# Research Paper Comparison: Two Approaches to Drug-Resistant Epilepsy Prediction

Connectivity-Based Machine Learning vs Convolutional Neural Networks

## EXECUTIVE SUMMARY

This document compares two cutting-edge approaches for predicting drug-resistant epilepsy (DRE) in newly diagnosed patients. Both papers achieve excellent results but use fundamentally different methodologies:

Paper 1 (Connectivity Approach): 91.5% accuracy using brain network connectivity features
Paper 2 (CNN Approach): 99% accuracy using automatic feature extraction from raw EEG signals

## PAPER OVERVIEW COMPARISON

**Paper 1 (Machine Learning/Connectivity) vs Paper 2 (CNN)**

| Aspect | Paper 1 (ML/Connectivity) | Paper 2 (CNN) |
|---|---|---|
| **Approach** | Manual feature extraction + Traditional ML | Automatic feature extraction + Deep Learning |
| **Data Size** | 139 patients | 101 patients |
| **Features** | 216 network connectivity features | Raw EEG signals (automatic extraction) |
| **Key Method** | Phase Lag Index + Tree Bagger | 1D CNN + Clinical features |
| **Best Accuracy** | 91.5% | 99% |
| **Sensitivity** | 97% | 96% |
| **Specificity** | 81% | 72% |
| **Focus** | Brain network connectivity | Raw signal patterns |

**Fondamental Differences :**

**1. Feature Extraction Philosophy:**

- Paper 1:"Let's calculate specific brain connectivity measures that we know are important"

- Paper 2:"Let the AI figure out what patterns are important from raw data"

**2. Data Representation:**

- Paper 1: Converts EEG → Connectivity matrices → Graph theory metrics

- Paper 2: Uses raw EEG signals → CNN learns patterns directly

**3. Interpretability:**

- Paper 1: We know WHY it works (frontotemporal theta connectivity)

- Paper 2:We know THAT it works, but CNN features are "black box"

## CNN Architecture Explanation (Paper 2):

### 7-Layer CNN Structure:

```
Input EEG Signal (90 seconds)
         ↓
[Convolutional Layer 1] - Kernel: 5×10, ReLU activation
         ↓
[Convolutional Layer 2] - Kernel: 5×10, ReLU activation
         ↓
[Flatten Layer] - Converts 2D to 1D
         ↓
[Fully Connected Layer 1] - Dense connections
         ↓
[Fully Connected Layer 2] - Dense connections
         ↓
[Fully Connected Layer 3] - Dense connections
         ↓
[Fully Connected Layer 4] - Dense connections, Softmax output
         ↓
Output: DRE Probability
```

Detailed Layer Breakdown and Visuals

**Input Layer :**

**Input Layer**

Raw EEG data (90 seconds, 19 channels)

Input

19 × 23040

Params: 0

**Technical Details:**

**Activation:** None

**Parameters:** 0

**Dimensions:** 19 × 23040

**Function:**

Receives preprocessed EEG signals from 19 electrodes, sampled at 256 Hz for 90 seconds

**Layer 1-2: Convolutional Layers**

- Purpose: Extract local patterns from EEG signals

- Kernel Size: 5×10 (5 time points × 10 channels)

- Function: Detects specific EEG patterns like spikes, waves, frequency changes

- Activation:ReLU (Rectified Linear Unit) - f(x) = max(0,x)

**Conv Layer 1**

First convolutional layer with ReLU

Convolutional

Kernel: 5×10, Filters: 32
Params: 1,632

**Technical Details:**

**Activation:** ReLU
**Parameters:** 1,632
**Dimensions:** Kernel: 5×10, Filters: 32

**Function:**

Extracts local temporal-spatial features from EEG signals using 32 filters

**Conv Layer 2**

Second convolutional layer with ReLU

Convolutional

Kernel: 5×10, Filters: 64
Params: 10,304

**Technical Details:**

**Activation:** ReLU
**Parameters:** 10,304
**Dimensions:** Kernel: 5×10, Filters: 64

**Function:**

Further feature extraction with increased filter depth for complex pattern recognition

**Layer 3: Flatten Layer**

- Purpose: Convert 2D feature maps to 1D vector

- Function: Prepares data for fully connected layers

- Example: [64×128] matrix → [8192] vector

### Flatten Layer

Converts 2D feature maps to 1D vector

Flatten

1D Vector

Params: 0

**Technical Details:**

**Activation:** None

**Parameters:** 0

**Dimensions:** 1D Vector

**Function:**

Reshapes multi-dimensional feature maps into a single vector for dense layers

**Layer 4-7: Fully Connected Layers**

- Purpose: Learn complex relationships between extracted features

- Function: Combine low-level patterns into high-level decisions

- Output: Softmax function gives probability of DRE

### Dense Layer 1

First fully connected layer

Fully Connected

128 neurons

Params: Variable

**Technical Details:**

**Activation:** ReLU

**Parameters:** Variable

**Dimensions:** 128 neurons

**Function:**

Dense connections learn complex relationships between extracted features

Dense Layer 2

Second fully connected layer

Fully Connected

64 neurons

Params: 8,256

**Technical Details:**

**Activation:** ReLU

**Parameters:** 8,256

**Dimensions:** 64 neurons

**Function:**

Further abstraction and feature combination for final classification

Dense Layer 3

Third fully connected layer

Fully Connected

32 neurons

Params: 2,080

**Technical Details:**

**Activation:** ReLU

**Parameters:** 2,080

**Dimensions:** 32 neurons

**Function:**

Final feature processing before classification output

**Final Layer :**

Output Layer

Classification output with Softmax

Output

2 classes (DRE/Non-DRE)

Params: 66

**Technical Details:**

**Activation:** Softmax

**Parameters:** 66

**Dimensions:** 2 classes (DRE/Non-DRE)

**Function:**

Produces probability distribution over DRE and Non-DRE classes

## Implementation recommendations

**Phase 1: Start with Connectivity Approach (Paper 1)**

- Implement Phase Lag Index calculation

- Focus on frontotemporal electrode regions

- Use theta band (4-8 Hz) analysis

- Apply Tree Bagger ensemble learning

- Target: 91.5% accuracy with interpretable results

**Phase 2: Add CNN Capabilities (Paper 2)**

- Implement 7-layer CNN architecture

- Extract automatic features from raw EEG

- Combine with clinical variables

- Target: 99% accuracy with automated processing

**Phase 3: Hybrid Approach**

- Combine connectivity features with CNN features

- Use ensemble methods for final prediction

- Maintain interpretability while maximizing accuracy

- Target: >95% accuracy with clinical understanding

A python example of the hybrid approach :

```python
# Combine both approaches
def hybrid_dre_prediction(eeg_data, clinical_data):
    # Extract connectivity features (Paper 1)
    connectivity_features = extract_phase_lag_index(eeg_data)

    # Extract CNN features (Paper 2)
    cnn_features = cnn_model.extract_features(eeg_data)

    # Combine all features
    combined_features = np.concatenate([
        connectivity_features,
        cnn_features,
        clinical_data
    ])

    # Use ensemble prediction
    prediction = ensemble_model.predict(combined_features)
    return prediction
```