

Data Retrieval via Natural Language Processing

Marie Cho, Changxu Liu

Institute for Computing in Research

August 1st, 2022

Abstract. Modern devices allow users to search for certain files, but solely utilize the names of the files, instead of the contents stored within. As humanity progresses technologically, the increasing amounts of data will serve as a barrier to a practical application of the latter task in terms of time and cost. This will necessitate accurate methods for finding information in a timely and effective manner. In this work, we create a system to search for relevant content within texts and images in respect to a user's queries utilizing NLP that is both productive and accurate when tested on over 300,000 sources of data.

Keywords: Deep learning, natural language processing, neural networks

1 Introduction

Data retrieval systems using deep learning have advanced drastically and been a popular and heavily researched topic for the past several years. Latest developments in natural language processing (NLP) allow incorporating contexts and complex lexical structures into data representations or embedding, which in turn enables contextual understanding in the model.

While many modern devices now implement a software to search for file names on the computer, most do not allow searches for the content within files themselves. The nature of the latter is restricted by techniques used with mass files and broad domains. Recent models attempting to successfully achieve this task are based on word frequencies and statistics rather than the semantics of the text.

NLP is used in translators, grammar and spell check, and virtual assistants.

2.2 Neural Networks

2.2.1 Recurrent Neural Networks

Recurrent neural networks are commonly used in tasks with sequential data, such as that which is found in natural language processing. Introduced in the 1980s, these are neural networks that contain a short term memory by being able to factor in the result of a previous element of a sequence into the processing the next.

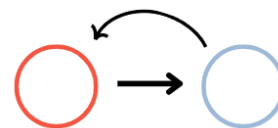


Figure 1: Recurrent Neural Networks

2 Background

2.1 Natural Language Processing

Natural Language Processing (NLP) is concerned with the ability for computers to understand human speech and text. The field of NLP began after realizing the importance of interlingual communication and hoped for an automatic translator. As technology advanced, NLP diverged into several divisions, each regarding a different aspect of language, such as the lingual interaction between humans and computers to mimic a person to person conversation. Today,

A model developed more recently in 2017 are transformers. In contrast to recurrent neural networks, which proalyze data as a whole.

2.2.2 Convolutional Neural Networks

When dealing with visual data, convolutional neural networks are frequently used to extract their features. Convolutional neural networks function by passing filters, called kernels, through an input. These kernels represent various features that may be present in the data.

By performing matrix multiplication, activation values are created to represent how well different sections of an image match up with the kernel, and the feature it represents. The results may be passed through multiple layers to find increasingly sophisticated features.



Figure 2: Convolutional Neural Networks

2.2.3 Computer Vision

Computer vision (CV) is concerned with the ability for computers to understand and process visual data. The field of computer vision started with the detection of basic edges and geometric shapes, but has expanded to fill many roles in society. Applications of this technology include self driving vehicles, facial recognition, optical character recognition, and other visual tasks.

2.3 The BERT Model

A transformer model is a neural network that adopts self-attention, the ability to weigh the significance of a part of a data. For example, in the sentence "The apple is red," "The" and "is" may have less significance than "apple" and "red." Transformers are primarily used in NLP and CV. One such transformer model highlighted in this project is Bidirectional Encoder Representations from Transformers (BERT) Model, an unsupervised machine learning algorithm designed by Google to allow computers to understand the meanings of specific text and converts human language into semantic embeddings, or vectors.

2.4 K-Means Clustering

K-Means Clustering is an unsupervised machine learning algorithm for vector organization by which all the vectors in a space is separated into k groups. In this process, it will form k random centroids and will assign each vector to the closest centroid, labeled 1, 2, 3 ... k. Through trial and error, centroids will adjust themselves by taking the average of the vectors in its each respective cluster. This process is repeated over and over until in two consecutive runs, the cen-

troids do not move or the adjustment is minimal.

3 Data Processing

3.1 PDF Text Extraction

In many cases, PDF files may be scanned documents, in which case there will only be image data present in the document, which is not searchable. In order to extract the text data within, optical character recognition (OCR) is used.

Initially, we used the Pytesseract OCR module to quickly pull printed text data out of an image. However, it was unable to properly read handwritten text data accurately with the vast amount of variation present. A separate system was implemented to convert handwritten text into a retrievable format. To be able to properly extract handwritten text data, the handwritten text must be located and isolated within each page before being fed into the model. This was initially done using the Efficient and Accurate Scene Text Detector (EAST) model within the OpenCV library, made by [list research paper peoples here from who wrote that paper].

The results were found to be inconsistent. Furthermore, it would be better to have text be segmented on a line-by-line, rather than word-by-word, basis to better maintain the formatting of the document. A simpler and more effective method involved transforms in the OpenCV library to highlight text contours in scanned data.

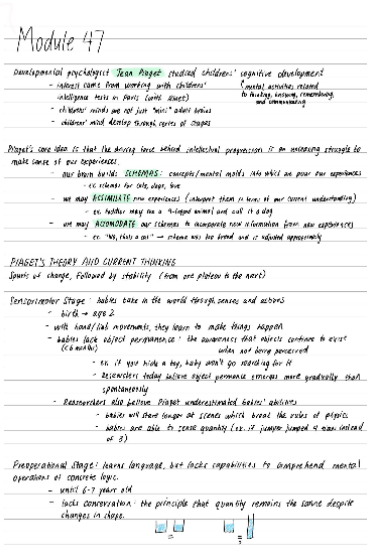


Figure 3: OCR

1. The scanned image is converted to grayscale and its colors are inverted.
2. A threshold is applied to convert each pixel to either 0 or its max value. This helps improve the contrast, makes the lighting consistent, and reduces any noise.
3. The remaining contours on the image are dilated in the x direction to consolidate each line of text into one shape.
4. A bounding box is created around the remaining contours, which mostly consists of each line. Visualization of each bounding box on original image.

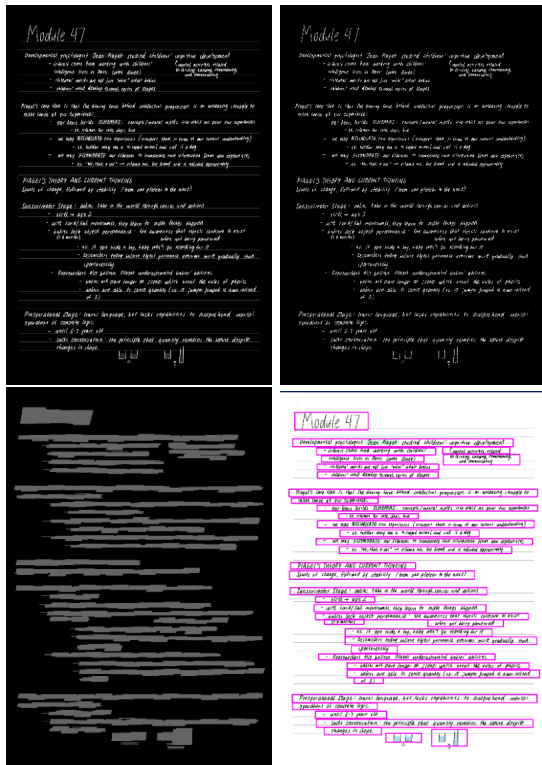


Figure 4: OCR

This method presents limitations to the format of image data in which text may be detectable, however since our main focus is on scanned documents for data retrieval, this should be adequate. Then, with each line of text cropped, it can be fed into a something this weekend - note: still spitting out gibberish; 5 layer convolutional neural network to perform feature extraction. After retrieving the

3.2 Image Captioning

For images and photographs that don't contain text, generating a caption to describe the

image will need to be done to make it searchable. To do this, a convolutional neural network is utilized to extract features from the input image. We used the pre-trained 50 layer deep ResNet model. Unique for its residuals which pass data past certain layers, it helped reduce the degradation problem, where the accuracy of neural networks decreased with a greater number of layers due to the model already solving everything before the end of the network.

Because classification isn't needed, the final fully-connected layer is removed from the network, and the features is then passed into a recurrent neural network, which decodes the feature vector into a basic caption that can be used to describe the image.

This model was trained on the Flickr8K dataset for X epochs. jexample images and captions;

4 Data Retrieval

4.1 The BERT Model

Within the vast field of NLP are various language models that have been developed to make such innovations possible. The BERT Model, an unsupervised machine learning algorithm, allows computers to understand the meanings of specific text. This serves as the bridge between human language and semantic embeddings, allowing the text files and queries to be comparable numerically. When a text is inputted into BERT, the model will create a tensor vector with numerical values equivalent to the meaning of the text.

4.2 Determining Proximity

Two important methods of measuring the proximity between vector points in vector space are the cosine similarity and the Euclidean distance. Although each has their own advantage, they take different approaches of identifying the closest set of points.

4.2.1 Cosine Similarity

Cosine Similarity is a measure of proximity between two vectors by determining the cosine of the angle between them. Thus, the smaller the angle difference, the higher the similarity score. Two vectors of equal orientation (0°) have a maximum similarity of 1, two vectors that are orthogonal (90°) have a similarity of

0, and two vectors that lie directly opposed to each other (180°) have a minimum similarity of -1. This method is therefore a judgement of orientation and direction, independent of the vectors' magnitude and weight.

$$\cos(A, B) = \frac{A \cdot B}{\|A\| \|B\|}$$

4.2.2 Euclidean Distance

In contrast, Euclidean Distance is the distance between two points in space, or the length of a line segment connecting those two points. The greater the difference between the points, the greater the distance. Thus, this method does consider a vector's magnitude and weight when calculating the proximity.

$$d(A, B) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Rather than finding the physical distance between two points, finding the angle measure between vectors is more accurate when measuring document similarity in text analysis. Because texts can vary in terms of size and length yet can contain the same meaning as another, thus the same orientation, cosine similarity is more effective in catching the semantics and therefore more practical in this scenario.

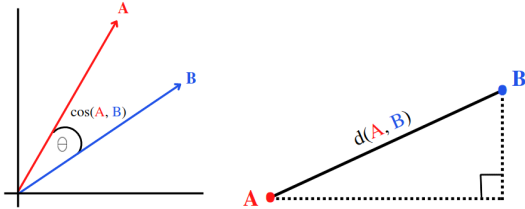


Figure 5: Cosine Similarity vs Euclidean Distance

4.3 Acceleration through K-Means Clustering

In preparation for the experiment, we used L2 normalization to make the lengths of all sentence, reference, and query embedding vectors equal to 1. Two different modes are used: base (without optimization) and kmeans (optimization using k-means clustering and reference vectors). All loading and query times are recorded for numerical representations of each reference point's efficiency.

Finding the most relevant item in terms of cosine similarity is equivalent to finding the one

with the lowest Euclidean distance. To speed up the top-k nearest neighbor search, it is critical to reduce the number of cosine similarity computations between a query and the items. K-means clustering optimizes the squared Euclidean distance in the embedding space between the centroids and the items, and the cosine similarity between a query and a centroid can serve as a proxy to the similarity between a query and the items of the cluster. Hence, we can limit our search to the cluster with the maximum distance between a query and the centroid.

When $k=1$, it degenerates to a full-blown search. When $k > 1$, it will reduce the overall search time but with the risk of sub-optimal search result. For example, when the embedding vector of a query is on the boundary between two clusters, our acceleration scheme focus on the one with a closer distance, hence leaving out some high-quality items from the search. The larger the k for clustering, the higher the chance of sub-optimal result but the lower the time for the search results.

5 Results

5.1 Base

The base mode took around 17 seconds to load and prepare and about 7 seconds to calculate the top 5 nearest neighbor vectors for each of the 10 queries. While the base mode is the 100% accurate since it compares the query to every sentence in the text documents, times will get increasingly slower as the system is exposed to more and more data.

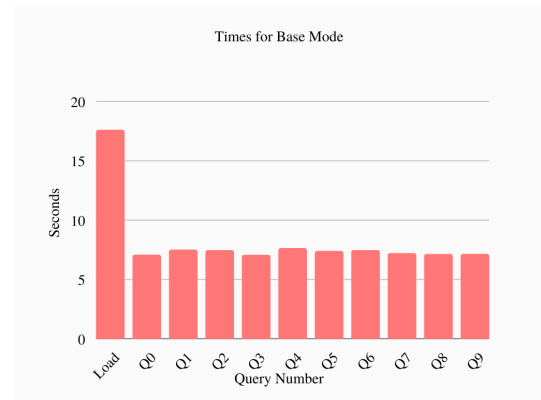


Figure 6: Loading and Query Times for Base Mode

5.2 K-Means Clustering

We experimented 6 runs with the kmeans mode, using 1, 2, 4, 8, 16, and 32 numbers of centroids. As shown in the graph, query times have decreased as the number of centroids increased. Ideally, at some point the time will be essentially 0. However, there is a limit, almost as if it was an asymptote, due to the for loop in which it recognizes if the label of each sentence is part of the closest centroid. Since that step cannot be optimized, there will always be some amount of wait time. Despite this, the overall wait time has reduced significantly.

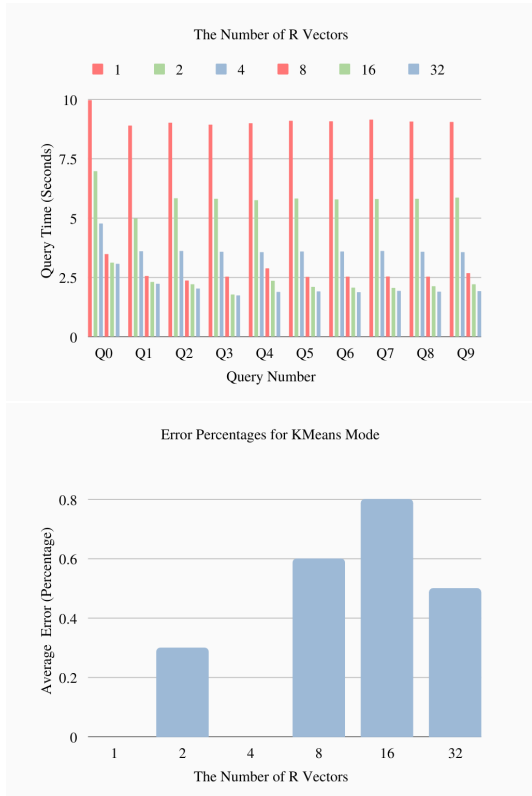


Figure 7: Query Times and Error Percentages for K-Means Mode

This comes as a trade-off with a risk of errors and inaccuracy, however. Errors, because they occur by chance, the accuracy of the search result can waver. Although overall errors increase, the percentage of error are less than 1%. Note that both factors may go above these thresholds as k increases.

6 Future Work

Throughout this project, we have been able to produce a data retrieval system through the use of natural language processing in addition to machine learning and neural networks. While

our program has been optimized and improved, we believe that further steps could be taken in the future to make this project more advanced. These are multiple next steps that can be taken to build on these findings:

1. Implementing the ability to search and sort through a even larger mass of files
2. Improving the accuracy of search results
3. Attention Mechanism in image cap, More detailed image cap (How?)
4. Format recognition in OCR

Acknowledgements

We would like to sincerely thank Optiver for providing us their space during this internship, Mark Galassi and Rhonda Crespo for running the ICR and giving us this research opportunity, our fellow interns for being supportive along the way, and finally our mentor Jim Davies for guiding us through this project.

References

- [1] Rebecca Reynoso (2021) *A Complete History of Artificial Intelligence*, G2.
- [2] Shamala Gallagher, Anna Rafferty, and Amy Wu (2004) *Natural Language Processing*, Stanford University.
- [3] Rani Horev (2018) *BERT Explained: State of the art language model for NLP*, Towards Data Science.
- [4] Daniela M. Witten and Robert Tibshiran (2010) *A Framework for Feature Selection in Clustering*, Journal of the American Statistical Association.
- [5] Nandan Thakur, Nils Reimers, Johannes Daxenberger, and Iryna Gurevych (2021) *Augmented SBERT: Data Augmentation Method for Improving Bi-Encoders for Pairwise Sentence Scoring Tasks*, Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics.