Suppressing word bias with Human-in-the-Loop

Kaan Oguzhan, Elifnaz Utkan
Department of Informatics, Technical University of Munich, Germany
{kaan.oguzhan,elifnaz.utkan}@tum.de

Abstract

Using Deep Neural Networks for sentiment analysis is shown great performance in the recent years by outperforming many classical methods Kim (5). Although their potential is promising, to train them perfectly one needs a perfect dataset. A perfect dataset in theory is a completely unbiased dataset that contains each and every word of a vocabulary with all possible combinations. Since getting such a dataset is practically impossible, classifiers are likely to learn some of the imperfections contained within the dataset. The resulting model therefore will contain *dataset bias*. To the extend of our knowledge deep learning models are mostly considered black box there is no direct method to make the model unlearn something. In this paper we propose Masking Layers to deal with this problem, a new type of layer that has a single purpose, dampen or enhance. This layer allows humans clear out some of the models imperfections. Experiments show that by using Masking Layers humans can force the model to dampen its faultily learned features thus suppressing the model bias to some extend.

1 Introduction

Deep neural networks have achieved remarkable performances in many natural language processing tasks. Although deep learning models become state of the art on many tasks, they are sub-symbolic learners and we don't have control over their micro knowledge due to their black-box nature. Therefore they suffer from the lack of interpretability. Explainable Artificial Intelligence (XAI) techniques are aimed at providing explanations to humans about how the decisions made by the deep learning models by providing visual representations like, Shrikumar et al. (12) Sundararajan et al. (14) Bach et al. (1) Ribeiro et al. (10). These explanations play an important role for supporting human-AI collaboration and gaining human's confidence and trust.

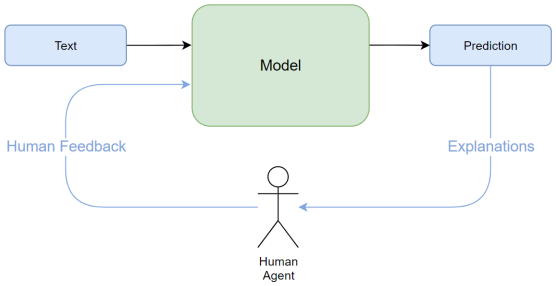

Figure 1: Model Overview

Some works focus on integrating human feedback before the training like Zhang et al. (15) demonstrates that learning with rationales can improve the accuracy of machine learning models. Furthermore Strout et al. (13) shows that models trained with human rationales also provides better explanations. Other works like Piyawat Lertvittayakumjorn (8) has focused on integrating human feedback after the model training and shown that, using the human feedback to disable the irrelevant features with the intent of reducing unintended biases in the deep learning model is promising.

Building on top of the idea of feature disabling, we found that binary feature disabling, as it is the case in Piyawat Lertvittayakumjorn (8), can be further extended by letting the network learn the dampening. The dampening will therefore be more fine grained. Also since features are learned from a very high dimensional representational space[1] it is not easy to say by disabling a feature completely exactly what information will be removed from the model. Moreover it is also not easy say that the network will learn to treat each feature completely disjoint.

Thus we tried a different approach and let the network do the disabling by training it second time on human feedback as it is visualised on Figure 1. Although one might be tempted to think about directly

[1] In our case and in (8)'s case it is GloVe Embeddings

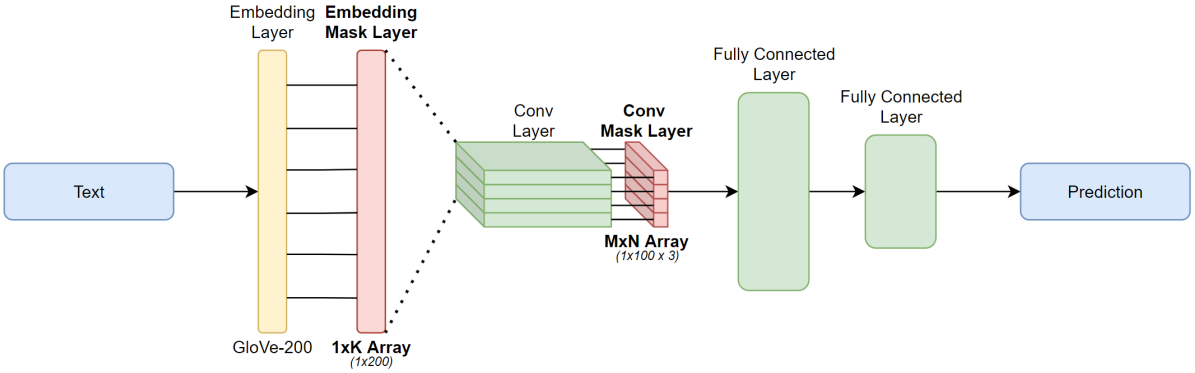

Figure 2: Model Architecture. Note that layers are color coded by the stage of the pipeline at which they will be trained. (Yellow -> Never) (Green -> Only at first stage) (Red -> Only at second stage) for more info look at Figure 3

training the network one more time and use the human feedback as the new training set. This approach will most likely ruin what has been learned by the previous training, since the dataset generated from human feedback is only a tiny fraction of the complete dataset in terms of size. Therefore we needed a method of pushing the human feedback into the network while preserving the trained parameters.

Since we don't want to remove features completely, but at the same time, given the human feedback let the network learn the human feedback, we introduced a new type of Layer, what we call as the Mask Layer. The sole purpose of the mask layer is two things dampen or enhance. The Mask Layer is an extremely small layer with learnable parameters, that sits on top of feature extracting layers such as GloVe (7) or Convolutional kernels. Unlike the classical method of doing a matrix multiplication followed by bias addition and applying a non-linearity function between layers, The Mask Layer only uses a very simple operation, element wise matrix multiplication. Parameters of the Mask layer are only element wise multiplied with the previous layer, thus the size of the mask layer is exactly same as the the output dimension of the previous layer and therefore usually extremely small.[2]

The full training loop is therefore consists of two training sections, as it will be discussed in more details on subsection 4.1. First training is for the CNN model and second training is for the mask layers.

[2]eg. a single 1x200 matrix after GloVe-200

2 Background and Related Work

In this section we will discuss background information about CNN's for text classification, explanation methods and various ways to collect feedback from humans agents.

2.1 CNN's for Text Classification

It has been shown that in text classification task convolution neural networks give promising results Kim (5) Gambäck and Sikdar (2) Zhang et al. (15). A standard deep CNN model consists of several layers. The first layer in the network is the embedding layer, that converts words into much denser, low dimensional representations. It is essentially a lookup table that maps inputs to a high dimensional embedding matrix. Then we have a convolutional layer with different filter sizes, that roll over the sentence matrix and reduce it into a low dimensional representation. In sentiment analysis task, convolutional layers are used to find and learn discriminative n-grams that will then be used for the decision of the correct class. Convolutional layers are followed by max pooling layers for further down sampling and only picking the most relevant n-grams occurrences. The pooling layer is followed by fully-connected layers for the decision. Lastly a softmax function is applied to calculate the class probability predictions.

2.2 Explanation Methods

There exist several methods used to produce local explanations. Model-specific methods are applied to only a specific class of models whereas model-agnostic methods are applicable to any model. LIME, Ribeiro et al. (10) is one successful example of the model-agnostic methods which implements

the idea of a local surrogate. LRP Bach et al. (1) and DeepLIFT Shrikumar et al. (12) are more specific and applicable to neural networks. In addition, they belong to the subgroup of the backpropagation methods which propagate from the output back to the input through the layers and assign relevance scores to every word in the input text. The other subgroup is called gradient methods and the Integrated Gradients method is a member of this subgroup Sundararajan et al. (14). Integrated Gradient methods explain the model's prediction using the gradient information.

2.3 Integrating Human in the Loop

We explored that many different ways of collecting feedback or information from humans have been proposed by recent works. In Ray et al. (9), humans interact with the system by playing a image-guessing game. They proposed this human-machine collaborative game using VQA to evaluate the efficacy of explanations. HINT Selvaraju et al. (11) gets spatial input regions which are found to be important for right decisions by human annotators and aimed to reduce the model bias for Image Captioning and Visual Question Answering (VQA) tasks. Some methods used in Human-in-the-Loop Reinforcement Learning are collecting binary feedback such as "good" or "bad" evaluation or asking humans to highlight salient regions of the visual environment state to learn what matters most to achieve the goal. (4)

For the text classification task, in FIND, Piyawat Lertvittayakumjorn (8) word clouds are generated for each feature and then there word clouds are used to evaluate the importance for predicting the correct classes. Humans are asked to decide whether the word clouds contain important n-grams for the prediction and disable the features if corresponding word clouds consist of irrelevant n-grams.

We examined this work in depth since our goals are aligned, for our task of sentiment analysis, we have discovered two weaknesses of this work. First, word clouds may contain both relevant and irrelevant n-grams, which leads to a trade-off between either keeping the feature completely or disabling and removing them completely. Such a trade-off will create ambiguity for the human agent. Second, there are only two options. Human agent can either disable the feature or keep it enabled by putting 0 or 1 in the mask layer.

Instead, with our approach, we changed the way humans are provided feedback and secondly we put a mask layer with learnable parameters, that will trained according to human feedback.

3 Model

3.1 Model Architecture

#	Type	Info	Dimensions
1	Embedding	GloVe-200	Vocab x 200
2	Mask		1 x 200
3	Convolution	1D Kernel	[3,4,5] x 100
4	Mask		3 x 100
5	Pool	Max Pool	-
6	Fully Connected		i:300 o:300
7	Fully Connected		i:300 o:150
8	Fully Connected		i:150 o:1

Table 1: The layers are ordered from top to bottom in the same order as they appear in the model

Our model architecture is very similar to classical CNN architecture as explained in subsection 2.1 What is novel in our model is the addition of Mask Layers after the Embedding and convolutional layers. A Visualization of the model can be seen in Figure 2. Also an short summary of the Layers are listed in Table 1.

Since the training is split into two separate stages, we will add *first training part*[3] and *second training part*[4] at the end of each layer description to point out at which stage of the training are the parameters of the layer getting updated.

1. Embeddings layer, this layer's task is to convert words into much denser, low dimensional representations. We used GloVe-200 Embeddings for our model. Parameters of this layer are *never updated* during any training.

2. Mask layer, this is a single 1xK matrix with no bias or activation function and all its parameters are initialized as all 1's, where K is the size of the embedding vector. Its parameters are multiplied matrix wise with the previous layer. *second training part*

3. Convolutional layer, the task of this layer is to learn seperate word relations / n-grams where

[3] The model is trained using only the IMDB Dataset

[4] The model is trained using only the Human feedback as the Dataset

N is the kernel's windows length and each feature is a seperate n-gram. In our model we use window sizes of 3,4,5 and each kernel learns 100 separate features. *first training part*

4. Mask layer, this layer is very similar to Layer-2. It is a single MxN Matrix with no bias or activation function, where M is the number of convolutional Layers from previous layer and N is the number of features extracted by each convolutional kernel from previous layer. In our model we have a 1x100x3 Matrix. *second training part*

5. Max pooling layer. *no parameter to update*

6. Fully connected layer. *first training part*

7. Fully connected layer. *first training part*

8. Fully connected layer. *first training part*

4 Experimental Setup

4.1 Pipeline

Our training pipeline consists of two main stages and two supplementary stages, they are executed in the following order: 1)Preparation stage, 2)First training stage, 3)Intermediate stage, 4)Second training stage. For visual representation of stages, refer Figure 3.

In the preparation stage we take the model described in subsection 3.1. Than initialize weights of the convolutional layers and fully connected layers using Xavier initialization Glorot and Bengio (3). Than set the weights of both Mask Layers as 1's. As described before in subsection 3.1 we use pre-trained GloVe embeddings. Also we split IMDB dataset(6) into 3 parts: Training set with 17.500 examples, validation set with 7,500 examples and test set with 25,000 examples. We use Binary Cross Entropy with Logit Loss[5] as our Loss function throughout both training stages.

In the first training stage, we freeze the weights of the Embedding Layer and Mask Layers. Note that mask layers do not make any contribution in this stage, since they are only element wise multiplications of 1's with the previous layer with no activation or bias, effectively not changing any value and keeping the gradient flow unchanged. Then we train the model over the train set for 20 Epochs. For the training we use Adam optimizer with learning rate of $4e^{-4}$ and L2 regularization with $\lambda = 5e^{-3}$. After each epoch during the training, we evaluate the model using validation set and keep the model that performs the lowest Loss throughout all epochs as the *base model*.

In the intermediate stage, we use the model trained on the first stage to generate explanations. More on how the explanations are generated will be explained in subsection 4.2. We than pick the n-grams, that are ranked highest positive and highest negative by the model, and display them to the human agent for selecting non-relevant n-grams. Than using the selected n-gram's we generate a second dataset consisting only of what is selected by the human agent. In the new dataset, selected n-grams are treated as complete sentences and labeled as the opposite classification of what the model predicts them as. We do not split the new dataset into smaller subsets and use it as single feedback set.

In the second stage of the training, except the embedding layer, we do the opposite layer freezing compared to the first stage. We freeze all convolutional and fully connected layers and unfreeze the mask layers. Than use the feedback set generated in the intermediate stage to train the model. For the training we are using Adam optimizer with learning rate $5e^{-7}$ and L2 regularization with $\lambda = 2e^{-1}$. After each Epoch during the training we evaluate the model using validation set and keep the model that performs highest accuracy throughout all epochs as the *final model*.

4.2 Human Feedback Collection

As discussed in subsection 2.2, for generating explainability feedback to the human agent, we have tried different methods, such as LIME Ribeiro et al. (10), Integrated Gradient using captum (14), LRP using iNNvestigate (1). On our experiments these methods were either slow or not capable of generating explanations for n-grams other than unigram. In order to find a method that can flexibly generate explanations for any n-grams and in a fast fashion, we decided to go as simple as possible. The simplest approach is likely just using the trained model and using it make a prediction of the subjected n-gram as a forward pass. On one hand this approach is extremely flexible as it can accept input of any length and can output a single score for the input. On the other hand, in terms of speed, it is easy to run into problems especially while using higher n-grams. For the analysis we have to enumerate

[5] PyTorch default implementation on BCEWithLogitsLoss

all possible unique n-gram pairs from the dataset. Then let the model make a prediction for each of them. This process can take a very long time for a big dataset due to pure numbers of the unique tokens. Also n-grams with high windows lengths will generate lots of unique pairs that can easily make the dataset look like a tiny fraction.

Another problem of this approach is that some n-grams, that receive very high prediction scores, might be very rare occurrences. So even if they are biased, supressing them will have tiny if not no contribution at all in terms of general unbiasing of the model. Therefore it will be more beneficial to eliminate rare words and only show frequent word to the human agent.[6]

As a solution to both problem, we have came up with the occurrence limiting. That is, during the n-gram enumeration, counting occurrences of the unique n-grams as well, than only evaluating a subgroup, that appear more than a threshold. This occurrence thresholding approach effectively solves two of our problems. 1) Thresholding reduces unique n-gram count substantially and therefore the model only has to predict a very small subgroup, hence prediction is very fast. 2) Very rare n-gram's that have very high scores will be removed.

4.3 Dataset

In this experiment we consider the task of sentiment classification into two classes (positive and negative). As mentioned on subsection 4.1 we train the model in two separate stages and therefore we use two different datasets.

For the first training stage we use the IMDB review Dataset (6) to train our model. It contains 12.500 positive and 12.500 negative highly polar movie reviews.

For the second training stage we use a dynamic dataset that is being generated from the human feedbacks as its explained in subsection 4.2. Note that this dataset is usually very small compared to the first one. For our tests, at each round we usually aim to get around ~ 75 annotations from the human agent.

5 Results

At the end of the first training stage the model achieves validation accuracy of %86.04. We will refer this model as the base model. After the first training stage as mentioned subsection 4.1 we do the second stage training and let the model learn the human feedback. During the second stage training the model is evaluated with validation set after each Epoch and the model that scores highest accuracy is picked for further evaluation, which we will refer as the final model.

For our tests on model bias, we pick some of the problematic n-gram's that are selected by the human agent. Than let the base model and final model predict scores on those n-grams and finally compare the two predictions.

In our evaluation we observed that the dampening is around %50[7], which means faulty positive and faulty negative n-grams are getting predicted %50 less intense than before. For numerical comparison refer to Table 2 and Table 3. Also we have found that around %50 suppression stays roughly same independent of the intensity of the prediction. The less positive a prediction is by the base model, the less the dampening is going to be in terms of prediction scores.[8]. It is also important to note that dampening approaches to %0 instead of going below %0 linearly, which would mean a change from positive to negative or vice versa.[9]

On top of the n-gram prediction improvements, we also observe around %0.20 increase in validation and test accuracy. The accuracy improvement is not drastic, this is partially because our validation set is just a hidden split from the same big dataset, which as we discussed earlier is inescapably biased. Therefore we assume the improvements to be higher in real world cases.

Note: Optionally one can prefer to have more dampening of biased n-grams in exchange for lower final accuracy. Therefore the best time to stop the second stage training is case dependent.

6 Discussions & Conclusions

Our model adjusts the importance of embedding and convolutional features during the training of the mask layers with learnable parameters. This boosts the predictive performance of the model, and to some extend eliminates the undesired effect of biases present in the dataset.

[6] This statement assumes that we have a limited human interaction window and we want to use it most effectively.

[7] Relative to base model's prediction score, not the score difference

[8] Same applies for negative predictions as well

[9] For demonstration of a very long training refer to Figure 4

6.1 Generalization to Other Models

We have not tested adaptation of the Mask layer to other architectures other than CNN. Some architectures that use word embeddings as their first layer should still be able to benefit from the mask layer sitting only on top of the embedding layer. While this may be possible, we did not conducted any experiment with LSTM's.

Furthermore, explainability part of our work can be used as an attempt to make the new advanced transformers interpretable.

6.2 Generalization to Other Tasks

We built our model for sentiment analysis on a review dataset, but it can be adapted for any text classification task with two classes. Nonetheless multi-class classification tasks might be challenging, since we suppressing selected n-grams by training the model with the same exact n-grams labeled as the opposite class of the first prediction. Therefore, adapting it to the multi-class classification task requires some modifications. Furthermore, our work can still be used for investigating the datasets for discovering hidden biases that cannot be seen easily.

6.3 Limitations

Since the n-grams are hand picked by human agents, receiving feedback to fine-tune the model takes time and effort. Additionally a significant change against a bias in the dataset requires lots of human effort and many repeated trainings, since for the mask layer training we set our learning rate low and do not allow sudden changes as it is a very precise update. Our primary goal is to reduce bias and overfitting, while making the predictions more coherent with human intuition without causing a decrease in performance of the classifier.

References

[1] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS ONE*, 10(7):e0130140.

[2] Björn Gambäck and Utpal Kumar Sikdar. 2017. Using convolutional neural networks to classify hate-speech. In *Proceedings of the First Workshop on Abusive Language Online*, pages 85–90, Vancouver, BC, Canada. Association for Computational Linguistics.

[3] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10). Society for Artificial Intelligence and Statistics*.

[4] Lin Guan, Mudit Verma, Sihang Guo, Ruohan Zhang, and Subbarao Kambhampati. 2020. Explanation augmented feedback in human-in-the-loop reinforcement learning.

[5] Yoon Kim. 2014. Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882.

[6] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

[7] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

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

[9] Arijit Ray, Yi Yao, Rakesh Kumar, Ajay Divakaran, and Giedrius Burachas. 2019. Can you explain that? lucid explanations help human-ai collaborative image retrieval.

[10] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 1135–1144, New York, NY, USA. Association for Computing Machinery.

[11] Ramprasaath R. Selvaraju, Stefan Lee, Yilin Shen, Hongxia Jin, Shalini Ghosh, Larry Heck, Dhruv Batra, and Devi Parikh. 2019. Taking a hint: Leveraging explanations to make vision and language models more grounded.

[12] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2019. Learning important features through propagating activation differences.

[13] Julia Strout, Ye Zhang, and Raymond J. Mooney. 2019. Do human rationales improve machine explanations?

[14] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. *CoRR*, abs/1703.01365.

[15] Ye Zhang, Iain Marshall, and Byron C. Wallace. 2016. Rationale-augmented convolutional neural networks for text classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 795–804, Austin, Texas. Association for Computational Linguistics.

A Appendix

A.1 Improvements for Positive and Negative Words

4-gram	Base Model	Final Model	Improvement
maman et la putain	0.98	0.76	22%
the incredible melting man	0.77	0.46	40%
postman always rings twice	0.52	0.25	52%
this is one film	0.51	0.23	55%
elvira mistress of the	0.46	0.22	52%
this movie on dvd	0.46	0.25	46%
the music and dance	0.42	0.20	52%
my cup of tea	0.42	0.20	52%

Table 2 – Faulty 4-grams that are predicted as **positive** by the base model. The predictions scores are ranging from 0 (Neutral) to 1 (Extremely positive) and show how intensely positive the model predicts subjected 4-gram.

4-gram	Base Model	Final Model	Improvement
crouching tiger hidden dragon	0.54	0.22	59%
nightmare on elm street	0.51	0.23	55%
ha ha ha ha	0.48	0.16	67%
the first minutes of	0.36	0.13	64%
the script for this	0.36	0.14	61%
first minutes of the	0.32	0.13	59%
until the last minutes	0.26	0.11	58%
house that dripped blood	0.23	0.10	57%

Table 3 – Faulty 4-grams that are predicted as **negative** by the base model. The predictions scores are ranging from 0 (Neutral) to 1 (Extremely negative) and show how intensely negative the model predicts subjected 4-gram.

A.2 Full Training Pipeline

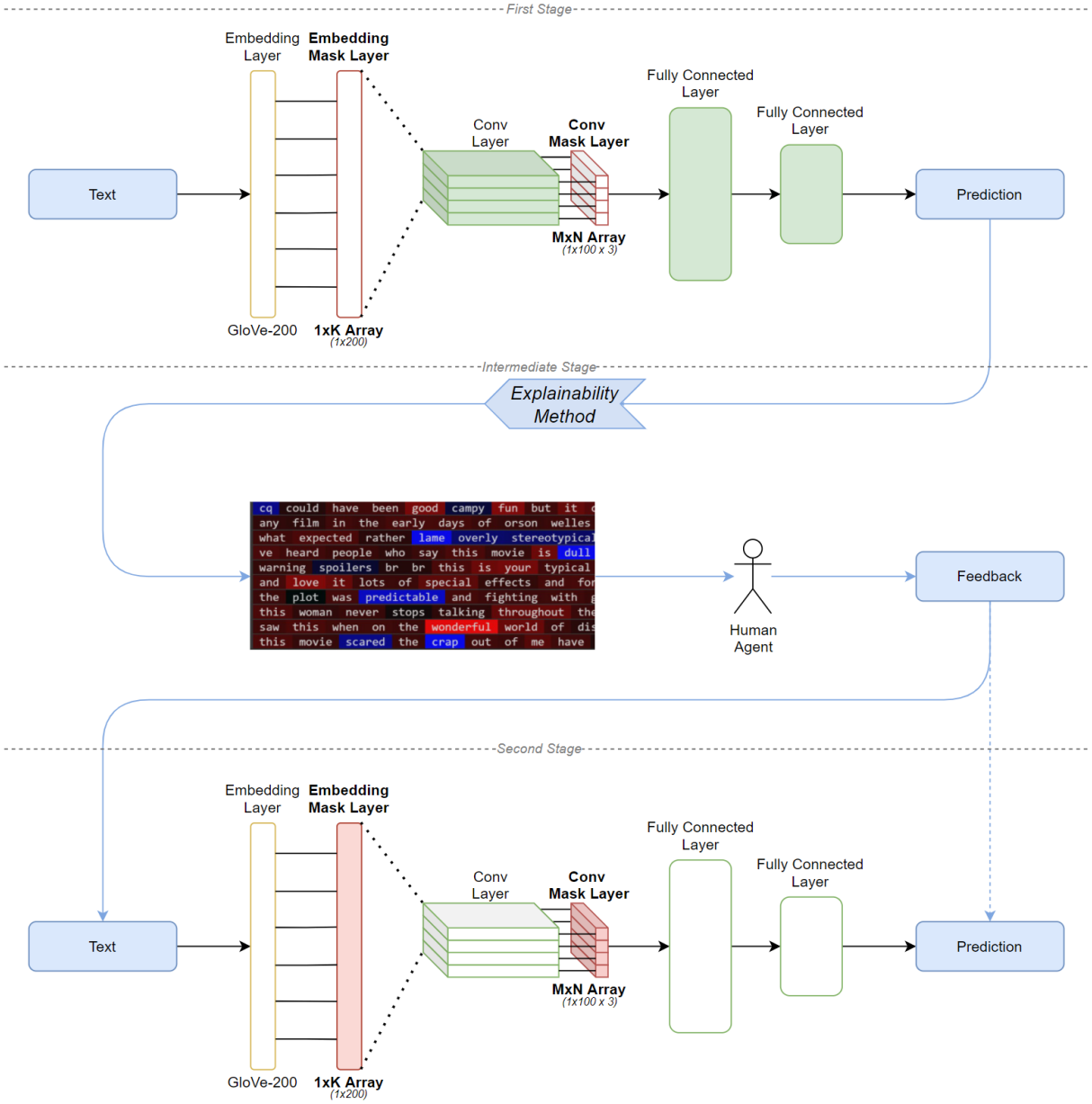

Figure 3 – Full training pipeline. Here the two main stages and the intermediate stage of the pipeline is visualized. Hollow Layers means its parameters are frozen and solid layers means their parameters will be trained by backpropagating from the prediction they produce

A.3 Extra Test for overfitting the feedback n-grams

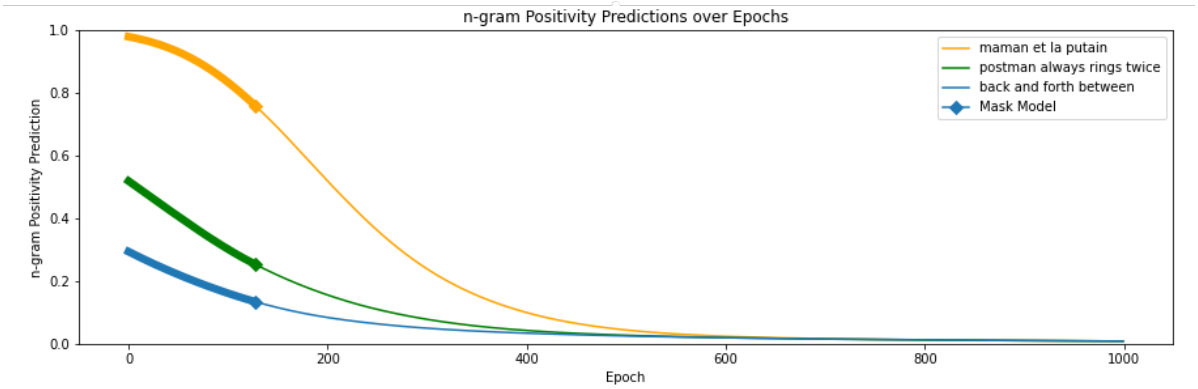

Figure 4 – Visualization of positivity dampening over 1000-Epochs. (Mask model is refers to the model that achieves highest validation accuracy)