AUX-DROP:

HANDLING HAPHAZARD INPUTS IN ONLINE LEARNING USING AUXILIARY DROPOUTS

Presented by: Zoheen Shahzad, Btech 3rd year ece (JMI)

PROBLEM STATEMENT

LEARNING FROM HAPHAZARD INPUTS

Many existing methods assume that streaming data has a fixed, time-invariant size, and train models accordingly.

But this isn't always true. The dimension of inputs can vary over time. The inputs can have missing data, missing features, obsolete features, sudden features and an unknown number of the total features.

HAPHAZARD INPUTS-DEFINITION

Haphazard inputs refer to streaming data with a dynamic feature space, where the number, type, and availability of features can vary over time without prior information.

CHARACTERISTICS

1.Streaming data 2.Missing data
3.Missing features 4.Obsolete features
5.Sudden features 6.Unknown number of features

EXISTING MODELS

Characteristics	Online Deep Learning Methods like ODL	ODL + Online Data Imputation	ODL + Extrapolation	ODL + Prior Information	ODL + Gaussian Noise
Streaming data (C1)	✓	✓	✓	✓	✓
Missing data (C2)	×	✓	×	✓	✓
Missing features (C3)	×	×	×	×	✓
Obsolete features (C4)	×	×	✓	×	✓
Sudden features (C5)	×	×	×	×	×
Unknown no. of features (C6)	×	×	×	×	×

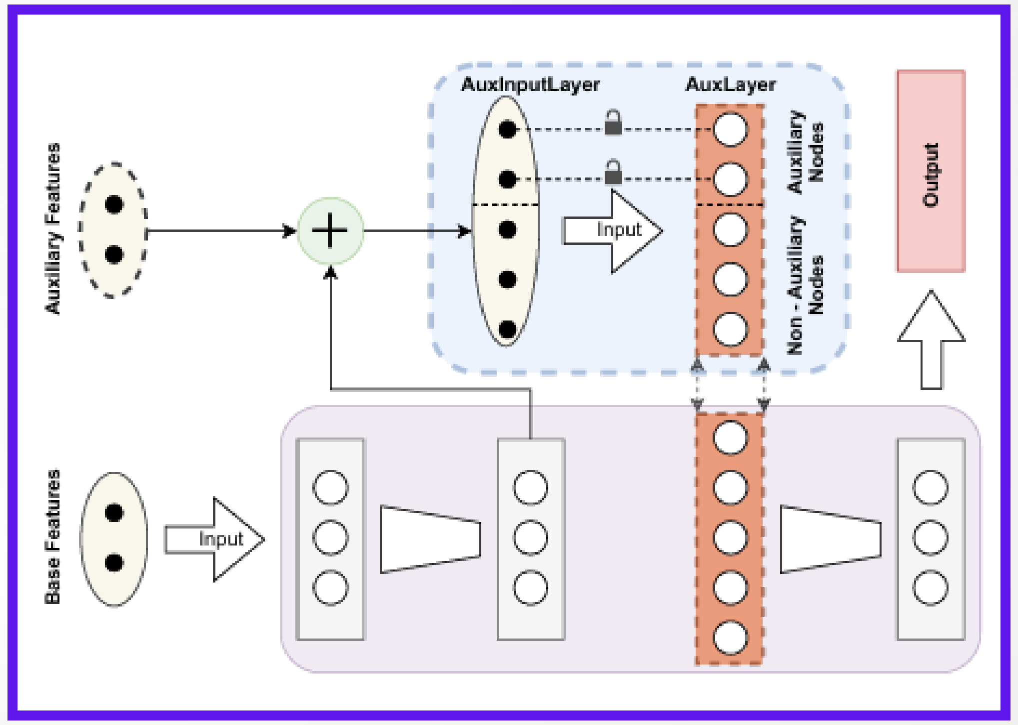
AUX-DROP

Characteristics	Aux-Drop
Streaming data (C1)	✓
Missing data (C2)	✓
Missing features (C3)	✓
Obsolete features (C4)	✓
Sudden features (C5)	✓
Unknown no. of features (C6)	✓

The core of Aux-Drop lies in utilizing the concept of dropout to accommodate the ever-changing characteristics of haphazard inputs. Dropout drops the nodes randomly from a hidden layer whereas we employ selective dropout along with the random dropout.

FEATURES

1. Handles Haphazard Inputs Efficiently
2. Selective + Random Dropout
3. Lightweight and Scalable
4. Minimal Architectural Changes
5. Dynamic Handling of Features

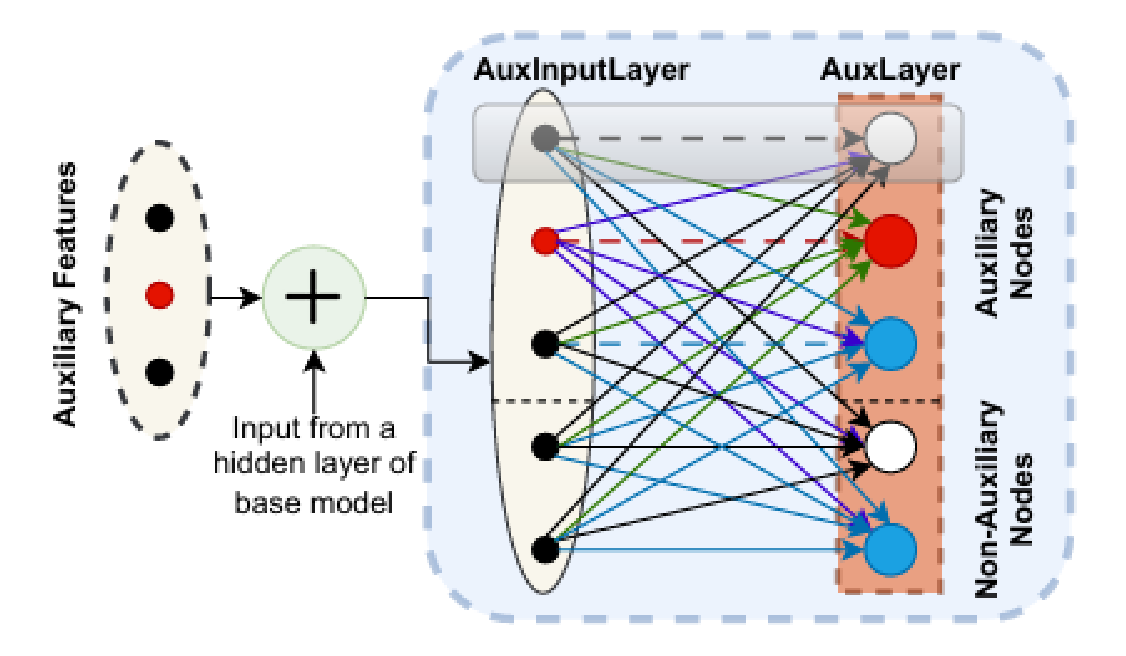


HOW IT WORKS

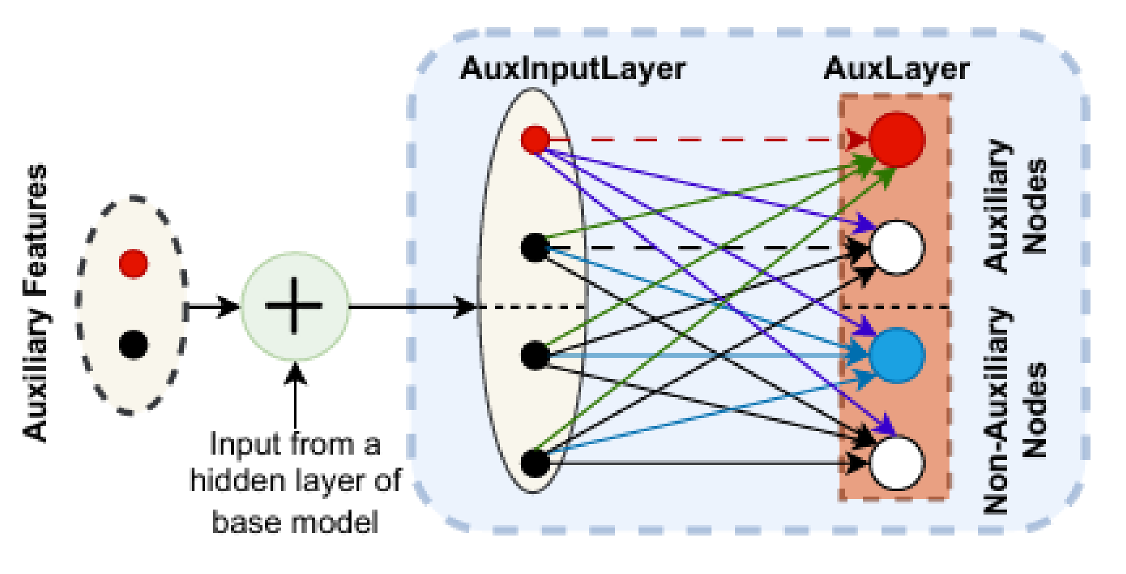
At a high level, Aux-Drop picks one hidden layer (the "AuxLayer") in the online model and, at each time step *t*:

1. Split features into always-present base features X^B_T and haphazard auxiliary features X^A_T
2. Compute the network up to layer *z - 1,(taking zth layer as auxlayer)* yielding hidden activations $H_{Z-1,T}$
3. Form the AuxInputLayer by concatenating $H_{Z-1,T}$ with whatever auxiliary features have arrived.
4. Drop nodes in the AuxLayer via two masks:
   - Selective dropout: drop every node whose corresponding auxiliary feature is missing.
   - Random dropout: from the remaining nodes, drop a further $|M_{Z-1,T}|.d-$ |selective| nodes at random.
5. Forward through the pruned AuxLayer and the rest of the network to predict y(hat), observe the output y, compute loss, and then update only the unfrozen weights via online gradient descent.

By always stripping out exactly the missing-feature nodes (plus a bit of extra random dropout), Aux-Drop prevents co-adaptation on any subset of auxiliary inputs yet still learns from whatever subset arrives.



(e) Instance $t+3$: Sudden feature arrives



(d) Instance $t+2$: An auxiliary feature becomes obsolete

IMPACT AND LIMITATION

Impact of Aux-Drop

1.Online Learning
Makes deep learning more robust to streaming, unpredictable data

2.Efficiency
Lightweight compared to models like Aux-Net (fewer parameters)

3.Performance
Outperforms prior methods (Aux-Net, OLVF, OLSF) on varied datasets

4.Plug-and-Play
Easy to integrate with existing online DL models like ODL/OGD

5.Feature Robustness
Reduces over-reliance on unstable or missing features

Limitations of Aux-Drop

1. Needs Base Feature
Assumes at least one always-available base feature to anchor the model

2. AuxLayer Size Tuning
Choosing dropout rate & AuxLayer size is still a hyperparameter problem

3.. AuxNode Creation Overhead
Adding new nodes dynamically (for sudden features) still incurs some cost

4.. Does Not Impute
It skips missing data—it doesn't try to estimate or fill in values

THANK YOU