1} Answer:

Classifying animals and birds in video clips using Machine Learning involves several steps, from data preprocessing to model selection and evaluation. Here's how the process might unfold:

Data Collection and Preprocessing:

1. Video Collection: Camera use karke different photos caprture karne ka.

2. Frame Extraction: As video is just images playes one after the other rapidly , ek ek frame ko extract karege. And feed the photos to the machine learning model.

3. Annotation: Annotate the frames with labels indicating the species of animals or birds present. This annotation can be done manually or through crowdsourcing. This labeled data will be used to train and validate the machine learning model.

4. Data Augmentation: filters lagana like lighting thik karna , or rotate photos.

5. Data Split: Divide the annotated data into training, validation, and testing sets. A common split might be 70% for training, 15% for validation, and 15% for testing.

Feature Extraction:

6. Feature Extraction: Convert the images (frames) into numerical representations that a machine learning model can understand. Commonly used techniques include Convolutional Neural Networks (CNNs) for images. CNNs automatically learn hierarchical features from the images, which can be used for classification. Since it reads only pixels.

Model Selection and Training:

7. Model Choice: There are lots of model that uses images but we should use a model that is accurate and is cost efficient and gives us a better result.

8. Model Architecture: Design the architecture of the CNN. This usually consists of multiple convolutional layers, pooling layers, and fully connected layers. The final layer will have as many nodes as there are species to classify.

9. Training: Train the CNN using the labelled training data. During training, the model will adjust its internal parameters to minimize the difference between its predictions and the actual labels.

Model Evaluation and Deployment:

10. Validation: Evaluate the trained model's performance on the validation set. Use metrics like accuracy, precision, recall, and F1-score to assess how well the model is classifying species.

11. Hyperparameter Tuning: Adjust hyperparameters like learning rate, batch size, and number of layers to optimize the model's performance.

12. Testing: Once satisfied with the model's performance on the validation set, test it on the held-out testing set to get an unbiased estimate of its real-world performance.

13. Deployment: Deploy the trained model to the camera spots within the wildlife sanctuary. The model will process the video frames in real-time and predict the species of animals and birds present.

2} Answer:

Data Collection and Preprocessing:

1. Parallel Texts: These apps require large datasets of parallel texts, which are sentences in the source language paired with their translations in the target language. These datasets are used for training and fine-tuning the translation models.

2. Cleaning and Tokenization: The texts are cleaned to remove special characters, punctuation, and formatting issues. Then, the texts are tokenized into words or subword units. Tokenization breaks down sentences into smaller units, which makes handling various languages more manageable. Stopwords is a function of WordCloud libarary that does this work.

3. Subword Segmentation: Instead of working with individual words, these apps often use subword units, like subword pieces or characters. This helps in handling languages with complex morphology or agglutinative structures, where words are formed by adding prefixes, suffixes, etc.

Neural Machine Translation (NMT) Model:

4. Encoder-Decoder Architecture: Most modern translation models use a neural network architecture with an encoder-decoder setup. The encoder encodes the source sentence into a fixed-dimensional vector (context), and the decoder generates the translation from this context.

5. Attention Mechanism: The attention mechanism allows the decoder to focus on different parts of the source sentence as it generates each word of the translation. This enables the model to capture the context and maintain the sentence's meaning during translation.

Translating Words between Languages:

6. Word Embeddings: Words are converted into high-dimensional vectors called word embeddings. These embeddings capture semantic relationships between words, enabling the model to understand similarities and differences between words in different languages.

7. Cross-Lingual Embeddings: Some models use shared embeddings across languages, allowing them to capture relationships between words in different languages. This helps maintain the structure and meaning of sentences during translation.

8. Translation: During translation, the source sentence is fed into the encoder, and the encoder's output (context) is then passed to the decoder. The decoder generates the translation word by word, considering the context and the attention mechanism's guidance.

3} Answer:

Data Preprocessing:

1. Text Cleaning: Remove any non-ASCII characters, special characters, and punctuation from the text. This includes removing emojis, URLs, and other non-text elements that might not contribute to the classification.

2. Lowercasing: Convert all text to lowercase to ensure uniformity and reduce the complexity of the vocabulary.

3. Tokenization: Split the text into individual words or tokens. This is important for later feature extraction.

4. Stopword Removal: Remove common stopwords (e.g., "the", "and", "is") that don't carry significant meaning.

5. Stemming or Lemmatization: Reduce words to their base forms. For example, "running" and "runs" could be stemmed to "run".

Feature Extraction:

6. TF-IDF Vectorization: Convert the preprocessed text data into TF-IDF (Term Frequency-Inverse Document Frequency) vectors. TF-IDF considers both the term frequency and the importance of the term in the entire corpus of documents. This creates a numerical representation of the text, which can be used for clustering.

Clustering:

7. K-Means Clustering: Apply the K-Means clustering algorithm to group similar texts together. K-Means divides the data into 'k' clusters based on the similarity of the TF-IDF vectors.

8. Identifying Spam Cluster: Examine the resulting clusters and look for clusters that contain a significant number of messages. These clusters could potentially represent spam messages if they exhibit certain characteristics like frequent use of promotional words, repetitive content, etc.

Thought Process and Shaky Scenarios:

1.Cluster Interpretation: A challenge is interpreting the clusters correctly. Some clusters may contain legitimate content that's similar in nature. It's important to manually inspect a sample of texts from each cluster to ensure the classification is accurate.

2. Sparse Data: If the data is sparse, meaning there are very few common words across messages, clustering might be less effective. In such cases, you might need to consider more sophisticated techniques that handle sparse data well.

3. Subjectivity: Identifying spam can sometimes be subjective. What one person considers spam might be legitimate content for another. The model may not capture this nuance.

4. New Types of Spam: If the data includes new types of spam that the model hasn't encountered during training, it might struggle to identify them accurately.