Human-in-the-loop pipelines for Hatespeech detection

Jiří Balhar
Charles University
balhar.j@gmail.com

Tohid Ebrahim Ajdari
Technical University of Munich
tohid.ebrahim@tum.de

Adam Hrín
University of Helsinki
adam.hrin@tum.de

Abstract

While the performance gains of deep learning methods in NLP are indisputable there have been raising concerns about the non-interpretability of the models, especially when deployed in high impact settings. In our work we try to address these issues in a setting of Hatespeech detection in social media platforms. For this task we use the HateXplain dataset and a pretrained BERT model. The focus of this project is experimenting with human-in-the-loop (HITL) approaches and debiasing specific characteristics of the model, like biases towards communities. We propose three different HITL pipelines, one global and two local approaches. The global approach uses local explanation method called integrated gradients to find global features. First local approach works with the human rationales included in the HateXplain dataset, while the second one directly asks for feedback for specific samples in combination with LIME explanations. We were able to debias the jewish target group, especially with the global and the manual local methods. For some setups debiasing leads to a significant drop in performance of the model. To counter this drop we use the injection of former training data.

1 Introduction

In recent years, deep learning approaches have majorly propelled the field of natural language processing forward. Deep neural network methods are state-of-the-art for most of the NLP tasks as shown in (Otter et al., 2019) and (Torfi et al., 2021). However, deep learning models suffer the disadvantage of being black box architectures with very little interpretability power. Explaining why a deep, highly non-linear neural network makes a prediction is not trivial (Belinkov and Glass, 2019). Possible solution is to ask the model to generate an explanation alongside its prediction. This explanation could subsequently be used as valuable information for humans who could provide a feedback to the model. Humans can provide training data for re-training the model and directly accomplish tasks that are otherwise hard to achieve for the machine learning pipeline. These human-in-the-loop (HITL) approaches help tackle the problem of imperfect datasets that might be biased towards certain sub-populations or may not work effectively in the wild due to overfitting as explored in (Lertvittayakumjorn et al., 2020).

We try to solve the task of hatespeech classification with HITL approach utilizing explainability techniques. Hate speech towards various subgroups of society has become an increasingly concerning phenomenon, especially on online social media platforms. (Williams, 2019) Moreover, our hope is that we can eliminate the bias of hatespeech detection systems to prevent flagging neutral posts containing trigger words as hateful, e.g. posts and comments supporting the targeted sub-group.

2 Background and Related Work

Our work is built upon works of previous groups (Oguzhan and Utkan, 2021) and (Li and Zengin, 2021) that use a CNN based approach to solve a movie review sentiment classification task. To separate the first learning step from the re-training they use an additional mask layer that is trained in the second stage after the human feedback has been provided. However, with the advent of transformer based architectures, we have seen a shift from CNN approaches to the usage of transfer learning on pretrained language representations like BERT (Devlin et al., 2019). In (Mozafari et al., 2019) the authors investigate the BERTs ability to identify hateful context within social media content by using fine-tuning methods based on transfer learning and achieve considerable results on two Twitter datasets.

2.1 Explainability Methods

Myriad of methods for interpreting machine learning black-box models have been studied (Lertvittayakumjorn and Toni, 2021). In our work we have explored and utilized couple of approaches. First one focuses on obtaining trustable explanations with a method "Local Interpretable Model-agnostic Explanations" - LIME (Zhang et al., 2019). Another method called Integrated Gradients (Sundararajan et al., 2017) tries to attribute the prediction of a deep network to its input features. We use these methods to provide humans with global or local explanations that help them to give informed feedback.

2.2 Human-in-the-loop

In order for humans to be able to trust the model and be confident about its decisions, humans should be able to interact with the system. There are multiple ways how humans can affect the behavior of a model. As proposed in (Wu et al., 2021) that can be achieved either by affecting the data processing stage, by designing the human interaction architecture to be independent of the system, or by introducing an interventional model re-training. There has been increasing research about combining HITL and the re-training approach with various NLP frameworks to solve multiple NLP problems (Wu et al., 2021).

In the realm of hatespeech detection with HITL a few works exist, such as (Fanton et al., 2021) for generating informed and non-aggressive responses, called counter narratives, that try to combat the hate speech by undermining the impact of hateful content. In our work we aim at reducing bias in hatespeech classification task by using HITL.

3 Model

For our work we switch from convolutional architecture as used in the previous human-in-the-loop projects (Oguzhan and Utkan, 2021). Instead we employ the more modern transformer architecture BERT (Devlin et al., 2019). In this project we use the Huggingface's Transformer library (Wolf et al., 2020) with small pretrained BERT model `google/bert_uncased_L-2_H-128_A-2`. On top this BERT model we add a sequence classification head using the standard Huggingface classes.

In addition to using the standard library we also utilize AdapterHub (Pfeiffer et al., 2020) that builds on top of the Huggingface library. AdapterHub allows us to include new layers (Adapters) inside the BERT model that we use to separate the finetuning phases in our pipeline.

4 Experimental setting

In this section we explain the general structure of our experiment and go into the different retraining steps in more detail in later subsections.

Our setup consists of a base model training, followed by one of three different update methods depicted in Figure 1. These update methods are independent from each other, so they can be used in parallel, to change our model. Every of these update methods contain a step, where human influence is used in different manner, to generate a new dataset. We use these new datasets to update the parameters of our model, specifically the adapter ones, to change the predictions of our base model according to the human-in-the-loop.

4.1 Dataset

For the project we use the HateXplain dataset (Mathew et al., 2020). The authors of the paper collected data from social platforms Twitter and Gab. They used the Amazon Mechanical Turk platform to label their data as hate speech, offensive, or normal. Additionally, the annotators were also advised to specify target groups for the samples, if they contain a community as a target of hate or more generally as context. For the hate speech or offensive labeled samples, they also gave rationales to their decision, by marking text phrases, which they considered for their decision. Each sample was labeled by three annotators in this manner. For our experiment we take the one true label for a sample for training with majority vote over the three annotators. We then convert the label to binary label toxic/non-toxic, considering the labels offensive and hate speech as toxic.

The dataset consists of 15383 training samples, 1922 validation samples and 1924 test samples respectively.

4.2 Base model training

As a base model we use a pretrained BERT model with a classification head, together with a newly initialized adapter from AdapterHub. For the base model training we freeze the layers of our adapter and only train the parameters of the BERT model and the classification head. To monitor the perfor-

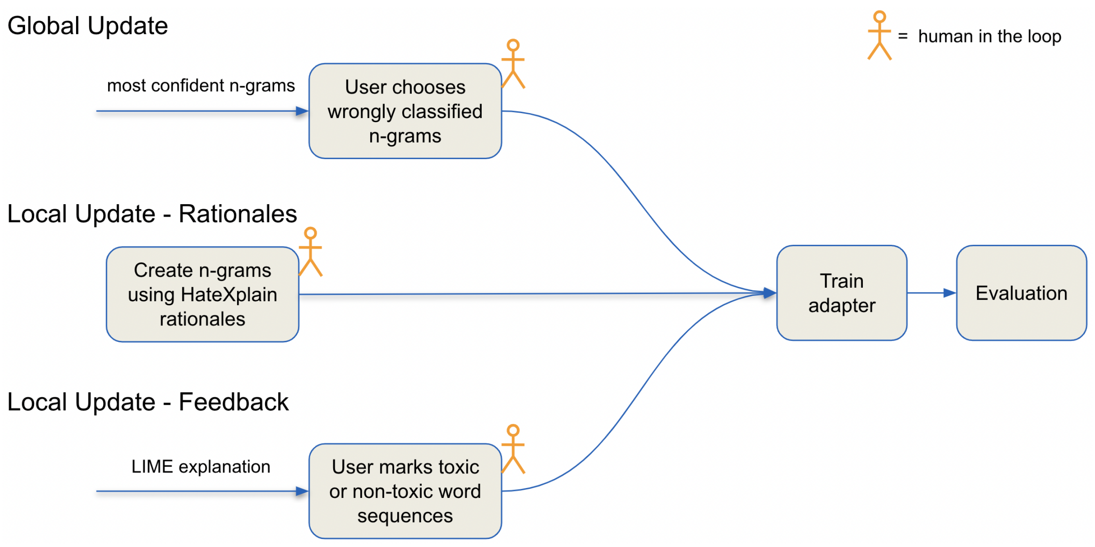
Global Update
most confident n-grams
User chooses wrongly classified n-grams
= human in the loop
Local Update - Rationales
Create n-grams using HateXplain rationales
Train adapter
Evaluation
Local Update - Feedback
LIME explanation
User marks toxic or non-toxic word sequences

Figure 1: Diagram showing the local and global update parts of our proposed pipelines. The base model training is omited from this diagram

mance in the training process and keep the best performing one, we use the validation set. For every of our updates, we unfreeze the adapter and freeze the rest of the model, to separate the efforts of the updates from the base model.

4.3 Global update

To find a way of representing a general explanation for our model, we first get explanations for every training sample by using the integrated gradients technique (Sundararajan et al., 2017).[1] This is a local explainability method for transformers, which calculates word attributions for tokens in samples, showing how the different words influence the predictions of a model.

We use the word attributions to find the most influential tokens, independent from whether they have a toxic or non-toxic effect. To build rich n-grams of variable length for the new dataset for the update, we search for other less influential words before and after the core words and combine them to create the n-grams.

The final decision, which of the n-grams will be used in the dataset, is taken by the human. In an UI, we display the 80 most toxic and 80 most non-toxic of the n-grams. We get the predictions for them from our own model. The human chooses which n-grams are wrongly classified by our model, and we use the picked n-grams as the final dataset for the global update, labeled by the opposite of the former prediction.

[1]At this point we could have used the LIME local explainability method but we chose intergrated gradients because of performance reason.

4.4 Local update

Instead of using every data sample of the training set and finding a global explanation, for the local updates we use specific samples to generate our update dataset. We have three different ways of selecting appropriate samples. First, the most confident but wrong ones, which our model assigns to the wrong class with a high probability. Second, the least confident samples, which our model predicts with a probability around 50 percent. We also tried using tf-idf to find low confidence samples, like (Li and Zengin, 2021) did in their work, but for us the results were insignificant/not good. With this algorithm we get outliers in the data with seldomly seen vocabulary. In the IMDB dataset used by (Li and Zengin, 2021) this results in reviews talking rarely reviewed films. For the HateXplain dataset, this leads to nonsense samples and samples written in languages different from English, which are not suitable to use for manual user feedback.

4.4.1 Local update - rationales

For this update, we don't need additional human feedback like in the other updates. As the human-in-the-loop mechanic here, we specifically use the human rationales already available in the HateXplain dataset.

At first, we use one of our specified sampling methods to collect a specific number of data samples from the training set. We generate the toxic samples for our new dataset out of two different ways. First, by using the rationale sequences as they are. Second, by combining the toxic sequences with non-toxic parts of the same samples, to generate longer and more natural toxic n-grams for the training.

This local update approach has the advantage over the manual method, that we can use a huge number of samples of the dataset for generating the n-grams, as the handlabeling is already done. But the lack of rationales for non-toxic samples inside the HateXplain dataset, dampens the performance of the update. We mix in neutral samples to balance the high ratio of toxic samples inside the update dataset.

4.4.2 Local update - manual

As for the other local update, we use one of our sampling methods to collect data samples from the training set. We show the full samples to a human agent directly, our human-in-the-loop approach for this update. To give the human additional information about the behavior of our model, we use LIME (Zhang et al., 2019) as a local explanation tool. The human is asked to relabel the sample based on their opinion, and also mark word sequences from the sample, that influenced their decision. In case of a non-toxic label, there are preselected words in the sequence, which our model considers as toxic influence, according to the LIME explanation. Thus, the humans give feedback about the context of the toxic-influential words, which should help the model to recognize cases in which toxic words inside a sample not automatically lead to a toxic label. We use the marked sequences from the human, together with eventually preselected words, to generate n-grams for the update dataset.

4.5 Model update tricks

Before running the local or global update on our human feedback dataset, we utilize two preprocessing techniques that improve the finetuning process. Firstly, because the resulting human feedback datasets is quite limited in size, we found that it helps to repeat the dataset 5-20 times to increase the number of training samples in one epoch. This procedure is very similar to increasing the total number of training epochs, but we found repeating the dataset more convenient. Secondly, we introduce a training stabilization technique we call *injection*. Alongside the human feedback samples, we also include some samples from the original training set. This empirically helps combat the catastrophic forgetting effect that we observe when we train on training samples with high confidence output probability.

5 Results

In this section we discuss the methodology of evaluation, report the results of the original model and then we examine the changes induced by finetuning on user feedback.

Generally, the aim of using the human-in-the-loop paradigm is not to improve accuracy. Rather we would like to uncover hidden biases of our original model using the explainability methods and then try to fix these using the human feedback. Therefore we will measure the validation and test accuracy only as a means of making sure that the model is still usable in practice and we consider slight decreases in accuracy not to be important. Our primary goals are to improve bias metrics for selected subgroups, decrease the influence of some selected input features (such as selected n-grams) or to change the output probability of some selected samples.

5.1 Methodology

To measure the impact of our proposed human-in-the-loop pipelines on the original model we use three main ways of evaluation. We measure the overall metrics to make sure that the model still has usable performance even after the human feedback changes. Namely we measure the accuracy, precision, recall, F1 score and AUROC. Using all of these metrics we can assess whether the model works for the overall distribution of the samples.

After this step, we then look at bias metrics for specific subgroups that are targets of hate-speech. We compare these metrics specifically for the groups that we have selected for "debiasing" in our experiments. Inspecting these results helps us understand whether our targeted updates have the desired impact over specific subsets of testing samples.

For more detailed inspection we also propose a visualization technique we call *sample probability plot* that builds upon a visualization proposed in Oguzhan and Utkan (2021). In this plot we measure the output probability of toxicity of selected training samples taken from the human feedback dataset. These samples differ for each pipeline since each pipeline has its own way of generating these update samples. What they have in common

Model	Sample method	Injection	Accuracy	Precision	Recall	F1 score
Original model	-	-	0.760	0.789	0.812	0.800
Global update	-	no	0.630	0.910	0.418	0.573
Global update	-	yes	0.758	0.801	0.789	0.795
Local update	HateXplain rationales	yes	0.745	0.739	0.882	0.804
Local update	most confident	no	0.257	0.348	0.287	0.315
Local update	most confident	yes	0.752	0.785	0.802	0.793
Local update	least confident	no	0.756	0.822	0.752	0.785
Local update	least confident	yes	0.749	0.797	0.776	0.786

Table 1: Qualitative overall results for all experiments

is that these samples should change its output toxicity during the second training - in the end these are the samples that we want to "debias" either towards the toxic or nontoxic side. Here we propose an improvement over the original visualization (Oguzhan and Utkan, 2021) by also adding random background samples from the original training set. The reasoning behind this is that we want to see how much the target samples change compared to all samples in the training set. In this way we want to prevent false conclusions that we could draw from a plot that doesn't include these samples. For example, if the impact of our second training is that the model learns to predict lower toxicity for all samples, we cannot say that we "debiased" our target subset because the relative change of target sample toxicity is still the same compared to other inputs. Including visualization of a random sample from the original training set helps us to avoid this false conclusion.

5.2 Biases present in the original model

The proposed Global update pipeline and Local update with manual feedback pipeline give the user an opportunity to interpret the original model. The global update pipeline shows the most salient n-gram features that the model considers toxic and non-toxic. In these lists of important features, the user is able to look for unwanted patterns and features. In the local update, the user is presented with local explanation of selected samples. In these explanations the user is again able to see what features the model uses to make its prediction.

By using the explainability steps in our proposed pipelines we have identified several possible targets for debiasing the model. In the non-toxic n-gram list we have identified reoccurring keywords "white" and "<user>". The latter is a special token used for anonymizing the HateXplain source posts. On the toxic side we have found 3 n-grams containing the word "jew" and "jews" as being highly toxic. In the local update we confirm our findings about the <user> token and the word "jew".

For demonstrating the pipelines, we have decided to mainly focus on debiasing the Jewish target group.

5.3 Global update

For the global update we select the following n-grams in the user interface: "jews", "jew" and "signs in warsaw ghetto warning the jews of typhus germans warning". We then repeat this dataset 20 times to get 60 training samples, we inject 500 random samples from the original training set and fine-tune the Adapter layers with the rest of the model frozen over 15 epochs with learning rate $3e-4$.

In this pipeline we also test the same experiment without the use of the injection.

5.4 Local update - rationales

For this local update we sample 2000 random hateful n-grams created using HateXplain rationales. To balance the training dataset, we inject 500 random positive samples from the original training set. We train on this dataset for 10 epochs with a learning rate of $3e-5$.

5.5 Local update - manual

In the manual local update, we test two sampling methods. We manually annotate 12 sentences sampled from the Jewish subset using the "most confident but misclassified" method. We obtain 32 training n-grams and we repeat the dataset 5 times to get 160 samples. Optionally we inject 1000 random original training samples. Using the second sampling method we annotate 12 least confident sentences with the Jewish target. We get 15 n-grams which we repeat to get 120 training samples.

Model	Sample method	Injection	Precision	Recall	F1 score
Original model	-	-	0.966	0.972	0.969
Global update	-	no	0.991	0.655	0.789
Global update	-	yes	0.981	0.881	0.929
Local update	HateXplain rationales	yes	0.961	0.977	0.969
Local update	most confident	no	0.826	0.107	0.189
Local update	most confident	yes	0.976	0.915	0.945
Local update	least confident	no	0.981	0.881	0.928
Local update	least confident	yes	0.988	0.910	0.947

Table 2: Qualitative results for the Jewish target

We then optionally inject 500 random training samples.

All four experiments (two sampling methods and with/without injection) are trained over 10 epochs with lr=$3e-4$.

5.6 Quantitative comparison

We report the results of our pipelines in Tables 1 and 2. From the tables we can see that there are two clear outliers in the experiments. First is global update without injection and the second is Local update (most confident) without injection. Both of these experiments are run with injection as well and there the model performance is preserved. Our conclusion is that the injection helps in these two cases. To show the effect of injection we also include the sample probability plot in Figure 2 and 3 in the Appendix. Compared to the third case without injection - Local update with least confident sampling - we can see that the use of injection is optional in this case. The only thing that is changed in this experiment compared to the other Local update experiment is the sampling method. Therefore our hypothesis is that trying to change the output probability of confident samples (that are used in the Global update and Local update with most confident sampling method) requires the injection to stabilize the training whereas changing the output probability of samples that the model is unsure about is possible even without the use of injection.

Comparing the original model and the rest of the updated models we consistently see a minor drop in general accuracy and F1 performance of up to 1.5%. Across the Jewish subset that we focus on, we can see a shift to higher precision at the expense of recall in most of the update experiments. Specifically we see the biggest improvement in precision in the global update with injection and in all local updates with the exception of the experiment "most confident + no injection" and the "HateXplain rationales" experiment. This means that for the successful experiments we were able to change the toxicity threshold for the jewish subset while preserving the general performance of the model.

6 Future work

In our work we have focused on the subgroup that we considered easiest to debias. Further work would be required to apply our proposed pipelines on the other target groups included in the dataset. Also, the adaptability of AdapterHub could be investigated further. Reusing a trained adapter for debiasing in a different task-similar model could be used as a general way of debiasing. We found that using injection empirically helps to preserve the performance of the model. We think it would be interesting to try to apply this technique in different settings and to investigate the principle behind this method more. Lastly, while working with our pipelines we frequently encountered samples that we wanted to give more detailed feedback to. For instance, in the global feedback we could add the possibility of giving more granular toxicity assesment.

7 Conclusion

We applied the idea of (Oguzhan and Utkan, 2021) to the hate speech scenario and developed own ideas for the challenge. In our experimental setup we managed to debias the jewish target group, as we showed in our results. We used different approaches to the human-in-the-loop idea, with global and local update steps, both by requesting additional human feedback and accounting it in in the retraining of the model, and by using the rationales included in the used HateXplain dataset. We can even counter the worsening of performance of our model, by using the injection of training data.

References

Yonatan Belinkov and James Glass. 2019. Analysis Methods in Neural Language Processing: A Survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Margherita Fanton, Helena Bonaldi, Serra Sinem Tekiroğlu, and Marco Guerini. 2021. Human-in-the-loop for data collection: a multi-target counter narrative dataset to fight online hate speech. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.

Piyawat Lertvittayakumjorn, Lucia Specia, and Francesca Toni. 2020. Find: Human-in-the-loop debugging deep text classifiers.

Piyawat Lertvittayakumjorn and Francesca Toni. 2021. Explanation-based human debugging of nlp models: A survey.

Qingyu Li and Ilayda Ezgi Zengin. 2021. Incorporating local update for sentiment analysis with human in the loop.

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2020. Hatexplain: A benchmark dataset for explainable hate speech detection.

Marzieh Mozafari, Reza Farahbakhsh, and Noel Crespi. 2019. A bert-based transfer learning approach for hate speech detection in online social media.

Kaan Oguzhan and Elifnaz Utkan. 2021. Suppressing word bias with human-in-the-loop. 6.

Daniel W. Otter, Julian R. Medina, and Jugal K. Kalita. 2019. A survey of the usages of deep learning in natural language processing.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. Adapterhub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020): Systems Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks.

Amirsina Torfi, Rouzbeh A. Shirvani, Yaser Keneshloo, Nader Tavaf, and Edward A. Fox. 2021. Natural language processing advancements by deep learning: A survey.

Matthew Williams. 2019. Hatred behind the screens: A report on the rise of online hate speech.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Xingjiao Wu, Luwei Xiao, Yixuan Sun, Junhang Zhang, Tianlong Ma, and Liang He. 2021. A survey of human-in-the-loop for machine learning.

Yujia Zhang, Kuangyan Song, Yiming Sun, Sarah Tan, and Madeleine Udell. 2019. "why should you trust my explanation?" understanding uncertainty in lime explanations.

A Appendix

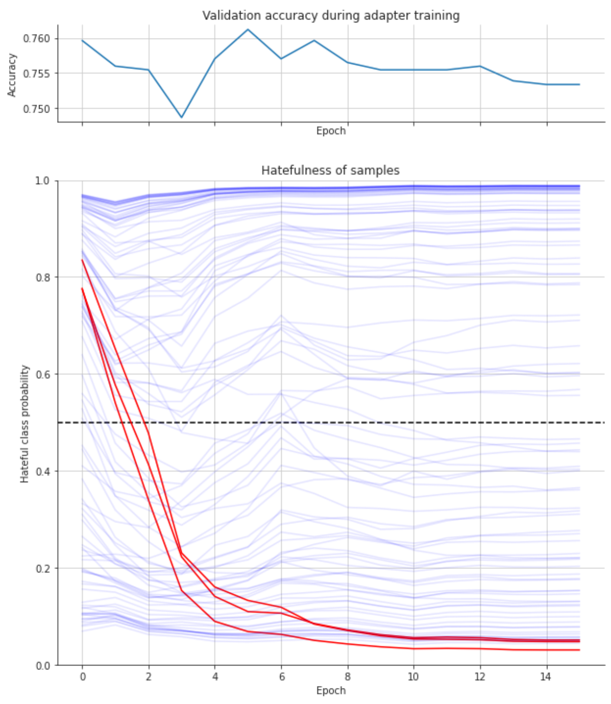

Figure 2: Prediction results for jewish update dataset with injection over epochs. Red lines depict the jewish samples that we want to debias. Blue lines depict control samples.

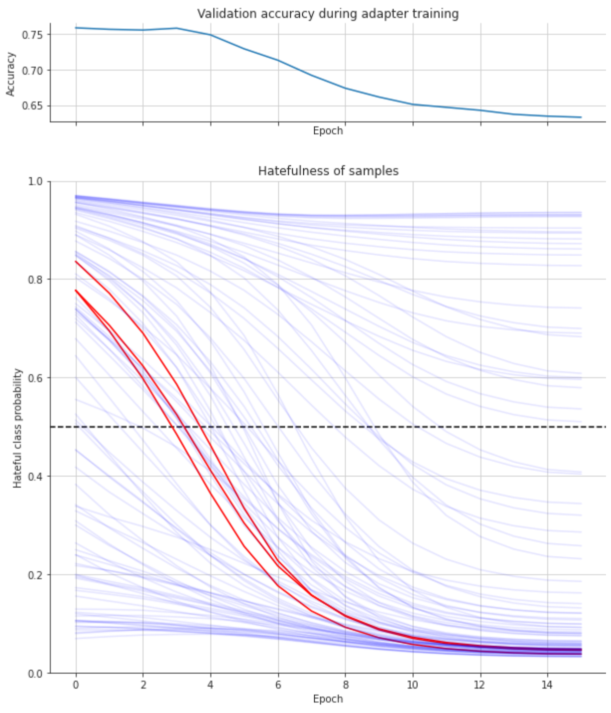

Figure 3: Prediction results for jewish update dataset without injection over epochs. Red lines depict the jewish samples that we want to debias. Blue lines depict control samples.