**Semi-Supervised Spam Detection in Twitter Stream**

**ABSTRACT**

Most existing techniques for spam detection on Twitter aim to identify and block users who post spam tweets. In this paper, we propose a semi-supervised spam detection (S3D) framework for spam detection at tweet-level. The proposed framework consists of two main modules: spam detection module operating in real-time mode and model update module operating in batch mode. The spam detection module consists of four lightweight detectors: 1) blacklisted domain detector to label tweets containing blacklisted URLs; 2) near-duplicate detector to label tweets that are near-duplicates of confidently pre labeled tweets; 3) reliable ham detector to label tweets that are posted by trusted users and that do not contain spammy words; and 4) multiclassifier-based detector labels the remaining tweets. The information required by the detection module is updated in batch mode based on the tweets that are labeled in the previous time window. Experiments on a large-scale data set show that the framework adaptively learns patterns of new spam activities and maintain good accuracy for spam detection in a tweet stream.

**EXISTING SYSTEM**

* Many social spam detection studies focus on the identification of spam accounts. Lee et al. [14] analyzed and used features derived from user demographics, follower/following social graph, tweet content, and the temporal aspect of user behavior to identify content polluters.
* Hu et al. [13] exploited social graph and tweets of a user to detect spam detection

on Twitter. They formulated spammer detection task as an optimization problem. Online learning has been utilized to tackle the fast evolving nature of spammer [12]. They have utilized both content and network information and incrementally updated their spam detection model for effective social spam detection.

* Tan et al. [21] proposed an unsupervised spam detection system that exploits legitimate users in the social network. Their analysis shows the volatility of spamming patterns in social network. They have utilized non spam patterns of legitimate users based on social graph and user link graph to detect spam pattern.
* Gao et al. [9] identified social spam by clustering posts based on text and URL

similarities and detected large clusters with bursty posting patterns. Incremental clustering-based approach has been used to detect spam campaigns on Twitter.

**Disadvantages**

* + There is no Semi-supervised learning.
	+ There is no option to find type of different spammers.

**PROPOSED SYSTEM**

* In the proposed system, the system proposes a semi-supervised framework for spam tweet detection. The proposed framework mainly consists of two main modules: 1) four lightweight detectors in the spam tweet detection module for detecting spam tweets in real time and 2) updating module to periodically update the detection models based on the confidently labeled tweets from the previous time window. The detectors are designed based on our observations made from a collection of 14 million tweets, and the detectors are computationally effective, suitable for real-time detection.
* More importantly, our detectors utilize classification techniques at two levels, tweet level and cluster level. Here, a cluster is a group of tweets with similar characteristics. With this flexible design, any features that may be effective in spam detection can be easily incorporated into the detection framework. The framework starts with a small set of labeled samples and updates the detection models in a semi-supervised manner by utilizing the confidently labeled tweets from the previous time window. This semi-supervised approach helps to learn new spamming activities, making the framework more robust in identifying spam tweets.

**Advantages**

* + **Confidently Labeled Tweets**- Tweets that are labeled by the first three detectors (i.e., blacklisted domain, near duplicate and reliable ham tweet) are considered as confidently labeled tweets.
	+ **Near-Duplicate Cluster Labeling** - Recall that the near duplicate detector computes a signature for each tweet to check if the tweet is a near duplicate of a labeled cluster. If the signature of a tweet does not match any relabeled cluster, then the tweet is passed to the next level detectors.

**SYSTEM REQUIREMENTS**

➢ **H/W System Configuration:-**

➢ Processor - Pentium –IV

➢ RAM - 4 GB (min)

➢ Hard Disk - 20 GB

➢ Key Board - Standard Windows Keyboard

➢ Mouse - Two or Three Button Mouse

➢ Monitor - SVGA

**Software Requirements:**

* Operating System - Windows XP
* Coding Language - Java/J2EE(JSP,Servlet)
* Front End - J2EE
* Back End - MySQL