Learning Dense Vector Representations for Named Entity Relations

Abdi-Hakin Dirie, Emily Kellison-Linn, Jason Tong

Question

Can we find a vector representation for the relationship between two entities, such that pairs of entities that exhibit the same relation have similar relation vectors?

Background

Mikolov *et al*. demonstrated semantic regularities in word vectors trained using the skip-gram model.

Results showed it to be possible to complete analogies such as:

king : queen :: man : ? (woman)
China : Beijing :: Spain : ? (Madrid)

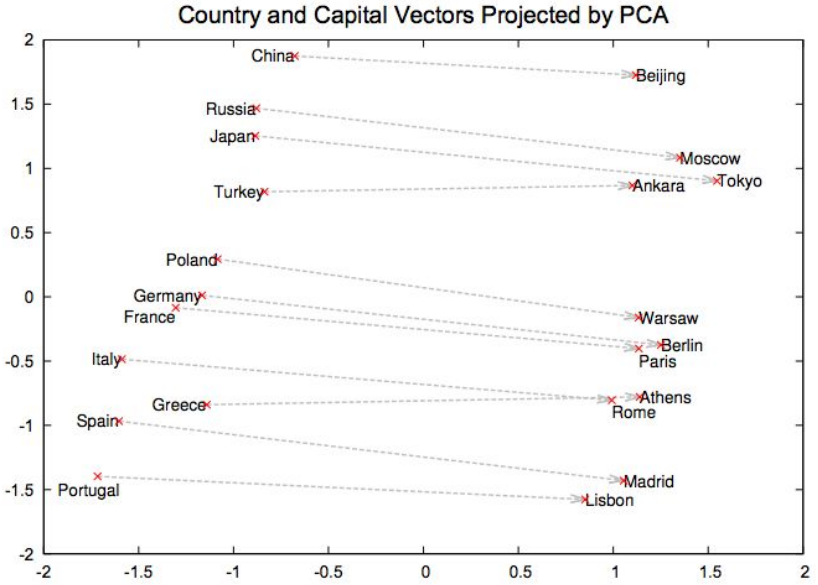


Figure 2: Two-dimensional PCA projection of the 1000-dimensional Skip-gram vectors of countries and their capital cities. The figure illustrates ability of the model to automatically organize concepts and learn implicitly the relationships between them, as during the training we did not provide any supervised information about what a capital city means.

Data

- Pairs of related entities mined from Wikidata
  - e.g. *spouse(Barack Obama, Michelle Obama)*
- Convert each entity to dense vector representation using Google's Word2Vec
- Each data point consists of two pairs and a label that says whether the pairs are instances of the same relation.

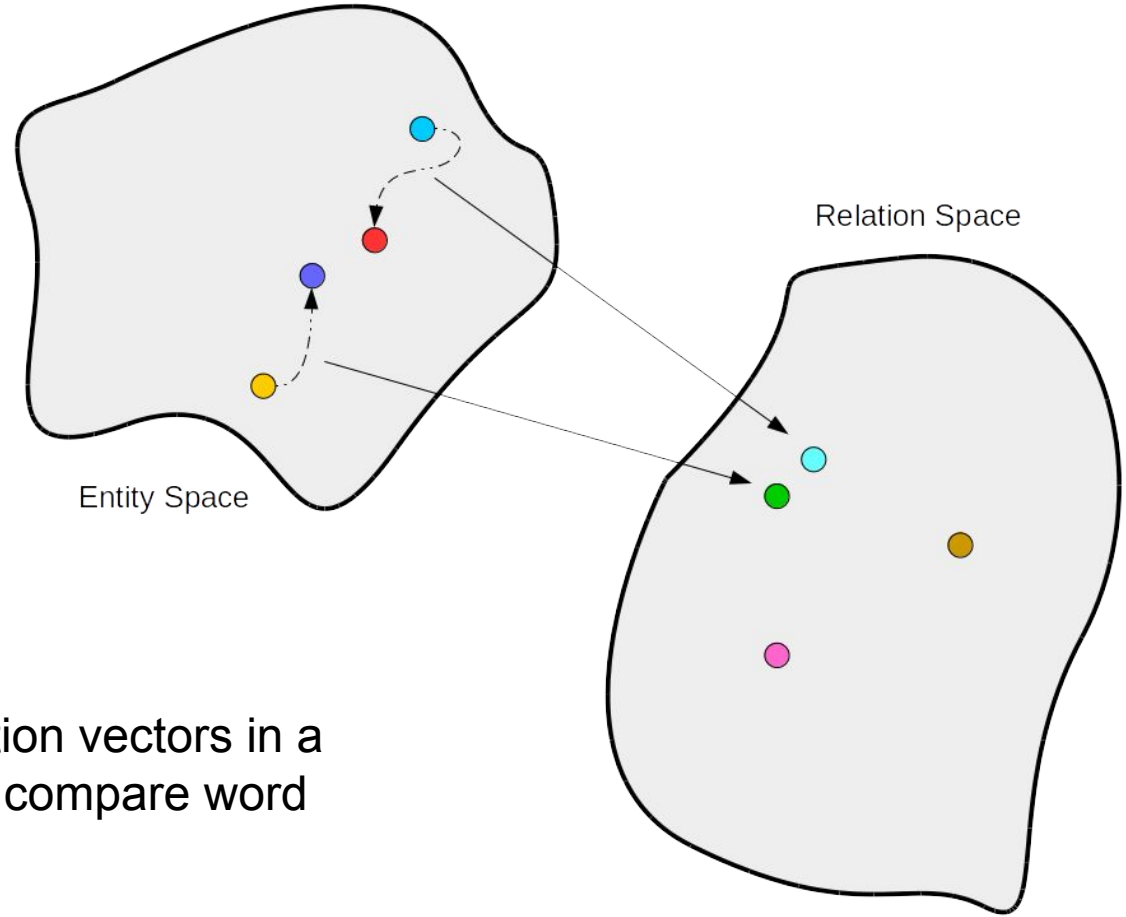
Example:

1. *spouse(Barack Obama, Michelle Obama)*
2. *spouse(Bill Gates, Melinda Gates)*
3. *capital(Russia, Moscow)*

1 and 2 → +1 (same relation)
1 and 3 → -1 (different relation)

Training: 3812 data points. Test: 200 data points

Visual Idea

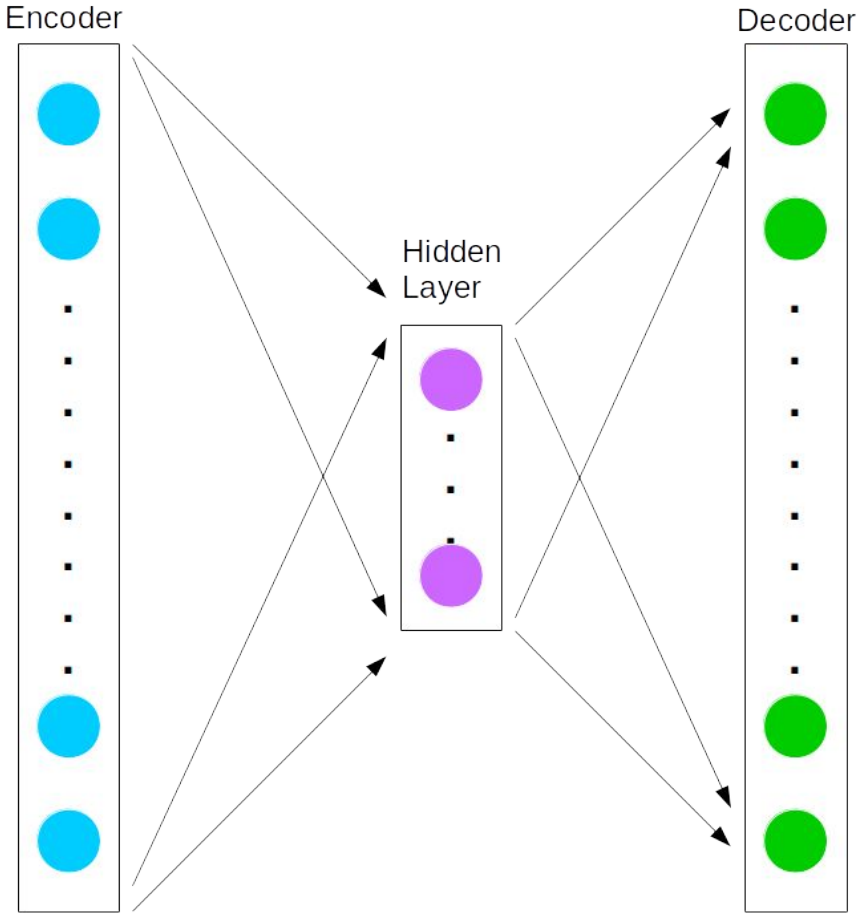


We’d like to compare relation vectors in a manner similar to how we compare word vectors.

Autoencoder

- Maps a pair of entities to another pair of the same relation
- Input dimension: 2000
- Output dimension: 2000
- Hidden layer dimension: 1000
  - Hidden will be our representation of a relation.

Autoencoder training set has ~29000 of these pair-to-pair examples.



Relation Representations

Concatenation

Simply concatenate the Word2Vec representation of each entity in the pair into one vector.

Difference

Take the difference between the two vector representation of the entities.

Autoencoder Output

Pass concatenation representation through the autoencoder and take the output of the hidden layer.

Evaluation methods

SVM

Use gaussian kernel SVM to classify pairs of relation vectors as “same relation” or “different relation”

Neural network

Use single-layer neural network to classify pairs of relation vectors as “same relation” or “different relation”

k-Nearest Neighbors

Use 10-Nearest Neighbors to label pairs of relation vectors to see if they are labeled with the same relation class.

Evaluation results

Classification accuracies on test set

Model	Concatenation	Subtraction	Autoencoder
SVM	64.0%	60.5%	**85.5%**
Neural network	57.0%	57.0%	54.0%
k-Nearest Neighbors	72.5%	75%	68.5%

Discussion

Greater-than-50% classification accuracy for concatenated vectors indicates that some regular pattern of relationships exists.

Unexpectedly, one-layer neural network models were weak at given task for the three representations.

Autoencoder representation achieved high accuracy with SVM classification.

Future Work

Build more complex autoencoder models for learning relation representations.

Investigate similarities and differences between relations in the vector space.

Enable bag-of-words approach to NER by checking relation representations between all pairs of named entities in a text against a known set of relations.

Use relation vectors for efficient discovery of a given type of relation in texts, which can aid the training of syntactic models for NER.