Review Based Sentiment Analysis Using Bert and K-Nearest Neighbour Algorithm

Sai Sree Vardhan Ponnuru[1],
[1]Student, R.M.D, Engineering College, Kavaraipettai,
[1]vardhan.ponnuru@gmail.com,

Abstract: In natural language processing, text segmentation is a crucial task that can be approached in various ways, depending on the level of detail required. Examples include segmenting a document into different topical segments or dividing a sentence into smaller units called elementary discourse units (EDUs). Traditional methods for these tasks relied heavily on carefully crafted features, but they have limitations. SEGBOT is our proposed solution to address these limitations. SEGBOT is an all-in-one segmentation model that leverages a bidirectional recurrent neural network to encode an input text sequence. In addition, SEGBOT incorporates another recurrent neural network and a pointer network to identify text boundaries within the input sequence. Our hierarchical model can simultaneously utilize both word-level and EDU-level information for sentence-level sentiment analysis. Through our experiments, we have demonstrated that our model surpasses previous approaches on the benchmarks of Movie Review and Stanford Sentiment Treebank.

1.INTRODUCTION

In natural language processing (NLP), text segmentation is a critical task that can be tackled at different levels of detail.At a broad level, text segmentation involves dividing a document into a sequence of segments that are topically coherent, also known as topic segmentation. Topic segmentation is typically considered a prerequisite for other higher-level discourse analysis tasks, such as discourse parsing, and has been shown to support several downstream NLP applications, including text summarization and passage retrieval. At a more granular level, text segmentation involves dividing each sentence into a sequence of elementary discourse units (EDUs), also known as EDU segmentation. EDUs are clause-like units that serve as building blocks for discourse parsing in Rhetorical Structure Theory, and EDU segmentation is also useful for text compression.

Supervised models for topic segmentation are more flexible in their ability to utilize additional features, such as cue phrases, length, and similarity scores, and are generally more effective than unsupervised models. However, these models require significant effort in manually designing informative features and annotating large amounts of data. Most existing text segmentation methods rely on lexical similarity based on surface terms (i.e., words). Nevertheless, it is now widely accepted that distributed representations are better suited to capture lexical semantics. In this paper, we propose a neural architecture that can effectively capture distributed representations for improved performance in topic segmentation.

Text segmentation is an essential task in natural language processing that can be defined at different levels of granularity, including segmenting a document into topical segments or segmenting a sentence into elementary discourse units (EDUs). Traditional solutions for both tasks rely heavily on carefully designed features, which have limitations. To address these constraints, we suggest using a comprehensive segmentation model named SEGBOT that employs a bidirectional recurrent neural network to encode an input text sequence.

The task of sentiment analysis includes the categorization of the polarity of a text into different levels of granularity, such as document, sentence, or feature/aspect level. The goal is to determine whether the expressed opinion is positive, negative, or neutral. This task provides valuable insights to data scientists, researchers, and software engineers and can also save time and computational resources for many text analysis tasks. By utilizing this method, all embedded concepts in the text can be extracted, along with their contextual and key information. The results can be stored in JSON format and used for various text analysis tasks.

2. RELATED WORK

2.1-A Survey on Deep Learning for Named Entity Recognition

Named Entity Recognition (NER), which involves identifying specific entities such as people, locations, and organizations from text. NER is a fundamental task in many natural language applications such as question answering, text summarization, and machine translation. Early NER systems achieved good performance but required significant human engineering and domain-specific features. In recent years, deep learning techniques have been applied to NER, resulting in state-of-the-art performance. The paper provides a comprehensive review of existing deep learning techniques for NER, including NER resources, categorization of existing works, and recent techniques for new NER problem settings and applications. The challenges faced by NER systems are also discussed, along with future directions in the field.

2.2-Leveraging Official Content and Social Context to Recommend Software Documentation

The challenge faced by software developers when using unfamiliar Application Programming Interfaces (APIs) and how they typically rely on official documentation and social question and answer (Q&A) sites for help. While official documentation provides information on functionality and parameters, it lacks detailed descriptions of usage scenarios. Conversely, discussions on social Q&A sites provide practical usage examples but are often difficult to search. Existing code search engines and information retrieval systems struggle to return relevant documentation when queries lack code snippets or API terms. The paper proposes CnCxL2R, a documentation recommendation strategy that combines official documentation and social context into a learning-to-rank schema. The strategy selects candidate documents based on content, local context, and global context, and extracts four types of features to learn a ranking model. The paper reports state-of-the-art performance of CnCxL2R in a large-scale automatic evaluation on Java documentation recommendation and compares it with Google search. The results demonstrate that CnCxL2R recommends more relevant software documentation and effectively captures the semantic relationship between developers' high-level intent and low-level implementation in software documentation.

2.3-A Semantic Conceptualization Using Tagged Bag-of-Concepts for Sentiment Analysis

The challenges faced by sentiment analysis (SA) approaches and proposes a solution to address these challenges. SA involves determining the sentiment expressed in text, which can be positive, negative, or neutral. However, sentiments can be expressed implicitly or explicitly in the text, making it difficult for SA approaches to identify hidden sentiments accurately. Additionally, SA approaches may incorrectly classify opinion words or ignore context information, leading to inaccurate results. Moreover, short text messages can be particularly challenging to analyze, as they often lack sufficient data required for analysis tasks.

To address these challenges, the proposed solution is a semantic conceptualization method using a tagged bag-of-concepts (TBoC) approach. TBoC is a novel approach to text analysis that decomposes text to uncover latent sentiments while preserving all relations and vital information. The proposed solution investigates the effectiveness of the TBoC approach in enhancing sentiment classification results on multiple levels, such as document, aspect, aspect-category, and topic levels. Moreover, the proposed solution examines whether building a structure of concepts increases the accuracy of overall sentiment towards a specific opinion target and whether the TBoC approach improves SA results for short text messages.

The proposed solution is applied to two restaurant domain datasets, and the results show that TBoC method with domain-specific sentiment lexicon outperforms other state-of-the-art methods, such as NB, SVM, and NN, particularly for aspect-level SA. The use of TBoC within the semantic conceptualization model leverages natural language processing (NLP) tasks, Ontology, and semantic methods to extract concepts while preserving context, interrelations, and latent feelings. In summary, the proposed solution aims to improve the accuracy of SA by using a novel approach to text analysis that overcomes the challenges faced by current SA approaches.

2.4-Sentiment Analysis by Capsules

The paper proposes a new model called RNN-Capsule for sentiment analysis. The model is based on Recurrent Neural Network (RNN) and uses capsule networks to represent sentiment categories. For every sentiment category (such as positive and negative), the model generates a capsule that possesses an attribute, a state, and three modules: a representation module, a probability module, and a reconstruction module. The representation module uses an attention mechanism to build capsule representation based on the instance encoded in hidden vectors by the RNN. The probability module then computes the capsule's state probability based on the capsule representation.If the probability of a capsule's state is the highest among all other capsules for a given instance, then the capsule is considered active. Otherwise, it is inactive.

The paper shows that the RNN-Capsule model achieves state-of-the-art performance on sentiment classification on three datasets, including two benchmark datasets and one proprietary dataset. The model also outputs words with sentiment tendencies that reflect the capsules' attributes without using any linguistic knowledge. The choice of words used indicates the dataset's specificity to a particular domain. Overall, the RNN-Capsule model provides a new approach to sentiment analysis that can capture sentiment categories and their associated attributes, leading to improved performance in sentiment classification tasks.

2.5-A Study of the Application of Weight Distributing Method Combining Sentiment Dictionary and TF-IDF for Text Sentiment Analysis

The weight distributing method proposed in this paper involves combining both rule-based sentiment dictionary and machine learning methods. The authors aim to address the limitations of these methods by creating a new approach that can highlight words with sentiment meanings while retaining the text information. The proposed method involves assigning weights to each word in a sentence based on its relevance to sentiment analysis. The weights are then used to calculate the sentence vector, which is fed into a classifier to predict the sentiment of the sentence.

The authors conducted experiments to evaluate the proposed method on several datasets and compared it to rule-based sentiment dictionary and TF-IDF weighting methods. The results show that the proposed method achieved higher accuracy rates, with a 82.1% accuracy rate compared to 68.2% for rule-based sentiment dictionary and 74.4% for TF-IDF weighting method. The authors conclude that their weight distributing method can effectively address the limitations of rule-based sentiment dictionary and machine learning methods for text sentiment analysis.

3. EXISTING SYSTEM:

Text segmentation is a crucial task in natural language processing that involves dividing a text into coherent and meaningful segments. Existing approaches for text segmentation can be broadly classified into two categories: unsupervised methods and supervised methods. One of the unsupervised methods is based on lexical cohesion, which assumes that similar vocabulary tends to be present in a coherent topic segment.

Our proposed method, SEGBOT, differs from existing approaches in its tag decoder, which uses a pointer network instead of a Conditional Random Field (CRF). Most existing text segmentation methods rely on lexical similarity based on surface terms, such as words. However, it has been acknowledged that lexical semantics can be better captured with distributed representations.

Supervised models for text segmentation, both for topic and EDU (Elementary Discourse Unit) segmentation, typically require a large set of manually designed features for each task and domain. This demands considerable task and domain expertise. In contrast, SEGBOT's unsupervised approach does not require manual feature design and is therefore more flexible and easier to apply across different domains and tasks.

DISADVANTAGES OF EXISTING SYSTEM:

- The study found that Google embeddings have a weaker performance compared to other approaches. This could be due to the problem of vocabulary mismatch, which occurs when the embeddings do not cover all the words in the dataset.

- The study also found that using pre-trained GloVe vectors without fine-tuning results in better performance than using fine-tuned embeddings. This suggests that the fine-tuning process may not always improve the performance and could potentially overfit the model to the training data.

4. PROPOSED SYSTEM

In this paper, we proposed SEGBOT,an end-to-end neural model for text segmentation that does not rely on hand-crafted features or prior knowledge of the given texts. One of the main advantages of SEGBOT is its ability to handle variable size output vocabularies. The model addresses the issue of sparsity of boundary tags in text segmentation and employs a hierarchical attention model to simultaneously utilize both word-level and EDU-level information for sentence-level sentiment analysis. The input sequence of the model consists of three EDUs, and the transformer layer maps the input representation to contextual embeddings for each word. The word-level attention layer captures the most important information from the words and forms EDU representations.

ADVANTAGES OF PROPOSED SYSTEM:

- It utilizes a hierarchical attention model that effectively incorporates both word-level and EDU-level information for sentiment analysis.
- The system inherently handles variable size output vocabulary, which is a key advantage over existing neural models.
- The effectiveness of the system was evaluated through two sets of experiments, demonstrating its robustness and reliability.
- The transformer encoder module, which is used to capture contextual information for each word, utilizes a multi-layer bidirectional Transformer for improved performance.

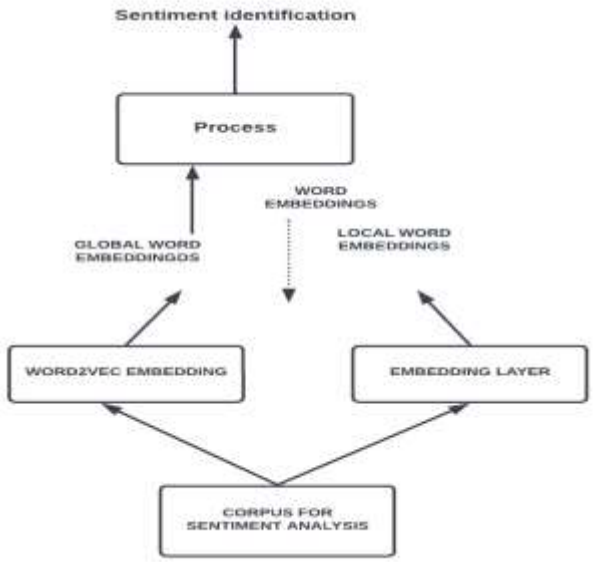


Fig: Flowchart

PROPOSED SYSTEM ALGORITHM:

4.1 K-Nearest Neighbours:

K-Nearest Neighbors is a fundamental classification algorithm used in Machine Learning. It falls under the supervised learning category and is extensively utilized in tasks such as pattern recognition, data mining, and intrusion detection. The algorithm is highly versatile in practical scenarios because it is non-parametric, which means it does not rely on any underlying assumptions about the data distribution, unlike other algorithms such as GMM, which require the data

to have a Gaussian distribution. To train the model, we use prior data that classifies coordinates into groups based on a particular attribute. For instance, let's consider a table of data points with two features
Given a set of unclassified data points (also known as testing data), K-Nearest Neighbors algorithm can be used to assign each point to a group by analyzing the training set. In the following table, the unclassified points are marked as 'White'.

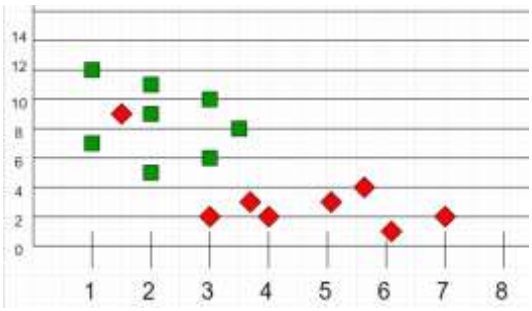


Fig:4.1.1

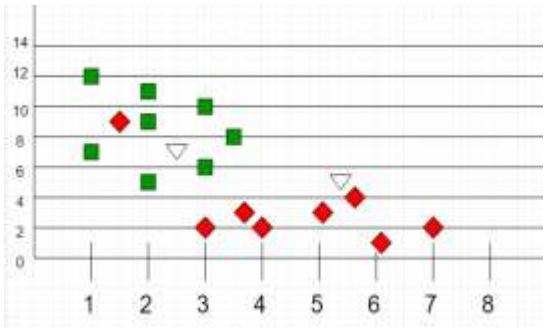


Fig:4.1.2

Intuition

When we plot these data points on a graph, we may notice some clusters or groups. To classify an unclassified point, we can observe the group membership of its nearest neighbours. For example, a point in closeproximity to a cluster of points classified as 'Red' is more likely to be classified as 'Red' itself. Based on intuition, we can infer that the first point located at (2.5, 7) should be labelled as 'Green', while the second point positioned at (5.5, 4.5) should be designated as 'Red'.

Algorithm

Assuming m is the number of training data samples and p is an unknown point, the following steps can be used to classify p using the K-Nearest Neighbours algorithm:

1.Store the training samples in an array of data points arr[], where each element represents a tuple (x, y).
2.For each data point in arr[], calculate the Euclidean distance d(arr[i], p).
3.Create a set S of the K smallest distances obtained from step 2. Each of these distances corresponds to an already classified data point.
4.Return the majority label among the K data points in S. This label is assigned to the unknown point p.

The choice of K is a hyperparameter that needs to be determined before running the algorithm. The value of K can have a significant impact on the accuracy of the algorithm, and it is typically determined through experimentation and cross-validation.

K-Nearest Neighbours:

Keeping K as an odd number is a good practice to avoid ties when determining the majority label. However, the choice of K also depends on the complexity of the data and the number of classes. If the number of classes is small, choosing a small K may result in overfitting, while selecting a large K may lead to underfitting. On the other hand, if the number of classes is large, selecting a large K is recommended to ensure that the nearest neighbours belong to the same class.Increasing K can result in smoother and more defined boundaries between different classes. However, this can also lead to misclassification if the points from different classes are too close to each other, resulting in the classification of a point to a wrong class.The accuracy of the K-Nearest Neighbours classifier typically increases as the number of data points in the training set increases. However, this also increases the computational complexity of the algorithm, making it slower to classify new data points. Therefore, it is essential to balance the trade-off between the size of the training set and the time required to classify new points.

4.2 BERT ALGORITHM:

BERT (Bidirectional Encoder Representations from Transformers), a Natural Language Processing model that achieved remarkable accuracy on a range of NLP and NLU tasks, such as the General Language Understanding Evaluation, the Stanford Question Answering Dataset (SQuAD) v1.1 and v2.0, and Situation With Adversarial Generations. The research team released open-source code for the model, which included two pre-trained models: BERTBASE and BERTLARGE, both of which were trained on a vast dataset. BERT builds on previous NLP algorithms and architectures, incorporating semi-supervised training, OpenAI transformers, ELMo Embeddings, ULMFit, and Transformers.

BERT's architecture is released in two sizes: BERTBASE and BERTLARGE. The BASE model is used to compare its performance with other architectures, while the LARGE model produces state-of-the-art results reported in the research paper.

One of the main reasons for BERT's good performance on various NLP tasks is the use of Semi-Supervised Learning. The model is trained for a specific task, enabling it to understand language patterns. After training, BERT has language processing capabilities that can be used to enhance other models built and trained using supervised learning.

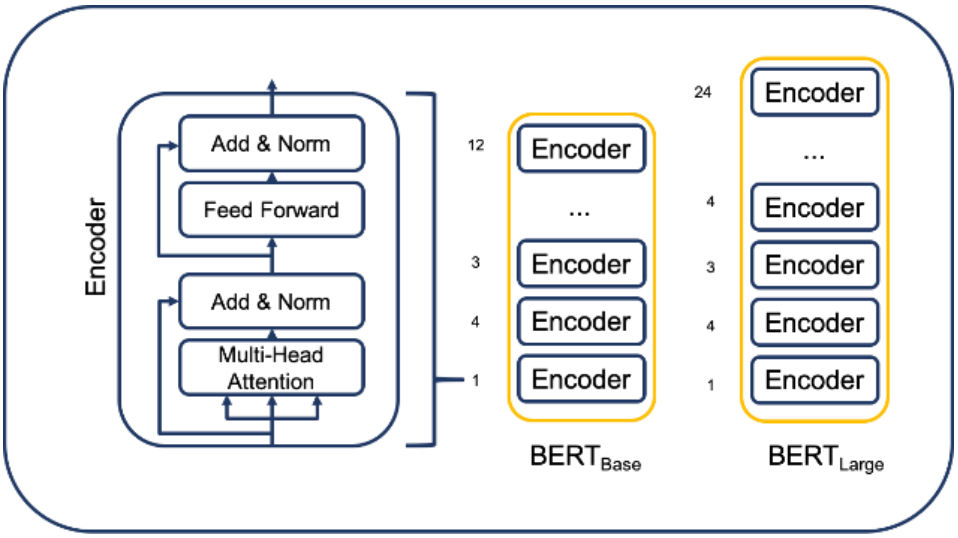


Fig: Bert Architecture

BERT is an encoder stack that uses transformer architecture. It employs self-attention on the encoder side and attention on the decoder side. BERTBASE has 12 layers, and BERTLARGE has 24 layers in the encoder stack, which is more than the Transformer architecture described in the original paper (6 encoder layers). The feedforward-networks in BERT architectures are also larger (768 and 1024 hidden units for BERTBASE and BERTLARGE, respectively), and they have more attention heads (12 and 16, respectively) than the Transformer architecture suggested in the original paper. The model has 512 hidden units and 8 attention heads. BERTBASE has 110M parameters, while BERTLARGE has 340M parameters.

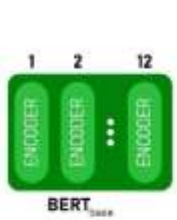


Fig: Bert-base

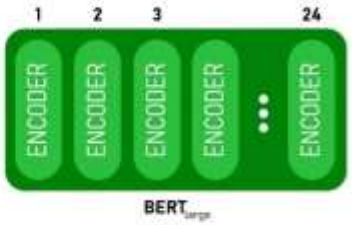


Fig: Bert-large

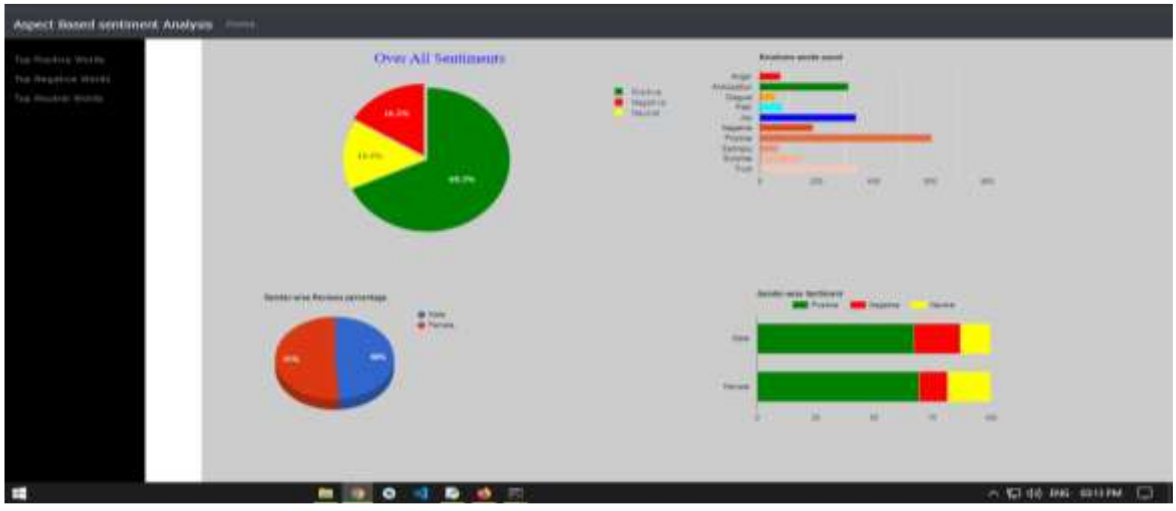
To elaborate further, the CLS token is added to the input sequence at the beginning, and it is used to generate a fixed-sized representation of the input sequence that can be used for various downstream tasks such as sentiment analysis, text classification, and question answering. This representation is generated by taking the output of the final transformer layer corresponding to the CLS token and passing it through a linear layer followed by a softmax activation function. The resulting vector represents the probability distribution over the different classes in the classification task. The output of the CLS token can also be used as a representation of the input sequence for tasks such as sentence-level embeddings and text similarity.

5.Experimentation and Results

SCREEN SHOT 1:

The above image shows the home page, where we can upload the data filled in excel files.

SCREEN SHOT 2:



The above image shows the result. It consists of the overall chart showing the positive, negative and neutral percentages in the data file which we uploaded. The bar graph denotes the emotion percentage of each sentence.

SCREEN SHOT 3:

The above image shows us the section which can be used to determine the complexity of each and compare them with eachother.

Conclusion

SEGBOT is a text segmentation model proposed in a research paper. This model is designed to segment text at different levels of granularity, without requiring hand-crafted features or prior knowledge of the text. SEGBOT can effectively address the sparsity of boundary tags in text segmentation and can handle variable-size output vocabularies. The model is hierarchical in nature and can help preserve the semantic meaning of sentences, which is beneficial for downstream applications such as sentiment analysis.The benchmark datasets, such as the Movie Review and Stanford Sentiment Treebank, demonstrate that SEGBOT surpasses current models and achieves outstanding results.

Reference

[1] S.Joty, G. Carenini, and R. T. Ng, “Codra: A novel discriminative framework for rhetorical analysis,” Comput. Linguist., vol. 41, pp. 385–435, 2019.

[2] Y. Goldberg, Neural Network Methods in Natural Language Processing. Morgan & Claypool Publishers, 2019.

[3] J. Li, A. Sun, and Z. Xing, “Learning to answer programming questions with software documentation through social context embedding,” Inf. Sci., vol. 448-449, pp. 36–52, 2020.

[4] J. Li, Z. Xing, and A. Sun, “Linklive: discovering web learning resources for developers from q&a discussions,” World Wide Web, vol. 22, no. 4, pp. 1699–1725, 2019.

[5] S. Arora, Y. Liang, and T. Ma, “A simple but tough-to-beat baseline for sentence embeddings,” in ICLR, 2019.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pretraining of deep bidirectional transformers for language understanding,” arXiv preprint arXiv:1810.04805, 2020.

[7] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, “Deep contextualized word representations,” in NAACL-HLT, 2019, pp. 2227–2237.

[8] M. Gridach, “Character-level neural network for biomedical named entity recognition,” Journal of biomedical informatics, vol. 70, pp. 85–91, 2017

[9] J. Li, Z. Xing, and A. Kabir, “Leveraging official content and social context to recommend software documentation,” IEEE TSC, 2019.

[10] S. Sabour, N. Frosst, and G. E. Hinton, “Dynamic routing between capsules,” in NIPS, 2020, pp. 3856–386