanced, we used the class weighting. Table 1 shows AUC for models on a test dataset. We also trained models for each category independently and their accuracy was almost the same. Logistic Regression has almost the same AUC as SVM. Since SVM is faster to train we used SVM. To increase generalization we also combined both category and total models which proved to give more stable results in practice.

3.2. Testing illicit content image models

As we described above CNNs are used to detect prohibited images and here we provide you with testing results for three different categories: weapon; medicine; alcohol and tobacco.

To test all these networks properly we've created several different test datasets. Specifically for testing model that was trained for weapon category three datasets were created:

- Set 1: images from "clean" ads that are free from prohibited content and were not rejected by any reason.
- Set 2: images from "missed" ads: those ads were rejected by moderation but were not labeled by text

Table 2. CNN models testing

MODEL	SET 1	SET 2	SET 3
WEAPON	0.038	0.484	0.595
ALCOHOL	0.034	0.435	0.558
MEDICINE	0.034	0.324	0.245

model as prohibited

- Set 3: images from ”caught” items: items that were rejected by moderation and also labeled by the text model as prohibited

Similar datasets were created for the rest of categories. Set 1 was used to measure false positives rate: as far as all these ads are totally free from prohibited content we should keep the rate of the model rejects for this set significantly low. Set 2 and set 3 were used to measure true positives rate: we know that these ads were blocked so its very likely for at least one image from each ad to be an image with prohibited content.

It is important to mention that almost every network demonstrates high rate on the set 2, up to 75 %, but the level of rejects on the set 1 is unacceptable. So we were forced to increase the threshold of prediction to keep the percentage of errors on the level of 3%. With this value on the set 1, networks were able to detect from 30% to 60% images on two other sets (table 2).

Finally we were able to significantly increase rate of detected ads using image-based model with several convolutional neural networks.

3.3. Testing incorrect category models

For training incorrect category model we used ads that were explicitly allowed by human moderators to ensure that they have correct category. We used 2.5 Mln ads for training and 1.3 Mln for testing. The accuracy (percentage of correct category predictions in the δ vicinity of top predicted categories) for corresponding δ parameter values are shown in figure 1. We tested fitting Logistic regression, SVM with linear kernel and Multinominal Naive Bayes models. SVM clearly dominates other models and 98% accuracy is achieved at $\delta = 0.65$.

3.4. Testing price models

For real estate price model testing we divided dataset of explicitly allowed by human moderation real estate objects into train/test in 80/20 ratio. Average relative error was measured as $\frac{\max(P_{predict}, P_{fact})}{\min(P_{predict}, P_{fact})} - 1$, where $P_{predict}$ is prediction price, P_{fact} – actual price. We compared two models: KNN and KNN with a correction multiplicative coefficients. The results are shown in table 3 for testing different models for Flats Rentals and Flats for Sale.

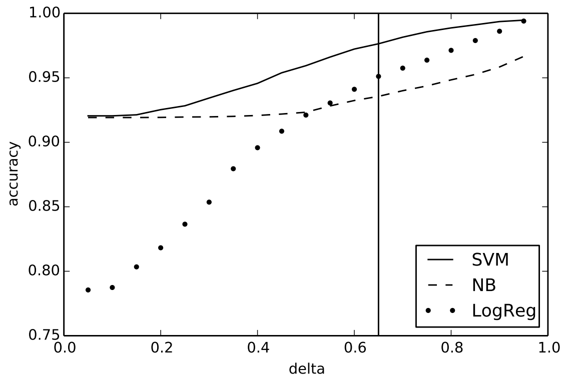

Figure 1. Category model testing

Table 3. RE price model testing

CATEGORY	KNN-ONLY	KNN + COEFF
FLATS RENTALS	0.197	0.183
FLATS FOR SALE	0.113	0.107

For car price model we evaluated Decision Tree Regressor that we described above against linear regression with L1 regularization (Lasso) (Tibshirani, 1996). We used 10-fold cross-validation to estimate RMSLE on a test set. Our experiments showed that Decision Tree Regressor (RMSLE = 0.268) has almost the same accuracy as liner regression (RMSLE = 0.269). Since model that it produces is easier to interpret and also give empirical price distribution (price distribution in the leaf that the car falls into) we selected it.

3.5. Testing duplicates models

The duplicates model was trained separately for various categories. Logistic regression and SVM with linear kernel were tested using 10-fold cross validation. Area Under Curve (AUC) was used as the quality metric. Results on a test dataset for Cars (98 985 samples) and Real Estate (12 273 samples) categories are shown in table 4. In both cases Logistic Regression clearly wins over SVM.

4. Efficient process design

We now discuss full moderation process where both Content Inspection System and human moderation team play important roles. Design of efficient moderation process

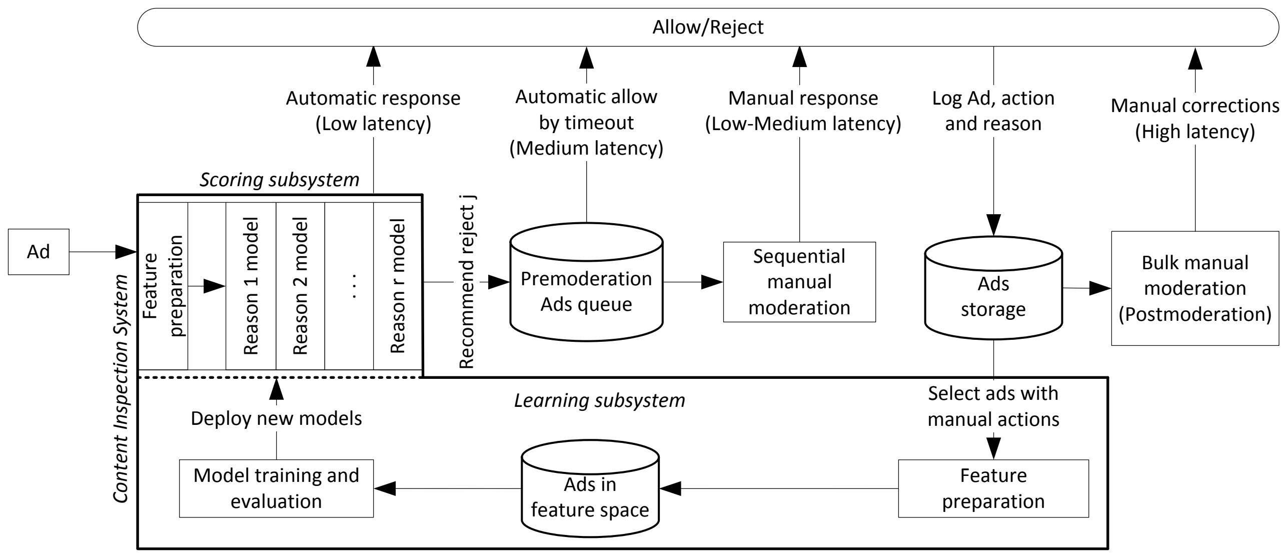
Allow/Reject
Automatic response (Low latency)
Automatic allow by timeout (Medium latency)
Manual response (Low-Medium latency)
Log Ad, action and reason
Manual corrections (High latency)
Scoring subsystem
Ad
Feature preparation
Reason 1 model
Reason 2 model
. . .
Reason r model
Recommend reject j
Premoderation Ads queue
Sequential manual moderation
Ads storage
Bulk manual moderation (Postmoderation)
Content Inspection System
Learning subsystem
Deploy new models
Select ads with manual actions
Model training and evaluation
Ads in feature space
Feature preparation

Figure 2. Moderation process

Table 4. Duplicate models testing

CATEGORY	MODEL	AUC
REAL ESTATE	LOGREG	0.914
REAL ESTATE	SVM	0.891
CARS	LOGREG	0.848
CARS	SVM	0.823

should satisfy following requirements:

1. Time on site for prohibited ads should be minimized to reduce reputation risks.

2. Verification time should be minimized to make ads appear as soon as possible on site to start attracting buyers.

3. Number of incorrect decisions should be kept withing acceptable levels.

Diagram showing full moderation process that satisfies those requirements is shown in Figure 2.

When ad is uploaded it is sent to the Scoring subsystem for automatic verification. Data that describes an ad is transformed into numeric feature vector and reject probabilities for each reason are predicted by corresponding models. If all reject probabilities are below corresponding allowance thresholds then the ad is automatically allowed. If at least one reject probability is above corresponding reject threshold then the ad is automatically rejected. To estimate allowance and reject thresholds for each reason we maximized number of automated decisions as much as possible at the same time keeping percentage of false allows and false rejects within acceptable levels. Because on this step decision is taken by models, we can take decisions very fast and able to scale to virtually any inflow volumes by launching multiple instances of Scoring Subsystem in parallel. This is how requirements (1) and (2) are satisfied.

If models cannot automatically decide which action should be taken then the ad is added to the priority queue (called premoderation queue) for manual moderation by humans. In this queue ad priority is equal to the maximum reject probability recommended by models. We are using priority queue because in order to satisfy (1) human moderation should first verify the most suspicious ads. This means the fewer human moderators we have the more efficient each of them will become, because they will first be checking the most simple cases. However this approach also has a drawback - ads with low reject probabilities (but not low enough to be automatically rejected) can stay in the queue for a very long time. If there is not enough people to verify all remaining content some ads might stay in the queue forever which will violate (2). In order to solve this issue each ad has a maximum lifetime inside premoderation queue. When this lifetime expires the ad is automatically allowed to satisfy (2).

So far we have relied on the accuracy of the decisions automatically taken by models. But what would happen when rules are changed and new reject reason is added (cold start problem) or if swindlers find a new way how to upload prohibited ads that are not recognized by models and in this case all of them become automatically allowed? This would violate (3). To address this issue there is a postmoderation process where dedicated human moderation team performs spot checks of the ads that have already been allowed and they can reject those ads if needed. Main advantage of this process is that it allows humans to spot new prohibited patterns that were unseen previously and they can reject ads that contain those patterns in bulk without the need to validate each ad individually. Main disadvantage of this process - it is hard to measure efficiency of postmoderation team because spot checks do not result in any actions if content is clean.

After each decision to allow or reject an ad this action is logged with all corresponding details about the ad, taken decision, author of this decision and list of rejected reasons (in case ad was rejected). In the Learning subsystem models are trained only on the actions taken by human moderation team. Actions taken automatically are not taken into account because they do not contain new knowledge. This is how we prevent degeneration of the system by prohibiting it to train on potentially erroneous decisions it might be taking.

In order to estimate and control number of errors, required for (3) we implemented a dedicated tool for continuous automatic testing of full moderation process. This tool draws samples of ads that were previously rejected by human moderation for each reason and tries to upload them for the second time. This allows us to measure recall (percentage of ads that were actually rejected) for models and for human moderation. Without this tool it is impossible to calculate unbiased estimates of recall because if at some point in time number of false allows would increase we will not be able to spot it based on the available data. If recall falls below tolerance levels then either models need to be retrained or some administrative actions needs to be taken to boost human moderation performance. This is how learning subsystem together with automatic testing tool ensures that (3) is satisfied as well.

5. Conclusions and future work

Moderation process shown in Figure 2 was successfully implemented at Avito.ru, one of the largest online classifieds in the world and the largest in Russia. As a result 80% of 1Mln+ ads that are sent to moderation per day are automatically verified and actionable recommendations on the remaining 20% are given. As the models start automatic moderation of simple cases humans can focus on a more complex ones. Also in time we discover how to train and implement more accurate predictive models. This allows us to lower reject threshold and raise acceptance threshold for the models and ad segments where system has evolved enough and is ready for this change. This leads to further automation.

Proposed process can also be expanded to crowdsource content verification tasks. Indeed because models produce most probable reject reason external moderator does not need to know large list of rules. It is only required from him to verify if specific problem is present in a specific ad. Crowdsourcing would allow to efficiently scale moderation team and perform cross checks of the decisions taken by humans. This could increase consistency of the training data, further increase model accuracy and push whole system into the next stage of its evolution.

Acknowledgments

We wish to thank Roman Pavlushko and Mikhail Tyurin for integration of Content Inspection System into existing moderation process and wise technological advice, Nikolay Golov for managing all the data required for model training, Dmitry Kovalev and Christoffer Norman for executive support during system adoption.

References

Boser, Bernhard E., Guyon, Isabelle M., and Vapnik, Vladimir N. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pp. 144–152, New York, NY, USA, 1992. ACM. ISBN 0-89791-497-X. doi: 10.1145/130385.130401. URL http://doi.acm.org/10.1145/130385.130401.

Gionis, Aristides, Indyk, Piotr, Motwani, Rajeev, et al. Similarity search in high dimensions via hashing. In *VLDB*, volume 99, pp. 518–529, 1999.

Jia, Yangqing, Shelhamer, Evan, Donahue, Jeff, Karayev, Sergey, Long, Jonathan, Girshick, Ross, Guadarrama, Sergio, and Darrell, Trevor. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C.J.C., Bottou, L., and Weinberger, K.Q. (eds.), *Advances in Neural Information Processing Systems 25*, pp. 1097–1105. Curran Associates, Inc., 2012.

Manning, Christopher D., Raghavan, Prabhakar, and Schtze, Hinrich. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

Platt, John C. Probabilities for sv machines. In *Advances in Large Margin Classifiers*, pp. 61–74. MIT Press, March 1999. URL http://research.microsoft.com/apps/pubs/default.aspx?id=69187.

Tibshirani, Robert. Regression shrinkage and selection via lasso. *Journal of Royal Statistical Society. Series B. (Methodological)*, 58(1):267–288, 1996. ISSN 0885-6125.

Yu, Hsiang-Fu, Huang, Fang-Lan, and Lin, Chih-Jen. Dual coordinate descent methods for logistic regression and maximum entropy models. *Mach. Learn.*, 85(1-2):41–75, October 2011. ISSN 0885-6125. doi: 10.1007/s10994-010-5221-8. URL http://dx.doi.org/10.1007/s10994-010-5221-8.

Zadrozny, Bianca and Elkan, Charles. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pp. 694–699, New York, NY, USA, 2002. ACM. ISBN 1-58113-567-X. doi: 10.1145/775047.775151. URL http://doi.acm.org/10.1145/775047.775151.