# Enhancing Cyberbullying Detection in Social Media using Semi-supervised Learning

**Diana Ionescu**
University Politehnica of Bucharest
313 Splaiul Independetei, Bucharest, Romania
ionescudiana98@yahoo.com

**Andrei Dumitrescu**
University Politehnica of Bucharest
313 Splaiul Independetei, Bucharest, Romania
andreidumitrescu99@yahoo.com

**Traian Rebedea**
University Politehnica of Bucharest
313 Splaiul Independetei, Bucharest, Romania
traian.rebedea@cs.pub.ro

## ABSTRACT

Cyberbullying has become a usual form of harassment nowadays because most of the time we use digital technologies to communicate with others. This type of bullying can affect our mental, emotional, and also physical health[1]. Also, the significant impact of cyberbullying is that it can spread easily and quickly around the world. In most of the cases, a cyberbullying attack is discovered too late, after all the negative effects have already affected the assaulted person. The researchers tried to find an automatic way to discover a potential cyberbullying attack on social media using the power of machine learning[2] The majority of the proposed solutions are classic supervised approaches as Decision Tree, Random Forest, Support Vector Machine, Naïve Bayes, though it was tried also semi-supervised or unsupervised approaches. This paper's aim is to leverage the large amount of unlabeled data that can easily be collected and using them alongside with semi-supervised learning approaches in order to solve the task of cyberbullying detection.

### Author Keywords
Natural language processing; cyberbullying; semi-supervised learning; text classification; pseudo-labeling.

### ACM Classification Keywords
I.2.7 Natural Language Processing

DOI: 10.37789/rochi.2022.1.1.15

## INTRODUCTION

Nowadays, social media platforms are widely used by each of us from an early age. On social media, you are free to express your opinions and your thoughts. You can interact easily with other persons and the communication process is facilitated all around the world. Unfortunately, people's behavior is not always ethical, therefore the phenomenon called cyberbullying starts to spread on this type of platforms. Cyberbullying is a form of harassment that includes actions like sending, posting or sharing negative content about a person using digital devices. As it is expected, cyberbullying leads to embarrassment, humiliation and in extreme cases to depression or low self-esteem. In time, a lot of studies and articles were posted about cyberbullying and its negative effects[3]. Even if cyberbullying can affect adults, the effects can be more dramatically on children because they hide the fact that they suffered[4]. In this case, everyone is encouraged to act if they discover a case of cyberbullying and to announce the authorities.

Taking into consideration the dangers mentioned above, it is decisive to find a way to detect accurately any type of cyberbullying in an automatic and quickly way. The majority of the proposed solutions in special literature about cyberbullying detection are based on supervised approaches [6, 9]. As a general conclusion of these kind of research papers is that their big limitations are related to the insufficient amount of labeled data that implies, in most of the cases, the overfitting of the models. To create a labeled dataset can be expensive because in most of the cases requires human annotations, which also result in subjectivity. Therefore, there are a limited number of qualitative labeled datasets [5] that are available. In this scenario, semi-supervised learning models have become a potential better solution because a large amount of unlabeled data can be crawled easily from the internet using APIs created by social media applications like Twitter.

Our main purpose is to implement a semi-supervised learning model for cyberbullying detection. We want to leverage the substantial amounts of unlabeled data that can be extracted from social media. In the following chapter will be presented our first baseline models for the cyberbullying detection task. The proposed experiments

[1] Cyberbullying: What is it and how to stop it. Accessed February 12, 2021. URL: Cyberbullying: What is it and how to stop it | UNICEF Romania

[2] How AI can help fight cyberbullying, author Dean Chester. Accessed February 12, 2021. URL: How AI can help fight cyberbullying – TechTalks (bdtechtalks.com)

[3] Hinduja, S. & Patchin, J. W. (2019). Connecting Adolescent Suicide to the Severity of Bullying and Cyberbullying. Journal of School Violence, 18(3), 333-346.

[4] Hamm MP, Newton AS, Chisholm A, et al. Prevalence and Effect of Cyberbullying on Children and Young People: A Scoping Review of Social Media Studies. JAMA Pediatr. 2015;169(8):770–777.

incorporate into them different semi-supervised learning techniques that can be used to build a bigger and balanced dataset and train better different baseline architectures.

## RELATED WORK

Semi-supervised methods make use of an important amount of unlabeled data that are used for training a model alongside with a labeled dataset. In order to leverage the unlabeled dataset, there are some popular semi-supervised algorithms that are commonly used such as: self-training (known also as pseudo-labeling) [12], co-training [1] applying regularization [7] or data augmentation [13, 14].

### Data Augmentation

Data augmentation is a process in which samples from a dataset are transformed in order to obtain other similar examples but with a little deviation, noise or difference. This principle is useful for helping a model to generalize. In computer vision, standard data-augmentation methods are rotations, flipping, resizing or changing the colors. On the other hand, in Natural Language Processing tasks, data-augmentation can be quite challenging, because there are situations when you want to preserve the semantic or syntactic context of a sample. In this case, there are used methods like: back-translation [3], synonym replacements or word replacements [11]. In this context back-translation [4] refers to the process of translating a sample from its original language into an intermediary one and then back to the original one. In this way a new sample with the same meaning is obtained but in a new formulation.

A state-of-the-art algorithm for data augmentation using images is MixUp [14]. This algorithm had been later integrated in more different approaches that could be also applied on images or on text. MixUp can be seen as a function that takes as input two samples and mixes them up by applying linear interpolation on feature vectors creating virtual training examples, as it is described in equation (1).

$$\tilde{x} = \lambda x_i + (1 - \lambda)x_j$$
$$\tilde{y} = \lambda y_i + (1 - \lambda)y_j \quad \textbf{(1)}$$

In this equation, the pairs ( $x_i$, $y_i$) and ( $x_j$, $y_j$) are two random samples from the dataset, where the x values are the features and the y values are their targets. Also, $\lambda$ is a constant parameter with values between [0, 1] interval. These simple equations are based on the Vicinal Risk Minimization [2] principle, which describes a vicinity around each sample in the dataset. This principle tries to define a vicinity distribution that measures the probability of finding the virtual feature-target $(\tilde{x}, \tilde{y})$ in the neighborhood of the data samples pair. Equation (1) is a new generic vicinity distribution that was discovered by the researchers and it is called MixUp.

### Pseudo-labeling

In order to use unlabeled data alongside with labeled data for training a supervised classification model, it is necessary to assign a potential label to the unannotated data. One popular method to achieve this is the pseudo-labeling method, also known as self-training. Initially, this algorithm was developed to be used in the fine-tuning stage of a model. In the case of this method, an unlabeled sample is firstly augmented and passed through the model that we want to fine-tune. By doing this, we obtain a probability distribution over the possible classes for the augmented sample. Using the probability distribution that we obtained, we can assign a pseudo-label to the unlabeled sample. Usually, the sample will receive the most probable label from the probability distribution.

After this step, we can append the new pseudo-labeled sample to the training set. The new completed training set is used in the next training step. In the original research paper [8] the researcher proposes a loss function that we can use to train our models, which takes advantage of the unlabeled samples. This loss function is described by equation (2).

$$L = \frac{1}{n}\sum_{m=1}^{n}\sum_{i=1}^{C} L(y_i^m, f_i^m) + \alpha(t)\frac{1}{n'}\sum_{m=1}^{n'}\sum_{i=1}^{C} L(y_i'^m, f_i'^m) \quad \textbf{(2)}$$

We can observe that the loss function is made out of two operands. The first pair of sums describes the average of the loss values obtained by the labeled samples. In an equivalent way, the second operand computes the average of the loss values obtained by the unlabeled samples. This second average is scaled by a balancing coefficient, which controls the importance of the pseudo-labels. The balancing coefficient changes its value in time.

Moreover, another common approach is to define two different loss functions: the first one for the original labeled data and the second one for the pseudo-labeled data. Also, this process can be updated to be applied during the training process and before every training step.

### Co-training

The co-training method is another technique that tries to make use of the unlabeled samples by associating pseudo-labels to them. In order to achieve this, the algorithm [1] assumes that a sample can be split into two distinct views, by partitioning his set of features in two disjunct sets. By applying this process, a sample will have two different representations. This method needs a set of labeled samples and a set of unlabeled samples. The first step of the algorithm is to select a subset from the unlabeled sample set. The next step is to train two classifiers using the two distinct representations of the labeled samples. Each classifier will be trained on a different representation. After the training process has ended, the classifiers are used to predict the labels of the samples from the selected subset. The samples that are labeled with a high degree of certainty by the two classifiers are added to the final labeled set. This process is repeated for a constant number of iterations. At each new step, the subset of unlabeled samples is rebuilt.

## PROPOSED SOLUTION

Our proposed solution are two baseline models used alongside with two different semi-supervised learning techniques. As mentioned previously, in practice there are not so many well-build datasets for the Cyberbullying Detection task and moreover, they usually are not balanced, containing more samples labeled as not cyberbullying. This fact defines the problem tried to be solved by my experiments. More concretely, starting from a labeled dataset and an unlabeled dataset we want to train two different baselines for the Cyberbullying Detection task: one based on Support Vector Machines, and one based on Neural Networks. Both models will be trained to output a probability that tells how likely is for a sample to be considered as cyberbullying.

The labeled dataset will be split into three different subsets: a training set, a validation set and a testing set. The testing set will remain constant throughout all the experiments. The unlabeled dataset will be used to generate new samples labeled as cyberbullying for the labeled dataset in order to better balance it. Initially, to represent the unlabeled dataset we used another labeled one and treated it as it did not provide a labeling. The newly generated samples will be added to the training set from the other dataset. This way, we can generate a better statistic on how many false positive samples get to influence the re-training of the model. Using the new enhanced dataset, we re-train the models proposed in experiments and test if the results improve on the fixed test set. After running the experiments on these two datasets, we also decided to use the dataset generated by ourselves as a potential source of cyberbullying samples.

### Input Representation

To make possible to run the experiments, the textual data has to be represented in a way that a Machine Learning model can process it and use it to learn. As the first step in obtaining the input representation, we will represent the words as word embeddings using the Word2Vec model. The flavor that we decided to use is the one made publicly available by Google and was trained on news. In this flavor, a word vector has a length of 300. After obtaining a list with the embeddings for each word in a sample we average them to obtain an overall embedding for the overall sample. If a word does not have a representation in the Word2Vec model it will be ignored and not included in the final average.

### Datasets

During the experiments we used three datasets: two labeled dataset and a custom unlabeled collection of tweets. The second labeled dataset will be used as an unlabeled dataset for the semi-supervised learning steps. However, the annotations will be used to analyzed and to compare the further prediction in these steps.

#### *Labeled dataset*

The dataset proposed by the researchers [5] is a combination of multiple datasets which contain samples gathered from different social media contexts such as: Wikipedia Talk pages, YouTube comments, Twitter comments and posts or Kaggle. The dataset is formed of a total 16846 samples, out of which only around 5000 samples are considered cyberbullying. This means that less than one third of the dataset is actually an instance of cyberbullying, so the dataset is not balanced. Another important detail is that the collection of datasets contains diverse types of cyberbullying such as hate-speech, sexism, racism, aggression, insults, or toxicity. A sample from the dataset contain three distinctive features: the original text, the label of the sample and an annotation that tells us the type of cyberbullying reflected in the sample. During the experiments run we did not use the annotation as the current goal is just to identify the instances of cyberbullying and not to also classify it into different sub-types.

#### *Unlabeled dataset*

For representing the unlabeled dataset, we decided to use a part of the Fine-Grained Balanced Cyberbullying Dataset [10]. This dataset is built from tweets and contains multiple classes of cyberbullying. These classes are: victim's age, gender, religion, ethnicity or other. The dataset offers 8000 samples of each type of cyberbullying and additionally 8000 samples which are non-offensive. This dataset is quite atypical as it also unbalanced, but this time it contains more offensive samples than the non-offensive ones. In order to balance it out, we to use just one type of cyberbully, namely the gender one as usually it is one of the most encountered ones. Alongside with this type sd also kept the samples which are non-offensive resulting in a balanced dataset of exactly 16.000 samples. This way, the newly obtained dataset will be used as the unlabeled one in our experiments. Additionally, it will have approximately the same number of samples as the labeled one, presenting a great opportunity to complete the other one. If our models label all the samples from this dataset correctly, they could in theory balance the other dataset by adding the 8000 cyberbullying samples. One potential problem with this dataset is that it contains samples just from one class of social media context, namely Twitter. The samples are also made out of two features: the text of the tweet and the label of it.

During our experiments, we tried to check if there are duplicates between this dataset and the test set which is built from the previous dataset. We observed that there are no duplicates between them. Also, we have tried to check the similarity between the labeled test set and the unlabeled dataset. To do this, we took every sample from the test set and computed the similarity between its Word2Vec representation and all the other samples' representation from the other dataset. To compute the overall similarity, we counted all the pairs considered similar and divided by

all total possible pairs. A pair is similar if the cosine distance between the two representation is lower than 0.4. Using this procedure, we obtained that a total of 36.3% of pairs are similar. We have also applied this procedure with another threshold of 0.2 and obtained just 0.9% similar pairs.

*Unlabeled dataset*

Social media platforms expose available APIs which enable programmatic access to their content. For example, using Twitter API it can be extracted information such as: tweets (tweeter posts), direct messages, users and more other features. Using the extracted information, we can build a custom unlabeled dataset. Additionally, we can use text queries over the extracted data to filter the information that could interest us more.

In order to compute the unlabeled dataset, we used a dataset from the Twitter platform [10] to extract the most used keywords from the samples annotated as cyberbullying. The first step was to remove all the special characters and to split each message into words. We compute a dictionary with the key being the word in lower case and the value being the number of occurrences of the word. We do not take into consideration the words with length one and the stop words. The list of stop words in English is one from RANKS NL[5]. The final result is a list with over 50k words that appear in the dataset. The most majority of words have under 10 occurrences. We manually selected 200 keywords with the biggest number of occurrences and that can refer to bullying. Some examples of keywords and the number of occurrences can be found in Tabel 1.

| **Word** | **Number of occurrences** |
|---|---|
| school / schools | 8536 / 384 |
| bullied / bullies / bullying | 4621 / 1663 / 566 |
| dumb | 5381 |
| gay | 4226 |
| black | 2806 |
| f**k / f**king | 5814 / 1610 |
| n***er / n***ers / n***a | 4486 / 1335 /445 |
| muslim / muslims | 2461 / 2484 |
| idiot / idiots | 2017 / 1470 |
| hate | 1264 |
| kill / killed / killing | 457 / 351/ 298 |

**Table 1. Examples of keywords.**

[5] Stopword Lists. Accessed February 12, 2021. URL: Stopwords (ranks.nl)

We also used a second approach to find the top 200 best keywords. We want a second method that is more automated to make a comparison with the first method that implies a manual filtering. We used the module feature_selection from sklearn library. The SelectKBest model selects the features based on the k highest scores. In our case, the scores are computed with the chi-squared function. The chi-squared function can be used on non-negative, categorical features and measures the degree of association between categorical variables. To apply this approach on our data, we created the features of a sample using the bag-of-words method. The conclusion is that by selecting 200 words using the previously mentioned methods, we obtain that 99 of them are the same. Using the second method, we found more abbreviations and jargons that have an insulting meaning.

Using the keywords obtained with the first method, we compute a query using Twitter API to get the tweets that contain that specific word. From the end point response, we created a csv file with information like author id, tweet id, creation time and text. The length of the dataset is around 80k samples. By representing the samples as described previously, only a total of around 60k samples had a representation. This dataset will be used in the co-training experiments which will be described later. Using this procedure, we add to the train set a total of 6925 samples. I used these samples to compute the similarity between them and the test set. I obtained a total of 34.34% pairs that are similar using the 0.4 threshold and 0.59% similar pairs out of the total using the 0.2 threshold.

### Semi-Supervised Approaches

*Pseudo-labeling*

The pseudo-labeling technique involves using an augmentation on your data. In the case of textual data, it is not so obvious on how to apply augmentation, but there are some pre-defined methods, such as: back-translation or synonym replacement. For my experiments we chose to use the back-translation process. More concretely, the text from a sample will be translated to German using an external API and then the obtained text will be translated back to the original language, such obtaining a different representation for the same sample which should also encode its meaning. To achieve the augmentation of the text data we used an open-source toolkit from Facebook entitled “fairseq” alongside the PyTorch API. The data is initially translated from English to German and afterwards from German to English, using a temperature of 0.9. This process is applied on the raw textual data on which we did not apply any type of preprocessing.

In the experiments firstly, we will train the models on the labeled data and the trained models will be used in the pseudo-labeling process. To pseudo-label the data we applied a process which involves two predictions. We use the model to predict on the original sample and on the

augmented version of it. If both predictions produce the same output for the sample and if it is labeled as an offensive one the respective sample will be added to the training set. With the newly formed trained set we re-train the models and compare the results obtained with the previous ones.

*Co-training*

Another semi-supervised technique that we applied was co-training. In this method we have to obtain two different representations for the same sample, representations which are used to train two distinct classifiers. In my experiments we used word-embeddings to represent the input data. The principal word-embeddings used are Word2Vec ones, but to obtain a second representation for the co-training process we use GloVe word vectors. Both types of methods represent textual data as numerical vectors which are designed to also encode the meaning of each word. GloVe and Word2Vec are trained differently so they produce distinct vectors for the same word, ensuring us with two different representations. Moreover, the length of the vectors produced by the two methods differ one from the other. The flavor of GloVe word embeddings used produces vectors of length 100. Similarly, to the experiments run with pseudo-labeling, I firstly train two different models similar in architecture on the labeled data. Each of the classifiers is trained on a different representation of data. The trained models are used to predict labels on the unlabeled dataset and if they predict the same label for the same sample and the label is cyberbullying the sample is added to the training set. Lastly, the models are re-trained, and the results obtained on the test set are compared to the previous ones.

**Implementation Details**

The initial labeled dataset is split into 2 parts: training set and validation set. The splitting is done in such a way that both splits contain the same percentage of cyberbullying samples relative to the total samples in a split. The testing set contains 30% out of the total samples. For the experiments that involve Neural Networks the dataset is actually split into a third part which will be used for validation. This part will contain 10% out of the total samples. The testing set remains fixed for all the experiments in order to ensure a fair comparison between experiments.

For the Support Vector Machine model, we used the classical implementation from the Scikit-Learn API, namely the SVC classifier. The important hyper-parameters used for this model are: a radial basis function kernel, the regularization parameter is set to 1.0 and tolerance (the stopping criterion) is set to 1e-3.

For the implementation of the Neural Network based model we used the TensorFlow API. The architecture contains an Input Layer, a hidden Dense Layer with 50 units and a Rectified Linear Unit activation function and a final Dense Layer with 2 units and a Softmax activation function. The input of the model is of shape: (Batch Size, Word Vectors Length). The output of the model is of shape: (Batch Size, 2). Each feature from the output represents a probability, the first one is the probability to be a non-offensive sample and the second one is the probability to be a cyber-bullying sample. The Softmax activation function is needed in order to map the input features of the layer to a probability distribution. The optimizer used for training is the Adam Optimizer with a learning rate set to 1e-5. The optimization loss function is Categorical Cross-entropy. These models are trained for a total of 100 epochs using a batch size of 32 samples. The training samples are shuffled at every iteration. In the training process a validation set is used. The model which performs the best regarding the loss value on the validation set is saved and used later for evaluation.

**RESULTS**

The we have run several types of experiments in order to better analyze the effectiveness of the proposed baselines. The first type of experiments that we run was to train a model on the labeled dataset, use the trained model alongside a semi-supervised technique to produce new cyberbullying samples and re-train the model on the newly obtained dataset. This way we validate that the models can learn on the original dataset and that the newly added samples improve the results. Both types of models will be used alongside with both methods of semi-supervised learning mentioned before. The second type of experiments try to repeat the process described above to see how long the results will improve if we constantly enhance the dataset. Lastly, we experimented with different thresholds for the models to consider a sample as cyberbullying. By doing this, we want to see what effect has on the re-training process including in the dataset less samples. The samples added should be more qualitative as by increasing the threshold less false positives should reach into the new dataset. For all these experiments the following paragraphs will present the results obtained.

**Pseudo-Labeling Experiments' Results**

In the original labeled dataset, there were a total of 16846 samples, which after splitting resulted in 11792 total samples in the training subset and 5054 total samples in the testing set. After using Word2Vec model to represent the input the number of samples dropped to 11565 samples in the training set (out of which approximately 32.35% are cyberbullying samples) and respectively 4982 samples in the test set (out of which approximately 32.15% are labeled as offensive). Additionally, for the Neural Network based experiments the training set was further split into two subsets, one that will be used for validation. For this case, the training set will have 9917 samples and the validation set 1650 samples (the percentages of the offensive samples are 32.34%, respectively 32.34%).

| | SVM Model | NN Model |
|---|---|---|
| Macro F1-Score before dataset enhancement | 76.96% | 73.53% |
| Macro F1-Score after dataset enhancement | **77.72%** | **74.55%** |
| Total Added Samples | 4999 | 4328 |
| Total True Cyberbullying Added Samples | 4705 | 3928 |
| False Positives Added Samples | 294 | 400 |

**Table 2.** ***Results using Pseudo-Labeling on the labeled test set***

Firstly, we would like to present the results obtained by the Support Vector Machine model on the original dataset and on the enhanced dataset after 1 iteration of pseudo-labeling. To the original dataset were added a total of 4483 samples, out of which 244 were actually labeled wrongly as abusive samples when in fact they were non-offensive samples. Table 2 presents the scores obtained by the *SVM* on the test set after being trained on the original and enhanced dataset. As it can be seen, the Accuracy score remains constant, but the Macro F1-Score is increased with ~1% after re-training with the new samples.

A similar experiment was run for the Neural Network model. An identical outcome can be observed for this model also, the only difference being that the results are a bit worse compared to the ones obtained by the SVM baseline. This time the original dataset is enhanced with a total of 4328 samples, out of which 400 are false positives.

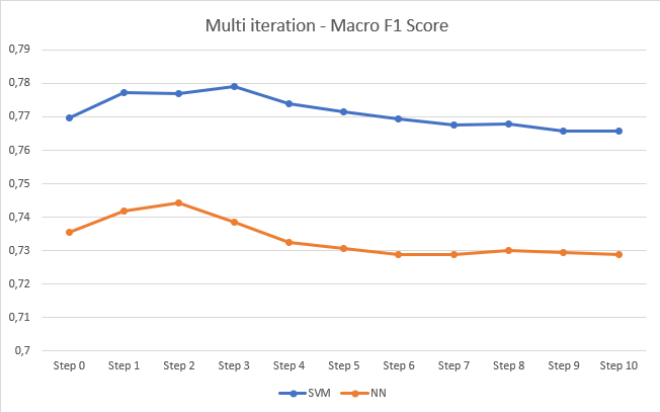


**Figure 1. Macro F1-Score Evolution Comparison**

The next experiments that we have run with both models were to repeat the process described previously and see how the macro F1-Score would evolve and how many new samples will be added from one iteration to another. One important aspect is that the datasets do not remove old added samples. These means that the dataset obtained at iteration "t + 1" will include the reunion between the dataset used in iteration "t" with the newly annotated samples. Therefore, the training set size will increase at each iteration, but the size of unlabeled set will decrease. The behavior is plotted in Figure 1 and Figure 2.

In both graphs the Ox axis represents the number of iterations on which the experiments were run. In the former graph the Oy axis represents the evolution of the macro F1-Score obtained by retraining the model on the newly enhanced datasets. In the second graph the Oy axis represents the number of newly added samples to the training set. As it can be seen, repeating the experiment multiple times will initially improve results on both models and in time the newly added samples will converge to 0. Moreover, the macro F1-Score will start to decrease in value with more iterations run.

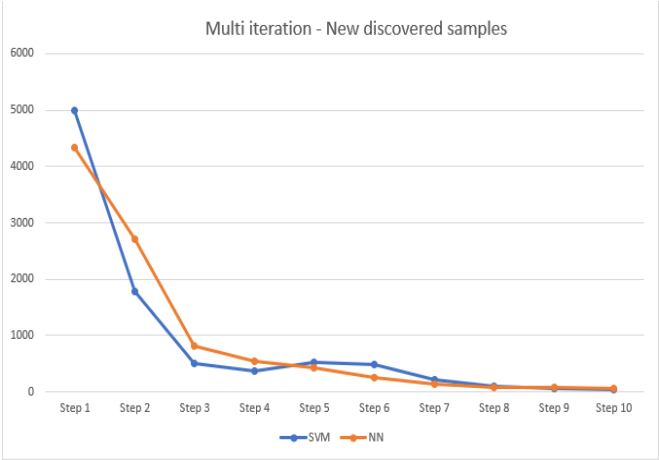


**Figure 2. Newly Discovered Samples Comparison**

This could be explained by the fact that the model starts to add multiple false positives in the training set which will decrease the quality of the dataset over time.

The last type of experiments that we ran with the pseudo-labeling technique was to see what effect has a more rigorously samples' selection process. We decided to select the samples which have a probability of being offensive bigger than a given threshold. During the experiments we increased the threshold value, and we compared the results obtained. The observations were made with only 1 iteration of enhancing the training set and only for the Neural Network based model.

| | *Threshold* | | |
|---|---|---|---|
| | 50% | 60% | 70% |
| Macro F1-Score before dataset enhancement | 73.53% | 73.53% | 73.53% |
| Macro F1-Score after dataset enhancement | **74.55%** | **73.79%** | **73.79%** |
| Total Added Samples | 4328 | 2680 | 1256 |
| Total True Cyberbullying Added Samples | 3928 | 2482 | 1181 |
| False Positives Added Samples | 400 | 198 | 75 |

**Table 3. NN Results using Pseudo-Labeling and Threshold Variation**

In Table 3 there are presented the results obtained. As expected, fewer new samples make it to the dataset. The results still improves but not so much as in the case of the default 50% threshold. More importantly, the percentage of added false positives decreases with an increased threshold. This could assure us in obtaining a more qualitative dataset.

### Co-training Experiments' Results

For the co-training-based experiments we would like to firstly present a similar statistic about the dataset splitting samples when we use GloVe word vectors. The testing set will have a total of 4994 samples (out of which 32.07% are offensive samples) and the training set will have 11614 samples (out of which 32.22% are offensive samples). For the Neural Network based experiments the training set will contain 9957 samples (out of which 32.22% are offensive samples) and the validation set will contain 1659 samples (out of which 32.24% are offensive samples). An important detail is that after pseudo-labeling unlabeled samples we will add in the new train set only the samples that have a representation for both Word2Vec and GloVe flavors. This rule is applied for both the SVM and NN based models.

Similar experiments were run with the co-training technique. In Table 4 are presented the results obtained by both models on the original dataset and on the enhanced dataset after 1 iteration. Moreover, the table includes a statistic about how many new samples were added to the dataset. As it can be seen from the table, the GloVe flavor performs a bit worse than the Word2Vec flavor in the case of both types of models. The results improve for all models with around ~1% in terms of Macro F1-Score after the retraining is done. Similarly, to the pseudo-labeling experiments, the Support Vector Machine model performs better than the Neural Network model.

| | *SVM Model* | | *NN Model* | |
|---|---|---|---|---|
| | ***Word 2Vec*** | ***Glove*** | ***Word 2Vec*** | ***Glove*** |
| Macro F1-Score before dataset enhancement | 76.96 % | 74.57 % | 73.53 % | 70.83 % |
| Macro F1-Score after dataset enhancement | **77.58 %** | **75.36 %** | **74.39 %** | **73.02 %** |
| Total Added Samples | 4483 | | 4174 | |
| Total True Cyberbullying Added Samples | 4239 | | 3839 | |
| False Positives Added Samples | 244 | | 335 | |

**Table 4. Results obtained using co-training by all models**

Also, we have run the experiment with the increasing threshold for the co-training technique on the Neural Network based model. The results obtained are similar to the ones obtained in the pseudo-labeling experiments. The results are presented in Table 5 for 3 different threshold values and both flavors of the model. It seems that in the case of co-training increasing the threshold filters out better than in the case of pseudo-labeling the number of false positives that are added to the new dataset. Also, the GloVe variation still performs worse than the Word2Vec variation. This observation could be explained by the fact that a word vector in the context of GloVe has fewer features than the Word2Vec embeddings.

### Results Obtained on Custom Dataset

As a final experiment, we wanted to evaluate how our custom build dataset would affect the training experiments. Observing that in general the Support Vector Machine based model performed the best on all the experiments we decided to use it to run this experiment. Also, we have used just the co-training technique, as both semi-supervised learning methods performed similarly, and the back-translation process takes considerably more time to complete. Moreover, for this experiment we decided to also add non-cyberbullying samples in the training set. This way, we could replicate a scenario more appropriate to a real-life situation where we want both types of samples. In order to still balance better the dataset, we have added all the found cyberbullying samples and smaller number (70% out of total number of found offensive samples) of non-offensive samples.

| | ***Threshold*** | | | | | |
|---|---|---|---|---|---|---|
| | ***50%*** | | ***60%*** | | ***70%*** | |
| | ***W2Vec*** | ***GloVe*** | ***W2Vec*** | ***GloVe*** | ***W2Vec*** | ***GloVe*** |
| F1-Score before dataset enhancement | 73.53% | 70.83% | 73.53% | 70.83% | 73.53% | 70.83% |
| F1-Score after dataset enhancement | **74.39%** | **73.02%** | **73.94%** | **72.95%** | **73.64%** | **72.07%** |
| Total Added Samples | 4174 | | 2450 | | 1085 | |
| Total True Cyberbullying Added Samples | 3839 | | 2297 | | 1021 | |
| False Positives Added Samples | 335 | | 153 | | 64 | |

**Table 5. NN Results using co-training and Threshold Variation**

In Table 6 the results are presented. The results also improve by using this custom dataset. An interesting observation is that the models add fewer offensive samples relative to the true size of the dataset. This can be explained by the fact that our hard-coded unlabeled dataset was balanced. More than likely, in this custom build dataset most extracted samples aren't cyberbullying, reflecting a real-life scenario where most of the interactions are not offensive ones.

| | ***Word2Vec*** | ***Glove*** |
|---|---|---|
| Macro F1-Score before dataset enhancement | 76.96% | 74.57% |
| Macro F1-Score after dataset enhancement | **77.35%** | **74.63%** |
| Total Added Samples | 6925 | |
| Total Offensive Added | 4074 | |

**Table 6. Result obtained by the SVM using co-training and custom build dataset**

## CONCLUSION AND FUTURE WORK

In conclusion, in this step of the project we presented a proof-of-concept for semi-supervised approaches in the context of Cyberbullying Detection. From all the presented experiments, we can conclude that semi-supervised learning represents a powerful mechanism for increasing the performance of even some simple baseline models like a Support Vector Machine Classifier or a small Neural Network. We used pseudo-labeling with back-translation as an augmentation process and co-training algorithms to generate more data. With the new labeled data, we can increase the training set size and also to balance it. After the re-training of the baseline models on the new training set, we achieve an increase of the Macro F1-Score with 1% - 3%. We used a labeled dataset as an unlabeled dataset for these experiments in order to observe the behavior of the models on a new collection of data.

Regarding to the new unlabeled data extracted from Twitter, we observed that in average the percentage of extracted cyberbullying messages is around 6-7%, based on how the Support Vector Machine baseline performed on it. Therefore, we will continue to collect data in order to have a sufficient number of cyberbullying messages to significantly increase the training set and to have an equilibrium between the two classes. Also, as further work, we will continue the experiments with more powerful models and other semi-supervised approaches such MixText.

## REFERENCES


1. Blum, A., and Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. Proceedings of the eleventh annual conference on Computational learning theory, (pp. 92-100).
2. Chapelle, O., Weston, J., Bottou, L., and Vapnik, V. (2000). Vicinal risk minimization. Advances in neural information processing systems, 13.
3. Chen, J., Wu, Y., and Yang, D. (2020). Semi-supervised models via data augmentation for classifying interactive affective responses. arXiv preprint arXiv:2004.10972.
4. Edunov, S., Ott, M., Auli, M., and Grangier, D. (2018). Understanding back-translation at scale. arXiv preprint arXiv:1808.09381.
5. Elsafoury, F., Katsigiannis, S., Pervez, Z., and Ramzan, N. (2021). When the timeline meets the pipeline: A survey on automated cyberbullying detection. IEEE Access, 9, (pp. 103541-103563).
6. Islam, M. M., Uddin, M. A., Islam, L., Akter, A., Sharmin, S., and Acharjee, U. K. (2020). Cyberbullying detection on social networks using machine learning approaches. IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE), (pp. 1-6).

7. Laine, S., and Aila, T. (2016). Temporal ensembling for semi-supervised learning. arXiv preprint arXiv:1610.02242.
8. Lee, D. H. (2013). Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. Workshop on challenges in representation learning, ICML, (p. 896).
9. Van Hee, C., Jacobs, G., Emmery, C., Desmet, B., Lefever, E., Verhoeven, B., & Hoste, V. (2018). Automatic detection of cyberbullying in social media text. PloS one, (p. 13.10).
10. Wang, J., Fu, K., & Lu, C. (2020). SOSNet: A Graph Convolutional Network Approach to Fine-Grained Cyberbullying Detection. 2020 IEEE International Conference on Big Data (Big Data), (pp. 1699-1708).
11. Wei, J., and Zou, K. (2019). Eda: Easy data augmentation techniques for boosting performance on text classification tasks. arXiv preprint arXiv:1901.11196.
12. Xie, Q., Dai, Z., Hovy, E., Luong, T., and Le, Q. (2020). Unsupervised data augmentation for consistency training. Advances in Neural Information Processing Systems, 33, (pp. 6256-6268).
13. Xie, Q., Luong, M. T., Hovy, E., and Le, Q. V. (2020). Self-training with noisy student improves imagenet classification. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, (pp. 10687-10698).
14. Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. (2017). mixup: Beyond empirical risk minimization. arXiv preprint arXiv:1710.094