Overview of LSTMs and word2vec

and a bit about compositional distributional semantics if there's time

Ann Copestake

Computer Laboratory
University of Cambridge

November 2016

Outline

RNNs and LSTMs

Word2vec

Compositional distributional semantics

Some slides adapted from Aurelie Herbelot.

Outline.

RNNs and LSTMs

Word2vec

Compositional distributional semantics

Motivation

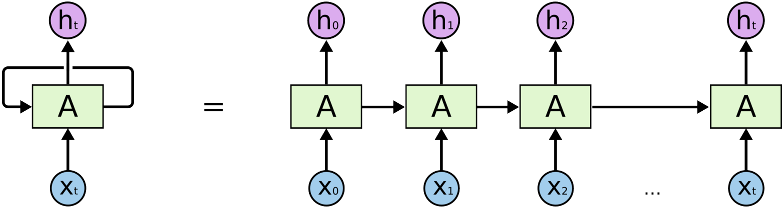
- Standard NNs cannot handle sequence information well.
- Can pass them sequences encoded as vectors, but input vectors are fixed length.
- Models are needed which are sensitive to sequence input and can output sequences.
- RNN: Recurrent neural network.
- Long short term memory (LSTM): development of RNN, more effective for most language applications.
- More info: `http://neuralnetworksanddeeplearning.com/` (mostly about simpler models and CNNs)
  `https://karpathy.github.io/2015/05/21/rnn-effectiveness/`

  `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`

Sequences

- Video frame categorization: strict time sequence, one output per input.
- Real-time speech recognition: strict time sequence.
- Neural MT: target not one-to-one with source, order differences.
- Many language tasks: best to operate left-to-right and right-to-left (e.g., bi-LSTM).
- attention: model 'concentrates' on part of input relevant at a particular point. Caption generation: treat image data as ordered, align parts of image with parts of caption.

Recurrent Neural Networks



http://colah.github.io/posts/2015-08-Understanding-LSTMs/

RNN language model: Mikolov et al, 2010

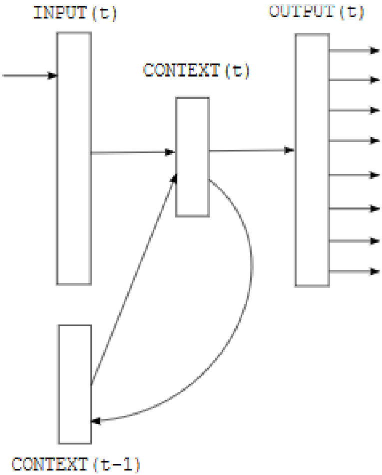


Figure 1: *Simple recurrent neural network.*

RNN as a language model

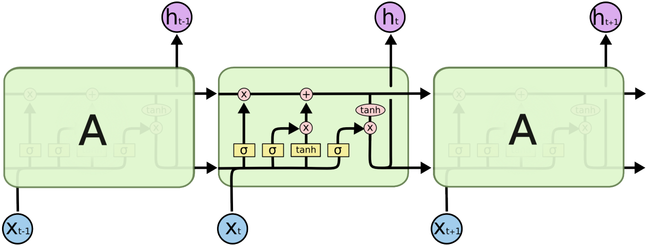
- Input vector: vector for word at t concatenated to vector which is output from context layer at $t-1$.
- Performance better than n-grams but won't capture 'long-term' dependencies:

  She shook her head.
  She decided she did not want any more tea, so shook her head when the waiter reappeared.
- not the same as long distance dependency in linguistics

Long Short Term Memory Networks (A)

- An RNN has just one layer in its repeating module.
- An LSTM has four layers that interact, each one with a gate. Gates are ways to let information through (or not):
  - Forget gate layer: look at previous cell state and current input, and decide which information to throw away.
  - Input gate layer: see which information in the current state we want to update.
  - Update layer: propose new values for the cell state.
  - Output layer: Filter cell state and output the filtered result.
- For instance: store gender of subject until another subject is seen.

Long Short Term Memory Networks



http://colah.github.io/posts/2015-08-Understanding-LSTMs/

State-of-the-art (possibly out of date already . . .)

- LSTMs have essentially replaced n-grams as language models for speech.
- Image captioning and other multi-modal tasks which were very difficult with previous methods are now feasible.
- Many traditional NLP tasks work very well with LSTMs, but not necessarily the top performers: e.g., POS tagging and NER: Choi 2016 — dynamic feature induction.
- Neural MT: broken away from plateau of SMT, especially for grammaticality (partly because of characters/subwords), but not yet industry strength.
- Definitely not there yet for text normalization: '33rpm' normalized to 'thirty two revolutions per minute'

https://arxiv.org/ftp/arxiv/papers/1611/1611.00068.pdf

Outline.

RNNs and LSTMs

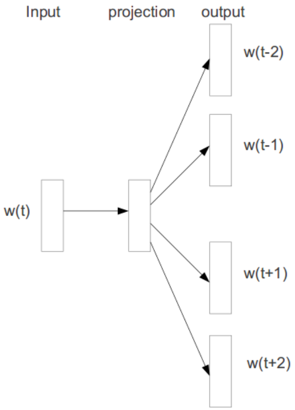
Word2vec

Compositional distributional semantics

Embeddings

- embeddings: distributional models with dimensionality reduction, based on prediction
- word2vec: as originally described (Mikolov et al 2013), a NN model using a two-layer network (i.e., not deep!) to perform dimensionality reduction.
- two possible architectures:
  - given some context words, predict the target (CBOW)
  - given a target word, predict the contexts (Skip-gram)
- Very computationally efficient, good all-round model (good hyperparameters already selected).

The Skip-gram model (A)



Features of Word2Vec representations (A)

- A representation is learnt at the reduced dimensionality straightaway: we are outputting vectors of a chosen dimensionality (parameter of the system).
- Usually, a few hundred dimensions: dense vectors.
- The dimensions are not interpretable: it is impossible to look into ‘characteristic contexts’.
- For many tasks, word2vec (skip-gram) outperforms standard count-based vectors.
- But mainly due to the hyperparameters and these can be emulated in standard count models (see Levy et al).

What Word2Vec is famous for (A)

BUT . . . see Levy et al and Levy and Goldberg for discussion

The actual components of Word2Vec (A)

- A vocabulary. (Which words do I have in my corpus?)
- A table of word probabilities.
- Negative sampling: tell the network what *not* to predict.
- Subsampling: don't look at all words and all contexts.

Negative sampling (A)

Instead of doing full softmax (very expensive), word2vec is trained using logistic regression to discriminate between real and fake words:

- Whenever considering a word-context pair, also give the network contexts which are not the actual observed word.
- Sample from the vocabulary. The probability to sample something more frequent in the corpus is higher.
- The number of negative samples will affect results.

Subsampling (A)

- Instead of considering all words in the sentence, transform it by randomly removing words from it:
  considering all sentence transform randomly words
- The subsampling function makes it more likely to remove a frequent word.
- Note that word2vec does not use a stop list.
- Note that subsampling affects the window size around the target (i.e., means word2vec window size is not fixed).
- Also: weights of elements in context window vary.

Using word2vec

- predefined vectors or create your own
- can be used as input to NN model
- many researchers use the gensim Python library
  `https://radimrehurek.com/gensim/`
- Emerson and Copestake (2016) find significantly better performance on some tests using parsed data
- Levy et al's papers are very helpful in clarifying word2vec behaviour
- Bayesian version: Barkan (2016)

  `https://arxiv.org/ftp/arxiv/papers/1603/1603.06571.pdf`

Outline.

RNNs and LSTMs

Word2vec

Compositional distributional semantics

Compositional semantics

- Compositional semantics is about providing a meaning representation for an entire sentence.
- Classically (e.g., Montague) based on syntax and morphology, meaning expressed as in logic:

  every white cat is asleep

  $\forall x[[\mathit{white}'(x) \wedge \mathit{cat}'(x)] \rightarrow \mathit{asleep}'(x)]$

- Structures built deterministically from a rich syntactic analysis (quantifier scope possibly underspecified).
- Useful in applications like database interfaces where predicates can be grounded.
- Can be automatically induced for limited domains.

Compositional distributional semantics

- Compositional distributional semantics is typically about the meaning of short phrases: e.g., *white cat*.
- Not so good at full sentences, or quantifiers, modals etc, which compositional semantics deals with.
- In standard compositional distributional semantics, *white cats* does not refer to entities which are white and cats, but is more like creation of a new concept.
- Hence (perhaps) success of additive models.
- Unlike formal semantics, compositional distributional semantics is good at semi-compositionality (compound nouns, *heavy table* vs *heavy rain* vs *heavy taxation* etc).

Next time

- In two weeks ...
- Possibly: more advanced distributional semantics techniques ...