

Translatoron:

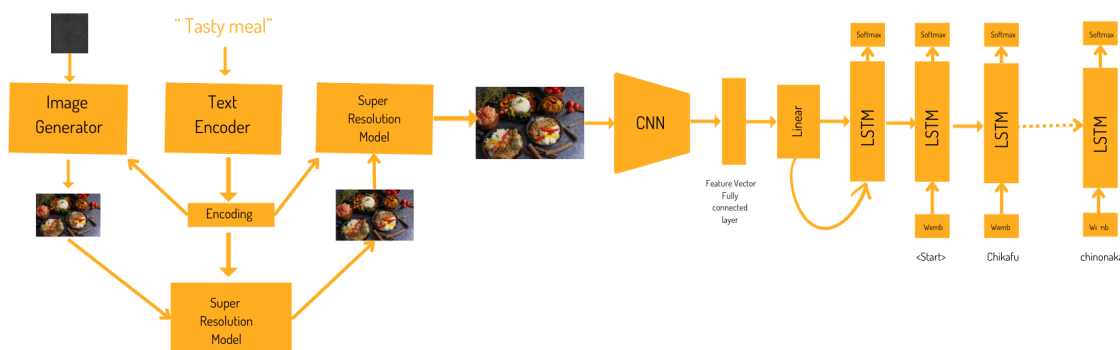
Bidirectional Text-to-Image Translation System

Project Description

Translatoron is a bidirectional text-to-image machine translation system that allows for seamless translation between images and text which is an extremely useful feature in building universal language systems. Translatoron is inspired to solve the limited availability of machine translation systems for low-resource languages especially those in Africa. Through the use of an architecture that combines a diffusion model and a CNN-LSTM model, the translatoron, makes it possible for the first time to use dynamic mono-lingual datasets that simply describe pictures in one language to optimize a universal language model. This is extremely important as it lowers the required skill level for data creation for machine translation systems for low-resource languages.



High-level overview



Low-level overview

Youtube Link

<https://youtu.be/UO5VdlcAHTc>

Model Development

This project was developed using Jupyter Notebook and Google Colab. It is preferable to use Google Colab TPU or GPU for best performance. If you use CPU then it's advisable to use Jupyter Notebook for best performance. The CNN-LSTM model is trained on the Flickr8K dataset and the diffusion model uses the Stable Diffusion opensource architecture. The web-application

Scope

Even though the end objective of this project is to create a universal translation system this project will specifically focus on the foundational unit of the translation system which is the translatron: the image-to-text and text-to-image conversion model. We will aim to prove that it is possible to convert textual data into an image format, and be able to retrieve the textual meaning from the generated image. If this is proved, by induction it is possible to retrieve any other language from the hidden layer image interface, and therefore to convert between any two sets of languages.

Setting Up The Notebook

Find the link to the full project [here](#).

TextToImage Model

```
!pip install pillow
```

```
from pathlib import Path

import tqdm

import torch

import pandas as pd

import numpy as np

from diffusers import StableDiffusionPipeline

from transformers import pipeline, set_seed

import matplotlib.pyplot as plt

import matplotlib.pyplot as plt

import cv2
```

ImageToText Model

```
import numpy as np
from PIL import Image
import os
import string
import tensorflow as tf
from pickle import dump
from pickle import load
from keras.applications.xception import Xception # to get pre-trained model Xception
from keras.applications.xception import preprocess_input
from keras.preprocessing.image import load_img
from keras.preprocessing.image import img_to_array
from keras.preprocessing.text import Tokenizer # for text tokenization
from keras.preprocessing.sequence import pad_sequences
from keras.utils import to_categorical
from keras.models import Model, load_model
from keras.layers import Input, Dense # Keras to build our CNN and LSTM
from keras.layers import LSTM, Embedding, Dropout
from tqdm import tqdm_notebook as tqdm # to check loop progress
tqdm().pandas()
```

```
from keras.layers import Add
```

Technical Requirements

Hardware:

- A high level computer system containing an Nvidia 3050 GPU with 4GB of video memory, an intel i7 octacore CPU and GPU with 28 GB of ram.
- CUDA

Software requirements:

- Tensorflow
- Keras
- Huggingface
- GPT-2
- Stable-diffusion-2
- Google Colab/Jupyter Notebook

End Product

End-Product: Web hosted conversion system between images and text

Development Platform: Google Colab/Jupyter Notebook

Data: The trained models where trained on the Flickr8K dataset

Data Preparation: Due to the complexity of the project and the intense resources required we did not discriminate between training images. However for future iterations, specific images can be selected from dataset that have a specific relation to Africa to make a second Afro-specific dataset. This will help compare results in the reverse phase of converting from low resource languages back to the high resource languages.

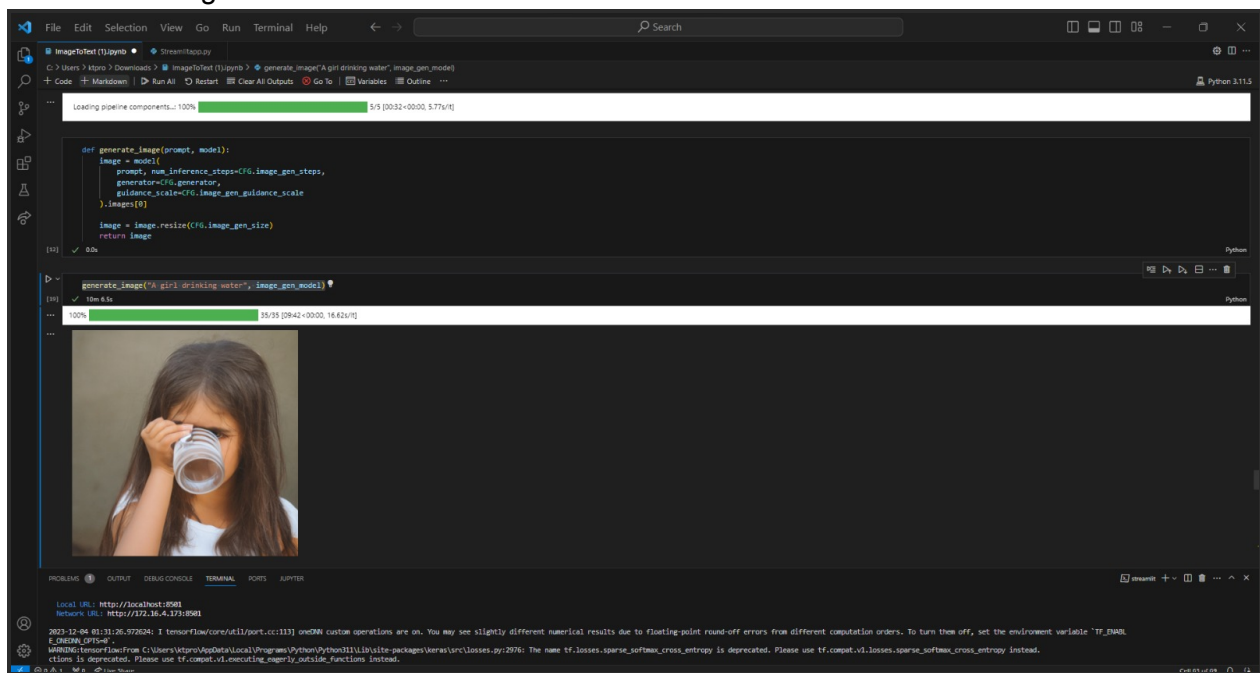
Installation and Running the Application

1. Download the model_mtbest_221123.hdf5, image_gen_model.pkl, tokenizer.p, features.p models from the google drive and save them locally in the same location
2. Download the Streamlit app from the google drive into the same location.
3. Modify the path you want your generated image to be saved in, for example :
Output_path= "C:\Users\ktpro\Downloads\archive\img4"
Img4 is the name you of image you want to save
4. Img_path = "C:\Users\ktpro\Downloads\archive\img4"
The im_path and the output_path should be the same
5. Open the terminal and navigate to the location of the application
6. Run the command : streamlit run Streamlitapp.py
7. You should be setup

Usage

Notes

Generated images




Generated annotation

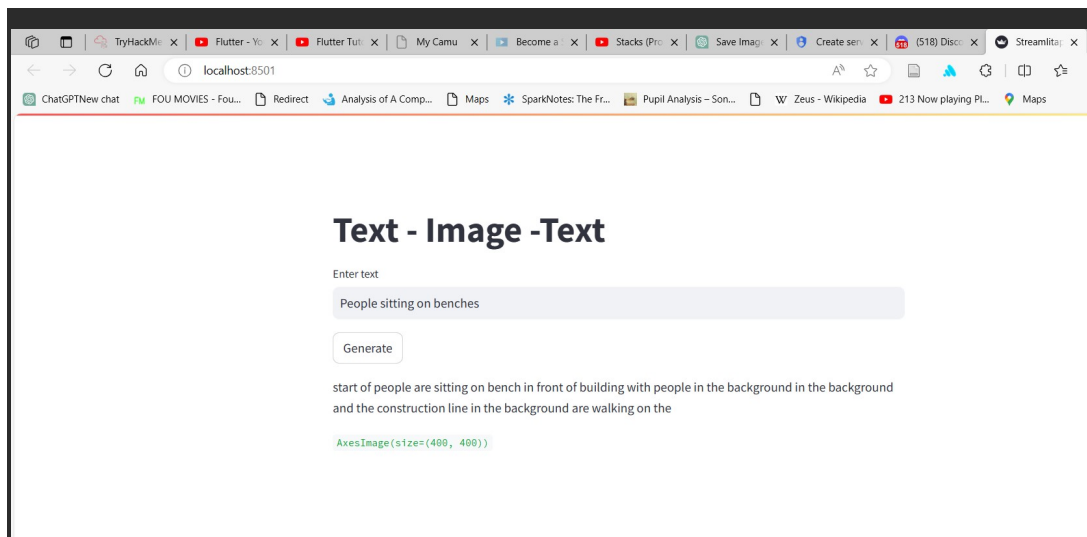
```
img_path = "C:\\Users\\ktpu\\Downloads\\archive\\images\\99172998_7c080ceef.jpg"
max_length = 20
tokenizer = load(open("C:\\Users\\ktpu\\Downloads\\archive\\tokenizer.g", "rb"))
model = load_model("C:\\Users\\ktpu\\Downloads\\archive\\models\\model_atbest_221123.h5")
xception_model = Xception(include_top=False, pooling="avg")
photo = extract_features(img_path, xception_model)
img = Image.open(img_path)
description = generate_desc(model, tokenizer, photo, max_length)
print(description)
plt.imshow(img)
```

1/1 [=====] - 1s 65ms/step
on
start of skier is skiing down snowy mountain over snow covered mountain range in the background and red sled in the background and two people on the snow in the

outplotlib.image.AxesImage at 0x2401a5ac3d0



Web App



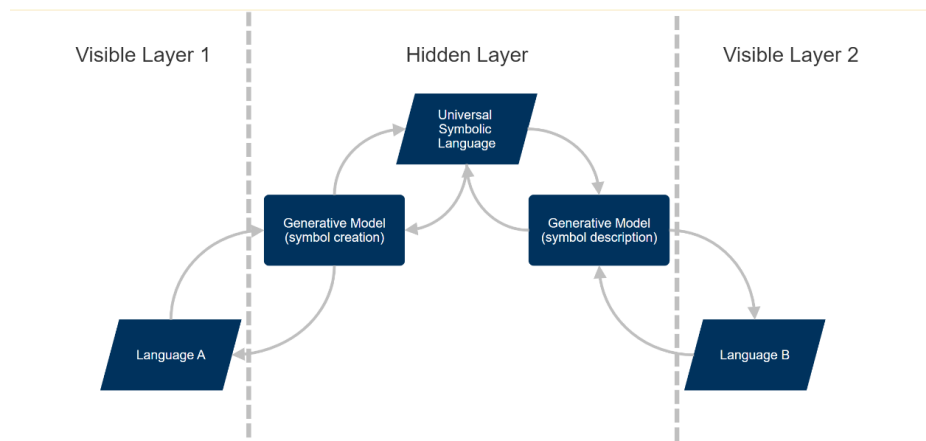
Contributing

Pull requests are welcome. For major changes, please open an issue first to discuss what you would like to change.

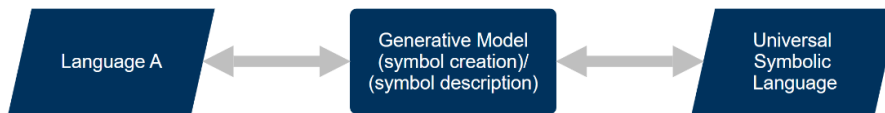
Please make sure to update tests as appropriate.

Model Architecture Visualizations

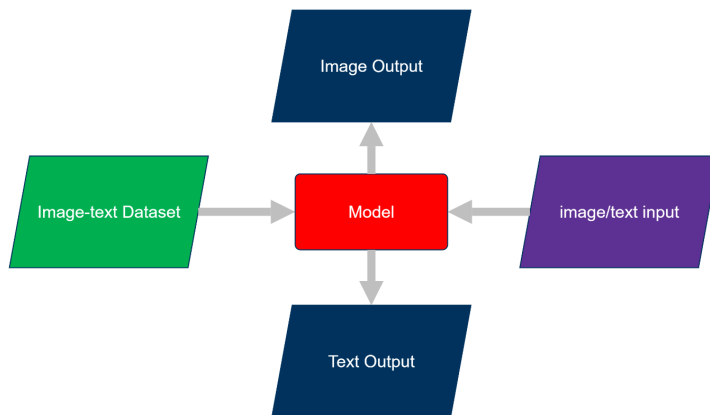
Full bidirectional model



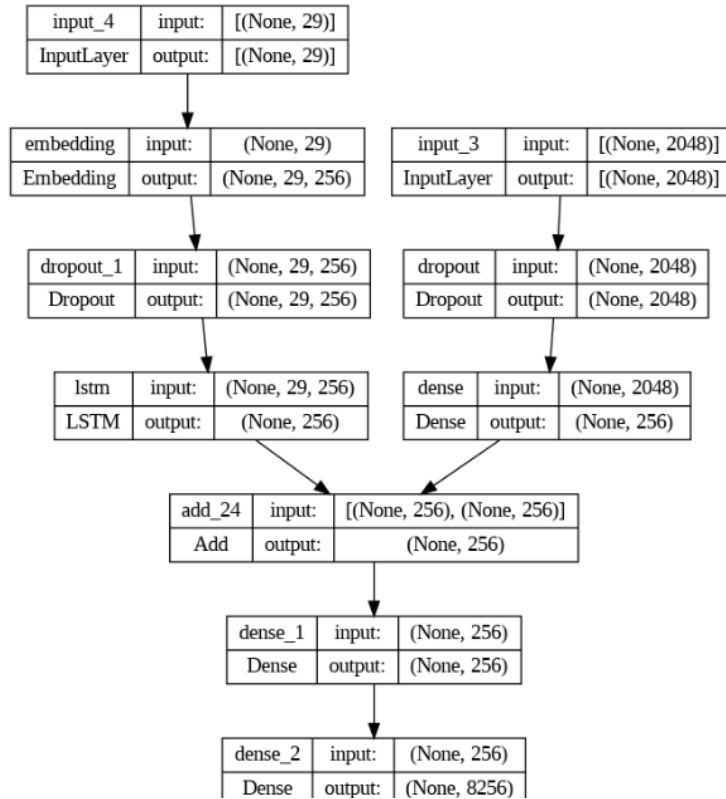
High-level diagram of the Translateron



Conceptual Visualization



Layerwise feature combination



References

Srinivasan, K., Raman, K., Chen, J., Bendersky, M., & Najork, M. (2021). WIT: Wikipedia-based Image Text Dataset for Multimodal Multilingual Machine Learning. arXiv (Cornell University). <http://arxiv.org/pdf/2103.01913.pdf>

Team, N., Costa-Jussà, M. R., Cross, J. H., Çelebi, O., Elbayad, M., Heafield, K., Heffernan, K. S., Kalbassi, E., Lam, J., Licht, D. J., Maillard, J., Sun, A., Wang, S., Wenzek, G., Youngblood, A., Akula, B., Barrault, L., Gonzalez, G. M., Hansanti, P., . . . Wang, J. (2022). No language left behind: Scaling Human-Centered Machine Translation. arXiv (Cornell University). <https://doi.org/10.48550/arxiv.2207.04672>

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M., Lacroix, T., Roziere, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., & Lample, G. (2023). LLAMA: Open and Efficient Foundation Language Models. arXiv (Cornell University). <https://doi.org/10.48550/arxiv.2302.13971>

Parida, S., Panda, S., Biswal, S. P., Kotwal, K., Sen, A., Dash, S. R., & Motlicek, P. (2021). Multimodal Neural Machine Translation System for English to Bengali. Proceedings of the

FirstWorkshop on Multimodal Machine Translation for Low Resource Languages (MMTLRL 2021). https://doi.org/10.26615/978-954-452-073-1_006

MandeepKharb. (n.d.). Youtube/GenerativeAI/TextToImageGenerator.ipynb at main · MandeepKharb/Youtube. GitHub.
<https://github.com/MandeepKharb/Youtube/blob/main/GenerativeAI/TextToImageGenerator.ipynb>