Latent distance graphs from news data

Luka Bizjak
Artificial Intelligence Laboratory
Jozef Stefan Institute
Ljubljana, Slovenia
l.bizjak@student.fmf.uni-lj.si

Miha Torkar
Jozef Stefan International Postgraduate School
Artificial Intelligence Laboratory
Jozef Stefan Institute
Ljubljana, Slovenia
miha.torkar@ijs.si

Aljaž Košmerlj
Artificial Intelligence Laboratory
Jozef Stefan Institute
Ljubljana, Slovenia
aljaz.kosmerlj@ijs.si

ABSTRACT

Network analysis is one of the main topics in modern data analysis, since it enables us to reason about systems by studying their inner relations, for example we can study a network by analyzing its edges. However, in many cases it is impossible to detect or measure the network directly, due to noisy data for example. We present a method for dealing with such systems, more concretely we present a probabilistic model called latent distance network, which we use to model news data from **EventRegistry**. In the end of the article we also present experimental results on predictions of latent distance model with methods of machine learning.

1. INTRODUCTION

News articles offer a constant stream of new information about business activities, political events, natural disasters and a variety of other topics. Finding structure in such data is inherently difficult, because of the high levels of noise, repetitions and lack of structure. In order to systematically extract useful information from news, [6] developed a system called **EventRegistry**. It is able to collect news articles from various sources and languages, group them together according to their content and then extract relevant information from the texts about the grouped articles (entities, people, locations, topics). The groups of articles about the same content are called events and in this work we will explore the structure of connections between various topics of such events. Namely we will build a network of connections between topics that appear in the news events, track the evolution of connections through time and predict future relationships.

This will be done through the framework of latent distance graphs, because each event will be placed in an embedded space according to its content. The distance to other events in this space will then represent the similarity between them. Latent distance graphs are an example of metric random graphs, where the probability of connection between two nodes is dependant on their position in the space. These types of graphs are also called network models, which represent multi-dimensional data. In our case the transformation of the events from EventRegistry will yield vectors in a 300 dimensional Euclidean space that are then used for calculation of a distance function that determines probability of connection between nodes. The procedure to extract a network of connections from the events data can be generalized, because one can apply it in any case where the distance between objects is well defined.

The rest of the paper is organised as follows, section 2 describes the news data that we were using. In section 3 we introduce the word embeddings that are used to obtain vector forms of news events. In section 4 we explain the latent distance model. section 5 presents the analysis done on the graphs and section 6 concludes by pointing out the main results, stressing some difficulties of our approach and giving directions for further work.

2. DATA

We have worked with the news data from the EventRegistry and in particular, we download all of the news events from business category in years 2017 and 2018. This yields theevents overall and it was obtained with python package **EventRegistry**. Every event that was obtained contains information about which entities were involved and **EventRegistry** also provides classification of events into certain categories as defined in the *dmoz* taxonomy. These categories are structured hierarchically and are divided into various subcategories, where, for example, category "Business" is split further into "Banking and services" and then into "Investing". For further details see [6].

3. WORD EMBEDDINGS

Here we give a brief overview of how the word (word2vec) embeddings are being calculated, for detailed explanation we advise reader to consult [9]. One of the algorithms which can calculate the word embeddings is called word2vec and it can be trained by one of the sub-algorithms, called Continuous-Bag-Of-Words (CBOW) and Skip-Gram (SG). We give more precise explanation of the Skip-Gram algorithm since we use

it in the process of generating the latent model.

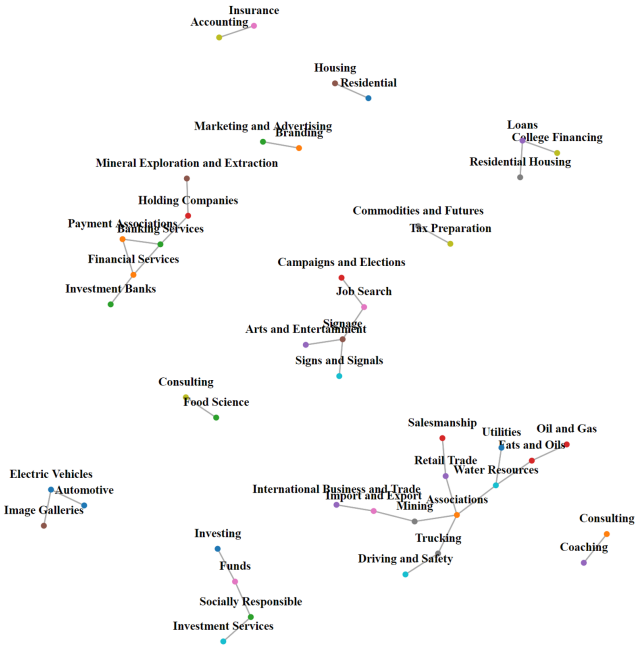

Figure 1: Latent distance graph of the Business category

3.1 Skip-Gram

The Skip-Gram (SG) sub-algorithm trains a shallow neural network (NN) to learn the vector representation of each word in the sentence from its surrounding context words. Namely for each word w_n, the neural network predicts the surrounding context words Con, where the user has to define the number of surrounding words that are being predicted. For example if $Con = 2$, we predict the pair (w_{n-1}, w_{n+1}). The NN that is being trained has only one hidden layer between the input and output layer, which in practice gives two transformations of the input vector. The first one from the input to the hidden layer and the second one from the hidden layer to the output. Hence we have the target word at the input layer of the NN and the context words are at the output layer.

In order to give more detailed overview of the algorithm we need to set some notation first. Let x be a word in vocabulary and let N be the size of the vocabulary, so $x_1, ..., x_N$ are all the words in the vocabulary. Then let K be the dimension of the hidden layer and $C = Con$ be the number of context words. Also denote by W the $N \times K$ weight (transformation) matrix. The equation for hidden layer is in this case then

$$\mathbf{h} = W^T x = W^T_{k,\cdot}, \tag{1}$$

where $W_{k,\cdot}$ is the k^{th} row of the transformation matrix W. From the hidden layer to the output layer we apply another transformation matrix that is denoted by $W^{'}$. The output layer in this case then consists of C multinomial distributions and we use the matrix $W^{'}$ to calculate the score vector $\mathbf{u}$ for the j^{th} unit on c^{th} context word as follows:

$$u_{c,j} = u_j = (W^{'}_{\cdot,j})^T \mathbf{h} = (W^{'}_{\cdot,j})^T W^T_{\cdot,k}, \tag{2}$$

for all $c = 1, 2, .., C$. The final output is the given by:

$$\mathbb{P}(w_{c,j} = w_{out,c} | w_I) = y_{c,j} = \frac{exp(u_{c,j})}{\sum_{i=1}^{N} exp(u_i)}, \tag{3}$$

here $w_{c,j}$ is the j^{th} word on the c^{th} panel of the output layer and $w_{out,c}$ is the true c^{th} word among the output context words, thus $y_{c,j}$ is the final output of the j^{th} unit on the c^{th} panel. The authors of both CBOW and SG [8] have noted that SG is slower than CBOW but produces better results for infrequent words. For more detailed exposition please see [9].

3.2 Events to vectors

In order to obtain vector forms of events, we use all of concepts and their weights that are extracted from the news article of every event by EventRegisty. In particular the vector form of each event is calculated as

$$\text{Event}_i \equiv e_i = \sum_{c \in C_i} w(c) \cdot \text{word2vec}(c), \quad \forall e_i \in E, \tag{4}$$

where C_i represents the set of all concepts of the event i and $w(\cdot)$ is gives a value of the weight of the concept in the event in the in the interval $(0, 100]$. Each event is then represented as a numeric vector in $\mathbb{R}^N$, where N is dependant on the dimension of the word2vec space. In our case we choose $N = 300$, since we are using the pre-trained word2vec model by Google.

4. LATENT DISTANCE NETWORK MODEL

Latent distance networks or graphs can be considered as particular examples of random graphs ([2]). Random graph consists of the set V of vertices or nodes, set E of edges. It is normally denoted by $G(n, p)$, where $n = card(V)$ and p is the probability of an edges between two vertices. One of the most basic examples of random graphs is the Erdos-Renyi graph $G_{ER}(n, p)$, which is defined by saying that each edge is included in the graph with probability p independent from every other edge. Having this example in mind, we can define random graphs with n vertices by giving probability distribution for edges, this is construction carries over to latent distance graphs, where probabilities for edges are derived from probability distribution of distances between vertices.

The latent distance graph is represented by $N \times N$ adjacency matrix. The latent distance model is given as follows, we first define a distance function on $\mathbb{R}^N$ by:

$$d(x_k, x_{k'}) = \rho e^{-\frac{\|x_k - x_{k'}\|^2}{\tau}}, \tag{5}$$

where ρ represents the sparsity of the network and τ represents characteristic distance scale. Each vertex of the network is at this stage represented by some vector $x_k \in \mathbb{R}^N$ (as describe in 3.2). To get a random model we need to specify some probability distribution of distances between vertices, which will give us the probabilities for edges between vertices. These then correspond to the adjacency matrix of our network and in the case of the latent distance graph it is given by:

$$A_{k,k'} \sim Bern(d(x_k, x_{k'})). \tag{6}$$

Thus the entries of the adjacency matrix A are given by Bernoulli distribution of distances between vertices of the

graph. Such networks were also considered in [7] in connection with Hawkes processes defined on networks. Note that one could choose any other appropriately normalized metric i.e. taking values in $[0,1]$, on $\mathbb{R}^N$ and the construction would work as well. Moreover in Erdos-Renyi graph $G(n,p)$ the sharp threshold for the connectedness is given by $\frac{ln(n)}{n}$ (see [4]). To the best of our knowledge no such sharp threshold is know for the latent distance graphs.

4.1 Generating the latent model

In this section we describe how we generated the latent distance graph from news data.

4.1.1 *News data latent model*

After we obtain the numeric forms of the events via the procedures described in section 3 we are able to form a latent model on top of news data. The embedded events now correspond to vectors in $\mathbb{R}^N$, which gives us a finite set of vectors $(x_i)_{i=1}^K \subset \mathbb{R}^N$. We can then construct the weight matrix, which is basically the matrix of distances between different pairs of points,

$$W = (w)_{i,j}^K = (d(x_i, x_j))_{i,j}^K, \tag{7}$$

where $d(.,.)$ is the distance function (5). Once we have the weight matrix we can generate the latent distance graph by defining matrix of probabilities as the adjacency matrix:

$$p_{i,j} = Bern(W_{i,j}), \tag{8}$$

where $p_{i,j}$ represents the probability between two nodes x_i and x_j.

5. GRAPH ANALYSIS

We perform some basic analysis on latent distance graphs derived from news data. We generated the adjacency matrices for each day in a one year time period, thus we get 365 graphs. Each represents the activity of the events from the business category. We also perform clustering of the events into 100 most frequent subcategories interacting with business category. We perform this so that we replace the adjacency matrix which depends on number of events we consider, say K, with a matrix depending on fixed number of parameters and make a aggregated distance matrix W_{agg}, which we define as follows:

$$(W_{agg})_{k,l} = \sum_{i,j}^{K} \mathbf{1}_{c_k=i, c_l=1} w_{i,j}. \tag{9}$$

Now we can generate a new graph whose nodes now represent categories under consideration. Example of such a graph is given in Figure 1. Then we can generate adjacency matrices of these graphs for a fixed time period, one year in our case.

5.1 Graph evolution

Now that we have sequence of graphs $\{G_1, G_2, .., G_{365}\}$, we can view this sequence as evolution of the network in given time frame. Thus we can check the interaction of category i in $\{G_1, G_2, .., G_{365}\}$ by just summing over all the adjacency matrices and looking into appropriate row. For example in the Table 1 below we show interaction of subcategories **Banking and services** and **Oil and Gas**. To be concise we only show top five interacting subcategories.

We also computed the degree of the the nodes in $\{G_1, G_2, .., G_{365}\}$ and plotted the time series of each node, see Figure 2. From Figure 2 we can see that there is some change in the degrees through time and this opens up a possible direction to study such networks dynamically.

Dependencies between categories					
Fixed category	Cat1	Cat2	Cat3	Cat4	Cat5
Banking and services	Holding Companies	Financial Services	Finance	Payment Associations	Investment Banks
Oil and Gas	Fats and Oils	Mining and Drilling	Import and Export	Payment Associations	Job Sharing

Table 1: Dependencies of categories in the dynamical network

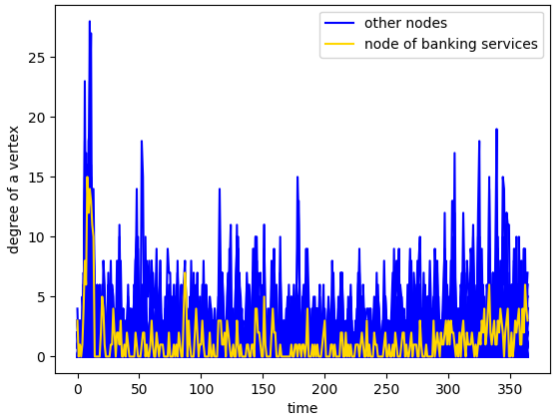

Figure 2: Degrees of nodes through time

5.2 Predictions

We use neural network model to make predictions about potential structure of the network in the near future. Specifically we build the model on the top level categories i.e. **Buisness**, **Politics** and others. We use the aggregation process (9) to generate all the adjacency matrices for all levels, which can then be put into vector form, so that we can then use them as inputs for LSTM Neural Network model [5]. For the model we used LSTM Neural Network, where we used three residual LSTM layers and final dense layer. We optimized with mean squared error (MSE) with ADAM optimizer [5]. The results of the experiments are displayed in Figure 3 and Figure 4.

6. CONCLUSIONS AND FUTURE WORK

In this work we collect data from EventRegistry about all business events from years 2017 and 2018 and build a latent distance model on top of it. We are able to do this by transforming the textual news data to numeric vector forms through word embedding algorithm word2vec. The latent graph model that is produced in this manner gives us a reasonably good representation of EventRegistry data as

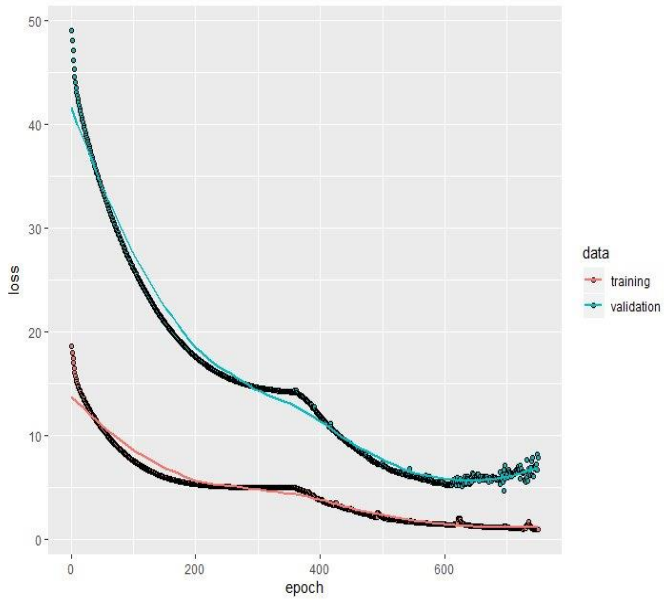

Figure 3: MSE of learning on whole Neural Network

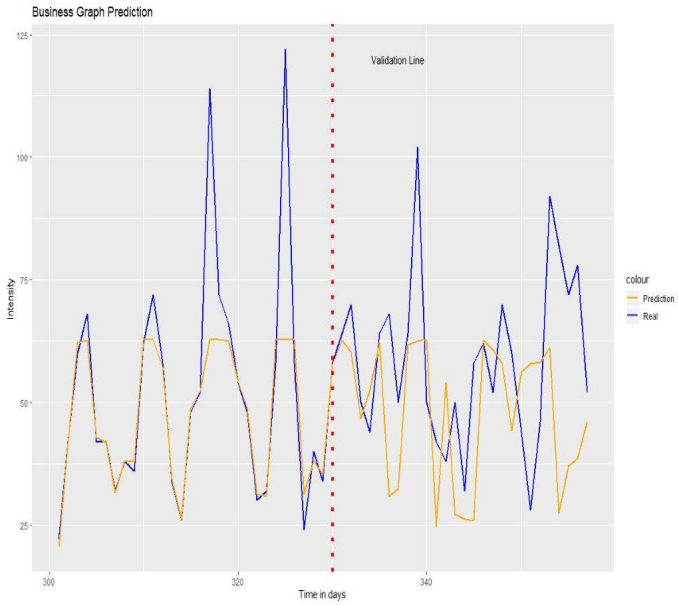

Figure 4: Training and prediction scores for Business category

can be seen in Figure 1. However we believe that this model could be used on other similar data sets as long as difference between object can be defined with some metric. In order to avoid issues with noise we used a more compact representation of events, where we clustered them into a predefined set of categories. In this compact form the adjacency matrices of graphs were then fed into a LSTM model for prediction of how the graph will evolve in the next step. The results of this part were reasonably good and can be seen in the Figure 3 and Figure 4.

Let us now point out some difficulties of this approach. The first one is that it seems that the representation depends to some extent on word embeddings that are used. This can be seen from example in Table 1, where we have strong connection between **Oil and Gas** category and subcategory **Fats and Oils** which should not be connected. The second problem is the sparsity of the adjacency matrices, which makes it very hard to achieve good performance with machine learning techniques as well perform spectral analysis [1] on the adjacency matrices. This last point is connected to theory of dynamical graphs, where our presented sequence $\{G_1, G_2, .., G_{365}\}$ serves as one example. In future work we would like to try to extend the LSTM model from above to predict lower level categories as well. This would be done in several steps where each level would be predicted after the previous one. Finally we would like to understand how to resolve the sparsity problem (some techniques for dealing with this problem are presented in [3]) and then apply techniques from dynamical graphs [1], in particular we would like to know spectral distortions [1] of these graphs.

7. ACKNOWLEDGMENTS

The research project leading to these results received funding from the European Union Research and Innovation programme Horizon 2020 under grant agreement No. 675044 (BigDataFinance). We also wish to thank Jakob Jelenčič for his help on implementation of prediction models and M. Besher Massri for the help with the graph visualization.

8. REFERENCES

[1] L. C. Aleardi, S. Salihoglu, G. Singh, and M. Ovsjanikov. Spectral measures of distortion for change detection in dynamic graphs. In *International Conference on Complex Networks and their Applications*, pages 54–66. Springer, 2018.

[2] B. Bollobás and B. Béla. *Random graphs*. Number 73. Cambridge university press, 2001.

[3] E. J. Candès. Mathematics of sparsity (and a few other things). In *Proceedings of the International Congress of Mathematicians, Seoul, South Korea*, volume 123. Citeseer, 2014.

[4] P. Erdős and A. Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1):17–60, 1960.

[5] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

[6] G. Leban, B. Fortuna, J. Brank, and M. Grobelnik. Event registry: learning about world events from news. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 107–110. ACM, 2014.

[7] S. Linderman and R. Adams. Discovering latent network structure in point process data. In *International Conference on Machine Learning*, pages 1413–1421, 2014.

[8] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[9] X. Rong. word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738*, 2014.