Final report - CS224W: Using Wikipedia pages traffic data to infer pertinence of links

- Marc Thibault; SUNetID number: 06227968
- Mael Trean; SUNetIDnumber: 06228438

I - Introduction/Motivation/Problem Definition

Intro

Wikipedia is a tremendous source of knowledge. Its main strength rely in the easiness of navigating throughout pages, domains, concepts, etc. In order to make the most out of the mass of content this website provides, it is crucial to make sure that the links between articles are relevant and adapted. Wikipedia recently launched a Kaggle competition based of its traffic data for each article, the goal of which is to conceive a prediction algorithm in order to forecast future traffic, which is not our ambition. Nevertheless, this data can be of first utility when evaluating the closeness of two articles. Indeed, we believe that pages with a similar traffic pattern most likely are visited by the same people at the same time, ergo vehiculating proximity information.

Data exploration

Time series data Our dataset consists of two datasources. One is the features space we want to explore, that is the traffic time series for each article of Wikipedia. This data can be accessed by interfacing with an API, where one can query the traffic time series of a given article over a given time window, down to a given precision (the thinner precision being hourly).

However, in order to prototype our project as quickly as possible, we used as a working basis the data made available in the Kaggle competition "Web Traffic Time Series Forecasting" (`https://www.kaggle.com/c/web-traffic-time-series-forecasting`). It only spans over a limited selection of articles (circa 10,000 articles), and is limited to daily traffic recordings, from January 2015 to December 2016. Exploring some of these time series yields the following figure:

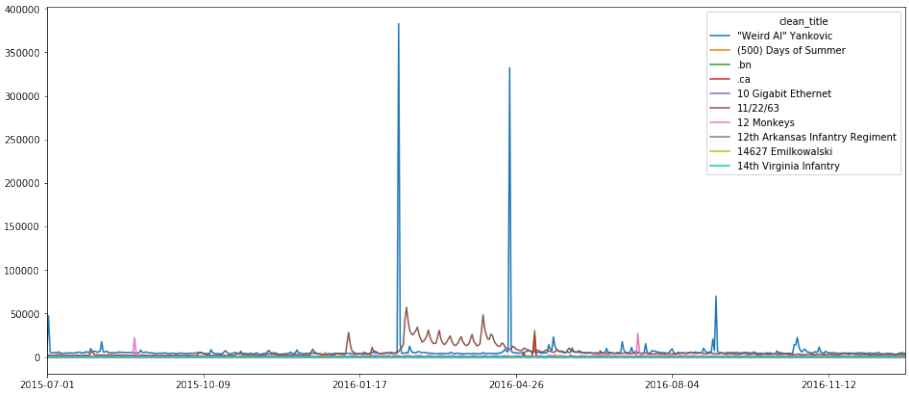

Figure 1: Random times series samples

The first few main observations are the following:

- The overall scale (trend and variance) of the time series depend a lot on the considered article;
- The considered time series are highly spiky: traffic on a given page really follows moments of surge and long calm and few-frequented periods;

These two observations led us to develop metrics to compute the alignment of such time series, considering to what extend these spikes are simultaneous.

The Wikipedia Graph Our graph data comes from Jure Leskovec's category-based pruning of the main Wikipedia graph. It comprises of the WCC of the articles of the biggest categories of Wikipedia. We chose to use this network as a basis, and to rely on its information to validate our time-series based approach.

The first task we had to achieve was matching the time series recordings from Kaggle to nodes in the graph. This was done by matching page names and domains to the node information provided by Jure Leskovec's dataset.

Overall, restricting our analysis to articles present both as nodes in this pruned version of the Wikipedia graph, and as time series recordings, we end up with 6,400 nodes and 28,086 edges.

II - Related Work

We began by studying the following paper:

- Kim, M. and Leskovec, J., 2011, April. The network completion problem: Inferring missing nodes and edges in networks. In *Proceedings of the 2011*

SIAM International Conference on Data Mining (pp. 47-58). Society for Industrial and Applied Mathematics.

The main point of this article is to complete missing data in the case of networks. To do so, they cast the problem in the Expectation Maximization (EM) framework with a particular graph model. The model is fitted with the graph structure; then the missing data is inferred from the model's parameters, which are then again fitted to the graph, and so on and so forth.

They voluntarily use only the graph structure, choosing not to use external data to refine their prediction of missing data. The following paper tackles this issue by using browsing history data to infer missing links.

- West, R., Paranjape, A. and Leskovec, J., 2015, May. Mining missing hyperlinks from human navigation traces: A case study of Wikipedia. In *Proceedings of the 24th international conference on World Wide Web* (pp. 1242-1252). Conferences Steering Committee.

The main idea of this paper is to find relevant metrics to infer missing links on the Wikipedia graph. The authors used actual browsing data from Wikipedia-based games to find candidate missing links, filtered out existing links, and defined metrics (relatedness and path frequency) to rank and find the most plausible non-existing links.

This article relies a lot on a small sample of browsing paths. This measurement relies on human interaction, which yields cost and bias, making it hardly scalable and reproducible for large networks.

- Clauset, A., Moore, C. and Newman, M.E., 2008. Hierarchical structure and the prediction of missing links in networks. *arXiv preprint arXiv:0811.0484*.

This article, as the two previous ones, tries to predict missing links in networks. It relies on the network structure of the underlying hierarchical model, built from nodes label representing their categories. This method is thus using both network structure and extraneous labelization.

However, this model does not predict well links between categories, making it a rare event. The Wikipedia network is especially useful because it allows the user to navigate easily between distant yet related concepts. We will thus use another source of extraneous data to try and reproduce these links.

Our main idea is to use Wikipedia traffic data as a core feature to evaluate the structural properties of graphs. The main asset of this method is that it represents a new insight into the data: indeed, traffic patterns are external to the graph structure and can represent ground truth for a relatedness between articles.

III - Model/Algorithm/Method

Designing closeness metrics Our main challenge was to feature engineer the time series we have, in order to extract significant and relevant events. Indeed, we need to quantify the occurrence of a surge in a page's traffic data.

We built the following three closeness metrics to quantify proximity between two time series:

- `Distance Normalized` consists in simply computing the euclidean distance between traffic time series, each being normalized by its standard deviation.

$$d_{normalized}(i,j) = ||\frac{S_i(t)}{std(S_i)} - \frac{S_j(t)}{std(S_j)}||_2$$

- `Distance Simplified` computes the distance between simplified time series. Simplified time series contain 1s or 0s, depending on whether the daily traffic above the 90^{th} percentile.

$$d_{simplified}(i,j) = ||1_{S_i(t)>q_{90\%}(S_i)} - 1_{S_j(t)>q_{90\%}(S_j)}||_2$$

- `Distance Surge` is meant to have a more dynamic approach of detecting a surge. In order to have a thinner understanding of surges, we need to compare every recording of the time series to a "normal" regime. We subtracted the time series to its rolling mean (window size being 10 days). We chose to only keep the positive values of these residuals in order to extract the days where traffic is significantly above average, as being quantified as above 3 times standard deviation.

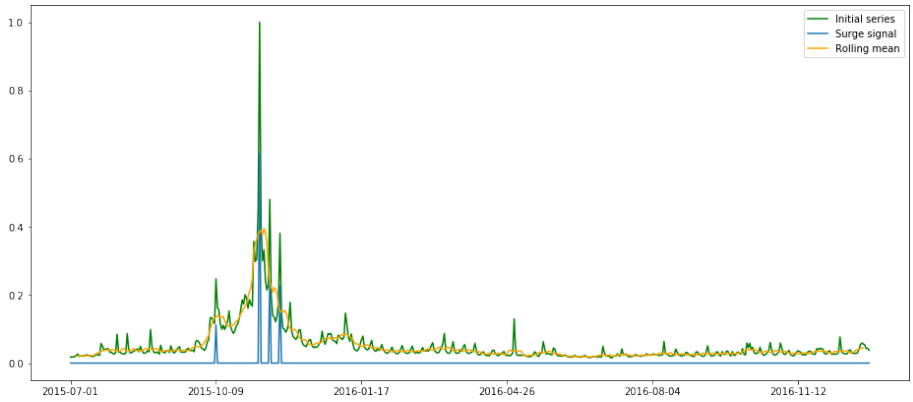

Figure 2: Construction of the surge series

This extraction process yields a new time series we called the surge series. It consists of a time series with, for each day, a value between 0 and 1, quantifying the presence and the intensity of a surge in traffic data.

The last step is computing the distance between two such surge series for two Wikipedia pages.

$$d_{surge}(i,j) = ||S_{i,surge} - S_{j,surge}||_2$$

Selecting nodes

- Our initial graph comes from the network dataset built by Leskovec and co. in `https://snap.stanford.edu/data/wikispeedia.html`. The main idea is to consider articles which belong in the most represented categories in wikipedia. Within this graph, we only keep the nodes which belong to the biggest SCC (Strongly Connected Component).
- The second step is matching the nodes from the Wikipedia graph to the Time Series data obtained on Kaggle. This subset, after preprocessing, represents ~7000 time series.
- The third step of filtering out nodes from our analysis comes from the observation that, for many pages, no significant surge can be seen in traffic data. Given our previous quantification of a surge coming from the surge series, we only kept the nodes which have at least 2 surges during the studied period. This final step leaves us with 3890 nodes, that is 15 million node pairs to study.

Learning edge presence Our methodology here will be the following:

- First test that our traffic-based distances actually make sense when compared to the underlying network structure;
- Then aggregate these results into a single prediction of existence of an edge.

We want to make sure that article pairs with a link in the Wikipedia graph usual have a smaller distance than those which do not have an edge. This would validate our approach, and be the grounds for building a classifier of edge occurrence, based only on traffic data.

The idea behind it is that two articles which are close to each other for all the metrics can be considered to be related in our paradigm. Each distance delivers insights which can uncover different aspects of the time series. Consequently, we will need to combine our various distance metrics in order to produce a classifier which takes into account several concepts of closeness.

Our classification problem will have the following formulation:

$$y_{i,j} = 1_{\{edge(i,j)\}} \sim \left(d_{normalized}(i,j), d_{simplified}(i,j), d_{surge}(i,j)\right)$$

In order to aggregate these metrics, we will apply the following Machine Learning algorithm to classify the presence of an edge :

- Naive Bayes with Gaussian distribution
- Random Forests

Quantifying prediction results We will study the accuracy matrices, comparing our prediction with the actual ground truth of edge existence, both on train and test data (representing respectively 70 and 30% of our pairs of edges.)

One main issue with this classification task is the huge class unbalance between the node pairs with an edge, and the node pairs without an edge. Indeed, the Wikipedia graph adjacency matrix is extremely sparse: 15,000 out of 15,000,000 of node pairs have an edge, that is .1%.

In order to take this unbalance into account in evaluating the quality of our precision, we will take the F-score as a final target metric on the test set.

IV - Results and findings

Evaluating distances relevance We computed, for each pair of nodes, our 3 metrics of distance between traffic time. We now want to make sure that these distances behave differently, depending on the existence of an edge or not between the Wikipedia articles.

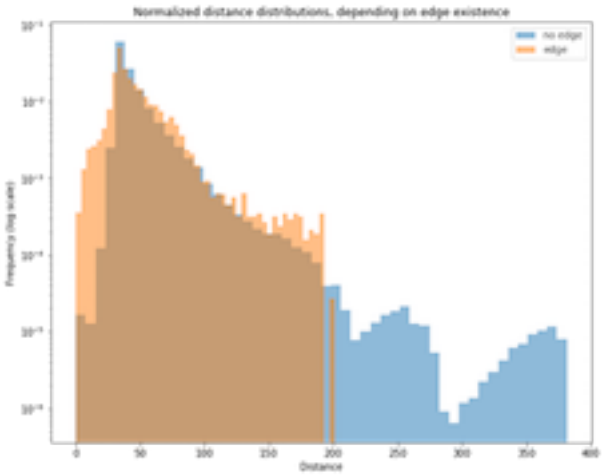

Figure 3: Distributions of normalized distance

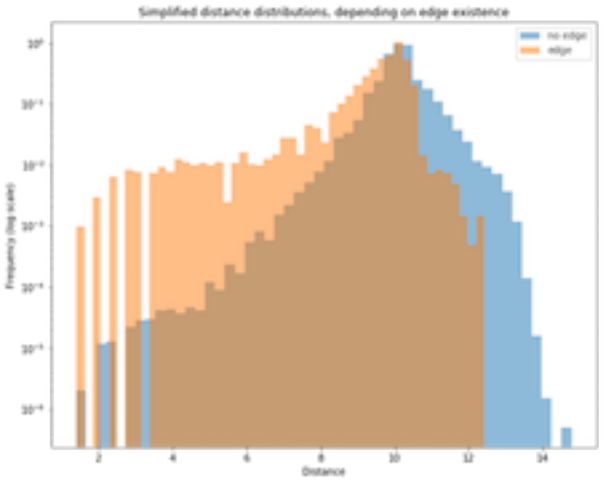

Figure 4: Distributions of simplified distance

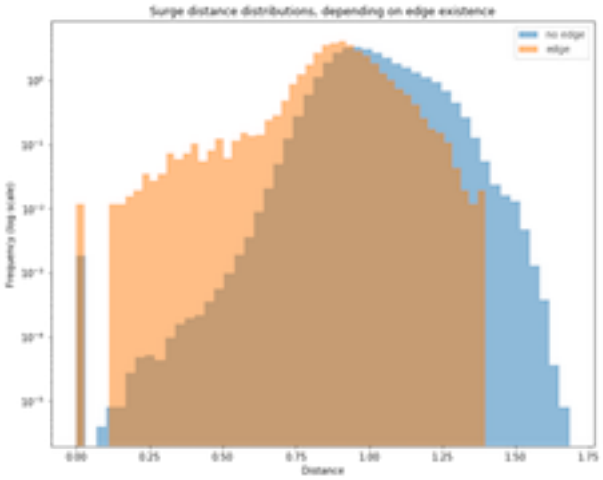

Figure 5: Distributions of surge distance

We now see that, for these three hand-made metrics based on traffic data, we observe our initial hypothesis: edge existence drives lower distances distributions. This can be especially seen for the normalized distance, for which there is a clear separation, meaning that a distance bigger than 200 can not represent an edge.

We also see that, for each of the considered distance, the lower distances tend to be more present when there is an edge. However, each distance taken separately is not sufficient to predict edge existence, because:

- first, there is no clear separation between the distance distributions, depending on edge existence;
- on areas where pairs with an edge are more present than pairs with no edge, the difference is not significant enough to beat the huge class unbalance we witnessed beforehand.

Learning results Here are the results of our learning procedures, on test data (out of sample):

- Naive Bayes with Gaussian distribution:

  Accuracy matrix

Prediction	$y_{true} = 0$	$y_{true} = 1$
$y_{pred} = 0$	7447798	8642
$y_{pred} = 1$	33201	579

  F-score: 2.76%

We see that the Naive Bayes approach does not yield interesting results. Indeed, most of the edges we predicted actually do not exist; less than 2% of these are relevant. This leads to a really bad F-score.

- Random Forests

  Accuracy matrix

Prediction	$y_{true} = 0$	$y_{true} = 1$
$y_{pred} = 0$	7481056	5880
$y_{pred} = 1$	113	3341

  F-score: 52.72%

These results are better than the previous ones. Indeed, the edges we predict are, for the most part, existing (97%). Furthermore, we still manage to correctly classify more than 35% of the existing edges.

The ones we are not detecting (the 5880 $y_{true} = 1$ *and* $y_{predict} = 0$) can lack surge coherence, on the period we used to sample traffic data.

Uncovering non-existing edges We now reach the most crucial part of our project: using time series traffic data to suggest new links in the Wikipedia graph. Using the Random Forest classifier, we have 113 false positives (pairs we predict as linked, which are not).

We can formulate the hypothesis that our prediction is relevant, and that these pairs of articles are related. Here is a list of a few of these relevant pairs:

- “Santa Clause” - “The Twelve Days of Christmas (song)”
- “N.W.A” - “Easy-E”
- “Defenders (comics)” - “Iron Fist (comics)”
- “Magneto (comics)” - “Jean Grey”
- “Princess Margaret, Countess of Snowdon” - “Princess Alice of Battenberg”
- “Venezuala” - “Cold War”
- “South Africa” - “Australia”
- “List of films based on Marvel comics” - “Avengers (comics)”

We see that most of these pairs are relevant and indicate articles with a strongly related content. Even if links between those articles might not be explicitly needed, this pattern could be used as a recommendation engine for further reading.

We also realized that this method comes short when articles lack coordination in the surge structure. However, such a behavior is highly unlikely when considering longer time windows for the samples.

In order to leverage this initial analysis for actual industrial application, we can use thinner traffic analysis, with hourly data. These raw distance metrics come short for precise edge prediction, but they might be especially useful when combined with more classical features to evaluate the proximity between concepts and people.