Time-dependent Word Embeddings

Nikolai Stulov   Anna Shalova
Andrey Demidov   Mariya Kuzmina

Skolkovo Institute of Science and Technology

December 2018

Skoltech
Skolkovo Institute of Science and Technology

Outline

Introduction

- Word embeddings can be reduced to computing matrix factorization
- In large collections, it is interesting to explore word dynamics, e.g. semantic shift. Requires geometry, this is where word embeddings come to help
- A natural idea is to compute embeddings at each time step. However,
  1. Computing matrix factorization for each time is costly
  2. Interpretable embeddings require proper alignment
- Different approaches, including word count-based, topic modeling, co-occurrence and PMI, matrix factorization, well-known neural networks

Ridge regression for dynamic embeddings [2]

- Formulate optimization problem

$$\min_{U(1),\dots,U(T)} \frac{1}{2}\sum_{t=1}^{T} ||Y(t) - U(t)U(t)^\top||_F^2 + \\ + \frac{\lambda}{2}\sum_{t=1}^{T} ||U(t)||_F^2 + \frac{\tau}{2}\sum_{t=2}^{T} ||U(t-1) - U(t)||_F^2$$

  where $\lambda > 0$ and $\tau > 0$ are regularization coefficients, and $||A||_F = \sqrt{\sum_{i,j} A_{ij}^2}$ is Frobenius norm.
- It decomposes in time!
- Still quartic in $U(t)$

Skoltech
Skolkovo Institute of Science and Technology

Ridge regression for dynamic embeddings [2]

- Relaxed optimization problem for one time step $0 < t < T$

$$\min_{U(t),W(t)} \frac{1}{2}||Y(t) - U(t)W(t)^T||_F^2 + \frac{\gamma}{2}||U(t) - W(t)||_F^2 +$$
$$+\frac{\tau}{2}\left(||U(t-1) - U(t)||_F^2 + ||W(t-1) - W(t)||_F^2\right) +$$
$$+\frac{\lambda}{2}\left(\|U(t)\|_F^2 + \|W(t)\|_F^2\right)$$

- It is a regression problem! Easy to see when setting gradient to zero
- Solution is given by $U(t)A = B$, where

$$A = W^T(t)W(t) + (\gamma + \lambda + 2\tau)I$$
$$B = Y(t)W(t) + \gamma W(t) + \tau\left(U(t-1) + U(t+1)\right)$$

Projector-splitting integrator for dynamic embeddings [1]

$$\|\dot{Y}(t) - \dot{A}(t)\|_F = \min$$

$$Y(t) = U(t)S(t)V(t)^\top$$

$$\text{subject to } Y_0 = U_0 S_0 V_0^\top,$$

$$U(t)^\top \dot{U}(t) = 0, \ V(t)^\top \dot{V}(t) = 0, \ rank(Y) = r$$

$$\begin{cases} \dot{U}(t) = & \left(I - U(t)U(t)^\top\right) \dot{A}(t)V(t)S(t)^{-1} \\ \dot{V}(t) = & \left(I - V(t)V(t)^\top\right) \dot{A}(t)^\top U(t)S(t)^{-1} \\ \dot{S}(t) = & U(t)^\top \dot{A}(t)V(t) \end{cases}$$

Projector-splitting integrator for dynamic embeddings [1]

$$\dot{Y}(t) = P(Y(t))\dot{A}(t)$$

$$P(Y)Z = ZVV^\top - UU^\top ZVV^\top + ZVV^\top$$

$$UU^\top = P_{R(Y)},\ VV^\top = P_{R(Y^\top)}$$

which leads to the splitting method for $t \in [t_0, t_1]$:

- $\dot{Y}_I = \dot{A}P_{R(Y_I^\top)},\ Y_I(t_0) = Y_0$
- $\dot{Y}_{II} = -P_{R(Y_{II})}\dot{A}P_{R(Y_{II}^\top)},\ Y_{II}(t_0) = Y_I(t_1)$
- $\dot{Y}_{III} = P_{R(Y_{III})}\dot{A},\ Y_{III}(t_0) = Y_{II}(t_1)$
- $Y_1 = Y_{III}(t_1)$

Semantic analysis

Aligned Word2Vec

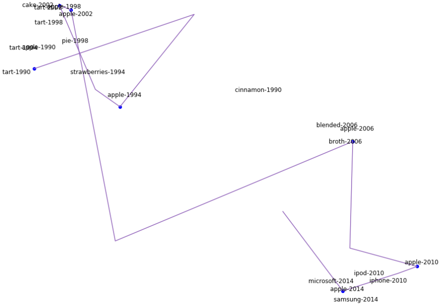

Figure: Trajectory of word apple through time for aligned Word2Vec

Semantic analysis

Dynamic Word2Vec

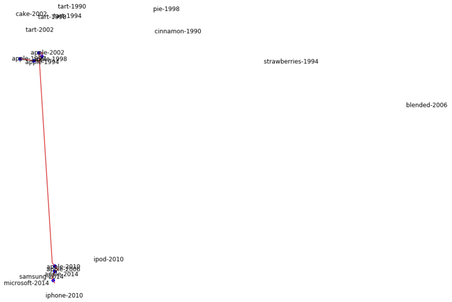

Figure: Trajectory of word apple through time for dynamic Word2Vec

Semantic analysis

Integrated Word2Vec

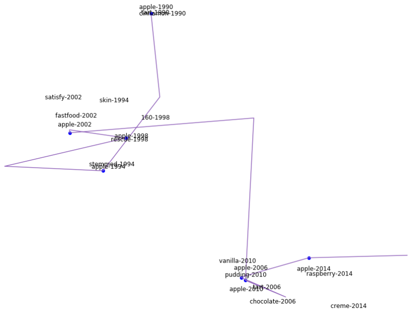

Figure: Trajectory of word apple through time for integrated Word2Vec

Comparison

K-means

$$NMI(L, C) = \frac{2I(L, C)}{H(L) + H(C)}$$

Table: NMI

Clusters	Aligned	Ridge	Projector
5	0.1786	0.2090	0.0758
10	0.2199	0.2452	0.0931
15	0.2157	0.2488	0.0969
20	0.2211	0.2514	0.0983

Comparison

K-means

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}, \beta = 5$$

Table: F_β

Clusters	Aligned	Ridge	Projector
5	0.5866	0.5023	0.3254
10	0.4946	0.4486	0.1585
15	0.4077	0.4007	0.1223
20	0.3778	0.3539	0.0829

References

Christian Lubich and Ivan Oseledets.
A projector-splitting integrator for dynamical low-rank approximation.
ArXiv e-prints, page arXiv:1301.1058, January 2013.

Zijun Yao, Yifan Sun, Weicong Ding, Nikhil Rao, and Hui Xiong.
Dynamic word embeddings for evolving semantic discovery.
Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining - WSDM '18, 2018.

Appendix

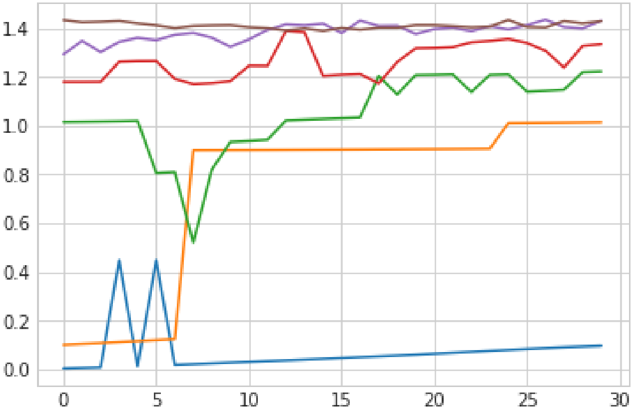

Figure: divergence of U and V for small random matrices