Applied Artificial Intelligence
An International Journal

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/uaai20

Semi-Supervised Self-Training of Hate and Offensive Speech from Social Media

Safa Alsafari & Samira Sadaoui

To cite this article: Safa Alsafari & Samira Sadaoui (2021) Semi-Supervised Self-Training of Hate and Offensive Speech from Social Media, Applied Artificial Intelligence, 35:15, 1621-1645, DOI: 10.1080/08839514.2021.1988443

To link to this article: https://doi.org/10.1080/08839514.2021.1988443

APPLIED ARTIFICIAL INTELLIGENCE
2021, VOL. 35, NO. 15, 1621–1645
https://doi.org/10.1080/08839514.2021.1988443

Taylor & Francis
Taylor & Francis Group

RESEARCH ARTICLE

Semi-Supervised Self-Training of Hate and Offensive Speech from Social Media

Safa Alsafari [a,b] and Samira Sadaoui [a]

[a]Computer Science Department, University of Regina, Regina, SK, Canada; [b]Computer Science and Engineering Department, University of Jeddah, Saudi Arabia

ABSTRACT

Improving Offensive and Hate Speech (OHS) classifiers' performances requires a large, confidently labeled textual training dataset. Our study devises a semi-supervised classification approach with self-training to leverage the abundant social media content and develop a robust OHS classifier. The classifier is self-trained iteratively using the most confidently predicted labels obtained from an unlabeled Twitter corpus of 5 million tweets. Hence, we produce the largest supervised Arabic OHS dataset. To this end, we first select the best classifier to conduct the semi-supervised learning by assessing multiple heterogeneous pairs of text vectorization algorithms (such as N-Grams, World2Vec Skip-Gram, AraBert and DistilBert) and machine learning algorithms (such as SVM, CNN and BiLSTM). Then, based on the best text classifier, we perform six groups of experiments to demonstrate our approach's feasibility and efficacy based on several self-training iterations.

ARTICLE HISTORY
Received 6 March 2021
Revised 24 September 2021
Accepted 27 September 2021

Introduction

Research Scope

Monitoring and detecting Offensive and Hateful Speech (OHS) on social media are of crucial importance. Good progress was made for OHS detection for the English language (Ousidhoum et al. 2019; Zampieri et al. , 2019) and Arabic as well (Albadi, Kurdi, and Mishra 2018; Alsafari, Sadaoui, and Mouhoub 2020c; Mubarak and Darwish 2017; Mubarak et al. 2020; Mulki et al. 2019). Across all the inspected languages, past studies on OHS classification explored supervised machine learning, relying on the availability of annotated corpora. Particularly, in the Arabic setting, the training datasets, which are few in number and small-sized, are insufficient to build efficient OHS classifiers. The goal of our research is to take advantage of the abundant amount of content on social media, like Arabic Twitter, to develop a robust OHS classification model. Nevertheless, hand annotating the large volume of texts is a tedious task that necessitates experts in the spoken language, as demonstrated in the previous research on the

CONTACT Safa Alsafari sba166@uregina.ca Computer Science Department, Regina, University of Regina, Sk, Canada; Samira Sadaoui sadaouis@uregina.ca

supervised classification for the Arabic language (Alsafari, Sadaoui, and Mouhoub 2020a,). For leveraging the many available textual data, we investigate Semi-Supervised Self-Training (named SSST), starting with a small labeled dataset (supervised learning) and a massive unlabeled dataset (unsupervised learning). We select SSST due to its efficacy (Li and Liang 2019; Li et al. 2020) because it may achieve superior performance than supervised learning and requires much fewer data to be annotated, hence saving much human effort, time and cost (Zhu and Goldberg 2009).

Moreover, some studies showed that semi-supervised classifiers' performance might decrease in specific applications, and one possible cause is that labeled and unlabeled data have different distributions (a mismatch between classes) (van Engelen and Hoos 2019), which is common in image classification. However, this is not the case in text classification because no new classes will appear in the unlabeled data as we are dealing with only two possible classes, "Clean" and "Offensive/Hate." Lastly, semi-supervised learning must be safe in the sense that it should not significantly lower the predictive performance after using the unlabeled data, which means that the generalization accuracy is not statistically significantly worse (Li and Liang 2019). In the OHS detection field, only one research (Rosenthal et al. 2020) built a large-scale English OHS corpus where the most confident pseudo-labeled data were chosen based on the predictions returned from four models (like LSTM and BERT), but without any re-training iterations.

Contributions

Our primary objective is to induce an efficient Arabic OHS classifier by enhancing its learning quality through the SSST iterations. The success of self-training lies in choosing only trusted data from the unlabeled dataset (van Engelen and Hoos 2019; Zhu and Goldberg 2009). At each iteration, the SSST approach must employ only highly confident predictions for all the classes to avoid learning from unreliable and noisy data (van Engelen and Hoos 2019; Zhu and Goldberg 2009). Our study is the first time the SSST process is carried out for many iterations and also presents in each iteration detailed results about the data (unlabeled, pseudo-labeled, and training) and predictive performances. Our contributions to the OHS classification field are the following:

(1) Devise an SSST process based on a large-scale unlabeled corpus of 5 million Arabic tweets. The OHS classifier gradually teaches itself by considering human-annotated data (called SEED) and confident pseudo-labeled data obtained from unlabeled data. We construct the SEED dataset by fusing two recent labeled and evaluated OHS Arabic datasets, called here OHS1 (Alsafari, Sadaoui, and Mouhoub 2020c) and OHS2 (Mubarak et al. 2020).

(2) Build a new supervised Arabic OHS dataset with pseudo labels, obtained after many SSST iterations. This dataset is the largest one for Arabic OHS detection. Since past Arabic datasets are of small sizes, we intend to make this new corpus available for researchers to validate new learning algorithms and word-embedding models.
(3) Perform six groups of experiments to validate our SSST approach. To fairly evaluate all the produced classifiers, we utilize the same testing dataset to report on the predictive performances:

- The first experiment trains and compares several combinations of Machine Learning Algorithms (MLAs) and Text Vectorization Algorithms (TVAs) to choose the best baseline classifier for the SSST process. As MLAs, we choose SVM, CNN and BiLTSM. TVAs are of different types, including (a) a combination of character and word N-Grams and (b) three pre-trained word-embeddings: the recent contextualized AraBert and the non-contextualized W2V Skip-gram and DistilBert. We note that we pre-trained the Skip-gram network on a large textual corpus (19.4 million data) that we collected from several data sources.
- The second experiment demonstrates the feasibility of the iterative self-training approach for text classification. The goal is to improve the selected classifier's performance progressively through many SSST iterations (a total of 15).
- The third experiment assesses the quality of the new supervised OHS dataset using several heterogeneous text classifiers that we train only on the confidently pseudo-labeled data obtained from the sizable unlabeled dataset.
- The fourth experiment examines the impact of imbalanced data on the SSST accuracy. For this purpose, we gradually reduce the class distribution ratio of the supervised dataset over the learning iterations until the two classes become equally distributed.
- The fifth experiment explores the SSST efficiency in the context of low-resource labeled data by varying the proportions of the SEED dataset.
- The sixth experiment investigates ensemble-based self-training, using Maximum Voting and Average Voting strategies and two text classifiers, to select confident pseudo-labeled data at each re-training iteration.

We structure the paper as follows. Section 2 reviews representative studies on self-training approaches. Section 3 builds the SEED training and testing datasets as well as the large-scale unlabeled dataset scraped from Twitter. This section also compares the distributions of labeled and unlabeled datasets. Section 4 describes in detail our SSST framework. Sections 5, 6, 7, 8 and 9

conduct diverse experiments to demonstrate the effectiveness of our self-training approach. Section 10 summarizes our findings and highlights some research directions.

Related Work

Lately, researchers have become interested in exploring self-labeled methods to address the scarcity problem of labeled data, such as self-training approaches. This section reviews such approaches that employed single classifiers to conduct self-training.

Based on standard MLAs, such as SVM, KNN and CART, the work (Wu et al. 2018) developed an SSST method that utilizes a clustering method called "density peaks" to determine the distribution structure of the training dataset. For increasing the classifiers' performance, the authors incorporated the identified structure into the SSST process that has only two iterations. They evaluated the SSST framework, which is of the cluster assumption type, using two synthetics and two actual datasets. Overall, the proposed framework surpasses two semi-supervised learning methods: (1) semi-supervised tri-training based on co-training and (2) semi-supervised FCM based on self-training, for three out of the four tested datasets. However, as noted by the authors, the framework is less suitable for overlapping datasets, where it is more challenging to learn the structure of data space.

The study (Li et al. 2020) defined an SSST approach using on the "local cores" concept to tackle the adequacy and scarcity of labeled data, well-known problems in machine learning. The authors solved the problem of the insufficient initial labeled dataset by looking for the local cores in the unlabeled dataset. The authors utilized local cores to show the data distribution of both spherical and non-spherical data, and their labels are predicted via co-training or active labeling. Then, the local cores are utilized to augment the labeled dataset. Next, two base classifiers of SVM and KNN are trained on the augmented labeled dataset over one learning iteration. Using several UCI datasets, the experiments showed that the proposed method is superior to several other self-labeled methods.

In the context of speech recognition, the research (Kahn, Lee, and Hannun 2020) devised an SSST approach based on the encoder-decoder with attention model. It comprises several phases: 1) training a robust acoustic model using a small paired dataset, 2) fitting a language model with a large-scale text dataset to produce the pseudo-labels, 3) adopting two filtering techniques to remove noisy pseudo-labels, and 4) training an ensemble of acoustic models to augment pseudo-label diversity. Based on a paired and unpaired speech recognition corpus with clean and noisy settings, the experiments with single and ensembles models showed that

the SSST approach performance is much improved than a baseline model trained on only the paired dataset, over three iterations. In this work, ensembles of five and four models outperformed the single model with a clean and noisy setting, respectively. Later on, another work (Xu et al. 2020) explored the fusion of the SSST defined in (Kahn, Lee, and Hannun 2020) with unsupervised pre-training to take advantage of unlabeled audio data. The authors experimented with the combined approach using two benchmark datasets and attained a high performance. They concluded that the two approaches complement each other for speech recognition.

For image classification, the study (Nartey et al. 2020) proposed to use an "easy-to-hard" self-training approach based on CNN to leverage unlabeled data by pseudo labeling them and then adding the most confident examples to the labeled dataset. The image classifier was then trained using the expanded dataset. The most confident pseudo-labeled samples were selected based on a confidence threshold, and the authors experimented with three threshold settings, top 5%, top 10%, and top 20%. The proposed SSST method obtained higher accuracy than fine-tuning over two standard and three coarse image datasets. It also outperformed three supervised approaches on five out of the six datasets. According to the experiments, the best confidence threshold for pseudo-labeled selection is 10%.

In another study (Xie et al. 2020), the authors utilized self-training for image classification based on the teacher-student paradigm. The authors trained a teacher model using EfficientNets as the baseline classifier and the imageNet as the training dataset. They used the model to produce the pseudo-labeled data for 300 million unlabeled images. Both labeled and pseudo-labeled images noised via dropout and data augmentation are then employed to train the student model, which will be used as a teacher model in the next iteration. After three iterations, the model reached an accuracy of 88.4% with an improvement of 2.0% over the state-of-the-art.

For object detection and segmentation, (Zoph et al. 2020) examined the effect of self-training in comparison with pre-training. For self-training, the authors employed the teacher-student model called EfficientNet-B7 and for pre-training, the RetinaObject detection model. The authors found that, unlike pre-training that can lower the performance when combined with stronger augmentation or high data regime, self-training improves the accuracy across dataset sizes and augmentation regimes. The experimentation was done with one training round. Furthermore, they highlighted that self-training was better than pre-training in terms of flexibility and scalability.

The research (He et al. 2020) examined the effectiveness of noisy SSST specifically for neural sequence generation tasks under low and high resource settings. Using the "Base Transformer Architecture." The authors observed

that self-training with noisy inputs generated via perturbation and dropout for both machine translation and text summarization tasks is more effective than regular self-training.

Lastly, in the domain of OHS detection, to benefit from the vast content of posts on Twitter, the study (Rosenthal et al. 2020) developed a large-scale supervised English dataset using an ensemble of four single classifiers. The authors trained the four classifiers, Bert, PMI, LSTM and FastText, based on a small labeled dataset: Then, the most confidently classified positive examples of the unlabeled dataset are retained. The confident data are the aggregation of the confidences obtained by the four models. The new labeled dataset increased the predictive performance compared to the original dataset. The authors also thoroughly examined easy and challenging examples.

Training and Testing Offensive/Hate Speech Datasets

Recently, the work (Alsafari, Sadaoui, and Mouhoub 2020c) constructed a robust OHS corpus written in the top two Arabic languages: (1) Modern Standard Arabic, which is understandable by all Arabic speakers, and (2) the Gulf Arabic dialect, which is spoken by the countries of the Arabian Peninsula. For collecting the data, the authors queried the Twitter platform using four searching strategies: keyword-based, hashtag-based, profile-based, and defensive-based. After cleaning the scraped data, they obtained 5340 tweets that were annotated using a rigorous labeling process (Alsafari, Sadaoui, and Mouhoub 2020c):

- **Clean**: tweets that do not contain any offensive and hateful language, such as profanity, insults, threats, and swear words.
- **Offensive/Hateful**: tweets that attack or threaten individuals or groups based on their protected characteristics, including religion, race, gender, ethnicity, and nationality.

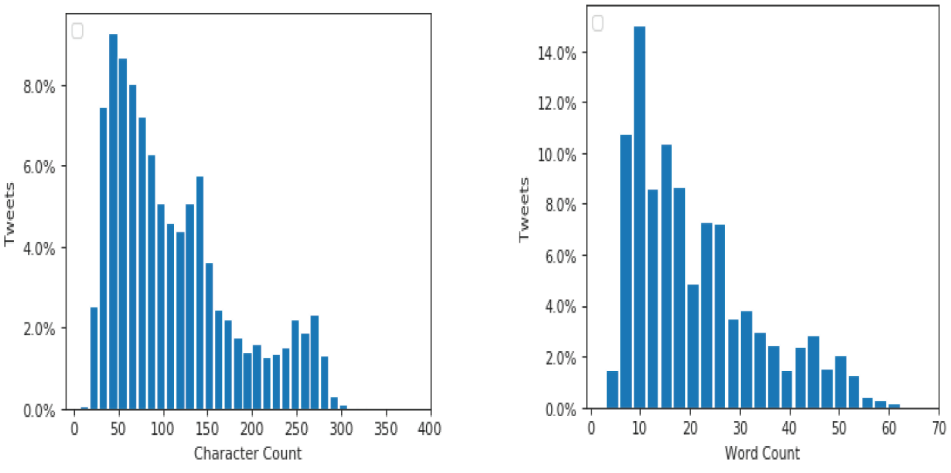
Furthermore, the three studies (Alsafari, Sadaoui, and Mouhoub 2020c, 2020a, 2020b) employed successfully this Arabic corpus (called here OHS1) to evaluate and compare the performances of several OHS classification models based on supervised and ensemble learning. One issue of deteriorating the SSST accuracy is an insufficient initial training dataset (Li et al. 2020). Hence, the first training dataset should be sufficiently large to build a strong learning hypothesis from the beginning. Consequently, in addition to OHS1, we utilize another high-quality annotated OHS Arabic dataset (called here OHS2) developed recently by (Mubarak et al. 2020) so that our target classifier possesses sufficient and adequate data to train on initially. OHS2 contains 8,000 tweets

labeled as Clean or Offensive. The annotation is compatible with the guidelines of the competition OffensEval2019, and around 19% of posts were considered offensive/hate. This corpus was employed as a benchmark Arabic dataset by the competition OffensEval-2020 for the multilingual offensive language classification (Zampieri et al., 2020). We preprocess both OHS1 and OHS2 as follows: (1) normalize English and Arabic numbers by replacing them with "@number," (2) normalize elongated words by eliminating repetition of three or more characters, (3) normalize hashtags by deleting underscores and the symbol #, (4) normalize the three Arabic letters, Alef, Alef Maqsoura and Ta Marbouta, and lastly (5) remove non-Arabic characters, diacritics, punctuation, emojis, users' mentions, and stop words.

As exposed in Table 1, we divide both OHS1 and OHS2 datasets into 70% training and 30% testing data using the stratified splitting method. After that, we combine OHS1 and OHS2 training data to produce the SEED dataset that we utilize to develop the initial classifier. The SEED dataset has a class distribution ratio of Clean to Offensive/Hate (O/H) equals 3:1, which is acceptable. So, no need to re-balance the SEED dataset. Moreover, we build the testing dataset by merging OHS1 and OHS2 test data. In each SSST iteration, we use this new dataset to assess

Table 1. Training and testing OHS datasets.

Dataset	Size	Training Data	Testing Data
OHS1	5340	3738	1602
OHS2	8000	5600	2400
Combined	13140	9338 (SEED)	4002



((a)) Character Counts ((b)) Word Counts

Figure 1. Data distribution of labeled SEED dataset.

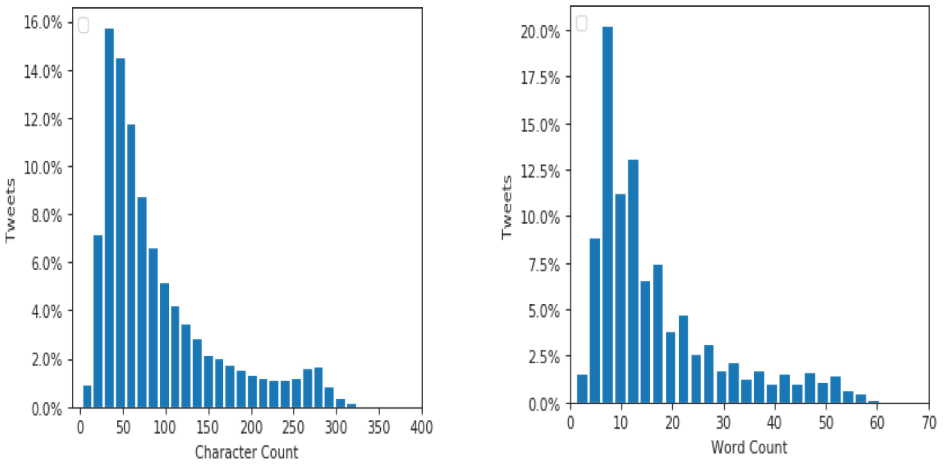
fairly the re-trained OHS classifiers' performance. The distribution of the SEED dataset based on character and word counts is illustrated in Figure 1. As we can see the majority of the tweets are short texts with less than 60 words and 300 characters.

Large-scale Unlabeled Corpus

Our objective is to gather representative social media posts. For this purpose, we employ Arabic Twitter as the main platform to collect our textual corpus. To scrape diverse, realistic, and unbiased data, we use generic keywords. In particular, we use 12 Arabic prepositional keywords, given below:

From:فـي In:فـي; On: عل‌ى; To: الـى; But: لـكن; All:: كـل; And: و .

Our approach motivation is to avoid potential dialectical or search bias in the crawled data, which is common in most available hate speech datasets (Ousidhoum, Song, and Yeung 2020). Our approach should provide an unbiased dataset that will allow us to train a robust Arabic OHS detector with a good generalization capability. Using the previously mentioned keywords, we extract a tally of 8 million public tweets randomly through the Twitter API. Subsequently, we preprocess the entire corpus and remove duplicated, short (less than three words), and non-Arabic tweets. Furthermore, to increase dataset lexical divergent and to mitigate redundancies, using the Jaccard similarity metric, we delete similar tweets that exceed a similarity threshold of 80%. After cleaning the corpus, we ended up with 5 million reliable tweets. We normalize the whole corpus as done previously



((a)) Character Counts ((b)) Word Counts

Figure 2. Data distribution of unlabeled dataset (5 million tweets).

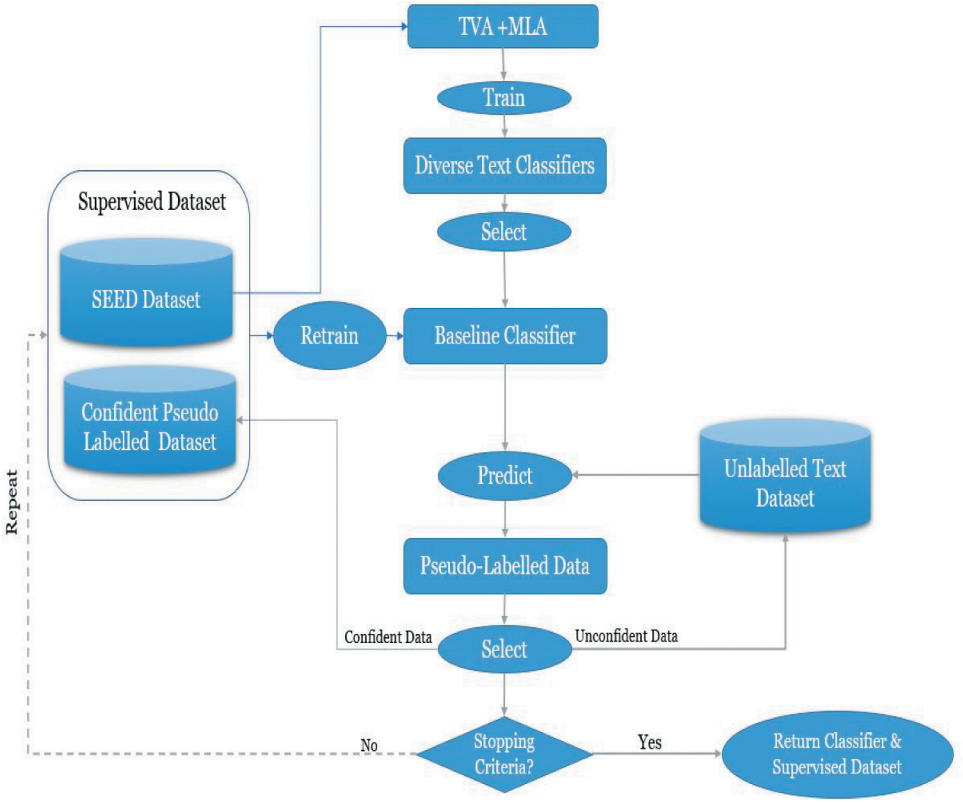


Figure 3. Semi-supervised self-training for OHS classification.

for the SEED dataset. As depicted in Figure 2, most of the tweets in this unlabeled corpus are short texts and have a similar distribution to the tweets in the SEED dataset, with less than 60 words and 300 characters.

Semi-Supervised Self-Training for OHS Detection

There are two primary types of semi-supervised classification (van Engelen and Hoos 2019; Zhu and Goldberg 2009): (1) transductive learning that only predicts the classes for an unlabeled sub-dataset using the labeled data, and (2) inductive learning (known as the true semi-supervised classification) that uses pseudo-labeled data to build a classifier to be used for future predictions. Our approach is of the inductive type that adopts self-training so that the classifier teaches itself from its own predictions. One benefit of self-training is that it is a wrapper method since we can employ any MLA, standard or advanced, to conduct the SSST process over several iterations (Zhu and Goldberg 2009).

Figure 3 illustrates the SSST framework to construct a robust text classification model, based on three reliable datasets

- The SEED training corpus of 9338 instances,
- The large-scale unlabeled corpus of 5 million instances,
- The testing corpus of 4002 instances.

After several self-training iterations, the SSST approach returns:

- A large-scale supervised Arabic dataset, which other researchers can utilize since in the Arabic setting, the very few datasets are of small sizes.
- An improved OHS classifier as it is trained on much more data. The classifier improves its predictive performance over the training iterations.

Algorithm 1: Semi-Supervised Self-Training for OHS Detection
Inputs: seedDataset, unlabeledDataset, testDataset, MLAs, TVAs, confidenceLevel, nc (number of classifiers)
Outputs: base classifier (improved), supervisedDataset (enlarged)
begin
//Select Optimal Baseline Classifier
1: candidateClassifiers = ∅
2: **for** c = 1 to nc **do**
2.1: $classifier_c$ = train(TVA+MLA, seedDataset)
2.2: candidateClassifiers = addClassifier(candidateClassifiers, $classifier_c$)
3: evaluatePerformances(candidateClassifiers, testDataset)
4: baseClassifier = selectBestClassifier(candidateClassifiers)
//Conduct SSST Iterations
5: supervisedDataset = seedDataset
6: **do**
6.1: pseudoLabeledDataset = predictLabels(baseClassifier, unlabeledDataset)
//Build confident pseudo-label sub-dataset
6.2: confSubset = selectTrustedData(pseudoLabeledDataset, confidenceLevel)
6.3: supervisedDataset = addData(supervisedDataset, confSubset)
6.4: shuffle(supervisedDataset)
6.5: baseClassifier = train(baseClassifier, supervisedDataset)
6.6: evaluatePerformance(baseClassifier, testDataset)
//Remove high confident predicted data
6.7: unlabeledDataset = removeTrustedData(unlabeledDataset, confSubset)
while *(predictionProbabilities > = confidenceLevel) and (unlabeledDataset* ≠∅);

In Algorithm 1, we present the general SSST framework that comprises three main phases described below. In the experiments, we first select trusted pseudo-labeled data based on a confidence threshold of the probability scores of both classes. Next, based on this threshold, we choose trusted data by addressing the imbalanced class distribution and adopting ensemble learning in each iteration.

A. Select best baseline classifier for the SSST process: for selecting the best classifier for the self-training process, we train and assess multiple heterogeneous OHS classifiers based on standard and deep learning algorithms (called MLAs). Before training any MLA, we first transform the textual data into numerical vectors using Text Vectorization Algorithms (called TVAs), such as word embeddings. Word embedding is a mechanism that maps a word into a fixed and real-valued vector to capture the semantic and syntactic information of the word. It converts each word into an m*n matrix where m is the sequence length of the text and n is the embedding dimension. The text classifiers take advantage of word embeddings to extract discriminative and effective features. Word embedding initializes the weights of the input layer of the deep neural networks, and its quality significantly impacts the learners' performance.

In the first experiment, we train several pairs of TVAs (such as N-Grams, Word2Vec Skip-Gram, AraBert and DistilBert) and MLAs (such as SVM, CNN and BiLTSM) using the SEED training dataset. We then evaluate the performances of the obtained classifiers using the same testing dataset. The configurations of the chosen MLAs and TVAs are explained in detail in Section 6. Finally, we choose the pair that optimally learns the OHS detection rules as the baseline classifier for conducting the SSST iterations.

B. Build confidently labeled sub-dataset at each iteration: we employ the optimal classifier (TVA+MLA) to label the unlabeled dataset artificially at each iteration. Subsequently, we retain only the highly confident predictions of both classes, Clean and Offensive/Hate, for the next learning step. We keep the prediction probabilities that are more than the threshold of 0.999 to reflect a high trust in the data. This data selection will help make the SSST safer (Li and Liang 2019) because wrongly predicted classes may mislead the SSST process. Lastly, we retain the least confident instances in the unlabeled dataset to be re-classified in the next iteration with less misclassification error.

C. Fine-tune baseline classifier at each iteration using the new supervised dataset: we re-train the inductive classifier on the expanded training dataset: the confident pseudo-labeled sub-dataset together with the previous training dataset. We shuffle the new supervised dataset before re-training. For a fair performance comparison, we assess each newly trained model using the same testing dataset. We keep iterating the two phases B and C until no prediction probabilities of 0.999 are found for both classes or the unlabeled dataset has been entirely processed.

Experiment#1: Baseline Classifier Selection

For choosing the appropriate baseline classifier for the entire SSST process, we first develop multiple heterogeneous OHS models: a standard classifier based on N-Grams and two deep neural-network classifiers based on three word

embeddings. We pre-train the non-contextualized word embedding Word2Vec SkipGram (W2VSG) and utilize two recent pre-trained contextualized word embeddings: AraBert and DistilBert. Using the SEED training dataset, we train seven OHS classifiers: SVM with Word/Character N-Grams, CNN and BiLSTM with W2VSG, CNN and BiLSTM with AraBert, CNN and BiLSTM with DistilBert. In the following, we describe the configurations of the machine learning algorithms and text vectorization techniques.

Text Vectorization Configurations

Word/Character N-Grams (WCNG)

We combine word-ngrams with character-ngrams. This combination returns more representative features by using the power of word-ngrams with the morphological insight of the lower-range char-ngrams. We experimentally analyzed several ranges of char-ngrams and word-ngrams. The combination of (1–5) char-ngrams and (1–3) word-ngrams yielded the best results.

Our Pre-trained Non-contextualized Word Embedding

We locally trained W2VSG network with an ensemble of Wikipedia dump of three million Arabic sentences: a tweet dataset of 6.5 million Arabic tweets, and an united nation dataset of 9.9 million Arabic sentences. Actually, we extracted the tweets specifically to build a corpus that is closely related to the hate speech domain. The resulting dataset comprises 19.4 million sentences with 0.5 billion tokens. We trained the word embedding with this vast unlabeled textual corpus, based on the hyper-parameter values proved efficient in several studies (Mikolov et al. 2013): 300 for the dimension size of the word vector and 5 for the size of the window context. We assign 5 to the minimum word count, which leads to a vocabulary with a size of 1.1 million words.

Existing Pre-trained Contextualized Word Embedding

As a contextualized word embedding, we employ AraBert and (Antoun, Baly, and Hajj 2020) DistilBert(Sanh et al. 2019). AraBert is a new pre-trained Arabic language embedding based on the bidirectional transformer architecture devlin2019bert. It was trained on a sizable Arabic dataset containing 70 million sentences, a vocabulary size of 64 K tokens, and an embedding dimension of 768. AraBert was reported to achieve high accuracy in three NLP tasks using eight datasets (Antoun, Baly, and Hajj 2020).

The multilingual DistilBert is a smaller and faster version of Bert. In (Sanh et al. 2019) was trained on a collection of Wikipedia consisting of 104 different languages. DistilBert was reported to retain 97% of the capabilities of language understanding, and was 60% faster and 40% smaller than Bert.

Classifier Configurations

We train three classifiers, SVM, CNN, and BiLSTM, and tune their hyper-parameters using the random search. For SVM, we tune both the penalty parameter C and the kernel parameter σ. For deep neural networks, we first fix the random seed and random search over the number of layers, units, batches and epochs, dropout rate, and learning rate. The early stopping criterion is based on the validation loss. However, the instability of deep neural models as a function of the random initialization, mini-batch ordering, and non-determinism in the computation platform can result in a significant variation in the performances (Bhojanapalli et al. 2021; Madhyastha and Jain 2019; Xia et al. 2020). To reduce the effect of randomness in deep learning models, using the selected hyper-parameters, we train each of the classifiers 150 times using different random seed and choose the most performing model based on the validation set. We train the text classifiers with the Adam optimizer using the P100 Cloud GPU. The architectures of the three classifiers are described below.

Support Vector Machine (SVM)

SVM, a traditional classification method, was adopted successfully in numerous text classification tasks (Borrajo, Romero, and Iglesias 2015; Liu, Bi, and Fan 2017). Based on the margin maximization, with linear or nonlinear kernel functions, SVM finds a hyper-plane that separates the target classes' contents. For this study, we employ the Radial Basis Function (RBF) kernel and L1 regularization.

Convolutional Neural Network (CNN)

This deep network was successfully adopted for object and text recognition (Georgakopoulos et al. 2018; Hughes et al. 2017; Wang et al. 2016). The standard CNN consists of convolutional, pooling and fully-connected layers. For text classification, the convolutional layer acts as N-gram feature extractor. First, the input sequence is converted to a 2-D matrix using an embedding layer. Second, the convolutional layer followed by dropout and max pooling layers will transom the embedding matrix into one-dimensional vector. The final layer outputs the probability distributions over the target classes. In our work, we adopt a CNN with 250 filters, a kernel size of 2, a stride of 1, and a dropout rate of 0.2. All the hidden layers use ReLU activation function, while the dense layer uses Sigmoid function along with the binary cross-entropy loss.

Recurrent Neural Network (Bilstm)

This network type is best suited for sequential data such as speech and language (Graves, Mohamed, and Hinton 2013; Lee and Dernoncourt 2016). It is constituted by stacking layers sequentially. The first step is to convert the input sequence to its vector representation using an embedding layer. The

first-word vector is fed to a hidden layer that produces a hidden vector to the second layer. The latter processes this vector with the second word's vector representation to output the second hidden vector. This process repeats until the end of the sequence is reached. We use BiLSTM, a variation of recurrent neural network, with one hundred hidden units, equal to the maximum sequence length. BiLSTM consists of a forward pass to process the sequence from left to right and a backward pass for the opposite direction processing. A fully connected layer processes the output of both forward and backward passes using the Sigmoid function and cross-entropy loss. The hidden layer uses Relu function with a dropout rate of 0.2.

Heterogeneous Classifier Evaluation

In our SSST experiment, for each iteration, we retrain the base classifier with the newly built supervised dataset and then use it to predict the labels of the vast unlabeled text corpus. For selecting a model for self-training, the prediction accuracy is one of the essential aspects to consider. Other aspects are the model size and inference speed for one instance (mini-seconds). We assess each of the seven classifiers with the three model aspects: Accuracy, Size and Inference speed. For the accuracy evaluation, we adopt the F-macro, Precision and Recall, which are more suitable for unbalanced datasets.

As observed, CNN with W2VSG yields the best outcome across the three accuracy metrics. In terms of size and inference speed, SVM+WCNG is the best model. However, this model is the least performing across the classification metrics. When comparing deep classifiers, we can see that CNN+W2VSG is more efficient in terms of inference speed and size. The time gap between SVM +WCNG and CNN+W2VSG is only 16.8 ms. Thus, considering the three model aspects, CNN+W2VSG is the best overall model and consequently we select it for our SSST experiments. This model is the best in terms of accuracy, which is a crucial factor for the success of the self-training process. We also note that contextualized word embeddings, AraBert and DistilBert, are slow and generate large size models, which are not suitable for the semi-supervised learning context because of the many iterations and the large size of the training datasets.

Experiment#2: SSST Process Evaluation

This section thoroughly evaluates our SSST process using the SEED dataset and the vast unlabeled dataset. First, we train the CNN+W2VSG classifier using the SEED dataset and then use the learned model to classify the whole unlabeled dataset. Only highly confident pseudo-labeled examples with a probability greater or equal to 0.999 will be added to the training dataset to produce the next classifier. At each SSST iteration, Table 3 presents the unlabeled dataset, confidently self-labeled sub-dataset, new training dataset, and count of Clean (C)

Table 2. Selecting optimal baseline classifier for SSST.

Model	Precision	Recall	F1-Score	Model Size (MB)	Inference Speed (ms)
SVM+WCNG	80.89	87.17	83.32	**29**	**21.5**
CNN+W2VSG	**87.69**	**89.60**	**88.59**	55	38.3
BiLSTM+W2VSG	86.68	89.49	87.95	58	91.2
CNN+AraBert	86.57	88.51	87.47	544	68.7
BiLSTM+AraBert	87.14	87.87	87.50	544	71.8
CNN+DistilBert	76.40	71.07	71.76	541	68.5
BiLSTM+DistilBert	85.74	87.16	86.41	542	70.1

Table 3. Augmenting training dataset at each SSST iteration.

Iteration	Unlabeled Data	Confident Labeled Data	Training Data
Initial	NA	NA	9338
1	5000000	4457	13795
2	4995543	1227	15022
3	4994316	557	15579
4	4993759	498	16077
5	4993261	1569	17646
6	4991692	949	18595
7	4990743	852	19447
8	4989891	4672	24119
9	4985219	437	24556
10	4984782	1570	26126
11	4983212	13804	39930
12	4969408	3971	43901
13	4965437	3137	47061
14	4962277	4562	51623
15	4957715	1154	52777

to Offensive/Hate (OH) instances. We conduct a large number of SSST iterations compared to previous studies, even though we are dealing with complex classification models. As shown in the related work section, most studies tried only one iteration, with a maximum of three iterations, which are not enough to properly assess the iterative self-training process.

Since the prediction probabilities' confidence level is very high for both classes (0.999), the training dataset size increases slowly but safely. In addition to the initial phase, we conduct 15 self-training iterations to produce a new OHS corpus of 52,777 labeled instances, the largest supervised dataset known in the Arabic context. We stop at 15 iterations because it takes a considerable amount of time training the classifier 150 times for one single iteration to solve the randomness issue. Since our unlabeled corpus is still large, we can keep iterating if we want to build a more extensive training dataset.

We can observe that the ratio of offensive/hate to clean is increasing across the iterations. A manual analysis of the confident labeled data suggests that this imbalanced class ratio can be due to the dominance of a single offensive hashtag that was trending during the unlabeled corpus collection. After few iterations, the classifier was confident to label instances with this specific hashtag as offensive/hate. In most iterations, the imbalanced class ratio is acceptable, and the accuracy is very satisfactory. Also, the size of the selected confident labeled

dataset fluctuates across the iterations due to changes in training data. Moreover, adding correctly pseudo labeled data to the training dataset increases the classifier labeling confidence, increases the size of the confident labeled data, and adds noisy predicted labels to the data. Therefore, the classifier labeling confidence and the size of the confident labeled data are reduced.

Table 4 exposes the size of each extended training dataset and the predictive performance at each iteration, Precision, Recall and F1-score.

The accuracy is high from the initial phase, with an F-score of 88.59% and the best F1-Score of 89.60%. Overall, Precision and F1-score are increasing gradually using the self-labeled examples, with 1.01% improvement in F1-score and 1.51% in Precision over the baseline performance. However, the Recall values fluctuate slightly over the iterations due to the noisiness in the pseudo-labeled examples. The best recall of 90.73% is obtained in iteration #13, with an improvement of 1.13% over the initial Recall.

Experiment#3: Pseudo-labeled Data Quality Evaluation

To check the SSST approach efficacy, we train and compare the five classifiers on the SEED dataset (human-annotated) and the self-labeled dataset (machine-annotated). The latter is obtained by discarding the SEED data from the training dataset obtained in Table 4 at the last iteration, consisting of 43439 (52777–9338) instances. We then assess all the classifiers' performance using the same test dataset, as presented in Table 5. The self-labeled dataset is of high quality as the F1-score ranges from 87.81% to 81.77%. The highest accuracy is returned with the classifier CNN+W2VSG, and the lowest outcome by BiLSTM +AraBert. We can also conclude that the pseudo-labeled dataset has a similar quality to the human-annotated dataset. For example, the gap between SEED-based CNN+W2CSG and self-labeled based CNN+W2VSG is only 0.78%.

Table 4. Predictive performance at each iteration.

Iteration	Training Data	Precision	Recall	F1-Score	FNR
Initial	9338	87.69	89.60	88.59	13.94
1	13795	88.29	90.03	89.11	13.42
2	15022	88.50	90.17	89.28	13.28
3	15579	88.15	90.32	89.15	12.69
4	16077	88.34	90.17	89.20	13.16
5	17646	88.49	89.94	89.18	13.78
6	18595	88.15	90.20	89.10	12.95
7	19447	88.73	90.05	89.36	13.71
8	24119	88.77	90.26	89.48	13.29
9	24556	88.67	90.10	89.35	13.57
10	26126	89.17	89.44	89.31	15.42
11	39930	89.11	89.69	89.39	14.82
12	43901	88.73	90.58	**89.60**	12.53
13	47061	88.46	**90.73**	89.51	**12.00**
14	51623	88.57	90.05	89.28	13.59
15	52777	**89.20**	89.53	89.37	15.26

Table 5. Performance comparison using SEED vs. self-labeled data.

Training Data	SEED Data Only			Pseudo-Labeled Data Only		
Model	Precision	Recall	F1-Score	Precision	Recall	F1-Score
SVM+WCNG	80.89	87.17	83.32	81.00	85.26	82.77
CNN+W2VSG	**87.69**	**89.60**	**88.59**	**86.63**	**89.22**	**87.81**
BiLSTM+W2VSG	86.68	89.49	87.95	86.52	87.87	87.16
CNN+AraBert	86.57	88.51	87.47	83.56	82.55	83.03
BiLSTM+AraBert	87.14	87.87	87.50	79.58	85.20	81.77

Experiment#4: Class Distribution Ratio Minimization

The fourth experiment assesses the class distribution ratio of the supervised dataset on the SSST performance. Each iteration self-trains the model CNN +W2CSG on the supervised dataset by gradually reducing the class distribution ratio of Clean to Offensive/Hate (OH) until the classes are equally distributed. The ratio begins with 3:1 for the SEED dataset, and then after four iterations, it reaches 1:1 where the classes are equally distributed. In each iteration, to attain the desired ratio, we compute the size of the OH samples to be selected as follows:

$$OH = \frac{N_c - N_{oh} * R_{new}}{R_{new}} \tag{1}$$

Where N_c is the number of clean instances, N_{oh} the number of OH instances, and R_{new} the target class distribution ratio.

Table 6 exposes the results for each SSST iteration, consisting of two main parts: (1) classifier training based on unlabeled data, retained confident OH data (OH) and training data, and (2) predictive performance on the testing dataset, using Precision (P), Recall (R) and F1-score (F1). Based on the above formula (1), we select the positive examples from the pseudo-label dataset to attain the selected ratio in each iteration. For instance, to attain the ratio of 2.5 to 1 in the first iteration, we keep the top 356 confidently predicted OH examples. Those selected examples have a high confidence range of [0.999796, 0.999821]. We show the confidence range to make sure we are keeping only high prediction probabilities. After four iterations, the supervised dataset increased of 4510 samples using only positive examples with very high confidence.

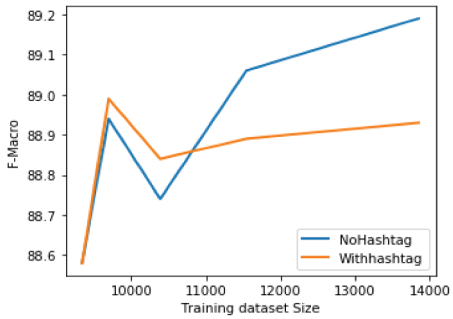
After a thorough examination of the pseudo-labeled exampled selected at each iteration, we find out that most of the highly confident OH examples come from one particular offensive and obscene hashtag. Although, those tweets are offensive, the classifier is not leaning much from seeing similar OH examples

Table 6. Lowering class imbalance ratio at each SSST iteration.

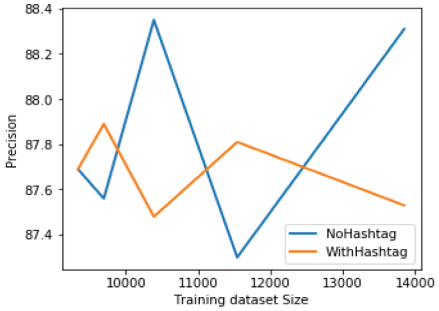
I#	UnlabData	OH Data	ConfiRange	TrainData	Ratio	P	R	F1
In.	NA	NA	NA	9338	3:1	87.69	89.60	88.58
1	5000000	356	0.999796–0.999821	9694	2.5:1	**87.89**	90.28	**88.99**
2	4999644	692	0.999457–0.999908	10386	2:1	87.48	90.50	88.84
3	4998952	1154	0.999643–0.999570	11540	1.5:1	87.81	90.16	88.89
4	4997798	2308	0.998828–0.999827	13848	1:1	87.53	**90.65**	88.93

Table 7. Lowering class imbalance ratio after removing hashtag examples.

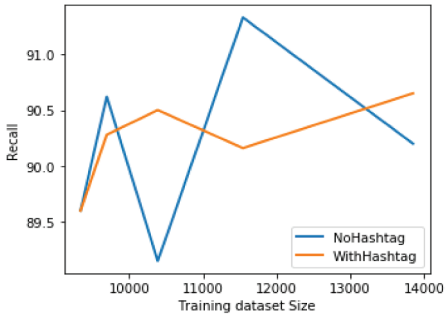
I#	UnlabData	OH Data	ConfiRange	TrainData	Ratio	P	R	F1
In.	NA	NA	NA	9338	3:1	87.69	89.60	88.58
1	4948874	356	0.999860–0.999943	9694	2.5:1	87.56	90.62	88.94
2	4948518	692	0.999605–0.999643	10386	2:1	**88.35**	89.15	88.74
3	4947826	1154	0.999649–0.999661	11540	1.5:1	87.30	**91.33??**	89.06
4	4946672	2308	0.998446–0.998663	13848	1:1	88.31	90.20	**89.19**



((a)) F-Macro



((b)) Precision



((c)) Recall

Figure 4. CNN+W2VSG performance by lowering class distribution ratio.

coming from the same hashtag in each iteration. Thus, we repeat our experiment by ignoring any tweets from this hashtag. The results of this experiment are presented in Table 7. As we can see, the accuracy improved to 89.19% as a result of removing those tweets.

Figure 4 shows the classifier performance as we lower the class distribution ratio with and without the obscene hashtag. As we can see, balancing the data resulted in an improvement of the overall performance. Furthermore, we can see that removing the tweets from the obscene hashtag yielded the best F1-score and Precision.

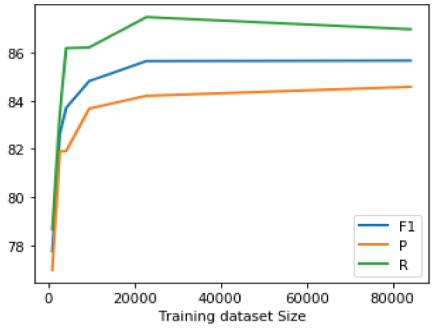
Table 8. Classifier performance with 10%, 25% and 50% of SEED dataset.

Iteration	Training Dataset	Precision	Recall	F1-Score
10% SEED Data				
Initial	934	76.99	78.67	77.76
1	2664	81.89	83.41	82.60
2	4138	81.91	86.18	83.70
3	9500	83.67	86.21	84.81
4	22770	84.20	**87.47**	85.64
5	84096	**84.57**	86.97	**85.66**
25% SEED Data				
Initial	2334	83.16	86.21	84.50
1	5084	84.44	87.35	85.74
2	6466	85.03	86.73	85.83
3	8974	84.98	87.43	86.10
4	17276	85.22	88.32	86.60
5	57236	**87.12**	**88.41**	**87.74**
50% SEED Data				
Initial	4669	85.90	89.33	87.41
1	10591	86.64	89.15	87.79
2	36697	87.21	88.25	87.71
3	52479	86.50	89.32	87.77
4	62439	**87.43**	88.82	**88.09**
5	113531	86.30	**89.81**	87.85

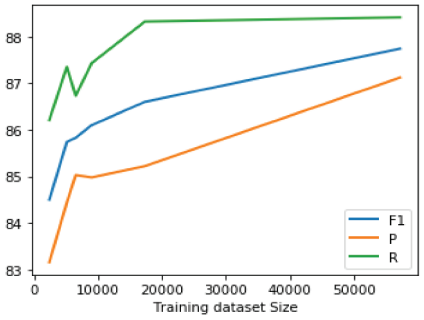
Experiment#5: Low-resource SEED Dataset Impact

To assess our self-learning approach in the setting of low-resource training data, we experiment with three different proportions of the SEED dataset, 10% (934 data), 25% (2334 data) and 50% (4669 data). For each proportion, we conduct the initial training of the CNN+W2VSG model and then five SSST iterations. Table 8 reports the accuracy for each proportion. As observed, our approach works best with small-sized datasets. When starting with less than 1000 labeled examples, the baseline classifier has an F1-score of only 0.77%. Extending the dataset with self-labeled examples resulted in a significant performance gain with an improvement of 7.9% in the F1-score after five iterations. Overall, the improvement persisted but diminished as we increase the initial labeled dataset proportion. For the 25% proportion, we achieved an improvement of 4.2% in the F1-score. For the 50% proportion, we gained the largest improvement at the end of iteration #4, with a gain of 0.68% in the F1-score.

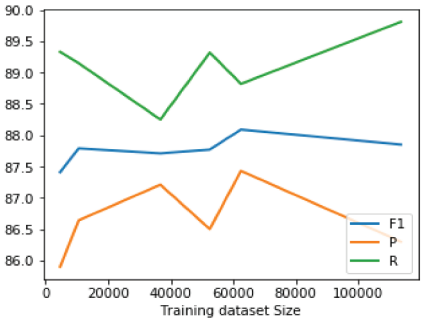
Figure 5 shows the classifier performance with 10%, 25%, and 50% of the SEED dataset over five iterations. As we can see, our SSST approach works well with smaller datasets. This finding is consistent with previous studies on semi-supervised classification, such as (Elshaar and Sadaoui 2020; Viegas, Cepeda, and Vieira 2018).



((a)) 10% SEED



((b)) 25% SEED



((c)) 50% SEED

Figure 5. Classifier performance by varying size of SEED dataset.

Experiment#5: Ensemble-based Self-training

In this section, we incorporate ensemble learning within the self-training process. We experiment with two ensemble strategies for the data selection part using two text classifiers. For ensemble learning, we utilize the Maximum Voting and Average Voting based on the confidence threshold. We choose the top two performing classifiers from Table 2, which are CNN+W2VSG and BiLSTM+W2VSG. With Maximum Voting, we first select the instances with the maximum probability scores. We then retain only the scores above the confidence threshold. Using the second strategy, we first compute the average of the two probability scores for each instance and then select all the instances with the average value above the confidence threshold.

Tables 9 and Tables 10 expose the augmented training dataset and the performance results for each iteration. The best CNN model is obtained with the averaged pseudo-labeled data, and improves the baseline classifier by 0.92% after one iteration. The retrained BilSTM model with the Average Voting is the most performing and improves the initial classifier by 1.44% after two iterations.

Table 9. Performance with averaged probability scores.

	Iteration	Training Dataset	Precision	Recall	F1-Score
CNN+W2VSG	initial	9338	87.69	89.60	88.59
	1	16418	87.99	**91.40**	**89.51**
	2	45569	**88.58**	90.28	89.39
BiLSTM+W2VSG	initial	9338	86.68	89.49	87.95
	1	16418	87.57	89.93	88.66
	2	45569	**87.82**	**91.37**	**89.39**

Table 10. Performance with maximum probability scores.

	Iteration	Training Dataset	Precision	Recall	F1-Score
CNN+W2VSG	initial	9338	87.69	89.60	88.59
	1	38433	**88.44**	90.57	**89.43**
	2	106423	87.63	**90.95**	89.11
BiLSTM+W2VSG	initial	9338	86.68	89.49	87.95
	1	38433	86.95	**90.92**	88.66
	2	106423	**88.89**	89.83	**89.35**

Our results show that the ensemble-based selection of confident pseudo labeled data achieves comparable results to classical self-training but very early in the process.

Discussion

Our self-training approach leverages the unlabeled data to improve the performance of the hate speech detection model. We showed that the self-training method produced good quality pseudo-labeled data; the latter is much larger than the initial labeled dataset. This new supervised dataset is publicly available for the research community to advance Arabic hate speech detection systems further. We also demonstrated that our framework returned promising results in different experimental scenarios, with the most promising being the low resource setting where only a few manually annotated instances are available. We showed that an improvement of up to 7% was obtained from using additional pseudo- labeled data. Additionally, our self-training framework could be used as a tool to augment the dataset for any text classification tasks.

Although the experimental results of self-training are encouraging, we addressed some challenging issues, including (i) the imbalanced class ratio of the selected confident pseudo-labeled data, and (ii) the under-representation of hard examples in the selected highly confident data as the proposed selection strategy only selects non-confusing examples for retraining. However, focusing the attention on the hard examples may be more beneficial in boosting the classifier's performance.

Furthermore, we manually examined the misclassified posts, which revealed some dominant patterns that are still challenging for the classifier even when trained on a larger dataset. For example, many of the hate/offensive posts

incorrectly labeled as clean include implicit hate, such as the tweet: "The kitchen awaits you. you are sitting among men without work – المطبخ ينتظرك جالسة وسط الرجال بلا شغل"

This tweet has no offensive or slur words making it difficult for the text classifier to make a correct decision. Another prevalent patterns incorrectly labeled as clean are the tweets with dialectal offensive or hateful terms that are underrepresented in the training corpus, like the tweet: "Exterminate the nomads exterminate the nomads exterminate the nomads. – ابيدو البدو ابيدو البدو ابيدو البدو ابيدو البدو " where the word "exterminate: ابيدو " is a rare term found only in some Arabic dialects.

On the other hand, clean tweets incorrectly labeled as hate/offensive tend to counter and negate hate speech, such as the tweet: "Not all **** are barbaric or all **** are barbaric or we do not judge people by one person" -ليس جميع ***** همج او جميع ***** همج لا نحكم على شعب من شخص"

Another dominating pattern in false positives instances are the tweets that contain slur or abusive words but are neither hate nor offensive, such as the post: "lonely session to mourn my life and my sh*** exam

-جلسة وحدانية عشان ألطم على حظي وأختباري اللي زي ال*"

Conclusion and Future Work

In this paper, we harnessed a large amount of social media posts to enable more robust learning of offensive and hateful speech for the Arabic language. We successfully exploited the iterative semi-supervised self-training approach to develop a robust text classifier, a deep neural network combined with a pre-trained word embedding, using a small labeled dataset and a rich unlabeled dataset. Moreover, we built a reliable supervised Arabic corpus that researchers can utilize to assess new learning algorithms and word embedding mechanisms. It is worth mentioning that this is the first time semi-supervised learning is conducted with so many iterations, even though deep neural networks re-training is very time-consuming and the running time increases with the size of the textual corpus. We conducted six groups of experiments and showed that SSST is a competent approach to developing robust OHS classifiers with much less human effort and time.

There are several research directions for OHS classification, given below:

- Incorporate active learning within the SSST process. Generally speaking, active learning selects training data based on their estimated informativeness. The idea here is to utilize active learning in each iteration to choose the best samples from the unlabeled dataset to be labeled by the OHS classifier.

- Combine SSST with incremental learning. Instead of re-training from scratch, the OHS classifier is updated incrementally with the new labeled data in each iteration by retaining past learned knowledge. Incremental learning has been shown to improve predictive accuracy and time efficiency compared to static learning.
- Consider the socio-cultural context for hate speech detection. We will investigate demographic information together with social graphing to build multi-view text classification models. The objective here is to improve the classifier's interpretability and predictions.

Acknowledgments

This project was funded by the Deanship of Scientific Research (DSR), Jeddah University, Jeddah. The authors, therefore, acknowledge with thanks DSR technical and financial support.

Disclosure statement

No potential conflict of interest was reported by the author(s).

ORCID

Safa Alsafari http://orcid.org/0000-0001-8023-8833
Samira Sadaoui http://orcid.org/0000-0002-9887-1570

References

Albadi, N., M. Kurdi, and S. Mishra. 2018. Are they our brothers? Analysis and detection of religious hate speech in the Arabic twittersphere. In Proceeding of IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), 69–76, Barcelona, Spain .

Alsafari, S., S. Sadaoui, and M. Mouhoub. 2020a. Deep learning ensembles for hate speech detection. 32th International Conference on Tools with Artificial Intelligence, ICTAI, Virtual Conference.

Alsafari, S., S. Sadaoui, and M. Mouhoub. 2020b. Effect of word embedding models on hate and offensive speech detection. In arXiv, CC BY 4.0.

Alsafari, S., S. Sadaoui, and M. Mouhoub. 2020c. Hate and offensive speech detection on Arabic social media. *Online Social Networks and Media* 19:100096. doi:10.1016/j.osnem.2020.100096.

Antoun, W., F. Baly, and H. Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding.

Bhojanapalli, S., K. Wilber, A. Veit, A. S. Rawat, S. Kim, A. Menon, and S. Kumar. 2021. On the reproducibility of neural network predictions.

Borrajo, M., R. Romero, and E. Iglesias. 2015. A linear-RBF multikernel SVM to classify big text corpora. *BioMed Research International* 2015:1–14.

Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805,.

Elshaar, S., and S. Sadaoui. 2020. Detecting bidding fraud using a few labeled data. In 12th International Conference on Agents and Artificial Intelligence - Volume 2: ICAART, 17–25, Valletta, Malta.

Georgakopoulos, S. V., S. K. Tasoulis, A. G. Vrahatis, and V. P. Plagianakos. 2018. Convolutional neural networks for toxic comment classification. In Proceedings of the 10th Hellenic Conference on Artificial Intelligence, Patras, Greece.

Graves, A., A. Mohamed, and G. Hinton. 2013. Speech recognition with deep recurrent neural networks. In 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, may, 6645–49.

He, J., J. Gu, J. Shen, and M. Ranzato. 2020. Revisiting self-training for neural sequence generation.

Hughes, M., I. Li, S. Kotoulas, and T. Suzumura. 2017. Medical text classification using convolutional neural networks. In *Studies in Health Technology and Informatics*235:246–25, IOS Press, Amsterdam, Netherlands.

Kahn, J., A. Lee, and A. Hannun. 2020. Self-training for end-to-end speech recognition. In Proceeding of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 7084–88, Barcelona, Spain.

Lee, J. Y., and F. Dernoncourt. 2016. Sequential short-text classification with recurrent and convolutional neural networks. In Proceedings of the 2016 Conference of the North {A} merican Chapter of the Association for Computational Linguistics: Human Language Technologies, 515–20, San Diego, CA, USA.

Li, J., Q. Zhu, Q. Wu, and D. Cheng. 2020. An effective framework based on local cores for self-labeled semi-supervised classification. *Knowledge-Based Systems* 197:105804. doi:10.1016/j.knosys.2020.105804.

Li, Y.-F., and D.-M. Liang. 2019. Safe semi-supervised learning: A brief introduction. *Frontier Computing Science, Springer-Verlag* 13 (4):669–76. doi:10.1007/s11704-019-8452-2.

Liu, Y., J.-W. Bi, and Z.-P. Fan. 2017. A method for multi-class sentiment classification based on an improved one-vs-one (OVO) strategy and the support vector machine (SVM) algorithm. *Information Sciences, Elsevier* 394:38–52. doi:10.1016/j.ins.2017.02.016.

Madhyastha, P., and R. Jain. 2019. On model stability as a function of random seed. 929–39.

Mikolov, T., I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems* 26, 3111–3119, Red Hook, NY, USA.

Mubarak, H., A. Rashed, K. Darwish, Y. Samih, and A. Abdelali. 2020. Arabic offensive language on Twitter: Analysis and experiments.

Mubarak, H., and K. Darwish. 2017. Abusive language detection on Arabic social media. In Proceeding of the First Workshop on Abusive Language Online, 52–56. Association for Computational Linguistics, Vancouver, BC, Canada.

Mulki, H., H. Haddad, C. B. Ali, and H. Alshabani. 2019. L-HSAB: A levantine Twitter dataset for hate speech and abusive language. In Proceeding of the Third Workshop on Abusive Language Online, 111–18. Association for Computational Linguistics, Florence, Italy.

Nartey, O. T., G. Yang, J. Wu, and S. K. Asare. 2020. Semi-supervised learning for fine-grained classification with self-training. *IEEE Access* 8:2109–21. doi:10.1109/ACCESS.2019.2962258.

Ousidhoum, N. L. Zizheng, Z. Hongming, S. Yangqiu and Y. Dit-Yan. 2019. Multilingual and multi-aspect hate speech analysis. In Proc. of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), 4675–84, Hong Kong, China.

Ousidhoum, N., Y. Song, and D.-Y. Yeung. 2020. Comparative evaluation of label-agnostic selection bias in multilingual hate speech datasets. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 2532–42. Association for Computational Linguistics.

Rosenthal, S., P. Atanasova, G. Karadzhov, M. Zampieri, and P. Nakov. 2020. A large-scale semi-supervised dataset for offensive language identification.

Sanh, V., L. Debut, J. Chaumond, and T. Wolf. 2019. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *ArXiv preprint arXiv:191910.01108*.

van Engelen, J. E., and H. Hoos. 2019. A survey on semi-supervised learning. *Machine Learning, Springer* 109 (2):373–440. doi:10.1007/s10994-019-05855-6.

Viegas, J. L., N. M. Cepeda, and S. M. Vieira. 2018. Electricity fraud detection using committee semi-supervised learning. In 2018 International Joint Conference on Neural Networks (IJCNN), 1–6, Rio de Janeiro,Brazil .

Wang, J., Y. Yang, J. Mao, Z. Huang, C. Huang, and X. Wei 2016. CNN-RNN: A unified framework for multi-label image classification. 2285–94.

Wu, D., M. Shang, X. Luo, J. Xu, H. Yan, W. Deng, G. Wang. 2018. Self-training semi-supervised classification based on density peaks of data. *Neurocomputing* 275 (C):180–191. doi:10.1016/j.neucom.2017.05.072.

Xia, M., A. Anastasopoulos, X. Ruochen, Y. Yang, and G. Neubig. 2020. On model stability as a function of random seed.

Xie, Q., M.-T. Luong, E. Hovy, and Q. V. Le. 2020. Self-training with noisy student improves imagenet classification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June, Seattle, WA, USA.

Xu, Q., A. Baevski, T. Likhomanenko, P. Tomasello, A. Conneau, R. Collobert, G. Synnaeve, and M. Auli. 2020. Self-training and pre-training are complementary for speech recognition.

Zampieri, M., P. Nakov, S. Rosenthal, P. Atanasova, G. Karadzhov, H. Mubarak, L. Derczynski, Z. Pitenis, and Ç. Çöltekin. 2020. SemEval-2020 Task 12: Multilingual offensive language identification in social media (OffensEval 2020), 1425–47, Barcelona, Spain.

Zampieri, M., S. Malmasi, P. Nakov, S. Rosenthal, N. Farra, and R. Kumar. 2019. Predicting the type and target of offensive posts in social media. In Proceedings of the 2019 Conference of the North American Chapter of the Association for ComputationalLinguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 1415–20, Minneapolis, Minnesota.

Zhu, X., and A. Goldberg. 2009. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning,* 3:1-130. Morgan and Claypool, San Rafael, California, USA

Zoph, B., G. Ghiasi, T.-Y. Lin, Y. Cui, H. Liu, E. D. Cubuk, and Q. V. Le. 2020. Rethinking pre-training and self-training.