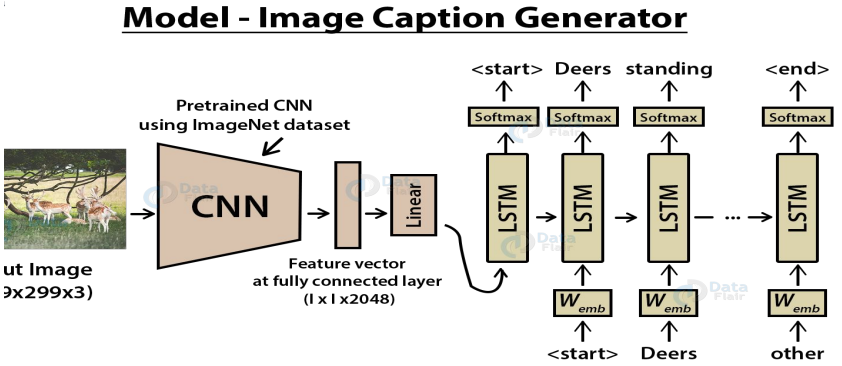
# Image Caption Generator

Chitra Sura
Cwid- 20012226

- This project aims to construct an **Image Caption Generator** by **integrating Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs).**
- Using TensorFlow and Keras.




- Step-1: CNN responsible for feature extraction, encoding the input image, specifically using Inception V3  pre trained on Images to extract image features.
- Step-2: features are passed as embeddings into an RNN built with Long Short-Term Memory (LSTM) layers.

# CAPTIONS

A brown horse and a black foal on the beach

A brown horse stands near a black horse that is sitting on the ground .

A large brown horse stands over a small black colt that is kneeling on the sand .

An adult horse approaching a foal on a sandy plain .

A thin brown horse standing and a small black horse sitting on sand

**Key Difference:**

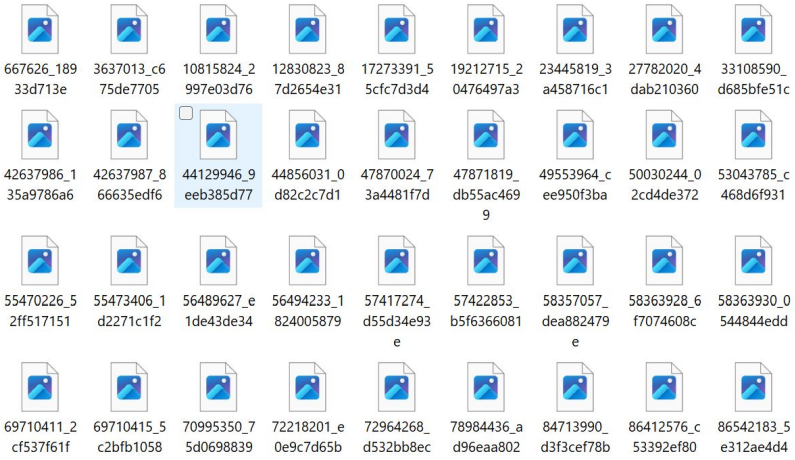
This approach differs from others by providing the image embedding as the initial input to the RNN network only once, allowing the LSTM to generate descriptive captions for the images. This combination of CNN and LSTM enables the model to learn and produce meaningful captions based on the extracted image features.

# Dataset

The dataset used is Flickr 8K, consisting of 8,091 images each one paired with five different captions to provide clear descriptions.

```
image,caption
1000268201_693b08cb0e.jpg,A child in a pink dress is climbing up a set of stairs in an entry way .
1000268201_693b08cb0e.jpg,A girl going into a wooden building .
1000268201_693b08cb0e.jpg,A little girl climbing into a wooden playhouse .
1000268201_693b08cb0e.jpg,A little girl climbing the stairs to her playhouse .
1000268201_693b08cb0e.jpg,A little girl in a pink dress going into a wooden cabin .
1001773457_577c3a7d70.jpg,A black dog and a spotted dog are fighting
1001773457_577c3a7d70.jpg,A black dog and a tri-colored dog playing with each other on the road .
1001773457_577c3a7d70.jpg,A black dog and a white dog with brown spots are staring at each other in the street .
1001773457_577c3a7d70.jpg,Two dogs of different breeds looking at each other on the road .
1001773457_577c3a7d70.jpg,Two dogs on pavement moving toward each other .
1002674143_1b742ab4b8.jpg,A little girl covered in paint sits in frogt of a painted rainbow with her hands in a bowl .
1002674143_1b742ab4b8.jpg,A little girl is sitting in front of a large painted rainbow .
1002674143_1b742ab4b8.jpg,A small girl in the grass plays with fingerpaints in front of a white canvas with a rainbow on it
1002674143_1b742ab4b8.jpg,There is a girl with pigtails sitting in front of a rainbow painting .
1002674143_1b742ab4b8.jpg,Young girl with pigtails painting outside in the grass .
1003163366_44323f5815.jpg,A man lays on a bench while his dog sits by him .
1003163366_44323f5815.jpg,A man lays on the bench to which a white dog is also tied .
1003163366_44323f5815.jpg,a man sleeping on a bench outside with a white and black dog sitting next to him .
1003163366_44323f5815.jpg,A shirtless man lies on a park bench with his dog .
1003163366_44323f5815.jpg,man laying on bench holding leash of dog sitting on ground
1007129816_e794419615.jpg,A man in an orange hat starring at something .
1007129816_e794419615.jpg,A man wears an orange hat and glasses .
1007129816_e794419615.jpg,A man with gauges and glasses is wearing a Blitz hat .
1007129816_e794419615.jpg,A man with glasses is wearing a beer can crocheted hat .
```

captions

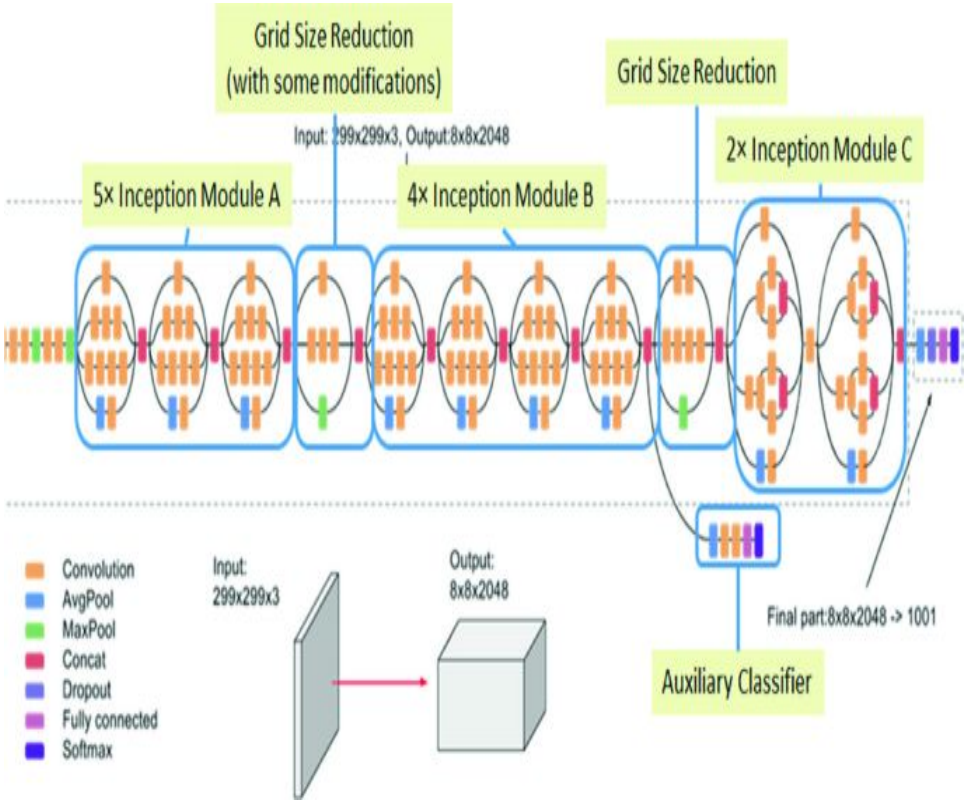



Images

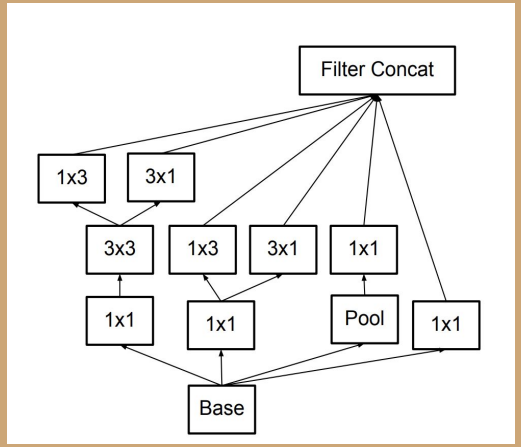
# ENCODER MODEL

**Inception V3-**
a convolutional neural network (CNN) architecture designed for image recognition tasks

- ➢ Input Layer:
- ➢ Stem:
- ➢ Inception Modules:
- ➢ Reduction Blocks:
- ➢ Auxiliary Classifiers:
- ➢ Fully Connected Layers:



**Inception Modules:**
The main characteristic of InceptionV3 is the repeated use of inception modules. **Each module comprises parallel convolutional operations of varying filter sizes (1x1, 3x3, 5x5), pooling, and concatenation**. The purpose is to capture features at different scales and complexities.




```
# Remove the last layer of the Inception V3 model
def get_encoder():
    image_model = tf.keras.applications.InceptionV3(include_top=False, weights='imagenet')
    new_input = image_model.input
    hidden_layer = image_model.layers[-1].output

    image_features_extract_model = tf.keras.Model(new_input, hidden_layer)
    return image_features_extract_model
```

**get_encoder() -** extracts the image feature using the InceptionV3 architecture in TensorFlow Keras.
**aim - r**emove the last classification layer of the InceptionV3 model

# Why Inception V3?

- Analysis of the models' performance showed that  ResNet-50 achieved 97.5%
- ResNet  has a simpler, single-scale processing unit with data pass-through connections.

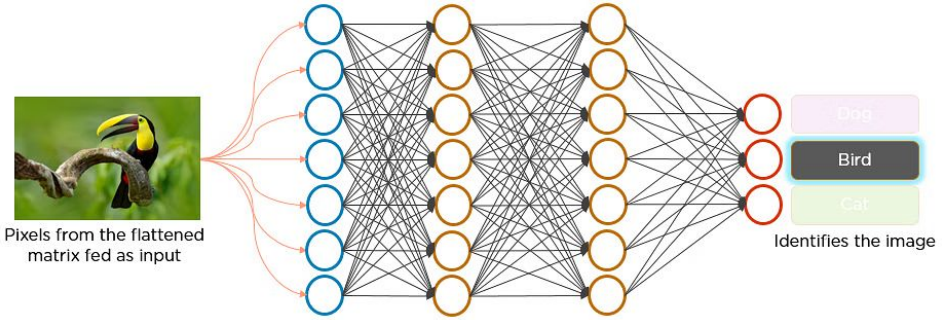
- Analysis of the models' performance showed that  Inception-V3 achieved 95.5%
- Inception  divides processing by scale, merges the results, and repeats.

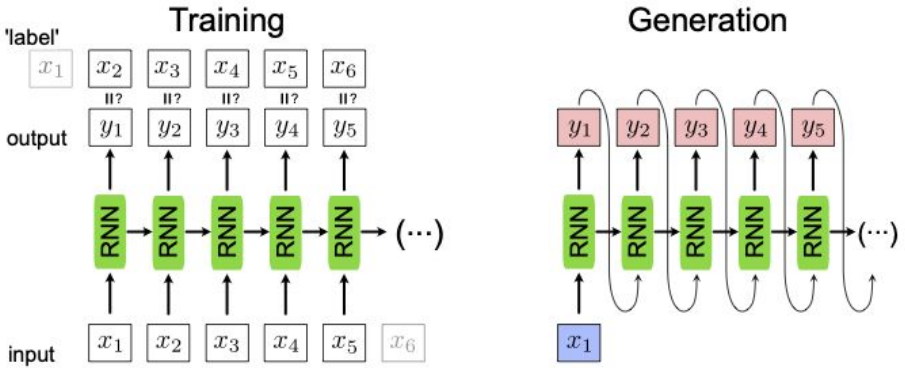
# Why not Resnet-50?

```
WARNING:matplotlib.image:Clipping
input data to the valid range for
imshow with RGB data ([0..1] for
floats or [0..255] for integers).
```

# Encoder- CNN

Image detection




# Decoder - RNN

Caption Generation

# CNN

- Feature Extraction:
- Pre-trained Models:
- Transfer Learning:
- Feature Maps:
- Contextual Information:
- Dimensionality Reduction:

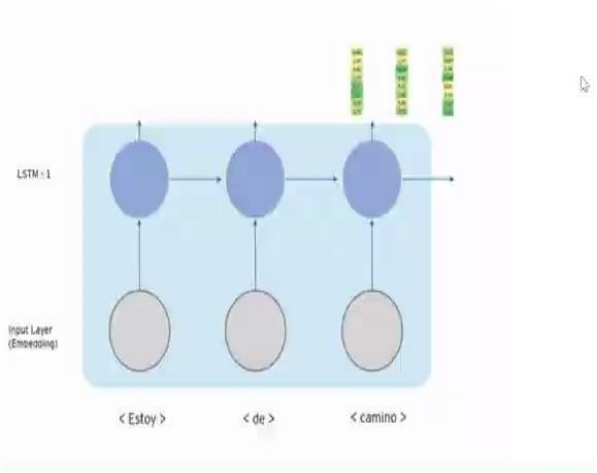
# RNN

- Sequence Generation:
- Long Short-Term Memory (LSTM) or Gated Recurrent Unit (GRU):
- Word Embeddings:
- Initial State:
- Word Generation:
- Context Vector and Attention:
- Training Objective:
- Capturing Temporal Dependencies:
- Beam Search or Sampling:

# Important Concept

**Decoder Architecture:**

Attention Mechanism: Implementing an attention mechanism in the decoder to focus on different parts of the image when generating different words. Attention can improve the model's ability to associate image features with specific words in the cap



# Estimated caption

```
def synthesize_captions(captions):
    # Tokenize captions into words
    tokenized_captions = [caption.split() for caption in captions]

    # Create a dictionary to hold word frequencies
    word_freq = {}
    for caption in tokenized_captions:
        for word in caption:
            if word not in word_freq:
                word_freq[word] = 1
            else:
                word_freq[word] += 1

    # Select the most common words across captions
    common_words = [word for word, freq in word_freq.items() if freq >= len(captions) // 2]

    # Generate a single synthesized statement
    synthesized_statement = []
    for word in common_words:
        synthesized_statement.append(word)

    return ' '.join(synthesized_statement)
```

```
['<start> A girl smiles as she runs across
the white sandy beach in her swimsuit .
<end>',
 '<start> a little girl in a flowered
bathingsuit runs through the sand at the
beach <end>',
 '<start> A little girl in a flower
swimsuit running across the beach with
waves in the background . <end>',
 '<start> There is a little girl running on
the beach . <end>',
 '<start> The young girl is running on a
sandy beach . <end>']
```

<start> A girl runs across the sandy beach in swimsuit .a little running is on

```
# Generate and print the synthesized statement
synthesized = synthesize_captions(captions)
print("Synthesized Statement:")
print(synthesized)
```

# References

Dataset - https://www.kaggle.com/datasets/adityajn105/flickr8k

https://thinkautonomous.medium.com/rnns-in-computer-vision-image-captioning-597d5e1321d1

https://github.com/angeligareta/image-captioning/blob/master/notebooks/image-captioning.ipynb

# THANK YOU