Report

2021114004

1 Negative Sampling

Negative sampling is a technique used in word embedding models to improve the efficiency of training. The goal of negative sampling is to simplify the computation required to train the model by reducing the number of computations required to update the weight parameters.

In word2vec, the training process involves predicting a word given its context words, or vice versa. This is done by computing the probability of observing a context word given a target word, or the other way around. The traditional approach to training word2vec involves computing this probability for all words in the vocabulary, which can be computationally expensive.

Negative sampling simplifies this process by sampling a small number of "negative" examples (words that do not appear in the context of the target word) for each training example. Instead of computing the probability for all words in the vocabulary, the model only needs to compute the probability for a small number of words. The number of negative examples is typically much smaller than the size of the vocabulary, which makes the training process much more efficient.

To approximate the word2vec training computation using negative sampling, we modify the objective function that the model is optimizing. The original objective function involves computing the softmax function over the entire vocabulary, which is computationally expensive. Instead, we use negative sampling to approximate the objective function. The new objective function involves computing a sigmoid function for the positive example (the observed context word) and a sigmoid function for each negative example. The negative examples are sampled randomly from a noise distribution that is used to estimate the probability of a word being a context word. By minimizing the difference between the predicted probability and the observed

probability, the model learns to predict the context words for each target word.

Overall, negative sampling is a useful technique for training word embedding models like word2vec because it simplifies the training process and makes it more efficient.

2 Semantic Similarity

Semantic similarity refers to the extent to which two words or phrases have similar meanings. It is an important concept in natural language processing and can be measured using word embeddings, which are vector representations of words in a high-dimensional space.

Word embeddings capture the semantic meaning of words by representing them as points in a vector space, where the distance between points reflects the semantic similarity between the corresponding words. This allows us to measure the semantic similarity between two words by computing the distance between their corresponding vectors in the embedding space.

Two common techniques for measuring semantic similarity using word embeddings are:

1. Cosine similarity
2. Euclidean distance

```
print(get_top_words("water"))
print(get_top_words("running"))
print(get_top_words("great"))
print(get_top_words("again"))
print(get_top_words("worst"))
```

```
['water', 'spilled', 'dripping', 'sweat', 'gallons', 'guts', 'smell', 'gore', 'splattering', 'shed']
['running', 'machine', 'waste', 'run', 'consuming', 'period', 'twilight', 'transported', 'unit', 'concrete']
['great', 'wonderful', 'fantastic', 'fabulous', 'directing', 'superb', 'masterful', 'capturing', 'fine', 'remarkable
']
['again', 'recreated', 'cheers', '1941', 'againthe', 'mute', 'operation', 'renewed', 'twice', 'eager']
['worst', 'scariest', 'best', 'blah', 'lastly', 'rendered', 'trimmed', 'jerk', 'restraint', 'struggled']
```

Figure 1: Top 10 similar words for 5 through SVD

Figure 2: Top 10 words for 'titanic' through an offtheshelf embedding