TEXT+IMAGE MULTIMODAL SEARCH USING MOBILENET

A Major Project Report submitted in the partial fulfillment of the requirements for the award of the degree

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Submitted by

D.Hemanth Kumar	**19471A0514**
P.Naveen	**19471A0547**
P.Pavan Kumar	**19471A0543**

Under the Esteemed Guidance of

Shaik Rafi M.Tech., (Ph.D)

Asst.Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NARASARAOPETA ENGINEERING COLLEGE: NARASARAOPETA

(AUTONOMOUS)

Accredited by NAAC with A+ Grade and NBA under Cycle -1
Approved by AICTE, New Delhi, Permanently Affiliated to JNTUK, Kakinada
KOTAPPAKONDA ROAD, YALAMANDA VILLAGE, NARASARAOPET-522601

2022-2023

NARASARAOPETA ENGINEERING COLLEGE

(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the major project entitled **"TEXT + IMAGE MULTIMODEL SEARCH USING MOBILENET"** is a bonafide work done by **"D.Hemanth Kumar (19471A0514), P.Naveen (19471A0547), P.Pavan Kumar (19471A0543)"** in partial fulfillment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** in the Department of **COMPUTER SCIENCE AND ENGINEERING** during 2022-2023.

PROJECT GUIDE

Shaik Rafi M.Tech.,(Ph.D)

Asst. Professor

PROJECT CO-ORDINATOR

M.Sireesha M.Tech., Ph.D

Assoc. Professor

HEAD OF THE DEPARTMENT

Dr.S.N.TirumalaRao M.Tech.,Ph.D

Professor

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We wish to express our thanks to various personalities who are responsible for the completion of the project. We are extremely thankful to our beloved chairman sir **M.V.KoteswaraRao,**B.sc who took keen interest on us in every effort throughout this course. We owe out sciencer gratitude to our beloved principal **Dr.M.Sreenivasa Kumar,**M.Tech.,Ph.D(UK).,MISTE.,FIE(I) for his kind attention and valuable guidance throughout the course.

We express our deep felt gratitude to **Dr.S.N.TirumalaRao,**M.Tech.,Ph.D H.O.D. CSE department and our guide **Shaik Rafi** M.Tech.,(Ph.D) of CSE department whose valuable guidance and unstinting encouragement enable me to accomplish my project successfully in time.

We extend our sincere thanks to **M.Sireesha,** M.Tech.,Ph.D Assoc.prof coordinator of the project for extending her encouragement. Their profound knowledge and willingness have been a constant source of inspiration for me throughout this project work.

We extend our sincere thanks to all other teaching and non-teaching staff to department for their cooperation and encouragement during my B.Tech degree. We have no words to acknowledge the warm affection, constant inspiration and encouragement that we receive from my parents. We affectionately acknowledge the encouragement received from my friends and those who involved in giving valuable suggestions had clarifying out doubts which had really helped me in successfully completing my project.

By:

D.HemanthKumar (19471A0514)

P.Naveen (19471A0547)

P.Pavan Kumar (19471A0543)

ABSTRACT

Multimodal search is a type of search that combines multiple modes of input to retrieve relevant results from a collection of multimedia data. This includes text, images, audio, and video data. Unlike traditional unimodal search, which only uses one type of input, multimodal search takes advantage of the complementary information provided by different modes of input.

The search process involves a combination of techniques from computer vision, natural language processing, and speech recognition. The system extracts features from each mode of input and then applies a machine learning algorithm to identify relevant results based on the similarity between the features.

Multimodal search has numerous applications, such as in e-commerce, social media, and digital libraries. It allows users to search for products, find similar images and videos, and explore multimedia collections in a more intuitive and efficient way. In addition, multimodal search has the potential to enable new applications in fields such as healthcare, where multimodal data is becoming increasingly important for diagnosis and treatment.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

INSTITUTE VISION AND MISSION

INSTITUTION VISION

To emerge as a Centre of excellence in technical education with a blend of effective student centric teaching learning practices as well as research for the transformation of lives and community,

INSTITUTION MISSION

M1: Provide the best class infra-structure to explore the field of engineering and research

M2: Build a passionate and a determined team of faculty with student centric teaching,imbibing experiential, innovative skills

M3: Imbibe lifelong learning skills, entrepreneurial skills and ethical values in students for addressing societal problems

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION OF THE DEPARTMENT

To become a centre of excellence in nurturing the quality Computer Science &Engineering professionals embedded with software knowledge, aptitude for research and ethical values to cater to the needs of industry and society.

MISSION OF THE DEPARTMENT

The department of Computer Science and Engineering is committed to

M1: Mould the students to become Software Professionals, Researchers and Entrepreneurs by providing advanced laboratories.

M2: Impart high quality professional training to get expertize in modern software tools and technologies to cater to the real time requirements of the Industry.

M3: Inculcate team work and lifelong learning among students with a sense of societal and ethical responsibilities.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Specific Outcomes (PSO's)

PSO1: Apply mathematical and scientific skills in numerous areas of Computer Science and Engineering to design and develop software-based systems.

PSO2: Acquaint module knowledge on emerging trends of the modern era in Computer Science and Engineering

PSO3: Promote novel applications that meet the needs of entrepreneur, environmental and social issues.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Educational Objectives (PEO's)

The graduates of the programme are able to:

PEO1: Apply the knowledge of Mathematics, Science and Engineering fundamentals to identify and solve Computer Science and Engineering problems.

PEO2: Use various software tools and technologies to solve problems related to academia, industry and society.

PEO3: Work with ethical and moral values in the multi-disciplinary teams and can communicate effectively among team members with continuous learning.

PEO4: Pursue higher studies and develop their career in software industry.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Outcomes

1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. Problem analysis: Identify, formulate, research literature, and analyzecomplex engineering problems reaching substantiated conclusions using firstprinciples of mathematics, natural sciences, and engineering sciences.

3. Design/development of solutions: Design solutions for complexengineering problems and design system components or processes that meetthe specified needs with appropriate consideration for the public health andsafety, and the cultural, societal, and environmental considerations.

4. Conduct investigations of complex problems: Use research-basedknowledge and research methods including design of experiments, analysis andinterpretation of data, and synthesis of the information to provide validconclusions.

5. Modern tool usage: Create, select, and apply appropriate techniques,resources, and modern engineering and IT tools including prediction andmodeling to complex engineering activities with an understanding of thelimitations.

6. The engineer and society: Apply reasoning informed by the contextualknowledge to assess societal, health, safety, legal and cultural issues and theconsequent responsibilities relevant to the professional engineering practice.

7. Environment and sustainability: Understand the impact of theprofessional engineering solutions in societal and environmental contexts,and demonstrate the knowledge of, and need for sustainable development.

8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Project Course Outcomes (CO'S):

CO425.1: Analyse the System of Examinations and identify the problem.

CO425.2: Identify and classify the requirements.

CO425.3: Review the Related Literature

CO425.4: Design and Modularize the project

CO425.5: Construct, Integrate, Test and Implement the Project.

CO425.6: Prepare the project Documentation and present the Report using appropriate method.

Course Outcomes – Program Outcomes mapping

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C425.1		✓											✓		
C425.2	✓		✓		✓								✓		
C425.3				✓		✓	✓	✓					✓		
C425.4			✓			✓	✓	✓					✓	✓	
C425.5					✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
C425.6									✓	✓	✓		✓	✓	

Course Outcomes – Program Outcome correlation

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C425.1	2	3											2		
C425.2			2		3								2		
C425.3				2		2	3	3					2		
C425.4			2			1	1	2					3	2	
C425.5					3	3	3	2	3	2	2	1	3	2	1
C425.6									3	2	1		2	3	

Note: The values in the above table represent the level of correlation between CO's and PO's:

1. **Low level**

2. **Medium level**

3. **High level**

Project mapping with various courses of Curriculum with Attained PO's:

Name of the course from which principles are applied in this project	**Description of the device**	**Attained PO**
C3.2.4, C3.2.5	Gathering the requirements and defining the problem, plan to develop a **smart bottle for health care using sensors.**	PO1, PO3
CC4.2.5	Each and every requirement is critically analyzed, the process model is identified and divided into **five modules**	PO2, PO3
CC4.2.5	Logical design is done by using the unified modelling language which involves individual team work	PO3, PO5, PO9
CC4.2.5	Each and every module is tested, integrated, and evaluated in our project	PO1, PO5
CC4.2.5	Documentation is done by all our four members in the form of a group	PO10
CC4.2.5	Each and every phase of the work in group is presented periodically	PO10, PO11
CC4.2.5	Implementation is done and the project will be handled by the **hospital management and in future updates in our project can be done based on air bubbles occurring in liquid in saline.**	PO4, PO7
CC4.2.8 CC4.2.	**The physical design includes hardware components like sensors, gsm module, software and Arduino.**	PO5, PO6

Index

1.INTRODUCTION

1.1 INTRODUCTION

In today's world, we are surrounded by an ever-increasing amount of multimedia data in the form of text, images, and videos. However, traditional unimodal search methods that only rely on one type of input may not always be effective in retrieving relevant information. This is where multimodal search, which combines multiple modes of input, comes in.

One popular form of multimodal search is text+image search, which allows users to input a query in the form of text, image, or a combination of both, and retrieve relevant results from a collection of multimedia data that includes both text and images. This type of search is increasingly being used in applications such as e-commerce, social media, and digital libraries, as it enables users to search for products, find similar images, and explore multimedia collections in a more intuitive and efficient way.

Text+image multimodal search involves a complex process of extracting features from both the text and image inputs, and then applying a machine learning algorithm to identify relevant results based on the similarity between the features. This process requires a combination of techniques from computer vision, natural language processing, and information retrieval.

Overall, text+image multimodal search has the potential to revolutionize the way we search for and access multimedia data, and is becoming increasingly important as the amount of multimedia data continues to grow.

1.2 EXISTING SYSTEM

- The existing system of multimodel search has generative model approach to multimodal search, which improves the quality of textual-visual retrieval by generating realistic images that match textual descriptions.
- It takes text input to generate images similar to the input text.
- Most models uses rankings for images to display the relavent images based on text

1.3 PROPOSED SYSTEM

- The proposed system of multimodel search has improves the quality of textual-visual retrieval by generating realistic images that match both textual and image inputs.
- It uses Text Encoder, Image Encoder and Retrieval Methods to display relavent test and images for the given query .
- Text Encoder is "A pre-trained language model, such as BERT" that encodes textual descriptions into a fixed-length vector representation.
- Image Encoder is "A pre-trained convolutional neural network (CNN)" that encodes images into a fixed-length vector representation.
- Retrieval Method are the methods for retrieving relevant images based on a given textual query, such as nearest-neighbor search.

1.4 SYSTEM REQUIREMENTS

HARDWARE REQUIREMENTS

- Processor : Intel core i5 with 1.6 GHZ minimum.
- Hard Disk : 30 GB or more.
- RAM : 1GB or more.

SOFTWARE REQUIREMENTS

- Operating System : Windows 11
- Programming Language : Python
- Editor : Google Colab
- Browser : Any Latest Browser like Chrome

2. LITERATURE SURVEY

Multimodal search is the task of retrieving multimedia content, such as images, videos, and audio, based on a combination of textual and visual queries. It is an important research area that aims to enable users to find relevant multimedia content based on their specific needs and preferences.

In a typical multimodal search scenario, the user provides a textual query that describes the type of content they are looking for. The system then uses this textual query to retrieve multimedia content that is relevant to the user's needs, based on both the textual and visual content of the multimedia items.

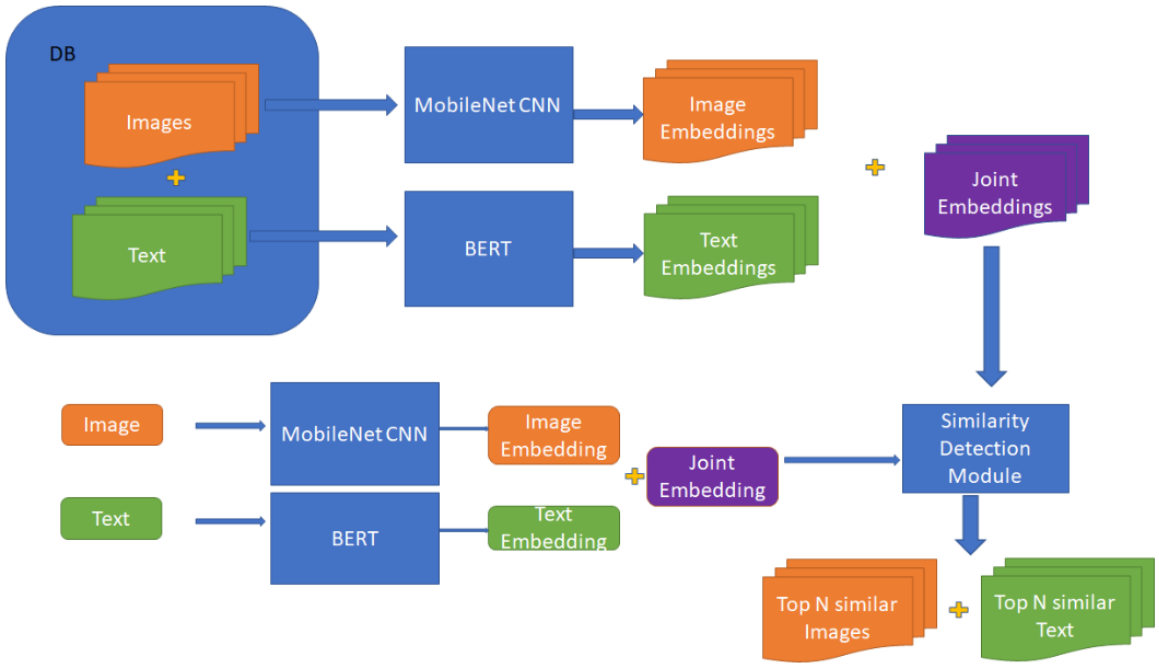
To achieve this, multimodal search systems typically use machine learning algorithms and deep neural networks to analyze the textual and visual content of the multimedia items. They generate embeddings or feature vectors that capture the key semantic and visual information of the content. These embeddings are then used to perform content-based retrieval, where the system retrieves multimedia items that have similar embeddings to the user's textual query.

Multimodal search has been the focus of much research in recent years, with many advances in machine learning and deep learning techniques leading to improvements in the accuracy and efficiency of these systems. The state-of-the-art approaches often involve developing models and techniques that can effectively integrate textual and visual information, such as joint embedding spaces, adversarial learning methods, and generative models.

Overall, multimodal search is an important and challenging research area that has the potential to enable more efficient and effective multimedia search. A literature survey of multimodal search would explore the current state-of-the-art approaches, highlight the key trends and research questions in this field, and identify areas for further research and development.

3. METHODOLOGY

3.1 ARCHITECTURE



3.2 MOBILENET CNN FOR IMAGE CLASSIFICATION

MobileNet is a type of convolutional neural network (CNN) that is designed to be efficient for use on mobile devices. It was first introduced by Google in 2017 and has since become a popular architecture for mobile applications due to its lightweight design.

The MobileNet architecture consists of a series of depthwise separable convolution layers. This means that each convolutional layer is split into two parts: a depthwise convolution that applies a single filter to each input channel, followed by a pointwise convolution that applies a 1x1 convolution to combine the outputs of the depthwise convolution.

The depthwise separable convolution reduces the number of parameters and computations required by the network, which makes it much faster and more efficient than traditional convolutional neural networks. This makes it ideal for use on mobile and embedded devices where computational resources are limited.

There are several variants of the MobileNet architecture, including MobileNet V1, MobileNet V2, and MobileNet V3, each with different improvements and optimizations to increase accuracy and efficiency.

MobileNet CNN works by using a specific type of convolutional layer called depthwise separable convolution.

A typical convolutional layer in a CNN applies a set of filters to the input image to extract features. The number of filters determines the depth of the output volume. Each filter is a small matrix that is slid across the input image to produce a feature map.

In contrast, a depthwise separable convolution separates the convolution into two steps: a depthwise convolution and a pointwise convolution.

The depthwise convolution applies a single filter to each input channel separately, producing a set of feature maps with the same depth as the input. This step reduces the number of computations required compared to a traditional convolutional layer.

The pointwise convolution applies a 1x1 filter to combine the output of the depthwise convolution, producing a final output volume. The pointwise convolution is used to mix and match the feature maps produced by the depthwise convolution.

MobileNet CNN uses a sequence of these depthwise separable convolutions, followed by other layers like pooling and fully connected layers, to extract features and classify images.

The depthwise separable convolution reduces the number of parameters and computations required by the network, making it much faster and more efficient than traditional convolutional neural networks. This makes it ideal for use on mobile and embedded devices where computational resources are limited.

3.3 LAYERS OF CNN

The layers of a Convolutional Neural Network (CNN) typically include:

Input Layer: The input layer takes the raw data, which is typically an image or a set of images, and prepares it for processing by the network.

Convolutional Layer: The convolutional layer applies a set of filters to the input image to extract features. Each filter is a small matrix that is slid across the input image to produce a feature map.

Activation Layer: The activation layer applies a non-linear activation function, such as ReLU, to the output of the convolutional layer. This helps to introduce non-linearity into the network and improve its ability to model complex patterns in the data.

Pooling Layer: The pooling layer reduces the dimensionality of the data by downsampling the feature maps produced by the convolutional layer. This helps to reduce the computational complexity of the network and prevent overfitting.

Fully Connected Layer: The fully connected layer performs classification or regression tasks by taking the output of the pooling layer and transforming it into a vector of class probabilities or regression values.

Dropout Layer: The dropout layer randomly drops out some of the neurons in the network during training. This helps to prevent overfitting and improve the generalization performance of the network.

Batch Normalization Layer: The batch normalization layer normalizes the inputs to the network to reduce the internal covariate shift. This helps to improve the stability and speed of the training process.

3.4 BERT FOR TEXT CLASSIFICATION

BERT stands for Bidirectional Encoder Representations from Transformers. It is a pre-trained deep learning model for natural language processing (NLP) developed by Google in 2018.

BERT is a type of transformer model, which is a neural network architecture that processes input data using self-attention mechanisms. The self-attention mechanism allows the model to focus on different parts of the input sequence to generate a contextual representation of each word or token.

What sets BERT apart from other NLP models is that it is pre-trained on a large corpus of text data using a masked language modeling task and a next sentence prediction task. This pre-training allows the model to learn general language understanding, which can then be fine-tuned for specific downstream NLP tasks such as question answering, sentiment analysis, and named entity recognition.

BERT uses a bidirectional approach, meaning that it takes into account the entire input sequence to generate its representations. This is different from traditional NLP models, which only look at the input sequence in a forward or backward direction.

BERT has achieved state-of-the-art results on a wide range of NLP benchmarks, and its pre-trained weights have been made available to the research community for further exploration and fine-tuning. Its success has led to the development of other transformer-based models, such as GPT-2 and T5.

3.5 EMBEDDINGS

3.5.1: Image Embedding

Image embeddings refer to a low-dimensional, dense vector representation of an image that captures its features and characteristics. These embeddings are commonly used in computer vision tasks such as image retrieval, object recognition, and image classification.

The process of generating image embeddings involves passing an image through a deep neural network, typically a convolutional neural network (CNN). The CNN extracts features from the image at various levels of abstraction, with deeper layers capturing more abstract and high-level features. The final output of the CNN is a high-dimensional feature vector that captures the image's characteristics.

To obtain a lower-dimensional representation of the feature vector, dimensionality reduction techniques such as principal component analysis (PCA) or t-distributed stochastic neighbor

embedding (t-SNE) can be applied. These techniques preserve the most important features while reducing the dimensionality of the vector.

Image embeddings can be used in a variety of ways. One common application is in image retrieval systems, where the embeddings of a query image are compared to the embeddings of a database of images to find the most similar images. Image embeddings can also be used in image classification tasks, where the embeddings of an image are fed into a classifier to predict the class of the image.

Overall, image embeddings provide a powerful tool for representing and comparing images in a low-dimensional space, enabling efficient and accurate image processing and analysis.

3.5.2: Text Embedding

Text embeddings are low-dimensional, dense vector representations of textual data such as words, sentences, or documents. They are commonly used in natural language processing (NLP) tasks such as text classification, sentiment analysis, and language translation.

The process of generating text embeddings involves representing words, sentences, or documents as numerical vectors that capture their semantic meaning. There are several methods for generating text embeddings, including:

1. Count-based methods: These methods represent words as vectors based on their frequency of occurrence in a corpus of text. Examples include term frequency-inverse document frequency (TF-IDF) and Latent Semantic Analysis (LSA).
2. Prediction-based methods: These methods use neural networks such as Word2Vec, GloVe, and fastText to predict the context of a word based on its neighboring words. The resulting embeddings capture the semantic relationships between words.
3. Contextualized methods: These methods use pre-trained language models such as BERT and GPT-2 to generate embeddings that take into account the context of the surrounding words. These embeddings are highly contextual and capture the nuances of language.

Text embeddings can be used in a variety of ways. For example, in text classification tasks, the embeddings of a sentence or document are fed into a classifier to predict the category of

the text. In language translation tasks, embeddings of the source language are generated and used to generate embeddings of the target language.

Overall, text embeddings provide a powerful tool for representing and analyzing textual data in a low-dimensional space, enabling efficient and accurate natural language processing.

3.5.3: Joint Embeddings

Joint embeddings refer to the creation of a shared vector space that can be used to represent different types of data, such as images and text, in a unified way. The idea is to learn a single set of embeddings that can capture the semantic relationships between different types of data, enabling cross-modal retrieval and analysis.

The process of creating joint embeddings involves training a model that can encode both image and text data into a shared vector space. One common approach is to use a deep neural network, such as a convolutional neural network (CNN) for images and a recurrent neural network (RNN) or transformer-based model for text, to generate embeddings for each modality. The embeddings are then optimized to be as close as possible in the shared vector space, so that similar images and text are represented by similar vectors.

Joint embeddings have a wide range of applications, including image captioning, visual question answering, and cross-modal retrieval. For example, in image captioning, the model generates a description of an image using the joint embeddings of the image and text. In visual question answering, the model uses joint embeddings to relate the image and text components of a question and generate an answer.

Overall, joint embeddings provide a powerful tool for representing and analyzing multiple types of data in a unified way, enabling cross-modal analysis and retrieval.

3.6 SIMILARITY DETECTION MODULE

A similarity detection module is a component of a machine learning system that is used to measure the similarity between two pieces of data. The data can be of any type, such as images, text, audio, or video. The module typically uses a similarity metric, such as cosine similarity or Euclidean distance, to measure the degree of similarity between the data points.

In image-based similarity detection, the module may use features extracted from a convolutional neural network (CNN) to compare two images. The similarity metric is used to compare the features of the two images to determine how similar they are.

In text-based similarity detection, the module may use text embeddings generated from pre-trained models such as BERT or GloVe. The similarity metric is then used to compare the embeddings of two pieces of text to determine how similar they are.

In audio and video-based similarity detection, the module may use features such as spectrograms or waveforms to compare two pieces of audio or video data.

Applications of similarity detection modules include content-based image retrieval, plagiarism detection, recommendation systems, and fraud detection.

Overall, similarity detection modules provide a powerful tool for measuring the similarity between different types of data, enabling efficient and accurate analysis and retrieval.

3.7 TRANSFER LEARNING

Transfer learning is a machine learning technique that involves reusing pre-trained models for a new task or dataset. Rather than starting the learning process from scratch, a pre-trained model is used as a starting point, and then fine-tuned to improve performance on the new task or dataset.

In transfer learning, the pre-trained model is usually trained on a large dataset, such as ImageNet for image classification or BERT for natural language processing. The model has learned to recognize a wide range of features in the data and has already gone through a process of optimizing its weights and biases.

The pre-trained model can then be used as a feature extractor for the new task or dataset, with the learned features being fed into a new neural network architecture, which is then fine-tuned on the new data. This allows the new model to learn from a smaller dataset and converge faster, as it is already starting from a point of knowledge.

Transfer learning has several advantages, including reducing the amount of training data needed for a new task, reducing the computational resources required for training, and improving the generalization and accuracy of the model.

Applications of transfer learning include image classification, natural language processing, speech recognition, and recommendation systems. For example, a pre-trained model for image classification can be used to recognize objects in medical images, and a pre-trained model for natural language processing can be used to generate captions for images.

Overall, transfer learning is a powerful technique that can save time, resources, and improve performance in machine learning tasks by leveraging existing knowledge and pre-trained models.

4.LIBRARIES AND MODULES

4.1 LIBRARIES

4.1.1 numpy

NumPy is a Python library for numerical computing that provides support for multi-dimensional arrays and matrices, along with a large collection of mathematical functions to operate on these arrays. It is one of the most popular libraries in the scientific Python ecosystem, and is used extensively in fields such as data science, machine learning, and engineering.

Some of the key features of NumPy include:

1. ndarray: a fast and efficient multidimensional array object that allows mathematical operations to be performed on entire arrays rather than individual elements.
2. Broadcasting: a powerful tool for performing arithmetic operations between arrays with different shapes and dimensions.
3. Mathematical functions: a large collection of mathematical functions that operate on arrays, including basic operations like addition and multiplication, as well as more advanced functions like trigonometric functions, logarithms, and statistics functions.
4. Indexing and slicing: NumPy provides several ways to index and slice arrays, including boolean indexing, integer indexing, and advanced indexing.
5. Integration with other scientific Python libraries: NumPy is designed to work seamlessly with other libraries in the scientific Python ecosystem, including SciPy, Matplotlib, and Pandas.

NumPy is an essential tool for many data scientists and scientific computing practitioners. Its efficient implementation of array operations allows for fast computations on large datasets, making it ideal for applications like numerical simulations, data analysis, and machine learning.

4.1.2 pandas

Pandas is a Python library for data manipulation and analysis. It provides data structures for efficiently storing and manipulating data, as well as tools for cleaning, merging, reshaping, and analyzing data.

The two primary data structures provided by Pandas are:

1. Series: a one-dimensional array-like object that can hold any data type, including numerical, categorical, and textual data. Series are useful for representing time-series data, as well as for labeling and indexing data.
2. DataFrame: a two-dimensional table-like object that consists of columns, each of which can hold a different data type. DataFrames are commonly used for working with structured data, such as CSV files, SQL tables, and Excel spreadsheets.

Some of the key features of Pandas include:

1. Data cleaning and preparation: Pandas provides tools for handling missing data, converting data types, and transforming data into a format that is suitable for analysis.
2. Data aggregation and grouping: Pandas makes it easy to group data by one or more variables and apply functions to each group. This is useful for tasks like summarizing data and computing statistics.
3. Data visualization: Pandas integrates with Matplotlib, a popular data visualization library, to provide easy-to-use tools for creating charts and graphs from data.
4. Integration with other Python libraries: Pandas is designed to work seamlessly with other Python libraries, such as NumPy, SciPy, and Scikit-learn.

Pandas is widely used in the data science community for tasks like data cleaning and preparation, exploratory data analysis, and statistical modeling. Its powerful data manipulation tools make it a versatile tool for working with a wide range of data types and structures.

4.1.3 Tensorflow

TensorFlow is an open-source machine learning framework developed by Google that is used to build and train machine learning models. It provides a set of tools and libraries that make it easy to design, build, and deploy machine learning models at scale.

Some of the key features of TensorFlow include:

1. High-level APIs: TensorFlow provides high-level APIs, such as Keras and Estimators, that make it easy to build and train machine learning models without requiring deep knowledge of the underlying algorithms.
2. Low-level APIs: TensorFlow also provides low-level APIs that give developers fine-grained control over the computation graph and allow them to optimize their models for performance.
3. Distributed computing: TensorFlow can distribute the training of a machine learning model across multiple devices and machines, allowing for faster training and the ability to train larger models.
4. TensorBoard: a visualization tool that helps developers monitor the progress of their machine learning models and visualize the computation graph.
5. Compatibility with different platforms: TensorFlow can be used on a wide variety of platforms, including CPUs, GPUs, and TPUs (Tensor Processing Units), as well as on mobile devices and in the cloud.

TensorFlow is widely used in the machine learning community for tasks such as image classification, natural language processing, and reinforcement learning. Its flexibility, scalability, and ease of use make it a popular choice for building and deploying machine learning models in a variety of industries, including healthcare, finance, and e-commerce.

4.1.4 matplotlib

Matplotlib is a popular Python library for creating static, animated, and interactive visualizations in Python. It provides a wide range of visualization tools for scientific and engineering applications, as well as for data analysis and exploration. Some of the key features of Matplotlib include:

1. Support for a wide range of plotting types, including line plots, scatter plots, bar charts, histograms, and 3D plots.
2. Customizable plotting options, including control over colors, line styles, marker styles, and axis labeling.
3. Support for interactive plotting in Jupyter notebooks and web applications using tools such as ipywidgets and Bokeh.
4. Integration with other scientific Python libraries such as NumPy, SciPy, and Pandas.
5. Cross-platform compatibility, with support for Windows, Mac OS, and Linux.

Matplotlib is widely used in academia, research, and industry for scientific visualization, data analysis, and machine learning applications. It is also a popular tool for data exploration and visualization in the data science and analytics communities.

Overall, Matplotlib is a powerful and flexible visualization tool that provides a wide range of options for creating high-quality plots and visualizations in Python.

4.1.5 Sklearn

Scikit-learn (also known as sklearn) is a popular open-source Python library for machine learning. It provides a range of supervised and unsupervised learning algorithms, including classification, regression, clustering, dimensionality reduction, and more.

Scikit-learn is built on top of NumPy, SciPy, and matplotlib, which are popular scientific computing libraries in Python. It provides a consistent interface for applying machine learning algorithms and evaluating their performance. The library is designed to be simple and efficient, with a focus on ease-of-use and code readability.

Scikit-learn is widely used in academia and industry for a variety of applications, including natural language processing, image recognition, and financial forecasting. It also provides tools for data preprocessing, model selection, and hyperparameter tuning.

To use Scikit-learn, you typically import the relevant module, instantiate the appropriate estimator object, fit the estimator object to your data, and then use the estimator object to make predictions or perform other tasks. The library also provides extensive documentation and examples to help you get started.

4.1.6 Sentence_transformer

Sentence Transformer is a Python library for generating high-quality sentence and document embeddings, which are vector representations of text that capture semantic meaning. The library is built on top of the Hugging Face Transformers library and uses state-of-the-art pre-trained language models to generate the embeddings.

The Sentence Transformer library provides a simple interface for generating embeddings from text, allowing you to easily integrate them into your machine learning pipeline. You can use the embeddings to perform a variety of tasks, such as text classification, clustering, and semantic similarity analysis.

One of the key benefits of Sentence Transformer is its ability to generate multilingual embeddings, meaning it can generate embeddings for text in multiple languages. This makes it particularly useful for applications that require processing text in multiple languages.

Sentence Transformer provides a wide range of pre-trained models, including BERT, RoBERTa, DistilBERT, and more. It also allows you to fine-tune these models on your own data, allowing you to generate embeddings that are optimized for your specific use case.

Overall, Sentence Transformer is a powerful tool for generating high-quality text embeddings that can be used for a variety of machine learning applications.

4.1.7 PIL

PIL (Python Imaging Library) is a library for working with images in Python. It provides a set of functions to manipulate and process images, such as cropping, resizing, and converting between image formats. PIL is a third-party library, which means it is not included in the standard Python distribution and needs to be installed separately. In newer versions of

Python, the "Pillow" library is a fork of the original PIL library and is commonly used instead. Therefore, PIL can be considered a library or a module, depending on the context.

4.2 MODULES

4.2.1 Time Module

The “time” module is a built-in module in Python, which means that it is included in the Python standard library and does not require any additional installation or setup. It provides various time-related functions that allow you to measure time intervals, pause the execution of a program, and format dates and times.

The time module in Python provides various time-related functions that can be used to measure time intervals, pause program execution, and format dates and times. Here are some common use cases for the time module:

1. Measuring time intervals: You can use the time.time() function to get the current time in seconds since the epoch (January 1, 1970, 00:00:00 UTC), and calculate the difference between two timestamps to measure the time taken by a block of code to execute. This can be useful for profiling your code or optimizing performance.
2. Pausing program execution: You can use the time.sleep() function to pause the execution of a program for a specified number of seconds. This can be useful if you need to introduce a delay between operations, or if you want to prevent a program from consuming too many resources.
3. Formatting dates and times: You can use the time.strftime() function to format a timestamp as a string according to a specified format string. This can be useful for displaying dates and times in a user-friendly format, or for generating filenames or log entries that include a timestamp.
4. Working with dates and times: The time module also provides functions for working with dates and times, such as time.localtime() to get the current local time, time.gmtime() to get the current UTC time, and time.mktime() to convert a time tuple to a timestamp.

In addition to these basic use cases, the time module can be used in conjunction with other modules, such as datetime and calendar, to perform more advanced operations with dates and

times. Overall, the time module is a versatile tool for working with time-related operations in Python.

To use the time module in Python, you can simply import it using the import time statement.

4.2.2 Pickle Module

The “pickle” module is a built-in module in Python, which means it is part of the Python standard library and does not require any additional installation. The module provides functions to serialize and deserialize Python objects, allowing you to save and load complex data structures in a compact binary format.

The pickle module in Python is used for serializing and deserializing Python objects. Serialization is the process of converting an object into a format that can be stored or transmitted, and deserialization is the process of converting the serialized data back into an object.

The pickle module provides a way to store and retrieve complex Python objects, including custom objects that you define, in a compact binary format. This allows you to save objects to a file or database, or transmit them over a network, and later reload them into your program.

Here are some common use cases for the pickle module:

Storing and loading machine learning models: Machine learning models can be complex objects with many parameters, and training them can be computationally expensive. By using pickle, you can save the trained model to a file and later reload it into memory, saving time and resources.

Caching: If your program performs expensive calculations or generates data that takes a long time to compute, you can use pickle to store the results in a file or database. This way, the next time your program runs with the same input, it can simply load the cached results instead of re-computing them.

Sharing data between processes: If you have multiple processes running in parallel and need to share data between them, you can use pickle to serialize the data and transmit it between the processes.

Saving user preferences: If your program has user-specific preferences or settings, you can use pickle to store these preferences in a file or database, and later load them when the program starts up.

Overall, the pickle module is a powerful tool for serializing and deserializing Python objects, and can be used in a wide variety of applications. However, it's important to note that pickle is not secure against maliciously constructed data, so you should only unpickle data from trusted sources

4.2.3 Similarity Detection module

A similarity detection module is a component of a machine learning system that is used to measure the similarity between two pieces of data. The data can be of any type, such as images, text, audio, or video. The module typically uses a similarity metric, such as cosine similarity or Euclidean distance, to measure the degree of similarity between the data points.

In image-based similarity detection, the module may use features extracted from a convolutional neural network (CNN) to compare two images. The similarity metric is used to compare the features of the two images to determine how similar they are.

In text-based similarity detection, the module may use text embeddings generated from pre-trained models such as BERT or GloVe. The similarity metric is then used to compare the embeddings of two pieces of text to determine how similar they are.

In audio and video-based similarity detection, the module may use features such as spectrograms or waveforms to compare two pieces of audio or video data.

Applications of similarity detection modules include content-based image retrieval, plagiarism detection, recommendation systems, and fraud detection.

Overall, similarity detection modules provide a powerful tool for measuring the similarity between different types of data, enabling efficient and accurate analysis and retrieval.

5. ALGORITHMS

5.1 NEAREST NEIGHBOR ALGORITHM

The nearest neighbor algorithm is a simple algorithm used in machine learning and data analysis for solving classification and regression problems.

In the context of classification, the nearest neighbor algorithm works as follows:

Given a new data point to be classified, find the distance between the new point and all other points in the training dataset.Identify the training point that is closest to the new point based on the chosen distance metric (e.g., Euclidean distance or Manhattan distance). This point is known as the "nearest neighbor."

Assign the class label of the nearest neighbor to the new point.

The algorithm can be extended to use k-nearest neighbors (k-NN), where the k closest training points are identified, and the new point is assigned the class label that is most common among its k neighbors.

In the context of regression, the nearest neighbor algorithm works similarly, but instead of assigning a class label to the new point, it assigns a numerical value based on the average or median of the target variable for the k nearest neighbors.

One of the main advantages of the nearest neighbor algorithm is its simplicity and ease of implementation. However, it can be sensitive to outliers and high-dimensional datasets, and its performance may depend heavily on the choice of distance metric and the value of k.

5.1.1 Cosine Similarity

Cosine similarity is a metric used to measure the similarity between two vectors in a high-dimensional space. It is commonly used in natural language processing and information retrieval to compare the similarity of two documents or sentences.

Cosine similarity is calculated by taking the dot product of the two vectors and dividing it by the product of their magnitudes. The result is a value between -1 and 1, where 1 indicates that

the two vectors are identical, 0 indicates that the two vectors are orthogonal (i.e., have no similarity), and -1 indicates that the two vectors are diametrically opposed.

The formula for cosine similarity is:

cosine_similarity = (A . B) / (||A|| ||B||)

where A and B are the two vectors being compared, . denotes the dot product, and ||A|| and ||B|| denote the magnitudes of the two vectors.

$$Cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \, \|\vec{b}\|} = \frac{\sum_1^n a_i b_i}{\sqrt{\sum_1^n a_i^2} \sqrt{\sum_1^n b_i^2}}$$

where, $\vec{a} \cdot \vec{b} = \sum_1^n a_i b_i = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n$ is the dot product of the two vectors.

One advantage of cosine similarity is that it is not affected by the length of the vectors, only their direction. This makes it useful for comparing documents of varying lengths.

Applications of cosine similarity include document classification, information retrieval, and recommendation systems. For example, in a document classification task, cosine similarity can be used to compare the similarity between the features of two documents and classify them accordingly.

Overall, cosine similarity is a powerful tool for measuring the similarity between two vectors in a high-dimensional space, enabling efficient and accurate analysis and retrieval.

6.IMPLIMENTATION

```
import numpy as np
import pandas as pd
import tensorflow as tf
physical_devices = tf.config.experimental.list_physical_devices('GPU')
assert len(physical_devices) > 0, "Not enough GPU hardware devices available"
config = tf.config.experimental.set_memory_growth(physical_devices[0], True)
train_df = pd.read_csv(r'/content/drive/MyDrive/train.csv)


print(len(train_df))
print(len(train_df['label_group'].unique()))
labelGroup_df = train_df.groupby('label_group')

import matplotlib.pyplot as plt
from PIL import Image

groupCount = 0
for groupName,groupDf in labelGroup_df:
   print(groupName)
   imgCount=0
   for index,row in groupDf.iterrows():
      print(row['title'])
      imagePath = r' /content/drive/MyDrive/train_images'+row['image']
      pil_im = Image.open(imagePath, 'r')
      plt.figure()
      plt.imshow(pil_im)
      plt.show()
      imgCount= imgCount+1
      if (imgCount==3):
         break
   groupCount = groupCount +1
   if (groupCount==10):
      break
from sklearn.model_selection import train_test_split
y = train_df.pop('label_group')
X = train_df
X_train,X_test,y_train,y_test = train_test_split(X,y,stratify=y,test_size=0.4,random_state=0)

from sentence_transformers import SentenceTransformer, util
model = SentenceTransformer('stsb-distilbert-base')
model.max_seq_length = 128


IMG_SIZE = 224
size = (IMG_SIZE,IMG_SIZE)
img_model = tf.keras.applications.MobileNet(input_shape = (IMG_SIZE, IMG_SIZE, 3),
include_top = False, weights = 'imagenet' )
```

```
def get_textEmbeddings(model,text):
    text_embedding = model.encode(text, convert_to_tensor=True)
    return text_embedding


def get_imageEmbeddings(model,imagePath):
    image = tf.keras.preprocessing.image.load_img(imagePath,target_size= size)
    input_arr = tf.keras.preprocessing.image.img_to_array(image)
    input_arr = np.array([input_arr])
    img_embeddings = model(input_arr)
    meanImgEmb1 = np.mean(img_embeddings,axis =0)
    meanImgEmb2 = np.mean(meanImgEmb1,axis=0)
    meanImgEmb = np.mean(meanImgEmb2,axis=0)
    return meanImgEmb


print(np.shape(X_train))
embeddings_cache = {}
def get_imageEmbeddings(model, imagePath):
    if imagePath in embeddings_cache:
        return embeddings_cache[imagePath]
    else:
        ...
        embeddings_cache[imagePath] = img_emb
        return img_emb
def get_imageEmbeddings(model,imagePath):
    try:
        img_emb = embeddings_cache[imagePath]
        return img_emb
    except KeyError:
        ...
import time

text_embeddings={}
image_embeddings={}
start_time = time.time()
for index,row in  X_train.iterrows():
    txt_emb = get_textEmbeddings(model,str(row[3]))
    imagePath =r'/content/drive/MyDrive/train_images'+row[1]
    img_emb = get_imageEmbeddings(img_model,imagePath)
    text_embeddings[row[0]] = txt_emb
    image_embeddings[row[0]] = img_emb
end_time = time.time()
print(str(end_time-start_time))


import pickle
with open('./textEmb','wb') as handle:
```

```
    pickle.dump(text_embeddings,handle)
with open('./imgEmb','wb') as handle:
    pickle.dump(image_embeddings,handle)


keyList=[]
cembList=[]
imageList=[]
titleList=[]
for index, row in X_train.iterrows():
    #start_time=time.time()
    txt_emb = text_embeddings[row[0]]
    print(np.shape(txt_emb))
    img_emb = image_embeddings[row[0]]
    print(np.shape(img_emb))
    cmb_emb = np.concatenate((txt_emb,img_emb),axis=0)
    print(np.shape(cmb_emb))
    norm = np.linalg.norm(cmb_emb)
    cmb_emb_normal = cmb_emb/norm
    keyList.append(row[0])
    cembList.append(cmb_emb_normal)
    imageList.append(row[1])
    titleList.append(row[3])


from sklearn.neighbors import NearestNeighbors
kneigh = NearestNeighbors(n_neighbors=5,leaf_size=5000,algorithm='kd_tree')
kneigh.fit(cembList)


def getNeighbours(query_emb):
    posting_id_list=[]
    neigh_dist,neigh_ind = kneigh.kneighbors(X=query_emb.reshape(1,-1), n_neighbors=5,
return_distance=True)
    for ind in neigh_ind:
        #print(str(ind))
        for ind1 in ind:
            posting_id_list.append(str(ind1))
    return posting_id_list


postingidList=[]
matchesList=[]
index =0
for val in keyList:
    query_emb = cembList[index]
    postingid_list = getNeighbours(query_emb)
    postingidList.append(val)
    matchesList.append(" ".join(postingid_list))
```

```
    index =index +1
    if index==100:
        break


index =0
for item in postingidList:
    print(titleList[index])
    print(keyList[index])
    imagePath = r'/content/drive/MyDrive/train_images'+ imageList[index]
    pil_im = Image.open(imagePath, 'r')
    plt.figure()
    plt.imshow(pil_im)
    plt.show()
    matching_indices = matchesList[index].split(' ')
    print('==================')
    for ind in matching_indices:
        print(titleList[int(ind)])
        print(keyList[int(ind)])
        imagePath = r'/content/drive/MyDrive/train_images'+ imageList[int(ind)]
        pil_im = Image.open(imagePath, 'r')
        plt.figure()
        plt.imshow(pil_im)
        plt.show()
    index= index +1
    if index == 10:
        break


testkeyList=[]
testcembList=[]
testimageList=[]
testtitleList=[]
for index, row in X_test.iterrows():
    #start_time=time.time()
    txt_emb = get_textEmbeddings(model,str(row[3]))
    imagePath = r'/content/drive/MyDrive/train_images'+row[1]
    img_emb = get_imageEmbeddings(img_model,imagePath)
    text_embeddings[row[0]] = txt_emb
    image_embeddings[row[0]] = img_emb
    cmb_emb = np.concatenate((txt_emb,img_emb),axis=0)
    norm = np.linalg.norm(cmb_emb)
    cmb_emb_normal = cmb_emb/norm
    testkeyList.append(row[0])
    testcembList.append(cmb_emb_normal)
    testimageList.append(row[1])
    testtitleList.append(row[3])
```

```
testpostingidList=[]
testmatchesList=[]
index =0
for val in testkeyList:
    query_emb = testcembList[index]
    postingid_list = getNeighbours(query_emb)
    testpostingidList.append(val)
    testmatchesList.append(" ".join(postingid_list))
    index =index +1
    if index==100:
        break


%matplotlib inline
index =10
while index <20:
    print(testtitleList[index])
    print(testkeyList[index])
    imagePath = r'/content/drive/MyDrive/train_images'+ testimageList[index]
    pil_im = Image.open(imagePath, 'r')
    plt.figure()
    plt.imshow(pil_im)
    plt.show()
    matching_indices = testmatchesList[index].split(' ')
    print('==================')
    for ind in matching_indices:
        print(titleList[int(ind)])
        print(keyList[int(ind)])
        imagePath = r'/content/drive/MyDrive/train_images'+ imageList[int(ind)]
        pil_im = Image.open(imagePath, 'r')
        plt.figure()
        plt.imshow(pil_im)
        plt.show()
    index= index +1
```

6.1 TRAIN.CSV FILE

	A	B	C	D	E
1	posting_id	image	image_pha	title	label_group
2	train_12922	0000a68812	94974f937d	Paper Bag \	249114794
3	train_33862	00039780df	af3f9460c2	Double Tap	2.938E+09
4	train_22885	000a190fdd	b94cb00ed	Maling TTS	2.396E+09
5	train_24065	00117e4fc2	8514fc58ea	Daster Batil	4.093E+09
6	train_33691	00136d1cf4	a6f319f924	Nescafe \x	3.649E+09
7	train_24643	0013e7355f	bbd097a787	CELANA WA	2.661E+09
8	train_18025	00144a49c5	f815c9bb83	Jubah anak	1.835E+09
9	train_18061	0014f61389	eea7e1c0c0	KULOT PLIS	1.566E+09
10	train_86570	0019a3c675	ea9af4f483	[LOGU] Tem	2.36E+09
11	train_83168	001be52b2b	e1ce953d1a	BIG SALE SE	2.631E+09
12	train_15983	001d7f5d9a	bec8d09693	Atasan Raju	2.462E+09
13	train_24966	001e11145b	eab5c29596	PASHMINA	509010932
14	train_27717	001e11145b	eab5c29596	PASHMINA	509010932
15	train_99856	001f4c8331	d8a6082bb5	Lampu led	4.206E+09
16	train_42875	001f5580b0	dc85e17506	Charger VIZ	1.932E+09
17	train_41964	002039aaf8	e98c873acc	Korek Kupi	349297863
18	train_30090	0027aaf8dd	b3cccc26cc3	MARKS & SI	1.575E+09
19	train_30543	00286d2760	be85837b8(	Saffron 0,25	2.353E+09
20	train_29859	002f978c58	bf38f0e083	HnKfashion	3.416E+09
21	train_29613	00303ad1c0	e48d9b652(	Madame Gi	2.098E+09
22	train_99939	00324695e3	a734d84d34	Safi Derma	3.625E+09
23	train_22384	003524b707	ba35c44a3f	Kangaroo T	531768918

	A	B	C	D	E
34235	train_32964	ffda9710b7	af26e0f0d3	Gunting Ku	2.077E+09
34236	train_21769	ffdc88728f6	baacc99e34	BOLDe Supe	2.756E+09
34237	train_36688	ffdd054398	9ba6cc38a4	MAYCREATI	1.142E+09
34238	train_22870	ffdf26064e4	c5643f3119:	MMTOYS M	724975459
34239	train_14050	ffe09691e8	ea6c871852	Lock & Lock	2.231E+09
34240	train_99208	ffe71f38a13	ab91d46e91	Napoclean	2.1E+09
34241	train_28767	ffea625e5d	bbb5cc6394	Active Masl	2.413E+09
34242	train_18961	ffeaf56f2aa	b258cda722	Kepala Sow	132875370
34243	train_11971	ffec183c8e5	bcd7c2695d	Kurma Date	3.402E+09
34244	train_34457	ffec72f1201	ab37362b17	Johnson\xe	3.561E+09
34245	train_34804	ffec72f1201	ab37362b17	Johnson\xe	3.561E+09
34246	train_94581	fff12227503	b0cbce308f	Baterai Batı	3.092E+09
34247	train_40282	fff1c07ceef	e3cd72389f	Masker Bah	3.777E+09
34248	train_76905	fff40169137	be86851f72	MamyPoko	2.736E+09
34249	train_61497	fff421b78fa	ad27f0d08c	KHANZAAC	4.101E+09
34250	train_36309	fff51b87916	e3b13bd1d4	Kaldu NON	1.664E+09
34251	train_17921	ffffa0ab2ae	af8bc4b2d2	FLEX TAPE F	459464107
34252					
34253					
34254					

This train.csv file contains total 34251 records of data

7.TESTING

7.1 TESTING

Testing is the process of evaluation a software item to detect differences between given input and expected output. It is also to assess the features of the software item. Testing assesses the quality of the product. Software testing is a process that should be done during the development process after implementation (coding). In other words software testing is a verification and validation process.

- **Verification:** verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to.
- **Validation:** validation is the process to make sure the product satisfies the specified requirements at the end of the development phase. In other words, to make sure the product is built as per customer requirements.

7.1.1 BASICS OF SOFTWARE TESTING

In order to start the testing process the primary thing is requirements of software development cycle. Using this phase the testing phase will be easier for testing.
The capacity of the software can be calculated by executing the code and inspecting the code in different conditions such as testing the software by subjecting it to different sources as input and examining the results with respect to the inputs.

There are two basics of software testing: black box testing and white box testing:

- **Black box testing:** Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing.
- **White box testing:** White box testing is a testing technique that takes into account the internal mechanism of a system. It is also called structural testing and glass box testing.

Black box testing is often used for validation and white box testing is often used for verification.

7.1.2 FUNCTIONAL AND NON-FUNCTIONAL TESTING

- **Functional testing:** Defines the specified function of a particular code in the program.This type of testing gives us a brief description about the program's performance and security in the various functional areas.
- **Non-functional testing:** Defines the capabilities of particular software like its log data etc. It is opposite to functional testing and so will not describe the specifications like security and performance.

The performance of the particular program not only depends on errors in coding. The errors in the code can be noticed during execution, but the other types of errors can affect the program performance like when the program is developed based on one platform that may not perform well and give errors when executed in different platform. So, compatibility is another issue that reduces the software performance.

7.2 AIM OF TESTING

The main aim of testing is to analyze the performance and to evaluate the errors that occur when the program is executed with different input sources and running indifferent operating environments. There are different types of approaches for testing a .NET framework based application are;

- Unit testing
- Validation testing
- Integration testing
- User acceptance testing
- Output testing
- Black box and white box testing.

i. Unit Testing:

This is the approach of taking a small part of testable application and executing it according to the requirements and testing the application behavior. Unit testing is used for detecting the defects that occur during execution (MSDN, 2010).When an algorithm is executed, the integrity should be maintained by the data structures. Unit testing is made use for testing the functionality of each algorithm during execution. Unit testing reduces the ambiguity in the units.

In this project, I have developed an application using different phases like encryption, decryption. So for getting the correct output, all the functions that are used are executed and

tested at least once so as to making sure that all the control paths, error handling and control structures are in proper manner.

ii. **Validation Testing:**

Validation is the process of finding whether the product is built correct or not. The software application or product that is designed should fulfill the requirements and reach the expectations set by the user. Validation is done while developing or at the final stage of development process to determine whether it is satisfies the specified requirements of user. Using validation test the developer can qualify the design, performance and its operations. Also the accuracy, repeatability, selectivity, Limit of detection and quantification can be specified using "Validation testing" (MSDN, 2010).

iii. Integration Testing:

This is an extension to unit testing, after unit testing the units are integrated with the logical program. The integration testing is the process of examining the working behavior of the particular unit after embedding with program. This procedure identifies the problems that occur during the combination of units. The integration testing can be normally done in three approaches;

- Top-down approach
- Bottom-up approach
- Umbrella approach

(a) Top-down approach:

In the top-down approach the highest-level module should be considered first and integrated. This approach makes the high-level logic and data flow to test first and reduce the necessity of drivers. One disadvantage with top-down approach is its poor support and functionality is limited (MSDN, 2010)

(b) Bottom-up approach:

Bottom-up approach is opposite to top-down approach. In this approach, the lowest level units are considered and integrated first. Those units are known as utility units. The utility units are tested first so that the usage of stubs is reduced. The disadvantage in this method is that it needs the respective drivers which make the test complicated, the support is poor and the functionality is limited (MSDN, 2010).

(c) Umbrella approach:

The third approach is umbrella approach, which makes use of both the top – bottom and bottom - top approaches. This method tests the integration of units along with its functional data and control paths. After using the top - bottom and bottom-top approaches, the outputs are integrated in top - bottom manner.

The advantage of this approach is that it provides good support for the release of limited functionality as well as minimizing the needs of drivers and hubs. The main disadvantage is that it is less systematic than the other two approaches (MSDN, 2010).

iv. User Acceptance Testing :

This is the process of obtaining the confirmation from the user that the system meets the set of specified requirements. It is the final stage of project; the user performs various tests during the design of the applications and makes further modifications according to the requirements to achieve the final result. The user acceptance testing gives the confidence to the clients about the performance of system.

v. Output Testing:

After completion of validation testing the next process is output testing. Output testing is the process of testing the output generated by the application for the specified inputs. This process checks weather the application is producing the required output as per the user's specification or not. The "output testing" can be done by considering mainly by updating the test plans, the behavior of application with different type of inputs and with produced outputs, making the best use of the operating capacity and considering the recommendations for fixing the issues (MSDN, 2010).

vi. Black box and white box testing:

- **Black box testing:** Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing.
- **White box testing:** White box testing is a testing technique that takes into account the internal mechanism of a system. It is also called structural testing and glass box testing.

8.TIME COMPLIXITY

8.1 ON CPU

```
[17] print(np.shape(X_train))

     (20550, 4)

[19] import time
     text_embeddings={}
     image_embeddings={}
     start_time = time.time()
     for index,row in  X_train.iterrows():
         txt_emb = get_textEmbeddings(model,str(row[3]))
         imagePath = '/content/drive/MyDrive/train_images/'+row[1]
         img_emb = get_imageEmbeddings(img_model,imagePath)
         text_embeddings[row[0]] = txt_emb
         image_embeddings[row[0]] = img_emb
     end_time = time.time()
     print(str(end_time-start_time))


     13280.067488193512
```

✓ 3h 41m 20s completed at 6:10 PM

8.2 ON GPU

```
import time
text_embeddings={}
image_embeddings={}
start_time = time.time()
for index,row in  X_train.iterrows():
    txt_emb = get_textEmbeddings(model,str(row[3]))
    imagePath = '/content/drive/MyDrive/Colab Notebooks/MultiModalSearch-main/MultiModalSearch-main/shopee-product-matching/train_images/'+row[1]
    img_emb = get_imageEmbeddings(img_model,imagePath)
    text_embeddings[row[0]] = txt_emb
    image_embeddings[row[0]] = img_emb
end_time = time.time()
print(str(end_time-start_time))
```

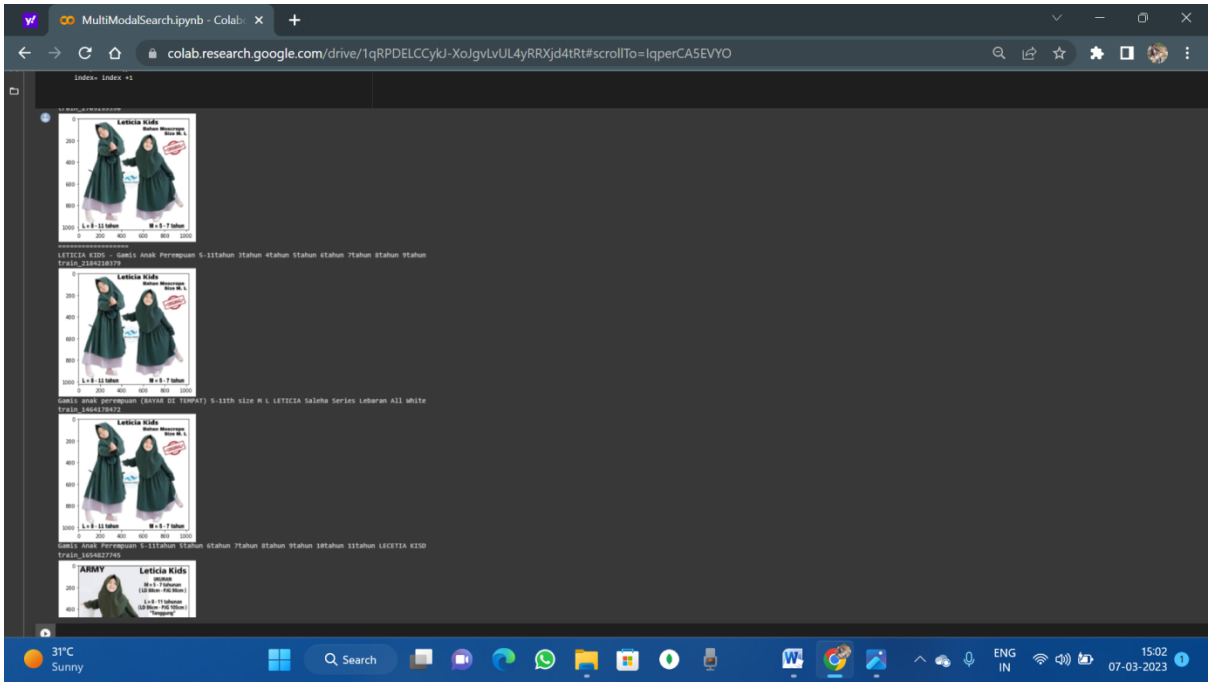
```
174.64926314353943
```

```
[ ] import pickle
    with open('./textEmb','wb') as handle:
```

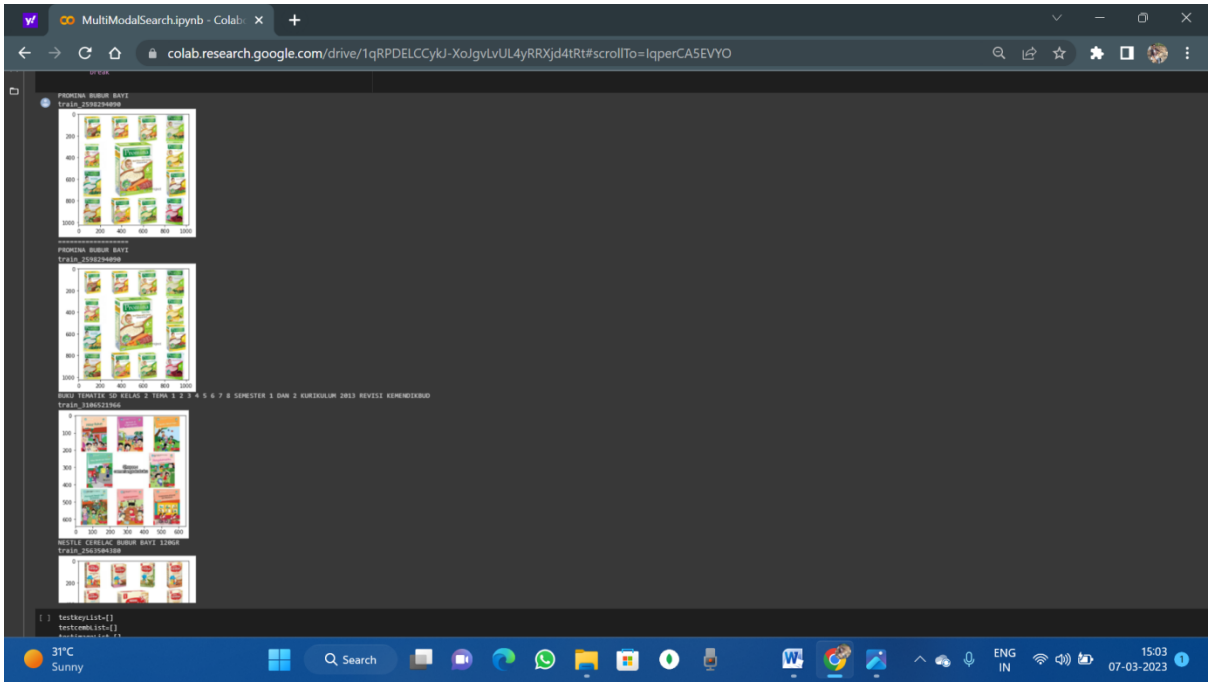
Automatic saving failed. This file was updated remotely or in another tab. Show diff

✓ 2m 54s completed at 10:39 AM

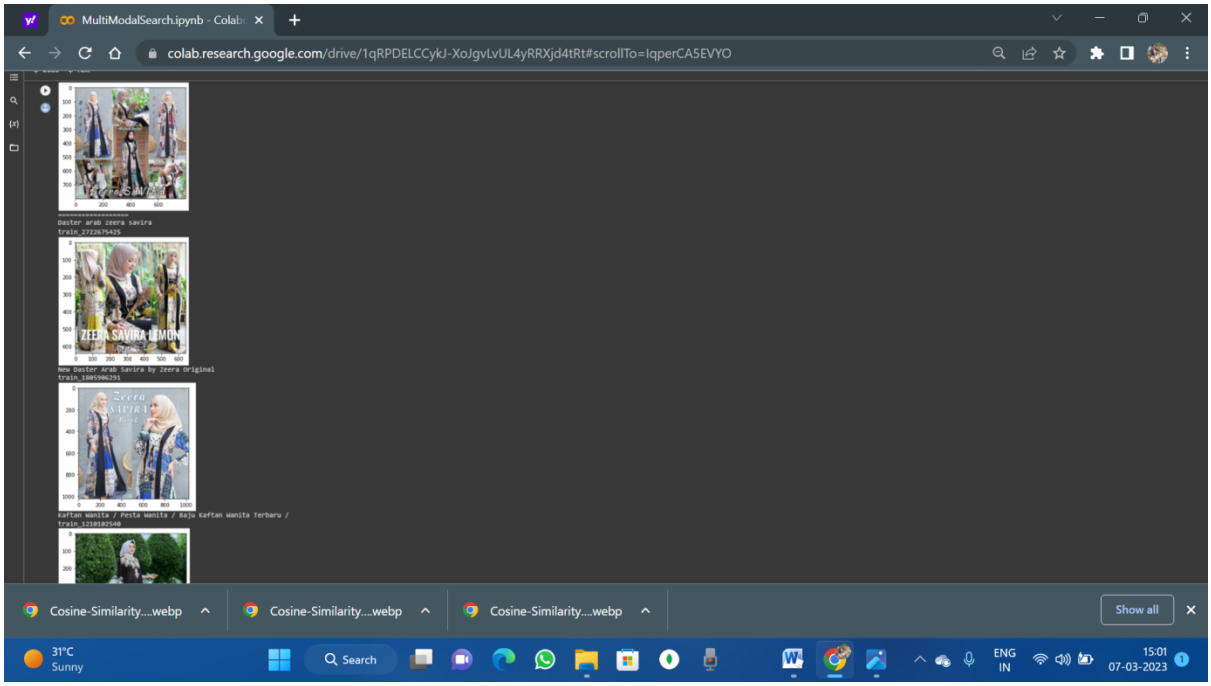
9.OUTPUT



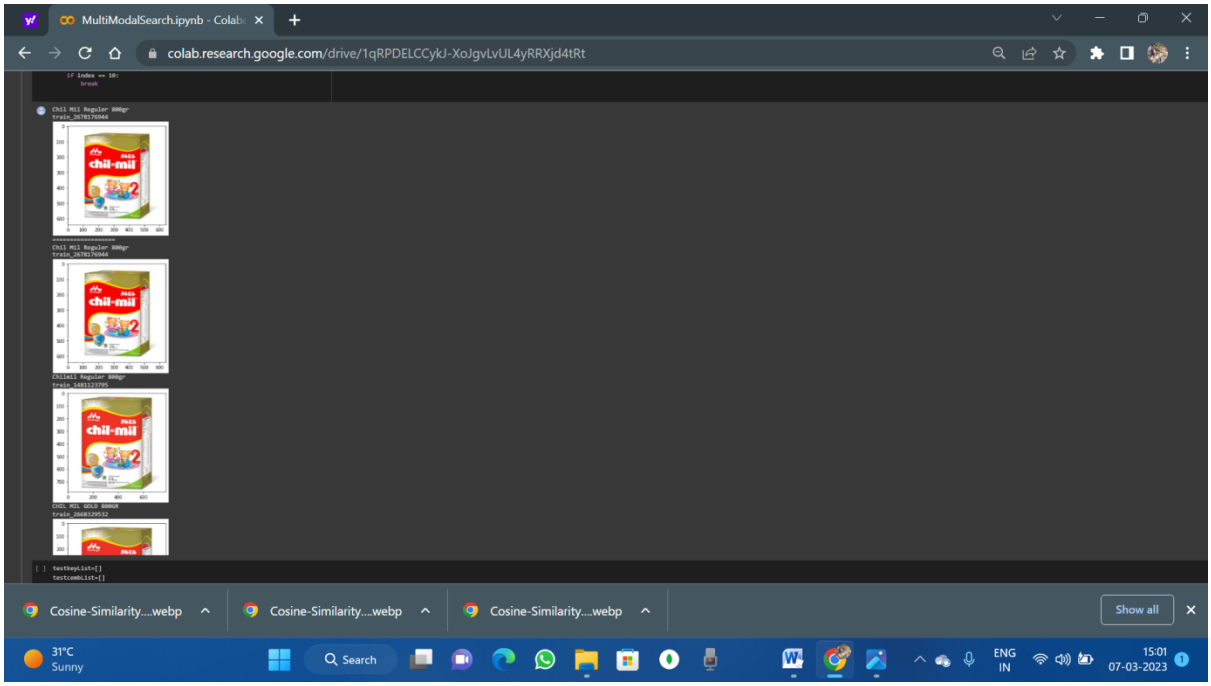
Here we can see the similar images of the girls dress which is in green color . The first image is given as input to the query and the other images are results for input.



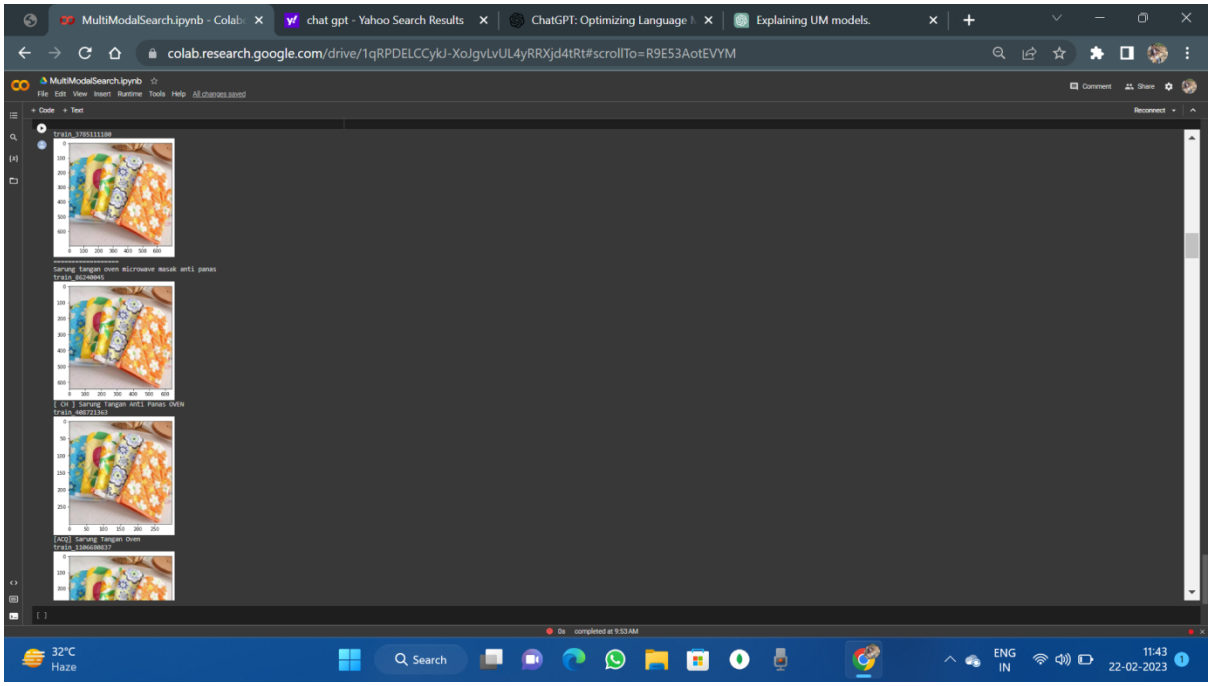
Here we can see the similar images of the food item. The first image is given as input to the query and the other images are results for input.



Here we can see the similar images of the girls dress which is in different colors . The first image is given as input to the query and the other images are results for input.



Here we can see the similar images of the chill-mil powder packet. The first image is given as input to the query and the other images are results for input.



Here we can see the similar images of a set of microwave covers. The first image is given as input to the query and the other images are results for input.

10.CONCLUSION AND FUTURE SCOPE

10.1 CONCLUSION

- Multimodal search is a powerful and growing area of research in the field of information retrieval. By combining different modalities, such as text, images, and videos, multimodal search can provide more accurate and relevant results than traditional text-based search methods.
- One of the key challenges in multimodal search is the need to effectively integrate and process data from multiple modalities. This can be achieved through the use of joint embedding, which allows for the learning of a shared representation space that captures the relationships between different modalities.
- Recent advances in deep learning and natural language processing have enabled the development of state-of-the-art models, such as BERT and GPT-3, that can be fine-tuned for specific NLP tasks and combined with other modalities in a joint embedding space. These models have shown great promise in improving the accuracy and efficiency of multimodal search applications.
- Overall, multimodal search is a rapidly evolving field with many exciting opportunities for research and development.

10.2 FUTURE SCOPE

The future scope of multi-model search is promising, as it has the potential to significantly improve the accuracy and relevance of search results. By combining multiple models, such as natural language processing, computer vision, and machine learning algorithms, multi-model search can provide a more comprehensive understanding of search queries and deliver more relevant results to users.

One area where multi-model search is expected to make a significant impact is in e-commerce. With the growth of online shopping, consumers are increasingly relying on search engines to find products that meet their needs. Multi-model search can help retailers provide more personalized recommendations to customers by taking into account their search history, browsing behavior, and preferences**.**

Overall, the future of multi-model search is promising, and it is likely to continue to grow in popularity as more businesses and industries recognize its potential benefits.

11.REFERENCES

[1] Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., ... & Dean, J. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv preprint arXiv:1609.08144.

[2] Karpathy, A., & Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 3128-3137).

[3] Kiela, D., & Bottou, L. (2014). Learning image embeddings using convolutional neural networks for improved multi-modal semantics. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 36-45).

[4] Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing, 13(4), 600-612.

[5] Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In 2009 IEEE Conference on Computer Vision and Pattern Recognition (pp. 248-255).IEEE.

[6] "Multi-Objective Bayesian Optimization for Neural Architecture Search" by Yingbo Zhou et al. (AAAI 2021) - This paper proposes a multi-objective Bayesian optimization approach for neural architecture search that can optimize multiple objectives simultaneously .

TEXT+IMAGE MULTIMODAL SEARCH USING MOBILENET

D.Hemanth Kumar[1], P.Naveen[2], P.Pavan Kumar[3]
Shaik Rafi[4]

[1,2,3.]Student, Department of CSE, Narasaraopeta Engineering Collage, Narasaraopeta, Guntur(D.T), Andhra Pradesh, India.

[4] Professor, Department of CSE, Narasaraopeta Engineering Collage, Narasaraopeta, Guntur(D.T), Andhra Pradesh, India.

hemanthkumardarsi4@gmail.com [1], pittanaveen3315@gmail.com [2], paramkusampavankumar1@gmail.com[3], shaikrafinrt@gmail.com[4]

ABSTRACT:

Multi-modal search is a task of retrieving relevant results from a database using multiple modalities such as text and images. The goal of the multi-modal search is to provide more accurate and comprehensive search results by integrating different types of data. This method is frequently employed across a number of industries, including e-commerce, healthcare, social media, and entertainment. The Multi-modal search requires the use of various techniques such as feature extraction, similarity measures, and machine learning algorithms. The Multi-modal search has grown in importance as a study topic in the fields of information retrieval and computer vision as a result of the expansion of multi-modal data availability.

1.INTRODUCTION:

The Multimodal search is a type of search technique where the system retrieves results based on the user's query using different modalities, such as text, image, audio, and video. It combines different modalities to achieve more accurate and relevant search results. In a typical multimodal search system, the input query may consist of text, image. The system extracts relevant features from each modality and combines them to form a joint feature representation. The joint representation is then used to retrieving the relevant results based on the user's query. For example, in an e-commerce website, the user may enter a query for a product, and the system may retrieve the results based on the text description of the product, as well as the image of the product.

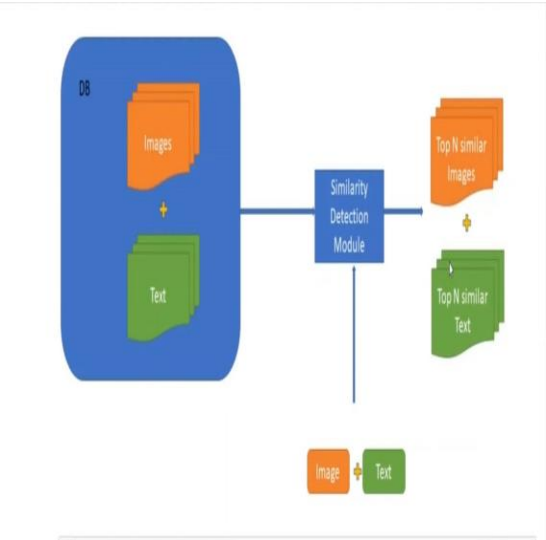


Fig 2(a) Architecture of the MultiModel Search

2.LITERATURE SURVEY:

Written by Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell, "Learning Multi-modal Similarity" is a research paper. This paper proposes a method for learning a similarity metric that can compare both textual and visual representations of objects. The method uses a combination of triplet loss and cross-modal matching to learn a joint embedding space for the two modalities.

"Multi-Modal Search with Image and Text Queries" by Yushi Jing and Katja Hofmann. This paper proposes a multi-modal search system that allows users to search for images using text queries and vice versa. The system uses a combination of deep learning and information retrieval techniques to perform the search.

"Multi-Modal Deep Learning for Image and Text Recognition" by Hyunjung Shin and Dongsuk Yook. This study suggests a deep learning paradigm for text and picture recognition that mixes recurrent and convolutional neural networks.

3. PROPOSED SYSTEM:

3.1 The Basic Mechanism:

Identify the Text and Image Datasets: The first step in building a multi-model search system for text and images is to identify the datasets that will be included in the search. These could include databases of text documents, image repositories, or other sources of information.

Data Preprocessing: Once the datasets have been identified, the data needs to be preprocessed. This involves cleaning and standardizing the data so that it can be easily searched.

Text Embedding: The text data needs to be Embedded so that it can be searched quickly and efficiently. This involves creating the vectors for the text data that can be used to search for specific terms or phrases.

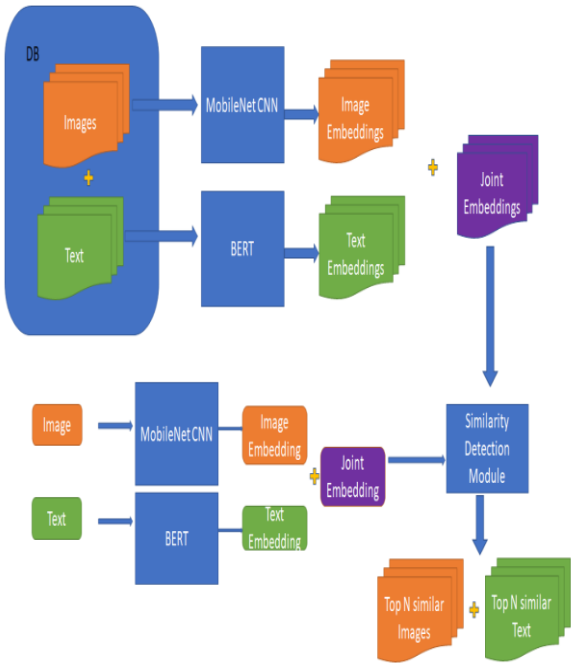
Image Embedding: The image data needs to be Embedded as well. In order to do this, features from the photos, such as colours, forms, and textures, must be extracted using computer vision algorithms. Afterwards, you may use these characteristics to look for related photographs.

Query Processing: When a user enters a search query, the system will process the query and search across both the text and image datasets. This involves using the text index to identify relevant text results and using the image features to identify

relevant image results. The results can then be ranked based on relevance.

Machine Learning: To improve the accuracy and relevance of search results, the machine learning techniques can be applied. To enhance text search results, for instance, natural language processing (NLP) models might be applied, while image recognition models can be used to improve image search results. Large volumes of data can be used to train these models to find patterns and relationships in the data that can be utilised to enhance search results.

3.2 Working Mechanism:



Here the Database contains both images and texts. Now we are using MobileNet CNN for the classification of Images and BERT for the Text Representation.

3.2.1 MobileNet CNN for Image Classification:

Convolutional neural network (CNN) architecture called MobileNet is made to be effective for embedded and mobile devices. This is accomplished by combining pointwise convolutions with depthwise separable convolutions.

Using depthwise separable convolutions, a standard convolution is split into two independent processes: a depthwise convolution, which applies a single filter to each input channel separately, and a pointwise convolution, which combines the output of the depthwise convolution using a 1x1 convolution.

As opposed to utilising a single filter for all input channels, depthwise convolution applies a distinct filter to each input channel, reducing the number of parameters in the network. This can significantly lower the convolution's computational cost.

The pointwise convolution essentially performs a linear transformation on the result of the depthwise convolution by combining it with a 1x1 convolution. As a result, the network can learn more intricate correlations between the input and output, which increases its expressive capacity.

Overall, to lower the number of parameters in the network and increase its efficiency for mobile and embedded devices, the MobileNet architecture combines depthwise separable convolutions with pointwise convolutions.

MobileNet can still achieve high accuracy on a variety of image recognition tasks despite its effectiveness.

A typical convolutional neural network (CNN) for image classification consists of several layers, including:

Input layer: The input image, which is commonly shown as a matrix of pixel values, is fed into this layer.

Convolutional layer: This layer applies to the input image a number of filters, each of which recognises a particular feature, such as edges or corners. The result of this layer is a collection of feature maps that show where each filter was activated in the image.

Activation layer: This layer provides non-linearity into the model and enables it to learn more complicated features by applying a non-linear activation function to the convolutional layer's output.

Pooling layer: This layer decreases the spatial dimensionality of the feature maps by Down sampling them, which helps to reduce the number of parameters in the model and prevent overfitting.

Dropout layer: During training, this layer randomly removes a portion of the neurons from the model, which aids in avoiding overfitting and enhancing generalisation.

Fully connected layer: Every neuron in the layer before and every neuron in the layer after are connected in this layer, enabling the model to learn intricate non-linear correlations between the features and the output.

Output layer: This layer generates the model's final output, which is often a probability distribution over dataset classes.

3.2.2 BERT for Text Classification:

A pre-trained language model called BERT (Bidirectional Encoder Representations from Transformers) was created by Google in 2018. For natural language processing (NLP) applications including text classification, question answering, and language translation, it is a sort of transformer-based neural network architecture.

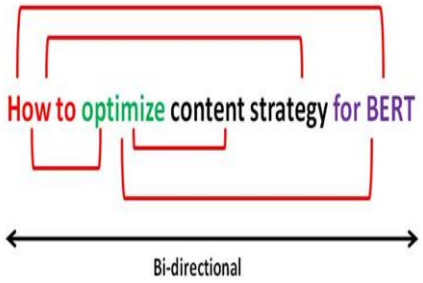
In contrast to conventional NLP models, which are trained on only one direction, BERT is dubbed bidirectional because it is trained on both the left and right context of each word in a sentence. This enables BERT to fully understand a sentence's context and generate more precise predictions.

The pre-training process of BERT involves training a large transformer-based neural network on a massive amount of text data, such as Wikipedia and other web sources. BERT can acquire a general knowledge of natural language and the relationships between words through this method, which can then be tailored for certain NLP tasks.

When using BERT for a particular NLP task, such as sentiment analysis or question answering, the pre-trained model is

refined using a smaller dataset that is tailored to the task. The pre-trained model's final layers are removed or added during fine-tuning, and the model is trained via back propagation on the task-specific dataset.

In a variety of NLP tasks, BERT has achieved state-of-the-art performance, and its pre-trained weights are publicly available, making it easy for researchers and developers to use it for their own NLP applications.



3.2.3 Embeddings:

Image Embedding:

The process of turning an image into a vector representation that can be fed into machine learning models is known as image embedding. Convolutional neural networks (CNNs) that have already been trained to extract high-level features from images, like VGG, ResNet, or MobileNet, can be used to create image embedding.

The process of image embedding involves passing an image through a pre-trained CNN, which extracts high-level features from the image. These features are then flattened and fed into a fully connected layer to generate a fixed-size vector representation of the image, which is also called an image embedding.

Text Embedding:

The process of transforming text into a numerical vector representation that may be fed into machine learning models is referred to as text embedding.Text embedding can be achieved using pre-trained language models such as BERT, GPT, or Word2Vec, which are trained to extract semantic information from text.

The process of text embedding involves passing the text through a pre-trained language model, which generates a sequence of contextualized embeddings for each word in the text. These embeddings capture the meaning and context of each word, as well as the relationships between words in the sentence.

Joint Embedding:

Joint embedding refers to the process of creating a shared vector space that can represent both images and text in a way that captures their relationships. This is achieved by combining image embedding and text embedding into a single vector space, where each image and text has a corresponding vector representation that Is close to similar images and text in terms of their meaning.

3.2.4 Similarity detection Module:

After performing all the above operations we store the joint embeddings for all the data. Now we follow the same strategy for the Input text and image which is given by the user. Now it's time to compare the similarity between the joint embedding from our dataset to the user provided text and image joint embedding.To find the similarity there are many different algorithms like Cosine similarity, Jaccard similarity, Levenshtein distance, Sequence alignment.

Here, similarity detection is accomplished using cosine similarity. A measure of similarity between two non-zero vectors in an inner product space is called cosine similarity. Between -1 and 1, it determines the cosine of the angle formed by the two vectors. When two vectors have a cosine similarity of 1, they are said to be identical, when they have a cosine similarity of 0, they are said to be orthogonal, and when they have a cosine similarity of 1, they are diametrically opposed.

In the domain of machine learning and natural language processing, cosine similarity is typically used to compare the similarity between two documents or chunks of text. In this instance, the vectors are produced by converting each document into a vector of word embeddings or a bag of words.

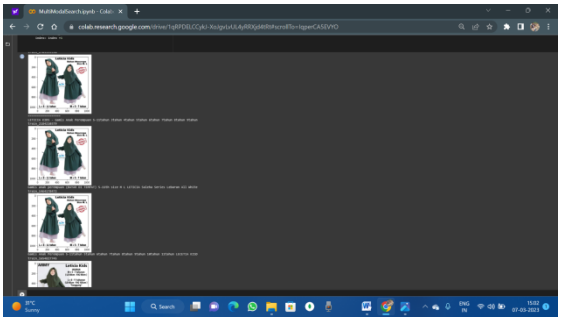
The Cosine Similarity increases as the papers' angles get closer (Cos theta).

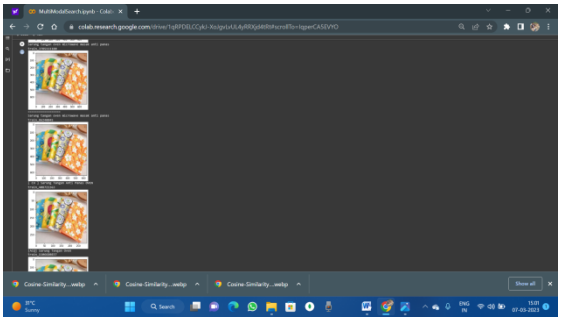
$$Cos\theta = \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|\,\|\vec{b}\|} = \frac{\sum_1^n a_i b_i}{\sqrt{\sum_1^n a_i^2}\sqrt{\sum_1^n b_i^2}}$$

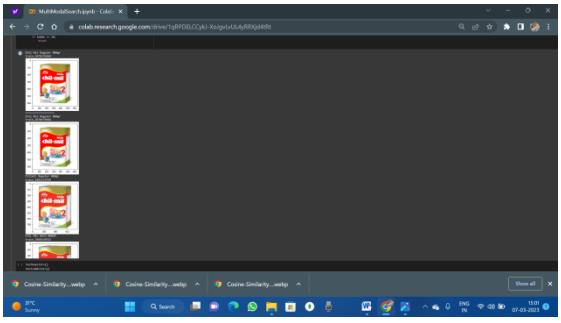
where, $\vec{a}\cdot\vec{b} = \sum_1^n a_i b_i = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n$ is the dot product of the two vectors.

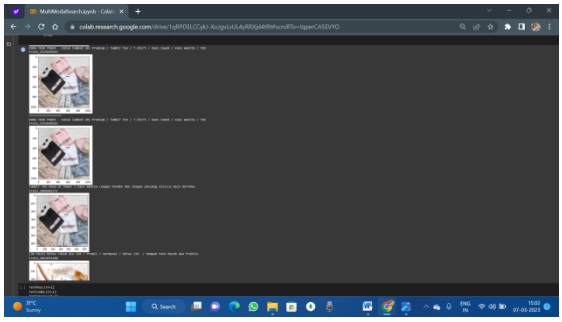
By checking the similarity between both joint embeddings from our database to user inputs we get top 'N' nearest results as output. For that we are using nearest neighbour 'kd-tree' algorithm.

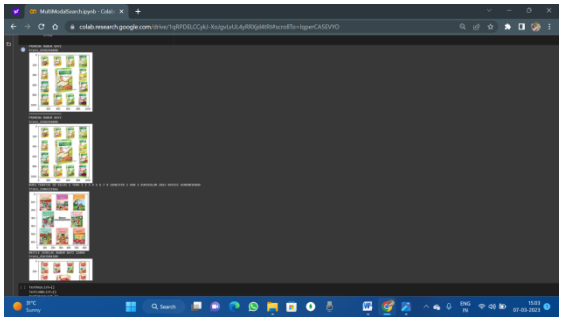
4. Result:











4. Conclusion:

Multi-model search combining text and image is a powerful technique that can enhance the accuracy and relevance of search results.By analyzing both visual and textual features, the multi-model search can provide a more comprehensive understanding of the content and context of a given query. Applications for multi-model text and picture search include visual search, image captioning, and recommendation systems. The quality of the individual models used to handle textual and visual data, as well as the efficacy of the methods used to combine them, are key factors in the success of multi-model search for text and images. The In general, multi-model search for text and images is a fascinating area of study that has the potential to greatly increase the precision and relevance of search results as well as open up new applications in a variety of fields.

5. References:

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., ... & Dean, J. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv preprint arXiv:1609.08144.

Karpathy, A., & Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 3128-3137).

Kiela, D., & Bottou, L. (2014). Learning image embeddings using convolutional neural networks for improved multi-modal semantics. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 36-45).

Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing, 13(4), 600-612.

Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In 2009 IEEE Conference on Computer Vision and Pattern Recognition (pp. 248-255).IEEE.

"Multi-Objective Bayesian Optimization for Neural Architecture Search" by Yingbo Zhou et al. (AAAI 2021) - This paper proposes a multi-objective Bayesian optimization approach for neural architecture search that can optimize multiple objectives simultaneously .

AB-7

by Vamshikrishna Namani

Submission date: 08-Mar-2023 04:29PM (UTC+1000)
Submission ID: 2031915112
File name: Base.docx (300.88K)
Word count: 2264
Character count: 12607

TEXT+IMAGE MULTIMODAL SEARCH USING TF MOBILENET,BERT

D.Hemanth Kumar[1], P.Naveen[2], P.Pavan Kumar[3]
Shaik Rafi[4]

[1,2,3.]Student, Department of CSE, Narasaraopeta Engineering Collage, Narasaraopeta, Guntur(D.T), Andhra Pradesh, India.

[4] Professor, Department of CSE, Narasaraopeta Engineering Collage, Narasaraopeta, Guntur(D.T), Andhra Pradesh, India.

hemanthkumardarsi4@gmail.com [1], pittanaveen3315@gmail.com [2], paramkusampavankumar1@gmail.com[3], shaikrafinrt@gmail.com[4]

1.ABSTRACT:

Multi-modal search is a task of retrieving relevant results from a database using multiple modalities such as text and images. The goal of the multi-modal search is to provide more accurate and comprehensive search results by integrating different types of data. This method is frequently employed across a number of industries, including e-commerce, healthcare, social media, and entertainment. The Multi-modal search requires the use of various techniques such as feature extraction, similarity measures, and machine learning algorithms. The Multi-modal search has grown in importance as a study topic in the fields of information retrieval and computer vision as a result of the expansion of multi-modal data availability.

2.INTRODUCTION:

The Multimodal search is a type of search technique where the system retrieves results based on the user's query using different modalities, such as text, image, audio, and video. It combines different modalities to achieve more accurate and relevant search results. In a typical multimodal search system, the input query may consist of text, image. The system extracts relevant features from each modality and combines them to form a joint feature representation. The joint representation is then used to retrieving the relevant results based on the user's query. For example, in an e-commerce website, the user may enter a query for a product, and the system may retrieve the results based on the text description of the product, as well as the image of the product.

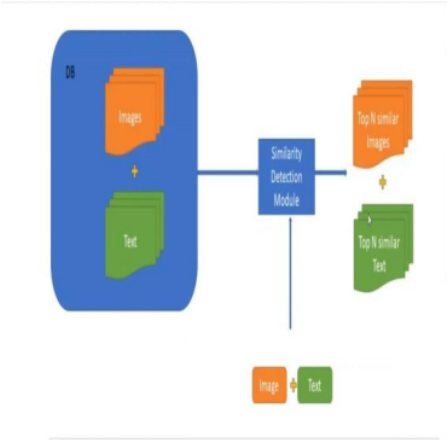


Fig 2(a) Architecture of the MultiModel Search

3.LITERATURE SURVEY:

Written by Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell, "Learning Multi-modal Similarity" is a research paper. This paper proposes a method for learning a similarity metric that can compare both textual and visual representations of objects. The method uses a combination of triplet loss and cross-modal matching to learn a joint embedding space for the two modalities.

"Multi-Modal Search with Image and Text Queries" by Yushi Jing and Katja Hofmann. This paper proposes a multi-modal search system that allows users to search for images using text queries and vice versa. The system uses a combination of deep learning and information retrieval techniques to perform the search.

"Multi-Modal Deep Learning for Image and Text Recognition" by Hyunjung Shin and Dongsuk Yook. This study suggests a deep learning paradigm for text and picture recognition that mixes recurrent and convolutional neural networks.

4. PROPOSED SYSTEM:

4.1 The Basic Mechanism:

Identify the Text and Image Datasets: The first step in building a multi-model search system for text and images is to identify the datasets that will be included in the search. These could include databases of text documents, image repositories, or other sources of information.

Data Preprocessing: Once the datasets have been identified, the data needs to be preprocessed. This involves cleaning and standardizing the data so that it can be easily searched.

Text Embedding: The text data needs to be Embedded so that it can be searched quickly and efficiently. This involves creating the vectors for the text data that can be used to search for specific terms or phrases.

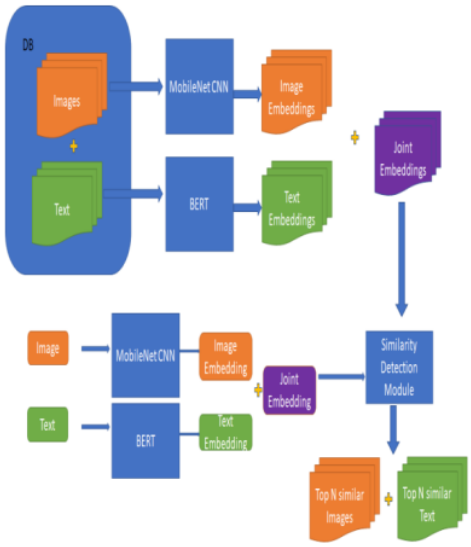
Image Embedding: The image data needs to be Embedded as well. In order to do this, features from the photos, such as colours, forms, and textures, must be extracted using computer vision algorithms. Afterwards, you may use these characteristics to look for related photographs.

Query Processing: When a user enters a search query, the system will process the query and search across both the text and image datasets. This involves using the

text index to identify relevant text results and using the image features to identify relevant image results. The results can then be ranked based on relevance.

Machine Learning: To improve the accuracy and relevance of search results, the machine learning techniques can be applied. To enhance text search results, for instance, natural language processing (NLP) models might be applied, while image recognition models can be used to improve image search results. Large volumes of data can be used to train these models to find patterns and relationships in the data that can be utilised to enhance search results.

4.2 Working Mechanism:



Here the Database contains both images and texts. Now we are using MobileNet CNN for the classification of Images and BERT for the Text Representation.

4.2.1 MobileNet CNN for Image Classification:

Convolutional neural network (CNN) architecture called MobileNet is made to be effective for embedded and mobile devices. This is accomplished by combining pointwise convolutions with depthwise separable convolutions.

Using depthwise separable convolutions, a standard convolution is split into two independent processes: a depthwise convolution, which applies a single filter to each input channel separately, and a pointwise convolution, which combines the output of the depthwise convolution using a 1x1 convolution.

As opposed to utilising a single filter for all input channels, depthwise convolution applies a distinct filter to each input channel, reducing the number of parameters in the network. This can significantly lower the convolution's computational cost.

The pointwise convolution essentially performs a linear transformation on the result of the depthwise convolution by combining it with a 1x1 convolution. As a result, the network can learn more intricate correlations between the input and output, which increases its expressive capacity.

Overall, to lower the number of parameters in the network and increase its efficiency for mobile and embedded devices, the MobileNet architecture combines depthwise separable convolutions with pointwise convolutions.

MobileNet can still achieve high accuracy on a variety of image recognition tasks despite its effectiveness.

A typical convolutional neural network (CNN) for image classification consists of several layers, including:

Input layer: The input image, which is commonly shown as a matrix of pixel values, is fed into this layer.

Convolutional layer: This layer applies to the input image a number of filters, each of which recognises a particular feature, such as edges or corners. The result of this layer is a collection of feature maps that show where each filter was activated in the image.

Activation layer: This layer provides non-linearity into the model and enables it to learn more complicated features by applying a non-linear activation function to the convolutional layer's output.

Pooling layer: This layer decreases the spatial dimensionality of the feature maps by Down sampling them, which helps to reduce the number of parameters in the model and prevent overfitting.

Dropout layer: During training, this layer randomly removes a portion of the neurons from the model, which aids in avoiding overfitting and enhancing generalisation.

Fully connected layer: Every neuron in the layer before and every neuron in the layer after are connected in this layer, enabling the model to learn intricate non-linear correlations between the features and the output.

Output layer: This layer generates the model's final output, which is often a probability distribution over dataset classes.

4.2.2 BERT for Text Classification:

A pre-trained language model called BERT (Bidirectional Encoder Representations from Transformers) was created by Google in 2018. For natural language processing (NLP) applications including text classification, question answering, and language translation, it is a sort of transformer-based neural network architecture.

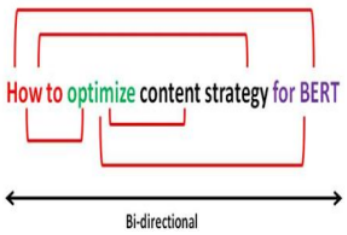
In contrast to conventional NLP models, which are trained on only one direction, BERT is dubbed bidirectional because it is trained on both the left and right context of each word in a sentence. This enables BERT to fully understand a sentence's context and generate more precise predictions.

The pre-training process of BERT involves training a large transformer-based neural network on a massive amount of text data, such as Wikipedia and other web sources. BERT can acquire a general knowledge of natural language and the relationships between words through this method,

which can then be tailored for certain NLP tasks.

When using BERT for a particular NLP task, such as sentiment analysis or question answering, the pre-trained model is refined using a smaller dataset that is tailored to the task. The pre-trained model's final layers are removed or added during fine-tuning, and the model is trained via back propagation on the task-specific dataset.

In a variety of NLP tasks, BERT has achieved state-of-the-art performance, and its pre-trained weights are publicly available, making it easy for researchers and developers to use it for their own NLP applications.



4.2.3 Embeddings:

Image Embedding:

The process of turning an image into a vector representation that can be fed into machine learning models is known as image embedding. Convolutional neural networks (CNNs) that have already been trained to extract high-level features from images, like VGG, ResNet, or MobileNet, can be used to create image embedding.

The process of image embedding involves passing an image through a pre-trained CNN, which extracts high-level features from the image. These features are then flattened and fed into a fully connected layer to generate a fixed-size vector representation of the image, which is also called an image embedding.

Text Embedding:

The process of transforming text into a numerical vector representation that may be fed into machine learning models is referred to as text embedding.Text embedding can be achieved using pre-trained language models such as BERT, GPT, or Word2Vec, which are trained to extract semantic information from text.

The process of text embedding involves passing the text through a pre-trained language model, which generates a sequence of contextualized embeddings for each word in the text. These embeddings capture the meaning and context of each word, as well as the relationships between words in the sentence.

Joint Embedding:

Joint embedding refers to the process of creating a shared vector space that can represent both images and text in a way that captures their relationships. This is achieved by combining image embedding and text embedding into a single vector space, where each image

and text has a corresponding vector representation that Is close to similar images and text in terms of their meaning.

4.2.4 Similarity detection Module:

After performing all the above operations we store the joint embeddings for all the data. Now we follow the same strategy for the Input text and image which is given by the user. Now it's time to compare the similarity between the joint embedding from our dataset to the user provided text and image joint embedding.To find the similarity there are many different algorithms like Cosine similarity, Jaccard similarity, Levenshtein distance, Sequence alignment.

Here, similarity detection is accomplished using cosine similarity. A measure of similarity between two non-zero vectors in an inner product space is called cosine similarity. Between -1 and 1, it determines the cosine of the angle formed by the two vectors. When two vectors have a cosine similarity of 1, they are said to be identical, when they have a cosine similarity of 0, they are said to be orthogonal, and when they have a cosine similarity of 1, they are diametrically opposed.

In the domain of machine learning and natural language processing, cosine similarity is typically used to compare the similarity between two documents or chunks of text. In this instance, the vectors are produced by converting each document into a vector of word embeddings or a bag of words.

The Cosine Similarity increases as the papers' angles get closer (Cos theta).

$$Cos\theta = \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|\,\|\vec{b}\|} = \frac{\sum_1^n a_i b_i}{\sqrt{\sum_1^n a_i^2}\sqrt{\sum_1^n b_i^2}}$$

where, $\vec{a}\cdot\vec{b} = \sum_1^n a_i b_i = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n$ is the dot product of the two vectors.

By checking the similarity between both joint embeddings from our database to user inputs we get top 'N' nearest results as output. For that we are using nearest neighbour 'kd-tree' algorithm.

5. The Conclusion:

Multi-model search combining text and image is a powerful technique that can enhance the accuracy and relevance of search results.By analyzing both visual and textual features, the multi-model search can provide a more comprehensive understanding of the content and context of a given query. Applications for multi-model

5. The Conclusion:

Multi-model search combining text and image is a powerful technique that can enhance the accuracy and relevance of search results.By analyzing both visual and textual features, the multi-model search can provide a more comprehensive understanding of the content and context of a given query. Applications for multi-model text and picture search include visual search, image captioning, and recommendation systems. The quality of the individual models used to handle textual and visual data, as well as the efficacy of the methods used to combine them, are key factors in the success of multi-model search for text and images. The In general, multi-model search for text and images is a fascinating area of study that has the potential to greatly increase the precision and relevance of search results as well as open up new applications in a variety of fields.

6. References:

[1] Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., ... & Dean, J. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv preprint arXiv:1609.08144.

[2] Karpathy, A., & Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 3128-3137).

[3] Kiela, D., & Bottou, L. (2014). Learning image embeddings using convolutional neural networks for improved multi-modal semantics. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 36-45).

[4] Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing, 13(4), 600-612.

[5] Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In 2009 IEEE Conference on Computer Vision and Pattern Recognition (pp. 248-255).IEEE.

AB-7

ORIGINALITY REPORT

13%	8%	7%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	dokumen.pub Internet Source	2%
2	Submitted to University of Nottingham Student Paper	1%
3	www.researchgate.net Internet Source	1%
4	Submitted to University of Glamorgan Student Paper	1%
5	erboristeriaverona.it Internet Source	1%
6	"Advances in Multimedia Information Processing – PCM 2018", Springer Science and Business Media LLC, 2018 Publication	1%
7	Submitted to Berlin School of Business and Innovation Student Paper	1%
8	V. Rekha, L. Venkateswara Reddy, Sachin Vasant Chaudhari, Arepalli Gopi, C. Nithiya,	1%

	Shaik Khaleel Ahamed. "Automated Deep Learning with Wavelet Neural Network based Rice Plant Classification", 2023 International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT), 2023 Publication	
9	Submitted to Morgan State University Student Paper	1 %
10	Submitted to University of Hertfordshire Student Paper	1 %
11	Gaihua Wang, Guoliang Yuan, Tao Li, Meng Lv. "An multi-scale learning network with depthwise separable convolutions", IPSJ Transactions on Computer Vision and Applications, 2018 Publication	<1 %
12	Jingwei Song, Mitesh Patel, Maani Ghaffari. "Fusing Convolutional Neural Network and Geometric Constraint for Image-based Indoor Localization", IEEE Robotics and Automation Letters, 2022 Publication	<1 %
13	Submitted to University of Strathclyde Student Paper	<1 %
14	ebin.pub Internet Source	<1 %

15	Jacob J. Senecal, John W. Sheppard, Joseph A. Shaw. "Efficient Convolutional Neural Networks for Multi-Spectral Image Classification", 2019 International Joint Conference on Neural Networks (IJCNN), 2019 Publication	<1 %
16	Submitted to Liverpool John Moores University Student Paper	<1 %
17	hal.archives-ouvertes.fr Internet Source	<1 %
18	pureadmin.qub.ac.uk Internet Source	<1 %
19	"Computer Vision – ECCV 2018", Springer Science and Business Media LLC, 2018 Publication	<1 %

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		


NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)
Approved by AICTE, Permanently Affiliated to JNTUK, Kakinada, NIRF Ranking (251-300 Band), Accredited by NBA (Tier-I) & NAAC with 'A+' Grade
Kotappakonda Road, Yellamanda (Post), Narasaraopet - 522601, Palnadu Dist., Andhra Pradesh, INDIA. Website:www.nrtec.in
PAPER ID
NECICAIEA2K23018
International Conference on
Artificial Intelligence and Its Emerging Areas
NEC-ICAIEA-2K23
17th & 18th March, 2023
Organized by Department of Computer Science and Engineering in Association with CSI
Certificate of Presentation
This is to Certify that Darsi Hemanth Kumar , Narasaraopeta engineering college has presented the paper title Text + Image multimodel search using mobilenet in the International Conference on Artificial Intelligence and Its Emerging Areas-2K23 [NEC-ICAIEA-2K23], Organized by Department of Computer Science and Engineeringin Association with CSI on 17th and 18th March 2023 at Narasaraopeta Engineering College, Narasaraopet, A.P., India.
Convenor
Dr.S.V.N.Srinivasu
Chief-Convenor
Dr.S.N.Tirumala Rao
Principal, Patron
Dr. M. Sreenivasa Kumar


NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)
Approved by AICTE, Permanently Affiliated to JNTUK, Kakinada, NIRF Ranking (251-300 Band), Accredited by NBA (Tier-I) & NAAC with 'A+' Grade
Kotappakonda Road, Yellamanda (Post), Narasaraopet - 522601, Palnadu Dist., Andhra Pradesh, INDIA. Website:www.nrtec.in
PAPER ID
NECICAIEA2K23018
International Conference on
Artificial Intelligence and Its Emerging Areas
NEC-ICAIEA-2K23
17th & 18th March, 2023
Organized by Department of Computer Science and Engineering in Association with CSI
Certificate of Presentation
This is to Certify that Pitta Naveen , Narasaraopeta engineering college has presented the paper title Text + Image multimodel search using mobilenet in the International Conference on Artificial Intelligence and Its Emerging Areas-2K23 [NEC-ICAIEA-2K23], Organized by Department of Computer Science and Engineeringin Association with CSI on 17th and 18th March 2023 at Narasaraopeta Engineering College, Narasaraopet, A.P., India.
Convenor
Dr.S.V.N.Srinivasu
Chief-Convenor
Dr.S.N.Tirumala Rao
Principal, Patron
Dr. M. Sreenivasa Kumar

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Approved by AICTE, Permanently Affiliated to JNTUK, Kakinada, NIRF Ranking (251-300 Band), Accredited by NBA (Tier-I) & NAAC with 'A+' Grade
Kotappakonda Road, Yellamanda (Post), Narasaraopet - 522601, Palnadu Dist., Andhra Pradesh, INDIA. Website:www.nrtec.in

International Conference on

PAPER ID
NECICAIEA2K23018

Artificial Intelligence and Its Emerging Areas

NEC-ICAIEA-2K23

17th & 18th March, 2023

Organized by Department of Computer Science and Engineering in Association with CSI

Certificate of Presentation

This is to Certify that Paramkusam Pavan Kumar , Narasaraopeta engineering college has presented the paper title Text + Image multimodel search using mobilenet in the International Conference on Artificial Intelligence and Its Emerging Areas-2K23 [NEC-ICAIEA-2K23], Organized by Department of Computer Science and Engineeringin Association with CSI on 17th and 18th March 2023 at Narasaraopeta Engineering College, Narasaraopet, A.P., India.

Convenor
Dr.S.V.N.Srinivasu

Chief-Convenor
Dr.S.N.Tirumala Rao

Principal, Patron
Dr. M. Sreenivasa Kumar

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Approved by AICTE, Permanently Affiliated to JNTUK, Kakinada, NIRF Ranking (251-300 Band), Accredited by NBA (Tier-I) & NAAC with 'A+' Grade
Kotappakonda Road, Yellamanda (Post), Narasaraopet - 522601, Palnadu Dist., Andhra Pradesh, INDIA. Website:www.nrtec.in

International Conference on

PAPER ID
NECICAIEA2K23018

Artificial Intelligence and Its Emerging Areas

NEC-ICAIEA-2K23

17th & 18th March, 2023

Organized by Department of Computer Science and Engineering in Association with CSI

Certificate of Presentation

This is to Certify that Shaik Rafi , Narasaraopeta engineering college has presented the paper title Text + Image multimodel search using mobilenet in the International Conference on Artificial Intelligence and Its Emerging Areas-2K23 [NEC-ICAIEA-2K23], Organized by Department of Computer Science and Engineeringin Association with CSI on 17th and 18th March 2023 at Narasaraopeta Engineering College, Narasaraopet, A.P., India.

Convenor
Dr.S.V.N.Srinivasu

Chief-Convenor
Dr.S.N.Tirumala Rao

Principal, Patron
Dr. M. Sreenivasa Kumar