PADERBORN UNIVERSITY

PADERBORN UNIVERSITY

MONOTONICITY IN DEEP LEARNING: STATE OF THE ART

SEMINAR: MACHINE LEARNING PRESENTATION

Varun Nandkumar Golani

Paderborn, February 18, 2021

Introduction

- Deep Neural Network(s) (DNN): State-of-the-art results on tasks like image, speech recognition etc. [3, 4].

Introduction

- Deep Neural Network(s) (DNN): State-of-the-art results on tasks like image, speech recognition etc. [3, 4].
- Monotonicity: Increase in the input variable results in increase (or decrease) of an output variable [1]

Introduction

- Deep Neural Network(s) (DNN): State-of-the-art results on tasks like image, speech recognition etc. [3, 4].
- Monotonicity: Increase in the input variable results in increase (or decrease) of an output variable [1]
- Input feature $\uparrow$, output variable $\uparrow$ ($\downarrow$) $\Rightarrow$ Monotonic increasing (decreasing)

Introduction

- Deep Neural Network(s) (DNN): State-of-the-art results on tasks like image, speech recognition etc. [3, 4].
- Monotonicity: Increase in the input variable results in increase (or decrease) of an output variable [1]
- Input feature $\uparrow$, output variable $\uparrow$ ($\downarrow$) $\Rightarrow$ Monotonic increasing (decreasing)
- E.g.: Monotonic decreasing trend, SCHUFA score (German credit score) and the loan interest rates

Introduction

- Deep Neural Network(s) (DNN): State-of-the-art results on tasks like image, speech recognition etc. [3, 4].
- Monotonicity: Increase in the input variable results in increase (or decrease) of an output variable [1]
- Input feature $\uparrow$, output variable $\uparrow$ ($\downarrow$) $\Rightarrow$ Monotonic increasing (decreasing)
- E.g.: Monotonic decreasing trend, SCHUFA score (German credit score) and the loan interest rates
- All (Subset) input features, output variable $\Rightarrow$ Total (Partial) monotonicity

PADERBORN UNIVERSITY

Categorization of techniques to enforce Monotonicity

- Configuration of the Network Architecture (CNA)
  - Special layers which ensures monotonicity
  - Positive (or Negative) weight constraints on the monotone input features
- Penalizing Non-Monotonicity (PNM)
  - Penalty for non-monotonicity in the loss function
  - General loss function: training error + non-monotonicity error + regularization term (if required)

Outline

Tests for monotonicity

- Heuristic Test
- Global Monotonicity Verification (GMV)

Techniques for enforcing monotonicity

- Configuration of the Network Architecture (CNA)
  - Monotone MIN-MAX Networks (MMMN)
  - Deep Lattice Networks (DLN)
- Penalizing Non-Monotonicity (PNM)
  - Domain Adapted Neural Network(s) (DANN)
  - Point-wise Loss (PWL)
  - Certified Monotonic Neural Network(s) (CMNN)

Heuristic Test [7]

- Degree of Monotonicity $DgrMon(D) = \dfrac{\#Monotone\ Pairs(D)}{\#Comparable\ Pairs(D)} \in [0,1]$

Removed variable(s)	Comparable pairs	DgrMon
- (original data)	5717	0.9645
AGE	6984	0.9255
KM	7635	0.9280
WEIGHT	6131	0.9444
COLOR	7714	0.9615
CC	6018	0.9658
HP	6799	0.9657
QUART-TAX	6402	0.9647

Figure: Heuristic test for Toyota Car Dataset. Image Source: [7]

Global Monotonicity Verification (GMV) [2]

- Checks for all the data points in the input domain $\mathcal{X}$
- Solves the problem via an optimization approach
- Let f be a diff. fn., x_α be a subset of input features that are monotonic
- To verify monotonicity we have to show $\partial_{x_l} f(x) \geq 0,$ *for all* $l \in \alpha,\ x \in \mathcal{X}$
- Therefore, monotonicity can be verified by solving:
  - $U_\alpha := \min_{x,\, l \in \alpha} \{\partial_{x_l} f(x), x \in \mathcal{X}\}$
- For verification it is sufficient to show that $U_\alpha \geq 0$

Outline

Tests for monotonicity

- Heuristic Test
- Global Monotonicity Verification (GMV)

Techniques for enforcing monotonicity

- Configuration of the Network Architecture (CNA)
  - Monotone MIN-MAX Networks (MMMN)
  - Deep Lattice Networks (DLN)
- Penalizing Non-Monotonicity (PNM)
  - Domain Adapted Neural Network(s) (DANN)
  - Point-wise Loss (PWL)
  - Certified Monotonic Neural Network(s) (CMNN)

Monotone MIN-MAX Networks (MMMN) [5]

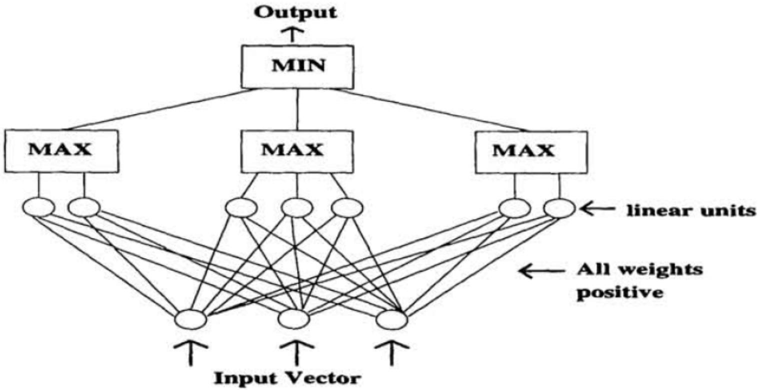

Figure: Architecture of a 3 layer Total Monotonic MIN-MAX Network. Image Source: [5]

Deep Lattice Networks (DLN) [8]

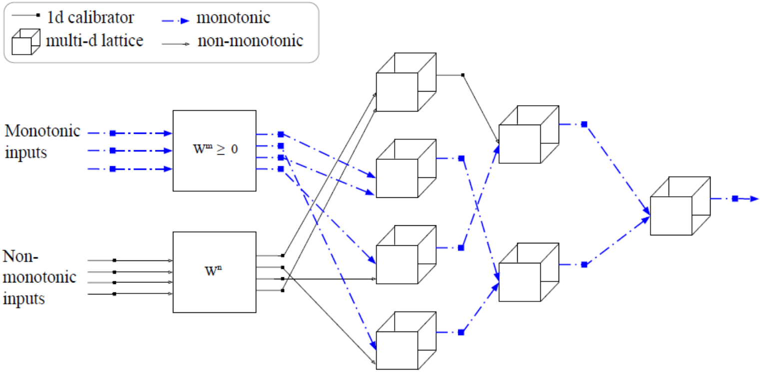

Figure: An example of a 9 layer DLN: calibrators, linear embedding, calibrators, ensemble of lattices, calibrators, ensemble of lattices, calibrators, lattice, calibrator. Image Source: [8]

Outline

Tests for monotonicity

- Heuristic Test
- Global Monotonicity Verification (GMV)

Techniques for enforcing monotonicity

- Configuration of the Network Architecture (CNA)
  - Monotone MIN-MAX Networks (MMMN)
  - Deep Lattice Networks (DLN)
- Penalizing Non-Monotonicity (PNM)
  - Domain Adapted Neural Network(s) (DANN)
  - Point-wise Loss (PWL)
  - Certified Monotonic Neural Network(s) (CMNN)

Domain Adapted Neural Network(s) (DANN) [9]

- Incorporates monotonic domain knowledge in the training process
- $Loss_D(\hat{Y}_1, \hat{Y}_2) = \sum_{i=1}^{m} \mathbb{I}\left((x_1^i < x_2^i) \wedge (\hat{y}_1^i > \hat{y}_2^i)\right).ReLU(\hat{y}_1^i - \hat{y}_2^i)$
- Interpretation of $Loss_D(\hat{Y}_1, \hat{Y}_2)$:
  - $x_1^i < x_2^i$ and $\hat{y}_1^i < \hat{y}_2^i \Rightarrow$ monotonic behaviour $\Rightarrow [Loss_D(\hat{Y}_1, \hat{Y}_2)]_i = 0$
  - $x_1^i < x_2^i$ and $\hat{y}_1^i > \hat{y}_2^i \Rightarrow$ non-monotonic relationship $\Rightarrow$ $[Loss_D(\hat{Y}_1, \hat{Y}_2)]_i = Penalty$
  - ReLU part of the equation can be thought of as a weight i.e. magnitude of the error

Point-wise Loss (PWL) [6]

- Incorporates a priori knowledge regarding monotonicity
- Can be used with any NN/DNN architecture
- Partial monotonicity with NN/DNN
- Penalizes negative (non-monotonic) gradients
- Objective Function:
  $min\ \mathcal{L}_{\text{mono}} = min \left\{ \sum_{i=1}^{n} max(0, -\nabla_{\cdot M} f(x_{\text{i}}; \theta)) + \mathcal{L}_{\text{NN}} \right\}$
- Does not guarantee monotonicity

Certified Monotonic Neural Network(s) (CMNN) [2]

- NN/DNN guaranteed to be monotonic
- Two step process:
  - Step 1: Train an NN f with loss function $\mathcal{L}(f) + \lambda R(f)$
    - $R(f) = 0$ means that f is monotonic with respect to the monotonic features x_α in the dataset
  - Step 2: Check if $U_\alpha \geq 0$ (Test: GMV)
    - If Yes, the algorithm terminates
    - If No, increase λ and go back to Step 1
- GMV extended to DNN by considering them as two layer networks

Summary & Comparison

Table: The best state-of-the-art technique to enforce monotonicity is **CMNN** [2].

	MMMN [5], [7] (1998,2010)	DLN [8] (2017)	PWL [6] (2019)	CMNN [2] (2020)	DANN [9] (2018)
Category	CNA	CNA	PNM	PNM	PNM
How is Mono. enforced?	Positive weight constraints, MIN and MAX layers	Linear Embedding, Calibration & Ensemble of Lattice layers	Loss function by penalizing negative gradients	2 steps, Loss Function + Global Mono. Verification	Mono. constraint in the loss function
Test for Mono.	Yes	No	No	Yes	No
Mono. Guarantee	Yes	Yes	No	Yes	No
Can the technique be applied to existing NN/DNN architectures?	No	No	Yes	Yes	Yes

Conclusion

- Monotonicity increases the interpretability of the black box NN/DNN models [2]

Conclusion

- Monotonicity increases the interpretability of the black box NN/DNN models [2]
- Monotonicity makes the learner biased towards specific hypothesis in the hypothesis space

Conclusion

- Monotonicity increases the interpretability of the black box NN/DNN models [2]
- Monotonicity makes the learner biased towards specific hypothesis in the hypothesis space
- In some cases, monotonicity can improve performance in situations with limited data, data with poor quality and noisy training data [9]

Conclusion

- Monotonicity increases the interpretability of the black box NN/DNN models [2]
- Monotonicity makes the learner biased towards specific hypothesis in the hypothesis space
- In some cases, monotonicity can improve performance in situations with limited data, data with poor quality and noisy training data [9]
- Monotonicity should only be used if there exists a total/partial monotonic relationship between the input features and output variable

References I

- Toptal, https://www.toptal.com/machine-learning/monotonic-ai-models
- Liu X., Han X., Zhang N., Liu Q.: Certified Monotonic Neural Networks. In: 34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.
- He K., Zhang X., Ren S., Sun J.: Deep Residual Learning for Image Recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016). doi:10.1109/CVPR.2016.90
- Heymann J., Drude L., Haeb-Umbach R.: Wide Residual BLSTM Network with Discriminative Speaker Adaptation for Robust Speech Recognition. Computer Speech and Language (2016).

References II

- Sill, J.: Monotonic Networks. In: Jordan M., Kearns M., and Solla S. (eds.), Advances in Neural Information Processing Systems 10, pages 661–667. MIT Press (1998). `http://papers.nips.cc/paper/1358-monotonic-networks.pdf`
- Gupta, A., Shukla, N., Marla, L., Kolbeinsson, A., Yellepeddi, K.: How to Incorporate Monotonicity in Deep Networks While Preserving Flexibility?. In: NeurIPS 2019 Workshop on Machine Learning with Guarantees, Vancouver, Canada (2019).
- Daniels H., Velikova M.: Monotone and partially monotone neural networks. In: IEEE Transactions on Neural Networks, 21(6):906–917 (2010). ISSN 1045-9227. doi:10.1109/TNN.2010.2044803

References III

You S., Ding D., Canini K., Pfeifer J., Gupta M.: Deep lattice networks and partial monotonic functions. In: Guyon I., Luxburg U. V., Bengio S., Wallach H., Fergus R., Vishwanathan S., Garnett R. (eds.), Advances in Neural Information Processing Systems 30, pages 2981–2989. Curran Associates, Inc. (2017).

Muralidhar N., Islam M. R., Marwah M., Karpatne A., Ramakrishnan N.: Incorporating Prior Domain Knowledge into Deep Neural Networks. In: IEEE International Conference on Big Data (2018). doi:10.1109/BigData.2018.8621955