A Study on Microblog and Search Engine User Behaviors: How Twitter Trending Topics Help Predict Google Hot Queries

Federica Giummolè
Università Ca' Foscari Venezia, Italy
Email: giummole@unive.it

Salvatore Orlando
Università Ca' Foscari Venezia, Italy
Email: orlando@unive.it

Gabriele Tolomei
Università Ca' Foscari Venezia, Italy
Email: gabriele.tolomei@unive.it

ABSTRACT

Once every five minutes, *Twitter* publishes a list of *trending topics* by monitoring and analyzing tweets from its users. Similarly, *Google* makes available hourly a list of *hot queries* that have been issued to the search engine. We claim that *social trends* fired by Twitter may help explain and predict *web trends* derived from Google. Indeed, we argue that information flooding nearly real-time across the Twitter social network could anticipate the set of topics that users will later search on the Web. In this work, we analyze the time series derived from the *daily volume index* of each trend, either by Twitter or Google. Our study on a real-world dataset reveals that about 26% of the trending topics raising from Twitter "as-is" are also found as hot queries issued to Google. Also, we find that about 72% of the *similar* trends appear first on Twitter. Thus, we assess the relation between comparable Twitter and Google trends by testing three classes of time series regression models. First, we find that Google by its own is not able to effectively predict the time behavior of its trends. Indeed, we show that autoregressive models, which try to fit time series of Google trends, perform poorly. On the other hand, we validate the *forecasting* power of Twitter by showing that models, which use Google as the *dependent* variable and Twitter as the *explanatory* variable, retain as significant the past values of Twitter 60% of times. Moreover, we discover that a Twitter trend *causes* a similar Google trend to later occur about 43% of times. In the end, we show that the very best-performing models are those using past values of *both* Twitter and Google.

I INTRODUCTION

Twitter[1] is one of the most popular online social media and microblogging platform where people share information nearly real-time by reading and writing so-called *tweets*. Each tweet posted by a user is a text message, which at most may contain 140 characters. Typically, a tweet expresses opinions or feelings about something referring to either personal or public interests. However, people, governments, and institutions are increasingly using Twitter for public safety management, especially in case of emergency [1,2], as well as for sensitizing public awareness of real-world social issues (e.g., to drive the revolutionary protests during the so-called *Arab spring* [3]).
Also, the size of the Twitter network is rapidly growing and has already surpassed 500 million registered users [4]. Thereby, the number of tweets users are exchanging daily is increasing as well (i.e., about 400 million as of June 2012 [5]).

From all the considerations above, it turns out that Twitter may rely on a huge amount of user-generated data, which can be analyzed to provide better monetization for the company and new services for the end users (e.g., *personal advertising*).
Some analyses aim at showing what are the *topics* that users are most interested in or talking of. To this end, Twitter periodically extracts and publishes a list of the top-10 *trending topics*, namely text strings referring to *social trends*.
Each trending topic is the succinct textual representation of a "standing out fact", as extracted from user tweets. It may either refer to a long-lasting or a sudden effect on the volume of tweets (e.g., nba *vs.* election 2012). Unfortunately, the details about the technique Twitter uses to generate its lists of trending topics are not publicly known, even though recent work is able to infer them with high accuracy [6].

Similarly to Twitter, once every hour *Google*[2] releases a list of the top-20 trending search keywords (i.e., the so-called *hot trends* or *hot queries*), which we refer to as *web trends*.
Google allows to quantitatively examine how trending is a hot query. Specifically, it computes the search fraction of a given query in a time range and a geographical region. This analysis indicates the likelihood of a random user to ask for a query from a certain location at a certain time.

[1] http://www.twitter.com
[2] http://www.google.com

The aim of this work is to investigate whether any relationship occurs between social trends as extracted from Twitter and web trends as output by Google. Intuitively, we claim that the same topics that appear as trending on Twitter could *later* become a trending query on Google. The rationale of this intuition is that information flooding nearly real-time across the Twitter social network could anticipate the set of topics that users will be interested in – and consequently will search for by means of queries – in the next future.
Concretely, this manuscript extends a previous work of ours [7], and provides the following contributions. First, we deeply describe the *Trend Bipartite Graph* (TBG), which is used to represent the lexical similarity between any pair of social and web trends, as extracted from real-world Twitter and Google datasets. The TBG uses a *threshold* to link those trends that are most likely related.
Second, we evaluate the *cross-correlation* between time series derived from Twitter and Google trends, which are linked according to the TBG.
Last, we measure the ability of Twitter in actually *predicting* and *causing* a Google trend to later occur by conducting an exhaustive comparison of several time series regression models. This final step shows that models that include Twitter in the regression function better fit and forecast our time series data.
Though we do not discuss how our findings could be exploited here, we guess they may lead to several search engine optimization strategies, e.g., the preemptive caching of fresh results for queries that very likely will trend, or the focused and dynamic refreshing of crawled pages.

The rest of the work is organized as follows. Section II introduces the motivation behind our work through a preliminary study of some real-world examples. In Section III, we give an overview of the most valuable work about social network analysis, especially focused on Twitter, and time series regression from Web data. In Section IV we discuss some useful principles about time series analysis that will be used all along the paper. Section V describes the research steps we pursue to explore whether and what relationship may occur between social and web trends. Section VI presents the experiments we conducted, and discusses the results we obtained. Finally, in Section VII we summarize our work and figure out possible future research directions.

II WHICH CAME FIRST: TWITTER OR GOOGLE?

To motivate our work, we present a preliminary study of some real-world examples of social and web trends, which exhibit interesting relationships. Furthermore, we choose Twitter and Google out of many other available microblogs and commercial search engines because they are undoubtedly the most widely used, at least in the U.S.. Clearly, other interesting analyses may be conducted on different platforms (e.g., YouTube vs. Bing), and they are left as subjects of future work.

In Fig. 1, we show the plots of four pairs of time series, which describe the *daily volume indexes* of four actual trends. These trends are shared between Twitter and Google during the first two weeks of November 2012, and they refer to two people (i.e., Doug Martin and Nick Foles are two football players) and two events (i.e., the U.S. Presidential Elections and the New York City Marathon).
At this stage, we only aim to give hints on why our research is worth to explore, and the concepts of *trend time series* and *trend volume index* will be clarified in the upcoming sections.
The main issue concerns the different time granularity of observations of trend volumes as derivable from Twitter and Google. In principle, since we can collect a large percentage of all the real tweets, we can derive a hourly (or even finer-grained) trend volume index for its trends. Unfortunately, Google only publishes a daily-aggregate analysis of its web trend volumes. Putting all together, no finer-grained results (e.g., hourly-based) can be produced from trend data. This forces us to plot our trend time series on a daily basis.
Though the limitation above, plots in Fig. 1 still reveal that a *relation* exists between each pair of Twitter's and Google's trend time series. Moreover, we may characterize this relation by inspecting the first-time occurrence of a trend. As opposed to the volume indexes, the first-time occurrence of a trend can be derived hourly from both Twitter and Google. Indeed, Twitter updates its set of trending topics once every five minutes whereas Google does it every hour. We find that 66% of times the same trend appear first on Twitter than on Google. This percentage raises up to 72% if we pair trends that are not matching *exactly* yet having high lexical similarity according to our proposed *Trend Bipartite Graph* (TBG).
Interestingly enough, Twitter anticipates Google with a median value of 15 hours, which, of course, may not be fully appreciated when performing a 24-hour ag-

gregate analysis. Clearly, there is about 25% cases where the opposite happens, namely when a trend is fired from Google first, while the remaining are trends that appear at the same time.
It turns out that, to better capture and qualify the relation between Twitter and Google trends, we should have worked with at least *hourly*, instead of daily, time series. Nevertheless, we are still convinced that a daily analysis is worthwhile to be conducted because of two reasons: *(i)* there is evidence of some trends appearing on Twitter *more* than 24 hours before they are fired by Google (e.g., see the top-right trend election in Fig. 1); *(ii)* we might still appreciate those trends who come on Twitter just few hours before Google – such as those depicted in the remaining plots of Fig. 1 – by looking at the *sharpness* of the slopes of their daily-aggregate behaviors.

III RELATED WORK

Two distinct lines of research are explored in this work: *(i) Analysis of Social Network Data* and *(ii) Time Series Regression from Web Data*. In the following, we discuss both of them separately.

1 ANALYSIS OF SOCIAL NETWORK DATA

In the recent years, we have seen the sudden rising and exponential growth of many online *social network applications*, such as *Flickr*, *MySpace*, *Facebook*, *Google+*, just to name a few. Besides all the above, *Twitter* has emerged as one of the most influential online social media service. Thereby, several studies have started analyzing data from Twitter.

Many work aim at classifying different types of users, their behaviors, and the relationships occurring among them according to the *following/follower* pattern [8–11].
Other studies focus on analyzing the content of the tweets (e.g., to get insights about opinions and/or sentiments [12]), and the way these are related to trending topics. In this last regard, one of the most representative and exhaustive study is proposed by Kwak *et al.* [13]. Among other contributions, the authors describe the relation between tweets and trending topics extracted from Twitter, and trends derived from other media, i.e., search queries on Google and CNN headlines. Although the subject above seems to be highly similar to the one we tackle in this paper, we would like to remark that we compare Twitter and Google trends from a very different perspective. Specifically, Kwak *et al.* aim at checking if Twitter trends and Google hot queries overlap (in fact, the authors consider a trending topic and a search keyword a match if the length of the longest common substring is more than 70% of either string). Instead, our goal is to test – through time series regression analysis – if Twitter trending topics can be used to model, explain, and predict the volume of Google hot queries.
Ruiz *et al.* [14] exploit the general findings on Twitter discussed in [13] yet studying the problem of correlating microblogging activity with stock market events. To achieve this goal, the authors use a graph representation of tweets, whose nodes are different entities (e.g., tweets, users, hashtags, and URLs) and edges model relationships between these entities (e.g., retweet between two tweets, URL presence between a URL and the tweets that contain it, etc.).
However, other past work used a graph to represent the relationships among users, entities, and topics in Twitter. For instance, Weng *et al.* [15] try to identify similar users on the basis of their favorite topics and their social connections. To this end, the authors apply a modified version of PageRank to find the most influential authors on the Twitter graph.

2 TIME SERIES REGRESSION FROM WEB DATA

Using Web data for predicting the behavior of a real time series is a well-investigated topic. However, to the best of our knowledge, this work is the first attempt trying to relate time series derived *both* from Web search and social network data.

Liu *et al.* [16] propose a unified model to predict the upcoming query trends on the basis of observations extracted from the query log of a commercial search engine. In a nutshell, the authors integrate classical time series models with the Cosine Hidden Periodicities Model in order to capture periodic information of query time series. Several differences between this work and our approach can be found. First, trend prediction works only for queries stored in the query log, thereby issued in the past. Moreover, our solution does not aim to introduce a new way of detecting upcoming web trends. In fact, we trustfully use query trends as they are provided by Google. Finally, we claim that external factors influencing web trends can be effectively captured by using an external signal as well, just as Twitter. This last aspect allows us to keep the prediction model simple without the need of combining multiple models together.

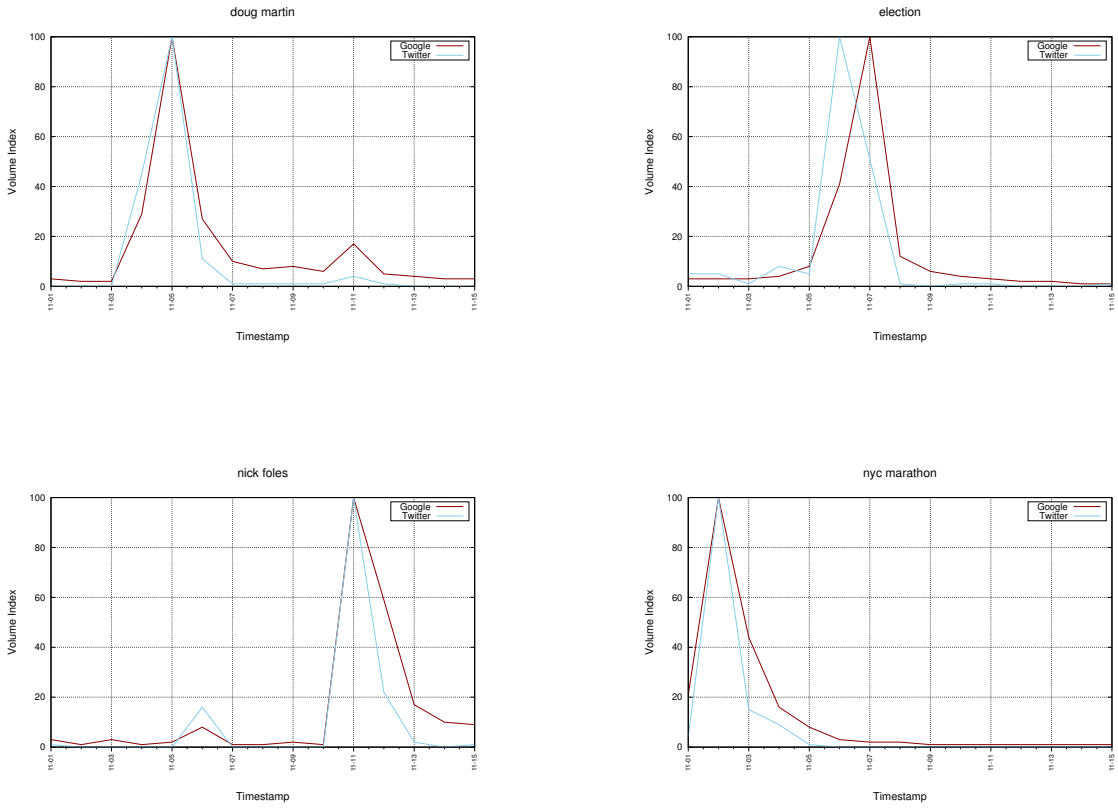

Figure 1: Time series of trend volumes as measured by Twitter and Google.

Recent work prove that Web search volume can predict the values of some economic indicators. For instance, Bordino *et al.* [17] show that daily trading volumes of NASDAQ-100 stocks are correlated with daily volumes of queries about the same stocks. In particular, query volumes often lead peaks of trading by one day or more. Ettredge *et al.* [18] use search logs to predict the job market while Choi and Varian [19] show how Google trends may be used to forecast unemployment levels, car and home sales, and disease prevalence in near real-time [19].
Goel *et al.* [20] show that what users are searching for online can also predict their collective future behavior days or even weeks in advance. The authors use search query volume to forecast the opening weekend box-office revenue for feature films, first-month sales of video games, and the rank of songs on the Billboard Hot 100 chart, finding that search counts are highly predictive of future outcomes. Ginsberg *et al.* [21] approximate the flu cases in the U.S. by using a search engine query log whereas Corely *et al.* [22] address a similar problem yet exploiting blog content.

From our side, we also conduct another research whose findings are described in [23]. There we show that, most of the time, trending topics on Twitter are actually referring to *named-entities* (e.g., people names, organizations, places, etc.). Several efforts have been done to collect and organize those entities, and *Wikipedia*[3] is undoubtedly the biggest and most representative knowledge base of entities to date. Therefore, trending topics may be easily mapped to the corresponding entities, namely linked to the referent Wikipedia articles. Following the same intuition motivating the present work, we show that Twitter is able to anticipate the volume of requests that users make to Wikipedia pages.

[3] `http://www.wikipedia.org`

IV TIME SERIES ANALYSIS

In this section, we introduce some basic concepts and notations about *time series*. Then, we discuss a set of techniques used in this paper for extracting knowledge from the time series concerning Twitter and Google trends.

1 BASIC CONCEPTS AND NOTATIONS

Let $\mathcal{T} = \{t_1, t_2, \ldots, t_T\}$ be a parameter space representing time and containing T *discrete*, *equally-lasting*, and *equally-spaced* slots.
A *time series* is a time-ordered sequence of *random variables* defined on the same probability space and indexed by time slots:

$$\mathcal{X} = \{X_t,\ t \in \mathcal{T}\} = \{X_t\}_{t=t_1}^{t_T}. \tag{1}$$

A time series can be usefully described through the first and second-order moments (i.e., the *mean* and the *variance*) of its composing random variables. To this end, let $E[\cdot]$ be the *expectation operator*. Thus, we define $\mu_t = E[X_t]$ and $\sigma_t^2 = \mathrm{Var}(X_t) = E[X_t - \mu_t]^2$ the mean and variance of each X_t ($t \in \mathcal{T}$) as functions of time.
A crucial issue when dealing with time series concerns *stationarity*. We define a time series *strictly stationary* if its statistical properties do not change over time. Formally, $\mathcal{X} = \{X_t,\ t \in \mathcal{T}\}$ is strictly stationary if, for any $\{t_1, \ldots, t_q\} \subseteq \mathcal{T}$ and any τ, the joint distribution of $X_{t_1}, \ldots, X_{t_q}$ is the *same* as the joint distribution of $X_{t_1+\tau}, \ldots, X_{t_q+\tau}$.
Since this is an extremely strong property, which means that *all* moments of *all degrees* of the series are the same *anywhere*, independent of time, the *second order* or *weak* stationarity is instead often used. A time series $\mathcal{X} = \{X_t,\ t \in \mathcal{T}\}$ is *weakly stationary* if the means and variances of its random variables are constant over time, and for any $t_i, t_j \in \mathcal{T}$ the *covariance* between X_{t_i} and X_{t_j} is *finite* and only depends on the time *lag* $\delta = j - i$. Eventually, any time series that is not stationary, either strictly or weakly, is called *non-stationary*.
Let $\mathcal{X}$ and $\mathcal{Y}$ be two (weakly) stationary time series. By definition, the means and variances of all their random variables are constant, and we denote them by $\mu_\mathcal{X}, \mu_\mathcal{Y}$ and $\sigma_\mathcal{X}^2, \sigma_\mathcal{Y}^2$, respectively. Given $t \in \mathcal{T}$ and a lag δ, such that $t + \delta \in \mathcal{T}$, we define the *cross-covariance* as:

$$c_{XY}(\delta) = E[(X_{t+\delta} - \mu_\mathcal{X})(Y_t - \mu_\mathcal{Y})]. \tag{2}$$

Furthermore, we compute the *cross-correlation* as the cross-covariance normalized in the range $[-1, 1]$:

$$r_{XY}(\delta) = \frac{c_{XY}(\delta)}{\sqrt{\sigma_\mathcal{X}^2 \cdot \sigma_\mathcal{Y}^2}} = \frac{c_{XY}(\delta)}{\sigma_\mathcal{X} \cdot \sigma_\mathcal{Y}}, \tag{3}$$

where $\sigma_\mathcal{X}$ and $\sigma_\mathcal{Y}$ are the *standard deviations* of $\mathcal{X}$ and $\mathcal{Y}$.
Intuitively, the cross-correlation gives hints about the presence of correlation between two time series when time-shifted by the lag δ (i.e., *lagged relationship*).
In particular, when one or more $X_{t+\delta}$ are predictors of Y_t and $\delta < 0$, we say that X *leads* Y. Conversely, when one or more $X_{t+\delta}$ are predictors of Y_t and $\delta > 0$, we say that X *lags* Y (or, equivalently, Y *leads* X).
Many problems related to time series analysis deal with identifying which variable is leading and which is lagging. Some others assume a certain variable (X) to be leading of another one (Y), namely they aim to use the values of X to *predict* future values of Y.
However, it is worth remarking that cross-correlation is designed for stationary time series (at least in a weak sense). Indeed, estimating the cross-correlation between two non-stationary time series may lead to a misleading evaluation of their actual lagged relationship.

2 TIME SERIES REGRESSION

Regression analysis is one of the most powerful statistical tools for modeling relationships among variables. In its most general form, a *regression model* aims at relating a *dependent* variable Y to a parametric function of a set of *independent* variables (or *inputs*) $X_1, \ldots, X_r$.
The widest used is the *linear regression model*, which assumes that Y can be written as the sum of two terms. The first one is a deterministic component depending on $X_1, \ldots, X_r$ and linear in the parameters. The second term represents a random error component including all the influent factors on Y that are not considered in the deterministic component. Using matrix notation, it can be written as follows:

$$Y = \mathbf{X}\boldsymbol{\beta} + \epsilon,$$

where $Y = (Y_1, \ldots, Y_k)^T$ is a random vector, $\mathbf{X} = (x_{i,j})$ is a full rank $k \times r$ matrix of observed values for $X_1, \ldots, X_r$, $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_r)^T$ is an unknown r-dimensional parameter and $\epsilon = (\epsilon_1, \ldots, \epsilon_k)^T$ is the error component that is assumed to have a multivariate normal distribution with zero mean and uncorrelated (thus independent) components, $\epsilon \sim \mathcal{N}_k(0, \sigma^2 I_k)$, with I_k the identity matrix of order k.

The most common technique for estimating the coefficients $\boldsymbol{\beta}$ is *Ordinary Least Squares* (OLS), where values $\hat{\boldsymbol{\beta}} = (\hat{\beta}_1, \ldots, \hat{\beta}_r)^T$ are chosen so as to *minimize* the residual sum of squared [24, 25]:

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \{(Y - \mathbf{X}\boldsymbol{\beta})^T (Y - \mathbf{X}\boldsymbol{\beta})\}.$$

The resulting OLS estimator is thus computed as follows:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T Y.$$

The linear regression model can be fruitfully used for modeling relationships among time series. However, in this context new issues arise. In particular, one variable can influence another with a specific time lag. Furthermore, special care should be taken when dealing with non-stationary time series, since *spurious regression* may occur [26, 27].
In this work we focus on three different classes of time series regression models, that are briefly described below.

Autoregressive Models (AR). The simplest regression model for time series is the one relating a variable (Y_t) *only* to a linear combination of p of its lags ($Y_{t-1}, Y_{t-2}, \ldots, Y_{t-p}$). We call it autoregressive model, which is typically denoted by AR(p) and where p is the *lag order* of the model:

$$Y_t = \alpha + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \ldots + \phi_p Y_{t-p} + \epsilon_t, \quad (4)$$

Distributed Lag Models (DL). In some cases, however, a dependent variable could be *only* related to lags of an explanatory variable. We denote by DL(q) a distributed lag model, where a dependent time series variable as observed at a given time (Y_t) is related to a linear combination of the values that an explanatory time series variable assumes at the same time (X_t) up to q time lags ($X_{t-1}, X_{t-2}, \ldots, X_{t-q}$), and q is called the *lag order* of the model:

$$Y_t = \alpha + \psi_1 X_t + \psi_2 X_{t-1} + \ldots + \psi_{q+1} X_{t-q} + \epsilon_t. \quad (5)$$

Autoregressive Distributed Lag Models (ADL). Some analyses require using a regression model that has *both* lags of dependent and explanatory variables, or what we call autoregressive distributed lag model, denoted by ADL(p, q):

$$\begin{aligned} Y_t = \alpha + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \ldots + \phi_p Y_{t-p} + \\ + \psi_1 X_t + \psi_2 X_{t-1} + \ldots + \psi_{q+1} X_{t-q} + \delta t + \epsilon_t. \end{aligned} \quad (6)$$

A right estimation and interpretation of an ADL(p, q) model depends on whether the time series variables Y_t and X_t are stationary or not. Indeed, if the variables are stationary the model parameters can be safely estimated by using OLS. Conversely, if Y_t and X_t are non-stationary OLS can lead to wrong parameters estimation, or what we have referred to as *spurious regression problem*.
Anyway, spurious regression vanishes if Y_t and X_t are *cointegrated* [28], and the OLS estimation still works fine even for non-stationary time series.

V SOCIAL VS. WEB TRENDS ANALYSIS

In this section, we introduce and formalize the main elements and techniques used in our work.

Trend Vocabularies. Let $\mathcal{V}_X = \{x_1, x_2, \ldots, x_n\}$ be the *vocabulary* set of all the *trending topics* as provided by Twitter. Similarly, we denote by $\mathcal{V}_Y = \{y_1, y_2, \ldots, y_m\}$ the *vocabulary* set of all the *hot queries* released by Google.
It is worth remarking that the keywords of both vocabularies are not necessarily single-term, but may be composed of a sequence of terms.

Trend Scores. We refer to $\mathcal{T} = \langle t_1, t_2, \ldots, t_T \rangle$ as a discrete time interval. Functions s_X and s_Y assign *scores* to vocabulary keywords over time $\mathcal{T}$:

$$s_X : \mathcal{V}_X \times \mathcal{T} \longmapsto \mathbb{N}, \; s_Y : \mathcal{V}_Y \times \mathcal{T} \longmapsto \mathbb{N}.$$

For each keywords, they indicate the "strength" of its trending in a given time slot, as measured by Twitter and Google.

Trend Time Series. Each Twitter/Google trend can be modeled by a time series, composed of t_T *random variables*, namely $\mathcal{X} = \{X_t\}_{t=t_1}^{t_T}$ for Twitter, and $\mathcal{Y} = \{Y_t\}_{t=t_1}^{t_T}$ for Google. Each random variable evaluates to a trending score of a given trend. More formally, let $s_X(x_i, t)$ be the trending score of $x_i \in \mathcal{V}_X$, and $s_Y(y_j, t)$ be the trending score of $y_j \in \mathcal{V}_Y$, as measured at time $t \in \mathcal{T}$. The *observed time series* for $x_i \in \mathcal{V}_X$ and $y_j \in \mathcal{V}_Y$ correspond to the sequences of values assumed by each X_t and Y_t:

$$\mathcal{X}_i = \{X_t = s_X(x_i, t)\}_{t=t_1}^{t_T}, \;\; \mathcal{Y}_j = \{Y_t = s_Y(y_j, t)\}_{t=t_1}^{t_T}.$$

Moreover, we define the aggregate time series resulting from each pair of Twitter series $\mathcal{X}_i$ and $\mathcal{X}_j$, as follows:

$$\mathcal{X}_i \uplus \mathcal{X}_j = \{s_X(x_i, t) \oplus s_X(x_j, t)\}_{t=t_1}^{t_T},$$

where $\oplus$ is a normalized sum.

Trend Bipartite Graph (TBG). Analyzing *any* pair of time series derived from a Twitter trend (x_i) and a Google hot query (y_j) might be useless or even misleading. In this study, we focus on series associated with trends that are likely related, since they refer to vocabulary keywords that are somehow "similar". In addition, we aim at combining groups of series referring to Twitter trends that are alike. To this end, we introduce the *Trend Bipartite Graph* (TBG), as follows.

Definition V.1 (Trend Bipartite Graph)**.**
Let $\mathsf{TBG} = (\mathcal{V}_X, \mathcal{V}_Y, E, w, \eta)$ *be a* bipartite graph *where:*
– $\mathcal{V}_X$ *and* $\mathcal{V}_Y$*, the vocabularies of Twitter and Google trends, respectively, are the two* disjoint *sets of graph* nodes[4]*;*
– $E \subseteq \mathcal{V}_X \times \mathcal{V}_Y$ *is the set of graph* edges*;*
– $w : \mathcal{V}_X \times \mathcal{V}_Y \longmapsto [0, 1]$ *is an* edge weighting function*;*
– $\eta \in [0, 1]$ *is a weight* threshold*, such that* $E = \{(x_i, y_j) \in (\mathcal{V}_X, \mathcal{V}_Y) \mid w(x_i, y_j) \geq \eta\}$.

Intuitively, the TBG links any pair of trends from Twitter and Google with an edge weighted by a function w, which measures the pairwise *trend similarity*. The threshold η is in turn used to avoid linking those trends that are low related to each other.
Several trend similarity functions can be used. The simplest approach only looks at the lexical surfaces of trends, thereby computing a string similarity score (e.g., *Levenshtein distance*, *longest common substring*, *n-gram similarity*, etc.). More advanced solutions might take into account the *semantics* of each trend (e.g., by linking trends to referent *entities* of an external knowledge base like *Wikipedia* [29,30]). The TBG allows us to identify a set of "comparable" time series pairs, whose associated trends are similar, and which are defined by $\mathcal{S} = \{(\mathcal{X}_i, \mathcal{Y}_j) \mid (x_i, y_j) \in E\}$.

In a nutshell, for each $\mathcal{Y}_j$ we retrieve a set of related series $\mathcal{X}_i$ according to the TBG, and we aggregate them together obtaining $\mathcal{S}_{\mathcal{Y}_j} = \biguplus_{(\mathcal{X}_i, \mathcal{Y}_j) \in \mathcal{S}} \mathcal{X}_i$. We can finally define a set of comparable time series, where the first element of each pair results from the aggregation of several Twitter series:

$$\mathcal{D} = \{(\mathcal{S}_{\mathcal{Y}_j}, \mathcal{Y}_j) \mid \mathcal{S}_{\mathcal{Y}_j} \neq \emptyset\}.$$

It is worth remarking that we combine only Twitter series because it is more likely that multiple trending topics refer to the same Google hot query than vice versa. Furthermore, if $\eta = 1$ then $\mathcal{S}$ includes only those x_i that *exactly* matches y_j, namely those trends that are actually *shared* "as-is" between Twitter and Google. If this is the case, it holds that $\mathcal{D} = \mathcal{S}$.

Trend Forecasting and Causality. Our final goal is to check whether the evolution of a trend from Twitter is *significant* to explain and predict the behavior of its counterpart trend as extracted from Google.
Concretely, given a $\mathsf{TBG} = (\mathcal{V}_X, \mathcal{V}_Y, E, w, \eta)$ and the dataset of related pairs of time series $\mathcal{D}$, for each $(\mathcal{X}_i, \mathcal{Y}_j) \in \mathcal{D}$ we perform the following tasks.
Firstly, we evaluate the capability of $\mathcal{X}_i$ (Twitter) in *forecasting* $\mathcal{Y}_j$ (Google). Going further, we measure if $\mathcal{X}_i$ *causes* $\mathcal{Y}_j$ by performing a test for the *Granger causality* [31]. To achieve both tasks, we compare the three classes of regression models discussed in Section 2. In the end, we aim to show which model best fits our data, on average.

VI EXPERIMENTS

In this section, we describe the experiments we conducted on a real-world dataset of trends from Google and Twitter. The experimental phase is divided into three separate tasks:

1. Raw Data Crawling: to collect both Google and Twitter data, thereby deriving the actual time series of trends.

2. Time Series Datasets Building: to create the time series datasets from the "raw" Google and Twitter data crawled.

3. Time Series Regression Analysis: to conduct the regression analysis for exploring relation between Twitter and Google trends.

1 RAW DATA CRAWLING

In the very first step, we crawl all the data both from Google and Twitter, which are necessary for deriving the final datasets of time series. Concretely, we collect data for fifteen consecutive days, namely from 2012-11-01 at 00:00AM UTC to 2012-11-15 at 11:59PM UTC. However, since Google and Twitter have their own services and access policies for getting data, we describe this task separately.

[4]Though the same string may occur in both vocabularies, elements of those two sets are *conceptually* separated.

1.1 GOOGLE HOT QUERIES

Google has quite recently released *Google Hot Trends* as an addition to the more general *Google Trends* tool. More precisely, Google Hot Trends displays the top-20 *hot*, i.e., fastest rising, queries (search-terms) of the past hour in the United States. This is for searches that have recently experienced a sudden surge in popularity.

The actual Google trend data can be automatically retrieved by subscribing to a specific Atom feed[5], which hourly provides a ranked list of 20 trending queries.

Furthermore, Google may provide each hot query with a daily *search volume index*, which is a normalized integer score ranging from 0 to 100 computed as follows. Given a specific range of dates, i.e., $[d_{start}, d_{end}]$, and a hot query q, we denote by $svi(q, d)$ the search volume index of q on day d, such that $d_{start} \leq d \leq d_{end}$. Google assigns $svi(q, d^*) = 100$ on day d^* with the highest search volume traffic of q. For any other day $d' \neq d^*$, $svi(q, d')$ results from normalizing the search volume of q on d' with respect to d^*, and $svi(q, d^*)$.

At the end of the crawling step, we collect 24 daily lists, thereby resulting in $15 * 24 = 288$ lists, each one containing 20 hot queries. Therefore, we derive the vocabulary of hot queries $\mathcal{V}_Y$ from this raw dataset as follows. Firstly, we properly pre-process and clean each crawled hot query (e.g., by removing punctuation symbols from hot keyword searches). In addition, we considered only the top-10 queries, thereby restricting the original dataset to a total number of $2,880$ hot trends (possibly with repetitions). Finally, we remove any duplicates, thus resulting in 190 unique hot queries, meaning that $|\mathcal{V}_Y| = 190$. This vocabulary is also available for download.[6]

1.2 TWITTER DATA

Twitter allows developers to interact with its platform by exposing a useful REST Application Programming Interface (API).[7] Roughly, two main functionalities are available throughout this API: *Search* and *Streaming*.

We perform two crawling tasks running in parallel: from a side, we use the Search API to retrieve the list of top-10 trending topics, once every 5 minutes. It is worth noting that Twitter refreshes the list of trends *exactly* every 5 minutes, thereby no trend data are lost during this step. Moreover, at this stage we decide to keep the time granularity of our observations as finest as possible, even though, as we will see, our analysis is conducted on a daily basis. In fact, it might be the case that finer-grained data are useful for future work.

On the other hand, we collect a sample of the public tweets throughout the Streaming API.

Each of these APIs has its own policies to limit the rate of requests made by a client. To avoid standard rate limits, we upgrade our default access policies of both Search and Streaming APIs to *authenticated* and *gardenhose*, respectively. In this way, we can make up to 350 Search API requests per hour (instead of 150), and get our sample of Twitter timelines randomly-selected from a large collection of about 10% of the total public tweets (instead of 1%).

i) Trending Topics. This crawling task collects 12 top-10 lists of U.S. *trending topics* for each hour, thereby resulting in a total of up to $12 * 10 = 120$ hourly trends. However, to align this dataset of trends with that provided by Google, we need to "collapse" each block of top-10 lists into a combined top-10 hourly list. Since trends do not come with an absolute frequency value, we cannot extract the top-10 by counting how many times each trend occur in all the single lists of each hour. In fact, we have to consider the *ranking position* where a trend appear in each 5-minute slot, since it might happen that a high-ranked trend occurs less frequently than a low-ranked one.

More generally, this is an instance of the *rank aggregation* problem, which aims at combining several ordered lists in a proper and efficient way. Solutions to rank aggregation range from quite simple approaches (e.g., based on rank average) to more complex ones.

We use the functions provided by *RankAggreg*[8], which is an R package implementing two algorithms for rank aggregation proposed in [32], namely *Cross-Entropy Monte Carlo Algorithm* (CE) [33] and the *Genetic Algorithm* (GA) [34].

In particular, the rank aggregation is mapped to an optimization problem whose goal is to find the global ranked list such that its total distance from all the individual ordered lists is minimum. Two distance measures are included in the *RankAggreg* tool, i.e., *Spearman's footrule* and *Kendall's tau* [35], which in turn can be used by CE and GA to compute the final

[5] `http://www.google.com/trends/hottrends/atom/hourly`
[6] `http://bit.ly/YiO6AD`
[7] `https://dev.twitter.com/docs/api`
[8] `http://cran.r-project.org/web/packages/RankAggreg/`

solution. Anyway, it is worth remarking that neither CE nor GA guarantee the optimal solution. More details on this and the rank aggregation problem may be found in [32].
We combine each block of our 12 top-10 trend lists by using the CE algorithm in conjunction with the *Spearman's footrule* distance. This lead to the same overall number of crawled Google trends, that is $2,880$. From such a raw dataset, we extract the vocabulary set of Twitter trends by applying preprocessing and cleaning operations, thereby resulting in a total amount of 892 unique entries, i.e., $|\mathcal{V}_X| = 892$, which are available for download.[9]

ii) Public Timelines. The other crawling task we perform concerns the retrieval of a sample of tweets from the public timelines of Twitter. To be consistent with other collections, we focus only on tweets coming from the U.S., which are almost all written in English. As a result, we obtain a total amount of about 260 million tweets, which means that more than 17 million tweets have been crawled per day on average. It is worth remarking that these tweets are drawn from a 10% random sampling of the whole population of U.S. tweets during our period of observation. This means that the total number of U.S. tweets per day is about 170 million on average, which sounds compliant with the more recent statistics on daily rate of world-wide tweets (i.e., about 400 million [5]).

2 TIME SERIES DATASETS BUILDING

In this section, we discuss how actual time series of trends have been built from the raw datasets collected from Google and Twitter, as detailed above.

Each trend associated with $x_i \in \mathcal{V}_X$, $y_j \in \mathcal{V}_Y$ is observed during a time interval $\mathcal{T}$, which corresponds to the range of dates used to crawl our data.
However, only Google provides each hot query with a score $s_Y(y_j, t)$, namely the *daily search volume index* denoted by $svi(y_j, t)$ (see Section VI.1.1). Thus, $\mathcal{T}$ can be divided into single days, i.e., $\mathcal{T} = \{t_1, t_2, \ldots, t_{15}\}$. For each hot query y_j we obtain 15 daily observations, each one equals to the daily search volume index:

$$s_Y(y_j, t) = svi(y_j, t), \quad \text{where } t = t_1, \ldots, t_{15}.$$

Note that these are the finest-grained observations we can obtain for Google hot queries. Therefore, the resulting time series for y_j is $\mathcal{Y}_j = \{Y_t = svi(y_j, t)\}_{t=t_1}^{t_{15}}$.

Since Twitter trends do not come with any score, in order to make Google and Twitter time series comparable, we have to figure out a score also for each Twitter trend, which is similar to the Google's daily search volume index. Thereby, we exploit the whole collection of public tweets obtained as described in Section VI.1.2, and for each trend $x_i \in \mathcal{V}_X$ we compute the *daily trend volume index* $tvi(x_i, t)$, $t \in \mathcal{T}$, as follows. Let $count(x_i, t)$ be the number of occurrences of the trend x_i in the public set of tweets during the day t. Then, the daily trend volume index is:

$$tvi(x_i, t) = \left\lceil \frac{count(x_i, t)}{\max \bigcup_{t \in \mathcal{T}} count(x_i, t)} \right\rceil * 100, \quad (7)$$

where $\max \bigcup_{t \in \mathcal{T}} count(x_i, t)$ is the maximum daily count of x_i, as measured across all the days in the interval.
Like Google's search volume index, also Twitter's trend volume index is a normalized integer score ranging from 0 to 100. In this way, for each Twitter trend x_i we obtain 15 daily observations, each one equals to the daily trend volume index:

$$s_X(x_i, t) = tvi(x_i, t), \quad \text{where } t = t_1, \ldots, t_{15}.$$

Finally, the resulting time series for the Twitter trending topic x_i is $\mathcal{X}_i = \{X_t = tvi(x_i, t)\}_{t=t_1}^{t_{15}}$.

To compare Twitter's and Google's time series, we pair those derived from *similar* trends. To this end, we build the *Trend Bipartite Graph* (TBG) starting from the trend vocabularies $\mathcal{V}_X$ and $\mathcal{V}_Y$, according to Section V. We define the edge weighting function w as the string similarity between any pair of Twitter and Google trends x_i and y_j, respectively. Specifically, we use a *normalized longest common substring* score ($nlcs$), defined as follows:

$$nlcs(x_i, y_j) = \frac{|lcs(x_i, y_j)|^2}{|x_i||y_j|},$$

where $|lcs(x_i, y_j)|$ is the length of the longest string of characters that is a substring of both x_i and y_j.
Actually, we use two similarity thresholds, i.e., $\eta_1 = 1.0$ and $\eta_2 = 0.6$, thereby obtaining two graphs $\mathsf{TBG}_1 = (\mathcal{V}_X, \mathcal{V}_Y, E_1, nlcs, \eta_1)$ and $\mathsf{TBG}_2 = (\mathcal{V}_X, \mathcal{V}_Y, E_2, nlcs, \eta_2)$. We denote by $\mathcal{S}_1 = \{(\mathcal{X}_i, \mathcal{Y}_j) \mid (x_i, y_j) \in E_1\}$ and $\mathcal{S}_2 = \{(\mathcal{X}_i, \mathcal{Y}_j) \mid (x_i, y_j) \in E_2\}$ the two sets of time series pairs derived from TBG_1 and TBG_2, respectively.[10] [11] Obviously, $|\mathcal{S}_1| \leq |\mathcal{S}_2|$, since TBG_1 contains a less

[9] http://bit.ly/120h0tQ
[10] http://bit.ly/YIEsFD
[11] http://bit.ly/YrKnwA

number of edges: in particular, we find 50 pairs of trends for $\mathcal{S}_1$, and 69 for $\mathcal{S}_2$.
In addition, we aggregate and normalize the series associated with Twitter trends, which are connected to the the same Google hot query in the bipartite graphs. Thus, for the two graphs we generate two sets $\mathcal{D}_1$ and $\mathcal{D}_2$ (see Section V), where $\mathcal{D}_1 = \mathcal{S}_1$, while $\mathcal{D}_2$ contains pairs, each coupling an aggregate Twitter time series with one derived from Google.

3 TIME SERIES REGRESSION

Our experiments comprised the following steps: *(i) Test for weak stationarity*, *(ii) Cross-correlation*, and *(iii) Forecasting and Causality.*

Test for Weak Stationarity. First, we are interested in finding if time series on our dataset are stationary or not. To this end, we inspect the *autocorrelation* plots of each individual time series $\mathcal{X}_i$ and $\mathcal{Y}_j$, separately. Indeed, the autocorrelation of a non-stationary variable appears *strongly positive* and *non-noisy* out to a high number of lags (often 10 or more) meaning it is slow to decay. Conversely, the autocorrelation of a stationary variable usually decays into "noise" and/or hits negative values within a few lags.
According to this, all the time series in both $\mathcal{D}_1$ and $\mathcal{D}_2$ turn out to be stationary. This is reasonable considering that our time series all range from 0 to 100, thereby no *trending*[12] (either increasing or decreasing) nor *seasonality* could occur.
Fig. 2 shows the autocorrelation plots of the time series of the trend barack obama, as observed by Twitter and Google.

Cross-correlation. In order to figure out if any two (weakly) stationary time series are related to each other, we compute their *cross-correlation* (see Eq. 3). The cross-correlation of two time series measures the correlation between their random variables at a given time lag (i.e., *lagged relationship*).
Our trend time series are made of daily observations, thereby cross-correlation can be computed for time lags greater than or equal to one day, and no finer-grained result can be produced. As expected, coherently with such limitation, the maximum cross-correlation we find for each pair of time series $(\mathcal{X}_i, \mathcal{Y}_j)$ mostly occurs at lag 0.
This means that, on a daily basis, the series almost overlap, as depicted in Fig. 3. These plots show the cross-correlation for the same pairs of trend time series of Fig 1.
All the trends exhibit their maximum cross-correlation at lag 0, except for the trend election. There, the maximum cross-correlation is at lag -1, namely the value of the Google time series for that trend is significantly predicted by the value of the Twitter time series for the same trend, as measured one day before. Anyway, even those trends having maximum cross-correlation at lag 0 show significant values for negative lags as well, which perhaps would be more evident if a hourly analysis was possible.
Nevertheless, the considerations above on cross-correlation are compliant with our preliminary findings in Section II.

Forecasting and Causality. In the last experimental stage, we evaluate two phenomena about social and web trends: *(i) forecasting* and *(ii) causality*. The former refers to the power of Twitter trends in predicting their counterpart Google hot queries whereas the latter goes a step further and tries to determine *causality* between Twitter and Google by performing a *Granger-causality* test [31]. Both issues require dealing with the time series regression models described in Section IV.2.

To assess the first aspect, we consider each pair of time series $(\mathcal{X}_i, \mathcal{Y}_j)$ in our running datasets $\mathcal{D}_1$ and $\mathcal{D}_2$, individually. We start fitting each series to *autoregressive* models, i.e., AR(p). Intuitively, this means that we are trying to explain the behavior of a trend time series from Google $Y_{j,t}$ at day t, by only considering the values of the *same* series as measured up to p days before (i.e., $Y_{j,t-1}, \ldots, Y_{j,t-p}$). Thereby, these models assume Google trends depending *only* on themselves, and Twitter having no influence.
On the other hand, we introduce the second class of regression models, namely *distributed lag* DL(q). As opposed to AR(p), DL(q) models try to fit a Google time series from $Y_{j,t}$ at day t, by only looking at the paired time series from Twitter as measured up to q days before (i.e., $X_{j,t}, \ldots X_{j,t-q}$).
We measure how many AR(p) models retain as *significant* their p-lagged component, and, similarly, how many DL(q) models keep as *significant* their q-lagged component. Significance is determined by computing the *p-value*,[13] which is the probability of observing a test statistic at least as large as the one calculated assuming the *null hypothesis* is true. To choose whether the null hypothesis is in fact true or false, a trade-off on the the *p-value* is needed. Typically, the null hy-

[12]Here the term "*trending*" refers to a property of time series, and *not* to our trends.
[13]Here the p of *p-value* is totally unrelated to the lag order p of autoregressive models.

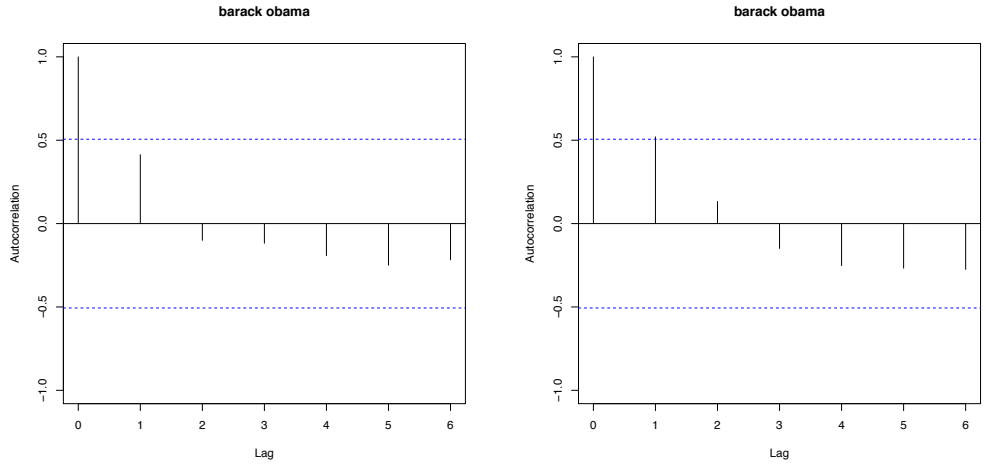

Figure 2: Autocorrelation plots of the time series for barack obama in Twitter (left) and Google (right).

	AR (%)	R^2	$\hat{R}^2$	p	DL (%)	R^2	$\hat{R}^2$	q
$\mathcal{D}_1$	10.0	0.09	0.02	1	**60.0**	0.79	**0.75**	1
	18.0	0.20	0.04	2	30.0	0.79	0.72	2
	14.0	0.29	0.03	3	16.0	**0.84**	0.71	3
$\mathcal{D}_2$	13.3	0.10	0.02	1	**56.7**	0.79	**0.75**	1
	15.0	0.19	0.03	2	25.0	0.80	0.73	2
	13.3	0.28	0.01	3	18.3	**0.84**	0.73	3

Table 1: Time series regression: AR(p) vs. DL(q).

pothesis is rejected if *p-value* is below a *significance level* α. In our experiments, we set $\alpha = 0.05$ and the null hypothesis assumes the p- or q-lagged component to be *not* significant. Thereby, rejecting the null hypothesis indicates such coefficients are useful for the model to fit the data.
For each class we limit the lag order of the model p and q up to a maximum of 3 days, since there is no evidence of cross-correlation for more than one or two days in the past.

As the last step, for each tested model we compute the *coefficient of determination*, denoted by $R^2 \in [0, 1]$ and averaged by all the pairs of time series. This is generally used to describe how well a regression line fits a dataset, and provides a measure of how well future outcomes are likely to be predicted by the model. The greater R^2 for a model the better it fits the data. In addition, we measure the *adjusted* R^2, denoted by $\hat{R}^2$, as a variation of R^2 that considers the number of explanatory terms in a model. Unlike R^2, the $\hat{R}^2$ increases only if the new term improves the model more than would be expected by chance.

In Tab. 1, we show the results for both the two datasets $\mathcal{D}_1$ and $\mathcal{D}_2$, which can be interpreted as follows. Each entry in the table measures the percentage of models AR(p) and DL(q) where the p- and q-lagged component was significant. Very unlikely, time series derived from Google hot queries can be explained by AR(p) models, namely only using their past values. Indeed, only 10% of the times the negative lag-1 component is significant. Conversely, when DL(q) models are used, the negative lag-1 component of Twitter is significant 60% of the times.

Furthermore, we check if there is any causality between Twitter and Google trends. This is achieved by running a test for *Granger-causality*, which is executed by comparing AR(p) models with *autoregressive distributed lag* models, i.e., ADL(p, q). These models use up to p and q values from *both* the dependent (i.e., Google) and explanatory (i.e., Twitter) trend time series, respectively. To test for Granger-causality we measure the ratio of ADL(p, q) models where q-lagged component of the explanatory variable is significant. Clearly, this value is less than (or at most equal to) that we find when comparing AR(p)

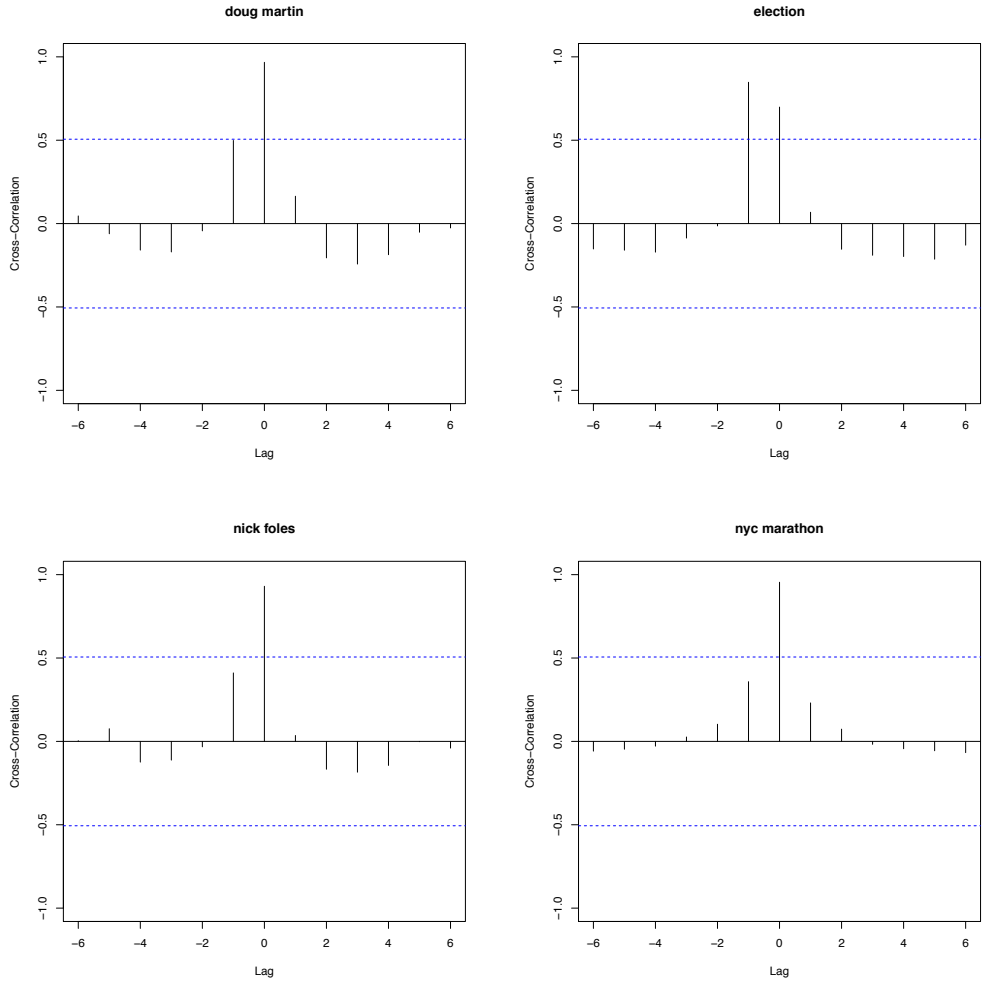

Figure 3: Cross-correlation plots of four sample pairs of trend time series.

	$\mathcal{D}_1$			$\mathcal{D}_2$		
(p,q)	ADL (%)	R^2	$\hat{R}^2$	ADL (%)	R^2	$\hat{R}^2$
$(1,1)$	**42.0**	0.84	**0.79**	**43.3**	0.85	**0.80**
$(2,2)$	18.0	0.86	0.76	16.7	0.88	0.79
$(3,3)$	18.0	**0.92**	0.76	18.3	**0.93**	0.78

Table 2: Test for Granger-causality using ADL(p,q).

with DL(q) models because we are now evaluating a relation that is "stronger" than forecasting. Tab. 2 shows that in more than 40% pairs of time series Twitter trend "Granger-causes" Google hot queries if we limit to $q = -1$. This percentage evaluates to about 70% if we restrict only to those pairs who have already shown a significant q-lagged component in the corresponding DL(q) model.

Finally, in Fig. 4 we present the $\hat{R}^2$ values obtained from each class of time series regression models: AR(p), DL(q), and ADL(p,q).
From all the results above, ADL$(1,1)$ is the model that best fit our data, on average. This result sounds reasonable because it mixes the autoregressive component of Google with the prediction of Twitter, as captured one day before.

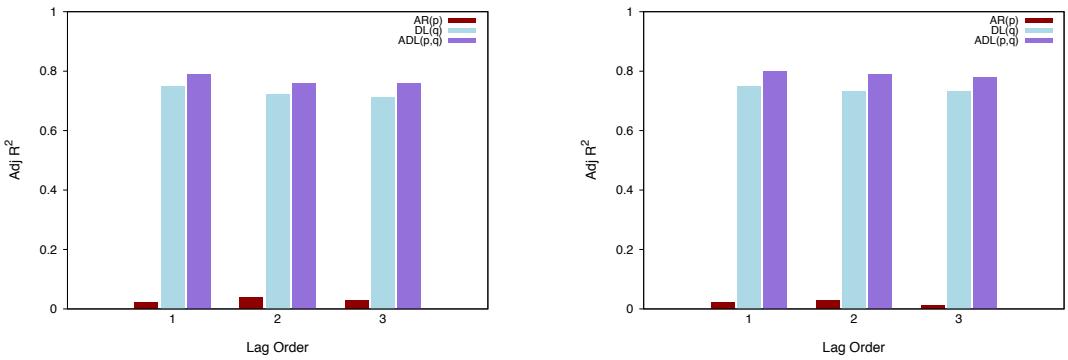

Figure 4: Evaluating time series regression models AR(p), DL(q), and ADL(p, q) on $\mathcal{D}_1$ (left) and $\mathcal{D}_2$ (right).

VII CONCLUSIONS

In this work, we explored possible relations between *trending topics* rising from Twitter (i.e., *social trends*) and *hot queries* issued to Google (i.e., *web trends*). We claimed that a trending topic on Twitter could later become a trending query on Google as well. The rationale of this is that information flooding nearly real-time across the Twitter social network could anticipate the set of topics that users will be interested in – and consequently will search for – in the near future.

To validate our claim, we provided the following contributions. First, we introduced the *Trend Bipartite Graph* (TBG) to represent the lexical similarity between any pair of social and web trends, as extracted from real-world Twitter and Google datasets. The TBG used a *threshold* to link those trends that were most likely related. Second, we evaluated the *cross-correlation* between time series derived from Twitter and Google trends, which were linked according to the TBG. Last, we measured the ability of Twitter in actually *predicting* and *causing* a Google trend to later occur by conducting an exhaustive comparison of several time series regression models. This final step showed that models that included Twitter in the regression function better fit and forecast our time series data. Specifically, we found that models, which used Google as the *dependent* variable and Twitter as the *explanatory* variable, retained as significant the past values of Twitter 60% of times. Moreover, we discovered that a Twitter trend *caused* a similar Google trend to later occur about 43% of times. In the end, we showed that the best-performing models were those using past values of *both* Twitter and Google.

We have already started exploring if similar conclusion can be drawn by analyzing other trending signals. Interestingly enough, we have shown that Twitter is also able to predict the volume of requests to Wikipedia articles, which corresponds to the set of extracted trending topics.

ACKNOWLEDGEMENTS

This work was partially supported by the National Project PON "TETRis" (no. PON01 00451).

References

[1] M. Williams, "Governments use Twitter for Emergency Alerts, Traffic Notices and More," http://bit.ly/aSS4Kk, January 2009.

[2] S. Paul, "How Social Media Is Changing Disaster Response," http://on.mash.to/Ht78io, April 2012.

[3] Z. Papacharissi and M. de Fatima Oliveira, "Affective news and networked publics: The rhythms of news storytelling on #egypt," *Journal of Communication*, vol. 62, no. 2, pp. 266–282, 2012.

[4] L. Dugan, "Twitter To Surpass 500 Million Registered Users On Wednesday," http://bit.ly/wApNwL, February 2012.

[5] D. Farber, "Twitter hits 400 million tweets per day, mostly mobile," http://cnet.co/KHlg8q, June 2012.

[6] L. Hardesty, "Predicting what topics will trend on Twitter," http://bit.ly/PIsBbr, October 2012.

[7] F. Giummolè, S. Orlando, and G. Tolomei, "Trending topics on twitter improve the prediction of google hot queries," in *Proceedings of SocialCom '13*. IEEE Computer Society, 2013.

[8] A. Java, X. Song, T. Finin, and B. Tseng, "Why we twitter: understanding microblogging usage and communities," in *Proceedings of WebKDD/SNA-KDD '07*. New York, NY, USA: ACM, 2007, pp. 56–65.

[9] B. Krishnamurthy, P. Gill, and M. Arlitt, "A few chirps about twitter," in *Proceedings of WOSN '08*. New York, NY, USA: ACM, 2008, pp. 19–24.

[10] D. Zhao and M. B. Rosson, "How and why people twitter: the role that micro-blogging plays in informal communication at work," in *Proceedings of GROUP '09*. New York, NY, USA: ACM, 2009, pp. 243–252.

[11] B. A. Huberman, D. M. Romero, and F. Wu, "Social networks that matter: Twitter under the microscope," *First Monday*, vol. 14, no. 1, 2009.

[12] B. J. Jansen, M. Zhang, K. Sobel, and A. Chowdury, "Twitter power: Tweets as electronic word of mouth," *JASIST*, vol. 60, no. 11, pp. 2169–2188, November 2009.

[13] H. Kwak, C. Lee, H. Park, and S. Moon, "What is twitter, a social network or a news media?" in *Proceedings of WWW '10*. New York, NY, USA: ACM, 2010, pp. 591–600.

[14] E. J. Ruiz, V. Hristidis, C. Castillo, A. Gionis, and A. Jaimes, "Correlating financial time series with micro-blogging activity," in *Proceedigs of WSDM '12*. New York, NY, USA: ACM, 2012, pp. 513–522.

[15] J. Weng, E.-P. Lim, J. Jiang, and Q. He, "Twitterrank: finding topic-sensitive influential twitterers," in *Proceedings of WSDM '10*. New York, NY, USA: ACM, 2010, pp. 261–270.

[16] N. Liu, J. Yan, S. Yan, W. Fan, and Z. Chen, "Web query prediction by unifying model," in *Proceedings of ICDMW '08*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 436–441.

[17] I. Bordino, S. Battiston, G. Caldarelli, M. Cristelli, A. Ukkonen, and I. Weber, "Web search queries can predict stock market volumes," *PloS One*, vol. 7, no. 7, p. e40014, 2012.

[18] M. Ettredge, J. Gerdes, and G. Karuga, "Using web-based search data to predict macroeconomic statistics," *Communications of the ACM*, vol. 48, no. 11, pp. 87–92, November 2005.

[19] H. Choi and H. Varian, "Predicting the present with google trends," *Economic Record*, vol. 88, pp. 2–9, 2012.

[20] S. Goel, J. M. Hofman, S. Lahaie, D. M. Pennock, and D. J. Watts, "Predicting consumer behavior with Web search," *Proceedings of the National Academy of Sciences*, vol. 107, no. 41, pp. 17 486–17 490, Oct 2010.

[21] J. Ginsberg, M. Mohebbi, R. Patel, L. Brammer, M. Smolinski, and L. Brilliant, "Detecting influenza epidemics using search engine query data," *Nature*, vol. 457, pp. 1012–1014, 2009.

[22] C. Corley, A. R. Mikler, K. P. Singh, and D. J. Cook, "Monitoring influenza trends through mining social media," in *Proceedings of BIOCOMP*, 2009, pp. 340–346.

[23] G. Tolomei, S. Orlando, D. Ceccarelli, and C. Lucchese, "Twitter anticipates bursts of requests for wikipedia articles," in *Proceedings of CIMK/DUBMOD '13*. New York, NY, USA: ACM, 2013.

[24] M. H. Kutner, C. J. Nachtsheim, and J. Neter, *Applied Linear Regression Models*, Fourth International ed. McGraw-Hill/Irwin, September 2004.

[25] N. Ravishanker and D. Dey, *A First Course in Linear Model Theory*, ser. Chapman&Hall/CRC Texts in Statistical Science. CRC, 2013.

[26] G. U. Yule, "Why do we Sometimes get Nonsense-Correlations between Time-Series?–A Study in Sampling and the Nature of Time-Series," *Journal of the Royal Statistical Society*, vol. 89, no. 1, pp. 1–63, January 1926.

[27] C. W. J. Granger, "Some properties of time series data and their use in econometric model specification," *Journal of Econometrics*, vol. 16, no. 1, pp. 121–130, May 1981.

[28] R. F. Engle and C. W. J. Granger, "Co-integration and error correction: Representation, estimation, and testing," *Econometrica*, vol. 55, no. 2, pp. 251–76, March 1987.

[29] R. Mihalcea and A. Csomai, "Wikify!: linking documents to encyclopedic knowledge," in *Proceedings of CIKM '07*. New York, NY, USA: ACM, 2007, pp. 233–242.

[30] C. Li, J. Weng, Q. He, Y. Yao, A. Datta, A. Sun, and B.-S. Lee, "Twiner: named entity recognition in targeted twitter stream," in *Proceedings of SIGIR '12*. New York, NY, USA: ACM, 2012, pp. 721–730.

[31] C. W. J. Granger, "Investigating causal relations by econometric models and cross-spectral methods," *Econometrica*, vol. 37, no. 3, pp. 424–438, July 1969.

[32] V. Pihur, S. Datta, and S. Datta, "Rankaggreg, an r package for weighted rank aggregation." *BMC Bioinformatics*, vol. 10, 2009.

[33] P.-T. de Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of Operations Research*, vol. 134, no. 1, pp. 19–67, 2005.

[34] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st ed. Boston, MA, USA: Addison-Wesley, 1989.

[35] R. Kumar and S. Vassilvitskii, "Generalized distances between rankings," in *Proceedings of WWW '10*. New York, NY, USA: ACM, 2010, pp. 571–580.