Chapter 20
Predicting Video Virality on Twitter

Irene Kilanioti and George A. Papadopoulos

20.1 Introduction

The diffusion of video content is fostered by the ease of producing online content via media services. It mainly happens via ubiquitous Online Social Networks (OSNs), where social cascades can be observed when users increasingly repost links they have received from others. Twitter is one of the most popular OSNs with its core functionality centered around the idea of spreading information by word-of-mouth [16]. It provides mechanisms such as retweet (forwarding other people's tweets), which enable users to propagate information across multiple hops in the network.

If we knew beforehand when a social cascade will happen or to what range it will evolve, we could exploit this knowledge in various ways. For example, in the area of content delivery infrastructure, we could prefetch content by replicating popular items and subsequently spare bandwidth. The knowledge of the evolution of social cascades could lead to reduction schemes for the storage of whole sequences of large social graphs and the reduction of their processing time.

Towards this direction, in this work we present a model for efficiently calculating the number of retweets of a video. The number of retweets is associated with a score depicting the influence of its uploader in the Twitter dataset, the increasing or decreasing trend the score depicts as well as the distance of content interests among users of the YouTube and Twitter community.

I. Kilanioti (✉) · G.A. Papadopoulos (✉)
Department of Computer Science, University of Cyprus, 1 University Avenue,
P.O. Box 20537, 2109 Nicosia, Cyprus
e-mail: ekoila01@cs.ucy.ac.cy

G.A. Papadopoulos
e-mail: george@cs.ucy.ac.cy

F. Pop et al. (eds.), *Resource Management for Big Data Platforms*,
Computer Communications and Networks, DOI 10.1007/978-3-319-44881-7_20

20.1.1 Contributions

Our work focuses on video virality over an OSN. Study of social cascades is active aiming at the prediction of the aggregate popularity of a resource or the individual behaviour of a user. Few works, however, combine detailed information both of the OSN and the media service with a small and easily extracted feature set. Our study proposes a prediction model that performs better than methods like support vector machines (SVM), stochastic gradient descent (SGD) and K-Nearest Neighbours (KNN), among others, and we, furthermore, proceed to incorporate our prediction model into a mechanism for content delivery with substantial improvement for the user experience.

The remainder of this paper is organized as follows. Section 20.2 reviews previous related work. Section 20.3 formally describes the addressed problem. Section 20.4 provides an outline of the methodology, followed by the preparation of the employed datasets. Our main findings are presented in Sect. 20.5, where also a validation is conducted. Section 20.6 investigates the incorporation of the proposed model into a content delivery mechanism. Section 20.7 concludes the work and discusses directions for future work.

20.2 Related Work

The field of predicting social virality is active [2, 5, 6, 13, 15, 17, 22], etc. Many studies focus on the prediction of the amount of aggregate activities (e.g. aggregate daily hashtag use [14]), whereas others focus either on the prediction of user-level behaviour, like retransmission of a specific tweet/URL [7, 15] or on the prediction of growth of the cascade size [5].

Although our work focuses solely on video sharing, we identify the following methods for virality prediction in general. Feature-based methods and time series analysis methods. They are both based on the empirical observation of social cascades. Our approach falls into the first category.

Feature-based methods are based on content, temporal and other features, and the learning algorithms schemes they use are based on simple regression analysis [5, 18], regression trees [2], content-based methods [19], binary classification [8, 9, 12] etc. They do not focus, though, on the underlying network infrastructure, and often encounter difficulty in extracting all the necessary features due to the large volume of accommodated graphs.

Time-series analysis works [20, 21], on the other hand, argue that patterns of a resource's growth of popularity are indicative for its future retransmissions.

Finally, we should mention that one branch of virality research is based on study of the evolution of cascades during a specific time-window [12, 14, 19], whereas there exist works that examine the cascades continuously over their entire duration [5].

20.3 Problem Description

We consider a directed graph $G(t) = (V(t), E(t))$ representing a social network that evolves through time, consisting at time t of V vertices and E edges. Edges between the nodes of the graph denote friendship in case of a social network (e.g. for Twitter B is a follower of A if there is an edge between B and A pointing at A).

Our problem is stated as follows (Table 20.1). We want to predict the number of retransmits of a video link by a user $v \in V$ after $u \in V$ has transmitted the link. User v is a follower of u.

We express this number, intuitively, as a combination of the following features: the $Score(u, t)$ of node u, $dScore(u, t)/dt$ of node u, and content distance between the content interests of the involved users both in the OSN and the media service. The validity of the predictors is analyzed in this paper. The intuition for their selection is based on the notion, that, the higher influence score a node depicts, the more influence it is expected to exert on other nodes of the social graph. Moreover, the $dScore/dt(u, t)$ expresses the popularity rise/fall of the node, and, lastly, the content distance associates the resource with the user context.

Denoting the output, the predicted output and the total number of predicted values by A_{u2v}, $\widehat{A_{u2v}}$ and M, we aim to find the values α, β, γ, so that:

$$A_{u2v} = \alpha \times Score(u, t) + \beta \times \frac{d\,Score(u, t)}{dt} + \gamma \times content\_dist \tag{20.1}$$

and

$$\sqrt{\frac{1}{M}\sum_{i=1}^{M}(\widehat{A_{u2v}} - A_{u2v})^2} \tag{20.2}$$

is minimum.

Table 20.1 Notation overview

$G(t) = (V(t), E(t))$	OSN graph G at time t of V vertices and E edges
A_{u2v}	Number of actions where u influenced v
$\widehat{A_{u2v}}$	Predicted output
M	Total number of predicted values
α, β, γ	Coefficients of feature set variables
U	Vector of YouTube interests of user u
V	Vector of Twitter interests of user v
Features set	
$Score(u, t)$	Score of node u at time t
$dScore = dScore(u, t)/dt$	Derivative of Score of node u at time t
$content\_dist$	Content distance

20.4 Proposed Methodology

20.4.1 Dataset

Interests of users were analyzed in [1] against directory information from http://wefollow.com, a website listing Twitter users for different topics, including Sports, Movies, News & Politics, Finance, Comedy, Science, Non-profits, Film, Sci-Fi/Fantasy, Gaming, People, Travel, Autos, Music, Entertainment, Education, Howto, Pets, and Shows.

The activity of Twitter users was quantified, and a variety of features were extracted, such as the number of their tweets, the fraction of tweets that were retweets, the fraction of tweets containing URLs, etc. Aggregated features of YouTube videos shared by a user in the dataset include the average view count, the median inter-event time between video upload and sharing, etc.

A sharing event in the dataset is defined as a tweet containing a valid YouTube video ID (with a category, Freebase topic and timestamp). We augmented the provided dataset with Tweet content information about the 15 million video sharing events included in the dataset, as well as information about the followers of the 87 K Twitter users.

20.4.2 User Score Calculation

A user score is calculated combining the number n of its followers, reduced by a factor of 1000 to compensate the wide range of followers in the dataset from zero to more than a million, a quantity b catering for users with reciprocal followership, calculated by taking an average of number of a user's followers to the number of users he follows, as well as the effect e of a user's tweet, measured by multiplying average number of retweets with number of user's tweets and normalizing it to correspond to the total number of tweets. The distribution of these combined metrics depicts large variance and we have applied a logarithmic transformation in order to avoid the uneven leverage of extreme values.

$$Score = \log\left(n + \left(\left(\frac{b}{100}\right) \times n\right) + e\right) \tag{20.3}$$

20.4.3 Content Distance

The content distance $content\_dist$ expresses a measure of similarity of user's u YouTube and his follower's v Twitter interests. Content distance is calculated using cosine similarity between vectors of user's u YouTube and user's v Twitter video interests, as follows:

$$content\_dist = 1 - \frac{U \cdot V}{\|U\| \|V\|} \tag{20.4}$$

20.5 Experimental Evaluation

By combining user ids, followership information, user features and tweet context we build a measure of A_{u2v}, expressing the number of times a user's u tweet is retweeted by his followers v. We aim to associate the independent variables of the features set (X dataframe) with the series depicting A_{u2v} (y) (Table 20.2).

20.5.1 Selection of Predictors

The regression summary of Table 20.3 shows that coefficients of all predictors are significant ($P > |t|$ is significantly less than 0.05). Therefore, $Score$, $dScore$ and $content\_dist$ can be considered as good predictors. We note that t here refers to $t - statistic$, denoting the quotient of the coefficient of dependent variable divided by coefficient's standard error. P refers to the $P - value$, a standard statistical method for testing an hypothesis. $P - value < 0.05$ means we can reject the hypothesis that the coefficient of a predictor is zero, in other words the examined coefficient is significant (Table 20.4).

Table 20.2 Regression results (i)

Dep. variable	A_{u2v}	R-squared	0.396
Model	OLS	Adj. R-squared	0.396
Method	Least squares	F-statistic	1.570e+04
Prob (F-statistic)	0.00	Log-likelihood	−8576.9
No. observations	71952	AIC	1.716e+04
Df residuals	71949	BIC	1.719e+04
Df model	3	Covariance type	nonrobust

Table 20.3 Regression results (ii)

	Coef	Std err	t	$P > \|t\|$	95 %	Conf. int.
$Score$	5.79e-05	3.78e-06	15.300	0.000	5.05e-05	6.53e-05
$dScore$	4.36e-05	4.51e-06	9.667	0.000	3.48e-05	5.25e-05
con_dist	0.389	0.002	213.060	0.000	0.386	0.393

Table 20.4 Regression results (iii)

Omnibus	2091.840	Durbin-Watson	1.723
Prob(Omnibus)	0.000	Jarque-Bera (JB)	2323.421
Skew	0.408	Prob(JB)	0.00
Kurtosis	3.333	Cond. No.	746

The selection of the above predictors comes as a result of comparing the $P-values$ of various metrics in the dataset and the combination of those with the lowest $P-value$. The metrics included the number of distinct users retweeted, fraction of the users tweets that were retweeted, average number of friends of friends, average number of followers of friends, number of YouTube videos shared, the time the account was created, the number of views of a video, etc., among many others.

20.5.2 Effect of Outliers

The regression plots for each predictor in Fig. 20.1 show the effect of outliers on the estimated regression coefficient. Regression line is pulled out of its optimal tracjectory due to the existent outliers. The detailed regression plots for individual predictors ($Score$, $dScore$ and $content\_dist$) appear in Figs. 20.2, 20.3, and 20.4

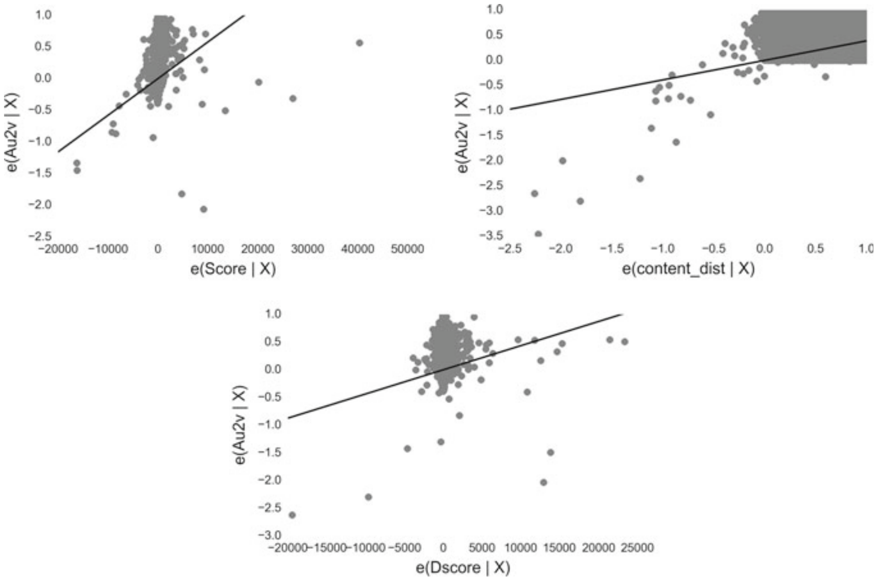

Fig. 20.1 Regression plots for each independent variable

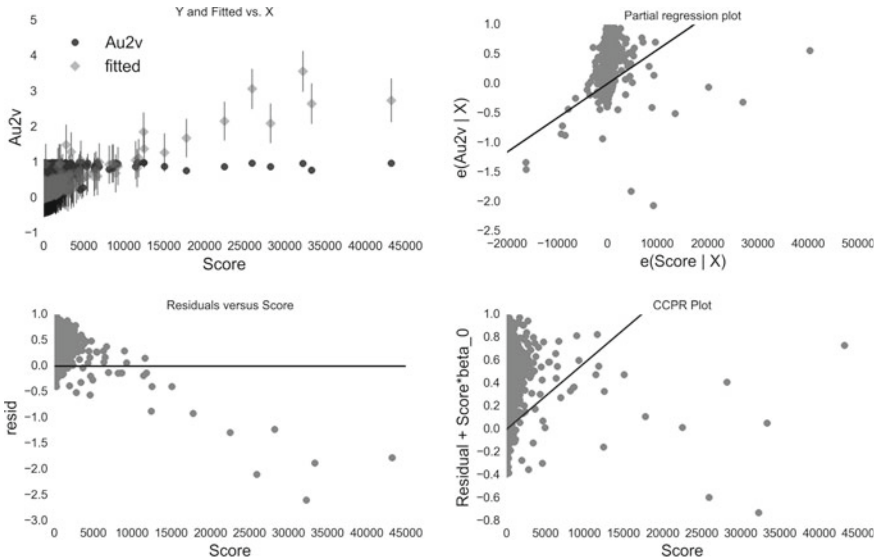

Fig. 20.2 Regression plots for $Score$

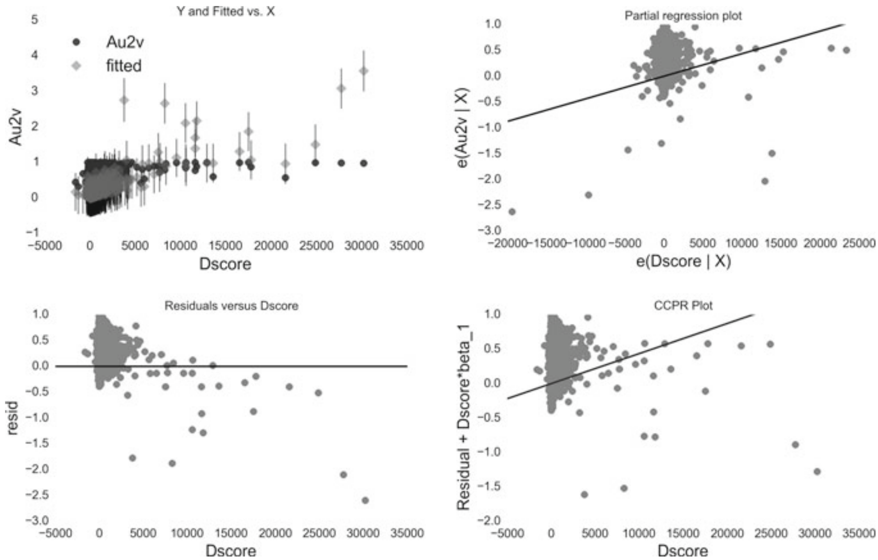

Fig. 20.3 Regression plots for $dScore$

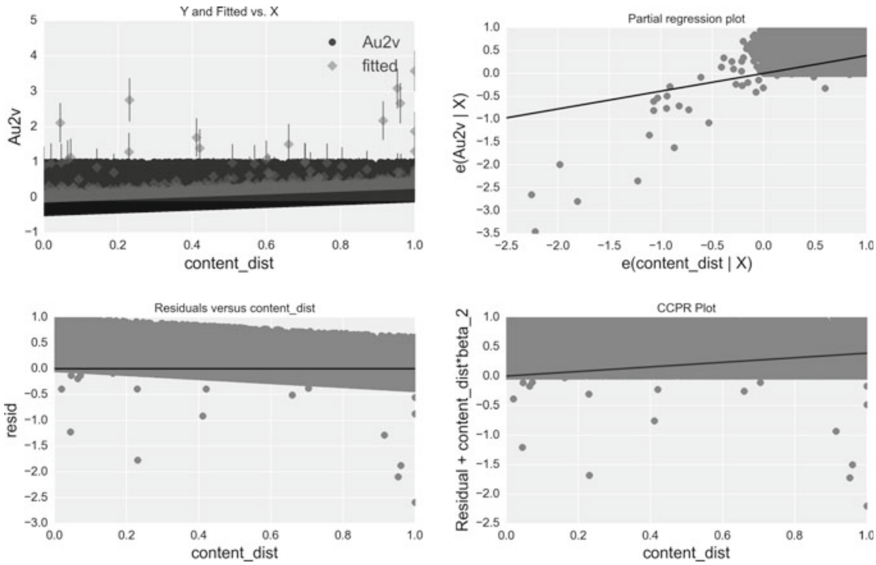

Fig. 20.4 Regression plots for *content_dist*

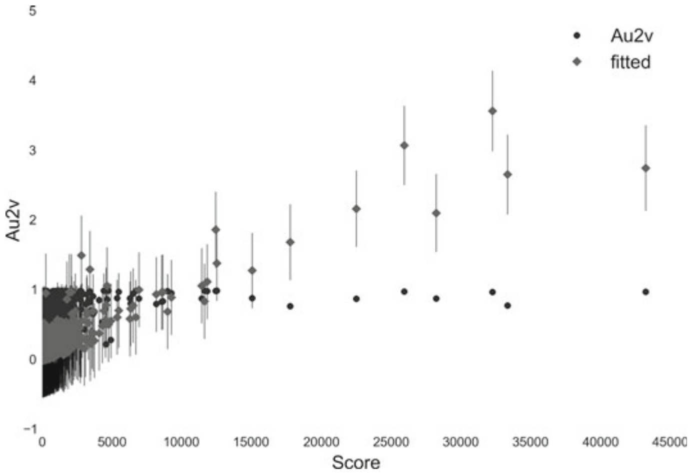

Fig. 20.5 Fitted values of A_{u2v} versus *Score*

respectively. The fitted (predicted) values of A_{u2v} and the prediction confidence for each independent variable appear in Figs. 20.5, 20.6, and 20.7. We observe that fitted values are quite close to the real values of A_{u2v} with the exception of the outliers. This suggests that removal of outliers would yield a better estimate, since it is obvious that the plot is skewed due to their presence.

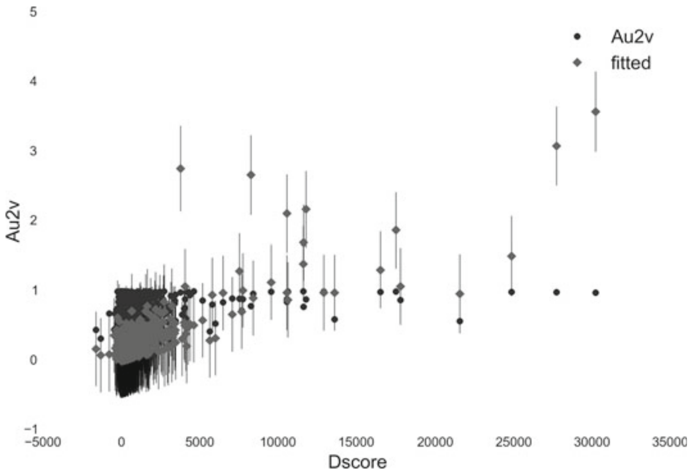

Fig. 20.6 Fitted values of A_{u2v} versus $dScore$

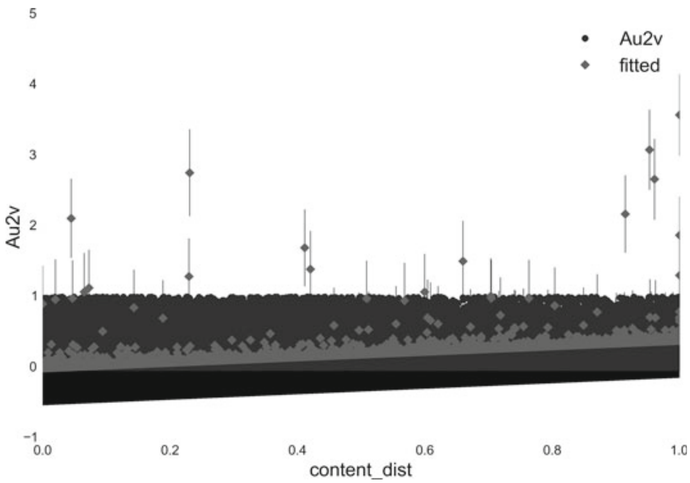

Fig. 20.7 Fitted values of A_{u2v} versus $content\_dist$

A rough estimate of detecting outliers can be based on the quantile distributions of each independent variable in Table 20.5. Observing Table 20.5 with an overview of data distribution we surmise that we could take values of $Score$ and $dScore$ only upto 10 and 5, respectively. The quantiles appearing on the table are calculated when data is rearranged in ascending order and divided into four equal sized parts. Thus, interpreting the second quantile we notice that 50 % of Score values are less than 2.562232. In the table, we notice that we have huge maximum values for $Score$ and $dScore$, but 75 % of the data are below 6.902750 and 2.308805, respectively. Thus,

Table 20.5 Outliers thresholds

	Score	*dScore*	*Content_dist*
Count	71952.000000	71952.000000	71952.000000
Mean	21.227703	15.880803	0.459111
Std	349.102908	292.727717	0.315837
Min	0.000000	–1610.253490	0.000000
25 %	0.787060	0.000140	0.172534
50 %	2.562232	0.526050	0.415394
75 %	6.902750	2.308805	0.724986
Max	43262.678131	30235.027960	1.000000

Table 20.6 Regression results without outliers (i)

Dep. variable	A_{u2v}	R-squared	0.629
Model	OLS	Adj. R-squared	0.629
Method	Least Squares	F-statistic	3.072e+04
Prob (F-statistic)	0.00	Log-Likelihood	13947
No. Observations	54473	AIC	−2.789e+04
Df Residuals	54470	BIC	−2.786e+04
Df Model	3	Covariance Type	nonrobust

Table 20.7 Regression results without outliers (ii)

	Coef	Std err	t	$P > \|t\|$	95 %	Conf.Int.
Score	0.1460	0.001	145.244	0.000	0.144	0.148
dScore	0.0200	0.001	25.819	0.000	0.018	0.022
con_dist	0.1656	0.003	65.690	0.000	0.161	0.171

Table 20.8 Regression results without outliers (iii)

Omnibus	10848.216	Durbin-Watson	1.966
Prob(Omnibus)	0.000	Jarque-Bera (JB)	22428.486
Skew	1.183	Prob (JB)	0.00
Kurtosis	5.070	Cond. No.	5.19

we select 10 and 5 as values to take most of the data and exclude data points with extremely large out of general range values (outliers).

Results of regression model on data obtained after removing outlier data points appear in Tables 20.6, 20.7, and 20.8. The results show considerable improvement with respect to regression with presence of outliers (Tables 20.2, 20.3 and 20.4). Also, Durbin–Watson statistic close to 2 confirms normality assumption of residuals, verifying the normality of error distribution, one of the assumptions of linear regression.

Figure 20.10 plots reinforce the argument that after removing outliers we get a better fit of regression line on each independent variable. Namely, the removal of outliers leads to better alignment of the path of regression line to the optimal path.

20.5.3 Tenfold Cross-Validation

We performed a tenfold cross validation on the dataset, fitting the regressor to 90 % of the data and validating it on the rest 10 % for the prediction of A_{u2v} dependent variable from $Score$, $dScore$ and $content - dist$ independent variables. Predictive modeling was conducted after removing outliers from the data. The results of the predictive modeling using linear regression show that we achieve a root mean squared error of 0.1873 (across all folds), which means that our prediction varies by 0.1873 from the real values of A_{u2v}. This shows a considerable improvement in prediction error compared to modeling with original data, where a root mean squared error of 0.2728 across all folds was achieved. Plots in both cases appearing in Figs. 20.8 and 20.9 depict how close our predictions are to the real values of the dependent variable (Fig. 20.10).

20.5.4 Classification and Comparison with Other Models

We predict a user popularity as follows. If A_{u2v} crosses a threshold, e.g. 30 %, i.e., if more than 30 % tweets of user u are retweeted by others users, then user u can be considered as a popular user.

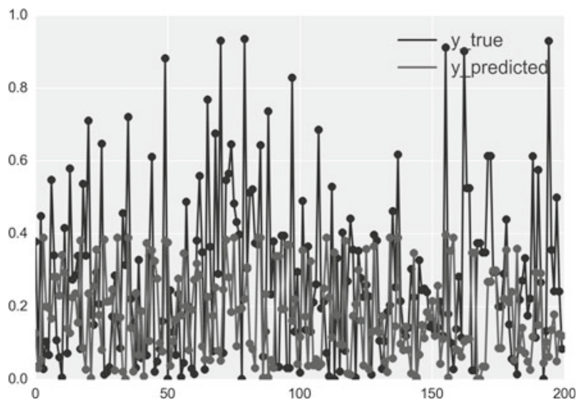

Fig. 20.8 tenfold cross-validation of A_{u2v}

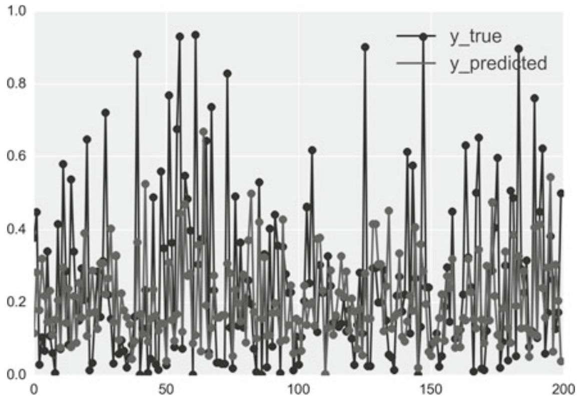

Fig. 20.9 tenfold cross-validation of A_{u2v} without outliers

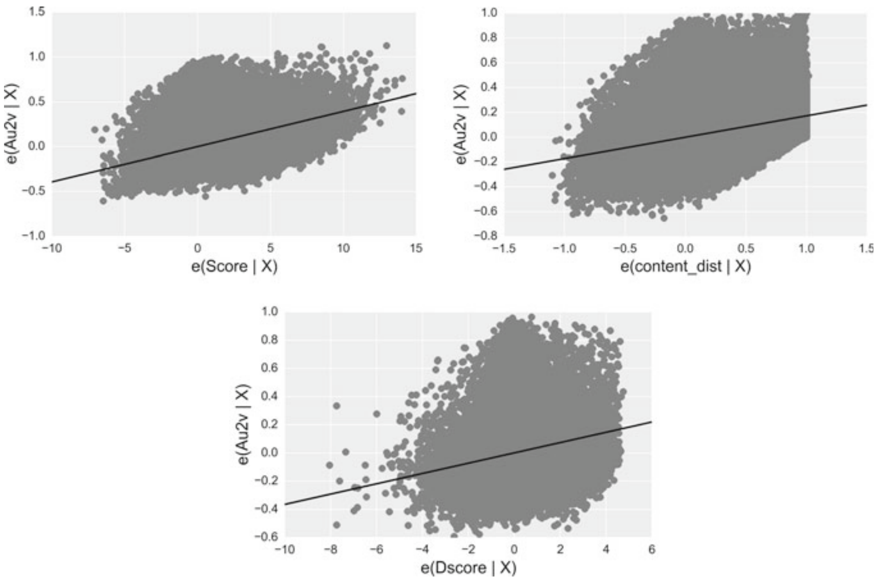

Fig. 20.10 Regression plots for each independent variable

Classification was conducted initially with three different methods: Linear Regression, i.e., the Predictive Model we present in this study, Random Forest and Naive Bayes methods. Area Under the Curve (AUC) is a score that computes average precision (AP) from prediction scores. This average precision score corresponds to the area under the precision-recall curve and the higher AUC represents better performance. Plots in Fig. 20.11 correspond to computed precision-recall pairs for different probability thresholds and the AUC score computes the area under these curves. Best performance is achieved by Linear Regression (0.699), followed by

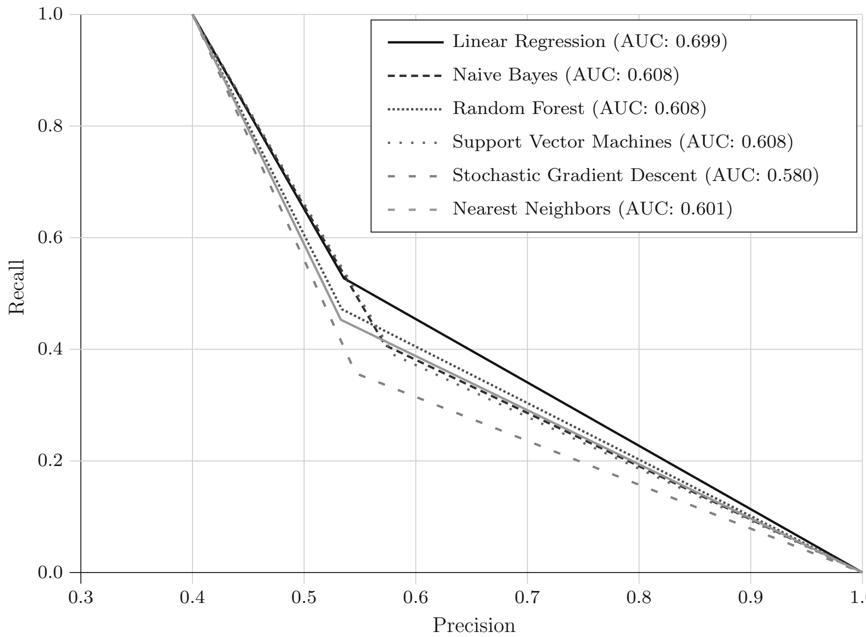

Fig. 20.11 Comparison with other models

Naive Bayes (AUC:0.608) and Random Forest (AUC:0.608). Complementary methods tested were support vector machines (SVM), stochastic gradient descent (SGD) and K-Nearest Neighbours (KNN).

SVM is a supervised learning model with associated learning algorithm that analyzes data used for classification and regression analysis. Given a set of training examples, each marked to belong to one of the two categories (popular/non-popular user), the SVM training algorithm builds a model that assigns new examples into each of the categories, acting as a non-probabilistic binary linear classifier.

Next classification model was stochastic gradient descent (SGD), a gradient descent optimization method for minimizing an objective function written as a sum of differentiable functions. It encompasses a popular algorithm for training a wide range of models in machine learning, including linear support vector machines, logistic regression and graphical models. Its use for training artificial networks is motivated by the high cost of running backpropagation algorithm over the full training set, as SGD overcomes this cost and still leads to fast convergence.

The last classifier implemented here was K-Nearest Neighbours (KNN), a method classifying objects based on closest training examples in the feature space. The input consists of positive, typically small, integer -15 in our case—of closest training examples in the feature space. In KNN classification, the output is a class membership (popular/non-popular user), whereas an object is classified by a majority vote of its neighbours, with the object being assigned to the class most common among its K-Nearest Neighbours.

After plotting the results of computed precision-recall pairs for various probability thresholds we observe that best performance is noticed in the case of our Predictive Model, followed by Naive Bayes (AUC:0.608), Random Forest (AUC:0.608), SVM (AUC:0.608), KNN (AUC:0.601), and, lastly, SGD (AUC:0.580).

20.6 Incorporation into Content Delivery Schemes

Content Distribution Networks (CDNs) aim at improving download of large data volumes with high availability and performance. Content generated by online media services circulates and is consumed over OSNs (with more than 400 tweets per minute including a YouTube video link [3] being published per minute). This content largely contributes to internet traffic growth [4]. Consequently, CDN users can benefit from an incorporated mechanism of social-awareness over the CDN infrastructure. In [10, 11] Kilanioti and Papadopoulos introduce a dynamic mechanism of preactive copying of content to an existing validated CDN simulation tool and propose various efficient copying policies based on prediction of demand on OSNs.

Rather than pushing data to all surrogates, they proactively distribute it only to social connections of the user likely to consume it. The content is copied only under certain conditions (content with high viewership within the media service, copied to geographically close timezones of the geo-diversed system used where the user has mutual social connections of high influence impact). This contributes to smaller response times for the content to be consumed (for the users) and lower bandwidth costs (for the OSN provider). Herein, we incorporate the proposed Predictive Model in the suggested policy [11] and prove that it further improves its performance.

The proposed algorithm encompasses an algorithm for each new request arriving in the CDN and an algorithm for each new object in the surrogate server (Table 20.9). Internally, the module communicates with the module processing the requests and each addressed server separately (Fig. 20.12).

- *For Every New Request in the CDN*
  Prinicipally we check whether specific time has passed after the start of cascade and, only in the case that the cascade has not ended, define to what extent the object will be copied. We introduce the *time_threshold* that roughly expresses the average cascade duration. The main idea is to check whether specific time has passed after the start of the cascade, and then define to what extent the object will be copied. Initially, we check whether it is the first appearance of the object (Fig. 20.13). The variable *o.timestamp* depicts the timestamp of the last appearance of the object in a request and helps in calculating the timer related to the duration of the cascade. If it is the first appearance of the object, the timer for the object cascade is initialized and *o.timestamp* takes the value of the timestamp of the request. If the cascade is

Table 20.9 Content delivery verification—notation overview

$G(t) = (V(t), E(t))$	Graph representing the social network
$V(t) = \{V_1(t), \ldots, V_n(t)\}$	Nodes representing the social network users
$E(t) = \{E_{11}(t), \ldots, E_{1n}(t), \ldots, E_{nn}(t)\}$	Edges representing the social network connections, where E_{ij} stands for friendship between i and j
$R = \{r_1, r_2, \ldots, r_\tau\}$	Regions set
$N = \{n_1, n_2, \ldots, n_\upsilon\}$	The surrogate servers set. Every surrogate server belongs to a region r_i
$C_i, i \in N$	Capacity of surrogate server i in bytes
$O = \{o_1, o_2, \ldots, o_w\}$	Objects set (videos), denoting the objects users can ask for and share
$S_i, o_i \in O$	Size of object i in bytes
Π_i	Popularity of object i, $i \in O$
$q_i = \{t, V_\psi, o_x\}, < x < w, 1 < \psi < n$	Request i consists of a timestamp, the id of the user that asked for the object, and the object id
$P = \{p_{12}, p_{13}, \ldots, p_{nw}\}$	User posts in the social network, where p_{ij} denotes that node i has shared object j in the social network
$pts_i, pte_i, 1 < i < \tau$	Peak time start and peak time end for each region in secs
$Q = \{q_1, q_2, \ldots, q_\zeta\}$	Object requests from page containing the media objects, where q_i denotes a request for an object of set O
Q_{hit}, Q_{total}	Number of requests served from surrogate servers of the region of the user/total number of requests
$X, Y \in R$	Closest timezones with mutual followers/with highest centrality metric values

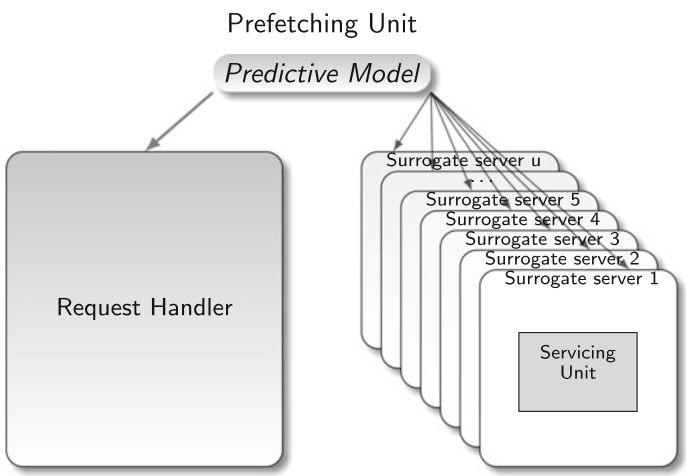

Fig. 20.12 The social-aware CDN mechanism

if $o.timestamp == 0$ **then**
  $o.timer = 0$;
  $o.timestamp = request\_timestamp$;
else if $o.timestamp$!= 0 **then**
  $o.timer = o.timer + (request\_timestamp - o.timestamp)$;
  $o.timestamp = request\_timestamp$;
end if
if $o.timer > time\_threshold$ **then**
  $o.timer = 0$;
  $o.timestamp = 0$;
else if $o.timer < time\_threshold$ **and** $user.Score > Score\_threshold$ **then**
  copy object o to surrogate that serves user's $V_i(t)$ timezone;
  for all user $V_y(t)$ that follows user $V_i(t)$ **do**
    find surrogate server n_j that serves $V_y(t)$'s timezone;
    copy object o to n_j;
  end for
else if o.timer $<$ time_threshold **then**
  copy object o to surrogates n_j that Subpolicy decides;
end if

Fig. 20.13 Algorithm for every new request ($timestamp$, $V_i(t)$, o) in the CDN

not yet complete (its timer has not surpassed a threshold), we check the importance of the user applying its Score.

For users with Score surpassing a threshold (average value: 1.2943 in the dataset), we copy the object to all surrogate servers of the user's timezone and to the surrogate servers serving the timezones of all user's followers. Otherwise, selective copying includes only the surrogates that the subpolicy decides. Subpolicy (Fig. 20.14) checks the X closest timezones where a user has mutual friends and out of them, the Y with the highest value of the combined feature set (Predictive Model($Score$, $dScore$, $content\_dist$)) as an average. Copying is performed to the surrogate servers that serve the Y timezones of highest combined feature set value, according to the coefficients derived from our analysis. We note here that variations of the Subpolicy include the replacement of the timezones depicting the highest average values of Predictive Model($Score$, $dScore$, $content\_dist$), with those being derived from the application of Naive Bayes, Random Forest, SVM, SGD, and KNN schemes.

find X timezones where (user $V_i(t)$ has mutual followers **and** they are closer to user's $V_i(t)$ timezone);
find the $Y \subseteq X$ that (belong to X **and** depict the highest average values of Predictive Model($Score$, $dScore$, $content\_dist$));
for all timezones that belong to Y **do**
  find surrogate server n_j that serves timezone;
  copy object o to n_j;
end for

Fig. 20.14 Subpolicy

```
if o.size + current_cache_size ≤ total_cache_size then
    copy object o to cache of surrogate n_k;
else if o.size + current_cache_size > total_cache_size then
    while o.size + current_cache_size > total_cache_size do
        for all object o' in current_cache do
            if (current_timestamp - o'.timestamp) + o'.timer > time_threshold then
                copy o' in CandidateList;
            end if
            if CandidateList.size>0 and CandidateList.size != total_cache_size then
                find o' that o'.timestamp is maximum and delete it;
            else if CandidateList.size==0 or CandidateList.size==total_cache_size then
                use LRU to delete any object o ∈ O;
            end if
        end for
    end while
    put object o to cache of surrogate n_k;
end if
```

Fig. 20.15 Algorithm for every new object o in the surrogate server n_k

- *For Every New Object in the Surrogate Server*
  Surrogate servers keep replicas of the web objects on behalf of content providers. In the case that the new object does not fit in the surrogate server's cache, we define the *time_threshold* as the parameter for the duration that an object remains cached. We check for items that have remained cached for a period longer than the *time_threshold* and we delete those with the largest timestamp in the cascade. In case there exist no such objects or all objects have the same timestamp, we prune the least recently used items first. To ensure that least recently used items are discarded, the algorithm keeps track of their usage (Fig. 20.15).

The nodes representing the surrogate servers, the origin server, and the users requesting the object (Fig. 20.16) in the simulated network topology are analyzed in detail in [10]. To simulate our policy and place the servers in a real geographical position, we used the geographical distribution of the Limelight network.

For the smooth operation of the simulator the number of surrogate servers was reduced by a ratio of 10 %, to ultimately include 423 servers. Depending on the closer distance between the surrogate region defined by Limelight and each of the timezones defined by Twitter (20 Limelight regions, 142 Twitter timezones), we decided where the requests from each timezone will be redirected. The population of each timezone was also taken into consideration. The INET generator [4] allowed us to create an AS-level representation of the network topology. Topology coordinates were converted to geographical coordinates with the NetGeo tool from CAIDA, a tool that maps IP addresses and Autonomous System (AS) coordinates to geographical coordinates, and surrogate servers were assigned to topology nodes. After grouping users per timezone (due to the limitations the large dataset imposes), each group of users was placed in a topology node. We placed the user groups in the nodes closer to those comprising the servers that serve the respective timezone requests, contributing this way to a realistic network depiction.

The heuristics applied in [11] are based on the observation that users are more influenced by geographically close friends, and moreover by mutual followers, as

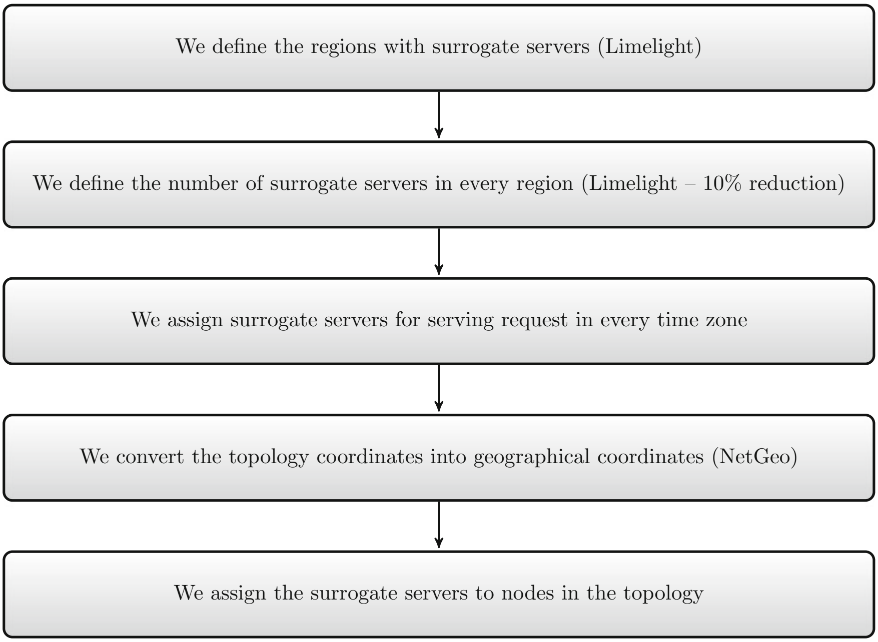

Fig. 20.16 Methodology followed

well as on the short duration of social cascades (about 80 % of the cascades end within 24 h, with 40 % of them ending in less than 3 h). In our prefetching algorithm, we introduce varying time thresholds for the cascade effect and the time an object remains in cache. Values given in the time threshold variable include thresholds covering the entire percentage of requests.

We examine Mean Response Time (MRT), a client-side metric that indicates how fast a CDN client is satisfied, for the most representative case of time threshold covering all the examined requests of our dataset. The trade-off between the reduction of the response time and the cost of copying in servers is expressed for all schemes used (Linear Regression, Naive Bayes, Random Forest, SVM, SGD, KNN) with an MRT decrease as the timezones increase and a point after which the MRT starts to increase again (Fig. 20.17). For the scheme augmented with our Predictive Model, namely the Linear Regression, this shift occurs with approximately 6 timezones out of the 10 used (for a fixed number of closest timezones with mutual followers). After this point the slight increase in the MRT is attributed to the delay for copying content to surrogate servers. The cost for every copy is related to the number of hops among the client asking for it and the server where copying is likely to take place. We observe that Linear Regression outperforms all the other schemes, depicting MRTs smaller than their respective. We note here that timezones with highest average values for each scheme, that Subpolicy defines, are precalculated, in order to reduce computational burden in the simulations.

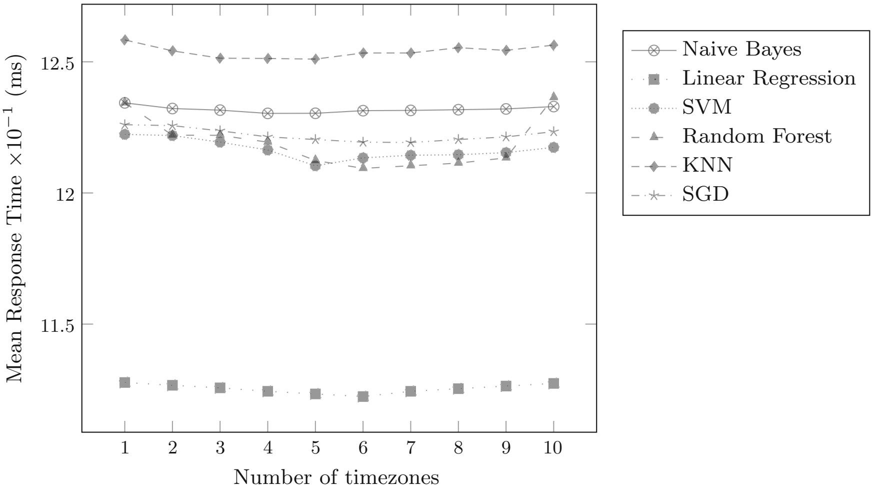

Fig. 20.17 Effect of timezones used as Y on Mean Response Time for various schemes ($X = 10$ closest timezones with mutual followers)

20.7 Conclusions

We come to the conclusion that video sharings over an OSN platform can be predicted with a small set of features extracted from both the platform and the media service. Despite the focused scope of this work and the limitations of its conduction solely with Twitter and YouTube data, the scale of the medium allows us to make assumptions for generalization across different OSNs and microblog platforms. We plan to extensively analyze this generalization in the future. Future extensions also include experimentation with variations of content distance interpretation among users, with various score assignment formulas, as well as subsequent verification in the realm of content delivery. We hope that our findings will broaden the view on the spread of information in web today.

References

1. Abisheva, A., Garimella, V.R.K., Garcia, D., Weber, I.: Who watches (and shares) what on YouTube? And when?: Using Twitter to understand Youtube viewership. IN: Proceedings of the 7th ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, NY, USA, 24–28 Feb 2014, pp. 593–602 (2014). doi:10.1145/2556195.2566588
2. Bakshy, E., Hofman, J.M., Mason, W.A., Watts, D.J.: Everyone's an influencer: quantifying influence on Twitter. In: Proceedings of the 4th International Conference on Web Search and Web Data Mining, WSDM 2011, Hong Kong, China, 9–12 Feb 2011, pp. 65–74 (2011). doi:10.1145/1935826.1935845

3. Brodersen, A., Scellato, S., Wattenhofer, M.: YouTube around the world: geographic popularity of videos. In: Proceedings of the 21st World Wide Web Conference, WWW 2012, Lyon, France, 16–20 April 2012, pp. 241–250 (2012). doi:10.1145/2187836.2187870
4. Cha, M., Kwak, H., Rodriguez, P., Ahn, Y., Moon, S.B.: I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system. In: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement, IMC 2007, San Diego, California, USA, 24–26 Oct 2007, pp. 1–14 (2007). doi:10.1145/1298306.1298309
5. Cheng, J., Adamic, L.A., Dow, P.A., Kleinberg, J.M., Leskovec, J.: Can cascades be predicted? In: Proceedings of the 23rd International World Wide Web Conference, WWW 2014, Seoul, Republic of Korea, 7–11 April 2014, pp. 925–936 (2014). doi:10.1145/2566486.2567997
6. Dow, P.A., Adamic, L.A., Friggeri, A.: The anatomy of large Facebook cascades. In: Proceedings of the 7th International Conference on Weblogs and Social Media, ICWSM 2013, Cambridge, Massachusetts, USA, 8–11 July 2013
7. Galuba, W., Aberer, K., Chakraborty, D., Despotovic, Z., Kellerer, W.: Outtweeting the Twitterers—Predicting Information Cascades in Microblogs. In: Proceedings of the 3rd Workshop on Online Social Networks, WOSN 2010, Boston, MA, USA, 22 June 2010
8. Hong, L., Dan, O., Davison, B.D.: Predicting popular messages in Twitter. In: Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28–April 1, 2011 (Companion Volume), pp. 57–58 (2011). doi:10.1145/1963192.1963222
9. Jenders, M., Kasneci, G., Naumann, F.: Analyzing and predicting viral tweets. In: Proceedings of the 22nd International World Wide Web Conference, WWW 2013, Rio de Janeiro, Brazil, 13–17 May 2013, Companion Volume, pp. 657–664 (2013)
10. Kilanioti, I.: Improving multimedia content delivery via augmentation with social information. The Social Prefetcher approach. IEEE Trans. Multimedia **17**(9), 1460–1470 (2015). doi:10.1109/TMM.2015.2459658
11. Kilanioti, I., Papadopoulos, G.A.: Socially-aware multimedia content delivery for the cloud. In: Proceedings of the 8th IEEE/ACM International Conference on Utility and Cloud Computing, UCC 2015, Limassol, Cyprus, 7–10 Dec 2015, pp. 300–309 (2015). doi:10.1109/UCC.2015.48
12. Kupavskii, A., Ostroumova, L., Umnov, A., Usachev, S., Serdyukov, P., Gusev, G., Kustarev, A.: Prediction of retweet cascade size over time. In: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, CIKM 2012, Maui, HI, USA, October 29–November 02, 2012, pp. 2335–2338 (2012). doi:10.1145/2396761.2398634
13. Kwak, H., Lee, C., Park, H., Moon, S.B.: What is Twitter, a social network or a news media? In: Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, 26–30 April 2010, pp. 591–600 (2010). doi:10.1145/1772690.1772751
14. Ma, Z., Sun, A., Cong, G.: On predicting the popularity of newly emerging hashtags in Twitter. JASIST **64**(7), 1399–1410 (2013). doi:10.1002/asi.22844
15. Petrovic, S., Osborne, M., Lavrenko, V.: RT to win! Predicting message propagation in Twitter. In: Proceedings of the 5th International Conference on Weblogs and Social Media, ICWSM 2011, Barcelona, Catalonia, Spain, 17–21 July 2011
16. Rodrigues, T., Benevenuto, F., Cha, M., Gummadi, P.K., Almeida, V.A.F.: On word-of-mouth based discovery of the web. In: Proceedings of the 11th ACM SIGCOMM Conference on Internet Measurement, IMC 2011, Berlin, Germany, 2–4 Nov 2011, pp. 381–396 (2011). doi:10.1145/2068816.2068852
17. Suh, B., Hong, L., Pirolli, P., Chi, E.H.: Want to be retweeted? Large scale analytics on factors impacting retweet in Twitter network. In: Proceedings of the 2nd IEEE International Conference on Social Computing, SocialCom / IEEE International Conference on Privacy, Security, Risk and Trust, PASSAT 2010, Minneapolis, Minnesota, USA, 20–22 Aug 2010, pp. 177–184 (2010). doi:10.1109/SocialCom.2010.33
18. Szabó, G., Huberman, B.A.: Predicting the popularity of online content. Commun. ACM **53**(8), 80–88 (2010). doi:10.1145/1787234.1787254
19. Tsur, O., Rappoport, A.: What's in a hashtag? Content based prediction of the spread of ideas in microblogging communities. In: Proceedings of the 5th International Conference on Web

Search and Web Data Mining, WSDM 2012, Seattle, WA, USA, 8–12 Feb 2012, pp. 643–652 (2012). doi:10.1145/2124295.2124320
20. Yang, J., Leskovec, J.: Patterns of temporal variation in online media. In: Proceedings of the 4th International Conference on Web Search and Web Data Mining, WSDM 2011, Hong Kong, China, 9–12 Feb 2011, pp. 177–186 (2011). doi:10.1145/1935826.1935863
21. Yang, S., Zha, H.: Mixture of mutually exciting processes for viral diffusion. In: Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16–21 June 2013, pp. 1–9 (2013)
22. Zaman, T.R., Herbrich, R., Van Gael, J., Stern, D.: Predicting information spreading in Twitter. In: Proceedings of the Workshop on Computational Social Science and the Wisdom of Crowds, Nips, vol. 104, pp. 17, 599–601 (2010)