NLP 1 pen & paper ex. 2 ard snijders 12854913

Distributional semantics and word embeddings (Deadline: 20 November)

(a) The distributional hypothesis states that the meaning of a word can be defined by its use and, therefore, it can be represented as a distribution of contexts in which the word occurs in a large text corpus. Describe different types of context that can be used for this purpose. [1 point]

Note: I was unsure whether this question alluded to the phase of choosing from the variety of possible word windows through which contexts can be obtained, or the different ways contexts can be weighted - either can heavily affect the resulting distribution of contexts, and in this respect I found the "different types of context" to be somewhat ambiguous, hence I discuss them both.

The distribution of contexts for a lexical item is obtained through the use of one of the following word windows:

- Unfiltered word windows: n words on either side of the lexical item
- Filtered word windows: n words on either side of lexical item, but stop-word or words with certain POS-tags are omitted
- Lexeme windows: filtered or unfiltered word windows, but using stems of words
- Dependencies. Here, the context of a lexical item is denoted by the dependency structure it belongs to

The obtained contexts per lexical item can then be weighted in various ways.

Example sentence: "This is an **example** sentence sentence"

- Binary model: if context c co-occurs with word w, it will be represented as 1 in $\vec{w}$; otherwise it will be represented as 0.
  {this 1} {is 1} {an 1} {cat 0} …. {sentence 1}
- Basic frequency model: a context c will be represented with the total number of times it co-occurs with word w
  {this 1} {is 1} {an 1} {sentence 2}
- Characteristic model: each context c is assigned a weight representing how characteristic that context is for the word w (calculated with PMI).

(b) The contexts can be weighted using Pointwise Mutual Information (PMI). Explain how PMI is calculated and how individual probabilities are estimated from a text corpus, giving the formulae. [1 point]

Pointwise Mutual Information compares how often a word and its context co-occur against how often we would expect them to occur if they were independent.

PMI can be used as a measure to express how characteristic a context is for a given word. It measures the discrepancy between the joint probability of the word and the context. The PMI of a word given a context can be computed as:

$$\log \frac{P(c|w)}{P(c)}$$

P(c | w) denotes the probability of the context given the word, and *P(c)* denotes the probability of the context occurring. P(c | w) is calculated as:

$$P(c|w) = \frac{f(w,c)}{f(w)}$$

Where f(w,c) denotes the frequency of word w in context c, and f(w) the frequency of word w in all contexts. The probability of a context P(c) is calculated as:

$$P(c) = \frac{f(c)}{\sum_k f(c_k)}$$

Here, f(c) simply denotes the frequency of the context appearing in the entire corpus. This frequency is then normalized by the total number of contexts in the entire corpus.

Taking together the formulae for P(c) and P(c|w) then yields:

$$PMI(w,c) = \log \frac{f(w,c)\sum_k f(c_k)}{f(w)f(c)}$$

(c) Some words occur very rarely in the corpus. How does this affect their PMI scores as contexts? [2 points]

PMI is sensitive to rare words: context words that do not occur frequently will have a higher PMI score. If a certain context is rare, its probability P(c) will be low, which in turn will evaluate to a smaller value for the denominator in the PMI formula:

$$\log \frac{P(w,c)}{P(w)P(c)}$$

Resultantly, rare contexts will have higher PMI values.

(d) The goal of distributional word clustering is to obtain clusters of words with similar or related meanings. The following clusters have been produced in two different noun clustering experiments.

(i) How are the clusters produced in the two experiments different with respect to the similarity they capture? What lexico-semantic relations do the clusters exhibit? [2 points]

For experiment 1, the words in the clusters are semantically similar: the different words belong to the same semantic category and, within clusters, could to some extent be substituted for each other.

For experiment 2, the words in the clusters are related but not similar: the words belong to the same topic (traffic/driving related words, concert related words, etcetera - making them related) but the words do not belong to the same semantic category.

(ii) The same clustering algorithm, K-means, was used in both experiments. What was different in the setup of the two experiments that resulted in the different kinds of similarity captured by the clusters? [3 points]

In the first experiment, it is likely that a smaller or dependency-based word window was employed. By using verbs and dependencyvectors we can capture a local relationship between contexts. The words in the resulting cluster share an element of context substitutability - for instance, the words *carriage, bike, train* play similar semantic roles / belong to the same semantic category, and in that sense, could be substituted for each other.

In the second experiment, it is likely that a larger window was used, resulting instead in clusters of words that belong to the same topic - These words are still related in a real-word sense (semantic relatedness), but they are not semantically similar insofar that these words have different semantic roles and cannot be substituted for another.

Put differently, *car* is **similar** to *vehicle*, but only **related** to *wheel* and *steering*.

(e) Outline how techniques from distributional semantics can be used in conjunction with a syntactic parser to help disambiguate prepositional phrase attachment ambiguities [5 points]

Structural ambiguity can occur in the process of sentence parsing when more than one syntactic structure can be assigned to a sentence. Prepositional phrase attachment ambiguity is a particularly difficult problem within this domain.

For instance, the following sentence has two possible syntactic structures, and is therefore syntactically ambiguous:

Eating soup with a spoon

The prepositional phrase *with a spoon* could be attached to the verb *eating*, but it could also be attached to the noun *soup* - the latter interpretation would be invalid as it would suggest the spoon is eaten too.

PP Disambiguation is typically attempted through application of for instance probabilistic rules (what is the most probable parse tree out of all possible parse trees?), or by incorporating other lexical or semantic information to assign the most appropriate syntactic struce.

Techniques from distributional semantics, in particular word embeddings, might be of use in overcoming this ambiguity, insofar that word embeddings have shown to capture the different kinds of features that are typically used when dealing with structural ambiguity. It might therefore be possible to resolve structural ambiguity by supplementing traditional methods with word embeddings. For instance, embeddings that capture the local dependency structure of lexical items might be of use to resolve prepositional phrase attachment ambiguities.

(f) The original skip-gram model learns dense word representations, i.e. word embeddings, by predicting a distribution of possible contexts for a given word. It thus treats the task as multiclass classification and uses the softmax function in its training objective. What problem arises with this model when it is being trained on a large text corpus and why?

When the softmax function is used, the denominator term implies iterating over the entire vocabulary, which can be computationally expensive / inefficient:

$$p(w_k|w_j) = \frac{e^{c_k \cdot v_j}}{\sum_{i \in V} e^{c_i \cdot v_j}}$$

This is especially problematic when training on a very large corpus.

(g) How does skip-gram with negative sampling address the above problem? Briefly describe the intuition behind skip-gram with negative sampling, giving the formula for its training objective. [1 point]

Rather than iterating over the whole vocabulary, negative sampling employs contrastive estimation, by choosing a small subset of k noise samples or negative samples (dissimilar words) which are not like the target word, effectively serving as a computationally cheap proxy for the whole vocabulary.

Doing this reconstructs the training objective into a binary, or positive/negative classification problem, as the objective in training now revolves around making the target word like the context words, and not like the k negative words / noise, which leads to the following expression:

$$\arg\max \sum_{(w_j, w_k) \in D_+} \log \frac{1}{1 + e^{-c_k \cdot v_j}} + \sum_{(w_j, w_k) \in D_-} \log \frac{1}{1 + e^{c_k \cdot v_j}}$$

Where for the first term, we want to maximize the probability of the word context pairs being positive, and for the second term, we maximize the probability of the word context pairs being negative.

(h) Skip-gram word embeddings have the following properties:

(i) They capture similarity in word meaning. The following examples show words most similar to greenish and poured, according to the skip-gram model.

greenish	poured
bluish	sipped
reddish	simmered
pinkish	boiled
brownish	spilled
grayish	splashed
silvery	drained
whitish	drank

What aspects of word meaning do skip-gram word embeddings capture, as demonstrated by these examples? Describe two different aspects.

It appears that in this instance, skip-gram word embeddings managed to capture two aspects of meaning: semantic similarity and morphological similarity.

In the first column, the words all represent colors, making them semantically similar. They are also morphologically similar in that they all specifically denote a faint resemblance/hue of color, denoted by the shared suffix *-ish*.

In the second column for *poured*, all words are about various actions involving liquids (which semantically are loosely equivalent), but only in the past tense, as denoted by the shared past tense morpheme *ed.*

In both instances, the skip-gram word embeddings manages to capture both semantic and morphological similarities between words.

Another way to interpret this is that skip-gram word embeddings can capture not only similarity in terms of meaning, but also recognize how different forms of a word belong in different contexts.

Skip-gram word embeddings also capture analogy, as demonstrated below. The underlined words are automatically selected by the model; other words are provided as input.

Relationship	The discovered analogy
woman – queen	man: king
France – Paris	Italy: Rome
Einstein – scientist	Picasso: painter
run – ran	look: looked

The following examples show some of the system errors in the analogy task:

Relationship	The discovered analogy
quick – quicker	small: larger
smart – smarter	hot: colder

Explain briefly why these errors arise [2 points]

Word embedding techniques can capture analogy by means of vector offsets: when given the task: “a is to b, as c is to d”, and only the words a, b and c or known, d can be approximated by calculating the word with the highest cosine similarity to c, given the offset between a and b.

In the first column of the second table, the words denotes a comparative relationship, whereas its corresponding discovered analogy denotes an antonymical relationship.

In distributional semantics, it is assumed that the meaning of a word can be derived from the context it appears in. For some reason, it appears that in everyday life/written text, we use a word and its antonym in highly comparable contexts. Resultantly, both terms will end up approximately neighbouring each other in semantic space, and it can therefore occur that the discovered “analogy” is of antonymic nature rather than comparative, as is the case in table 2.