Joint neural embeddings of synsets and entities

Minh Ngoc Le

VU Amsterdam University

December 10, 2014

BabelNet

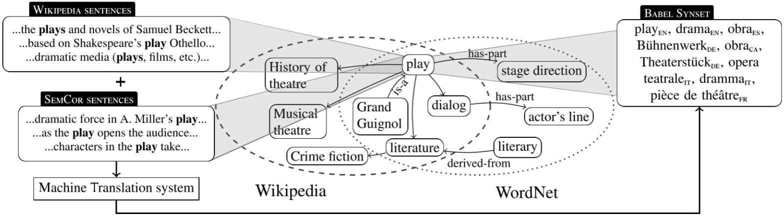
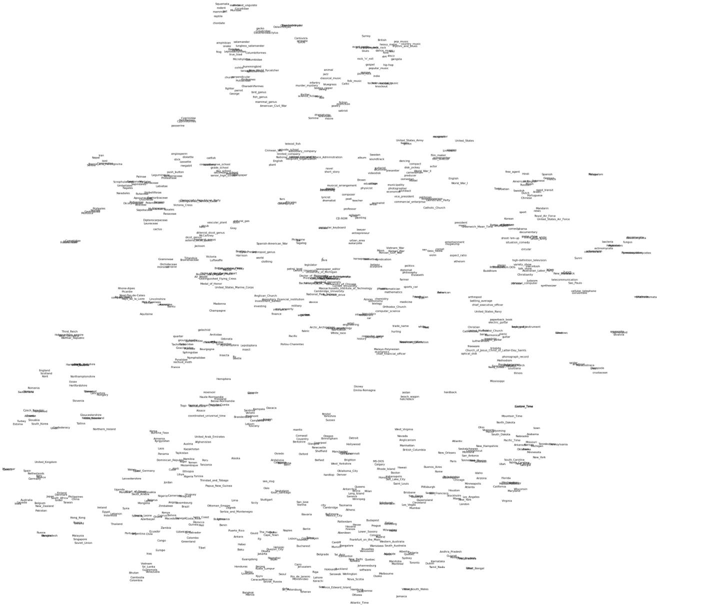


Figure: An illustrative overview of BabelNet (labeling nodes with English lexicalizations only): unlabeled edges are obtained from links in the Wikipages (e.g., Play (theatre) links to Musical theatre), whereas labeled ones from WordNet (e.g., play^1_n has-part stage direction^1_n). From Navigli & Ponzetto (2012).

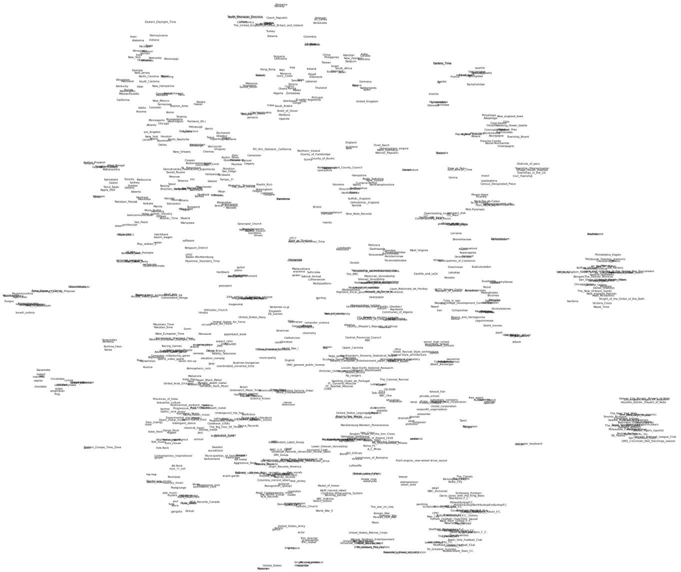
Statistics

- Using half of the synsets (for efficiency)
- Synsets: 1,435,175 (1.4 million)
  - WordNet only: 8,777 (0.6%)
  - Wikipedia only: 1,392,735 (97%)
  - WordNet+Wikipedia: 32,532 (2.3%)
- Unique relations: 13,146,719 (13 million)

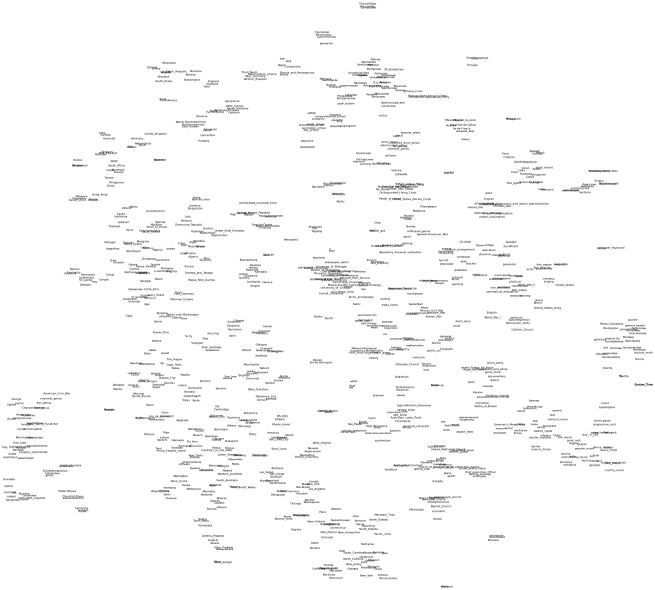
WordNet embeddings



Wikipedia embeddings



WordNet+Wikipedia embeddings



WordNet+Wikipedia embeddings: Zoom 1

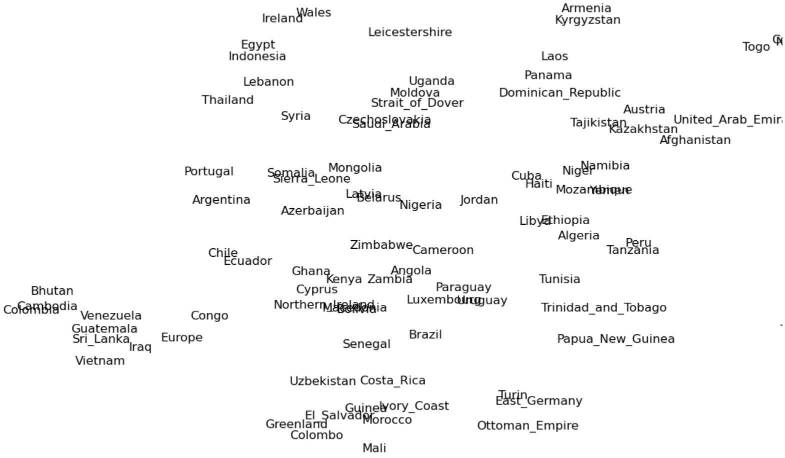


Figure: Countries (Leicestershire is a landlocked county in the English Midlands, Mali is the eighth largest country in Africa)

WordNet+Wikipedia embeddings: Zoom 2

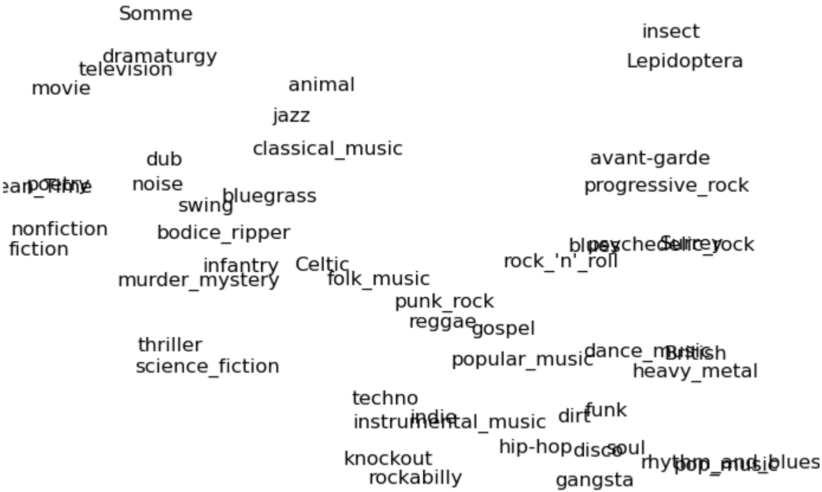


Figure: Movie and music genres

WordNet+Wikipedia embeddings: Zoom 3

Figure: Cities and states

WordNet+Wikipedia embeddings: Zoom 4

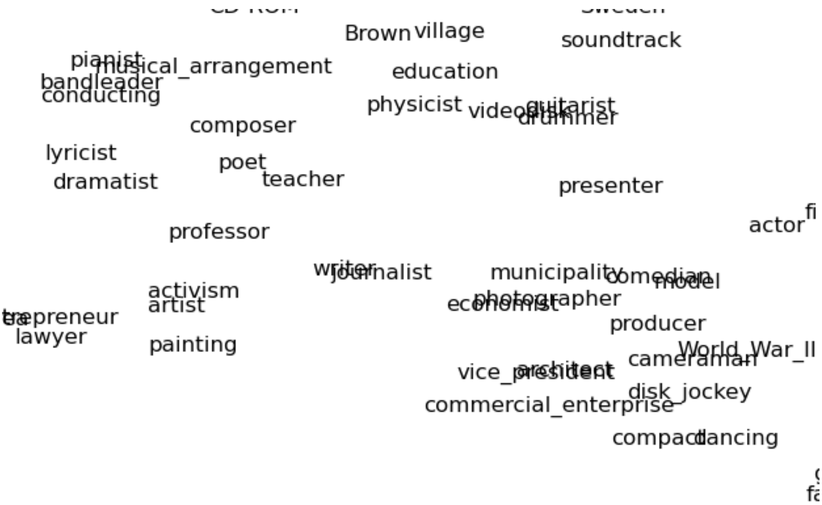


Figure: Occupations

Formalization

- Let D be the distribution of correct relations
- Let N be another random distribution (e.g. uniform over all possible combinations)
- Draw a large number (e.g. hundreds of millions) of relations from both
- Learn the function $h(e_1, r, e_2)$: how likely the relation was drawn from D instead of N
- For example, h(cat, has_part, tail)$= 0.8 \Rightarrow$ it is very likely that (cat, has_part, tail) was drawn from D instead of N

Building training set

Positive examples (D):

1. `cat has_part tail`
2. `dog has_part tail`
3. `car has_part wheel`
4. ...

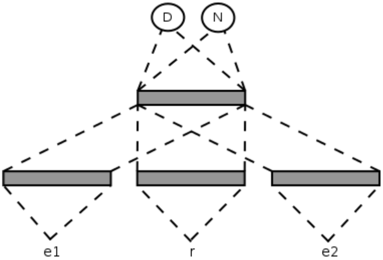
Negative examples (N):

1. `cat has_part wheel`
2. `car has_part tail`
3. `car has_part tail`
4. ...

- Generated 10 negative examples for each positive example
- Replicated positive examples to maintain the balance
- Total: 202,922,934 (203 million)

Model

- Projection layer: converts labels into embeddings
- Hidden layer: leaky rectified linear units (explained later)
- Output layer: two-class logistics



Leaky rectified linear units

- As normal units, inputs are weighted and accumulated into *net input* $z \in \mathbb{R}$
- *Activation function* compute the activation of the unit:

$$f(z) = \begin{cases} z & \text{if } z \geq 0 \\ 0.01z & \text{otherwise} \end{cases}$$

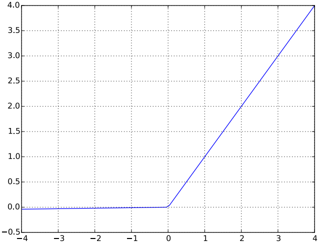


Figure: Activation depending on net input

Leaky rectified linear units

- Each unit acts as feature detector
- The detection boundary depends on the value of weights and biases
- At one side of the boundary, activation increase
- At the other side, activation stays (almost) unchanged

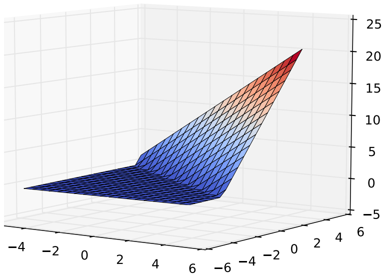


Figure: Activation of a leaky rectified linear unit with two inputs, $a = f(2x + 3y - 2)$

Forward propagation

- Given an input, hidden units detect various patterns
- Output units average them to make decision

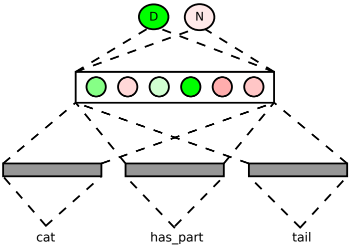
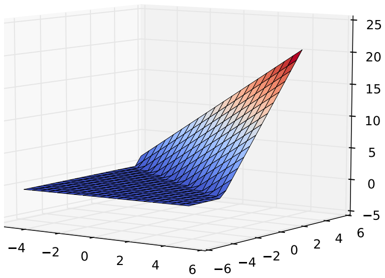


Figure: Forward propagation for an example: (cat, has_part, tail).

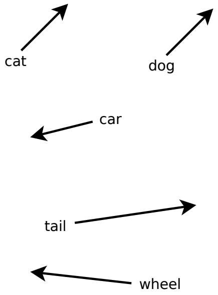
Backward propagation

- How to update an embedding based on the activation function in the image:
- If the activation is associated with the correct choice: move to the right
- If the activation is associated with the incorrect choice: move to the left



Backward propagation

- Similar concepts/entities appear in similar relations therefore are pushed to the same direction
- Dissimilar concepts/entities appear in different relations therefor are pushed to different directions
- After updating many times, similar synsets are clustered together



Applications

- Named entity disambiguation
- Word sense disambiguation
- Semantic parsing
- Similarity judgment
- ...

Thank you!