Methodology and Key Requirements

Follows a positivist paradigm with tools like Python, TensorFlow, Keras, and OpenCV. Dataset: Roboflow Food Detection. Figures include block diagrams and CNN/LSTM architectures.

Methodology and Data Source

Food Item Identification Research - Key Methodology Points

Philosophical Approach

- **Positivism paradigm** - objective, empirical measurement and analysis
- Focus on observable phenomena and quantitative methods
- Evidence-based model development and validation

Tools & Technologies

- **Python** - main programming language
- **TensorFlow & Keras** - deep learning model development
- **OpenCV** - image preprocessing (resizing, normalization, augmentation)
- **Statistical Tools** - Scikit-learn, Pandas for data analysis

Dataset

- **Source**: Roboflow Food Detection Dataset

  Dataset link: https://universe.roboflow.com/sulthan-hasanal-hakim/food-detection-oytkj/dataset/1
- **Sample**: Thousands of annotated food images
- **Split**: 70% training, 15% validation, 15% testing

Model Approaches

CNN (Convolutional Neural Networks)

- Primary model for image recognition
- Feature extraction through convolutional layers
- Effective for spatial pattern recognition

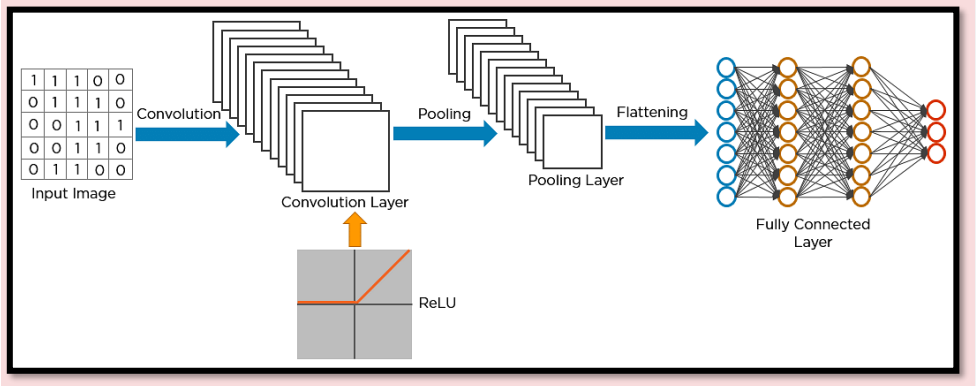


Figure 1 CNN (Biswal, 2023)

RNN (Recurrent Neural Networks)

- For sequential data processing
- Memory state maintenance

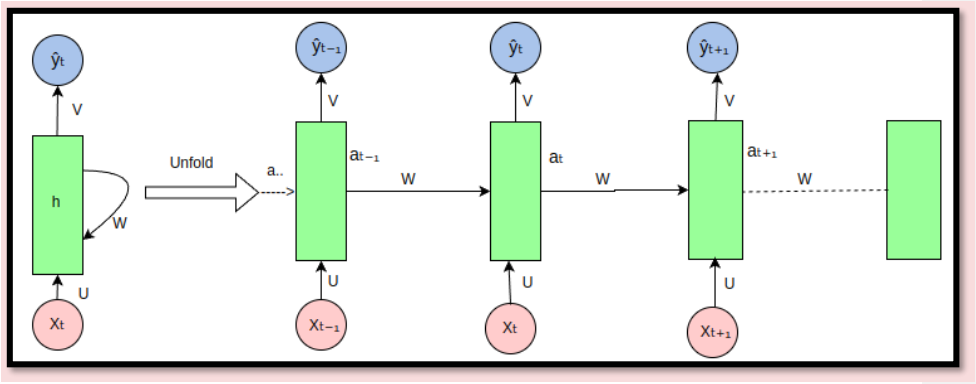


Figure 2 RNN (Poudel, 2023)

LSTM (Long Short-Term Memory)

- Advanced RNN variant
- Better long-term dependency learning

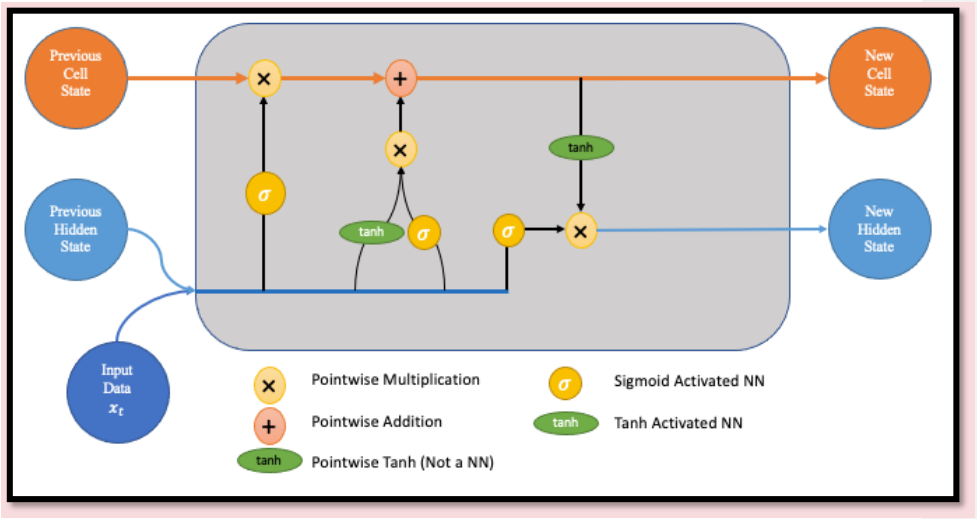


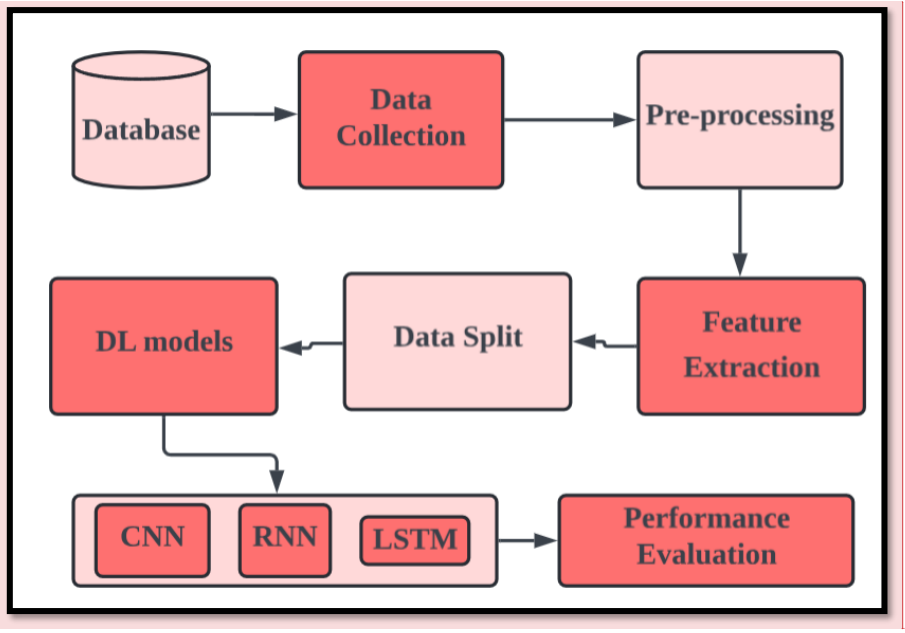
Figure 3 LSTM (Dolphin, 2022)

Analysis Process

1. **Data Preprocessing** - image resizing, normalization, augmentation
2. **Feature Extraction** - using CNN convolutional layers
3. **Model Development** - building and training neural networks
4. **Model Evaluation** - accuracy, precision, recall, F1-score, confusion matrix

Key Success Factors

- Data diversity for model generalization
- Proper annotation verification
- Balanced dataset splitting
- Comprehensive performance evaluation



Block diagrams and CNN/LSTM architectures

Commented [YB1]: The methodology follows a systematic approach using deep learning techniques (primarily CNNs) to identify food items in images. The research uses a large annotated dataset from Roboflow and employs standard machine learning practices for data preprocessing, model training, and evaluation.

Functional and Non-Functional Requirements

Functional: data preprocessing, model training, evaluation. Non-functional: performance, usability, maintainability, portability, and security.

Key Requirements

Functional Requirements

Data Processing

- **Input**: Images from directories, resize to 224x224 pixels
- **Augmentation**: Rotation, flipping, zoom, brightness adjustment for training data
- **Labels**: Load CSV annotations using Pandas

Model Architecture

- **CNN**: VGG16 with pre-trained ImageNet weights, Adam optimizer
- **RNN**: Sequential model with TimeDistributed layers
- **LSTM**: Sequential model with LSTM layers for sequence processing

Training & Evaluation

- **Training**: Early stopping to prevent overfitting
- **Validation**: Performance monitoring on validation dataset
- **Metrics**: Accuracy, classification reports, confusion matrices
- **Visualization**: Sample images and training history plots

Non-Functional Requirements

Performance

- **Speed**: 2 seconds per image processing, 10 images/second in batch mode
- **Scalability**: Support for GPU scaling and cloud deployment

Usability

- **Interface**: Easy-to-use UI for image upload and results viewing
- **Documentation**: User manuals and API documentation
- **Accessibility**: WCAG compliance for disabled users

Reliability

- **Availability**: 99.9% uptime target
- **Error Handling**: Robust fault tolerance and data integrity
- **Logging**: Comprehensive error tracking and user notifications

Security

- **Data Privacy**: GDPR/CCPA compliance
- **Authentication**: Role-based access control
- **Encryption**: Data protection in transit and at rest

Technical

- **Portability**: Cross-platform support (Windows, macOS, Linux)
- **Integration**: RESTful APIs, JSON/CSV data formats
- **Deployment**: Docker containerization, CI/CD pipelines