Graph Embeddings in Practice: A Telco Churn Prediction Use Case

PhD Researcher: Sandra Mitrović

Supervisor: Prof. Dr. Jochen De Weerdt

Department of Decision Sciences and Information Management, KU Leuven

Graph Embedding Day, Lyon
07 Sept 2018
Background

Classification task

• Churn prediction (CP)
  o Predicting the probability of a customer to stop using company’s services
  o Considered as the topmost challenge for Telcos [FCC report, 2009]
    • Despite not being novel
    • Given that acquisition costs are 5-10x higher than retention costs
      [Rosenberg et al, 1984]
What networks have to do with CP?

• Many different data sources and approaches used

• Recently, most frequently:
  o Data source: **Usage data**
    • **Call Detail Records (CDRs)**
    • w OR w/o: Socio-demographic, Subscription, Ordering, Call center (complaints), Invoicing…
  o Approach: **Social Network Analysis (SNA)**

• **CDRs -> call graphs**
  o Customer -> node
  o Call -> edge
  o Intensity of relationship -> edge weight

• **Graph featurization**

• **Better predictive performance** [Dasgupta et al., 2008; Richter et al., 2010; Backiel et al., 2016]
Call graph featurization

Extracting informative features from (call) graphs

• An intricate process, due to:
  o Complex structure / different types of information
    • Topology-based (structural)
    • Interaction-based (as part of customer behavior)
      • Edge weights quantifying customer behavior
  o Dynamic aspect
    • Call graph are time-evolving
    • Both nodes and edges volatile
      • Churn = lack of activity
Shortcomings of current related work

Not many studies account for **dynamic aspects of call networks** [Dasgupta et al, 2008; Richter et al, 2010; Kusuma et al, 2013; Huang et al, 2015; Backiel et al, 2016]

- Especially not **jointly with interaction and structural features**
  - Structural features are under-exploited [Phadke, 2013; Backiel et al, 2016]
  - Due to high computational time in large graphs (e.g. betweenness centrality) [Zhu, 2011]
- And **without using ad-hoc handcrafted features**
  - No featurization methodology [*]
  - Dataset dependent [*]
Our goal

• Performing “holistic” featurization of call graphs
  • Incorporating both interaction and structural information
  • Avoiding/reducing feature handcrafting
  • While also capturing the dynamic aspect of the network
Our goal

• Performing “holistic” featurization of call graphs
  • Incorporating both interaction and structural information
  • Avoiding/reducing feature handcrafting
  • While also capturing the dynamic aspect of the network
Integrating interaction and structural information

Interactions

• RFM (Recency-Frequency-Monetary) model [Hughes, 1994]
  • Standard for quantifying customer behavior/interactions (w.r.t. target event)
  • Many different variants found in literature
  • RFM operationalizations (our work):
    • Summary RFM ($RFM_s$) – total
    • Detailed RFM ($RFM_d$) – direction & destination sliced: $X_{out\_h}, X_{out\_o}, X_{in}, X \in \{R,F,M\}$
    • Churn RFM ($RFM_{ch}$) – only w.r.t. churners
RFM-Augmented networks

- Original topology extended
  - By introducing artificial nodes based on RFM
  - Structural information partially preserved

- Each of R, F, M partitioned into 5 quintiles
  - One artificial node assigned to each quintile
  - Interaction info embedded through extended topology

RFM features

- $RFM_s$
- $RFM_s \parallel RFM_{ch}$
- $RFM_d$
- $RFM_d \parallel RFM_{ch}$

Network topology

- 4 augmented networks
  - $AG_s$
  - $AG_{s+ch}$
  - $AG_d$
  - $AG_{d+ch}$
Our goal

• Performing “holistic” featurization of call graphs
  • Incorporating both interaction and structural information
  • Avoiding/reducing feature handcrafting
  • While also capturing the dynamic aspect of the network
RL: Node2vec -> scalable node2vec

Node2vec
- Accounts both for previous and current node
- Additional parameters ($p,q$)
- To make walks efficient, requires precomputation of probability transitions:
  - On node level (1st time)
  - On edge level (successive)
  - Alias sampling used for efficient sampling
    - reduces $O(n)$ to $O(1)$

However, does not scale well on large graphs!
(our case ~ 40M edges)

Scalable node2vec
- Accounts only for current node
- No additional parameters
- Requires precomputation of probability transitions only on node level
  - Alias sampling retained

Therefore, scales well even on large graphs!
Our goal

• Performing “holistic” featurization of call graphs
  • Incorporating both interaction and structural information
  • Avoiding/reducing feature handcrafting
  • While also capturing the dynamic aspect of the network
Dynamic graphs

Different definitions (current literature)
• \( G = (V, E, T) \)
• \( G = (V, E, T, \Delta T) \)
• \( G = (V, E, T, \sigma, \Delta T) \)

Standard approach
• Consider several static snapshots of a dynamic graph

Our setting
• Monthly call graph \( G = (V, E) \) ->
  Four temporal graphs \( G_i = (V_i, E_i, w_i), i = 1, \ldots, 4 \)
Methodology – Graphical overview
Experimental Evaluation

Research questions

• RQ1: Do features taking into account dynamic aspects perform better than static ones?

• RQ2: Do RFM-augmented network constructions improve predictive performance?

• RQ3: Does the granularity of interaction information (summary, summary + churn, detailed, detailed + churn) influence the predictive performance?

Experiments

- \( RFM_s \text{ stat. vs. } RFM_s \text{ dyn. vs. } AG_s \text{ stat. vs. } AG_s \text{ dyn. } \rightarrow \text{summary} \)
- \( RFM_{s+ch} \text{ stat. vs. } RFM_{s+ch} \text{ dyn. vs. } AG_{s+ch} \text{ stat. vs. } AG_{s+ch} \text{ dyn. } \rightarrow \text{summary+churn} \)
- \( RFM_d \text{ stat. vs. } RFM_d \text{ dyn. vs. } AG_d \text{ stat. vs. } AG_d \text{ dyn. } \rightarrow \text{detailed} \)
- \( RFM_{d+ch} \text{ stat. vs. } RFM_{d+ch} \text{ dyn. vs. } AG_{d+ch} \text{ stat. vs. } AG_{d+ch} \text{ dyn. } \rightarrow \text{detailed+churn} \)
Experimental results (1/2)

Prepaid

<table>
<thead>
<tr>
<th>RFM</th>
<th>Static</th>
<th>Dynamic</th>
<th>Augmented network</th>
<th>Static</th>
<th>Dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AUC</td>
<td>Lift</td>
<td>AUC</td>
<td>Lift</td>
<td>AUC</td>
</tr>
<tr>
<td>$RFM_s$</td>
<td>0.671</td>
<td>1.788</td>
<td>0.680</td>
<td>2.025</td>
<td></td>
</tr>
<tr>
<td>$RFM_s+ch$</td>
<td>0.671</td>
<td>1.789</td>
<td>0.689</td>
<td>2.014</td>
<td></td>
</tr>
<tr>
<td>$RFM_d$</td>
<td>0.683</td>
<td>1.857</td>
<td>0.692</td>
<td>2.063</td>
<td></td>
</tr>
<tr>
<td>$RFM_d+ch$</td>
<td>0.682</td>
<td>1.856</td>
<td>0.695</td>
<td>2.040</td>
<td></td>
</tr>
</tbody>
</table>

- RQ1 Answer: Dynamic better than static!
- RQ2 Answer: RFM-augmented networks improve predictive performance
- RQ3 Answer: Best performing interaction granularity is: summary+churn
  - Second best: detailed+churn
Experimental results (2/2)

Postpaid

<table>
<thead>
<tr>
<th>RFM</th>
<th>Static</th>
<th>Dynamic</th>
<th>Augmented network</th>
<th>Static</th>
<th>Dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AUC</td>
<td>Lift</td>
<td>AUC</td>
<td>Lift</td>
<td>AUC</td>
</tr>
<tr>
<td>$RFM_s$</td>
<td>0.741</td>
<td>3.367</td>
<td>0.743</td>
<td>3.403</td>
<td>$AG_s$</td>
</tr>
<tr>
<td>$RFM_{s+ch}$</td>
<td>0.741</td>
<td>3.369</td>
<td>0.758</td>
<td>3.858</td>
<td>$AG_{s+ch}$</td>
</tr>
<tr>
<td>$RFM_d$</td>
<td>0.750</td>
<td>3.750</td>
<td>0.757</td>
<td>3.874</td>
<td>$AG_d$</td>
</tr>
<tr>
<td>$RFM_{d+ch}$</td>
<td>0.750</td>
<td>3.751</td>
<td>0.767</td>
<td>3.885</td>
<td>$AG_{d+ch}$</td>
</tr>
</tbody>
</table>

- RQ1 Answer: Dynamic better than static!
- RQ2 Answer: RFM-augmented networks improve predictive performance
- RQ3 Answer: Best performing interaction granularity is summary+churn
  - Second best: summary
Shortcomings of current related work

• **Call graphs are mostly considered to be static** [Dasgupta et al, 2008; Richter et al, 2010; Kusuma et al, 2013; Huang et al, 2015; Backiel et al, 2016]
  
  - Despite: node/edge creation/deletion, node attributes/edge weights changes
  - Static approach has smoothing-out effect on customers’ behavioral changes, hindering the valuable behavioral shifts leading to churn event

• **Very few works explicitly address dynamic aspect**
  
  - **Time-series -based** [Lee et al, 2011; Chen et al, 2012; Zhu et al, 2013]
  - **Dynamic network –based (DN-based)**

    DN = a series of static networks defined over non-overlapping time-intervals
  
  • **Using ad-hoc hand-engineered features** [Hill et al, 2006; Saravanan et al, 2012]
    
    - No featurization methodology
    - Featurization effort propagates through a sequence of static networks
    - Interaction and structural features underexploited
  
  • **No discern of difference between behavior** in different time intervals [Hill et al, 2006; Saravanan et al, 2012]
Methodology

• We propose *sliding-window* approach
  • Overlapping intervals
  • As contrast to a single (static) and non-overlapping intervals

• We propose considering two different network types:
  • Shifted networks
  • Difference networks

• Applying RL on these networks
Networks considered

• Shifted networks
  • Given original graph $G = (V, E)$ for the observed time period $T$ and set of intervals $\{[t_i, t_{i+1})\}_{i=1, \ldots, n}$, s.t. $t_i < t_{i+1} < t_i + l$, where $l$ is interval length
  • Shifted network $S_i = (V_i, E_i)$ corresponds to time interval $[t_i, t_{i+1})$
    • **Unweighted** shifted network $S^{u}_i$ (all edges equally weighted)
    • **Weighted** shifted network $S^{w}_i$
      (cum. weights of the original edges vs. artificial edges = 50:50)

• Difference networks
  • Build upon shifted networks
  • Idea: delineate differences at network level by detecting bidirectional (+/-) changes in customer activity for consecutive time intervals
  • Comparing the presence of edges and their corresponding weights (in case of a weighted graph)
Derivation of difference networks (1/2)

Original network (UW) / Unweighted artificial (UWA)

- Given shifted networks $S_i = (V_i, E_i)$ and $S_j = (V_j, E_j)$ where $t_i < t_j$:
  - Decreased difference network
    \[
    D^-_{ij} = (V^-_{ij}, E^-_{ij}) \quad E^-_{ij} = \{ e \text{ with weight } w^i_e, \text{ if } e \in E_i \setminus E_j \} \cup \\
    \{ e \text{ with weight } |w^j_e - w^i_e|, \text{ if } e \in E_i \cap E_j \text{ and } w^j_e - w^i_e < 0 \}
    \]
  - Increased difference network
    \[
    D^+_{ij} = (V^+_{ij}, E^+_{ij}) \quad E^+_{ij} = \{ e \text{ with weight } w^i_e, \text{ if } e \in E_j \setminus E_i \} \cup \\
    \{ e \text{ with weight } w^j_e - w^i_e, \text{ if } e \in E_i \cap E_j \text{ and } w^j_e - w^i_e > 0 \}
    \]
Derivation of difference networks (2/2)

Weighted network (W)

- First: consider artificial edges as unweighted in order to detect differences in edges (previous case)

- Next: for the remaining ones we perform weights scaling to maintain the ratio between cumulative weights (original edges vs. artificial edges) be 50:50.
Experimental Evaluation

Setting:
- Two datasets – one prepaid, one postpaid
- Nine overlapping time intervals considered
- Stacked representations input to l2-regularized logistic regression
- Evaluation in terms of AUC & lift

Goal:
- Compare predictive performance of different representations obtained on various time periods (and corresponding networks)

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r_e(v)$</td>
<td>Node $v$ repr. obtained on the entire-period network</td>
</tr>
<tr>
<td>$r_{q_i}(v)$</td>
<td>Node $v$ repr. obtained on quarter-of-period network $w_i$</td>
</tr>
<tr>
<td>$r_{s_i}(v)$</td>
<td>Node $v$ repr. obtained on shifted network $S_i$</td>
</tr>
<tr>
<td>$\Delta r_{s_{ij}}(v)$</td>
<td>Vector difference of node $v$ repr. obtained on two consecutive shifted networks $S_i$ and $S_j$</td>
</tr>
<tr>
<td>$r_{i_{ij}}^+(v)$</td>
<td>Node $v$ repr. obtained on increase difference network $D_{ij}^+$</td>
</tr>
<tr>
<td>$r_{i_{ij}}^-(v)$</td>
<td>Node $v$ repr. obtained on decrease difference network $D_{ij}^-$</td>
</tr>
</tbody>
</table>
Experimental Results

- Adding shifted and difference network–based representations to static and the one based on non-overlapping intervals improves AUC

AUC_\text{W} > AUC_{\text{UW/UWA}}

Except for $r_e \parallel r_s^*$ for postpaid
Experimental Results

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$r_e$</th>
<th>$r_{q*}$</th>
<th>Type</th>
<th>Shifted</th>
<th>Delta</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$r_{s*}$</td>
<td>$r_{s*}</td>
<td></td>
</tr>
<tr>
<td>Prepaid</td>
<td>0.68000</td>
<td>0.69978</td>
<td>W</td>
<td>0.68010 (1.90333)</td>
<td>0.69374 (2.08470)</td>
<td>0.70149 (2.28820)</td>
</tr>
<tr>
<td></td>
<td>(1.97600)</td>
<td>(2.36861)</td>
<td>UW</td>
<td>0.68108 (1.92785)</td>
<td>0.69414 (2.07769)</td>
<td>0.70187 (2.29345)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UWA</td>
<td>0.75039 (3.83143)</td>
<td>0.76559 (3.89982)</td>
<td>0.76072 (3.81120)</td>
</tr>
<tr>
<td>Postpaid</td>
<td>0.76000</td>
<td>0.76488</td>
<td>W</td>
<td>0.76346 (3.92656)</td>
<td>0.77437 (3.82203)</td>
<td>0.76714 (3.94654)</td>
</tr>
<tr>
<td></td>
<td>(3.55300)</td>
<td>(4.10355)</td>
<td>UW</td>
<td>0.76729 (3.95400)</td>
<td>0.77539 (3.83143)</td>
<td>0.76559 (3.89982)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>UWA</td>
<td>0.75976 (3.89091)</td>
<td>0.77029 (3.81337)</td>
<td>0.76695 (3.92318)</td>
</tr>
</tbody>
</table>

- **Comparing $r_{e}$, $r_{q*}$, $r_{s*}$, $r_{d*}^{+/-}$ (in terms of AUC):**
  - $r_{q*}$ outperforms others except for postpaid unweighted ($r_{s*}$)
  - Weighted: $r_{e}$ performs the worst
  - Unweighted: $r_{d*}^{+/-}$ performs the worst

- **Comparing shifted and difference (in terms of AUC):**
  - Weighted: $r_{d*}^{+/-}$ outperforms $r_{s*}$
  - Unweighted: $r_{s*}$ outperforms $r_{d*}^{+/-}$
  - Combining $r_{s*}$ and $r_{d*}^{+/-}$ with $r_{e}$, $r_{q*}$ results become dataset-dependent
Additional analysis

- $r_{s1} \parallel r_{d^*/-}$

The results improved, but still could not win $r_{s^*}$ for unweighted
Conclusion

• We designed **RFM-augmentations** of original graphs
  o Enable conjoining interaction and structural information
• We devise a **scalable** adaption of the original node2vec approach
  o Relaxing random walk generation and avoiding grid search tuning for two additional parameters
• We attempt to take into account dynamic aspect of the networks
  o We propose applying **representation learning on top of:**
    • Networks obtained from non-overlapping intervals
    • Shifted networks (overlapping intervals)
    • Difference networks
  to **explicitly capture changes** in customer behavior.
• We demonstrate that compared to only static, non-overlapping intervals-based dynamic representations perform better and **adding shifted/difference network representations** results in even better performance improvements.
Future research

• Experiment with more sophisticated methods for assessing dynamic differences in customer behavior

• Analyzing the effect of applying temporal random walks

• Investigating how different approaches which involve shifting temporal aspect into the RL part affect predictive performance
References

• FCC, 2009. 13th Annual report and analysis of competitive market conditions with respect to mobile wireless, including commercial mobile services, Federal Communication Commission, WT Docket 10-133.


• Mikolov et al., 2013. Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119).

• Perozzi et al., 2014. Deepwalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 701-710). ACM.


Bibliography

Thank you!

Questions?

Email: sandra.mitrovic@kuleuven.be