Adversarially Constrained Autoencoder Interpolations using Wasserstein Autoencoder

Machine Learning

Lorenzo Palloni

University of Florence

lorenzo.palloni@stud.unifi.it

April 16, 2020

Introduction

- **Unsupervised Learning** context
- we aim to obtain "high-quality" **interpolations**
- interpolations example:

ooo INSERT AN IMAGE HERE ooo

Motivation

- uncover underlying structure of dataset
- better representations $\rightarrow$ better results in other tasks

Entity Embedding

- Entity Embedding
- maps each state of a categorical variable

$$x \in \left\{ \text{'red'},\ \text{'green'},\ \text{'blue'} \right\}$$

- in a D-dimensional Euclidean space
- where $D \in \mathbb{N}^+$ is user-defined[1]

$$x \in \left\{ [0.5,\ -1.2],\ [1.3,\ 0.23],\ [0.4,\ 1.1] \right\}.$$

[1] D might be chosen in range $[1,\ K-1]$.

Motivation

- Let x be a categorical variable with

  11981 different states.

- One Hot Encoding representation of x needs

  11981-dimensional vectors.

- Entity Embedding representation of x might be e.g.

  19-dimensional vectors.

- Explosions in dimensionality like this leads to
  1. drop in prediction performance (overfitting);
  2. computational cost in space and time.

Experiments - Dataset

- Dataset take from a Kaggle competition called

  $\rightarrow$ Categorical Feature Encoding Challenge;

  - $300k$ observations;
  - 23 variables (all categorical);
  - binary problem ($y \in \{0, 1\}$).

- Dataset divided into
  - 80% $\rightarrow$ train
  - 20% $\rightarrow$ test

Experiments - Neural Network hyperparameters

- To extract the Entity Embeddings we use the following architecture:
  1. input layer: concatenation of embedded features + other variables;
  2. first layer: 400 hidden units and ReLU activation;
  3. second layer: 600 hidden units and ReLU activation;
  4. output layer: logistic function.
- Training hyperparameters:
  - number of epochs: 2
  - number of observations per mini-batch: 32
  - optimization algorithm: Adam[2] (default values)
- Implementation in Tensorflow[3].

Experiments - Random Forest hyperparameters

- **Random search** with 4-fold **cross-validation** on:
  - number of decision trees:
    - 125
    - → 175
  - maximum number of features used by each tree in each split:
    - → 'sqrt'
    - 'log2'
  - max depth of each tree:
    - 10
    - → 20
    - None
  - minimum number of samples needed to perform a split:
    - 2
    - → 6

Experiments - Random Forest Results

	AUC
Train	0.9879
Test	**0.6121**

Figure: Random Forest + Entity Embeddings results.

	AUC
Train	0.6818
Test	**0.5640**

Figure: Random Forest + One Hot Encoding[2] results.

[2] Variables with max 50 states used.

Conclusion

- **Entity Embedding** is an useful technique to put into your **toolbox**;
- in some situations can lead to a **crucial** saving in computational resources.

References

Guo, C., & Berkhahn, F. (2016). Entity embeddings of categorical variables. arXiv preprint arXiv:1604.06737.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... & Kudlur, M. (2016). Tensorflow: A system for large-scale machine learning. In 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16) (pp. 265-283).