Other Word Embedding Types

Natalie Parde

UIC CS 421

Are there any other variations of Word2Vec?

- **fasttext**
  - An extension of Word2Vec that also incorporates **subword models**
  - Designed to better handle unknown words and sparsity in language

fasttext

- Each word is represented as:
  - Itself
  - A bag of constituent n-grams

super = <super> + <su, sup, upe, per, er>

fasttext

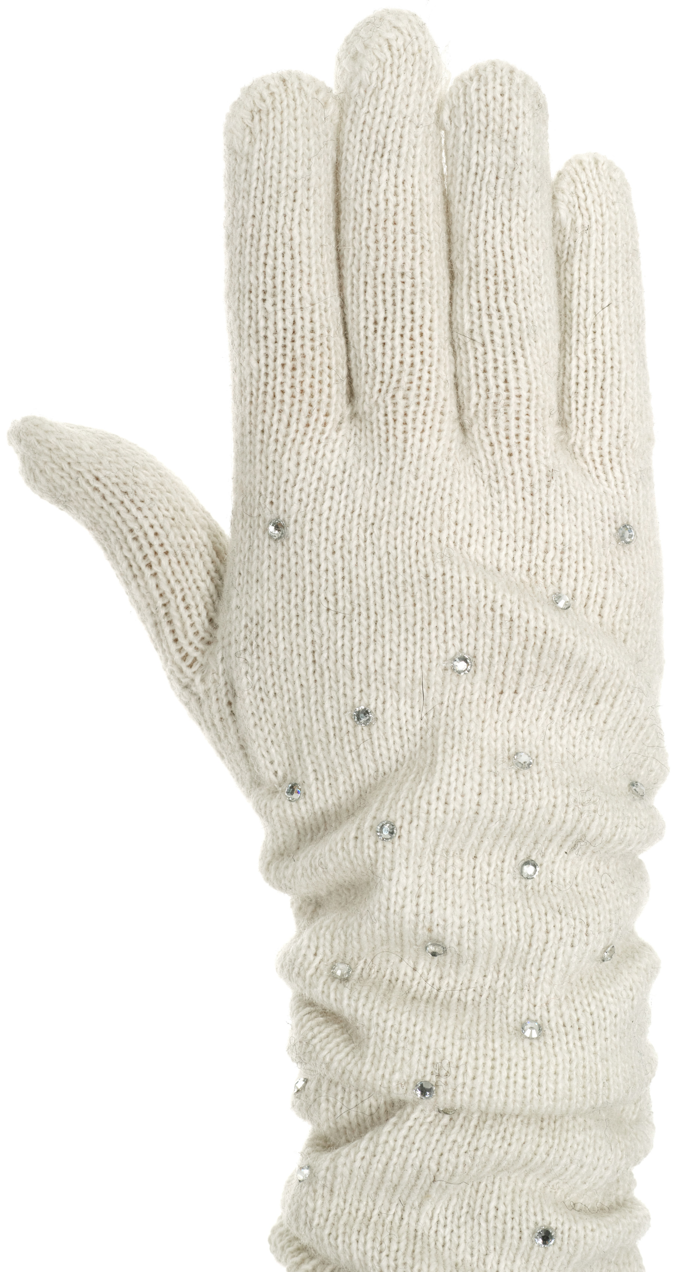
- Skip-gram embedding is learned for each constituent n-gram
- Word is represented by the sum of all embeddings of its constituent n-grams

- Key advantage of this extension?
  - Allows embeddings to be predicted for unknown words based on subword constituents alone

Source code available online: https://fasttext.cc/

Word2Vec and fasttext embeddings are nice …but what's another alternative?

- Word2Vec is an example of a **predictive** word embedding model
  - Learns to predict whether words belong in a target word's context
- Other models are **count-based**
  - Remember co-occurrence matrices?
- GloVE combines aspects of both predictive and count-based models



Global Vectors for Word Representation (GloVe)

- Co-occurrence matrices quickly grow extremely large
- Intuitive solution to increase scalability?
  - Dimensionality reduction!
    - However, typical dimensionality reduction strategies may result in too much computational overhead
- GloVe learns to predict weights in a lower-dimensional space that correspond to the co-occurrence probabilities between words

GloVe

- Why is this useful?
  - Predictive models → black box
    - They work, but why?
  - GloVe models are easier to interpret
- GloVe models also encode the ratios of co-occurrence probabilities between different words …this makes these vectors useful for word analogy tasks

How does GloVe work?

	c_1	…	c_n
t_1	123	…	456
…	…	…	…
t_n	0	…	789

Build a huge word-context co-occurrence matrix

How does GloVe work?

	c_1	…	c_n
t_1	123	…	456
…	…	…	…
t_n	0	…	789

Build a huge word-context co-occurrence matrix

Define soft constraints for each word pair

$$\underbrace{w_i^T}_{\text{Vector for } t_i} \underbrace{w_j}_{\text{Vector for } c_j} + \overbrace{b_i + b_j}^{\text{Scaler biases for } t_i \text{ and } c_j} = \log \underbrace{X_{ij}}_{\text{Co-occurrence count for } t_i c_j}$$

How does GloVe work?

	c_1	…	c_n
t_1	123	…	456
…	…	…	…
t_n	0	…	789

Build a huge word-context co-occurrence matrix

Define soft constraints for each word pair

$$w_i^T w_j + b_i + b_j = \log X_{ij}$$

Define a cost function

$$J = \sum_{i=1}^{V} \sum_{j=1}^{V} f(X_{ij})(w_i^T w_j + b_i + b_j - \log X_{ij})^2$$

Weighting function:

$$f(X_{ij}) = \begin{cases} (\frac{X_{ij}}{x_{max}})^{\alpha}, & X_{ij} < XMAX \\ 1, & \text{otherwise} \end{cases}$$

How does GloVe work?

	c_1	…	c_n
t_1	123	…	456
…	…	…	…
t_n	0	…	789

Build a huge word-context co-occurrence matrix

Define soft constraints for each word pair

$$w_i^T w_j + b_i + b_j = \log X_{ij}$$

Define a cost function

$$J = \sum_{i=1}^{V} \sum_{j=1}^{V} \overbrace{f(X_{ij})}(w_i^T w_j + b_i + b_j - \log X_{ij})^2$$

Minimize the cost function to learn ideal embedding values for w_i and w_j

How does GloVe work?

	c_1	…	c_n
t_1	123	…	456
…	…	…	…
t_n	0	…	789

Build a huge word-context co-occurrence matrix

Define soft constraints for each word pair

$$w_i^T w_j + b_i + b_j = \log X_{ij}$$

Define a cost function

$$J = \sum_{i=1}^{V} \sum_{j=1}^{V} \overbrace{f(X_{ij})} (w_i^T w_j + b_i + b_j - \log X_{ij})^2$$

Minimize the cost function to learn ideal embedding values for w_i and w_j

0.4	0.7	1.2	4.3	0.9	6.7	1.3	0.5	0.7	5.3

In sum, GloVe is a log-bilinear model with a weighted least-squares objective.

- Why does it work?
  - Ratios of co-occurrence probabilities have the potential to encode word similarities and differences
  - These similarities and differences are useful components of meaning
- GloVe embeddings perform particularly well on analogy tasks

Which is best …Word2Vec or GloVe?

- It depends on your data!
- In general, Word2Vec and GloVe produce similar embeddings
- Word2Vec → slower to train but less memory intensive
- GloVe → faster to train but more memory intensive
- Word2Vec and Glove both produce context-independent embeddings
- Contextual embeddings:
  - ELMo (Peters et al., 2018; https://www.aclweb.org/anthology/N18-1202/)
  - BERT (Devlin et al., 2019; https://www.aclweb.org/anthology/N19-1423/)