Distributed word representations: Retrofitting

Christopher Potts

Stanford Linguistics

CS224u: Natural language understanding

Central goals

- Distributional representations are powerful and easy to obtain, but they tend to reflect only similarity (synonymy, connotation).
- Structured resources are sparse and hard to obtain, but they support learning rich, diverse semantic distinctions.
- Can we have the best aspects of both? Retrofitting is one way of saying, “Yes”.
- Retrofitting is due to Faruqui et al. (2015).

The retrofitting model

$$\sum_{i \in \mathbf{V}} \alpha_i \left\| \boldsymbol{q}_i - \hat{\boldsymbol{q}}_i \right\|^2 + \sum_{(i,j,r) \in \mathbf{E}} \beta_{ij} \left\| \boldsymbol{q}_i - \boldsymbol{q}_j \right\|^2$$

- Balances fidelity to the original vector $\hat{\boldsymbol{q}}_i$
- against looking more like one's graph neighbors.
- Forces are balanced with $\alpha = 1$ and $\beta = \frac{1}{\mathbf{Degree}(i)}$

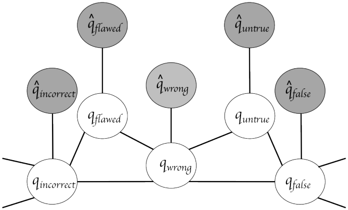
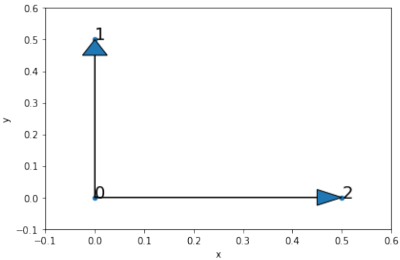
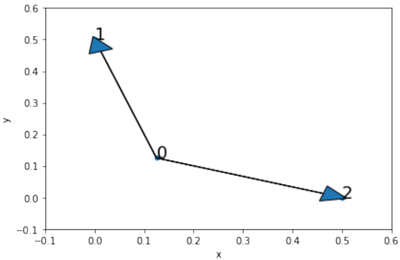


Figure 1: Word graph with edges between related words showing the observed (grey) and the inferred (white) word vector representations.

Simple retrofitting examples

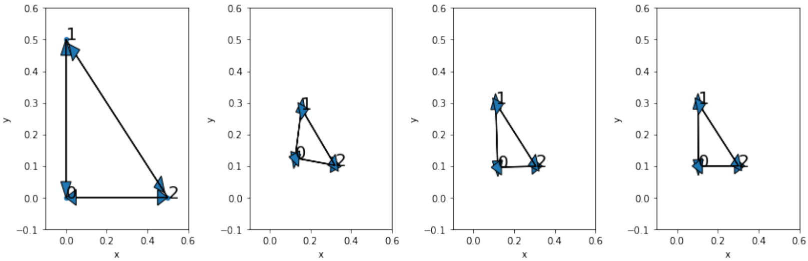
$$\sum_{i \in \mathbf{V}} \alpha_i \left\| \boldsymbol{q}_i - \hat{\boldsymbol{q}}_i \right\|^2 + \sum_{(i,j,r) \in \mathbf{E}} \beta_{ij} \left\| \boldsymbol{q}_i - \boldsymbol{q}_j \right\|^2$$





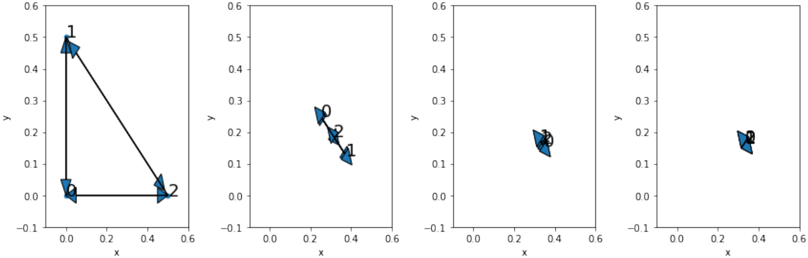
Simple retrofitting examples

$$\sum_{i \in \mathbf{V}} \alpha_i \left\| \boldsymbol{q}_i - \hat{\boldsymbol{q}}_i \right\|^2 + \sum_{(i,j,r) \in \mathbf{E}} \beta_{ij} \left\| \boldsymbol{q}_i - \boldsymbol{q}_j \right\|^2$$



Simple retrofitting examples

$$\sum_{i \in \mathbf{V}} \alpha_i \left\| \boldsymbol{q}_i - \hat{\boldsymbol{q}}_i \right\|^2 + \sum_{(i,j,r) \in \mathbf{E}} \beta_{ij} \left\| \boldsymbol{q}_i - \boldsymbol{q}_j \right\|^2$$



$\alpha = 0$

Extensions

Drop the assumption that every edge means ‘similar’:

- Mrkšić et al. (2016) AntonymRepel, SynonymAttract, and VectorSpacePreservation for different edge types.
- Lengerich et al. (2018): functional retrofitting to learn the semantics of any edge types.
- This work is closely related to **graph embedding** (learning distributed representations for nodes), for which see Hamilton et al. 2017.

Code snippets

```
[1]: import pandas as pd
     from retrofitting import Retrofitter

[2]: Q_hat = pd.DataFrame(
         [[0.0, 0.0],
          [0.0, 0.5],
          [0.5, 0.0]],
         columns=['x', 'y'])

     edges = {0: {1, 2}, 1: set(), 2: set()}

[3]: Q_hat

[3]:      x    y
     0  0.0  0.0
     1  0.0  0.5
     2  0.5  0.0

[4]: retro = Retrofitter(verbose=True)

[5]: X_retro = retro.fit(Q_hat, edges)

     Converged at iteration 2; change was 0.0000

[6]: X_retro

[6]:        x      y
     0  0.125  0.125
     1  0.000  0.500
     2  0.500  0.000

[7]: # For an application to WordNet, see `vsm_03_retrofitting`.
```

References I

Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1606–1615, Stroudsburg, PA. Association for Computational Linguistics.

William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Representation learning on graphs: Methods and applications. In *IEEE Data Engineering Bulletin*, pages 52–74. IEEE Press.

Benjamin J. Lengerich, Andrew L. Maas, and Christopher Potts. 2018. Retrofitting distributional embeddings to knowledge graphs with functional relations. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2423–2436, Stroudsburg, PA. Association for Computational Linguistics.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting word vectors to linguistic constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–148. Association for Computational Linguistics.