Final report:

Generating cartoons from text using NLP and image processing

1.0 Problem statement:

1.1 Introduction:

In the contemporary landscape of scientific communication and presentations, the need for captivating and accessible visual aids is ever-increasing. Cartoons and model illustrations, with their ability to simplify intricate concepts and inject humor into technical subjects, have become vital in both academic and non-academic settings. In the realm of presentations and scientific discourse, cartoons and model illustrations have emerged as indispensable tools for effectively conveying complex ideas and concepts. These visual representations serve as powerful aids in elucidating intricate scientific principles, capturing the audience's attention, and facilitating knowledge retention. However, creating these visual assets traditionally involves laborious manual work, demanding considerable time and effort from scientists and presenters, which is a time-consuming task, diverting valuable resources that could otherwise be dedicated to research and development. To streamline and revolutionize the process of generating cartoons, our project endeavors to harness the power of cutting-edge technologies in the fields of Natural Language Processing (NLP) and Image Processing. By leveraging the advancements in artificial intelligence and machine learning, the aim is to develop an automated system capable of translating textual input into visually appealing and contextually relevant cartoon illustrations.

1.2 Objective:

The project's primary objective is to relieve scientists, educators, and presenters of the burden of manual cartoon creation, freeing them to focus on the core content of their work and communication. With the proposed solution, users will be able to input descriptive text or technical explanations, and the system will autonomously generate cartoons or model illustrations that accurately represent the intended message.

1.3 Criteria for success:

Scope of the Solution: The primary focus of the solution is to cater to the needs of scientists, educators, presenters, and communicators who frequently require visual aids to complement their technical explanations and presentations. The system will be designed to handle various types of input, ranging from descriptive text to technical specifications, and translate them into contextually relevant and visually appealing cartoons.

Key aspects of the solution's scope include:

1. Text-to-Cartoon Translation: The system will take textual input as its primary source of information. It will employ NLP techniques to understand the input text, extract relevant context and key elements, and convert them into cartoon illustrations.

2. Cartoon Customization: The solution will strive to provide a level of customization, allowing users to specify certain details or preferences for the generated cartoons, such as style, color schemes, or characters.

3. Image Processing for Artistic Rendering: Advanced image processing algorithms can be utilized to create aesthetically pleasing cartoon illustrations with attention to detail and artistic rendering.

4. Animation Capabilities: In addition to static cartoons, the system can be aimed to support the generation of simple animations or model illustrations to further enhance the communication of dynamic processes or concepts.

5. Cross-Domain Applicability: The solution will be designed to be versatile and applicable across various domains, accommodating the cartoon generation needs of users from different scientific fields and subject areas.

Constraints:

1. Complexity of Input Interpretation: Translating natural language into accurate and contextually appropriate visual representations is a complex task. While the system will strive to be as accurate as possible, it may face challenges in understanding ambiguous or highly technical language.

2. Artistic Limitations: While the image processing algorithms will be designed to create visually appealing cartoons, the system's creative output may still be constrained by the limitations of AI-generated art. Achieving the same level of artistic expression as human artists may not be feasible.

3. Training Data Availability: The quality and diversity of training data available for both NLP and image processing models will significantly influence the system's performance. Insufficient or biased data may impact the accuracy and generalization of the cartoon generation process.

6. Real-Time Generation: Real-time cartoon generation from large or complex texts may pose challenges due to the time-consuming nature of the processing involved.

2.0 Key Data Sources and acquisition:

An image library was constructed from online vector clipart. The following are some of the sites that were used:

1. Vector Stock Websites: Websites like Shutterstock, Adobe Stock, and iStock provide a vast collection of vector clipart images. These platforms cater to designers and illustrators, offering high-quality vector graphics that can be downloaded for various purposes, including use in presentations and animations.

2. Pixabay and Freepik: Pixabay and Freepik are popular platforms that offer a mix of both raster and vector graphics. They provide a wide selection of vector clipart images that are available for free under Creative Commons licenses.

3. Vector Illustration Websites: Several websites are dedicated exclusively to vector illustrations and clipart. Examples include Vecteezy, Vectorportal, and VectorStock. These platforms focus on providing vector-based graphics suitable for a wide range of projects, including scientific presentations and model illustrations.

4. Public Domain Vector Art: Certain websites curate public domain vector art, which is free from copyright restrictions. Websites like Public Domain Vectors and Openclipart offer collections of vector clipart images that can be freely used for any purpose.

Some of the data were pre-labeled. The rest were manually labeled and curated.

3.0 Methodology:

1) Collect and preprocess data: Gather a dataset of cartoons and their corresponding descriptions or captions. I have scraped online to gather a vast number of cartoons. On top of mentioned databases, image downloader chrome plugin was used to collect some of the data. The cartoons collected as such had their file name as the key and hence not much cleaning was necessary.

2) Train an NLP model: There are multiple ways to deal with the text recognition. for example, we can take advantage of a pre-trained language model or train our own using a deep learning framework like TensorFlow or PyTorch. In this project we will be first try a pre-attention simple NLP model. Next, we'll be using pre-trained models such as GPT or BARD using their API.

3) Train a computer vision model: Again, multiple ways to do it such as using a pre-trained image recognition model like VGG or Inception, or train our own model if we have a large labeled dataset. For this project we'll try a pre-trained model from Hugging Face.

4) Design an architecture that combines the textual and visual information.

5) Evaluate and refine

4.0 Data wrangling:

Discovery, structuring and cleanup: The image libraries were sub-grouped based on databases. 1) emojis, 2) font clipart for HTML web building 3) science related images. At first, the science related images were selected for model building. Within that cohort a collection of bioscience related images existed. This image library contained 7044 files, of which after curation around 3500 images were high quality. High quality images were classified into 9 classes. They are: 1) Cells_tissue: images of whole cells and cell composite (organs and/or tissues) 2) DNA: DNA structure and its associated processes 3) Lab_equipments: laboratory equipment, instruments and processes 4) Macromolecules: intracellular and molecular polymeric structures barring DNA and RNA, such as protein, lipid, carbohydrates 5) Metabolites: cellular organic structures and chemical structures such as ATP, AMP etc. 6) Organelles: intracellular organelles such as Golgi bodies, autophagy and its associated processes 7) Parts: biological body parts with specific functions (similar to ‘Cells_tissue’ class) such as leaves, acorn etc. 8) RNA: RNA structure and its associated processes 9) Species: whole animal or plant species

Image libraries were all converted as png and their file extensions were removed for image name comparison using os module’s path.splitext() function. There were some issues with Windows and Mac based dependency and package compatibility (especially during SVG to PNG conversion). The codes were run in the accordingly operating systems for the appropriate functionalities.

Exploratory data analysis (EDA): An initial image name matching strategy was explored. Image package from PIL was used to open the images. File system navigation was performed using os module. Images were plotted using matlab.pyplot and for animation using tkinter.

5.0 Modeling:

The following 3 strategies and their feasibilities were explored:

1. Use a natural language processing (NLP) parts of speech algorithm for user input to call images using text-based image labels (located in file name): NLTK and Stanford’s POS-tagger was used.

Words were tokenized and lemmatized. For images only nouns were chosen and for actions on the image's verbs were chosen.

2. Use a pretrained model (e.g., CLIP, GLIDE, BIOBERT and DALL-E) for both text to image generation: This method was not very useful due to API cost. Also, BIOBERT has been discontinued.

3. De-novo training of an image and text embedding followed by fusing them onto an adversarial network architecture

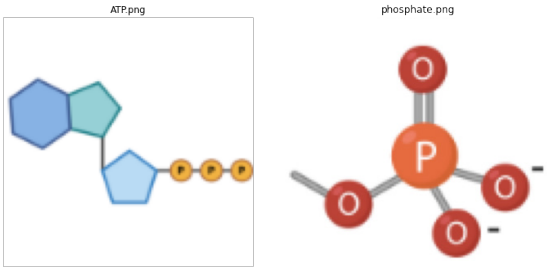

Figure 1: Image generation on canvas for text input - 'ATP and phosphate'

a

	a girl wearing a beanie	a chemical structure
image1	0.00	100.00
image2	0.00	100.00

b

	ATP	Phosphate
image1	22.30	77.70
image2	1.79	98.21

Figure 2: Granularity of image classification. (a) Image granularity was better for subjects with distinct features but fails (b) for images with chemical structure variations

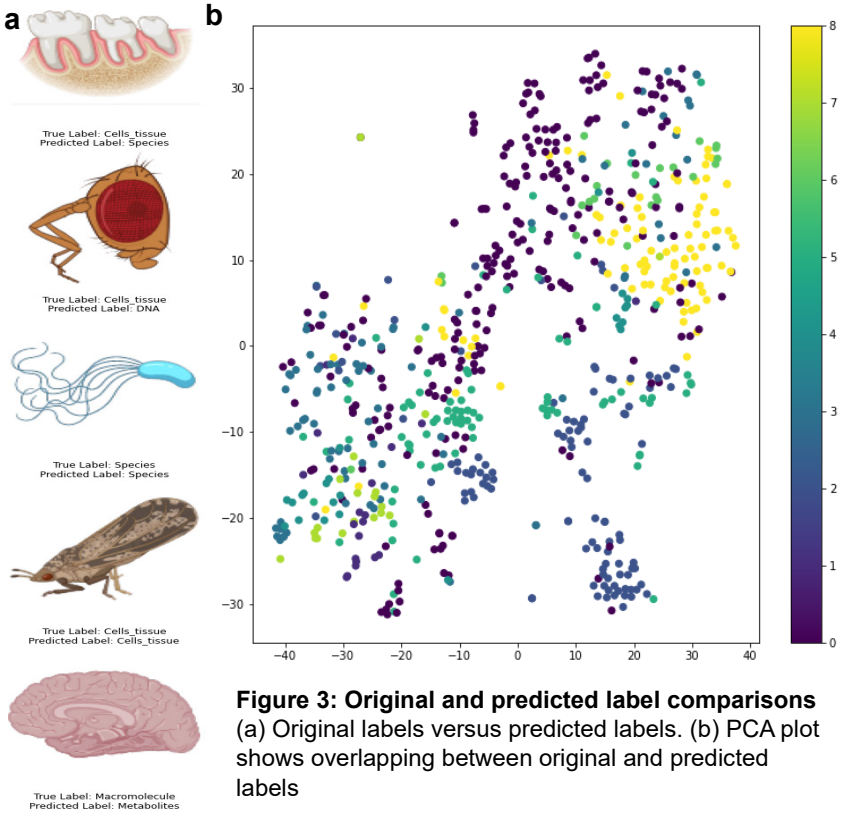

Figure 3: Original and predicted label comparisons (a) Original labels versus predicted labels. (b) PCA plot shows overlapping between original and predicted labels

Image-text matching based code was able to plot the images on a blank canvas (Figure 1). Image classification using pretrained image and text model CLIP performed better on distinct objects, but failed to distinguish between similar chemical structures such as ATP and phosphate (Figure 2). This indicates that the pre-trained models were not sufficiently adept at the resolution of interest.

For de-novo classification I used a transformer 2 based model (from HuggingFace). The image classification accuracy was 70% (Figure 3). The accuracy could have been improved by getting rid of non-overlapping labeling.

However, the amount of data needed per classification was much greater than available. Also the model classification is hardly as granular as was necessary; definitely not better than CLIP.

On the other hand, for text embeddings, for meaningful biological texts, I used both a bioscience dictionary pdf and converted their OCR into texts. I also used just a subset of 'Glycolysis' related text from Wikipedia using Beautiful Soup. Words were lemmatized and the data was represented as a network and a logo respectively. The prepositions were not taken out and hence they present were overwhelmingly greater in the number counts of lemmatized words.

Since the image training itself was not granular enough, I did not further finetune and refine the word embeddings. But a adversarial network based on this crude text and image embedding fusion failed to produce anything meaningful.

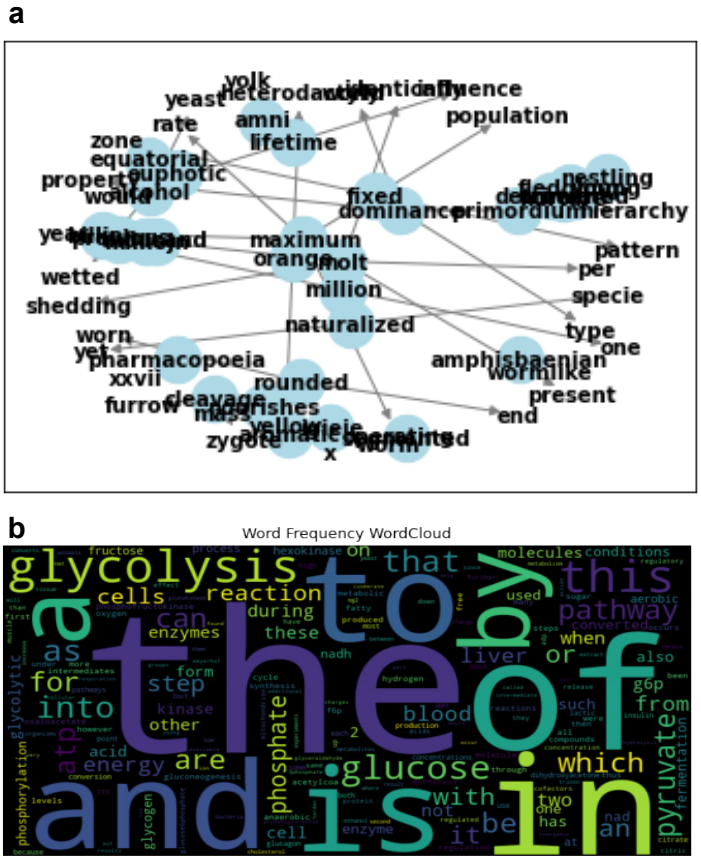

Figure 4: Texts embeddings using dictionary (a) and beautiful soup (b).

For the final model, the first approach was adopted where generally image labeling was matched with the text input based on similarity.

6.0 Results:

Three layers of similarity was measured. 1) Exact match: If the lemmatized input words exactly matched the image label 2) Synonym: If the word matched a synonym (from wordnet) 3) Present in the file name: If the word was present in the file name.

These were further finetuned case by case basis and exceptions were put in place. For example, with the present in file name option a lot of false positive images were put in place. And hence a priority based matching was established using the 'if….else if' conditionals.

To recommend more than one image, text matches were indexed and sequentially kept in a list based on the priority. And then during image generation a combination of these indexed texts were used to pull up the images with best match (Figure 4). The same technique was used for further analysis in other image datasets such as Fontawesome dataset for emojis and html web building clip arts (Figure 6).

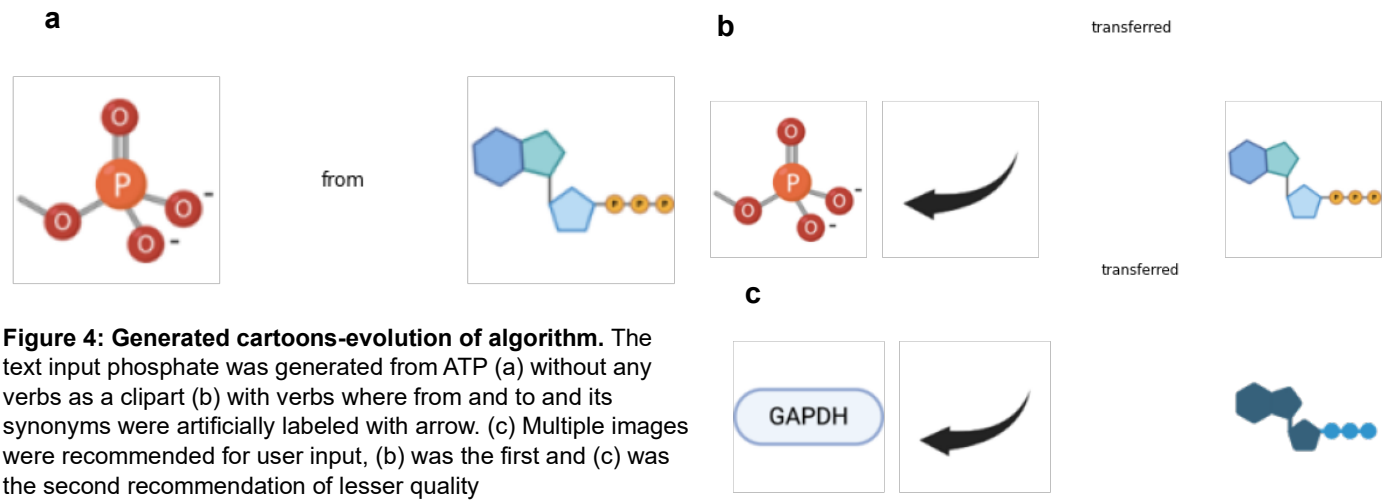

Figure 4: Generated cartoons-evolution of algorithm. The text input phosphate was generated from ATP (a) without any verbs as a clipart (b) with verbs where from and to and its synonyms were artificially labeled with arrow. (c) Multiple images were recommended for user input, (b) was the first and (c) was the second recommendation of lesser quality

Figure 6: Emojis and clipart for input text (sad)

7.0 Future directions: There are lost of scope for improvement for this project. The finalized strategy can further be modified and put in more case scenarios. The code can also be customized for further databases.

For the other strategies, bigger datasets are the obvious start point. API cost being an issue, using pretrained transformer 2, which is free of cost, is still an option. However, this may not be fitting for such a granular task and hence need to be further tuned. For each imaged non-overlapping labeling would be necessary. Also, multiple images for the same label is needed. Fr text embeddings, a good start point would be to limit the words to nouns and verbs, since those are the words that needs to be out in place in the cartoon generation.