Similarity and Dissimilarity with Custom and Pre-trained FastText Models

1- Data Preprocessing:

Before delving into the specifics of the custom and pretrained models and their respective results, it's crucial to discuss the preprocessing steps applied to the Yelp dataset's tips section, even for the training data or the test data. These preprocessing steps play a pivotal role in shaping the quality and effectiveness of the subsequent models. Here's an overview of the preprocessing pipeline: (Note that we only worked on the text attribute of the tips' section)

- Tokenization
- Converting all words to lower case
- Lemmatization
- Removing digits and punctuation
- Filtering out short words

Now, let's delve into the specifics of the custom and pretrained models and their respective results.

1- Custom Model:

In this section, we discuss the training and evaluation of a custom FastText model on the Yelp dataset's tips section. The model was trained on 10,000 samples of the text attribute using the Gensim library. Here's a breakdown of the key aspects:

- Model's Parameters:

  - Window Size = 3
  - Vector Size = 50
  - Minimum Count = 3

- Evaluation:

  The model was tested on 50 random words chosen from the dataset that it hasn't trained on, to determine the top 10 similar words and the top 10 opposite words for each word

- Evaluation Results:

The model's performance wasn't impressive but somehow satisfactory, as it tended to determine similarity based on letter patterns rather than semantic meaning. For instance, these are the results of the word "say":
"The top 10 similar words for the word (say):

may
pay
yay
saturday
close
today
weekday
tuesday
everyday
lay"

As you can see, the word "say" was associated with similar words based on letter similarity rather than context.

2- Pre-trained Model:

In this section, we discuss evaluation of the Facebook pretrained FastText model on the Yelp dataset's tips section. The model was evaluated on the same 50 random words chosen from the text attribute of the dataset. Here's a breakdown of the key aspects:

- Evaluation Results:

  The pretrained model demonstrated significantly impressive results compared to the custom model. It identified similar words based on semantic meaning rather than mere letter patterns. For example, these are the results of the word "great":

  "The Top 10 Similar Words for The Word 'great':
  fantastic
  terrific
  wonderful
  good
  geat
  excellent
  amazing
  tremendous
  nice
  awesome"

As you can see, the word “great” was associated with synonyms that reflected it’s positive connotations.

3- Opposite Words:

While both models struggled to identify accurate opposite words, the pretrained model's attempts were notably closer to the expected opposites due to it’s focus on semantic context.

4- Conclusion:

The comparison between the custom and the pretrained model underscores the importance of semantic understanding in word embeddings. While the custom model provided satisfactory results, its limitations in capturing semantic meaning were evident. On the other hand, the pretrained model, leveraging a vast corpus of text data, excelled in semantic representation, leading to more accurate associations between words.