Towards Reliable Online Clickbait Video Detection: A Content-Agnostic Approach

Lanyu Shang, Daniel (Yue) Zhang, Michael Wang, Shuyue Lai, Dong Wang

Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN 46556

Abstract

Online video sharing platforms (e.g., YouTube, Vimeo) have become an increasingly popular paradigm for people to consume video contents. Clickbait video, whose content clearly deviates from its title/thumbnail, has emerged as a critical problem on online video sharing platforms. Current clickbait detection solutions that mainly focus on analyzing the text of the title, the image of the thumbnail, or the content of the video are shown to be suboptimal in detecting the online clickbait videos. In this paper, we develop a novel content-agnostic scheme, Online Video Clickbait Protector (OVCP), to effectively detect clickbait videos by exploring the comments from the audience who watched the video. Different from existing solutions, OVCP does not directly analyze the content of the video and its pre-click information (e.g., title and thumbnail). Therefore, it is robust against sophisticated content creators who often generate clickbait videos that can bypass the current clickbait detectors. We evaluate OVCP with a real-world dataset collected from YouTube. Experimental results demonstrate that OVCP is effective in identifying clickbait videos and significantly outperforms both state-of-the-art baseline models and human annotators.

Keywords: Clickbait Video, Content Agnostic, Online Video Sharing, YouTube

1. Introduction

In the age of instant gratification, people are increasingly consuming more video contents from the Internet (e.g., YouTube[1], Vimeo[2]) than cable networks [1]. The time people spend on online media is expected to surpass the time they spend on traditional TV worldwide in 2019 [2]. For example, YouTube has over a billion users covering almost one-third of the Internet population and reaches billions of views per day [3]. An online video usually includes title, thumbnail, and the video content. The title and thumbnail are visible to the viewers *before* they click the video and are found to be the crucial factors that may attract users to click and watch a video. video, whose content clearly deviates from its title/thumbnail, is generated to entice viewers to click video and boost the viewership of the video consequently [3]. However, the spread of clickbait videos not only wastes the time of viewers but also decreases the trustworthiness of video-sharing platforms. In this paper, we focus on the problem of reliably detecting clickbait videos online video-sharing platforms.

The detection of clickbait videos requires a careful investigation of the relationship between the title/thumbnail and the video content. This problem cannot be fully addressed by current content-based solutions which only focus on the text of the title [4, 5], the image of the thumbnail [6, 7], or the content of the video [8, 9]. For example, several text-based clickbait detection techniques have been developed to identify clickbait from social media posts (e.g., the clickbait news headline detection using word embeddings [4], clickbait tweets detection using the linguistic feature analysis [10]). However, those solutions cannot be adopted to address the video clickbait detection problem because the content of the title may not be a reliable indicator to identify a clickbait video. For example, both videos shown in Figure 1 share the same title. The video shown in Figure 1(a) is a clickbait because it does not deliver the content the

[1] https://www.youtube.com
[2] https://vimeo.com/watch
[3] https://www.youtube.com/yt/about/press/

viewers expect to see (e.g., a plane with 13 engines on each wing as shown in its thumbnail). In contrast, Figure 1(b) is a non-clickbait video and the video does present the ten largest airplanes in the world including the one shown in the thumbnail.

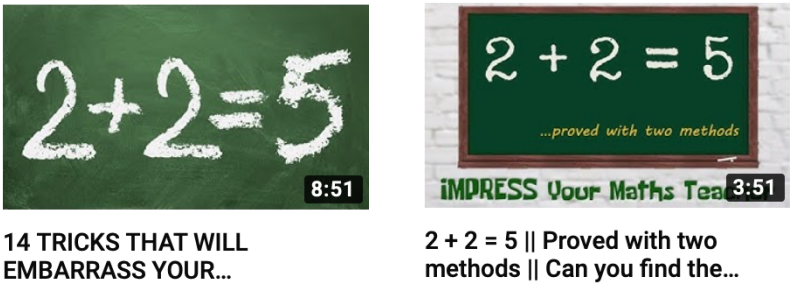
(a) Clickbait (b) Non-clickbait

Figure 1: Examples of Clickbait and Non-clickbait Video with Similar Titles

Similarly, image-based approaches that only focus on thumbnail features (e.g., convolutional method [6], self-consistency [11]) cannot solve the video clickbait detection problem. For example, Figure 2 shows two videos with very similar thumbnails. The video 2(a) in Figure 2 is a clickbait because the video only presents some irrelevant tricks, such as how to plot letters "L", "O", "V", "E" with math functions, and never explains the equation $2+2=5$. In contrast, the video shown in Figure 2(b) is a non-clickbait as its content explicitly discusses some tricky methods to derive the equation $2+2=5$ (e.g., $2\times 0+2\times 0=5\times 0$).

Moreover, video authentication and verification solutions have been developed to identify fake online videos [8] and AI-generated videos [12]. However, these solutions only focus on the video content itself and ignore the information exposed to viewers before the video is clicked (e.g., title thumbnail). For example, an old movie can be uploaded under a title/thumbnail identical or similar to a recently released one. Such a clickbait video can easily bypass the video-based detection systems as the video content is real [13]. Therefore, a reliable online clickbait video detection tool that explicitly considers the relationship

between the title/thumbnail and the video content has yet to be developed.



(a) Clickbait (b) Non-clickbait

Figure 2: Examples of Clickbait and Non-clickbait Video with Similar Thumbnails

In this paper, we develop a novel content-agnostic scheme, *Online Video Clickbait Protector (OVCP)*, to effectively detect clickbait videos whose titles/thumbnails are deviating from the video content. OVCP is content-agnostic in the sense that it does not analyze the content information of a video (i.e., title, thumbnail, video clips). Instead, OVCP investigates the comments and discussions from online users who watched the video. Our approach is motivated by the observation that users usually make complaints or sarcastic comments when they watch clickbait videos (e.g., "My body is crying for help watching this video" in a clickbait video titled "28 SIGNS YOUR BODY IS CRYING FOR HELP"). Additionally, users are observed to be less interactive in the discussion of clickbait videos than non-clickbait ones given the fact that the content of clickbait videos often appears to be tedious to the users [14]. The OVCP scheme explores the topological and semantic structure of the user comment networks and the latent linguistic features of user comments to identify the unique characteristics of clickbait videos. OVCP is then integrated with a set of state-of-the-art supervised classifiers to detect clickbait videos.

To the best of our knowledge, OVCP is the first content-agnostic approach to address the online clickbait video detection problem. OVCP is robust against who often generate clickbait videos that can bypass the current clickbait de-

tectors. The evaluation results on a real-world dataset collected from YouTube show that our OVCP scheme accurately detects clickbait videos and significantly outperforms state-of-the-art baselines and human annotators.

2. Related Work

2.1. Clickbait Detection

Our work bears some resemblance with literature on clickbait detection on social media platforms. The term clickbait was coined as "exaggerated headlines whose main motive is to mislead the reader to click on them" [15]. The problem of clickbait has been widely studied and many solutions were provided [10, 16, 17]. The first automatic clickbait detector was proposed in [10], where a set of handcrafted features (e.g., bag-of-words, n-grams and number of hashtags) have been selected to train a clickbait classifier. The follow-up solutions improve this feature engineering approach by developing deep neural network based approaches to automatically detect clickbaits without cherry picking the features [17, 18]. Unfortunately, these schemes rely on the analysis of the textual content (i.e., titles and descriptors) to detect clickbaits and cannot effectively address our problem because both clickbait videos and non-clickbait videos can share very similar titles and descriptors as shown in Figure 1.

We found there exist very few clickbait detection solutions that focus on video-based clickbaits. In a very recent work, Qu *et al.* developed a crowdsourcing-based approach where human annotators are invited to label whether the thumbnail of a video is clickbait or not [19]. This work requires a significant amount of human labor and can be time-consuming if the labeled dataset is large. The most relevant work was proposed by Zannettou *et al.*, who developed a deep neural network model to combine the features of the thumbnails and the statistical features of users' comments [20]. However, this approach relies on the direct analysis of the thumbnails which has been shown to be an unreliable indicator of clickbait videos (Figure 2). Moreover, the scheme assumes all the videos posted by the same user were clickbaits which is an oversimplified assumption for the

real-world scenarios where sophisticated users exist. In contrast, the proposed OVCP scheme is the first *content-agnostic* approach to address the clickbait video detection problem on video sharing platforms.

2.2. Misinformation Detection

The spread of misinformation on online social media has received a significant amount of attention in recent years [21, 22, 23, 24, 25, 26]. There exist two categories of misinformation detection problems that are related to ours. The first category is called *fake news detection* where *textual content* (e.g., news articles and social media posts) is analyzed to verify whether the news statement is truthful or not. For example, Vo *et al.* developed a fake news detection scheme that can identify a group of users who actively debunk fake information on social media and recommend fact-checking URLs posted from these users [22]. Wang *et al.* developed an estimation theoretical scheme that identifies truthful online social media posts by explicitly considering the reliability of data sources and source dependency [27]. The second category is commonly referred to as "image forgery detection" schemes. For example, Zhang *et al.* developed a novel fauxtography detector that can effectively track down misleading images on social media (e.g., Twitter, Reddit) [28]. Huynh-Kha *et al.* developed an image forgery detection scheme that can detect whether an image is manually edited by copy-move, splicing or both in the same image [29].

However, the above solutions are insufficient to address the clickbait video detection problem in this paper because sophisticated uploaders can often create clickbait videos by using credible titles and non-edited images to bypass the detection systems [20]. In this work, we propose a more robust content-agnostic approach that leverages the audience's comments to detect clickbait videos rather than on the analysis of textual descriptors or thumbnails.

2.3. Network Embedding for Comment Feature Extraction

Extracting topological and semantic information from the comment network is a key technique used in the proposed OVCP scheme. Various relevant network embedding techniques have been proposed [30, 31, 32, 33, 34]. For example,

DeepWalk uses short random walks to learn the latent representations to describe the topological structure of a network [32]. Node2vec designs a biased random walk procedure to learn the representation of a network that maximizes the likelihood of preserving network neighborhoods of a node [31]. These classical methods focus on capturing the topological structure of the nodes while ignoring the attributes of the edge. In a more recent study, Wang *et al.* developed SHINE, a network embedding technique that can capture the positive/negative emotions of the edge in a network [30]. Huang *et al.* further extended the effort by proposing LANE, a general network embedding technique designed for attributed networks [33]. Compared to SHINE, LANE is able to capture general categorical attributes on the edges of a network. Different from the above methods, our work encodes the unique sentiment and endorsement features of the user comment network for the purpose of detecting clickbait videos.

3. Problem Definition

In this section, we present the online clickbait video detection problem. First, we define a few key terms that will be used in the problem formulation.

Definition 1. ***Video*** *(V_i):* *it is a publicly available video instance uploaded to the online sharing platforms (Figure 3). Each video, denoted as V_i, contains five elements: title (T_i), thumbnail (M_i), description (D_i), comments (C_i), and the ground truth label (z_i) of the video. Formally, V_i is denoted as a quintuple $V_i = (T_i, M_i, D_i, C_i, z_i)$. Additionally, we define a video set of N videos as $\mathcal{V} = \{V_1, V_2, \cdots, V_N\}$.*

Definition 2. ***Title*** *(T_i):* *it is a piece of brief textual information provided by the video uploader to describe the video content. An example of a video title is shown in Figure 3(a).*

Definition 3. ***Thumbnail*** *(M_i):* *it is an image that provides a visual description of the video content. YouTube video creators are allowed to choose a thumbnail generated automatically by the YouTube algorithm, which is typically*

10 REAL LIFE GIANTS You Won't Believe Actually EXIST

2.2M views • 2 months ago

(a) Title and Thumbnail

10 REAL LIFE GIANTS You Won't Believe Actually EXIST

2.3M views 8.2K 1.1K SHARE SAVE

Interesting Facts
Published on Oct 2, 2018

SUBSCRIBE 1.1M

▶ Subscribe: https://goo.gl/vHN6qB

For copyright matters please contact us at:

SHOW MORE

(b) Description

OnlinePorkchop 5 days ago

TEN REAL LIFE CLICKBAITERS YOU NEVER KNEW EXISTED

Read more

16 REPLY

Hide replies

j jim crow 3 days ago

Yeah man, I want my 11 minutes back.

1 REPLY

I Ian Mc 1 day ago

bruh

REPLY

(c) Sample Comments

Figure 3: Example of an Online Video and Its Components

a frame from the video, or to upload a customized image. An example of a video thumbnail is shown in Figure 3(a).

Definition 4. ***Description*** *(D_i): it includes the description provided by the video creator, and the video's meta information (e.g., number of views, number of likes). An example of a video description is shown in Figure 3(b).*

Definition 5. ***Comments*** *(C_i): they include all the comments and reviews of a video from its viewers. An example of the comment section of a video is*

shown in Figure 3(c).

Definition 6. ***Clickbait Video (labeled as "true"):*** *a video is defined as clickbait if its title and/or thumbnail is intentionally crafted to attract viewers' attention, and entice them to click the accompanied video whose content clearly deviates from the title and/or thumbnail.*

Definition 7. ***Non-clickbait Video (labeled as "false"):*** *any video that is not in the clickbait video category.*

Based on the above definitions, the clickbait video detection is a *binary classification problem* that targets at classifying each video into two categories, i.e., clickbait or non-clickbait. The problem is formulated as:

$$\arg\max_{\tilde{z}_i} Pr(\tilde{z}_i = z_i | V_i),\ \forall 1 \leq i \leq N \tag{1}$$

where $\tilde{z_n}$ denotes the estimated label for video V_n.

4. Solution

In this section, we present the *Online Video Clickbait Protector (OVCP)* to address the problem formulated in the previous section. Our scheme consists of four major components as shown in Figure 4. This first component is a *Network Feature Extraction* module that extracts topological and semantic features from the audience interactive discussions and comments of a video. The second component is a *Linguistic Feature Extraction* module that learns the vector representations of user comments and extracts linguistic features of the comments (e.g., document embeddings). The third component is a *Metadata Feature Extraction* module that extracts auxiliary metadata features of the video (e.g., number of views, number of likes). The last component is a *Supervised Classification* module that identifies the clickbait videos using the features extracted from the first three components.

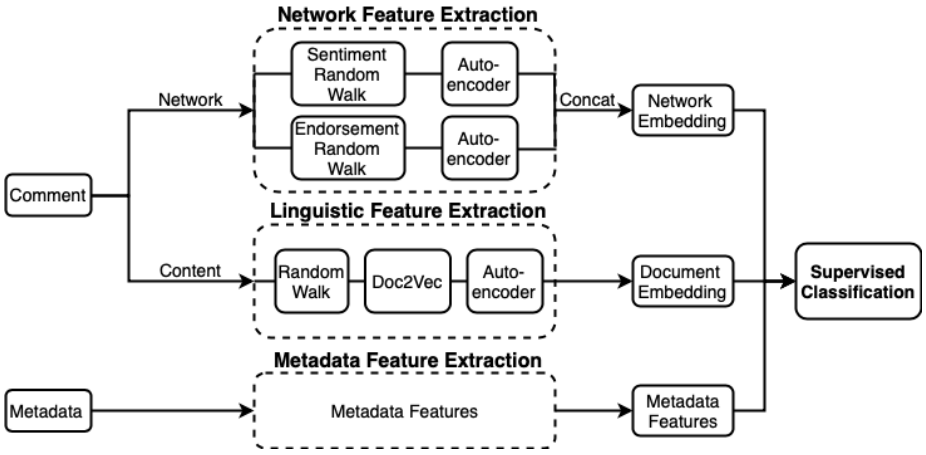


Figure 4: An Overview of OVCP

4.1. Network Feature Extraction

The network feature extraction component in OVCP is designed to capture characteristics from the audience's comments of an online video. We observe that clickbait and non-clickbait videos are different in terms of topological features (e.g., structure of comment threads) and semantic features (e.g., sentiments, endorsements) of audience's comments. To effectively capture both topological and semantic features, the network feature extraction component constructs a comment network, records the network attributes by a Random Walk scheme, and learns the vector representation via an autoencoding approach.

4.1.1. Comment Network Construction for Individual Video

We first construct a network of user comments that represents the topological structure and semantic attributes/features of the user comments. A *comment network* $\mathbf{G}$ for each video is defined as a directed graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, where $\mathbf{V}$ is the set of nodes and $\mathbf{E}$ is the set of directed edges between nodes. We define a source node $s \in \mathbf{V}$ to be the description of the video, and other nodes (i.e., $v \in \mathbf{V}, v \neq s$) to be all comments of the video. Each edge $(v, v') \in \mathbf{E}$ denotes a reply from comment v to comment v'.

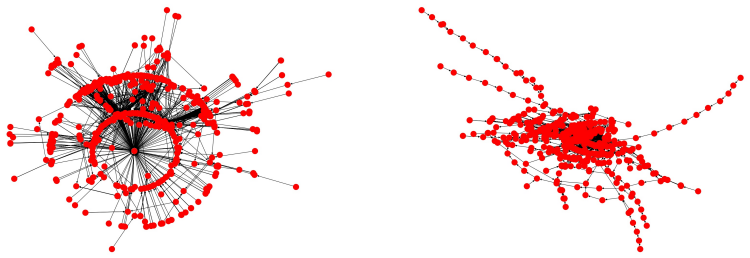
Different from other trending social media platforms (e.g., Reddit) that have a multi-level comment structure, YouTube only allows a two-level commentary mechanism, namely a top-level comment and its replies (i.e., second-level comments to the top-level comment). In other words, each comment thread is recorded as the top-level comment followed by all of its replies in a flat structure. To retrieve the comment network structure, in each comment thread, we connect the top-level comment node to the source node s and direct all second-level comment nodes to their top-level comment node. When a specific user (e.g., a username following a "+" or "@") is mentioned in a second-level comment, the comment node is redirected to the latest comment node of the mentioned user in the same thread. Note that a comment can receive any number of replies but can only reply to one comment. In other words, each node v can have any number of incoming edges but only one outgoing edge.

Figure 5 demonstrates the networks constructed from the comments of a clickbait video and a non-clickbait video. We observe that the comment network of a clickbait video has a few "hub" comment nodes that receive more replies than normal comment nodes. For example, a comment like "Who else come for the thumbnail?" receives a large number of replies from other viewers who suffer from the same clickbait video. We also observe that non-clickbait videos contain longer comment threads since users are more likely to discuss the topic presented in the video through interactive replies if the video is not a clickbait.

4.1.2. Semantic Feature Extraction with Random Walks

After the comment network $\mathbf{G}$ of a video is constructed, we further extract the semantic features (i.e., sentiment and endorsement) from the comment network. We observe that such semantic features of a comment can sometimes reveal the user's attitude and behavior towards a possible clickbait video. Therefore, it makes sense to incorporate them into our OVCP scheme. In particular, we adopt a Random Walk (RW) algorithm [31] in the constructed network to capture these two semantic features.

A random walk $RW(M, K)$ scheme often traverses a graph M times while



(a) Clickbait (b) Non-clickbait

Figure 5: Examples of the Comment Network Structure for Clickbait and Non-clickbait Videos
Note: for better visualization, we only keep threads with more than one comments in the plot.

‘

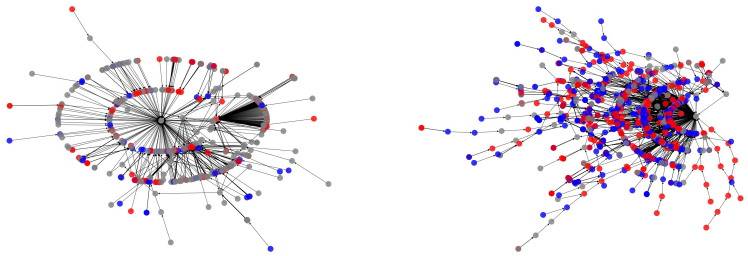
limiting the number of steps in each traverse path to be at most K [32]. In the OVCP scheme, we trace not only the reply direction of a comment node but also the attributes of the comment node. Intuitively, in each traversal of the comment network $\mathbf{G}$, we randomly select a comment node and record its path until it reaches the source node or the length of the path reaches K. We define the two attitude paths below to track the semantic features (i.e., sentiment and endorsement) of each comment node in the RW process. In particular, we define two semantic features as attributes of each comment node v: i) *sentiment attribute* $\alpha_s(v)$ is defined as the *polarity* score extracted by a sentiment analysis tool TextBlob[4], and ii) *endorsement attribute* $\alpha_e(v)$ is defined as the number of likes a comment received from other users.

Definition 8. ***Sentiment Path*** *(RW_s): is the random walk process that traverses the graph* $\mathbf{G}$ *from a randomly selected comment node* v_0 *and records the sentiment attribute* α_s *of each comment node on the path. In each step, the random walk process follows the direction to the next comment node that the current comment node replies to. Formally, for the* m^{th} *walk in the process,* $RW_s(m) = \{RW_s(m,0), RW_s(m,1), \cdots, RW_s(m,K-1)\}$ *where* $RW_s(m,k) =$

[4]https://textblob.readthedocs.io/

$\alpha_s(v_k)$ *represents the sentiment attribute of the* k^{th} *node* v_k *on the path.*

The Sentiment Path RW_s captures the sentiment feature of the comment network. Figure 6 demonstrates an example of the sentiment attribute in the comment network. We observe that the comment sentiments from non-clickbait videos are more diverse than clickbait videos. This is because the comments from viewers of non-clickbait videos primarily focus on the content of the video and usually reflect the diversified attitudes (e.g., like vs. dislike) towards the video. Another reason is that viewers often leave a clickbait video immediately without making any comment. Such rational user behavior reduces not only the number of comments a clickbait video receives but also its likelihood of getting diversified comments.



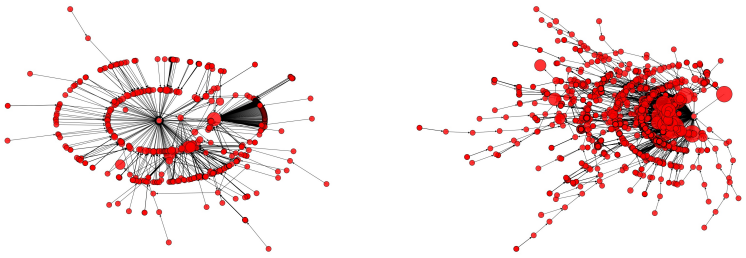
(a) Clickbait (b) Non-clickbait

Figure 6: Sentiment Feature Path. The color of each node represents the sentiment attribute of the corresponding comment, i.e., *red* - positive, *blue* - negative, *grey* - neutral.

Definition 9. ***Endorsement Path*** *(*RW_e*):* *is the random walk process that traverses the graph* $\mathbf{G}$ *from a randomly selected comment node* v_0 *and records the endorsement attribute* α_e *of each comment node on the path. In each step, the random walk process follows the direction to the next comment node that the current comment node replies to. Formally, for the* m^{th} *walk in the process,* $RW_e(m) = \{RW_e(m,0), RW_e(m,1), \cdots, RW_e(m,K-1)\}$ *where* $RW_e(m,k) = \alpha_e(v_k)$ *represents the endorsement attribute of the* k^{th} *node* v_k *on the path.*

The Endorsement Path RW_e captures the endorsement a comment receives

in the comment network. It reflects the agreement from other users on a particular comment. Figure 7 demonstrates an example of the endorsement attribute in the comment network. We observe that only a few comments in clickbait videos receive a large amount of endorsements from other users. We find these comments are often the ones that point out the video is a clickbait, which are appreciated/endorsed by other users. The degree of endorsements in non-clickbait videos is much more diversified since users are more engaged in the discussion of the video content instead of endorsing or disputing it.



(a) Clickbait (b) Non-clickbait

Figure 7: Endorsement Feature Path. The size of each node represents the endorsement attribute of the corresponding comment, i.e., the number of likes a comment received.

We perform the random walk process M times with a maximum length of K for each attribute path, and denote each path with $RW_s(m)$ and $RW_e(m)$ for the m^{th} walk of the *sentiment* and *endorsement* attribute respectively. If attribute paths $RW_s(m)$ and $RW_e(m)$ arrive at the source node s before the length of the path reaches K, i.e., $v_k = s$ and $k < K$, $RW_s(m)$ and $RW_e(m)$ are padded with neutral sentiment and zero endorsement respectively. We store the K attributes recorded in M random walk paths into feature matrices (H_s and H_e) of size $M \times K$, where $H_s(m,k) = RW_s(m,k)$ and $H_e(m,k) = RW_e(m,k)$.

4.1.3. Network Representation Learning

Given the high-dimensional network features of the comment network $\mathbf{G}$, namely H_s and H_e extracted by the Random Walk algorithm, we further encode

the network features into a low-dimensional vector space and learn the latent vector representation for them. An autoencoder is a generative deep learning technique that captures hidden patterns of high dimensional data with artificial neural networks [35]. It consists of an *encoder* $E(x) : \mathbb{R}^n \rightarrow \mathbb{R}^m$ that reduces the feature space dimension from n to m, where $m \ll n$. The reduction is done through one or more layers of neural networks, and a *decoder* $D(z) : \mathbb{R}^m \rightarrow \mathbb{R}^n$ that recovers the encoded vector to its original dimensionality.

It has been shown that deep autoencoder is an effective way of nonlinear dimensionality reduction [36]. In our scheme, we compress the network feature vectors H_s and H_e by independently training two stacked autoencoders with respect to the sentiment and endorsement attributes. Formally, our stacked autoencoder structure consists of six neural network layers, and the i^{th} layer is defined as:

$$\mathbf{Z}^i = \phi(\mathbf{W}^i \cdot \mathbf{X}^i + \mathbf{b}^i) \tag{2}$$

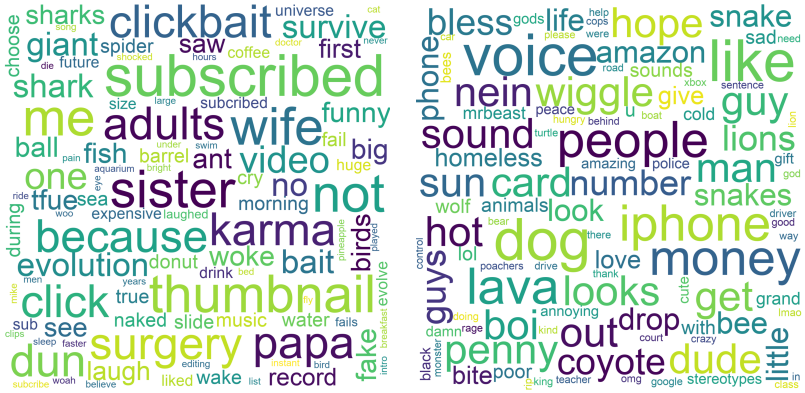
where $\phi(\cdot)$ is a nonlinear activation function, $\mathbf{W}^i$ is the weighting factor and $\mathbf{b}^i$ is the bias term. $\mathbf{X}^i$ and $\mathbf{Z}^i$ are the input and output of each layer. We set the input of the first layer to be the network feature vectors H_s and H_e for the sentiment and endorsement autoencoder, respectively. We use Mean Square Error (MSE) as the loss function for the stacked autoencoder and the rectified linear unit (ReLU) as the activation function.

Finally, the latent features (i.e., $\mathbf{Z}_s$ and $\mathbf{Z}_e$) learned from the stacked autoencoders are aggregated by a concatenation function, $\mathbf{Z} =< \mathbf{Z}_s, \mathbf{Z}_e >$, which represents the network features extracted from the comment network.

4.2. Linguistic Feature Extraction

In the second component, we extract linguistic features from the comment section by learning the document embedding of each comment. Figure 8 shows the word clouds of most frequent words from the comments of clickbait and non-clickbait videos, respectively. Intuitively, we observe that clickbait related words (e.g., “clickbait”, “bait”, “fake”, “thumbnail”) appear more frequently

in the comments of clickbait videos while content-related words (e.g., "lava", "card", "voice") appear more often in the discussion of non-clickbait videos.



(a) Clickbait (b) Non-clickbait

Figure 8: Word Clouds

Therefore, we employ a widely adopted document embedding technique, namely Doc2vec [37], to extract linguistic features from comments (i.e., comment embedding). Doc2vec, derived from the famous Word2vec framework, is designed to learn fixed-length continuous distributed vector representations for word sequences of variable-length.

A naïve approach is to simply embed the whole comment section of each video. However, we found such an approach performs poorly due to the extremely long or short comment length of some videos. In particular, we found a majority of comment threads only have one comment, but some threads have hundreds of comments. To capture the information of both short and long comment threads, we design a hierarchical embedding process as follows: 1) we first embed each single comment using the Doc2vec model; 2) each long comment thread is traversed using the Random Walk process and the recorded comment path is further embedded by calculating the mean of comment embeddings in the path (referred to as *path embedding*); 3) we train another independent 8-layer stacked autoencoder to reduce the dimension of the path embeddings.

Using this hierarchical embedding method, we can control the dimensionality of the linguistic feature space while preserving the information captured by the Doc2vec framework.

4.3. Metadata Feature Extraction

We further extract some complementary metadata features that are relevant in identifying clickbait videos but cannot be captured by network and linguistic clues discussed above. These metadata features are mainly selected based on empirical observations. For example, we observe that some clickbait videos contain URLs to malicious websites in the video descriptors. We extract a total of 13 metadata features from the collected real-world dataset. These metadata features are shown in Table 1. These features are computed to describe statistical characteristics of both the video content (e.g., video length, the number of views) and the comments of the video (e.g., the word count of a comment, the number of likes of a comment). The correlation plot of all the metadata features is shown in Figure 9. We observe that the extracted metadata features are relatively independent.

4.4. Supervised Classification

Finally, we combine all the network, linguistic and metadata features described above and perform the binary classification to detect online clickbait videos. In the supervised classification module, we adopt a few state-of-the-art classification algorithms and select the best-performed one as the classifier for our OVCP scheme. The supervised classifiers include probabilistic classifiers, support vector machine, boosting and ensemble methods, and neural networks. The detailed performance evaluation for the collection of classifiers and the OVCP scheme is presented in the following section.

5. Evaluation

In this section, we first describe the dataset we collected from YouTube. We then evaluate the performance of the OVCP scheme in comparison with

Table 1: Metadata Features

Feature	Description
`Comment Count`	Total # of comments for each video
`Dislike Count`	Total # of dislikes for each video
`Like Count`	Total # of likes for each video
`View Count`	Total # of views for each video
`Like to Dislike`	The ratio of like count to dislike count
`Daily View Count`	Avg. # of daily views for each video
`Like to View`	The ratio of like count to view count
`Duration`	Length of video in minutes
`Description URL Count`	Avg. # of URLs in the description
`Like Count per Comment`	Avg. # of likes for each comment
`Word Count per Comment`	Avg. # of words in each comment
`Clickbait Count`	Avg. # of words related to clickbait in each comment
`Weighted Clickbait Count`	Avg. # of words related to clickbait in each comment weighted by comment's like count

state-of-the-art baselines on the collected dataset. The results show that OVCP significantly outperforms both the compared baseline methods and human annotators in terms of accurately detecting online clickbait videos.

5.1. Dataset

YouTube is visited by over 1.9 billion logged-in users each month and over a billion hours of video are watched daily[5]. In view of the diversity and popularity of , we take YouTube as our data source to collect video information.

For the training dataset, considering the imbalanced nature of clickbait videos on YouTube, we leverage YouTube's recommendation system to effi-

[5]https://www.youtube.com/intl/en-US/yt/about/press/

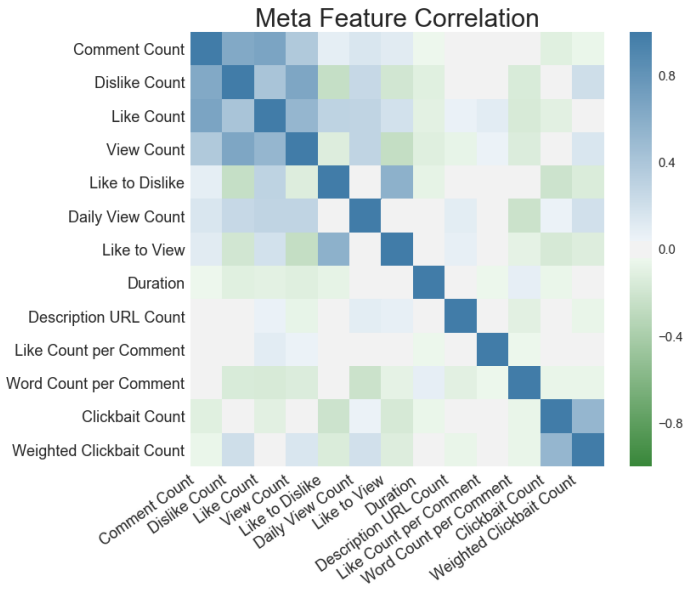


Figure 9: Metadata Feature Correlation

ciently collect clickbait videos being recommended to real users and to minimize human bias in the data collection process. First, we randomly assigned a set of trending videos recommended by YouTube to three independent human annotators who collectively identified a set of "seed" clickbait videos by manually watching them (we set the size of the seed set to be 40 in our experiment due to the high labeling cost). We then created a dummy account on YouTube and let YouTube recommend relevant videos based on the watching history of the "seed" videos. Using such a method, we collected a training dataset of 500 videos which is a reasonable sample size to study the online video detection problem on YouTube [38]. Note that, in the training set, we intentionally collect more clickbait videos so that the training set is relatively balanced. This allows us to train the model more effectively [39]. To ensure fairness, we use the same training data for all baselines. In order to evaluate the performance in a real-world scenario, we randomly collected 125 videos from the "trending video" section YouTube's homepage. This strategy allows the collected videos to have a simi-

lar clickbait video percentage as that on YouTube due to its nature of random sampling. In particular, following the common practice in supervised machine learning [40] and the Pareto principle (i.e., 80/20 rule)[6] of training/testing data split, we selected 500 videos as the training set and 125 videos as the testing set, and performed 5-fold cross-validation in our evaluation.

For each of these videos, we recorded its unique `videoID` and collected the ground-truth label (i.e., whether the video is a clickbait or not) based on the majority voting of human annotations. We also collected the title, description, thumbnail, comments and commentThreads [7], and the content of each video through the YouTube Data API[8]. An overview of the collected dataset is summarized in Table 2. We observe that the comments in the collected videos follow a "long tail" distribution. As shown in Figure 10, most comments in a thread have no replies (i.e., number of comments per thread is 1) and only a small portion of the threads have multiple comments.

Table 2: Data Trace Statistics

Dataset	**Train**		**Test**	
Video Label	**Clickbait**	**Non-clickbait**	**Clickbait**	**Non-clickbait**
Videos	128	372	13	112
Comment Threads	388,337	2,050,438	50,139	243,213
Comments	506,407	2,771,672	59,523	287,045
Comments per Thread	1.30	1.35	1.19	1.18
Distinct Users	419,392	2,072,927	51,998	246,481
Unique Words in Comments	156,201	551,552	37,330	119,270

5.2. Baselines

We integrate the following state-of-the-art supervised classifiers [41] with our scheme as discussed in Section 4.4: *Logistic Regression (LR)*, *Linear Sup-*

[6] https://en.wikipedia.org/wiki/Pareto_principle

[7] An attribute that records the structure of each comment thread in the comment section

[8] https://developers.google.com/youtube/v3/

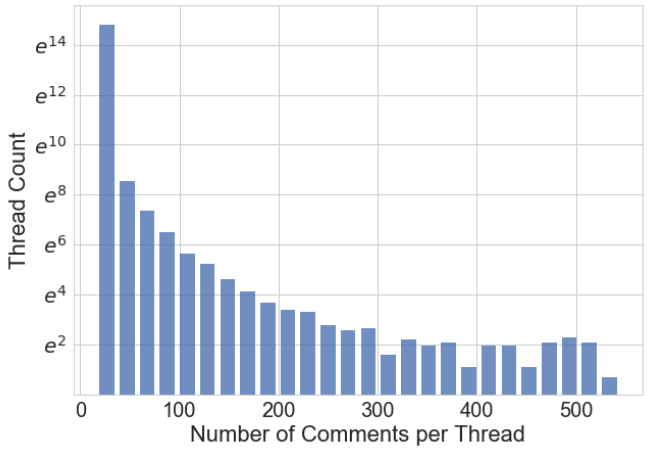


Figure 10: Distribution of Comments Count per Thread

port Vector Machine (SVM), *AdaBoost*, *Random Forest (RF)*, and *Multi-layer Perceptron (MLP)*. We select the best-performed one to be the one used in our OVCP scheme. We compare the performance of our scheme with the following representative baselines in clickbait detection [9].

- **Image-based Clickbait Detection (VGG-16):** A convolutional neural network approach (pre-trained on ImageNet) that detects the clickbait video using only the image content of the thumbnail [42].
- **Stop Clickbait (ASONAM16):** A linguistic-based clickbait classification approach that detects the clickbait video based on the title content of the video [43].
- **Clickbait Detection using Deep Learning (NGCT16):** A deep neural network approach that identifies clickbait videos using a compiled clickbait corpus from social media posts[15].

[9]The online video clickbait detection problem is a largely unsolved research problem. Therefore, we only have a limited number of baselines from the literature to compare against.

- **Clickbait Video Detection (SPW18):** A deep learning based approach that detects clickbait video on YouTube using features from the headline, thumbnail, comments and video statistics [20].

5.3. Detection Accuracy

In the first set of experiments, we evaluate the performance of our scheme when it is coupled with a set of classifiers and identify the best-performed one as the OVCP scheme. The evaluation metrics involved in measuring the effectiveness of clickbait video detection include *Accuracy*, *Precision*, *Recall*, and *F1-Score*, which are commonly used in binary classification tasks. In addition, we include two imbalanced metrics, *Cohen's Kappa (Kappa)* and *Matthews Correlation Coefficient (MCC)*, because our dataset is not perfectly balanced (see Table 2). For all classification methods and baselines, we perform 5-fold cross-validation in the parameter tuning process.

In the OVCP scheme, we set the parameters in each component primarily based on our empirical observation as follows: i) *random walk*: we set $M = 100$ and $K = 5$; ii) *stacked autoencoder (network)*: we set the size of encoding layers to be 256, 64 and 16, and the size of decoding layers to be 64, 256 and 500; iii) *Doc2vec*: the length of the comment embedding is set to be 256; and iv) *stacked autoencoder (linguistic)*: we set the size of encoding layers to be 128, 64, 32 and 16, and the size of decoding layers to be 32, 64, 128 and 256.

The performance of all compared schemes is summarized in Table 3. We observe that AdaBoost performs the best among all classifiers and thus is selected to be the default classifier in the OVCP scheme. We also observe that our scheme outperforms all baseline approaches all evaluation metrics. In particular, comparing to the *VGG-16*, *ASONAM16*, *NGCT16* and *SPW18* baselines, our scheme achieves a performance gain of 0.3262, 0.2219, 0.1943 and 0.1229 on the F1-score respectively. Although the *SPW18* approach considers various content-based features (e.g., headline, thumbnail, video statistics) in its model, it is found to be less effective than our scheme in detecting clickbait videos. The reason is that the *SPW18* method has a high-dimensional input vector size of

855 that requires a vast amount of training samples to achieve a reasonable performance. The *SPW18* approach also relies on visual features (i.e., thumbnail) which can be misleading in detecting the clickbait videos.

Table 3: Clickbait Classification Performance for All Methods

Algorithms	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Kappa**	**MCC**
AdaBoost (**OVCP**)	**0.8960**	**0.5000**	**0.4615**	**0.4800**	**0.4223**	**0.4227**
LR	0.8800	0.4286	0.4615	0.4444	0.3773	0.3776
SVM	0.8880	0.4545	0.3846	0.4167	0.3552	0.3567
RF	0.8640	0.3889	0.5385	0.4516	0.3763	0.3828
MLP	0.8480	0.2857	0.3077	0.2963	0.2112	0.2114
VGG-16	0.8240	0.1538	0.1538	0.1538	0.0556	0.0556
ASONAM16	0.8160	0.2222	0.3077	0.2581	0.1561	0.1588
NGCT16	0.8400	0.2667	0.3077	0.2857	0.1961	0.1968
SPW18	0.8560	0.3333	0.3846	0.3571	0.2765	0.2774

Additionally, we plot the Receiver Operating Characteristic (ROC) curve of all schemes to evaluate the robustness of their performance with respect to the classification threshold. As shown in Figure 11, our scheme consistently outperforms all baselines and achieves an increase of 0.09 on the AUC score compared to the baseline of best performance (i.e., NGCT16).

5.4. Feature Analysis

In the second experiment, we study the importance of features in each category (i.e., network, linguistic, metadata) and their combinations. The results are shown in Table 4. We observe that the combination of all features achieve the best results on all evaluation metrics which confirms their necessity in our model. We also observe that linguistic features achieve the best performance among all single feature categories, and the combination of linguistic and metadata features achieve the best performance among the combination of any two feature categories.

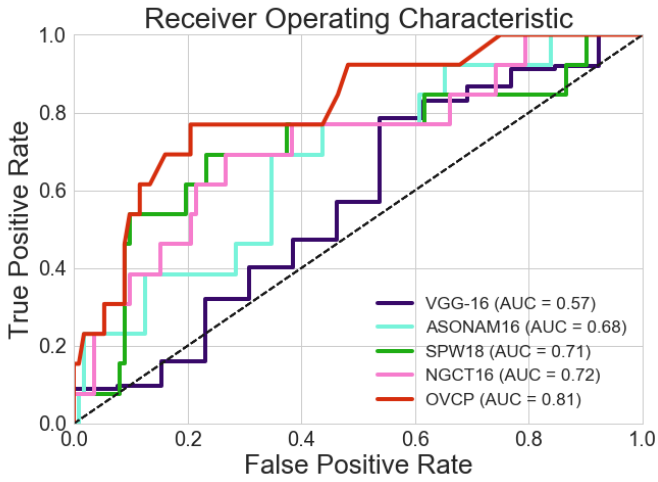


Figure 11: ROC Curve of All Schemes

Table 4: Classification Performance for All Feature Sets

Feature Set	Accuracy	Precision	Recall	F1-Score
All Features	**0.8960**	**0.5000**	**0.4615**	**0.4800**
Metadata	0.7680	0.1364	0.2308	0.1714
Network	0.8240	0.1538	0.1538	0.1538
Linguistic	0.8480	0.2000	0.1538	0.1739
Network & Metadata	0.8720	0.2857	0.1538	0.2000
Linguistic & Metadata	0.8400	0.2941	0.3846	0.3333
Network &Linguistic	0.8560	0.2727	0.2308	0.2500

5.5. Influence of Training Size

In the third experiment, we further study the robustness of our scheme against the size of the training set. In particular, we evaluate the performance of the OVCP scheme and all other baselines by increasing the size of the training set from 30% to 100% of the entire set. The F1-score results are reported in Figure 12. We observe that the performance of our scheme generally improves as the size of the training set increases and continues to outperform other baselines.

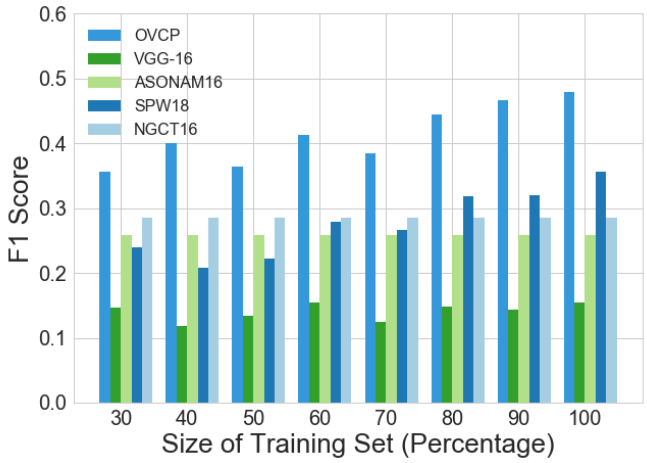


Figure 12: Size of Training Set vs. Performance (F1 Score)

5.6. Detection Time

In the fourth experiment, we measure the detection performance of the OVCP scheme against the time frame after the video is posted. Specifically, we limit the comment data in our scheme to be within a time frame from the corresponding video's publishing time, and increase the time frame from 10 minutes to 24 hours. The classification results are summarized in Table 5. We observe that the OVCP scheme using the first 10 minutes after the video's publication has already outperformed all content-based baselines with a non-trivial performance gain. Moreover, the OVCP scheme is observed to reach its optimal performance within 24 hours.

5.7. Detection Efficiency

In the fifth experiment, we empirically measure the efficiency (i.e., detection time per video) of all schemes. The results are shown in Figure 13. We observe that the OVCP scheme achieves the best detection performance (F1 score) with a reasonable detection time compared to other baselines. We also observe that the SPW18 approach is the slowest among all compared schemes. This is because SPW18 requires features to be extracted from multiple data modalities,

Table 5: Classification Performance v.s. Detecting Time Frame

Time Frame	Accuracy	Precision	Recall	F1-Score
OVCP (Within 10 Minutes)	0.8640	0.3571	0.3846	0.3704
OVCP (Within 30 Minutes)	0.8560	0.3684	0.5385	0.4375
OVCP (Within 1 Hour)	0.8640	0.3889	0.5385	0.4516
OVCP (Within 6 Hours)	0.8720	0.4118	0.5385	0.4667
OVCP (Within 12 Hours)	0.8880	0.4615	0.4615	0.4615
OVCP (Within 24 Hours)	0.8960	0.5000	0.4615	0.4800
OVCP (All Comments)	**0.8960**	**0.5000**	**0.4615**	**0.4800**
VGG-16	0.8240	0.1538	0.1538	0.1538
ASONAM16	0.8160	0.2222	0.3077	0.2581
NGCT16	0.8400	0.2667	0.3077	0.2857
SPW18	0.8560	0.3333	0.3846	0.3571

including image (i.e., thumbnail), texts (i.e., video title and description) and metadata. Such a large feature space significantly reduces the detection efficiency. The NGCT16 has the least detection time since it simply relies on pre-trained word embeddings to predict clickbait videos via a deep learning framework, which in turn limits its detection accuracy.

Finally, we study the computational complexity/scalability of our OVCP scheme. In particular, we perform an empirical study of different modules of the OVCP scheme for a detailed analysis. The results are shown in Figure 14. We observe that the linguistic and network feature extraction modules are the two dominant ones that consume most of the execution time of the OVCP scheme. The reason is that these dominant modules include neural network solutions (i.e., doc2vec and autoencoder) for latent feature extraction, which often involve a non-trivial amount of matrix operations in both learning and inference phases of the process [44]. However, we observe the computation time of those two dominant modules increase *linearly* as the size of the input data increases, which demonstrates the good scalability of our scheme with larger

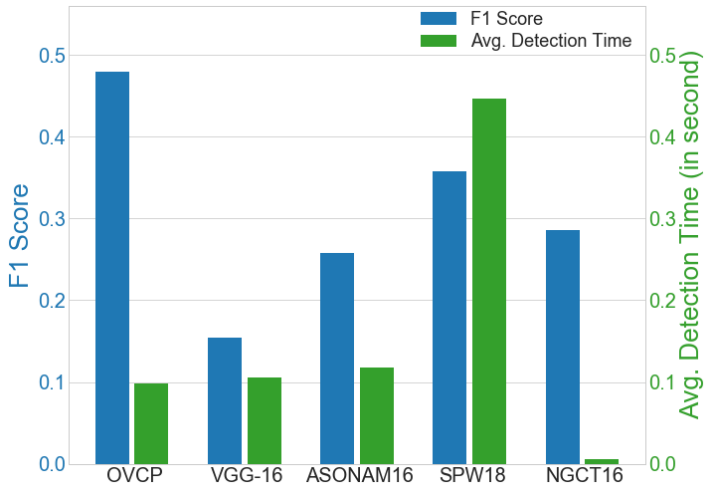


Figure 13: Performance (F1 Score) v.s. Average Detection Time Cost (per Video)

datasets.

We also note that, in our current implementation, the videos are detected in a sequential manner, i.e., one video is detected after another. This leads to the linear growth of the execution time as the number of videos becomes large. We observe that our OVCP scheme can be easily extended to further improve its speed by leveraging the parallel GPU programming frameworks such as CUDA[10] to execute many video streams in parallel on thousands of GPU cores. Moreover, elastic distributed computing systems such as AWS Kubernetes[11] can also allow new OVCP instances to spin up on virtual machines when the system is overloaded (e.g., when many new videos are uploaded). Considering this line of effort is beyond the scope of this paper, we plan to implement it in our future work.

[10] https://developer.nvidia.com/cuda-zone
[11] https://kubernetes.io/

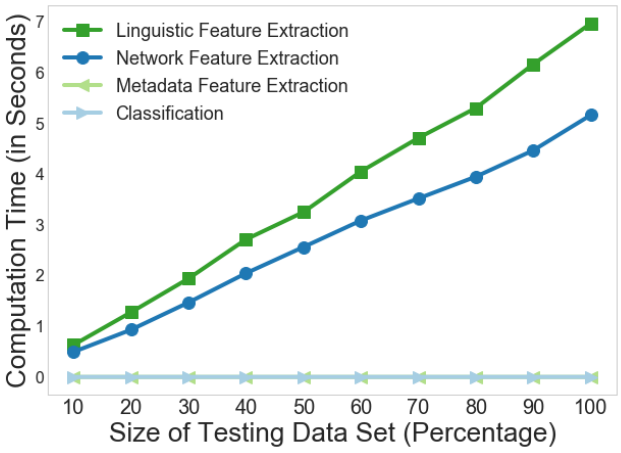


Figure 14: Computation Time for All Modules of OVCP

5.8. Human Performance Comparison

In the last experiment, we compare the performance of OVCP with human annotators[12]. In particular, we invited three independent annotators (different from the ones who were invited to generate the ground truth labels) to identify clickbait videos from the same testing dataset used for the OVCP scheme. First, we asked these annotators to annotate videos by only showing them the title, thumbnail and view count of each video which exactly the same information a YouTube user receives *before* clicking a video. Next, we asked these annotators to annotate the videos by also giving them access to the comment section of the videos (i.e., metadata and comment features). The performance of each individual annotator and the aggregated results using the majority voting (i.e., "overall" and "overall+comment") are shown in Table 6. We also report false positive rate (FPR), and false negative rate (FNR) in addition to accuracy and F1 score metrics to study what kinds of mistakes the human annators made.

[12]The difference between the human annotators in this experiment and the human annotators we used in collecting the ground truth labels is that we did not allow the human annotators in this experiment to watch the actual videos before they generate the annotations.

We observe that the OVCP scheme consistently outperforms human annotators in detecting clickbait videos, demonstrating the necessity of developing such a tool. We also observe the human annotators often perform worse when they are not allowed to access the comments of the video, especially in terms of false negative rate (missing many clickbait videos).

Table 6: Comparison between OVCP and Human Performance

	Accuracy	**F1-Score**	**FPR**	**FNR**
OVCP	**0.8960**	**0.4800**	**0.0536**	**0.5385**
Annotator 1 (without comments)	0.7040	0.0976	0.2321	0.8462
Annotator 1 (with comments)	0.7440	0.1579	0.1964	0.7692
Annotator 2 (without comments)	0.6880	0.1333	0.2589	0.7692
Annotator 2 (with comments)	0.7360	0.1951	0.2143	0.6923
Annotator 3 (without comments)	0.7520	0.1143	0.1786	0.8462
Annotator 3 (with comments)	0.7920	0.1875	0.1429	0.7692
Overall (without comments)	0.7280	0.1053	0.2054	0.8462
Overall (with comments)	0.7680	0.1714	0.1696	0.7692

6. Conclusion and Future Work

In this paper, we develop a content-free scheme (OVCP) to detect clickbait videos on online video sharing platforms. Our scheme leverages the comment and interaction between users who watched the video, and learns latent features from their unstructured and complex comments. We evaluate our scheme using the real-world data collected from YouTube. The results demonstrate that our scheme is more accurate than both state-of-the-art clickbait detection tools and human annotators in identifying online clickbait videos.

Finally, we also outline a few future research directions that can be built on the work from this paper. First, clickbait videos on YouTube are often

customized by experienced content creators who know how to game with the YouTube recommendation system. To make the OVCP scheme more robust and generalizable across different video platforms, we plan to extend OVCP's compatibility with other video sharing platforms using different commentary structures. Examples of such platforms include both video-centric platforms (e.g., Vimeo) and social-based platforms (e.g., Twitter). Second, there exist many marketing vendors from which content creators can buy comments and high-retention views. In future work, we will further study how to identify and filter such fake or machine-generated comments. A possible solution is to leverage the social media bot detection techniques [45] and train a classifier that can discriminate comments generated by humans and bots. Meanwhile, we can also adopt the truth discovery [21] and fact-checking [46] approaches to verify the truthfulness of the user comments. The authors believe the above extensions will further improve the effectiveness and robustness of the OVCP scheme in detecting online clickbait videos.

Acknowledgement

This research is supported in part by the National Science Foundation under Grant No. CNS-1831669, CBET-1637251, Army Research Office under Grant W911NF-17-1-0409. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

[1] GroupM, Groupm introduces state of digital report, available at https://www.groupm.com/news/groupm-introduces-state-digital-report, accessed 2019-01-07 (2018).

[2] R. Molla, Next year, people will spend more time online than they will watching tv. that's a first., available at https://www.recode.net/2018/6/8/17441288/internet-time-spent-tv-zenith-data-media, accessed 2019-02-14 (2018).

[3] M. Bärtl, Youtube channels, uploads and views: A statistical analysis of the past 10 years, Convergence 24 (1) (2018) 16–32 (2018).

[4] M. M. U. Rony, N. Hassan, M. Yousuf, Diving deep into clickbaits: Who use them to what extents in which topics with what effects?, in: Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017, ACM, 2017, pp. 232–239 (2017).

[5] D. Wang, T. Abdelzaher, L. Kaplan, Social sensing: building reliable systems on unreliable data, Morgan Kaufmann, 2015 (2015).

[6] M. D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in: European conference on computer vision, Springer, 2014, pp. 818–833 (2014).

[7] Y. Zhang, N. Vance, D. Zhang, D. Wang, On opinion characterization in social sensing: A multi-view subspace learning approach, in: 2018 14th International Conference on Distributed Computing in Sensor Systems (DCOSS), IEEE, 2018, pp. 155–162 (2018).

[8] O. Papadopoulou, M. Zampoglou, S. Papadopoulos, Y. Kompatsiaris, Web video verification using contextual cues, in: Proceedings of the 2nd International Workshop on Multimedia Forensics and Security, ACM, 2017, pp. 6–10 (2017).

[9] D. Y. Zhang, L. Song, Q. Li, Y. Zhang, D. Wang, Streamguard: A bayesian network approach to copyright infringement detection problem in large-scale live video sharing systems, in: 2018 IEEE International Conference on Big Data (Big Data), IEEE, 2018, pp. 901–910 (2018).

[10] M. Potthast, S. Köpsel, B. Stein, M. Hagen, Clickbait detection, in: European Conference on Information Retrieval, Springer, 2016, pp. 810–817 (2016).

[11] M. Huh, A. Liu, A. Owens, A. A. Efros, Fighting fake news: Image splice detection via learned self-consistency, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 101–117 (2018).

[12] Y. Li, M.-C. Chang, H. Farid, S. Lyu, In ictu oculi: Exposing ai generated fake face videos by detecting eye blinking, arXiv preprint arXiv:1806.02877 (2018).

[13] K. Fagan, Youtube's clickbait problem might not be fixable, available at https://yr.media/tech/youtubes-clickbait-problem-is-out-of-hand-and-there-may-be-no-fixing-it/, accessed 2019-02-25 (2018).

[14] To clickbait or not to clickbait: What you need to know about headlines and clickbaits, available at https://marketinginsidergroup.com/content-marketing/what-you-need-to-know-headlines-clickbaits/, accessed 2019-02-19 (2016).

[15] A. Agrawal, Clickbait detection using deep learning, in: 2016 2nd International Conference on Next Generation Computing Technologies (NGCT), IEEE, 2016, pp. 268–272 (2016).

[16] M. Potthast, T. Gollub, K. Komlossy, S. Schuster, M. Wiegmann, E. P. G. Fernandez, M. Hagen, B. Stein, Crowdsourcing a large corpus of clickbait on twitter, in: Proceedings of the 27th International Conference on Computational Linguistics, 2018, pp. 1498–1507 (2018).

[17] A. Anand, T. Chakraborty, N. Park, We used neural networks to detect clickbaits: You won't believe what happened next!, in: European Conference on Information Retrieval, Springer, 2017, pp. 541–547 (2017).

[18] P. Thomas, Clickbait identification using neural networks, arXiv preprint arXiv:1710.08721 (2017).

[19] J. Qu, A. M. Hißbach, T. Gollub, M. Potthast, Towards crowdsourcing clickbait labels for youtube videos.

[20] S. Zannettou, S. Chatzis, K. Papadamou, M. Sirivianos, The good, the bad and the bait: Detecting and characterizing clickbait on youtube, in: 2018 IEEE Security and Privacy Workshops (SPW), IEEE, 2018, pp. 63–69 (2018).

[21] D. Y. Zhang, J. Badilla, Y. Zhang, D. Wang, Towards reliable missing truth discovery in online social media sensing applications, in: 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), IEEE, 2018, pp. 143–150 (2018).

[22] N. Vo, K. Lee, The rise of guardians: Fact-checking url recommendation to combat fake news, arXiv preprint arXiv:1806.07516 (2018).

[23] X. Yin, J. Han, P. S. Yu, Truth discovery with multiple conflicting information providers on the web, IEEE Transactions on Knowledge and Data Engineering 20 (6) (2008) 796–808 (Jun. 2008). doi:10.1109/TKDE.2007.190745.

[24] D. Wang, M. T. Amin, S. Li, T. Abdelzaher, L. Kaplan, S. Gu, C. Pan, H. Liu, C. C. Aggarwal, R. Ganti, et al., Using humans as sensors: an estimation-theoretic perspective, in: Information Processing in Sensor Networks, IPSN-14 Proceedings of the 13th International Symposium on, IEEE, 2014, pp. 35–46 (2014).

[25] D. Wang, T. Abdelzaher, L. Kaplan, C. C. Aggarwal, Recursive fact-finding: A streaming approach to truth estimation in crowdsourcing applications, in: Distributed Computing Systems (ICDCS), 2013 IEEE 33rd International Conference on, IEEE, 2013, pp. 530–539 (2013).

[26] D. Zhang, D. Wang, N. Vance, Y. Zhang, S. Mike, On scalable and robust truth discovery in big data social media sensing applications, IEEE Transactions on Big Data (2018).

[27] D. Wang, L. Kaplan, H. Le, T. Abdelzaher, On truth discovery in social sensing: A maximum likelihood estimation approach, in: Proc. ACM/IEEE 11th Int Information Processing in Sensor Networks (IPSN) Conf, 2012, pp. 233–244 (Apr. 2012). doi:10.1109/IPSN.2012.6920960.

[28] D. Y. Zhang, L. Shang, B. Geng, S. Lai, K. Li, H. Zhu, M. T. Amin, D. Wang, Fauxbuster: A content-free fauxtography detector using social media comments, in: 2018 IEEE International Conference on Big Data (Big Data), IEEE, 2018, pp. 891–900 (2018).

[29] T. Huynh-Kha, T. Le-Tien, S. Ha-Viet-Uyen, K. Huynh-Van, M. Luong, A robust algorithm of forgery detection in copy-move and spliced images, IJACSA) International Journal of Advanced Computer Science and Applications 7 (3) (2016).

[30] H. Wang, F. Zhang, M. Hou, X. Xie, M. Guo, Q. Liu, Shine: Signed heterogeneous information network embedding for sentiment link prediction, in: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, ACM, 2018, pp. 592–600 (2018).

[31] A. Grover, J. Leskovec, node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2016, pp. 855–864 (2016).

[32] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: Online learning of social representations, in: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2014, pp. 701–710 (2014).

[33] X. Huang, J. Li, X. Hu, Label informed attributed network embedding, in:

Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, ACM, 2017, pp. 731–739 (2017).

[34] Y. Zhang, Y. Lu, D. Zhang, L. Shang, D. Wang, Risksens: A multi-view learning approach to identifying risky traffic locations in intelligent transportation systems using social and remote sensing, in: 2018 IEEE International Conference on Big Data (Big Data), IEEE, 2018, pp. 1544–1553 (2018).

[35] P. Vincent, H. Larochelle, Y. Bengio, P.-A. Manzagol, Extracting and composing robust features with denoising autoencoders, in: Proceedings of the 25th international conference on Machine learning, ACM, 2008, pp. 1096–1103 (2008).

[36] G. E. Hinton, R. R. Salakhutdinov, Reducing the dimensionality of data with neural networks, science 313 (5786) (2006) 504–507 (2006).

[37] Q. Le, T. Mikolov, Distributed representations of sentences and documents, in: Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32, ICML'14, JMLR.org, 2014 (2014).

[38] P. Bajaj, M. Kavidayal, P. Srivastava, M. N. Akhtar, P. Kumaraguru, Disinformation in multimedia annotation: Misleading metadata detection on youtube, in: Proceedings of the 2016 ACM workshop on Vision and Language Integration Meets Multimedia Fusion, ACM, 2016, pp. 53–61 (2016).

[39] X.-Y. Liu, J. Wu, Z.-H. Zhou, Exploratory undersampling for class-imbalance learning, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 39 (2) (2008) 539–550 (2008).

[40] P. M. Domingos, A few useful things to know about machine learning., Commun. acm 55 (10) (2012) 78–87 (2012).

[41] E. Alpaydin, Introduction to machine learning, MIT press, 2014 (2014).

[42] X. Zhang, J. Zou, K. He, J. Sun, Accelerating very deep convolutional networks for classification and detection, IEEE transactions on pattern analysis and machine intelligence 38 (10) (2015) 1943–1955 (2015).

[43] A. Chakraborty, B. Paranjape, S. Kakarla, N. Ganguly, Stop clickbait: Detecting and preventing clickbaits in online news media, in: Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, IEEE Press, 2016, pp. 9–16 (2016).

[44] J. A. Hertz, Introduction to the theory of neural computation, CRC Press, 2018 (2018).

[45] E. Ferrara, O. Varol, C. Davis, F. Menczer, A. Flammini, The rise of social bots, Communications of the ACM 59 (7) (2016) 96–104 (2016).

[46] D. Wang, B. K. Szymanski, T. Abdelzaher, H. Ji, L. Kaplan, The age of social sensing, Computer 52 (1) (2019) 36–45 (2019).