Applied Soft Computing Journal 101 (2021) 107023

Contents lists available at ScienceDirect

Applied Soft Computing Journal

journal homepage: www.elsevier.com/locate/asoc

Analyzing the effectiveness of semi-supervised learning approaches for opinion spam classification

Alexander Ligthart [a], Cagatay Catal [b,*], Bedir Tekinerdogan [a]

[a] Wageningen University & Research, Information Technology Group, Wageningen, The Netherlands
[b] Bahcesehir University, Department of Computer Engineering, Istanbul, Turkey

ARTICLE INFO

Article history:
Received 25 March 2020
Received in revised form 19 October 2020
Accepted 10 December 2020
Available online 25 December 2020

Keywords:
Opinion spam detection
Fake reviews
Semi-supervised learning
Machine learning

ABSTRACT

Opinion spam detection is concerned with identifying fake reviews that are deliberately placed to either promote or discredit a product. Opinionated social media like product reviews are increasingly important resources for people as well as businesses in the decision-making process and can be easily manipulated by opportunistic individuals. To reduce this increasing impact of opinion spams, opinion spam detection approaches have been proposed, which adopt mostly supervised classification methods. However, in practice, the provided data is largely not labeled and therefore semi-supervised learning approaches are required instead. To this end, this study aims to analyze the effectiveness of several semi-supervised learning approaches for opinion spam classification. Four different semi-supervised methods are evaluated on a dataset of both genuine and deceptive hotel reviews. The results are compared with several traditional classification methods using the same amount of labeled data. According to this study, the self-training algorithm with Naive Bayes as the base classifier yields 93% accuracy. Results show that self-training is the only approach, out of the four tested semi-supervised models, that outperforms traditional supervised classification models when limited data is available. This study further shows that self-training can mitigate labeling efforts while retaining high model performance, which is useful for scenarios where limited data is available or retrieving labeled data is more costly.

1. Introduction

Opinionated social media such as product reviews have become an important resource for individuals and organizations in the decision-making process. Product reviews can be placed by anyone and contain the word of mouth information that is in most cases not present in the product description itself regarding the quality, durability, product usage, etc. The rise of e-commerce platforms caused an enormous growth in the number of opinions spread online. Due to this trend, opinion spam detection has become a prominent issue.

Opinion spam, also known as fake reviews or review spam, refers to reviews that intentionally promote or discredit products. In recent years, opinion spam detection has attracted growing attention in research and business communities. The amount of publications on opinion spam is growing exponentially [1] in these days where the internet has become an integral part of life and false information spreads just as fast as accurate information on the web [2].

Supervised opinion spam classification methods are widely studied for the opinion spam classification problem. With the help of labeled data instances, algorithms can detect patterns in spammer reviews. A relatively undiscovered field in machine learning is semi-supervised learning (SSL). Semi-supervised learning refers to the automatic labeling of data instances with the goal of using non-labeled examples to improve performance [3]. Directing research efforts towards the implementation of semi-supervised learning is likely to be rewarding because it can potentially mitigate the labor-intensive problem of finding and labeling data. Especially for opinion spam detection, massive amounts of unlabeled data instances (i.e., product reviews) are available [4]. There is limited literature available on semi-supervised methods for opinion spam classification. Hence, it is not clear whether semi-supervised methods are substantially improving opinion spam detection performance compared to traditional supervised methods or not. Also, features that have been applied so far in this context are rather limited.

This study aims to measure the effectiveness of semi-supervised learning algorithms for opinion spam classification. To this end, the following research question is formulated:

* Corresponding author.
E-mail addresses: alexander.ligthart@wur.nl (A. Ligthart), cagatay.catal@eng.bau.edu.tr (C. Catal), bedir.tekinerdogan@wur.nl (B. Tekinerdogan).

https://doi.org/10.1016/j.asoc.2020.107023

How effective is semi-supervised learning for opinion spam classification?

To answer this research question, several experiments with semi-supervised classification methods are conducted in this study. Results of semi-supervised classification algorithms are compared with traditional classification methods. The main contribution of this in-depth study is to explore the potential semi-supervised classification algorithms in conjunction with different feature sets.

The remaining part of this paper is organized as follows: Section 2 presents the background and related work. Section 3 explains the methodology in general, dataset, semi-supervised learning techniques, and evaluation metrics. Section 4 presents the results. Section 5 provides the discussion, and finally, Section 6 provides the conclusions.

2. Background and related work

Clues to identifying spammers are usually hidden in multiple aspects such as content, behavior, relationships, and interaction with the review [5]. Opinion spam detection aims to identify multiple features that relate to a fake review. The most widely available feature is the review content, which refers to the actual textual information in the review. Besides the review content, the meta-data of the review can reveal valuable information. Some examples of meta-data are star rating, time of placement, reviewer IP address, statistics of interaction with the comment, and so on. Furthermore, real-life knowledge about the product could also reveal spammer clues.

Semi-supervised learning is a branch of machine learning that makes use of a small set of labeled data and a large set of unlabeled data to improve learning accuracy. Several assumptions are made when building semi-supervised learning models. Firstly, the *continuity assumption* states that tightly connected instances are likely to belong to the same class [6]. Secondly, the *cluster assumption* states that data tends to form clusters, and points in the same cluster are more likely to share class labels [3]. Lastly, the *manifold assumption* states that data points lie on a much lower dimension than the actual input space. Semi-supervised learning generally comprises six types of methods as depicted in Fig. 1. These methods are graph-based learning, self-training, co-training, multi-view learning, low-density separation (LDS), and generative models [6–8]. These six main methods are described in the following subsections.

2.1. Types of semi-supervised learning

Graph-based learning: Graph-based methods for semi-supervised learning aim at propagating label information of data samples to neighboring data samples until a global stable state is reached. A graph representation of the data is presented with nodes for each labeled or unlabeled example and connections represent similarities among labeled as well as unlabeled samples [8]. The graph is constructed using similarity of examples; two common methods are to connect each data point to its {\displaystyle k}nearest neighbors (kNN) or examples within a certain distance.{\displaystyle \epsilon} Graph-based learning has shown better classification accuracy compared to self-training at the cost of more computational complexity. It has become a gradually high performing technique [8].

Two popular algorithms for graph-based learning are *Label Propagation* and *Label Spreading*, which are used for classification and regression tasks. Label Propagation uses the raw similarity matrix constructed from the data without any modifications. Label Spreading aims at minimizing a loss function that has regularization properties and is created to be more robust to noise in data. Both of these algorithms aim to propagate labels through the dataset along high-density areas defined by unlabeled data instances. Label Spreading can be considered as a variant of Label Propagation that is often more robust to the noise in data. Both are default semi-supervised methods in the scikit-learn machine learning software library. While one study showed that basic SVM outperformed label propagation and label spreading for the text classification [9], two recent studies applied these algorithms successfully for sentiment classification [10,11]. Giasemidis et al. (2018) applied these algorithms on a dataset that consists of hundreds of thousands of Twitter messages and showed that they are fast and accurate for message stance classification [10]. Yang and Shafiq (2018) demonstrated that the label propagation algorithm is robust for large scale and parallel sentiment analysis of tweets [11].

Self-training: Self-training, also known as self-learning, is considered the most popular method for semi-supervised learning and has been used abundantly [6]. It is a fast and straightforward method proposed by Yarowsky [12] that trains a classifier on the partition of labeled data. Subsequently, predictions based on this classifier are evaluated. The most confidently labeled predictions with a confidence level of at least 80% are added to the labeled dataset. This process is repeated until the convergence. Self-training is hard to implement with discriminative classifiers like SVM. Self-training requires only little labeled examples but can suffer from poor prediction. Pavlinek & Podgorelec [13] used self-training and a topic modeling method based on Latent Dirichlet Allocation (LDA). In this study, the self-training approach was used to enlarge the labeled set from the unlabeled data points. The advantage of the self-training approach is its easy combination with any classification algorithm.

Co-training: Co-training is an extension of self-training that uses both labeled as well as unlabeled examples [14]. This algorithm requires an extra view of the data with different and complementary information about the instance. Two classifiers are trained separately, and the information obtained from training is shared with each other. For each class, each classifier predicts one unlabeled example to add to the set of labeled documents for each iteration, where predictions are most confident (i.e., at least greater than alpha). Co-training assumes that each training example in the dataset consists of two views (i.e., feature sets) of data [15]. Each view (i.e., X1 and X2) is a distribution of features that make up the example. Two feature sets should provide different and complementary information about the instance (e.g., review content information and meta-data in spam classification). The algorithm aims at training two classifiers on the labeled data for both views to iteratively construct additional labeled examples where the predictions are most confident. The condition is that two data views X1 and X2 are not directly corelatable with each other. Nigam & Ghani (2000) show that when there are no natural multiple views available, co-training on multiple views manually generated by random splits of features can still improve performance. Li et al. [16] showed that the co-training algorithm outperforms self-training algorithm-based models for review spam detection.

Multi-view Learning: Multi-view learning is considered a variant of co-training, that aims at producing multiple models based on different views (i.e., feature sets) of the data points. Multi-view learning is also known as data fusion or data integration from multiple feature sets. Multi-view learning labels data instances based on a majority vote. When there are no multiple views available, multiple views manually generated by random splits of feature sets can still improve the performance [17]. Bhattacharjee et al. [18] used a multi-view, semi-supervised, active learning

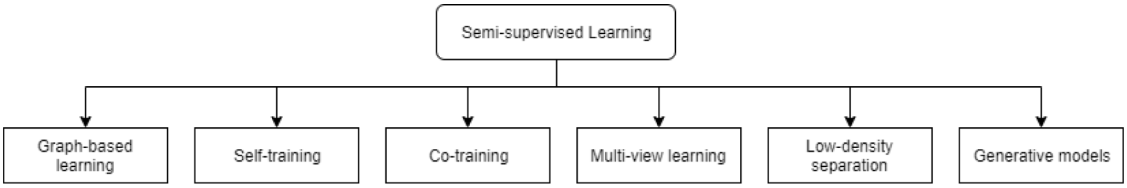

Fig. 1. Semi-supervised learning types.

method to identify malicious social media content. The multi-view approach helped to use complementary cues from several resources.

Low-Density Separation: Low-density separation attempts to place boundaries in the regions of the dataset where there are few data points, either labeled or unlabeled. The cluster assumption is an important and prominent issue in LDS [3]. A popular algorithm that applies low-density separation is Transductive SVM (TSVM), also known as S3VM or SSSVM. The method aims to maximize the margin between unlabeled data instances, assuming large margins between label classes. TSVM model is less susceptible to overfitting compared to the self-training and co-training model. However, it is classifier dependent. TSVM has clear optimization criteria but is hard to optimize and prone to local minima. Research efforts mainly go out to find solutions for the non-convex nature of TSVM, which makes it hard to find global minima of the cost function, an important task in many machine learning algorithms. Li et al. [19] proposed a novel two-view TSVM algorithm and showed that the new approach performs 5% better performance than a single view learning algorithm for review spam classification. Another advantage of the proposed approach is that it is more stable for noisy data.

Generative models: Generative models provide a way of treating missing information by providing a maximum likelihood estimation [6]. This probabilistic method turned out to be especially useful for semi-supervised learning, where the label is estimated. A popular generative algorithm in text classification is Multinomial Naive Bayes combined with the Expectation–Maximization (EM) algorithm. This method first trains a Naive Bayes classifier on the labeled data. Subsequently, the probabilities of unlabeled data associated with each class are calculated. A new Naive Bayes classifier is built with labeled data and estimated labels, which are used as true class labels. This process of classifying the unlabeled data is iterated until it converges to a stable classifier and set of labels [20]. Mukherjee and Venkataraman [21] proposed a novel unsupervised generative model for detecting opinion spam and evaluated the approach on three datasets. Linguistic and behavioral clues were used in an unsupervised Bayesian inference framework. They demonstrated that the proposed model outperforms the other algorithms across all the datasets.

2.2. Related work

Several researchers have studied the text classification problem with semi-supervised approaches. The work of Rout et al. [15] aims at testing multiple semi-supervised methods on the 'gold standard' dataset [22]. A variety of experiments are conducted with different input features. Co-training, EM, Label Propagation, and Label Spreading are included as semi-supervised methods with multiple base classifiers like kNN, RF, and LogR. They concluded that new dimensions like POS tags and LIWC features improve performance. Pavlinek & Podgorelec [13] used self-training as well as a newly designed topic modeling method based on Latent Dirichlet Allocation (LDA) for text classification on different news datasets. A newly proposed LDA method is compared with methods like Multinomial Naive Bayes, Self-Training, SVM, and EM-Multinomial NB. Different algorithms and ratios of initially labeled sets are tested. They showed that the newly proposed method could increase performance when limited data is available. Li et al. [16] studied the review spam identification problem. They collected their data and conducted co-training with two views of the data: the content and the user's behavior. Their model outperforms self-training models with only review content involved. Karimpour et al. [23] analyzed the Expectation–Maximization algorithm for improving web spam detection. Web content link-based features and linguistic features were extracted. Naive Bayes with EM algorithm yielded high performance. Hassan and Islam [24] compared semi-supervised with supervised techniques for detecting fake reviews. Different input features were used and compared. Naive Bayes outperformed all other methods used in their study.

Narayan et al. [25] applied PU-learning semi-supervised learning algorithm [26] with six classification algorithms, namely Decision Tree, Naive Bayes, SVM, K-NN, Random Forest, Logistic Regression to detect spam reviews and reported that this algorithm provides maximum accuracy of 78.12%. PU-learning is the acronym of *P*ositive and *U*nlabelled learning. Stanton and Irissappane [27] proposed a new algorithm called spamGAN based on the Generative Adversarial Networks (GANs) algorithms. GAN is a popular deep learning algorithm that provided state-of-the-results on images, but they applied this new algorithm on textual data. Researchers reported that spamGAN provides better performance than the other spam detection techniques on the TripAdvisor dataset. Manaskasemsak et al. [28] proposed a new semi-supervised graph-based partitioning technique called BeGP for opinion spam detection. This algorithm creates a user behavioral graph on a dataset collected from the Yelp.com website. The results are similar to the state-of-the-art solutions. Wu et al. [29] developed a semi-supervised collaborative learning technique and identified both spam messages and social spammers. They used a dataset created based on the data of the Weibo.com website. They reported that their approach provides better performance than several semi-supervised learning methods such as S3VM, self-training, and co-training. Li et al. [30] developed a semi-supervised deep social spammer detection model called SSDMV and investigated its performance on datasets created from Twitter and Sina Weibo. They concluded that SSDMV provided better performance than the existing algorithms in terms of effectiveness and robustness. Yelundur et al. [31] proposed a semi-supervised binary multi-target extension to the unsupervised model (SENTINEL) for the detection of review abuse. They performed experiments on the Amazon Customer Reviews dataset and demonstrated that their inference achieves faster convergence than online EM and stochastic gradient. Also, they showed that their model achieves higher precision/recall than unsupervised techniques. Alvari et al. [32] developed a semi-supervised causal inference for detecting Pathogenic Social Media (PSM) accounts that spread misinformation. They used a semi-supervised Laplacian SVM to detect these users. Features were extracted from the user activity log. They showed the effectiveness of the model in a dataset created from tweets. Yilmaz and

Durahim [33] used the dense document and node embeddings for spam review detection and developed a semi-supervised spam review detection framework. In this framework, they eliminated the manual feature engineering step and showed that the combined feature vectors provide better performance than the existing algorithms. Wang et al. [34] developed a new model based on semi-supervised recursive autoencoders for spam review detection. Experiments were performed on a dataset collected from Sina Weibo that is a popular social networking platform in China. They concluded that the proposed model is effective in detecting spam reviews. Deng et al. [35] proposed a new semi-supervised learning algorithm based on PU-Learning that uses multi-aspect features. Autoencoder algorithm was applied for dimensionality reduction and K-means was used to classify the data. Metadata features and content-related features were used for building the model. Zhang et al. [36] developed a semi-supervised learning based spammer group detection method. This model uses the Naive Bayes algorithm on a small labeled dataset and then, uses unlabeled data with the Expectation–Maximization algorithm to improve the classifier iteratively. They showed that the proposed model is efficient on Amazon.cn datasets. Imam et al. [37] proposed a semi-supervised learning approach for managing the Twitter spam drift. Since spam review characteristics can change over time, it is crucial to develop models that can handle this drift. They applied YATSI (Yet Another Two-Stage Idea) semi-supervised learning algorithm and Random Forest was selected as the base classifier in their model. Chengzhang et al. [38] used a three-view semi-supervised learning algorithm called tri-training for spam review detection. They showed that it provides better performance than the two-view co-training and single-view algorithm on the AliExpress dataset. Ahsan et al. [39] used the Active Learning algorithm that is a semi-supervised algorithm for fake review detection and reported that this algorithm provides promising opportunities. Aghakhani et al. [40] used Generative Adversarial Networks (GANs) for detecting deceptive reviews. Unlike the traditional GAN algorithm, they applied two discriminator models and one generative model. They reported that their model provided the same accuracy as the performance of supervised learning algorithms on TripAdvisor hotel reviews. Xu et al. [41] proposed the Semi-supervised Sequential Variational Autoencoder (SSVAE) algorithm and demonstrated that SSVAE improves the accuracy compared to the supervised classifiers on Large Movie Review Dataset (IMDB) and AG‘s News Corpus. This study focused on text classification and did not analyze its performance on fake review detection datasets.

3. Methodology

This research aims at analyzing the effectiveness of a variety of semi-supervised learning methods and different experimental setups for opinion spam classification. Four semi-supervised methods with different base estimators are compared with several traditional classification methods, namely SVM, NB, and RF. For each semi-supervised learning algorithm, different input features and ratios of initially labeled data instances are explored. Additionally, the best performing experimental setup for each semi-supervised model is tested against the same model with only positive polarity or only negative polarity data instances included.

3.1. Dataset

For this study, the ‘gold standard’ dataset is adopted [22]. The balanced dataset contains 1600 reviews of hotels in the area of Chicago, USA. There are 800 deceptive reviews and 800 genuine reviews. Both genuine and deceptive review items consist of 400 positive and 400 negative polarity instances. The genuine reviews are derived from the web, deceptive reviews are created with Amazon Mechanical Turk.

To test the generalization of the results, experiments with two additional datasets are conducted. The first additional dataset [42] is a balanced dataset of Yelp hotel reviews with 780 deceptive and 780 genuine reviews. The deceptive reviews consist of 496 positive and 284 negative polarity instances, the truthful reviews consist of 631 positive and 149 negative polarity instances. The second additional dataset [43] is a balanced dataset of Yelp restaurant reviews, with 800 deceptive and 800 genuine reviews. The deceptive reviews consist of 489 positive and 311 negative polarity instances, the truthful reviews consist of 529 positive and 271 negative polarity instances. This study focuses on classifying reviews judging from the content of the review since the dataset does not supply information on meta-data of the review or the opinion holder of the review.

3.2. Data preparation

Fig. 2 shows the flowchart of the data preparation phase. The dataset is downloaded from the website [22] and imported into the Python 3 Jupyter Notebook. Tokenization took place for further processing, after which stop words are removed from the data. Lemmatization is applied, removing inflectional endings from words to create a more basic representation of the text. In some experiments, POS tags are assigned to the tokens. After the pre-processing phase, vectorization is applied, where sparse vectors are created for each data instance with 0s representing absent words and 1s representing present words. Some experiments include vectors with TF-IDF values instead of 1s. Labels are assigned to each data instance as follows: a 0 or 1 for respectively genuine or deceptive reviews.

To reduce dimensionality, the top 1000 features that are considered the strongest predictors for the class label are selected and used for the experiments in this study. Feature selection is performed with the Chi-Square test. This method aims to identify features that class labels are highly dependent on and weeds out features that seem likely to be independent of class. This study tests both unigrams and bigrams. In both cases, the top 1000 features (i.e., highest F-value in chi-square) are included.

For example, for truthful reviews, some of the selected words are enormous, beer, especially, asked, and helpful. For deceptive reviews, some words are awful, fully, impossible, greeting, and historic.

A test is conducted to estimate the optimal number of features. The co-training algorithm with three different base classifiers is tested with instances of 500, 1000, and 5000 features. Details of the use of co-training algorithm (i.e., data views) are presented in Section 3.3. The accuracy values of co-training with SVM are respectively 0.85, 0.87, 0.86. For co-training with Naïve Bayes, the accuracy values are respectively 0.89, 0.91, 0.88. For co-training with Random Forests, the accuracy values are respectively 0.87, 0.88, 0.86. In each of the three experiments, the 1000-feature instance performs the best. After the feature selection, the dataset is split into the training set and testing set with training to the testing ratio of 80:20. For the semi-supervised learning experiments, a percentage of labels of the training set are removed.

3.3. Semi-supervised methods

Different experiments are conducted with the prepared data. Semi-supervised learning algorithms discussed below are included in our experiments. All experiments are conducted by using Python programming language and the scikit-learn library. Some external libraries are imported that are made compatible

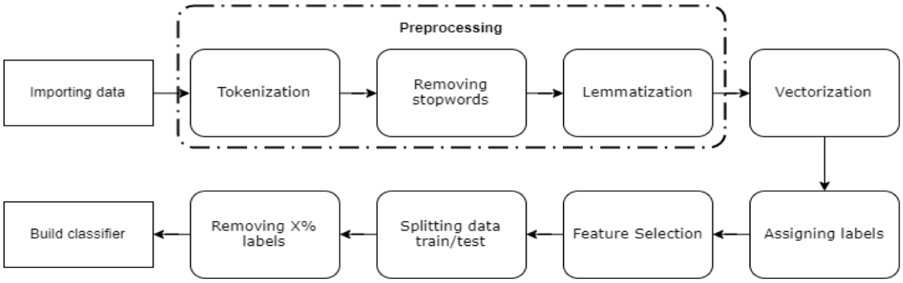

Fig. 2. Data Preparation Process.

with the scikit-learn. Grid Search is applied to select the optimal hyperparameters for base classifiers of classification models. Section 4 presents the results of the models.

Self-training: Several experiments are conducted with the self-training algorithm. The classification model is built with different base estimators (SVM, NB, RF), different input features (Unigrams, Bigrams, TF-IDF values, POS-tags), and different ratios of labeled data (5%, 10%, 20%). Results are compared with traditional supervised methods.

Co-training: The condition for the co-training is that two data views X1 and X2 are not directly co-relatable with each other. The first view X1 used is similar to features used in the self-training setup. The following linguistic features are extracted from each review to create the second view, X2: Polarity value (1 or 0), number of words, number of unique words, number of sentences, average number of words per sentence, and number of digits per word. Different experiments are conducted, with both X1, X2, and random splits of X1 features. Only bigrams are used for X1 because the self-training results show an overall increase in performance for bigrams compared to unigrams. Just like in self-training, different base estimators (SVM, NB, RF), different input features (Unigrams, Bigrams, TF-IDF values, POS-tags), and different ratios of labeled data (5%, 10%, 20%) are used.

TSVM (QNS3VM): The QNS3VM algorithm is a type of Transductive SVM with the quasi-newton optimization method included [44]. The quasi-newton method is an alternative way of finding the local minima of the loss function. This method first obtains an initial candidate class in a supervised manner and subsequently, makes use of increasing proportions of unlabeled data instances to find new insights. Different input features (Unigram, Bigram, TF-IDF, and POS) and ratios of labeled data (5%, 10%, 20%) are explored. No different base estimators are required since the method runs on Support Vector Machines only. The results of the TVM are compared with traditional classifiers and other semi-supervised methods.

Label propagation and label spreading: These algorithms only work if a proper similarity measure exists. However, in sentiment analysis, label propagation and spreading often favor topic similarity over sentiment polarity or other indicators [45], in this case, opinion spam indicators. Different input features (Unigram, Bigram, TF-IDF, and POS) and ratios of labeled data (5%, 10%, 20%) are explored.

3.4. Evaluation metrics

Four performance metrics are used for the evaluation of the experiments as shown in Table 1. Reported results of four metrics are obtained from the average results of five-fold cross-validation on the original dataset. Different evaluation metrics can be of interest to different types of analysis. Accuracy is a reliable performance metric for balanced datasets, but more specific evaluations can be applied with the precision, recall, and F1 score. For instance, when costs for falsely classified positive instances are high, like in spam detection, precision can be an essential indicator of the model. In other instances, (e.g., healthcare, banking), actual positives predicted as negative can have bad consequences. When a balance between precision and recall needs to be established or if the dataset is imbalanced, F1-score might be the best metric to use.

The Paired Student't-test, as proposed by Nadeau and Bengio [46], is used to compare the means of different models. This statistical test determines whether there is a significant difference between the average results of five-fold cross-validation with a confidence level of 95%.

4. Results

This section explains the performance results of different experiments. Table 2 shows the average results of 5-fold cross-validation experiments for traditional supervised learning models, which were used as a benchmark for judging the results of semi-supervised methods explored in this research.

For each table in this section, percentage values in rows indicate the ratio of labeled data used to train the models. Main columns (i.e., unigram, bigram, TF-IDF Bigram, POS unigram) show the input features that are used in the experimental setups and sub-columns (i.e., accuracy (Acc.), precision (Prec.), recall, F1-measure (F1)) under the main columns represent the performance results.

Additional experiments: The supervised learning experiments are tested on two additional datasets. For the Yelp hotel review dataset [42], the best performance (i.e., 73% accuracy) using limited labeled data (i.e., 20% labeled instances) was achieved when the Naïve Bayes algorithm was applied together with TF-IDF values. The same experimental setup with 100% of labeled data instances scores 81% accuracy.

The experiments on the Yelp restaurant review dataset [43] yielded a maximum accuracy of 71% with 20% of labeled data instances and Naïve Bayes classifier with TF-IDF values. The same setup on 100% labeled data reached 84% accuracy.

4.1. Self-training

Table 3 presents the average results of 5-fold cross-validation for the self-training model. For each experimental setup having different conditions, self-training outperforms traditional supervised machine learning methods. The performance results of the self-training model having different ratios of the labeled data instances perform almost as well as traditional supervised models trained on all 1600 labeled data instances.

The self-training model with Multinomial Naive Bayes as a base classifier and bigrams as input feature yield the best performance of 93% accuracy. The performance of the Naive Bayes classifier is relatively stable during the experiments with different

Table 1
Adopted evaluation metrics.

Metric	Description
Accuracy	The number of instances classified correctly
Precision	Fraction of relevant instances among retrieved instances
Recall	Fraction of the total amount of relevant instances actually retrieved
F1 score	Harmonic mean of precision and recall

Table 2
Results of traditional classifiers.

		Unigram				Bigram				TF-IDF Bigram				POS Unigram			
		Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
SVM	5%	0,78	0,85	0,60	0,69	0,71	0,74	0,58	0,63	0,70	0,79	0,55	0,60	0,64	0.68	0.60	0.57
	10%	0,80	0,82	0,82	0,81	0,82	0,81	0,84	0,82	**0,91**	0,91	0,92	0,91	0.71	0.70	0.67	0.68
	20%	0,84	0,85	0,82	0,83	0,86	0,87	0,84	0,85	0,88	0,90	0,87	0,88	0.76	0.79	0.71	0.74
	100%	0,87	0,88	0,87	0,87	0,90	0,90	0,90	0,90	0,93	0,94	0,92	0,93	0.87	0.87	0.87	0.87
NB	5%	0,78	0,70	0,86	0,76	0,81	0,74	0,89	0,80	0,70	0,72	0,40	0,50	0.74	0.75	0.71	0.67
	10%	0,87	0,83	0,94	0,88	0,90	0,88	0,92	0,90	0,85	0,87	0,89	0,86	0.82	0.79	0.84	0.81
	20%	**0,90**	0,90	0,91	**0,90**	**0,92**	0,91	0,93	**0,92**	**0,91**	0,90	0,93	**0,92**	**0.90**	0.90	0.89	**0.89**
	100%	**0,92**	0,91	0,93	**0,92**	**0,93**	0,93	0,94	**0,93**	**0,94**	0,93	0,95	**0,94**	0.92	0.92	0.93	0.92
RF	5%	0,68	0,83	0,40	0,50	0,64	0,54	0,41	0,44	0,60	0,56	0,41	0,42	0.53	0.61	0.53	0.45
	10%	0,79	0,83	0,78	0,79	0,84	0,83	0,89	0,84	0,84	0,83	0,86	0,84	0.65	0.67	0.76	0.66
	20%	0,83	0,82	0,87	0,84	0,85	0,86	0,86	0,85	0,84	0,84	0,85	0,84	0.80	0.80	0.81	0.80
	100%	0,85	0,84	0,86	0,85	0,86	0,85	0,86	0,86	0,86	0,86	0,86	0,86	0.84	0.82	0.87	0.84

ratios of labeled data. The bigram models with 20% of labeled data result in a higher accuracy compared to the experiments with 10% or 5% of labeled data.

Other base classifiers' performance (i.e., SVM and RF) steadily increase when more labeled data is used. Only in the experimental setup with TF-IDF values as input features, the base classifier SVM significantly outperforms ($p < 0.05$) base classifier Naive Bayes.

Additional experiments: The self-training experiments are tested on the Yelp hotel review dataset [42] as well. The best performing experiment with limited labeled data instances scores 73% accuracy, in the experimental setup Naïve Bayes and TF-IDF values on 20% labeled data. The self-training experiments on the Yelp restaurant review dataset [43] yielded a maximum accuracy of 66% with 20% of labeled data instances and the SVM classifier with bigrams as input features.

The results of the additional experiments are significantly lower compared to the experiments on the 'gold standard' dataset. This is in line with the results of the traditional supervised classifiers. However, in the case of self-training on the additional datasets [42,43], the semi-supervised setup does not increase performance over the traditional classifiers when the same amount of labeled data is used.

4.2. Co-training

The average results of 5-fold cross-validation experiments for the co-training model are shown in Table 4. In general, co-training does not outperform the self-training. However, there are significant differences in performance under different conditions. Only for experiments where lower amounts of labeled data are used, co-training outperforms some traditional classifiers.

When the linguistic feature set X2 is used, co-training with SVM significantly outperforms ($p < 0.05$) co-training with Naive Bayes (NB) as the base classifier. However, when co-training is applied on X1 randomly split into two feature sets, Naive Bayes still has the best performance overall. Random Forests with random split X1 as input features perform significantly better ($p < 0.05$) during the co-training experiments compared to self-training experiments. SVM or NB in co-training experiments performs better than Random Forests. TF-IDF values only improve results for SVM and the POS tags-based model yields improved results compared to regular unigrams.

Additional experiments: The co-training experiments are tested on the Yelp hotel review dataset [42]. The best performing experiment, SVM with TF-IDF values on 20% of labeled data instances, scores 69% accuracy. The co-training experiments on the Yelp restaurant review dataset [43] yielded a maximum accuracy of 69% on 20% of labeled data instances and the SVM classifier with TF-IDF bigrams as input features.

The results of the additional experiments are significantly lower compared to the experiments on the 'gold standard' dataset. The semi-supervised co-training setup does not increase performance over the traditional classifiers when the same amount of labeled data is used.

4.3. TSVM

Table 5 presents the average results of 5-fold cross-validation experiments for the TSVM model. TSVM models on unigrams, bigrams, and POS tagged unigrams perform significantly better than the traditional SVM model ($p < 0.05$). Especially, TSVM performs well on low ratios of labeled data. However, the model does not outperform the traditional Naive Bayes classifier at any ratio level.

TF-IDF bigram input features-based models with default hyperparameters yield the best performance, as shown in Table 9. POS tags as input features reduce the performance compared to regular unigrams, which is remarkable because all other semi-supervised models yield increased performance with POS tags compared to regular unigrams. The TSVM method is considered an improvement to the self-training approach with SVM as a base estimator. This is confirmed with our results. TSVM outperforms the self-training with SVM in the setting with unigrams, bigrams, and POS tags on every ratio of labeled data.

Additional experiments: TSVM experiments on the Yelp hotel review dataset [42] reached a maximum of 65% accuracy in the setup with bigrams as input features and 20% of the labeled data used. The same experimental setup on the Yelp restaurant dataset [43] reached 64% accuracy. The results are significantly lower compared to the experiments on the 'gold standard' dataset. The TSVM setup does not increase performance over the traditional classifiers when the same amount of labeled data is used.

Table 3
Results of self-training models.

		Unigram				Bigram				TF-IDF Bigram				POS Unigram			
		Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
Self-Training + SVM	5%	0,69	0,71	0,63	0,67	0,71	0,73	0,66	0,69	0,76	0,79	0,7	0,75	0.68	0.71	0.61	0.65
	10%	0,74	0,78	0,67	0,72	0,78	0,82	0,72	0,76	0,84	0,9	0,76	0,82	0.76	0.75	0.78	0.77
	20%	0,77	0,79	0,73	0,76	0,82	0,85	0,77	0,81	**0,9**	0,92	0,89	**0,9**	0.77	0.79	0.75	0.77
Self-Training + NB	5%	0,85	0,84	0,87	0,85	0,9	0,9	0,91	0,9	0,67	0,6	0,63	0,59	0.91	0.90	0.93	0.91
	10%	0,9	0,9	0,9	0,9	0,92	0,92	0,91	0,92	0,66	0,97	0,35	0,45	0.91	0.89	0.93	0.91
	20%	**0,91**	0,91	0,91	**0,91**	**0,93**	0,93	0,92	**0,93**	0,71	0,58	0,43	0,49	**0.91**	0.90	0.93	**0.92**
Self-Training + RF	5%	0,5	0,1	0,2	0,13	0,5	0,9	0,21	0,14	0,51	0,52	0,03	0,06	0.54	0.23	0.37	0.28
	10%	0,53	0,73	0,06	0,11	0,56	0,78	0,18	0,25	0,6	0,76	0,33	0,41	0.53	0.61	0.81	0.57
	20%	0,62	0,85	0,29	0,42	0,66	0,84	0,4	0,53	0,71	0,78	0,58	0,66	0.64	0.76	0.54	0.54

Table 4
Results of co-training models.

		X1 = Bigram X2 = Ling. Feat.				X1 = TF-IDF X2 = Ling. Feat.				Random Split X1 Bigrams				Random Split POS Unigram			
		Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
Co-Training + SVM	5%	0,64	0,67	0,68	0,65	0,58	0,72	0,48	0,46	0,68	0,72	0,65	0,66	0.70	0.70	0.70	0.70
	10%	0,74	0,78	0,67	0,72	0,73	0,86	0,56	0,67	0,78	0,80	0,74	0,77	0.73	0.70	0.81	0.75
	20%	**0,81**	0,82	0,80	**0,71**	**0,88**	0,90	0,85	**0,87**	0,83	0,86	0,79	0,82	0.81	0.79	0.85	0.82
Co-Training + NB	5%	0,65	0,70	0,71	0,63	0,56	0,60	0,39	0,46	0,77	0,88	0,64	0,73	0.71	0.69	0.80	0.73
	10%	0,66	0,69	0,67	0,65	0,56	0,55	0,48	0,50	0,79	0,76	0,87	0,81	0.73	0.70	0.81	0.75
	20%	0,78	0,73	0,91	0,81	0,59	0,60	0,55	0,57	**0,87**	0,87	0,89	**0,87**	**0.82**	0.81	0.85	**0.82**
Co-Training + RF	5%	0,56	0,59	0,62	0,51	0,57	0,72	0,44	0,41	0,66	0,78	0,61	0,60	0.70	0.71	0.77	0.71
	10%	0,63	0,79	0,45	0,51	0,65	0,84	0,40	0,49	0,75	0,86	0,64	0,72	0.70	0.64	0.94	0.76
	20%	0,76	0,75	0,78	0,76	0,77	0,78	0,75	0,76	0,82	0,83	0,82	0,82	0.80	0.73	0.94	0.82

Table 5
Results of the TSVM models.

		Unigram				Bigram				TF-IDF Bigram				POS Unigram			
		Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
TSVM with QNS3VM	5%	0.75	0.73	0.80	0.76	0.76	0.74	0.81	0.77	0.53	0.31	0.60	0.41	0.74	0.78	0.69	0.73
	10%	0.78	0.75	0.83	0.79	0.81	0.78	0.87	0.82	0.53	0.51	0.60	0.42	0.76	0.76	0.77	0.76
	20%	**0.83**	0.80	0.86	**0.83**	**0.83**	0.81	0.87	**0.84**	0.54	0.52	0.22	0.20	**0.81**	0.81	0.81	**0.81**

4.4. Label propagation and spreading

The average results of 5-fold cross-validation experiments for both label propagation and label spreading are shown in Table 6. Every single experimental setup performs poorly. One reason for this result might be the fact that the algorithm classifies data instances based on complex internal similarity measures and in this experiment, the topic similarity is measured over opinion spam indicators similarity. All experimental setups yield significantly worse performance ($p > 0.05$) than traditional methods. Bigrams and TF-IDF bigrams got slightly better results than unigrams or POS unigrams.

Additional dataset experiments: The label propagation and label spreading experiments are tested on the Yelp hotel review dataset [42]. The highest accuracy score is 56% on 20% labeled data. The experiments on the Yelp restaurant review dataset [43] on 20% labeled data yielded a maximum accuracy of 53%. Label propagation and spreading do not increase performance over the traditional classifiers when the same amount of labeled data is used.

4.5. Grid search

All previously described results are retrieved with the Grid Search optimization technique applied to find optimal hyperparameters. Grid Search is applied for each experimental setup (i.e., each ratio of labeled data instances). Resulting hyperparameters are used to obtain results for all experimental setups. Table 7 shows the optimal hyperparameters that result from Grid Search and that are used throughout the results. The hyperparameters were obtained iteratively, by testing different parameters to eventually obtain the optimal outcome. The hyperparameter spaces that are explored for each algorithm are shown in Table 8. In the case of self-training and co-training, only the hyperparameters of the base estimators SVM, NB, and RF are optimized. Table 9 shows the results of the models with default hyperparameters.

The input features of co-training in Table 9 are randomly split X1 values, the created linguistic features X2 are left out here. In general, differing hyperparameters dramatically influence results. For unigrams, POS tagged unigrams, and bigrams, experimental setups conducted with default hyperparameters yield worse performance compared to grid search optimized hyperparameters. However, the default setup of TSVM with TF-IDF input features increases performance dramatically. TSVM seems to heavily rely on hyperparameters tuning, which is not tuned specifically for TF-IDF values.

4.6. Aggregate results

Fig. 3 shows the mean accuracy of the traditional supervised and semi-supervised methods with their best-performing conditions. The results show that self-training is the only method that significantly outperforms traditional supervised classifiers. The 5% ($M = 0.90$, $SD = 0.0013$) experiment of the best performing self-training model performs significantly better ($p < 0.05$) than the 5% ($M = 0.81$, $SD = 0.0625$) experiment of the best performing traditional supervised model. The 10% ($M = 0.92$, $SD = 0.014$) and 20% ($M = 0.93$, $SD = 0.011$) self-training experiments perform better than the 10% ($M = 0.90$, $SD = 0.0600$) and 20% ($M = 0.92$,

Table 6
Results of label propagation and label spreading.

		Unigram				Bigram				TF-IDF Bigram				POS Unigram			
		Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
Label Propagation	5%	0,50	0,00	0,00	0,00	0,05	0,00	0,00	0,00	0,50	0,00	0,00	0,00	0.49	0.40	0.80	0.52
	10%	0,50	0,00	0,00	0,00	0,51	0,17	0,01	0,03	0,53	0,12	0,17	0,14	0.49	0.40	0.80	0.53
	20%	0,61	0,59	0,85	0,68	0,57	0,49	0,54	0,48	0,55	0,41	0,27	0,30	0.51	0.51	1.00	0.67
Label Spreading	5%	0,52	0,37	0,07	0,12	0,51	0,26	0,10	0,12	0,52	0,65	0,39	0,32	0.53	0.44	0.68	0.51
	10%	0,50	0,20	0,00	0,00	0,52	0,18	0,04	0,07	0,55	0,56	0,13	0,15	0.53	0.62	0.74	0.53
	20%	0,65	0,72	0,68	0,62	0,61	0,57	0,54	0,50	0,60	0,42	0,49	0,42	0.51	0.51	1.00	0.67

Table 7
Custom hyperparameters.

% labels	Method	Custom sklearn parameters
5%	SVM	SVC(probability = True, kernel = 'rbf', gamma = 'scale', C = 3)
	NB	MultinomialNB(alpha = 0.2)
	RF	RandomForestClassifier(n_estimators = 150, max_depth = 60, min_samples_split = 20)
	TSVM	TSVM(kernel = 'rbf',C = 0.0002, gamma = 20)
	Label Propagation	LabelPropagation(kernel = 'knn', n_neighbors = 10)
	Label Spreading	LabelSpreading(kernel = 'knn', n_neighbors = 30)
10%	SVM	SVC(probability = True, kernel = 'rbf', gamma = 'scale', C = 9)
	NB	MultinomialNB(alpha = 0.7)
	RF	RandomForestClassifier(n_estimators = 150, max_depth = 30, min_samples_split = 20)
	TSVM	TSVM(kernel = 'rbf',C = 0.0002, gamma = 20)
	Label Propagation	LabelPropagation(kernel = 'knn', n_neighbors = 200)
	Label Spreading	LabelSpreading(kernel = 'knn', n_neighbors = 200)
20%	SVM	SVC(probability = True, kernel = 'rbf', gamma = 'scale', C = 3)
	NB	MultinomialNB(alpha = 0.4)
	RF	RandomForestClassifier(n_estimators = 150, max_depth = 30, min_samples_split = 20)
	TSVM	TSVM(kernel = 'rbf',C = 0.0002, gamma = 20)
	Label Propagation	LabelPropagation(kernel = 'knn', n_neighbors = 50)
	Label Spreading	LabelSpreading(kernel = 'knn', n_neighbors = 150)

Table 8
Hyperparameter spaces.

Algorithm	Hyperparameters	Values explored
SVM	C	0.00001 to 10
	Gamma	scale, auto
NB	alpha	0.001 to 1
RF	n_estimators	10 to 250
	max_depth	10 to 200
	min_samples_split	2 tot 100
TSVM	C	0.00001 to 10
	Gamma	scale, auto
Label Propagation	Kernel	knn, rbf
	n_neighbors	1 to 250
Label Spreading	Kernel	knn, rbf
	n_neighbors	1 to 250

Table 9
Results based on default hyperparameters.

		Unigram				Bigram				TF-IDF Bigram				POS Unigram			
		Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
SVM	10%	0.57	0.00	0.00	0.00	0.57	0.00	0.00	0.00	0.57	0.00	0.00	0.00	0.55	0.55	0.99	0.69
NB	10%	0.80	0.77	0.78	0.77	0.87	0.89	0.81	0.85	0.59	0.60	0.06	0.11	0.87	0.85	0.91	0.87
RF	10%	0.64	0.64	0.42	0.48	0.61	0.59	0.30	0.39	0.65	0.66	0.48	0.52	0.74	0.75	0.77	0.75
Self-Training + SVM	10%	0.49	0.40	0.80	0.53	0.49	0.40	0.80	0.53	0.49	0.40	0.80	0.53	0.50	0.00	0.00	0.00
Self-Training + NB	10%	0.87	0.85	0.89	0.87	0.91	0.90	0.92	0.91	0.69	0.73	0.82	0.69	0.91	0.89	0.93	0.91
Self-Training + RF	10%	0.50	0.40	0.80	0.53	0.52	0.61	0.78	0.54	0.54	0.61	0.80	0.59	0.50	0.00	0.00	0.00
Co-Training + SVM	10%	0.58	0.58	0.40	0.41	0.57	0.79	0.40	0.37	0.50	0.30	0.60	0.40	0.53	0.56	0.07	0.10
Co-Training + NB	10%	0.75	0.76	0.77	0.75	0.78	0.77	0.83	0.79	0.75	0.87	0.63	0.71	0.71	0.77	0.70	0.70
Co-Training + RF	10%	0.65	0.68	0.63	0.64	0.70	0.72	0.66	0.69	0.66	0.71	0.56	0.62	0.67	0.70	0.57	0.63
TSVM	10%	0.49	0.20	0.40	0.26	0.49	0.20	0.40	0.26	0.88	0.90	0.85	0.87	0.48	0.29	0.60	0.39
Label Propagation	10%	0.50	0.00	0.00	0.00	0.50	0.00	0.00	0.00	0.56	0.77	0.15	0.17	0.50	0.00	0.00	0.00
Label Spreading	10%	0.50	0.00	0.00	0.00	0.50	0.00	0.00	0.00	0.51	0.10	0.01	0.02	0.50	0.00	0.00	0.00

SD = 0.0305) traditional supervised experiments, but the results are not significant. The best performing experimental setups are shown in Table 10 and corresponding results are visualized in Fig. 3.

Table 10
Best performing models.

Method	Features	Base Classifier	Accuracy			
			5%	10%	20%	100%
Traditional Supervised	Bigrams	Naive Bayes	0,81	0,90	0,92	0,93
Self-training	Bigrams	Naive Bayes	0,90	0,92	0,93	–
Co-training	TF-IDF + Ling. Feat.	SVM	0,58	0,73	0,88	–
TSVM	TF-IDF Bigrams	–	0.76	0.81	0.83	–
Label Spreading	TF-IDF Bigrams	–	0,52	0,50	0,65	–

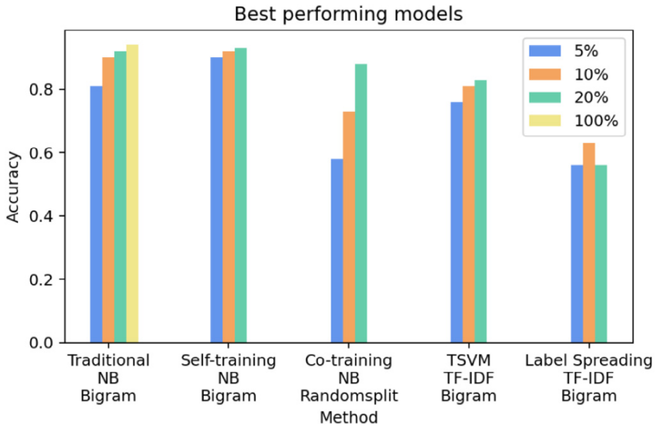

Fig. 3. Best performing models.

4.7. Positive polarity vs. negative polarity training

While half of the data instances in the 'gold standard' dataset is positive in polarity and the other half is negative in polarity. Table 11 shows the results of the models with the best performing experimental setups when only the positive reviews are included. Table 12 shows the results when only negative reviews are included. Traditional NB, self-training, and TSVM perform significantly better ($p < 0.05$) when either only positive polarity or only negative polarity data instances are used. Co-training in both the positive-only and negative-only experiments does not improve the performance.

5. Discussion

This study aims to measure the effectiveness of semi-supervised methods for opinion spam classification. To provide an extensive and complete overview, four semi-supervised methods were tested and compared with traditional supervised classification methods. Experimental setups with a variety of input features, ratios of labeled data instances, and performance metrics were explored in detail. The results indicate that the self-training algorithm is the only semi-supervised method that generally outperforms the traditional supervised classification methods when the same amount of labeled data instances is used. Co-training, transductive SVM, and label propagation models do not perform better than traditional supervised classifiers.

Machine learning models generally perform better when more labeled data is fed into the model. The results of most experimental setups confirmed this statement. However, the traditional supervised SVM model performs slightly better in the scenario with 10% labeled data compared to 20% labeled data, which is a remarkable finding. This study presents the average results of 5-fold cross-validation experiments. In the case of the 5-fold cross-validation evaluation approach, experimental results support the use of 20% labeled data.

Co-training models are known to outperform self-training models, even when there is no natural independent split in the feature sets available [17]. In this study, the co-training models with randomly split unigram or bigram features do not increase performance over self-training or even the traditional supervised models. The co-training experiments would likely perform better if a natural split in the feature set existed. Also, an increase in performance is expected when both feature sets would contain 1000 items instead of dividing the existing feature set into two sets of 500 items. The models with an artificially created feature set X2 with linguistic features as input perform better with base estimators SVM and random forests, but still do not outperform self-training with Naive Bayes. SVM and Random Forests generally perform better when data is dense, like when the linguistic feature set X2 is introduced.

Transductive SVM is considered an improvement to the self-training with SVM as the base classifier, that lets newly labeled data instances float and allows labels to switch during the labeling process before determining the best performing model. The results of the experiments partly confirm the improvement of TSVM over self-training with SVM. On every ratio of labeled data and every input feature, except for TF-IDF values, TSVM has improved results over self-training with SVM.

Label propagation and spreading perform poorly compared to other classifiers, which is expected. The algorithm propagates labels through densely populated areas of unlabeled data. Due to the sparse nature or one-hot-encoded text, these areas can be hard to define. The poor results are likely due to the algorithm targeting the wrong type of similarity in unlabeled data instances. For instance, topic similarity can be favored over opinion spam indicators when propagating labels [47].

In this study, Grid Search is applied to find the optimal hyperparameters for each base classifier or semi-supervised method. When the results of the experiments with Grid Search derived hyperparameters are compared to the experiments with default scikit-learn hyperparameters, significant differences in performance are revealed. Most methods show dramatic increases in performance due to optimized hyperparameters. However, the TSVM with TF-IDF input features performs better when default hyperparameters are used, compared to the same model using optimized hyperparameters for unigrams. This might be because the optimized hyperparameters for regular unigrams are used for TF-IDF unigrams experiments as well. For optimal results, Grid Search should be applied for the TF-IDF input features as well.

To test the generalization of the results, experiments on additional datasets are conducted. The different models perform much better on the 'gold standard' dataset [22] compared to the two Yelp review datasets [42,43]. A reason for this might be that the 'gold standard' deceptive reviews are crowdsourced, where the deceptive Yelp reviews are commercial fake reviews, placed by manipulative individuals. Also, the distribution of positive and negative polarity data instances is not balanced for the Yelp datasets, which might cause a decrease in the model's performance. The differences in performance of the models indicate that the research findings cannot be generalized well to real-life scenarios such as online review websites.

Table 11
Results based on only positive polarity data instances.

		Unigram				Bigram				TF-IDF Bigram				POS Unigram			
		Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
NB	5%	0.65	0.67	0.89	0.73	0.70	0.72	0.87	0.76	0.62	0.62	0.10	0.74	0.82	0.80	0.55	0.61
	10%	0.81	0.76	0.94	0.83	0.84	0.80	0.92	0.84	0.63	0.64	0.97	0.73	0.86	0.87	0.81	0.83
	20%	0.89	0.88	0.92	0.90	**0.92**	0.93	0.93	**0.93**	0.90	0.85	0.99	0.91	0.87	0.83	0.94	0.88
Self-Training + NB	5%	0.93	0.90	0.96	0.93	0.94	0.93	0.95	0.94	0.58	0.58	0.98	0.71	0.93	0.91	0.95	0.93
	10%	0.92	0.91	0.94	0.92	0.94	0.94	0.94	0.94	0.59	0.63	0.87	0.66	0.93	0.92	0.95	0.93
	20%	0.92	0.91	0.94	0.92	**0.94**	0.94	0.94	**0.94**	0.86	0.95	0.78	0.85	**0.94**	0.93	0.95	**0.94**
TSVM	5%	0.74	0.72	0.79	0.75	0.78	0.77	0.80	0.78	0.55	0.36	0.57	0.43	0.75	0.72	0.84	0.77
	10%	0.81	0.79	0.84	0.82	0.81	0.81	0.82	0.81	0.49	0.10	0.20	0.13	0.78	0.76	0.81	0.78
	20%	0.84	0.84	0.84	0.84	**0.86**	0.88	0.84	**0.86**	0.55	0.14	0.20	0.16	0.82	0.78	0.88	0.83
Label Propagation	5%	0.53	0.21	0.40	0.28	0.51	0.20	0.01	0.02	0.48	0.49	0.61	0.41	0.51	0.11	0.16	0.13
	10%	0.50	0.50	0.10	0.67	0.51	0.40	0.40	0.27	0.53	0.38	0.48	0.37	0.51	0.10	0.20	0.14
	20%	0.56	0.58	0.87	0.65	0.54	0.50	0.63	0.47	0.53	0.50	0.66	0.49	0.58	0.51	0.68	0.53
		X1 = Bigram X2 = Ling. Feat.				X1 = TF-IDF X2 = Ling. Feat.				Random Split X1 Bigrams				Random Split POS Unigram			
		Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
Co-Training + NB	5%	0.63	0.65	0.70	0.64	0.57	0.57	0.60	0.57	0.70	0.80	0.68	0.69	0.65	0.63	0.90	0.72
	10%	0.73	0.67	0.94	0.78	0.60	0.61	0.62	0.59	0.78	0.83	0.77	0.77	0.79	0.81	0.80	0.79
	20%	0.78	0.71	0.95	0.81	0.61	0.63	0.56	0.59	0.82	0.83	0.85	0.82	**0.84**	0.89	0.80	**0.83**

Table 12
Results based on only negative polarity data instances.

		Unigram				Bigram				TF-IDF Bigram				POS Unigram			
		Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
NB	5%	0.68	0.68	0.66	0.63	0.75	0.73	0.75	0.72	0.45	0.34	0.38	0.27	0.88	0.82	0.96	0.88
	10%	0.67	0.69	0.80	0.69	0.73	0.74	0.85	0.76	0.54	0.63	0.85	0.63	0.78	0.76	0.93	0.83
	20%	0.82	0.79	0.91	0.84	**0.89**	0.88	0.92	**0.90**	0.66	0.64	0.99	0.76	0.84	0.79	0.94	0.85
Self-Training + NB	5%	0.87	0.82	0.97	0.88	0.94	0.91	0.98	0.94	0.51	0.41	0.80	0.54	0.89	0.83	0.97	0.89
	10%	0.91	0.87	0.95	0.91	0.95	0.93	0.97	0.95	0.67	0.84	0.50	0.53	0.89	0.84	0.96	0.90
	20%	0.91	0.89	0.95	0.92	**0.95**	0.94	0.97	**0.96**	0.82	0.93	0.71	0.77	0.92	0.88	0.97	0.92
TSVM	5%	0.65	0.65	0.69	0.65	0.66	0.68	0.66	0.65	0.49	0.30	0.60	0.40	0.70	0.72	0.68	0.69
	10%	0.76	0.74	0.81	0.77	0.79	0.78	0.81	0.80	0.51	0.31	0.60	0.41	0.76	0.76	0.77	0.76
	20%	0.81	0.81	0.80	0.81	**0.84**	0.85	0.83	**0.84**	0.53	0.60	0.47	0.37	0.80	0.81	0.80	0.80
Label Propagation	5%	0.48	0.29	0.20	0.13	0.48	0.09	0.20	0.13	0.52	0.20	0.01	0.01	0.50	0.20	0.40	0.27
	10%	0.51	0.20	0.40	0.27	0.53	0.22	0.39	0.28	0.50	0.20	0.01	0.01	0.53	0.53	0.88	0.64
	20%	0.51	0.40	0.40	0.28	0.51	0.20	0.40	0.27	0.57	0.35	0.59	0.44	0.52	0.50	0.61	0.44
		X1 = Bigram X2 = Ling. Feat.				X1 = TF-IDF X2 = Ling. Feat.				Random Split X1 Bigrams				Random Split POS Unigram			
		Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
Co-Training + NB	5%	0.59	0.59	0.67	0.57	0.54	0.54	0.59	0.56	0.67	0.77	0.62	0.62	0.65	0.58	0.66	0.58
	10%	0.70	0.72	0.78	0.70	0.55	0.54	0.55	0.54	0.75	0.73	0.88	0.78	0.71	0.75	0.72	0.70
	15%	0.68	0.62	0.94	0.75	0.51	0.50	0.62	0.53	**0.82**	0.79	0.91	**0.84**	0.76	0.71	0.92	0.79

The results of the best performing models are compared to the same experimental setups trained on only positive polarity or only negative polarity data instances. The models perform better when trained on only one type of polarity. A possible explanation for this phenomenon is that the data instances that are fed to the model are now likely to be more similar to each other. Lesser variations to the training data could mean that the features that are strong predictors of class labels are now theoretically more often seen and therefore, the classifier can be more confident about the output label (i.e., high F-scores in the chi-square test). For real-world applications, if the polarity of the new data and training data is known with high confidence, performance will increase when two different models are trained on both positive polarity and negative polarity data instances. However, differences in performance are minor, and training a single model is more efficient.

There is an almost infinite number of possible experimental designs with differing conditions (i.e., approaches, algorithms, datasets, input features, features selection methods, evaluation metrics, partitions of the training and testing sets, etc.). A limited amount of experimental designs must be picked to retain the scope of the research. Including more designs could improve the validity of this research. However, current results offer very useful conclusions for practitioners and researchers in this field. The effectiveness of a variety of methods is explored in a way that it can be reproduced by other researchers. It can be concluded that for opinion spam classification problems, the deployed self-training models increase performance and mitigate the labor-intensive problem of labeling additional data instances.

6. Conclusion

In this study, the effectiveness of semi-supervised learning methods for opinion spam classification is explored with the help of the gold-standard dataset of hotel reviews developed by Ott et al. (2011) and two additional Yelp review datasets. Results show that the self-training algorithm can outperform traditional supervised classification methods when limited labeled data is available. Self-training on the ‘gold standard’ dataset with Naive Bayes as the base classifier yields the best overall performance of 93% accuracy. Both self-training with Naive Bayes as the base classifier and traditional supervised Naive Bayes score a maximum of 73% accuracy on the Yelp datasets.

Both semi-supervised and traditional supervised models perform well beyond the capabilities of human judges in opinion spam classification. The proposed semi-supervised approaches can mitigate labeling efforts while retaining high performance, which is useful for scenarios where retrieving labeled data is costly. However, the results cannot be generalized well to real-life scenarios. Possible directions for future research include exploring unsupervised approaches for opinion spam detection. Considering meta-data and real-life knowledge about reviews and reviewers can be rewarding for both unsupervised and supervised opinion spam detection. In some scenarios where binary classification is inadequate, multi-class classification is required. This is considered a challenging topic and requires more research efforts, especially when including semi-supervised learning. Furthermore, different application domains for semi-supervised opinion spam classification can be explored. It would be interesting to compare results on real-world datasets to the results of human judges.

CRediT authorship contribution statement

Alexander Ligthart: Conceptualization, Data curation, Writing - review & editing. **Cagatay Catal:** Methodology, Validation, Writing - review & editing. **Bedir Tekinerdogan:** Methodology, Validation, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] A. Bondielli, F. Marcelloni, A survey on fake news and rumour detection techniques, Inform. Sci. 497 (2019) 38–55, http://dx.doi.org/10.1016/j.ins.2019.05.035.

[2] S. Vosoughi, D. Roy, S. Aral, The spread of true and false news online, Science 359 (6380) (2018) 1146–1151, http://dx.doi.org/10.1126/science.aap9559.

[3] O. Chapelle, A. Zien, Semi-supervised classification by low density separation, 2005, p. 8.

[4] M. Crawford, T.M. Khoshgoftaar, J.D. Prusa, A.N. Richter, H. Al Najada, Survey of review spam detection using machine learning techniques, J. Big Data 2 (1) (2015) 23, http://dx.doi.org/10.1186/s40537-015-0029-9.

[5] H. Chen, J. Liu, Y. Lv, M.H. Li, M. Liu, Q. Zheng, Semi-supervised clue fusion for spammer detection in Sina Weibo, Inf. Fusion 44 (2018) 22–32, http://dx.doi.org/10.1016/j.inffus.2017.11.002.

[6] F. Hemmatian, M.K. Sohrabi, A survey on classification techniques for opinion mining and sentiment analysis, Artif. Intell. Rev. 52 (3) (2017) 1495–1545, http://dx.doi.org/10.1007/s10462-017-9599-6.

[7] Y.C.A. Padmanabha Reddy, P. Viswanath, B. Eswara Reddy, Semi-supervised learning: A brief review, Int. J. Eng. Technol. 7 (1.8) (2018) 81, http://dx.doi.org/10.14419/ijet.v7i1.8.9977.

[8] S.S. Sawant, M. Prabukumar, A review on graph-based semi-supervised learning methods for hyperspectral image classification, Egypt. J. Remote Sens. Space Sci. (2018) http://dx.doi.org/10.1016/j.ejrs.2018.11.001.

[9] A.B. Goldberg, X. Zhu, Seeing stars when there aren't many stars: Graph-based semi-supervised learning for sentiment categorization, 2006, p. 8.

[10] G. Giasemidis, N. Kaplis, I. Agrafiotis, J.R. Nurse, A semi-supervised approach to message stance classification, IEEE Trans. Knowl. Data Eng. 32 (1) (2018) 1–11.

[11] Y. Yang, M.O. Shafiq, Large scale and parallel sentiment analysis based on Label Propagation in Twitter Data, in: 2018 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering, TrustCom/BigDataSE, IEEE, 2018, pp. 1791–1798, August.

[12] Yarowsky D., Unsupervised word sense disambiguation rivaling supervised methods, in: Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics, 1995, pp. 189–196, http://dx.doi.org/10.3115/981658.981684.

[13] M. Pavlinek, V. Podgorelec, Text classification method based on self-training and LDA topic models, Expert Syst. Appl. 80 (2017) 83–93, http://dx.doi.org/10.1016/j.eswa.2017.03.020.

[14] A. Blum, T. Mitchell, Combining labeled and unlabeled data with co-training, in: Proceedings of the Eleventh Annual Conference on Computational Learning Theory, COLT' 98, 1998, pp. 92–100, http://dx.doi.org/10.1145/279943.279962.

[15] J.K. Rout, A. Dalmia, K.-K.R. Choo, S. Bakshi, S.K. Jena, Revisiting semi-supervised learning for online deceptive review detection, IEEE Access 5 (2017) 1319–1327, http://dx.doi.org/10.1109/ACCESS.2017.2655032.

[16] F. Li, M. Huang, Y. Yang, X. Zhu, Learning to identify review spam, 2011, p. 6.

[17] K. Nigam, R. Ghani, Analyzing the effectiveness and applicability of co-training, in: Proceedings of the Ninth International Conference on Information and Knowledge Management, CIKM '00, 2000, pp. 86–93, http://dx.doi.org/10.1145/354756.354805.

[18] S.D. Bhattacharjee, W.J. Tolone, V.S. Paranjape, Identifying malicious social media contents using multi-view Context-Aware active learning, Future Gener. Comput. Syst. 100 (2019) 365–379.

[19] G. Li, S.C. Hoi, K. Chang, Two-view transductive support vector machines, in: Proceedings of the 2010 SIAM International Conference on Data Mining, Society for Industrial and Applied Mathematics, 2010, pp. 235–244, April.

[20] N. Kamal, M. Andrew, M. Tom, Semi-supervised text classification using EM, in: O. Chapelle, B. Scholkopf, A. Zien (Eds.), Semi-Supervised Learning, The MIT Press, 2006, pp. 32–55, http://dx.doi.org/10.7551/mitpress/9780262033589.003.0003.

[21] A. Mukherjee, V. Venkataraman, Opinion Spam Detection: An Unsupervised Approach using Generative Models, Techincal Report, UH, 2014.

[22] M. Ott, Y. Choi, C. Cardie, J.T. Hancock, Finding deceptive opinion spam by any stretch of the imagination, in: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Vol. 1, Association for Computational Linguistics, 2011, pp. 309–319, June.

[23] J. Karimpour, A.A. Noroozi, S. Alizadeh, Web spam detection by learning from small labeled samples, Int. J. Comput. Appl. 50 (21) (2012) 1–5, http://dx.doi.org/10.5120/7924-0993.

[24] R. Hassan, Md.R. Islam, Detection of fake online reviews using semi-supervised and supervised learning, in: 2019 International Conference on Electrical, Computer and Communication Engineering, ECCE, 2019, pp. 1–5, http://dx.doi.org/10.1109/ECACE.2019.8679186.

[25] R. Narayan, J.K. Rout, S.K. Jena, Review spam detection using semi-supervised technique, in: Progress in Intelligent Computing Techniques: Theory, Practice, and Applications, Springer, Singapore, 2018, pp. 281–286.

[26] X.L. Li, B. Liu, Learning from positive and unlabeled examples with different data distributions, in: European Conference on Machine Learning, Springer, Berlin, Heidelberg, 2005, pp. 218–229, October.

[27] G. Stanton, A.A. Irissappane, GANs for semi-supervised opinion spam detection, 2019, arXiv preprint arXiv:1903.08289.

[28] B. Manaskasemsak, C. Chanmakho, J. Klainongsuang, A. Rungsawang, Opinion spam detection through user behavioral graph partitioning approach, in: Proceedings of the 2019 3rd International Conference on Intelligent Systems, Metaheuristics & Swarm Intelligence, 2019, pp. 73–77, March.

[29] F. Wu, C. Wu, J. Liu, Semi-supervised collaborative learning for social spammer and spam message detection in microblogging, in: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, 2019, pp. 1791–1794, October.

[30] C. Li, S. Wang, L. He, S.Y. Philip, Y. Liang, Z. Li, SSDMV: Semi-supervised deep social spammer detection by multi-view data fusion, in: 2018 IEEE International Conference on Data Mining, ICDM, IEEE, 2018, pp. 247–256, November.

[31] A.R. Yelundur, V. Chaoji, B. Mishra, Detection of review abuse via semi-supervised binary multi-target tensor decomposition, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 2134–2144, July.

[32] H. Alvari, E. Shaabani, S. Sarkar, G. Beigi, P. Shakarian, Less is more: Semi-supervised causal inference for detecting pathogenic users in social media, in: Companion Proceedings of The 2019 World Wide Web Conference, 2019, pp. 154–161, May.

[33] C.M. Yilmaz, A.O. Durahim, SPR2EP: a semi-supervised spam review detection framework, in: 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM, IEEE, 2018, pp. 306–313, August.

[34] B. Wang, J. Huang, H. Zheng, H. Wu, Semi-supervised recursive autoencoders for social review spam detection, in: 2016 12th International Conference on Computational Intelligence and Security, CIS, IEEE, 2016, pp. 116–119, December.

[35] H. Deng, L. Zhao, N. Luo, Y. Liu, G. Guo, X. Wang, et al., Semi-supervised learning based fake review detection, in: 2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications, ISPA/IUCC, IEEE, 2017, pp. 1278–1280, December.

[36] L. Zhang, Y. Yuan, Z. Wu, J. Cao, Semi-SGD: Semi-supervised learning based spammer group detection in product reviews, in: 2017 Fifth International Conference on Advanced Cloud and Big Data, CBD, IEEE, 2017, pp. 368–373, August.

[37] N. Imam, B. Issac, S.M. Jacob, A semi-supervised learning approach for tackling Twitter spam drift, Int. J. Comput. Intell. Appl. 18 (02) (2019) 1950010.

[38] J. Chengzhang, D.K. Kang, Detecting the spam review using tri-training, in: 2015 17th International Conference on Advanced Communication Technology, ICACT, IEEE, 2015, pp. 374–377, July.

[39] M.I. Ahsan, T. Nahian, A.A. Kafi, M.I. Hossain, F.M. Shah, Review spam detection using active learning, in: 2016 IEEE 7th Annual Information Technology, Electronics and Mobile Communication Conference, IEMCON, IEEE, 2016, pp. 1–7, October.

[40] H. Aghakhani, A. Machiry, S. Nilizadeh, C. Kruegel, G. Vigna, Detecting deceptive reviews using generative adversarial networks, in: 2018 IEEE Security and Privacy Workshops, SPW, IEEE, 2018, pp. 89–95, May.

[41] W. Xu, H. Sun, C. Deng, Y. Tan, Variational autoencoder for semi-supervised text classification, in: Thirty-First AAAI Conference on Artificial Intelligence, 2017, February.

[42] A. Mukherjee, V. Venkataraman, B. Liu, N.S. Glance, What yelp fake review filter might be doing? in: ICWSM, 2013, pp. 409–418, July.

[43] S. Rayana, L. Akoglu, Collective opinion spam detection: Bridging review networks and metadata, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015, pp. 985–994, August.

[44] F. Gieseke, A. Airola, T. Pahikkala, K. Oliver, Sparse quasi-newton optimization for semi-supervised support vector machines, in: Proceedings of the 1st International Conference on Pattern Recognition Applications and Methods, 2012, pp. 45–54, http://dx.doi.org/10.5220/0003755300450054.

[45] N.F.F. da Silva, L.F.S. Coletta, E.R. Hruschka, E.R. Hruschka Jr., Using unsupervised information to improve semi-supervised tweet sentiment classification, Inform. Sci. 355–356 (2016) 348–365, http://dx.doi.org/10.1016/j.ins.2016.02.002.

[46] C. Nadeau, Y. Bengio, Inference for the generalization error, in: Advances in Neural Information Processing Systems, 2000, pp. 307–313.

[47] N.F.F.D. Silva, L.F.S. Coletta, E.R. Hruschka, A survey and comparative study of tweet sentiment analysis via semi-supervised learning, ACM Comput. Surv. 49 (1) (2016) 1–26, http://dx.doi.org/10.1145/2932708.