Knowledge-Driven Event Embedding for Stock Prediction

(originally by X. Ding, Y. Zhang, T. Liu, J. Duan; 2016)[1]

Kovalev Evgeny

Faculty of Mathematics, Higher School of Economics

Moscow, 2018

Event-driven stock prediction

- event E = (Agent, Predicate, Object)
- learn distributed representations of structured events (event embeddings)
- use them as the basis to generate textual features for predicting price movements in stock markets

Event embeddings

- capture both the syntatic and the semantic information among events
- alleviate the sparsity of discrete events compared with one-hot feature vectors
- Ding et al. (2015):

$$\text{word embeddings of A, P, O} \; + \; \text{NTN} \; = \text{event embeddings}$$

- problems:
    - event embeddings cannot capture the relationship between two syntatically or semantically similar events if they do not have similar word vectors
    - events with similar word embeddings may be unrelated («Steve Jobs quits Apple» and «John leaves Starbucks»)

Idea

- incorporate the external information from knowledge graphs (Freebase and YAGO)
- knowledge graph:
    - vertice = entity, edge = relation
    - categorical knowledge
    - relational knowledge
- use vectors learned from unsupervised large corpora for initialization

Model overview

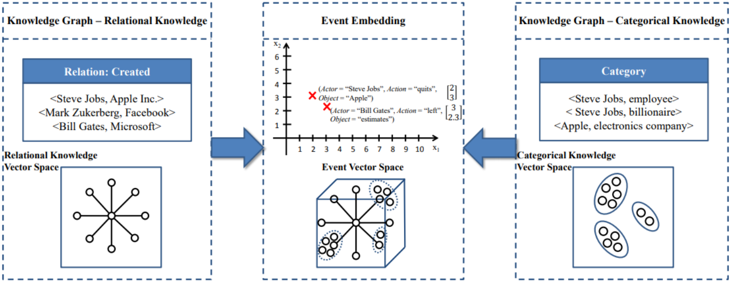
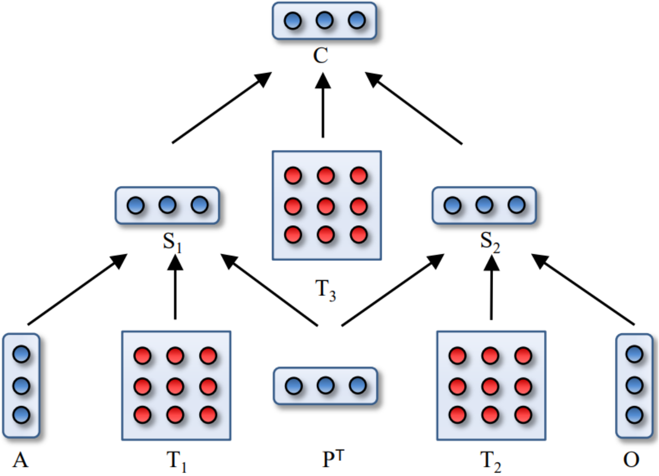


Figure 1: Incorporating knowledge graph into the learning process for event embeddings.

Event Embedding



Event Embedding

$$S_1 = g(A, P) = f\left(A^T T_1^{[1:k]} P + W \begin{bmatrix} A \\ P \end{bmatrix} + b\right) \in \mathbb{R}^d$$

- $T_1^{[1:k]} \in \mathbb{R}^{d \times d \times k}$ — tensor (a set of k matrices with $d \times d$ dimensions)
- $A^T T_1^{[1:k]} P = r \in \mathbb{R}^k$ — vector with entries $r_i = A^T T_1^{[i]} P,\ i = 1, \ldots, k$
- $W \in \mathbb{R}^{k \times 2d}$ — weight matrix, $b \in \mathbb{R}^k$ — bias vector, $f = \tanh$ — activation function
- pre-trained word embeddings (skip-gram algorithm)
- S_2, C are computed in the same way

Event Embedding

- event $E = \left(A, P, O\right)$
- corrupted event $E^r = \left(A^r, P, O\right)$ (replace each word in A with a random word from the vocabulary)
- margin loss:

$$L_\varepsilon = \text{loss}\left(E, E^r\right) = \max\left(0, 1 - g\left(E\right) + g\left(E^r\right)\right) + \lambda \|\Phi\|_2^2 \longrightarrow \min$$

  - $\Phi = \left(T_1, T_2, T_3, W, b\right)$ — the set of model parameters
  - the standard L_2 regularization, $\lambda = 0.0001$

Knowledge Graph Embedding

Differences from event embedding:

- relation is a tensor, not a vector
- a simpler NTN model, which is easier to train

The probability that entities e_1 and e_2 are in a relationship R:

$$g\left(e_1, R, e_2\right) = \mu_R^T f\left(e_1^T H_R^{[1:k]} e_2 + V_R \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} + b_R\right)$$

- $H_R^{[1:k]} \in \mathbb{R}^{d \times d \times k}$ — tensor (a set of k matrices with $d \times d$ dimensions)
- $e_1^T H_R^{[1:k]} e_2 = x \in \mathbb{R}^k$ — vector with entries $x_i = e^T H_R^{[i]} e_2,\ i = 1, \ldots, k$
- $V_R \in \mathbb{R}^{k \times 2d}$ — weight matrix, $b \in \mathbb{R}^k$ — bias vector, $f = \tanh$ — activation function

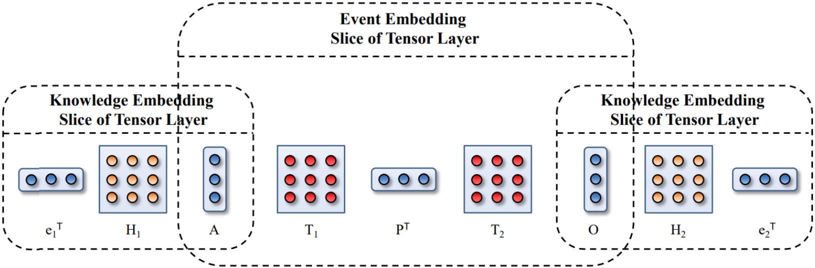
Knowledge Graph Embedding

- tuple $T^{(i)} = \left(e_1^{(i)}, R^{(i)}, e_2^{(i)}\right)$
- corrupted tuple $T_c^{(i)} = \left(e_1^{(i)}, R^{(i)}, e_c^{(i)}\right)$ (replace one of the entities with a random entity)
- loss:

$$L_K = \sum_{i=1}^{N} \sum_{m=1}^{M} \max\left(0, 1 - g\left(T^{(i)}\right) + g\left(T_c^{(i)}\right)\right) + \lambda \|\Omega\|_2^2 \longrightarrow \min$$

  - $\Omega = \left(\mu, H, V\right)$ — the set of model parameters
  - N - number of training tuples, M — number of counterparts for each correct tuple
  - the standard L_2 regularization

Joint Knowledge and Event Embedding



- loss:

$$L = \alpha L_{\varepsilon} + (1-\alpha) L_K \longrightarrow \min$$

- $\alpha \in [0,1]$ — parameter $(\alpha = 0.4)$

Event Similarity

Table 2: Experimental results on event similarity and its effect on S&P 500 index prediction. The improvement is significant at $p < 0.05$.

Methods	Spearman's Rank Correlation	Acc	MCC
DE	0.437	58.83%	0.1623
EB	0.591	64.21%	0.4035
KGEB	**0.616**	**66.93%**	**0.5072**

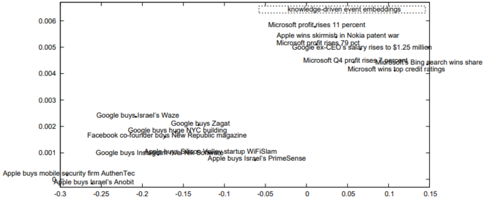
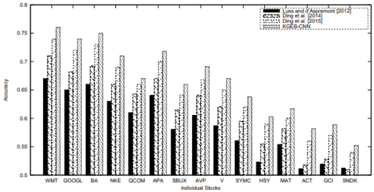


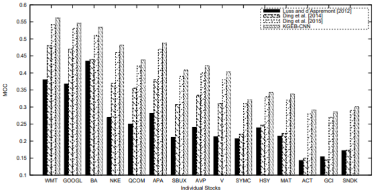
Figure 4: Two-dimensional PCA projection of 100-dimensional knowledge-driven event vectors.

Stock Prediction

Table 3: Experimental results on index prediction.

	Acc	MCC
Luss and d'Aspremont (2012)	56.38%	0.0711
Ding et al. (2014)	58.83%	0.1623
WB-CNN	60.57%	0.1986
Ding et al. (2015)	64.21%	0.4035
KGEB-CNN	**66.93%**	**0.5072**





Thank you for your attention!