Word Vectorization

Task:

Analysis: Compare and analyze which of the two word vectorizing methods performs better by using performance metrics such as accuracy, F1 score, precision, recall, and the confusion matrix on both the train and test sets. Write a detailed report on why one technique might perform better than the other. Also, include the possible shortcomings of both techniques (SVD andWord2Vec).

SVD with window size = 1

Train Metrics:
Train Accuracy: 0.7618
Train Precision: 0.7658
Train Recall: 0.7617
Train F1 Score: 0.7608

Train Confusion Matrix:
[[20998 3610 2205 3187]
[1325 26513 558 1604]
[1709 1675 21164 5452]
[1531 2546 3188 22735]]

Test Metrics:
Test Accuracy: 0.7454
Test Precision: 0.7504
Test Recall: 0.7454
Test F1 Score: 0.7441

Test Confusion Matrix:
[[1289 256 137 218]
[103 1667 30 100]
[122 119 1294 365]
[74 183 228 1415]]

SVD with window size = 2

Train Metrics:
Train Accuracy: 0.7939
Train Precision: 0.8048
Train Recall: 0.7939
Train F1 Score: 0.7955

Train Confusion Matrix:
[[24237 1241 1825 2697]
[2569 24138 564 2729]
[1965 367 21650 6018]
[2116 525 2117 25242]]

Test Metrics:
Test Accuracy: 0.7736
Test Precision: 0.7848
Test Recall: 0.7736
Test F1 Score: 0.7748

Test Confusion Matrix:
[[1491 92 133 184]
 [190 1497 39 174]
 [142 33 1303 422]
 [118 45 149 1588]]

SVD with window size = 3

Train Metrics:
Train Accuracy: 0.7980
Train Precision: 0.8019
Train Recall: 0.7980
Train F1 Score: 0.7987

Train Confusion Matrix:
[[22560 2040 2558 2842]
 [1258 25829 981 1932]
 [1379 646 23478 4497]
 [1459 1018 3627 23896]]

Test Metrics:
Test Accuracy: 0.7812
Test Precision: 0.7847
Test Recall: 0.7812
Test F1 Score: 0.7817

Test Confusion Matrix:

[[1390 146 163 201]
 [98 1630 60 112]
 [110 45 1430 315]
 [84 76 253 1487]]

Skip-gram Results

Skipgram with window size = 1

Train Metrics:

Train Accuracy: 0.9839
Train Precision: 0.9839
Train Recall: 0.9839
Train F1 Score: 0.9839

Train Confusion Matrix:
[[29432 256 207 105]
[35 29888 39 38]
[145 17 29430 408]
[169 29 481 29321]]

Test Metrics:
Test Accuracy: **0.8639**
Test Precision: 0.8641
Test Recall: 0.8639
Test F1 Score: 0.8639

Test Confusion Matrix:
[[1642 77 103 78]
[50 1779 43 28]
[94 26 1589 191]
[86 37 221 1556]]

Skipgram with window size = 2

Train Metrics:
Train Accuracy: 0.9826
Train Precision: 0.9826
Train Recall: 0.9826
Train F1 Score: 0.9826
Train Confusion Matrix:
[[29477 250 178 95]
[34 29938 8 20]
[167 44 29469 320]
[193 91 688 29028]]

Test Metrics:
Test Accuracy: **0.8589**
Test Precision: 0.8584
Test Recall: 0.8589
Test F1 Score: 0.8584

Test Confusion Matrix:
[[1655 77 90 78]
[43 1793 31 33]
[99 30 1586 185]
[122 52 232 1494]]

Skip-gram with window size = 5

Train Metrics:
Train Accuracy: 0.9838
Train Precision: 0.9839
Train Recall: 0.9837
Train F1 Score: 0.9837

Train Confusion Matrix:
[[29516 160 171 153]
[94 29845 18 43]
[139 32 28989 840]
[100 29 171 29700]]

Test Metrics:
Test Accuracy: **0.8637**
Test Precision: 0.8643
Test Recall: 0.8637
Test F1 Score: 0.8637

Test Confusion Matrix:
[[1635 78 94 93]
[78 1757 25 40]
[109 23 1536 232]
[89 35 140 1636]]

report on why one technique might perform better than the other. Also, include the possible shortcomings of both techniques (SVD andWord2Vec)

1. Introduction: Word vectorization is a fundamental step in natural language processing (NLP), enabling algorithms to process textual data effectively. This report investigates why Word2Vec might outperform Singular Value Decomposition (SVD) for word vectorization tasks, considering their strengths, weaknesses, and performance on various metrics.

2. Overview of Word Vectorization Techniques:

- **Singular Value Decomposition (SVD)**: SVD is a linear algebra technique used for dimensionality reduction. It decomposes a matrix into its constituent parts to capture

underlying patterns. In word vectorization, SVD is applied to co-occurrence matrices to generate word embeddings.

- **Word2Vec**: Word2Vec is a neural network-based technique that learns distributed representations of words in a continuous vector space. It captures semantic relationships between words based on their contexts in a large corpus.

3. Performance Metrics Comparison:

1. **Accuracy**: This metric indicates the overall correctness of the model's predictions. It's calculated as the ratio of correctly classified instances to the total instances in the dataset.

   Accuracy=Number of Correct Predictions/Total Number of Predictions

2. **Precision**: Precision measures the proportion of correctly predicted positive instances out of all instances predicted as positive. It helps in understanding the model's ability to avoid false positives.

   Precision=True Positives/True Positives+False Positives

3. **Recall**: Recall calculates the proportion of correctly predicted positive instances out of all actual positive instances. It gives insight into the model's ability to capture all positive instances without missing any.

   Recall=True PositivesTrue/ Positives+False Negatives

4. **F1 Score**: The F1 score is the harmonic mean of precision and recall, providing a balance between the two metrics. It's particularly useful when dealing with imbalanced datasets.

   F1 Score=2×Precision×Recall/Precision+Recall

5. **Confusion Matrix**: A confusion matrix is a table that visualizes the model's performance by displaying the number of true positives, true negatives, false positives, and false negatives.

1. Accuracy:

Accuracy measures the overall correctness of the classification model. It is calculated as the ratio of correctly predicted instances to the total instances.

- **Skipgram:** Across all window sizes (1, 2, and 5), Skipgram consistently achieves higher accuracy compared to SVD on both the training and test sets. This indicates that Skipgram is better at correctly classifying instances into their respective categories.
- **SVD:** While SVD shows decent accuracy, it falls short compared to Skipgram in all cases.

2. F1 Score:

F1 score is the harmonic mean of precision and recall. It provides a balance between precision and recall, making it a useful metric for imbalanced datasets.

- **Skipgram:** Skipgram demonstrates higher F1 scores across all window sizes, indicating a better balance between precision and recall compared to SVD.
- **SVD:** SVD achieves lower F1 scores compared to Skipgram, suggesting that it may have imbalances in precision and recall or is not performing as well in capturing the true positives and true negatives.

3. Precision and Recall:

Precision measures the ratio of true positives to the sum of true positives and false positives. Recall measures the ratio of true positives to the sum of true positives and false negatives.

- **Skipgram:** Skipgram consistently shows higher precision and recall compared to SVD across different window sizes, indicating its ability to better discriminate between classes and capture relevant instances.
- **SVD:** While SVD shows reasonable precision and recall values, they are generally lower than those of Skipgram, indicating its limitations in accurately identifying true positives and avoiding false positives.

4. Confusion Matrix:

A confusion matrix provides a detailed breakdown of the model's performance by showing the actual and predicted classes for each category.

- **Skipgram:** The confusion matrices for Skipgram generally exhibit lower misclassification rates across different classes, suggesting its superior performance in correctly classifying instances.
- **SVD:** SVD's confusion matrices indicate higher misclassification rates compared to Skipgram, implying its inferior performance in accurately predicting class labels.

Conclusion:

In summary, Skipgram consistently outperforms SVD across various performance metrics, including accuracy, F1 score, precision, recall, and the confusion matrix, for word vectorization tasks. Skipgram's ability to capture more nuanced relationships between words within different window sizes likely contributes to its superior performance. Therefore, based on the provided metrics, Skipgram emerges as the preferred word vectorizing method in this scenario.

4. Reasons for Word2Vec's Superiority: Several factors contribute to Word2Vec's superior performance compared to SVD:

- **Semantic Richness**: Word2Vec captures richer semantic information compared to SVD. By considering the context of words in sentences, Word2Vec can capture nuanced semantic relationships and similarities between words. This leads to more meaningful word embeddings that better represent the underlying semantics of the language.

- **Contextual Understanding**: Word2Vec models the contextual usage of words, allowing it to capture subtle semantic nuances. In contrast, SVD treats each word independently, potentially missing important contextual information. Word2Vec's ability to understand the context in which words appear enables it to generate more accurate and contextually relevant word embeddings.
- **Non-linearity**: Word2Vec employs non-linear activation functions and neural network architectures, enabling it to capture complex relationships between words. In contrast, SVD relies on linear transformations, which may limit its ability to capture non-linear relationships present in language data. The non-linear nature of Word2Vec allows it to learn more intricate patterns and associations between words, leading to more expressive word embeddings.
- **Efficient Training**: Word2Vec models can be efficiently trained on large corpora using techniques such as stochastic gradient descent. In contrast, SVD may face computational challenges when dealing with large datasets due to its computational complexity. Word2Vec's scalability and efficiency make it well-suited for training on massive datasets, allowing it to capture a diverse range of linguistic patterns and relationships.

5. Possible Shortcomings of SVD and Word2Vec:

- **SVD Shortcomings**:
    - **Limited Semantic Understanding**: SVD may struggle to capture the complex semantic relationships and nuances present in language data. Its linear nature and reliance on matrix factorization may result in less expressive word embeddings compared to Word2Vec.
    - **Difficulty with Contextual Understanding**: SVD treats each word independently and does not explicitly model the contextual usage of words. As a result, it may fail to capture the contextual variations in word meanings, leading to less accurate word embeddings.
- **Word2Vec Shortcomings**:
    - **Dependency on Pre-trained Models**: Word2Vec often relies on pre-trained models, which may not be optimal for specific domains or languages. Fine-tuning pre-trained models or training from scratch may require substantial computational resources.
    - **Handling Out-of-vocabulary Words**: Word2Vec may struggle with out-of-vocabulary words that are not present in the training corpus. Strategies such as subword embeddings or handling unknown words explicitly may be needed to address this limitation.

Hyperparameter tuning report

To experiment with different context window sizes for both SVD and Skipgram word vectors in a classification task, we have considered three different window sizes: 1, 2, and 3 for SVD, and 1, 2, and 5 for Skipgram. Since we are working on a news multi-class classification task, where understanding the broader semantic associations between words might be crucial for accurately categorizing news articles, we can adjust our rationale for choosing the context window sizes.

SVD:

1. **Window Size = 1:**
    - **Rationale:** Captures immediate neighboring words, which can be beneficial for identifying specific syntactic patterns within news articles, such as key phrases or expressions. This size might be useful for tasks like identifying entities or extracting keywords from headlines.
2. **Window Size = 2:**
    - **Rationale:** Provides a slightly broader context, allowing the model to capture relationships between words that occur in close proximity. This size may help capture more nuanced semantic associations within sentences or short paragraphs, which are common in news articles.
3. **Window Size = 3:**
    - **Rationale:** Offers an even broader context, enabling the model to capture relationships between words that span across multiple sentences or paragraphs. This size might be particularly useful for identifying themes or topics that are spread throughout a news article, providing a more comprehensive understanding of its content.

Skipgram:

1. **Window Size = 1:**
    - **Rationale:** Focuses on immediate neighboring words, while still preserving important semantic information. This size is suitable for identifying word associations within short segments of text, such as headlines or brief descriptions.
2. **Window Size = 2:**
    - **Rationale:** Strikes a balance between capturing local and broader context, making it suitable for capturing relationships between words within sentences or short paragraphs. This size might be effective for identifying related terms or phrases within the same context.
3. **Window Size = 5:**
    - **Rationale:** Explores a larger context window, which can help capture broader semantic associations between words that span across multiple sentences or

paragraphs. This size might be beneficial for identifying overarching themes or topics within news articles, providing a more holistic understanding of their content.

Below are the performance metrics for all three context window configurations:

SVD:

1. **Window Size = 1:**
    - Test Accuracy: 0.7454
    - Test Precision: 0.7504
    - Test Recall: 0.7454
    - Test F1 Score: 0.7441
2. **Window Size = 2:**
    - Test Accuracy: 0.7736
    - Test Precision: 0.7848
    - Test Recall: 0.7736
    - Test F1 Score: 0.7748
3. **Window Size = 3:**
    - Test Accuracy: 0.7812
    - Test Precision: 0.7847
    - Test Recall: 0.7812
    - Test F1 Score: 0.7817

Skipgram:

1. **Window Size = 1:**
    - Test Accuracy: 0.8639
    - Test Precision: 0.8641
    - Test Recall: 0.8639
    - Test F1 Score: 0.8639

2. **Window Size = 2:**
    - Test Accuracy: 0.8589
    - Test Precision: 0.8584
    - Test Recall: 0.8589
    - Test F1 Score: 0.8584
3. **Window Size = 5:**
    - Test Accuracy: 0.8637
    - Test Precision: 0.8643
    - Test Recall: 0.8637
    - Test F1 Score: 0.8637

Best Configuration:

For SVD, the configuration with a window size of 3 performs the best in terms of test accuracy, precision, recall, and F1 score. This suggests that for this specific classification task, a slightly larger context window provided by window size 3 helps the model capture more relevant syntactic and semantic information, resulting in improved performance.

For Skipgram, both window sizes 1 and 5 yield very similar and excellent results. Either of these window sizes can be considered as providing the best performance, indicating that Skipgram is less sensitive to the choice of context window size compared to SVD in this scenario.

Possible Reasons:

- **SVD (Window Size = 3):** A larger context window allows SVD to capture more semantic information and relationships between words, leading to better classification performance.
- **Skipgram (Window Size = 1 or 5):** Skipgram inherently captures semantic relationships between words by design, so it performs well with both smaller and larger context windows. The specific choice between window sizes 1 and 5 may not significantly impact performance due to Skipgram's ability to learn from both local and global context.