Latent Distance Graphs from News Data

Luka Bizjak, Miha Torkar, Aljaž Košmerlj

Jožef Stefan Insitute
Artificial Intelligence Laboratory

7.10.2019

Introduction

- News data offers a constant stream of new information about business activities, political events, natural disasters and many other topics
- Finding "relevant" structures in such data
- Fuzzy nature of data $\rightarrow$ difficulties
- Building models that overcome some of the difficulties
- Making predictions from this data
- For example we may want to monitor through news data which types of industries are doing well

Data

- We used data from the EventRegistry
- News events from the business category from 2017 to 2018
- To obtain the events one uses the python package EventRegistry
- Classification of events into certain categories
- Hierarchical structure of these categories

Example

"Business" $\rightarrow$ "Banking and services" $\rightarrow$ "Investing"

Events to vectors

- For later purposes we want to associate vector to each event
- We construct word embeddings via Google's pretrained word2vec model
- Events now correspond to vectors in 300-dimensional space
- We use all concepts and weights that are extracted from the news article of every event by EventRegistry

Formula:

$$Event_i = \sum_{c \in C_i} w(c) \cdot word2vec(c).$$

- C_i the set of all concepts of the event i
- $w(\cdot)$ gives the value of the weight of the concept in the event

Latent distance network model

Latent distance model is a random graph model, constructed via $N \times N$ adjacency matrix. We first define a distance function on $\mathbf{R}^N$ by:

$$d(x_i, x_j) = \rho e^{-\frac{\|x_i - x_j\|^2}{\tau}}.$$

- ρ represents the sparsity of the network
- τ represents characteristic distance scale

Each vertex of the network is represented by some vector $x_i \in \mathbf{R}^N$ and probabilities for edges are given by:

$$A_{i,j} \sim Bern(d(x_i, x_j)).$$

News data latent model

- We have numeric forms of data
- We can now use the above model to represent this data
- Embedded events correspond to vectors in $\mathbf{R}^{300}$
- We can easily construct the adjacency matrix, we let $W_{i,j} = (d(x_i, x_j))_{i,j}$, so the adjacency matrix is given by $A_{i,j} \equiv Bern(W_{i,j})$
- We perform aggregation procedure on W to get a matrix depending on fixed number of parameters. This is crucial, if we want to study the model evolving in time.

Aggregation formula:

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

Business category graph

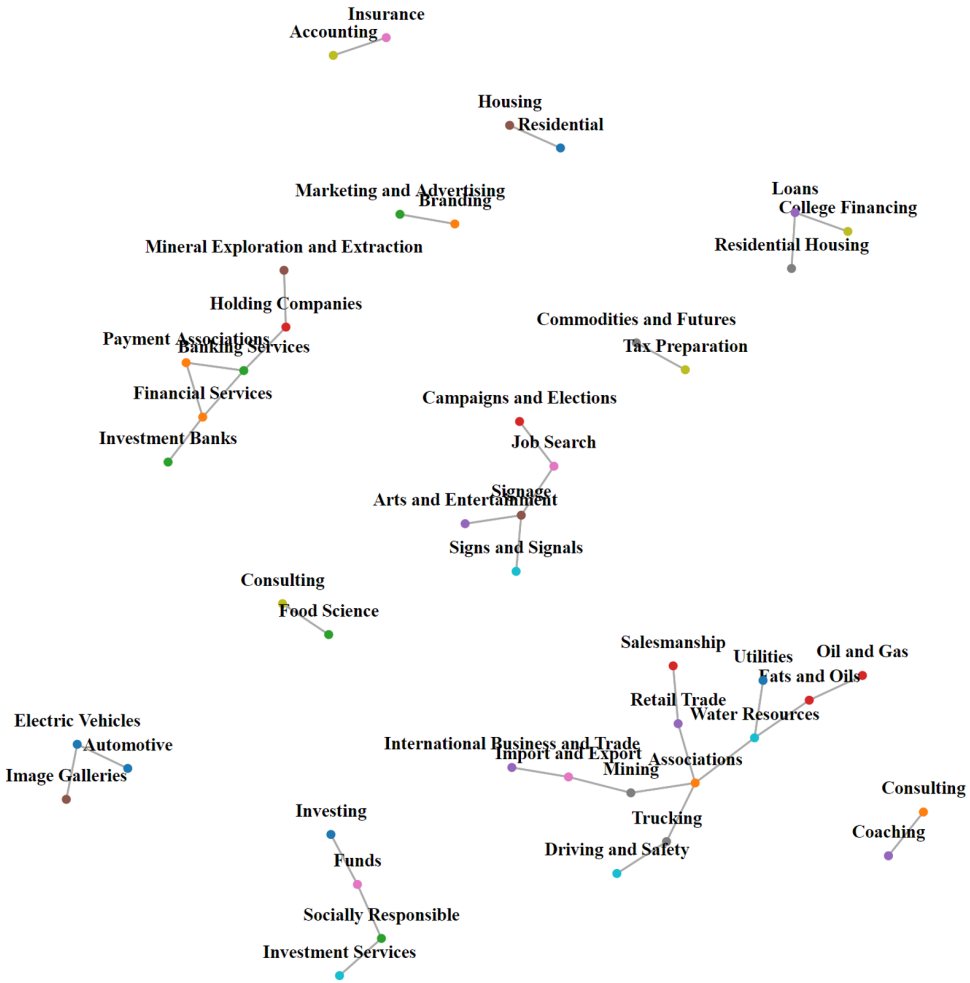

Figure 1: Latent distance graph of the business category

Graph analysis

- We generate the adjacency matrices for each day in one year time period
- We use the aggregated adjacency matrices (100 most frequent categories)
- We get a sequence of graphs $\{G_1, ..., G_{365}\}$
- With this we can check interactions between some categories
- Evolution of the degrees of nodes

Example of interactions

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

Evolution of nodes

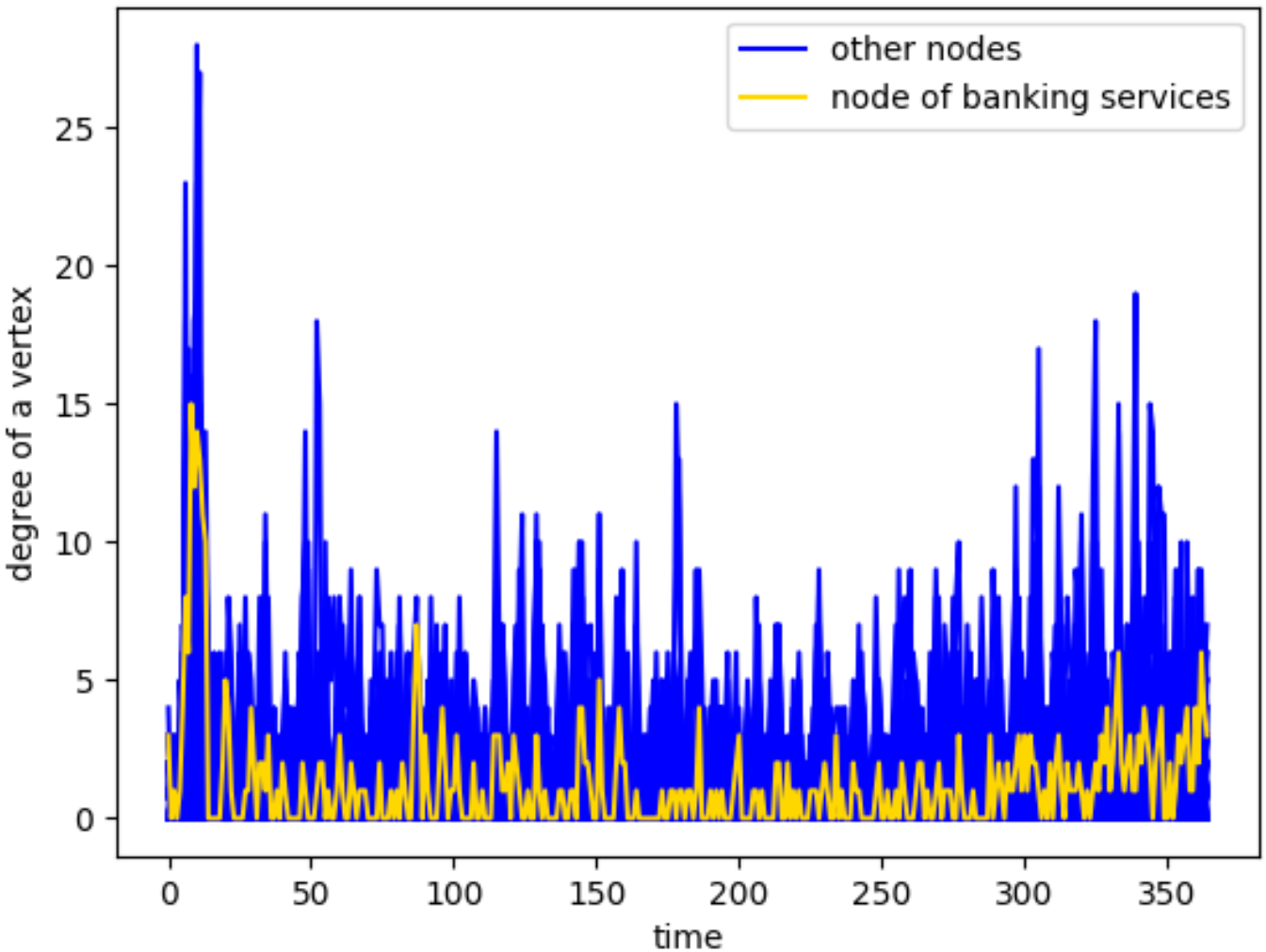

Figure 2: Degrees of nodes through time

Predictions

- We use neural networks to make predictions about potential structure of the network
- Model is built on top level categories i.e. Buisness, politics, etc.
- We are using LSTM Neural Network, where we used three residual LSTM layers and final dense layer
- We optimize MSE with ADAM optimizer

Model

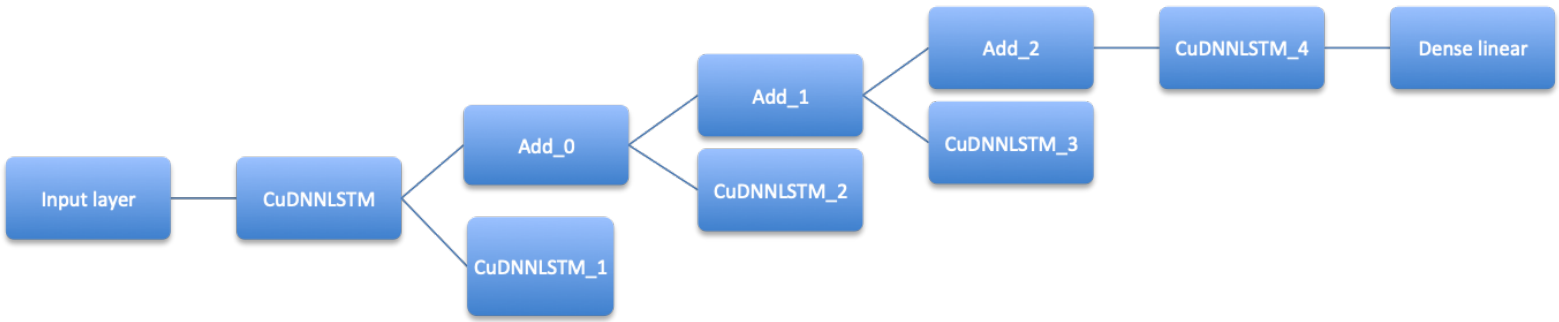

Figure 3: LSTM Neural Network model

Results

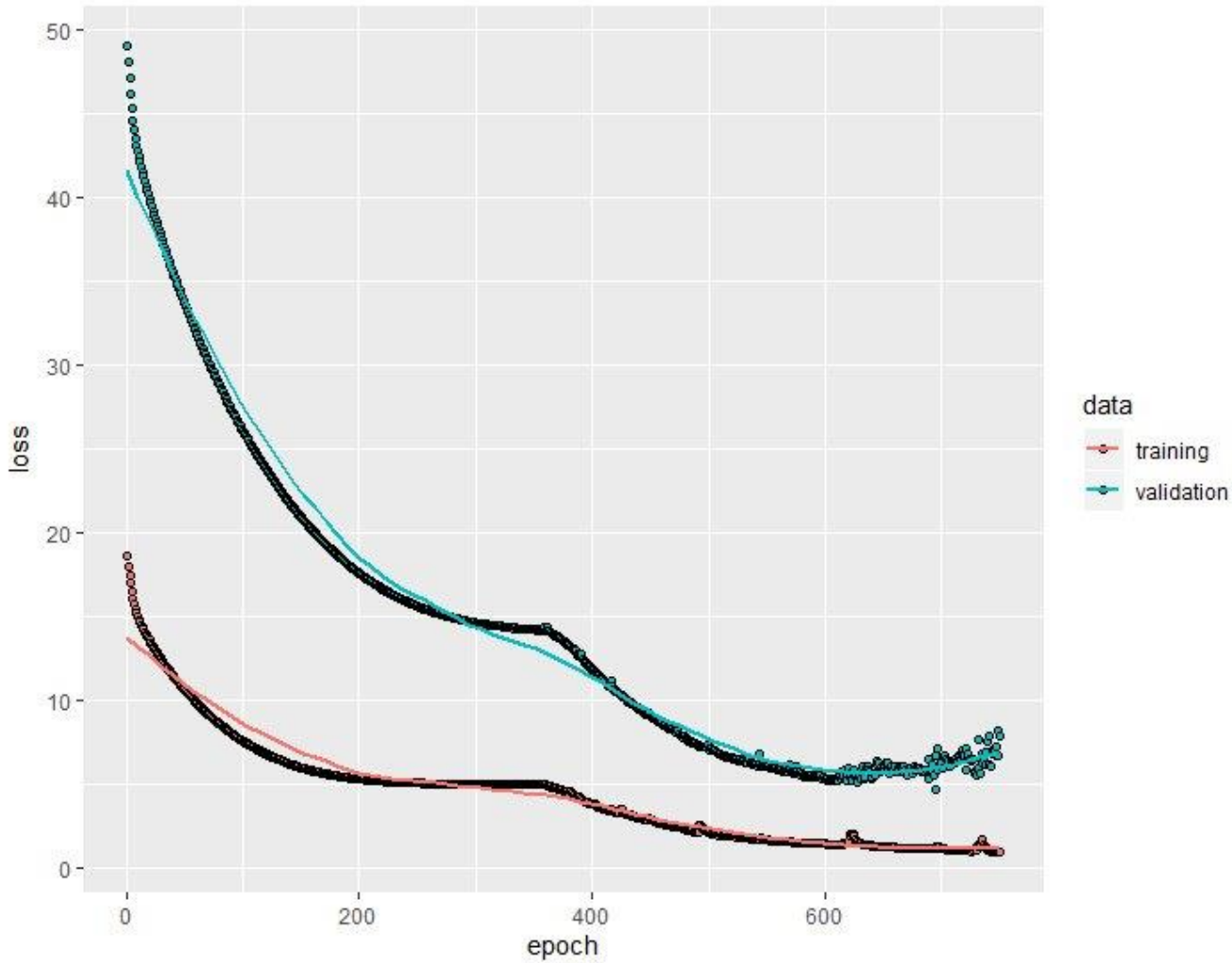

Figure 4: MSE of learning on whole NN

Results

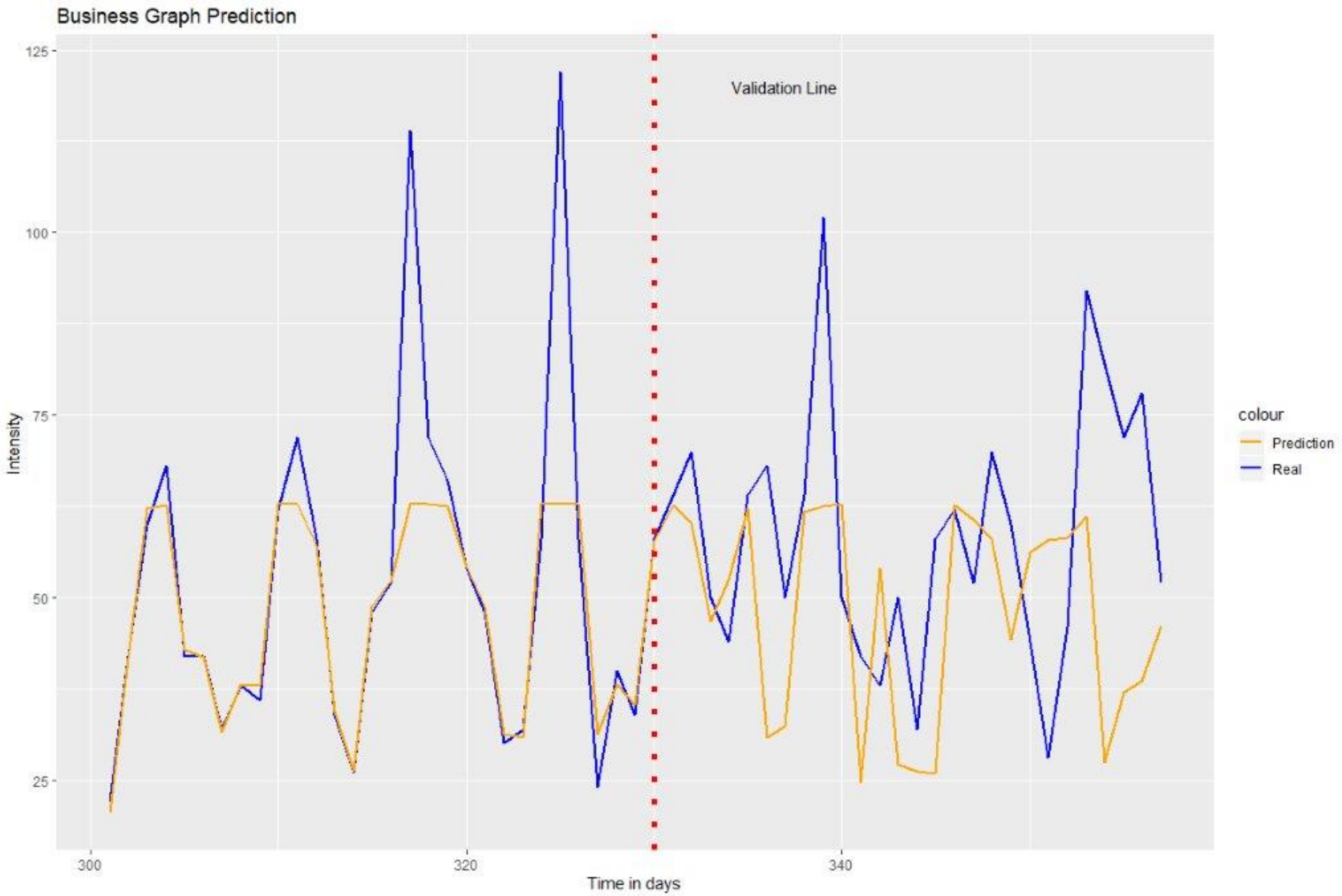

Figure 5: Training and prediction scores for Business category

Conclusion

- In this work we collected data from EventRegistry and built a latent distance model on top of it
- We get a reasonably good representation of EventRegistry data
- Latent model could be used in other cases as well
- Problem of noise
- Dependency on word embedding
- Sparsity of the adjacency matrices

Thank you