# Final Presentation

## Image Captioning using CNN ,LSTM and Attention mechanism

Presented By:

Sai Chandra Sri Ramula
B00116842
Oklahoma City University

# Image Captioning: generating clear and descriptive textual summaries of images

## Dataset

- Flickr8k dataset
  - 8,000 images with 5 captions per image.
  - Captions reflect diverse real-world scenarios

## CNN-LSTM Model

CNN (Encoder):
Extracts image features using pre-trained CNNs (DenseNet201, Xception, InceptionV3), producing fixed-length feature vectors.

LSTM (Decoder):
Generates captions sequentially from CNN features, predicting each word based on learned language patterns.

## Attention Based Model

The encoder (CNN) extracts spatial feature maps from the image. The decoder generates captions by applying attention, dynamically focusing on relevant image regions at each timestep to improve word alignment and caption accuracy.

## Evaluation

BLEU (Bilingual Evaluation Understudy) Score:
A metric that evaluates the quality of generated captions by comparing them to human-written references, based on overlapping n-grams. Higher BLEU scores indicate captions that are more fluent and semantically aligned with the reference descriptions.

Oklahoma City University | Sai Chandra Sri Ramula | Computer Science - Data Science

# 🎯 CNN + LSTM Model — Methodology

🧹 Preprocessing

- Images resized to 224x224 or 299x299 depending on the CNN used and normalized.
- Captions cleaned (lowercased, punctuation removed) and wrapped with <start> and <end> tokens.

🔤 Tokenization

- Captions converted into integer sequences using Keras Tokenizer.
- Vocabulary built from the dataset.
- Sequences padded to a fixed length for uniform input shape.

🧠 Model Architecture

- Encoder: Pre-trained CNNs (DenseNet201, Xception, InceptionV3) extract visual features from images.
- Decoder (LSTM):
  - Receives the encoded image features and the embedded input caption.
  - Generates captions one word at a time.
  - At each timestep, LSTM outputs a probability distribution over the vocabulary.
  - The word with the highest probability is selected as the next word (argmax).

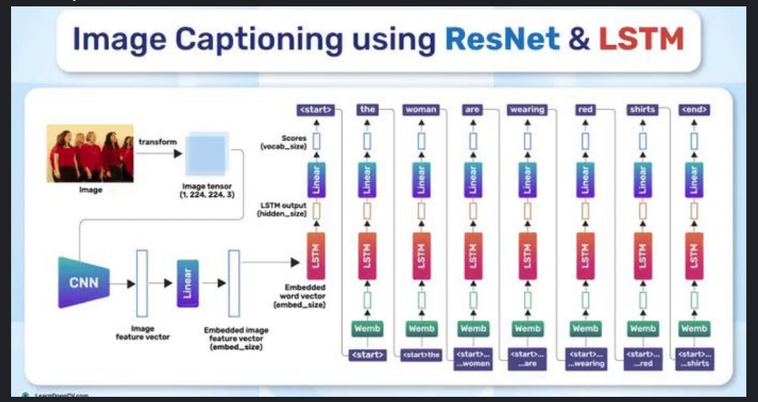
# 🎯 CNN + LSTM Model — Methodology

🔁 Training

- A custom DataGenerator constructs input-output pairs: (image features + partial caption) → next word.
- Model learns to predict the next word in the sequence based on context.
- Training setup:
  - Loss Function: Categorical Crossentropy (on predicted word probabilities).
  - Optimizer: Adam
  - Callbacks:
    - EarlyStopping – halts training if no improvement
    - ReduceLROnPlateau – lowers learning rate on stagnation
    - ModelCheckpoint – saves the best-performing model

📝 Prediction

- Captioning starts with <start>, and the model predicts one word at a time.
- The predicted word is fed into the next timestep.
- Stops when <end> token is generated or maximum length is reached.



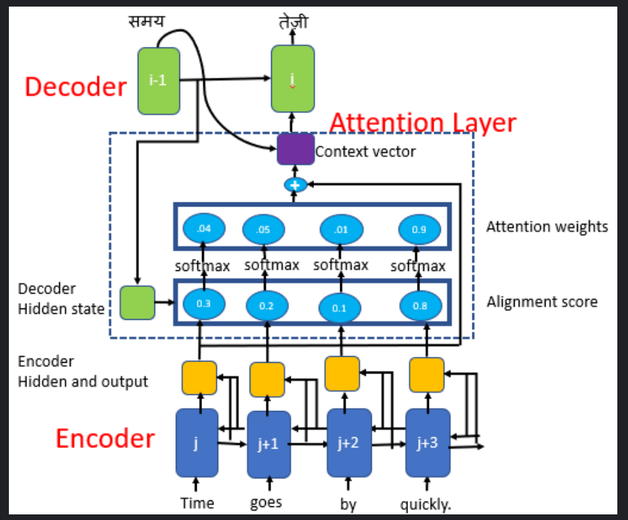
# 🎯 Attention-Based Model — Methodology

🧹 Preprocessing

- Images resized to 299×299 and preprocessed for Xception.
- Captions cleaned, tokenized, and padded to a maximum length of 34 tokens.
- Vocabulary limited to top 5000 words.

🧠 Model Architecture

- Encoder:
  - Xception (excluding top layer) extracts 10×10×2048 spatial features.
  - Features are projected to a lower-dimensional space using a Dense layer.
- Decoder with Bahdanau Attention:
  - Embeds input words and processes sequences using LSTM.
  - Attention scores are computed between each spatial image feature and the decoder's current hidden state.
  - These scores determine how much focus (weight) each image region gets — forming a context vector.
  - The context vector is concatenated with the word embedding and passed to the LSTM.
  - The output is a probability distribution over the vocabulary to predict the next word.



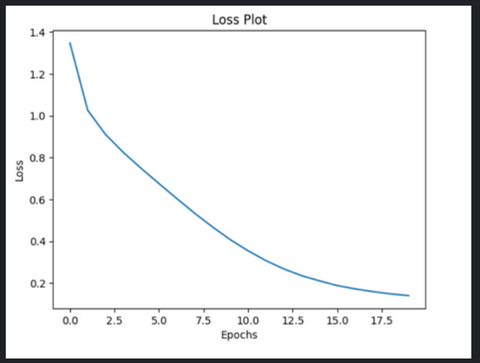
# 🎯 Attention-Based Model — Methodology

🔁 Training

- Custom training loop with teacher forcing (ground truth word passed at each step).
- Loss: Sparse Categorical Crossentropy (ignores <pad> tokens).
- Optimizer: Adam
- Trained for 20 epochs with batch size 64.

📝 Prediction & Visualization

- Generation starts from <start> and continues until <end> or 34 words.
- Attention heatmaps visualize where the model focused for each generated word.



# Thank you

Oklahoma City University | Sai Chandra Sri Ramula | Computer Science - Data Science