**Project Report: Word Analogy Finder using Pre-trained Word Embedding Models**

Abstract:

We, therefore, focus on the project for pre-trained word embeddings applied particularly on solving word analogy problems: FastText and Word2Vec. Word analogies include finding a word which can complete the analogy such that the semantic relationship for all the words is properly given. We will thereby establish the ability of such models to capture complex relationships and test their performance in an interactive environment.

Introduction

Word embeddings is one of the most fundamental tools for natural language processing (NLP) techniques today, in which words are presented as dense vectors in continuous vector space. These capture several semantic properties of words like similarity, analogy, and contextual relations. Both models adopted for this project, namely Word2Vec and FastText, share the concept of neural networks but handle word representations in different ways.

• Word2Vec (developed by Mikolov et al.) generates word vectors based on surrounding context and has been widely used for word similarity and analogy tasks.

• FastText (developed by Facebook) extends Word2Vec by representing words as bags of character n-grams, which helps improve the handling of out-of-vocabulary (OOV) words, especially rare or morphologically complex words.

This project aims to develop a tool that leverages these pre-trained models to find analogies, thus shedding light on how word embeddings capture semantic relationships between words.

Prior Related Work:

Extensive work has been done on word embeddings. Many tasks such as word similarity and analogy detection have been focused on word embeddings. Word2Vec was one of the earliest models that popularized the concept of word embeddings. Mikolov et al. (2013) presented the Skip-gram and Continuous Bag of Words (CBOW) models, designed to predict a word based on its context or vice versa. These models have proven to capture meaningful relationships, such as analogies: "king" - "man" + "woman" = "queen".

FastText, introduced by Facebook, takes a step forward in representing words not only as whole units but also as sequences of character n-grams. This allows FastText to handle rare or unseen words better than models like Word2Vec, which heavily rely on exact word matches.

Both of the models have been used on tasks such as machine translation, sentiment analysis, and document classification and show their efficiency in capturing the rich semantics of words.

Dataset:

The models employed in this project were pre-trained on large corpora:

•\tWord2Vec was trained on a humongous dataset of Google News articles with about 100 billion words, and the resulting embeddings represent semantic relationships based on context.

•FastText was trained on Wikipedia data and includes subword (n-gram) information, allowing it to represent out-of-vocabulary words based on their subword components.

These pre-trained models are available via the Gensim library, making it easy to load and use them for various NLP tasks without needing to train them from scratch.

Methodology:

The methodology for this project can be broken down into several key steps:

1. Model Loading: We use Gensim's API to load pre-trained models such as FastText and Word2Vec. Gensim provides an easy interface to work with these models and allows for efficient similarity computations.

2  fasttext\_model = load\_model('fasttext-wiki-news-subwords-300')

3  word2vec\_model = load\_model('word2vec-google-news-300')

4.\tFinding Analogies: The core activity in this assignment is word analogy solving. For an analogy in the form of "A : B :: C :?", we compute the most similar word to C while subtracting the influence of A and adding the influence of B. This can be done using Gensim's most\_similar() method that returns words which are most similar in the vector space.

5. Output = model.most\_similar(positive=[word\_b, word\_c], negative=[word\_a], topn=1)

For example in the analogy "king:man::queen:? : the function will calculate vector transformation from "king to "man and then, apply that transformation on the "queen" to guess the most likely word to fill the blank.

6.\tTesting the Models: We test the analogy function on a set of sample analogies, such as well-known examples: "king : man :: queen : woman". This is a preliminary validation of the models' performance.

7.\tword\_a, word\_b, word\_c = 'king', 'man', 'woman'

8.\t Interactive Testing: One of the functionalities of the project is the interactive testing of word analogies. Users can enter any three words (A, B, C) and the system returns the word that completes the analogy. This provides room for flexibility and engagement as users explore the relationships among words.

9.\txperiment\_ft = find\_analogies(fasttext\_model, word\_a, word\_b, word\_c)

Experimentations:

The system was tested on many word analogies

1. "king : man :: queen :?"

2. "paris : france :: london :?"

3. "cat : kitten :: dog : ?"

The models were expected to return the correct results:

• Word2Vec and FastText both correctly identified "woman" as the missing word for "king : man :: queen : ?".

• The interactive testing allowed for additional tests, where users could input different analogies and receive the results in real-time.

Results:

The results from testing both models were generally consistent:

•\tFor the analogy "king : man :: queen :?", both Word2Vec and FastText correctly predicted "woman".

•\tSimilarly, for "paris : france :: london :?", both models returned "england".

•\tThe interactive tool allowed the user to test a wide range of analogies and observe the models' predictions.

The results confirm that both Word2Vec and FastText are highly capable of solving word analogies and capturing semantic relationships.

Analysis:

The FastText model performed better than Word2Vec at handling rare or unseen words because it employed subword information. For instance, FastText could discover analogies of uncommon words or words that are rarely spelled, which Word2Vec could not do sometimes.

Both models performed exceptionally well with common words and standard analogies. This points out that pre-trained word embeddings perform very well on typical NLP tasks, especially in word similarity and analogy.

Conclusion:

This project shows the strength of FastText and Word2Vec in solving word analogy problems by demonstrating their ability to capture semantic relationships between words. Both models performed well, but with a slight edge for FastText when handling rare words. The interactive analogy testing feature is a great way to keep users engaged with word relationships, making it a very useful tool for educational and research purposes in NLP.

Conclusion Pre-trained word embeddings such as Word2Vec and FastText are really very effective in all the ranges of semantic tasks. These also bring ease to the user while using, making it worth for the developers and researchers in NLP fields.