Temporal Convolutional Networks and the Tennessee Eastman Process dataset

Foundations of Deep Learning 24/25

Università degli Studi di Milano-Bicocca

Master of Science in Data Science

Eduardo Mosca 925279

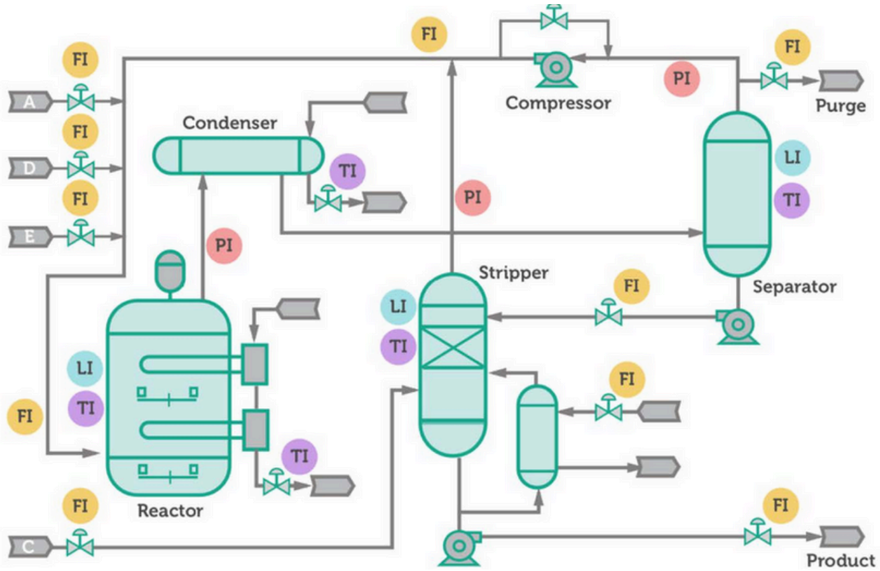
Introduction

- Developed project work around Tennessee Eastman Process(TEP) dataset
- Benchmark AD dataset, with papers summarizing DL methods on it
- Objective of reviewing literature and implementing industrial DL methods
- Want to effectively detect and classify anomalies in industrial TS setting
- Focus around Temporal Convolutional Networks(TCNs)
- Developed with Python(tensorflow), and colab for occasional model training
- Two main tasks: time series classification and anomaly detection
- TCN and TCN-AE architectures(inspired by Bai et al., Thill et al. respectively)

- The data was originally supplied by Rieth et al.
- Ive uploaded the parquet-converted files to this Kaggle page

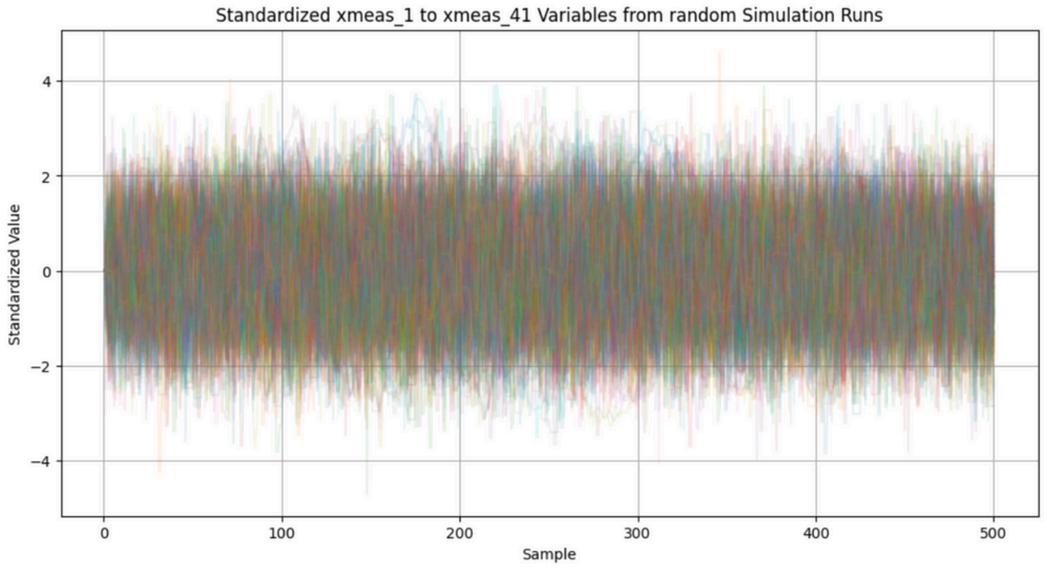
Dataset Introduction

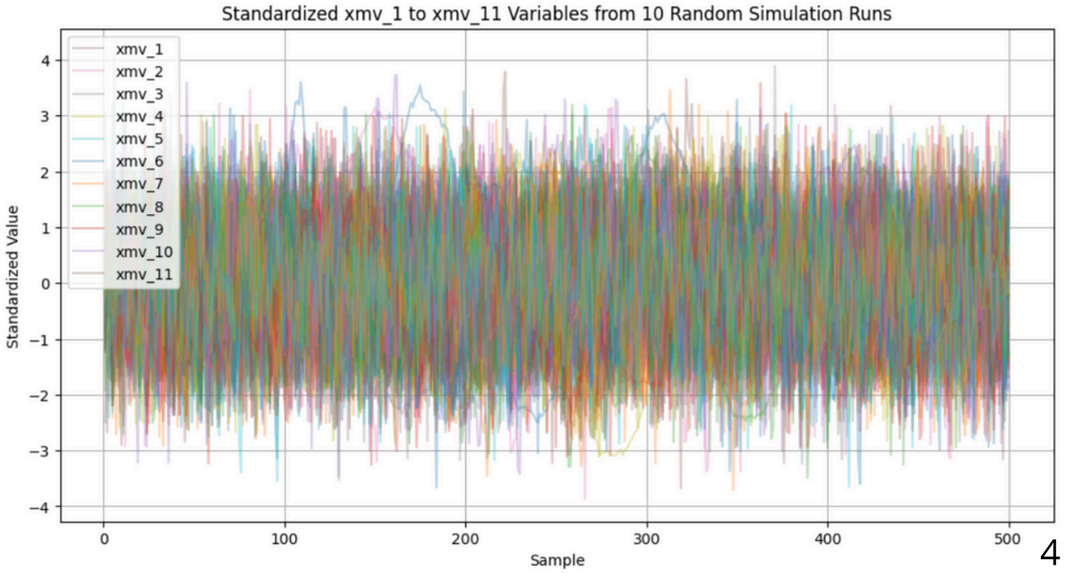
- Data about industrial chemical process simulations(normal and faulty)
- Train and Test datasets provided separately for both cases
- Anomalies characteristically introduced into the simulation some time into it
- 20 different types of anomalies, plus the normal operating conditions
- 500 normal simulations, and 500 for each anomaly type
- Not as much normal data as desirable, more anomalous data than normal



Normal Dataset Exploration

- Divided in train and test, differences among the two
- 500 simulations of normal conditions
- All continuous variables, 41 measured and 11 manipulated by operator
- In normal operating conditions, values typically oscillate within a confined range
- Not a lot of data, 250000 in total

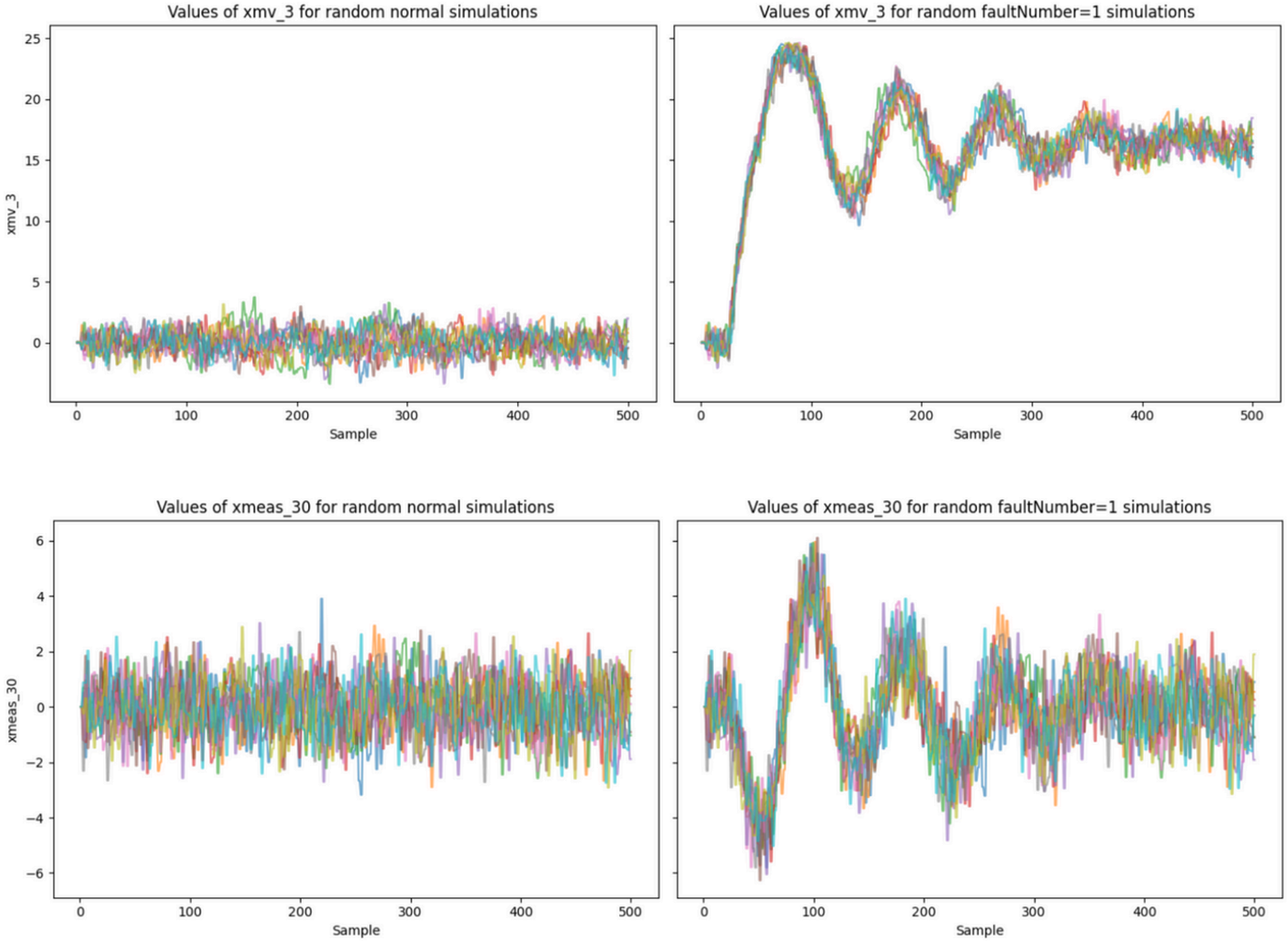


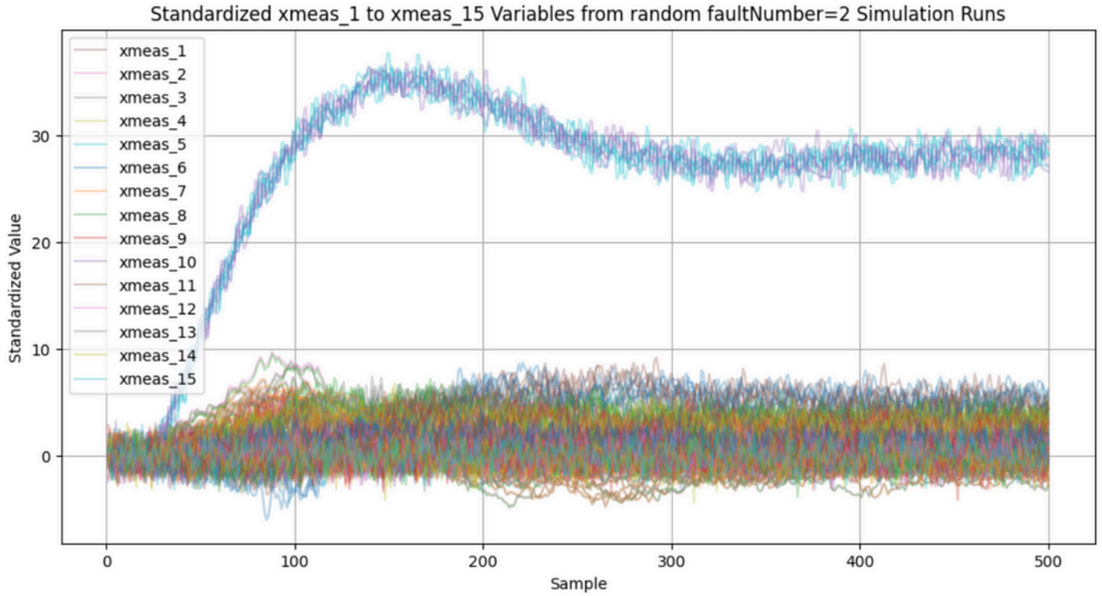


Anomalous Dataset Exploration

- Train and test sets provided separately
- 500 simulations for every anomaly type(class)
- Anomalies differ in nature, as the way theyre introduced is different
- Not all variables go crazy in every anomaly type
- Variables usually don’t go back to normal after anomaly onset
- The amount and set of variables that greatly react to anomalies is different between classes
- No time point or sequence anomaly labels supplied, only class number

Anomalous Dataset Exploration[cont]





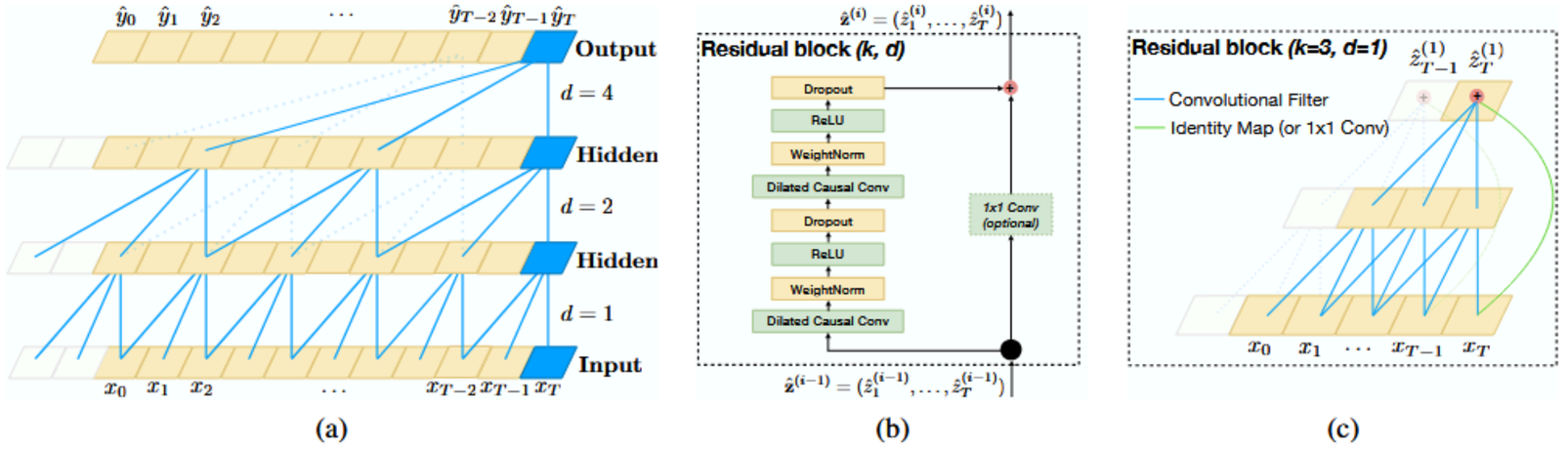
Approaching Development

- Two separate but connected model developments, for different tasks
- Extensive literature on what works on this dataset(mainly for AD)
- DL methods for industrial applications often subject to size constraints
- Lot of time spent finding and reading/trying to understand papers
- Restricted model configurations, want to know what is happening and methods
- One part developed after the other, so idea of what was working from trials

TCNs for Time Series Classification

- Goal was to build good model and understand this version of convolutions
- Approach was to implement TCN paper, experiment, possibly tweak for our case
- Base architecture, implemented variation, also existing keras-tcn library
- In terms of accuracy and CCE, satisfying outcome with small model

base architecture(TCN from Bai et al. 2018)



(a) (b) (c)

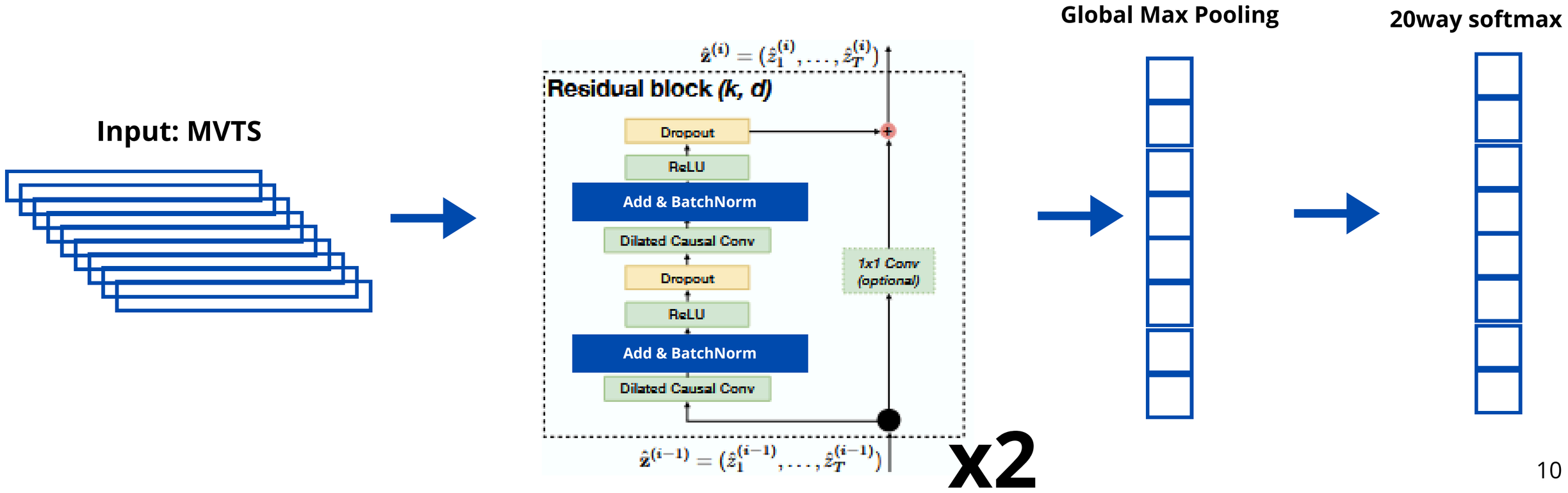
TCNs for TSC - Notes, Training, Testing, development setup

- Data was standardized(using anomaly-only data, which can be unrealistic)
- No further preprocessing(reminder: normal simulations not included)
- All implementations/architectures tended to perform similarly at the start
- Validation set to verify training, scaled using train set statistics
- Retrained selected config on entire train set before test set evaluation

- Surprisingly, Xavier init worked better than He, even if many relus
- Training stuck at plateau before introducing L2 decay in convolutions
- Architecture allowed for good performance even with smaller models
- Training converged very fast, could use high LR(0.01) and train for less epochs
- Adam as main optimizer, occasionally schedule LR, reduce LR on val loss plateau

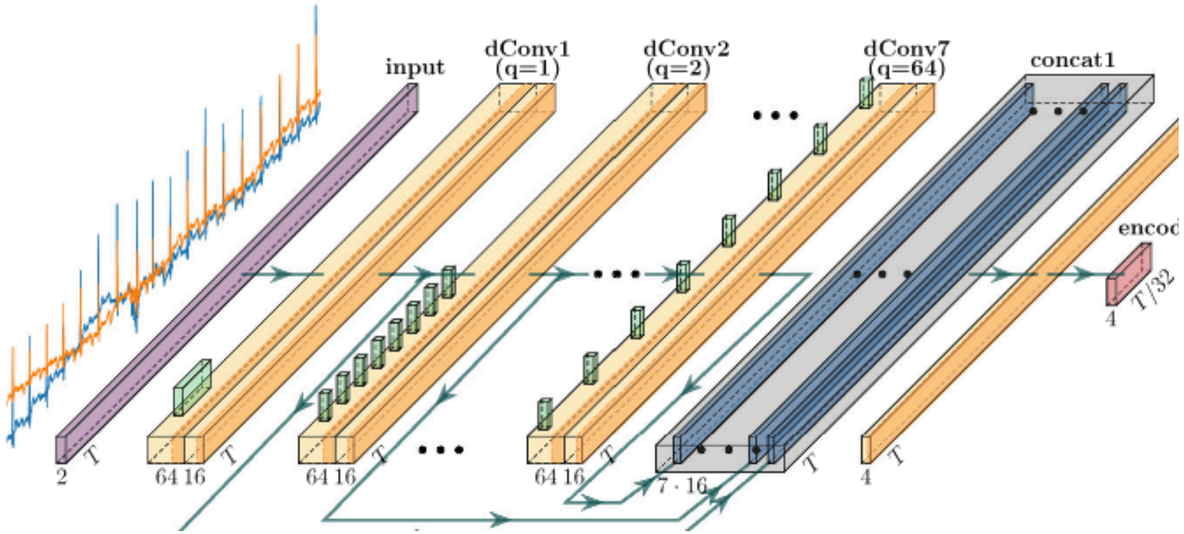
TCNs for TSC - Selected model

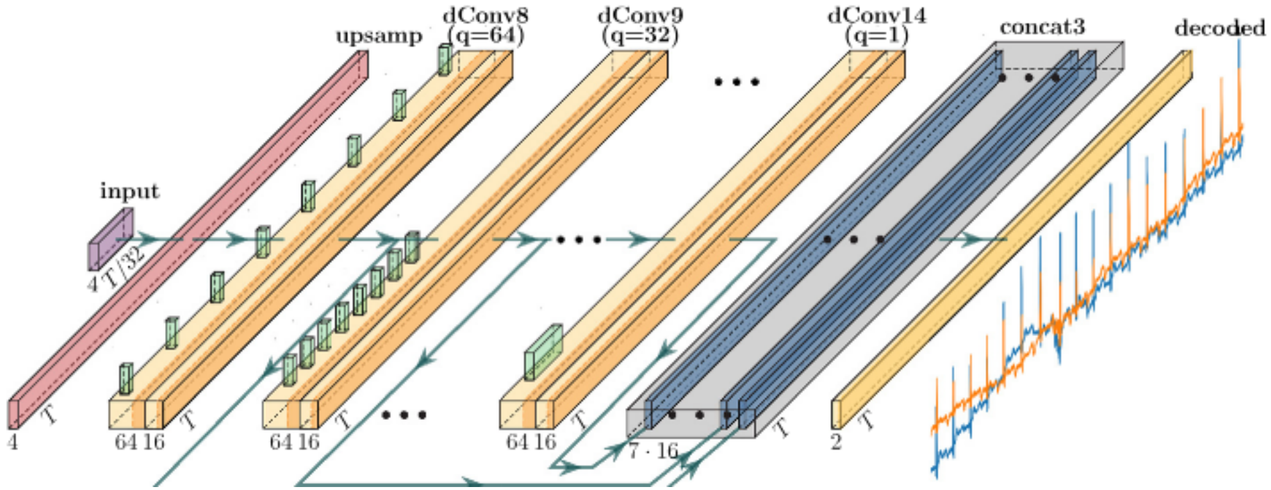
- Final model used base architecture, <400K trainable params
- Good generalization even if overfitting concerns and jumpy training due to high LR
- k=3,d=[1...128],dropout=0.1,filters=64,L2 decay and glorot uniform init on dConvs



TCN-AEs for Unsupervised AD

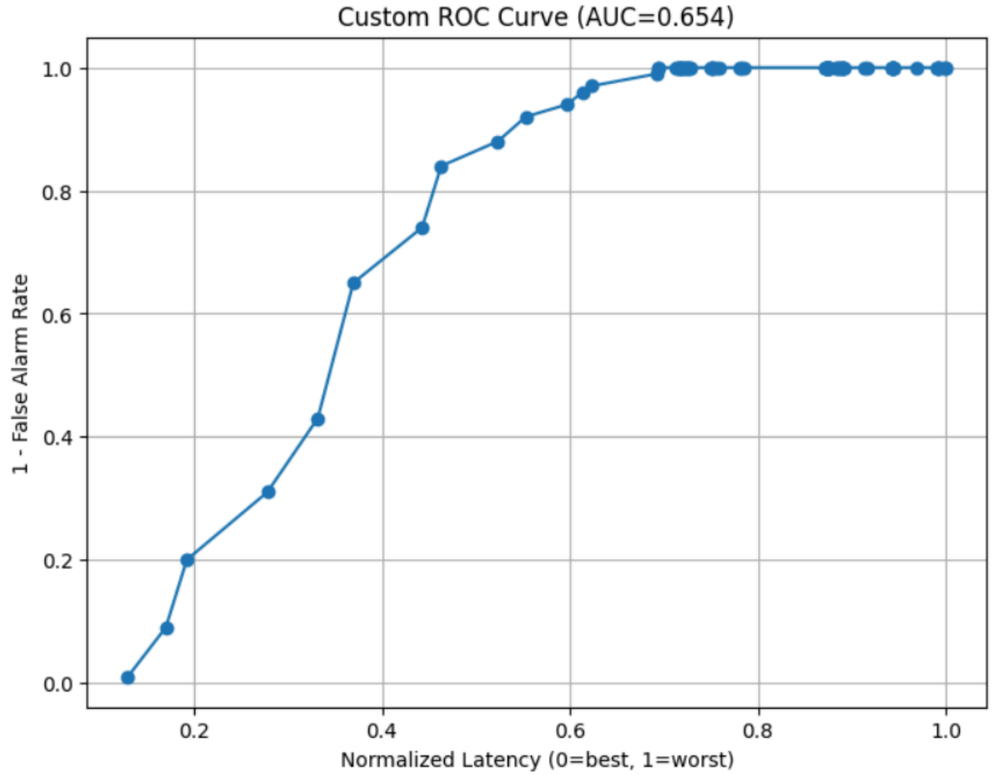
- Goal was to build a good “industrial” model, keeping desirable specifics in mind
- Two main implementations, also because original paper seemed unclear
- Reconstruction based approach trained only on normal data(not a lot)





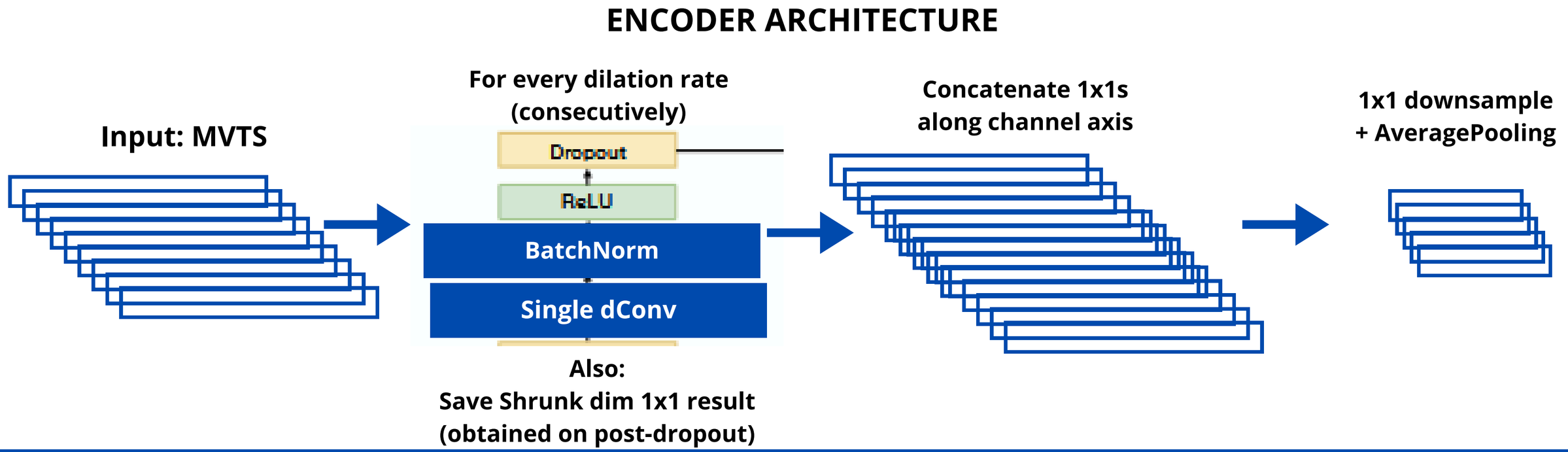
TCNAEs for AD - Notes, Training, Testing, development setup

- Trained only on normal samples(more realistic)
- Validation set to verify training, scaled using train set statistics
- Classifying algorithm based on point-wise mean squared reconstruction error
- Threshold tuning with FAR and Latency in mind(tuned on 10% anom train data)
- Retrained selected config on entire train set before test set evaluation



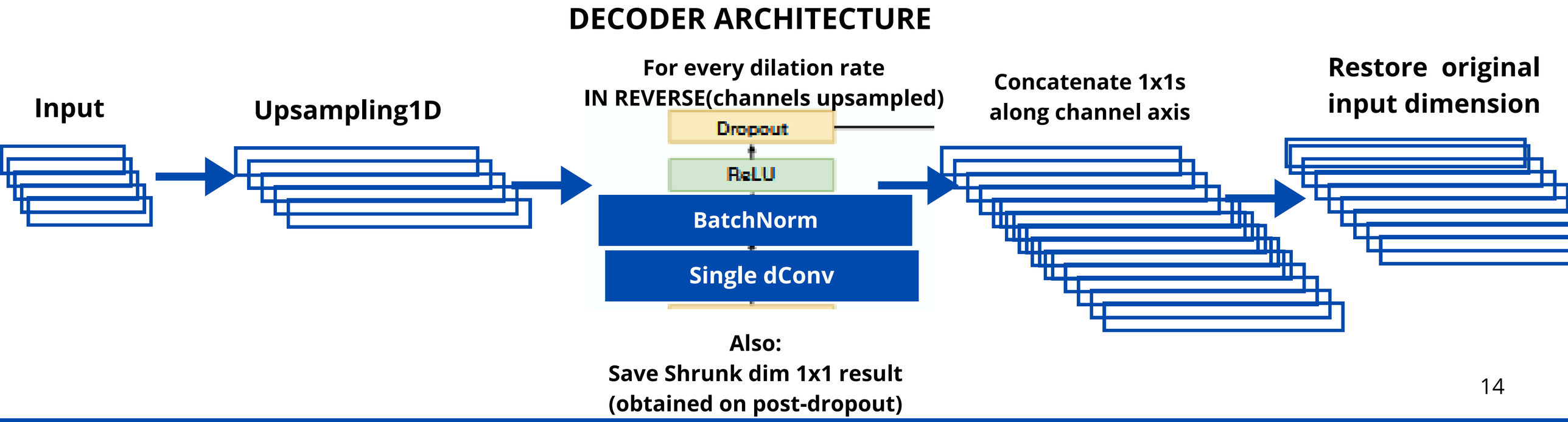
TCNAEs for AD - Outcome and selected model

- More normal samples could have helped model fit, also data aug/generation…
- Final config was again simpler, faster convergence with high learning rate
- Best model did not use kernel regularization, other params same as TCN
- Activations after every intermediate dConv, but not on concatenation
- avg Latency of 14hrs*, 0.2% FAR, 1.74h avg latency when successful, 73% ‘successful’



TCNAEs for AD - Outcome and selected model[cont]

- More normal samples could have helped model fit, also data aug/generation…
- Final config was again simpler, faster convergence with high learning rate
- Best model did not use kernel regularization, other params same as TCN
- Activations after every intermediate dConv, but not on concatenation
- avg Latency of 14hrs*, 0.2% FAR, 1.74h avg latency when successful, 73% ‘successful’



Thanks for your time!

Temporal Convolutional Networks and the Tennesse Eastman Process Dataset

Foundations of Deep Learning 24/25

Università degli Studi di Milano-Bicocca

Master of Science in Data Science

Eduardo Mosca 925279