Ling/CSE 472: Introduction to Computational Linguistics

5/21: Word Embeddings

Overview

- Term project milestone 2 feedback
- Word embeddings (J&M slides)
- RQs

Term project feedback

- Be sure to describe the *task* separately from the approach/model
    - What's the input?
    - What's the output?
        - Only then: How does the package you're working with approach it?
- Error categories should describe different ways in which the system is wrong and different linguistic properties of the input
    - Not guesses as to what happened system-internally

Word Embeddings

- => J&M slides

Reading questions

- I'm still confused on how vector semantics works fundamentally different than a simple frequency. Is the distinction that, simple word frequency is just the amount of times that a word appears in a document, whereas vector semantics deals with what words are most used with what other words? If that's the case, why do we have dense vector models of only 50-1000 dimensionality, when we should have vector spaces in the 50,000 - 100,000 to account for all words of the language?

- The chapter motivated why we want to use short vectors, but they gave a pretty large range of what a short vector would be (50-1000 entries). Is the length of a word vector in a word2vec model arbitrary, or what is it based on?

- Why does word2vec result in shorter, denser vectors, if it's still dealing with the full vocabulary?

Reading questions

- The reading states that dense vectors may help avoid overfitting since they contain fewer parameters than sparse vectors. Could you explain how this works? I don't really understand what the number of parameters has to do with the problem of overfitting.

-

Reading questions

- What is binary about the classification in word2vec? Is it that the model considers apricot as much as the word w?

- If the probability value returned is either true or false, how will the model distinguish between multiple true probabilities? Is it able to produce what is more likely, or is it always a tuple of just target and word?

- "Skip-gram makes the strong but very useful simplifying assumption that all context words are independent, allowing us to just multiply their probabilities" I did not understand this. What does it mean to assume that all context words are independent?

Reading questions

- In section 6.8.2, negative and positive examples of training instances are given. How is it known whether these are negative or positive - is it a given based on the L count (i.e. none of the negatives appear in the four words surrounding apricot, therefore they are negative) or is it a logical assumption being made for the training?

- From the reading: "A noise word is a random word from the lexicon, constrained not to be the target word t." Do we also restrict the noise word to not be in the context of t? Couldn't it accidentally be a positive example otherwise?

Reading questions

- I’m still not quite sure I understand why, when using word2vec, there are two separate embeddings for target and context words. I might be missing something, but why not just use the same vector space for both?

- The reading states that dense vectors may help avoid overfitting since they contain fewer parameters than sparse vectors. Could you explain how this works? I don't really understand what the number of parameters has to do with the problem of overfitting.

Reading questions

- The reading says that the skip-gram model bases the probability of a word being a real context word on the similarity between that and the context word's embeddings. But are words with similar embeddings actually all that likely to show up near each other? I was under the impression that similar words are more likely to be able to replace each other (like "cat" being used in similar contexts to "dog"), but that related words are more likely to be near each other (like "dog" showing up near "bone").

- When noise words are chosen based on their weighted unigram frequency, how exactly do we pick out this weight value (alpha)?

Reading questions

- It seems very tempting to look for other tools that are similar to word vector-learning in their use of unlabeled data. Are there any other NLP tasks, or really any other ML tasks in general, that are like this - supervised but seemingly effortless? Do those tasks have similarly weaknesses in terms of the potential vacuousness of the material you get out of it?

Reading questions

- Regarding the vector addition/subtraction interaction with embeddings, are the examples cherry picked? For example, how does this relate something like dragon - wings? Would that be lizard? or sea - water + sand = "sand sea" or maybe "desert"?

Reading questions

- Does first-order co-occurrence imply second-order co-occurrence? The converse does not seem true, but it seems to me that if two words are typically nearby each other, then they should have similar neighbors?

It's also often useful to distinguish two kinds of similarity or association between words (Schütze and Pedersen, 1993). Two words have **first-order co-occurrence** (sometimes called **syntagmatic association**) if they are typically nearby each other. Thus *wrote* is a first-order associate of *book* or *poem*. Two words have **second-order co-occurrence** (sometimes called **paradigmatic association**) if they have similar neighbors. Thus *wrote* is a second-order associate of words like *said* or *remarked*.

Reading questions

- I am curious about how is the difference between word2vec and BERT, glove, and ELMO. Since word2vec assign one vector value to one word, I am also curious how word2vec perform when facing words that have a different meaning, such as Apple as fruit or company.

Reading questions

- I find it interesting how embeddings can study historical semantics. It's cool to see what contexts the words were used in back in the day instead of just looking up the meaning of a word because a lot more information is provided. I do find it a little strange that in Figure 6.14, awful (1990s) is associated with words like 'wonderful' because I've never heard of that usage before, so I am wondering where they got their data from.

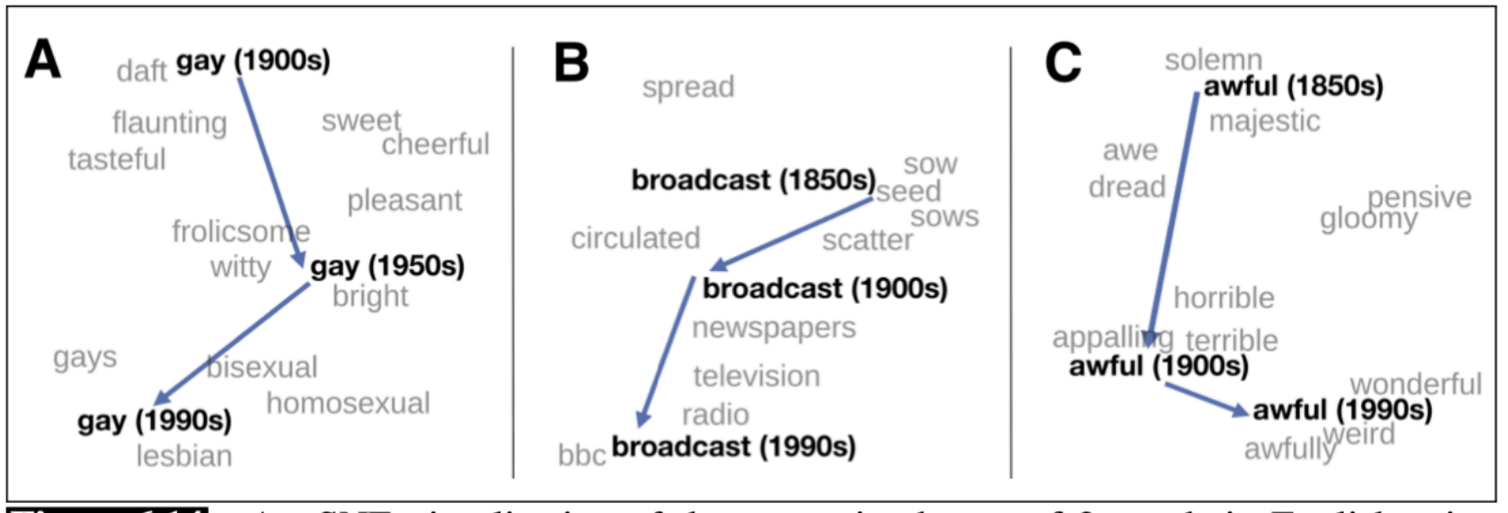

Figure 6.14 A t-SNE visualization of the semantic change of 3 words in English using word2vec vectors. The modern sense of each word, and the grey context words, are computed from the most recent (modern) time-point embedding space. Earlier points are computed from earlier historical embedding spaces. The visualizations show the changes in the word *gay* from meanings related to "cheerful" or "frolicsome" to referring to homosexuality, the development of the modern "transmission" sense of *broadcast* from its original sense of sowing seeds, and the pejoration of the word *awful* as it shifted from meaning "full of awe" to meaning "terrible or appalling" (Hamilton et al., 2016).

Reading questions

- How can biases be removed from algorithms trained on text if in reality there are specific biases in text and conversation? Would a bias-free model be an accurate model?

- I'm reading that attempts to reduce bias against gender and race have been made recently in 2016--2019 but not completely eliminate it. Is it even possible to do so and if it is indeed possible, what steps would need to be taken to completely eliminate bias? Does our more progressive society that pushes for more gender-neutral and inclusive language help in this development as well?

- Is there a 'performance tradeoff' in attempts to reduce bias? If so what is the 'acceptable' difference'?