Using convolutional autoencoders for precipitation nowcasting based on radar data

Andrei Mihai
Babeș-Bolyai University

WeADL 2022 Workshop

The workshop is organized under the umbrella of WeaMyL, project funded by the EEA and Norway Grants under the number RO-NO-2019-0133. Contract: No 26/2020.

Working together for a green, competitive and inclusive Europe

Goal

- Radar data prediction
  - From radar data gathered at one time step predict the radar data at the next time step
  - Very short time forecasting $\Rightarrow$ **nowcasting**
- Nowcasting as classification
  - Predict whether the values at a certain location will be higher or lower than a certain threshold
- Create a machine learning model based on **autoencoders**

Radar Data

- Data collected over central Romania
- Single polarization 458 S-band Weather Surveillance Radar - 98 Doppler (WSR-98D)
  - Full volume scan every 6 minutes

For AutoNowP experiments:

Base Reflectivity product (R)

- estimates the size of water droplets
- expressed in decibels relative to the reflectivity factor Z (dBZ)
- only lowest elevation angle was used

Data model

15	5	10	10	15
10	25	15	5	10
20	0	**10**	10	10
15	10	25	15	5
15	0	15	5	0

Figure: The data matrix at time stamp t. In red is the value of R01 at location $l = (3, 3)$.

20	30	10	15	10
15	15	10	20	5
20	10	**15**	20	25
10	5	10	10	20
15	5	10	15	20

Figure: The data grid at time stamp t-1. In blue is the neighbourhood of the location $l = (3, 3)$ of diameter $d = 3$.

Representation:

The instance corresponding to the location (3,3) at time t is the data grid with the data (15,10,20,10,15, 20,5,10,10) and is labeled with 10 (the value of R01 at location (3,3) and time t).

Data Preprocessing

- Data separated in 2 classes:
  - the *positive* class ("+") – instances having the label higher than a threshold τ
  - the *negative* class ("–") – instances having the label lower or equal to the threshold τ

- Datsets are normalized:

$$R'(l,t) = \frac{R(l,t) - R_{min}}{R_{max} - R_{min}},$$

where:

- $R(l,t)$ is the value of R at time t and location l;
- $R'(l,t)$ is the normalized value of R at time t and location l;
- R_{min} is the minimum value in the domain of R;
- R_{max} is the maximum value in the domain of R.

Autoencoders

- Are a type of *Deep Neural Networks*
- Learn low dimensional representations that capture the relevant characteristics of the input data

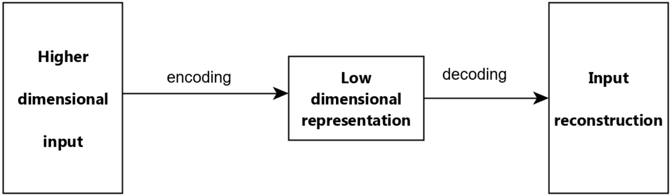

Figure: Abstract representation of an Autoencoder

- Convolutional autoencoders are able to capture spatial patterns in the input data by using convolutions as their building blocks.

AutoNowP Model

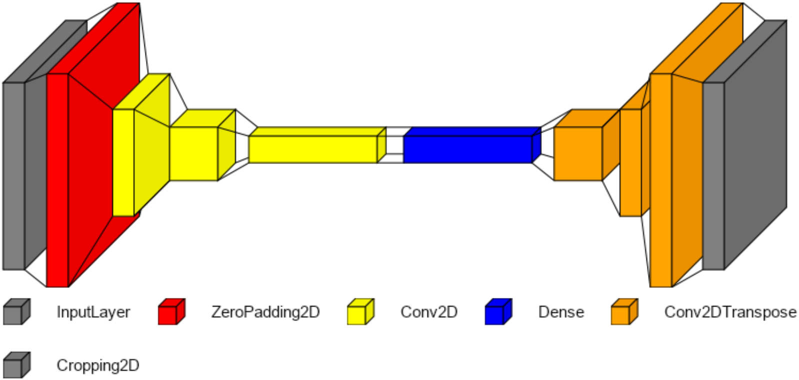

Figure: Architecture of a Convolutional Autoencoder.

Experiment overview

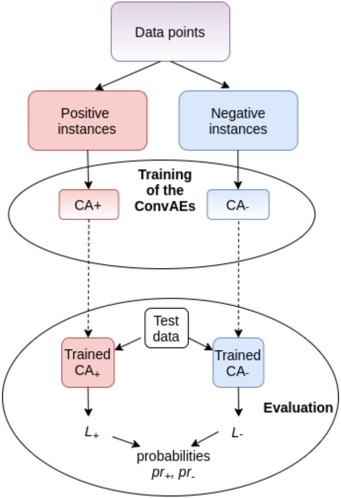

Figure: Overview of *AutoNowP*.

Training Loss Function

$$MSE_{greater}(x, x') = \frac{1}{d^2} \sum_{\substack{1 \leq i \leq d^2 \\ x_i > \tau}} (x_i - x_i')^2 \quad (1)$$

$$MSE_{lesser}(x, x') = \frac{1}{d^2} \sum_{\substack{1 \leq i \leq d^2 \\ x_i \leq \tau}} (x_i - x_i')^2 \quad (2)$$

$$L(x, x') = \alpha \cdot MSE_{greater}(x, x') + (1 - \alpha) \cdot MSE_{lesser}(x, x') \quad (3)$$

where:

- d is the diameter of the neighbourhood used for characterizing the input instances x;
- x instance for which we compute the loss ;
- x' is the autoencoder output for instance x (the reconstruction of x);
- τ is the chosen threshold, that differentiates between positive and negative class;
- α is the parameter that controls prioritization of grater or lesser MSE;
- x_i and x_i' denote the i-th component from x and x' respectively.

Computing Probabilities

$$p_+(q) = 0.5 + \frac{MSE_-(\widehat{q}, q) - MSE_+(\widehat{q}, q)}{2 \cdot (MSE_-(\widehat{q}, q) + MSE_+(\widehat{q}, q))} \tag{4}$$

$$p_-(q) = 1 - p_+(q). \tag{5}$$

where:

- $p_+(q)/p_-(q)$ are the probabilities that the query instance q is in the positive/negative class;
- $MSE_c(\widehat{q}, q)$ the MSE between q and the reconstruction $(\widehat{q})$ of q by the autoencoder CA_c $(c \in \{+, -\})$

Case study

- Dataset: radar data gatherd from **20** days from June 2010, 2017, 2018

Product of interest	**# instances**	**% of "+" instances**	**% of "-" instances**	**Entropy**
R01	9003688	3.44%	96.56%	0.216

Table: Description of the data set.

Metrics Used

- **Critical success index:** $CSI = \frac{TP}{TP+FN+FP}$
- **True skill statistic:** $TSS = \frac{TP \cdot TN - FP \cdot FN}{(TP+FN) \cdot (FP+TN)}$
- **Probability of detection:** $POD = \frac{TP}{TP+FN}$
- *Positive predictive value:* $PPV = \frac{TP}{TP+FP}$
- *Negative predictive value:* $NPV = \frac{TN}{TN+FN}$
- *Specificity:* $Spec = \frac{TN}{TN+FP}$
- *Area Under the ROC Curve:* $AUC = \frac{Spec+POD}{2}$
- *Area Under the Precision-Recall Curve:* $AUPRC = \frac{Precision+Recall}{2}$

Results per threshold

τ	*CSI*	*TSS*	*POD*	*PPV*	*NPV*	*Spec*	*AUC*	*AUPRC*
10	**0.615** ± 0.018	**0.861** ± 0.012	**0.876** ± 0.012	**0.674** ± 0.017	**0.996** ± 0.001	**0.985** ± 0.002	**0.931** ± 0.006	**0.775** ± 0.013
20	0.425 ± 0.072	0.471 ± 0.091	0.474 ± 0.092	0.810 ± 0.015	0.989 ± 0.001	0.997 ± 0.001	0.736 ± 0.046	0.642 ± 0.039
30	0.151 ± 0.046	0.157 ± 0.051	0.157 ± 0.028	0.812 ± 0.031	0.993 ± 0.001	1.000 ± 0.000	0.579 ± 0.014	0.485 ± 0.007

Table: Experimental results for different thresholds, with 95% CI

Key takes:

- In general performance decreases when threshold increases, as imbalance increases
- *Specificity* and *PPV* increases with threshold, since the number of False Positives decreases due to fewer positive values

Results - comparison to other classifiers

Model	*CSI*	*TSS*	*POD*	*PPV*	*NPV*	*Spec*	*AUC*	*AUPRC*
AutoNowP	**0.615** ± 0.018	**0.861** ± 0.012	**0.876** ± 0.012	**0.674** ± 0.017	**0.996** ± 0.001	**0.985** ± 0.002	**0.931** ± 0.006	**0.775** ± 0.013
Logistic Regression	0.672 ± 0.012	0.752 ± 0.013	0.757 ± 0.013	0.857 ± 0.005	0.992 ± 0.001	0.996 ± 0.000	0.876 ± 0.007	0.807 ± 0.008
Linear Support Vector Classifier (SVC)	0.685 ± 0.012	0.778 ± 0.007	0.783 ± 0.007	0.845 ± 0.015	0.992 ± 0.000	0.995 ± 0.000	0.889 ± 0.003	0.814 ± 0.009
Decision Trees	0.574 ± 0.007	0.725 ± 0.004	0.734 ± 0.006	0.724 ± 0.012	0.991 ± 0.001	0.990 ± 0.002	0.862 ± 0.002	0.729 ± 0.006
Nearest Centroid Classification	0.571 ± 0.006	0.793 ± 0.013	0.807 ± 0.013	0.662 ± 0.015	0.993 ± 0.001	0.986 ± 0.001	0.896 ± 0.006	0.735 ± 0.003

Table: Comparative results between *AutoNowP* and other classifiers. 95% CIs are used for the results.

Thank you!
Questions?