Project MONIKA: Monitoring Objects of Nearest Instance to K-Attributes

Dhominick John Billena, John Manuel Carado, Cristopher Ian Artacho, Allan Tacuel

December 8, 2023

Integrating KNN in Emotion Classification

Why is a Searching Algorithm Needed in Classification Problems?

Introduction

In the race towards sentience, people often wonder about the current state of how Artificial Intelligence (AI) perceive and interprets the idea behind emotions. It was one of the key factors towards understanding the current state of roadblocks that AI presents in mimicking how the Human Brain function. A question arise as to whether, we can simulate the idea behind how we humans perceive emotions using contextual analysis and creation of basic understanding of emotions in a simple form using Classification solutions and Algorithms with a searching algorithm that will be able to find the best possible emotion that the AI can relate to based on the given context.[8]

Current Idea of Where Emotion Classification / Sentiment Analysis is On

Even though many techniques, methods, and models have been created for emotion detection, the intricate nature of human emotions and the subtle and metaphorical ways we express them indicate that we need to go beyond traditional methods. To truly understand these complexities, we need to pay attention to the language challenges of expressing emotions when we're classifying words based on their emotional content in machine learning.

Emotional Models for Emotion Classification

It talks about two main ways to identify emotions from text: Discrete Emotion Models (DEMs) and Dimensional Emotion Models (DiEMs).DEMs are models that put emotions into specific categories. Some well-known DEMs are by Paul Ekman, Robert Plutchik, and Orthony, Clore, and Collins. Ekman and Plutchik focus on a few basic emotions, while the OCC model includes many more types of emotions.

On the other hand, DiEMs represent emotions in two or three dimensions. Examples of these models are Russell's Circumplex Model of Affect, Plutchik's Wheel of Emotions, and Russell and Mehrabian's 3-Dimensional Emotion Model. These models offer a more complex view of emotions, which is built on the idea of discrete emotions.

Smaller Set of Primary Emotions

The first models are based on the idea that there are a smaller set of primary emotions that are universal and can be used to describe all other emotions.

Paul Ekman's Model

The Paul Ekman model, which identifies six fundamental emotions: happiness, sadness, anger, disgust, surprise, and fear. These emotions are believed to originate from separate neural systems based on an individual's perception of a situation. Additionally, the combination of these emotions can give rise to complex emotions like guilt, shame, pride, lust, and greed. [2]

Robert Plutchik's Model

The Robert Plutchik model, similar to Ekman's model, posits that there are primary emotions that occur in opposite pairs and can combine to form complex emotions. Plutchik identifies eight primary emotions, including joy, sadness, trust, disgust, anger, fear, surprise, and anticipation. He also acknowledges varying degrees of intensity for each emotion, depending on how events are perceived. [1]

Orthony, Clore, and Collins Model

The Orthony, Clore, and Collins (OCC) model offers a different perspective. It disagrees with the concept of "basic emotions" but agrees that emotions are influenced by how individuals interpret events and can vary in intensity. OCC discretizes emotions into 22 categories, expanding on Ekman's basic emotions with additional classes such as relief, envy, reproach, self-reproach, and appreciation.[9]

It talks about two main ways to identify emotions from text: Discrete Emotion Models (DEMs) and Dimensional Emotion Models (DiEMs).

DEMs are models that put emotions into specific categories. Some well-known DEMs are by Paul Ekman, Robert Plutchik, and Orthony, Clore, and Collins. Ekman and Plutchik focus on a few basic emotions, while the OCC model includes many more types of emotions.

Current Emotion Detection Systems

Badugu and Suhasini used a rule-centered approach to detect emotions in tweets. They used the Sentiment140 dataset, which contains over a million tweets. After processing the tweets and tagging them with parts-of-speech, they matched the texts with emotions on the Russell's Circumplex Model of Affect. Texts that passed a validation check were assigned a sentiment score and classified into one of four emotion classes. Their model achieved an accuracy of about **85%**, but it had some limitations. It only considered the "not" form of negation and ignored other forms like "never". It also classified a large number of tweets into a limited number of classes, which could lead to inaccurate classifications. Furthermore, it ignored emojis, which can convey rich emotion content.[4]

Wikarsa and Thahir developed an app to detect the emotions of Twitter users. They collected 105 tweets using the Twitter API and preprocessed them by converting to lowercase, removing stop words, mentions, URLs, and converting emoticons into text. They used the Naive Bayes (NB) method to classify the texts into six emotions: happiness, sadness, fear, anger, surprise, and disgust. They used 10-fold cross-validation to check the accuracy of their classifications and reported an accuracy of **83%**. However, they suggested that using a larger training dataset and removing duplicate tweets could improve the system's performance. They also suggested that using other classification methods like Support Vector Machines (SVMs) and K-nearest neighbors (KNN) could enhance the performance.

Huang and his team used a four-part Episodal Memory Network (EMN) with a Self-Attention (SA) mechanism to identify emotions from text. They used the SemEval Dataset to detect four basic emotions: anger, fear, joy, and sadness. Their results showed that EMN performed better than other machine learning techniques like CNN, TLSTM, and RNN. When they combined EMN with SA, it did even better at understanding text and detecting emotions, with a precision of 65.8%, a recall of 63.5%, and an F1 score of 64.6%. However, their model could only detect a limited number of emotion categories, which could be a challenge for generalizing the model.

In relation to the previous study recommendations made by Wikarsa and Thahir, we can use a larger dataset to improve the performance of our model. We can also use other classification methods like SVMs and KNN to enhance the performance. We can also use a larger dataset to improve the performance of our model. We can also use other classification methods like SVMs and KNN to enhance the performance however,Matla and Badugu conducted a study where they compared the performance of K-Nearest Neighbors (K-NN) and Naive Bayes (NB) machine learning techniques in detecting emotions from tweets. They used the Sentiment 140 corpus for their study. Their results showed that under the same conditions, NB performed better than K-NN, achieving an accuracy of 72.06% compared to K-NN's 55.50%. [7]

To verify this results, we have tried to replicate the process presented here by Matla and Badugu. We used the same dataset and the same preprocessing techniques. We also used the same features and the same training and testing sets. However, we used the K-NN and NB classifiers from implementing them from scratch. We also used the same metrics to evaluate the performance of the classifiers speicifically on the results related to the K-NN model.

What is a Classification Problem?

Classification is a process of categorizing a given set of data into classes, It can be performed on both structured or unstructured data. The process starts with predicting the class of given data points. The classes are often referred to as target, label or categories. The classification predictive modeling is the task of approximating the mapping function from input variables to discrete output variables. The main goal is to identify which class/category the new data will fall into.

Term - Frequency Inverse Document Frequency (TF-IDF)

Term Frequency (TF) is a measure of how frequently a term occurs in a document. It is calculated as

the number of times the term appears in the document divided by the total number of terms in the document.

Inverse Document Frequency (IDF) is a measure of how important a term is. It is calculated as the logarithm of the total number of documents in the corpus divided by the number of documents where the term appears.

TF-IDF is a measure of how important a term is in a document. It is calculated as the product of the Term Frequency and the Inverse Document Frequency.

How is Term - Frequency Inverse Document Frequency (TF-IDF) used in Classification Problems?

Term - Frequency Inverse Document Frequency (TF-IDF) is used in Classification Problems to determine the importance of a term in a document. It is calculated as the product of the Term Frequency and the Inverse Document Frequency. [5] This is then used to further strenghten the accuracy of the classification algorithm by giving more weight to the words that are more important in the document. [5]

What is a Searching Algorithm?

A Search Algorithm is the modern standard for information control as the world is creating more and more information and sorting them through for retrieval is a tediuous process without the use of algorithms. Search is a fundamental process in data processing. The process involves finding a certain value in a set of values that is of the same type to prepare the data for processing. In the search process, the methods used vary depending on the methods used.

The most common search algorithms are the Linear Search and the Binary Search. The Linear Search is the most basic search algorithm that is used to find a value in a set of values. The Binary Search is a more advanced search algorithm that is used to find a value in a set of values that are sorted in a specific order. The Binary Search is more efficient than the Linear Search as it is able to find the value in a shorter amount of time.[3]

Comparision of Different Method for Searching a Word Dictionary

Searching the dictionary is a process that requires finding the exact case, characters, letters and sequence of the word in the dictionary. Among the different forms of searching algorithms, the Linear Search or Sequential Search and the Binary Search are the most common algorithms used in searching for a word in a dictionary. The Linear Search is the most basic search algorithm that is used to find a value in a set of values. The Binary Search is a more advanced search algorithm that is used to find a value in a set of values that are sorted in a specific order. The Binary Search is more efficient than the Linear Search as it is able to find the value in a shorter amount of time.[3] Given that our dataset is a set of value that is already sorted through since it was a dictionary, the Binary Search is the most efficient algorithm to use.

However, searching algorithms works mostly with the values associated with integers rather than characters right? So how do we use it in a dictionary? Well, we can use the ASCII value of the characters to compare the characters in the dictionary. The ASCII value of the characters are just integers that represent a values under the hood. See [10] for more information about ASCII values.

Linear Search

In a common use, linear search is a set of values that are not sorted in any particular order. The algorithm will start from the first value and compare it to the value that is being searched for. If the value is not found, the algorithm will move on to the next value and compare it to the value that is being searched for. The algorithm will continue to do this until the value is found or the end of the set of values is reached. If the value is found, the algorithm will return the index of the value.

Binary Search

In a common use, binary search is a set of values that are sorted in a specific order. The algorithm will start from the middle value and compare it to the value that is being searched for. If the value is not found, the algorithm will check if the value is greater than or less than the value that is being searched for. If the value is greater than the value that is being searched for, the algorithm will move on to the next value and compare it to the value that is being searched for. If the value is less than the value that is being searched for, the algorithm will move on to the previous value and compare it to the value that is being searched for. The algorithm will continue to do this until the value is found or the end of the set of values is reached. If the value is found, the algorithm will return the index of the value. If the value is not found, the algorithm will return an error that shows it was not found.[6]

Application of Searching Algorithms in Word Searching Problems

Searching algorithms are used in word searching problems. In word searching problems, the algorithm will search for a word in a set of words. The algorithm will start from the first word and compare it to the word that is being searched for. If the word is not found, the algorithm will move on to the next word and compare it to the word that is being searched for. The algorithm will continue to do this until the word is found or the end of the set of words is reached. If the word is found, the algorithm will return the index of the word. If the word is not found, the algorithm will return an error that shows it was not found.[5]

Sorting Algorithms

Sorting algorithms are used to sort a set of values in a specific order. The most common sorting algorithms are the Bubble Sort, the Insertion Sort, the Selection Sort, the Merge Sort, the Quick Sort, the Heap Sort, the Radix Sort, the Counting Sort, the Bucket Sort, and the Shell Sort. [3] In the pursuit of efficiently implementing emotional classification algorithms, the choice of sorting algorithms plays a critical role. In this section, we focus on two sorting algorithms, Merge Sort and Radix Sort, which have demonstrated their efficiency in both sequential and parallel sorting contexts especially with word sorting. [3]

Sorting Algorithm of Choice In Relation to Searching Algorithm

The stack of sorting algorithm of choice is a critical factor in the efficiency of the searching algorithm. The sorting algorithm of choice must be able to sort the set of values in a specific order. The sorting algorithm of choice must also be able to sort the set of values in a short amount of time. [3]

Merge Sort

Merge Sort is a prominent sorting algorithm known for its efficiency and stability. It is based on the divide-and-conquer approach, making it suitable for large datasets. Merge Sort divides the sequence into multiple subsequences, sorts them, and then merges them into a sorted sequence. [3] identified that, in a comparison study between sequential and parallel sorting algorithms, Merge Sort emerged as one of the fastest algorithms in its category. Stability in sorting is crucial for applications involving emotional classification, and noted that if the algorithm used for merging the sorted sequences is stable, then the entire Merge Sort algorithm remains stable. This stability ensures the preservation of order relationships within the data, an essential characteristic for emotional classification tasks.

Radix Sort

Radix Sort, on the other hand, is a non-comparative sorting algorithm that proves to be highly efficient when handling specific data types. It relies on a stable sorting algorithm to sort values by considering individual elements that compose those values, such as digits or alphabets. Radix Sort was found to be one of the fastest in its category during the comparison study of sequential and parallel sorting algorithms. In the context of emotional classification, this non-comparative sorting approach can be particularly advantageous. Highlighted that the sequential variation of Radix Sort first divides elements being sorted, such as numbers, words, or dates, into digits, and then sorts them from the least significant digit to the most significant one. This technique can be particularly useful when dealing with diverse data types and improving the overall efficiency of emotional classification algorithms.

Radix Sort Process

Radix Sort is a non-comparative sorting algorithm that sorts the elements by grouping them by individual digits or letters that share the same significant position and value. Radix Sort is a stable sort, meaning that the relative order of elements with equal values is preserved.

Application of Sorting Algorithms in Word Searching Problems

Using the sorting algorithm, we could efficiently apply the algorithm to the binary search algorithm to find the word in the dictionary. Using the index of the algorithm, we can then use the index of the sorted words in order to find the word in the dictionary and it's meaning. It is needed that the word to apply to the binary search is sorted as it relies on the concept of indexes in order to find the word in the dictionary. If the word is not sorted, the algorithm will not be able to find the word in the dictionary.

Clustering Analysis using K-Means Algorithm

Clustering is the process of dividing the entire data into groups (also known as clusters) based on the

patterns in the data. In clustering, we do not have a target to predict. We look at the data and then try to club similar observations and form different groups. Hence it is an unsupervised learning problem.

K-Means Clustering is an unsupervised learning algorithm that is used to solve clustering problems. The algorithm will start by selecting a random centroid for each cluster. The algorithm will then assign each data point to the cluster that is closest to the centroid. The algorithm will then calculate the new centroid for each cluster. The algorithm will then repeat the process until the centroids do not change.

Centroid-based Clustering

Centroid-based clustering is a clustering algorithm that is used to solve clustering problems. The algorithm will start by selecting a random centroid for each cluster. The algorithm will then assign each data point to the cluster that is closest to the centroid. The algorithm will then calculate the new centroid for each cluster. The algorithm will then repeat the process until the centroids do not change.

References

[1] Robert Plutchik. "A general psychoevolutionary theory of emotion". In: *Theories of emotion*. Elsevier, 1980, pp. 3–33.

[2] Paul Ekman et al. "Basic emotions". In: *Handbook of cognition and emotion* 98.45-60 (1999), p. 16.

[3] Darko Božidar and Tomaž Dobravec. *Comparison of parallel sorting algorithms*. Faculty of Computer and Information Science, University of Ljubljana, Slovenia, 2015.

[4] Srinivasu Badugu and Matla Suhasini. "Emotion detection on twitter data using knowledge base approach". In: *International Journal of Computer Applications* 162.10 (2017).

[5] Shahzad Qaiser and Ramsha Ali. "Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents". In: *International Journal of Computer Applications* 181 (July 2018). DOI: 10.5120/ijca2018917395.

[6] et al Riski Muhamad Fitrian Insan Taufik. *Digital Dictionary Using Binary Search Algorithm*. Journal of Physics: Conference Series, 2019.

[7] Matla Suhasini and Badugu Srinivasu. "Emotion detection framework for twitter data using supervised classifiers". In: *Data Engineering and Communication Technology: Proceedings of 3rd ICDECT-2K19*. Springer. 2020, pp. 565–576.

[8] Verma R. Nandwani P. *"A review on sentiment analysis and emotion detection from text."*. 2021. DOI: 10.1007/s13278-021-00776-6.

[9] Andrew Ortony, Gerald L Clore, and Allan Collins. *The cognitive structure of emotions*. Cambridge university press, 2022.

[10] http ://www.injosoft.se. "A. ASCII Code - The extended ASCII table". In: https://www.asciicode.com/. ().