Improved Abusive Comment Moderation with User Embeddings

John Pavlopoulos
Prodromos Malakasiotis
Juli Bakagianni
Straintek, Athens, Greece
{ip, mm, jb}@straintek.com

Ion Androutsopoulos
Department of Informatics
Athens University of Economics
and Business, Greece
ion@aueb.gr

Abstract

Experimenting with a dataset of approximately 1.6M user comments from a Greek news sports portal, we explore how a state of the art RNN-based moderation method can be improved by adding user embeddings, user type embeddings, user biases, or user type biases. We observe improvements in all cases, with user embeddings leading to the biggest performance gains.

1 Introduction

News portals often allow their readers to comment on articles, in order to get feedback, engage their readers, and build customer loyalty. User comments, however, can also be abusive (e.g., bullying, profanity, hate speech), damaging the reputation of news portals, making them liable to fines (e.g., when hosting comments encouraging illegal actions), and putting off readers. Large news portals often employ moderators, who are frequently overwhelmed by the volume and abusiveness of comments.[1] Readers are disappointed when non-abusive comments do not appear quickly online because of moderation delays. Smaller news portals may be unable to employ moderators, and some are forced to shut down their comments.[2]

In previous work (Pavlopoulos et al., 2017a), we introduced a new dataset of approx. 1.6M manually moderated user comments from a Greek sports news portal, called Gazzetta, which we made publicly available.[3] Experimenting on that dataset and the datasets of Wulczyn et al. (2017), which contain moderated English Wikipedia comments, we showed that a method based on a Recurrent Neural Network (RNN) outperforms DETOX (Wulczyn et al., 2017), the previous state of the art in automatic user content moderation.[4] Our previous work, however, considered only the texts of the comments, ignoring user-specific information (e.g., number of previously accepted or rejected comments of each user). Here we add *user embeddings* or *user type embeddings* to our RNN-based method, i.e., dense vectors that represent individual users or user types, similarly to word embeddings that represent words (Mikolov et al., 2013; Pennington et al., 2014). Experiments on Gazzetta comments show that both user embeddings and user type embeddings improve the performance of our RNN-based method, with user embeddings helping more. User-specific or user-type-specific *scalar biases* also help to a lesser extent.

2 Dataset

We first discuss the dataset we used, to help acquaint the reader with the problem. The dataset contains Greek comments from Gazzetta (Pavlopoulos et al., 2017a). There are approximately 1.45M training comments (covering Jan. 1, 2015 to Oct. 6, 2016); we call them G-TRAIN (Table 1). An additional set of 60,900 comments (Oct. 7 to Nov. 11, 2016) was split to development set (G-DEV, 29,700 comments) and test set (G-TEST, 29,700).[5] Each comment has a gold label ('accept', 'reject'). The user ID of the author of each comment is also available, but user IDs were not used in our previous work.

When experimenting with *user type* embeddings or biases, we group the users into the fol-

[1] See, for example, https://goo.gl/WTQyio.

[2] See https://goo.gl/2eKdeE.

[3] The portal is http://www.gazzetta.gr/. Instructions to download the dataset will become available at http://nlp.cs.aueb.gr/software.html.

[4] Two of the co-authors of Wulczyn et al. (2017) are with Jigsaw, who recently announced Perspective, a system to detect toxic comments. Perspective is not the same as DETOX (personal communication), but we were unable to obtain scientific articles describing it.

[5] The remaining 1,500 comments are not used here. Smaller subsets of G-TRAIN and G-TEST are also available (Pavlopoulos et al., 2017a), but are not used in this paper. The Wikipedia comment datasets of Wulczyn et al. (2017) cannot be used here, because they do not provide user IDs.

Proceedings of the 2017 EMNLP Workshop on Natural Language Processing meets Journalism, pages 51–55
Copenhagen, Denmark, September 7, 2017.

Dataset/Split	Gold Label		Comments Per User Type				Total
	Accepted	Rejected	Green	Yellow	Red	Unknown	
G-TRAIN	960,378 (66%)	489,222 (34%)	724,247 (50%)	585,622 (40%)	43,702 (3%)	96,029 (7%)	1.45M
G-DEV	20,236 (68%)	9,464 (32%)	14,378 (48%)	10,964 (37%)	546 (2%)	3,812 (13%)	29,700
G-TEST	20,064 (68%)	9,636 (32%)	14,559 (49%)	10,681 (36%)	621 (2%)	3,839 (13%)	29,700

Table 1: Comment statistics of the dataset used.

Dataset/Split	Individual Users Per User Type				Total
	Green	Yellow	Red	Unknown	
G-TRAIN	4,451	3,472	251	21,865 → 1	8,175
G-DEV	1,631	1,218	64	1,281 → 1	2,914
G-TEST	1,654	1,203	67	1,254 → 1	2,925

Table 2: User statistics of the dataset used.

lowing types. $T(u)$ is the number of training comments posted by user (ID) u. $R(u)$ is the ratio of training comments posted by u that were rejected.

Red: Users with $T(u) > 10$ and $R(u) \geq 0.66$.
Yellow: $T(u) > 10$ and $0.33 < R(u) < 0.66$.
Green: $T(u) > 10$ and $R(u) \leq 0.33$.
Unknown: Users with $T(u) \leq 10$.

Table 2 shows the number of users per type.

3 Methods

RNN: This is the RNN-based method of our previous work (Pavlopoulos et al., 2017a). It is a chain of GRU cells (Cho et al., 2014) that transforms the tokens $w_1 \ldots, w_k$ of each comment to the hidden states $h_1 \ldots, h_k$ ($h_i \in \mathbb{R}^m$). Once h_k has been computed, a logistic regression (LR) layer estimates the probability that comment c should be rejected:

$$P_{\text{RNN}}(\textit{reject}|c) = \sigma(W_p h_k + b) \quad (1)$$

σ is the sigmoid function, $W_p \in \mathbb{R}^{1\times m}$, $b \in \mathbb{R}$.[6]

***ue*RNN:** This is the RNN-based method with *user embeddings* added. Each user u of the training set with $T(u) > 10$ is mapped to a user-specific embedding $v_u \in \mathbb{R}^d$. Users with $T(u) \leq 10$ are mapped to a single 'unknown' user embedding. The LR layer is modified as follows; v_u is the embedding of the author of c; and $W_v \in \mathbb{R}^{1\times d}$.

$$P_{\textit{ue}\text{RNN}}(\textit{reject}|c) = \sigma(W_p h_k + W_v v_u + b) \quad (2)$$

***te*RNN:** This is the RNN-based method with *user type embeddings* added. Each user type t is mapped to a user type embedding $v_t \in \mathbb{R}^d$. The LR layer is modified as follows, where v_t is the embedding of the type of the author of c.

$$P_{\textit{te}\text{RNN}}(\textit{reject}|c) = \sigma(W_p h_k + W_v v_t + b) \quad (3)$$

***ub*RNN:** This is the RNN-based method with *user biases* added. Each user u of the training set with $T(u) > 10$ is mapped to a user-specific bias $b_u \in \mathbb{R}$. Users with $T(u) \leq 10$ are mapped to a single 'unknown' user bias. The LR layer is modified as follows, where b_u is the bias of the author of c.

$$P_{\textit{ub}\text{RNN}}(\textit{reject}|c) = \sigma(W_p h_k + b_u) \quad (4)$$

We expected *ub*RNN to learn higher (or lower) b_u biases for users whose posts were frequently rejected (accepted) in the training data, biasing the system towards rejecting (accepting) their posts.

***tb*RNN:** This is the RNN-based method with *user type biases*. Each user type t is mapped to a user type bias $b_t \in \mathbb{R}$. The LR layer is modified as follows; b_t is the bias of the type of the author.

$$P_{\textit{tb}\text{RNN}}(\textit{reject}|c) = \sigma(W_p h_k + b_t) \quad (5)$$

We expected *tb*RNN to learn a higher b_t for the red user type (frequently rejected), and a lower b_t for the green user type (frequently accepted), with the biases of the other two types in between.

In all methods above, we use 300-dimensional word embeddings, user and user type embeddings with $d = 300$ dimensions, and $m = 128$ hidden units in the GRU cells, as in our previous experiments (Pavlopoulos et al., 2017a), where we tuned all hyper-parameters on 2% held-out training comments. Early stopping evaluates on the same held-out subset. User and user type embeddings are randomly initialized and updated by backpropagation. Word embeddings are initialized to the WORD2VEC embeddings of our previous work (Pavlopoulos et al., 2017a), which were pretrained on 5.2M Gazzetta comments. Out of vocabulary words, meaning words not encountered or encountered only once in the training set and/or words with no initial embeddings, are mapped (during both training and testing) to a single randomly initialized word embedding, updated by backpropagation. We use Glorot initialization (Glorot and

[6] In our previous work (Pavlopoulos et al., 2017a), we also considered a variant of RNN, called *a*-RNN, with an attention mechanism. We do not consider *a*-RNN here to save space.

System	G-DEV	G-TEST
*ue*RNN	**80.68** (±**0.11**)	**80.71** (±**0.13**)
*ub*RNN	80.54 (±0.09)	80.53 (±0.08)
*te*RNN	80.37 (±0.05)	80.41 (±0.09)
*tb*RNN	80.33 (±0.12)	80.32 (±0.05)
RNN	79.40 (±0.08)	79.24 (±0.05)
*u*BASE	67.61	68.57
*t*BASE	63.16	63.82

Table 3: AUC scores. Standard error in brackets.

Bengio, 2010) for other parameters, cross-entropy loss, and Adam (Kingma and Ba, 2015).[7]

***u*BASE:** For a comment c authored by user u, this baseline returns the rejection rate $R(u)$ of the author's training comments, if there are $T(u) > 10$ training comments of u, and 0.5 otherwise.

$$P_{u\text{BASE}}(\textit{reject}|c) = \begin{cases} R(u), \text{ if } T(u) > 10 \\ 0.5, \text{ if } T(u) \leq 10 \end{cases}$$

***t*BASE:** This baseline returns the following probabilities, considering the user type t of the author.

$$P_{t\text{BASE}}(\textit{reject}|c) = \begin{cases} 1, \text{if } t \text{ is Red} \\ 0.5, \text{if } t \text{ is Yellow} \\ 0.5, \text{if } t \text{ is Unknown} \\ 0, \text{if } t \text{ is Green} \end{cases}$$

4 Results and Discussion

Table 3 shows the AUC scores (area under ROC curve) of the methods considered. Using AUC allows us to compare directly to the results of our previous work (Pavlopoulos et al., 2017a) and the work of Wulczyn et al. (2017). Also, AUC considers performance at multiple classification thresholds t (rejecting comment c when $P(\textit{reject}|c) \geq t$, for different t values), which gives a more complete picture compared to reporting precision, recall, or F-scores for a particular t only. Accuracy is not an appropriate measure here, because of class imbalance (Table 1). For methods that involve random initializations (all but the baselines), the results are averaged over three repetitions; we also report the standard error across the repetitions.

User-specific information always improves our original RNN-based method (Table 3), but the best results are obtained by adding user embeddings (*ue*RNN). Figure 1 visualizes the user embeddings learned by *ue*RNN. The two dimensions of Fig. 1 correspond to the two principal components of the user embeddings, obtained via PCA.The colors and numeric labels reflect the rejection rates $R(u)$ of the corresponding users. Moving from left to right in Fig. 1, the rejection rate increases, indicating that the user embeddings of *ue*RNN capture mostly the rejection rate $R(u)$. This rate (a single scalar value per user) can also be captured by the simpler user-specific biases of *ub*RNN, which explains why *ub*RNN also performs well (second best results in Table 3). Nevertheless, *ue*RNN performs better than *ub*RNN, suggesting that user embeddings capture more information than just a user-specific rejection rate bias.[8]

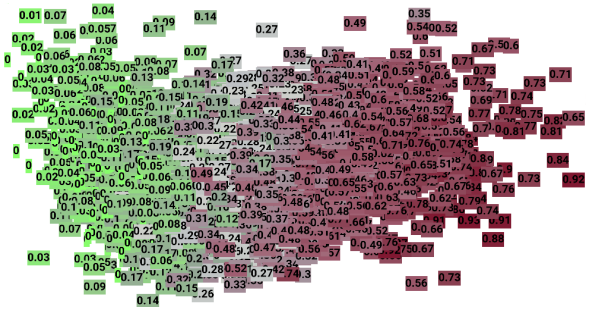

Figure 1: User embeddings learned by *ue*RNN (2 principal components). Color represents the rejection rate $R(u)$ of the user's training comments.

Three of the user types (Red, Yellow, Green) in effect also measure $R(u)$, but in discretized form (three bins), which also explains why user type embeddings (*te*RNN) also perform well (third best method). The performance of *tb*RNN is close to that of *te*RNN, suggesting again that most of the information captured by user type embeddings can also be captured by simpler scalar user-type-specific biases. The user type biases b_t learned by *tb*RNN are shown in Table 4. The bias of the Red type is the largest, the bias of the Green type is the smallest, and the biases of the Unknown and Yellow types are in between, as expected (Section 3). The same observations hold for the average user-specific biases b_u learned by *ub*RNN (Table 4).

Overall, Table 3 indicates that user-specific information (*ue*RNN, *ub*RNN) is better than user-type information (*te*RNN, *tb*RNN), and that embeddings (*ue*RNN, *te*RNN) are better than the scalar biases (*ub*RNN, *tb*RNN), though the differences are small. All the RNN-based methods outperform the two baselines (*u*BASE, *t*BASE), which do not consider the texts of the comments.

Let us provide a couple of examples, to illustrate the role of user-specific information. We en-

[7] We used Keras (http://keras.io/) with the TensorFlow back-end (http://www.tensorflow.org/).

[8] We obtained no clear clusterings with tSNE (van der Maaten and Hinton, 2008).

User Type	b_t of *tb*RNN	average b_u of *ub*RNN
Green	-0.471 (± 0.007)	-0.180 (± 0.024)
Yellow	0.198 (± 0.015)	0.058 (± 0.022)
Unknown	0.256 (± 0.021)	0.312 (± 0.011)
Red	1.151 (± 0.013)	0.387 (± 0.023)

Table 4: Biases learned and standard error.

countered a comment saying just "Ooooh, down to Pireaus..." (translated from Greek), which the moderator had rejected, because it is the beginning of an abusive slogan. The rejection probability of RNN was only 0.34, presumably because there are no clearly abusive expressions in the comment, but the rejection probability of *ue*RNN was 0.72, because the author had a very high rejection rate. On the other hand, another comment said "Indeed, I know nothing about the filth of Greek soccer." (translated, apparently not a sarcastic comment). The original RNN method marginally rejected the comment (rejection probability 0.57), presumably because of the 'filth' (comments talking about the filth of some sport or championship are often rejected), but *ue*RNN gave it a very low rejection probability (0.15), because the author of the comment had a very low rejection rate.

5 Related work

In previous work (Pavlopoulos et al., 2017a), we showed that our RNN-based method outperforms DETOX (Wulczyn et al., 2017), the previous state of the art in user content moderation. DETOX uses character or word n-gram features, no user-specific information, and an LR or MLP classifier. Other related work on abusive content moderation was reviewed extensively in our previous work (Pavlopoulos et al., 2017a). Here we focus on previous work that considered user-specific features and user embeddings.

Dadvar et al. (2013) detect cyberbullying in YouTube comments, using an SVM and features examining the content of each comment (e.g., second person pronouns followed by profane words, common bullying words), but also the profile and history of the author of the comment (e.g., age, frequency of profane words in past posts). Waseem et al. (2016) detect hate speech tweets. Their best method is an LR classifier, with character n-grams and a feature indicating the gender of the author; adding the location of the author did not help.

Cheng et al. (2015) predict which users will be banned from on-line communities. Their best system uses a Random Forest or LR classifier, with features examining the average readability and sentiment of each user's past posts, the past activity of each user (e.g., number of posts daily, proportion of posts that are replies), and the reactions of the community to the past actions of each user (e.g., up-votes, number of posts rejected). Lee et al. (2014) and Napoles et al. (2017) include similar user-specific features in classifiers intended to detect high quality on-line discussions.

Amir et al. (2016) detect sarcasm in tweets. Their best system uses a word-based Convolutional Neural Network (CNN). The feature vector produced by the CNN (representing the content of the tweet) is concatenated with the user embedding of the author, and passed on to an MLP that classifies the tweet as sarcastic or not. This method outperforms a previous state of the art sarcasm detection method (Bamman and Smith, 2015) that relies on an LR classifier with handcrafted content and user-specific features. We use an RNN instead of a CNN, and we feed the comment and user embeddings to a simpler LR layer (Eq. 2), instead of an MLP. Amir et al. discard unknown users, unlike our experiments, and consider only sarcasm, whereas moderation also involves profanity, hate speech, bullying, threats etc.

User embeddings have also been used in: conversational agents (Li et al., 2016); sentiment analysis (Chen et al., 2016); retweet prediction (Zhang et al., 2016); predicting which topics a user is likely to tweet about, the accounts a user may want to follow, and the age, gender, political affiliation of Twitter users (Benton et al., 2016).

Our previous work (Pavlopoulos et al., 2017a) also discussed how machine learning can be used in *semi-automatic* moderation, by letting moderators focus on 'difficult' comments and automatically handling comments that are easier to accept or reject. In more recent work (Pavlopoulos et al., 2017b) we also explored how an attention mechanism can be used to highlight possibly abusive words or phrases when showing 'difficult' comments to moderators.

6 Conclusions

Experimenting with a dataset of approx. 1.6M user comments from a Greek sports news portal, we explored how a state of the art RNN-based moderation method can be improved by adding user embeddings, user type embeddings, user biases, or user type biases. We observed improvements in

all cases, but user embeddings were the best.

We plan to compare *ue*RNN to CNN-based methods that employ user embeddings (Amir et al., 2016), after replacing the LR layer of *ue*RNN by an MLP to allow non-linear combinations of comment and user embeddings.

Acknowledgments

This work was funded by Google's Digital News Initiative (project ML2P, contract 362826).[9] We are grateful to Gazzetta for the data they provided. We also thank Gazzetta's moderators for their feedback, insights, and advice.

References

S. Amir, B. C. Wallace, H. Lyu, P. Carvalho, and Mario J. M. J. Silva. 2016. Modelling context with user embeddings for sarcasm detection in social media. In *Proceedings of CoNLL*, pages 167–177, Berlin, Germany.

D. Bamman and N.A. Smith. 2015. Contextualized sarcasm detection on Twitter. In *Proc. of the 9th International Conference on Web and Social Media*, pages 574–577, Oxford, UK.

A. Benton, R. Arora, and M. Dredze. 2016. Learning multiview embeddings of Twitter users. In *Proc. of ACL*, pages 14–19, Berlin, Germany.

H. Chen, M. Sun, C. Tu, Y. Lin, and Z. Liu. 2016. Neural sentiment classification with user and product attention. In *Proc. of EMNLP*, pages 1650–1659, Austin, TX.

J. Cheng, C. Danescu-Niculescu-Mizil, and J. Leskovec. 2015. Antisocial behavior in online discussion communities. In *Proc. of the International AAAI Conference on Web and Social Media*, pages 61–70, Oxford University, England.

K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *EMNLP*, pages 1724–1734, Doha, Qatar.

M. Dadvar, D. Trieschnigg, R. Ordelman, and F. de Jong. 2013. Improving cyberbullying detection with user context. In *ECIR*, pages 693–696, Moscow, Russia.

X. Glorot and Y. Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proc. of the International Conference on Artificial Intelligence and Statistics*, pages 249–256, Sardinia, Italy.

[9] See https://digitalnewsinitiative.com/.

D. P. Kingma and J. Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*, San Diego, CA.

J.-T. Lee, M.-C. Yang, and H.-C. Rim. 2014. Discovering high-quality threaded discussions in online forums. *Journal of Computer Science and Technology*, 29(3):519–531.

J. Li, M. Galley, Chris C. Brockett, G. Spithourakis, J. Gao, and B. Dolan. 2016. A persona-based neural conversation model. In *Proc. of ACL*, pages 994–1003, Berlin, Germany.

L. J. P. van der Maaten and G. E. Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

T. Mikolov, W.-t. Yih, and G. Zweig. 2013. Linguistic regularities in continuous space word representations. In *NAACL-HLT*, pages 746–751, Atlanta, GA.

C. Napoles, A. Pappu, and J. Tetreault. 2017. Automatically identifying good conversations online (yes, they do exist!). In *Proc. of the International AAAI Conference on Web and Social Media*.

J. Pavlopoulos, P. Malakasiotis, and Androutsopoulos I. 2017a. Deep learning for user comment moderation. In *Proc. of the ACL Workshop on Abusive Language Online*, Vancouver, Canada.

J. Pavlopoulos, P. Malakasiotis, and Androutsopoulos I. 2017b. Deeper attention to abusive user content moderation. In *EMNLP*, Copenhagen, Denmark.

J. Pennington, R. Socher, and C. Manning. 2014. GloVe: Global vectors for word representation. In *EMNLP*, pages 1532–1543, Doha, Qatar.

Z. Waseem and D. Hovy. 2016. Hateful symbols or hateful people? Predictive features for hate speech detection on Twitter. In *Proc. of NAACL Student Research Workshop*, pages 88–93, San Diego, CA.

E. Wulczyn, N. Thain, and L. Dixon. 2017. Ex machina: Personal attacks seen at scale. In *WWW*, pages 1391–1399, Perth, Australia.

Q. Zhang, Y. Gong, J. Wu, H. Huang, and X. Huang. 2016. Retweet prediction with attention-based deep neural network. In *Proc. of the International on Conference on Information and Knowledge Management*, pages 75–84, Indianapolis, IN.