Explainable AI

Manish Reddy Challamala
Department of Computer Science
University at Buffalo
Buffalo, NY 14214
manishre@buffalo.edu

Abstract

The objective of this project is to develop a machine learning model which can learn the explainable features for a task domain and learn to answer the variety of queries in that domain.

1 Introduction

XAI (explainable Artificial Intelligence), is the new perspective of machine learning, to produce more explainable models that can establish the trust by enabling the users to understand how the model is predicting the output by characterizing the features strength and weakness.
In this project we are going to combine both deep learning and Probabilistic graphical model, to achieve that a model could learn and answer variety of queries.
There are three different datasets of features:

1. Human determined features.
2. Deep learning features.
3. Explainable deep learning features

2 Theory

2.1. Probabilistic Graphical Models

Probabilistic graphical model is a structured model which represents the conditional independencies between random variables.

There are two types of network structures:

2.1.1. Bayesian network

2.1.2. Markov network

2.1.1 Bayesian Network

- Bayesian network is probabilistic graphical model for representing the multivariate probability distributions, in which nodes represent the random variables and the edges represent conditional probability distributions [CPD's] tables between random variables, which are used to calculate the probability dependencies between the variables.
- Bayesian networks are also called as belief or causal networks because they are directed acyclic graphs [DAG], even with a change of CPD at one node can affect the whole networks performance.
- The probability distribution is defined in the form of factors:

$$P(X_1, X_2, \dots, X_n) = \prod P(X_i | \prod X_i)$$

Where $\prod_{i=1}^{N} P(X_i | \prod X_i)$ *is* $P(node | parent(node))$

2.1.2 Markov network

- Markov networks are undirected acyclic graphs which are similar to the Bayesian network. As the graphs are undirected, instead of edes they have cliques which connect each node with their neighboring nodes.
- The probability distribution is defined in the form of factors of potential functions. Which can be written in the log-linear form.
- As there is no topological order for the Markov network, we use potential functions for each clique in the graphs.
- Joint distribution in the Markov network is proportional to the product of clique potentials.
- The conditional probability for Markov network is defined as

$$P(y|\theta) = \frac{1}{Z(\theta)} \prod \phi(y_c|\theta_c)$$

  Where $Z(\theta)$ is the partial function derives as summation of products of all potential factors.

2.2 Deep Learning:

Deep learning is the representation of multi-layer artificial neural network which can automatically learn the feature representations from the data provided. As the algorithms are learning the features from the data, we can apply the deep learning algorithms to supervised, unsupervised and semi-supervised learning problems.

We can use two types of models

2.2.1 Siamese network

2.2.2 Auto-Encoders

2.2.1 Siamese network

The idea of Siamese is twin network, where the two networks can share the same weights among them to learn the useful features that can help to compare between the inputs of the respective subnets.

Mostly, Siamese networks use the binary classification has activation function at the output, to classify whether the two inputs are the same class or not.

2.2.2 Auto-encoder

- An auto encoder is the unsupervised learning algorithm that applies backpropagation, so that setting of target variable is equal to input variable.
- The auto encoder tries to learn an approximation to a function such that output $x\hat{}$ is similar to input x.
- We can divide the autoencoder model into two structures encoder and decoder.
- The encoder model is forced to learn the compressed knowledge representations of the input, for example given an input image the encoder will detect the most important features of the input.
- The decoder on the other hand takes these features from encoder as input and try to reconstruct the input.
- So, from the above two statements, we can understand that the encoder and decoder share same structure.

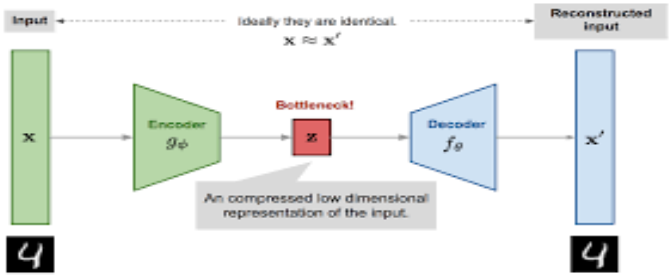


Fig: 2. Block Diagram of auto-encoder

3 Explanation:

3.1 Task1: Data Annotation.

In this task, we are manually going through all of the images in the dataset and assigning each feature of the image to specific class. In total there are 15 features each consists of different class. The 15 features and no of classes each feature contains is displayed below.

Table 1: Representation of no of classes each feature can take

Feature	**No of classes**
Pen_pressure	2
Letter_spacing	3
Size	3
Dimensions	3
Is_lowercase	2
Is_continuous	2
Slatness	4
Tilt	2
Entry stroke_'a'	2
Staff of 'a'	4
Formation of 'n'	2
Staff of 'd'	3
Exit stroked d	4
Word formation	2
constancy	2

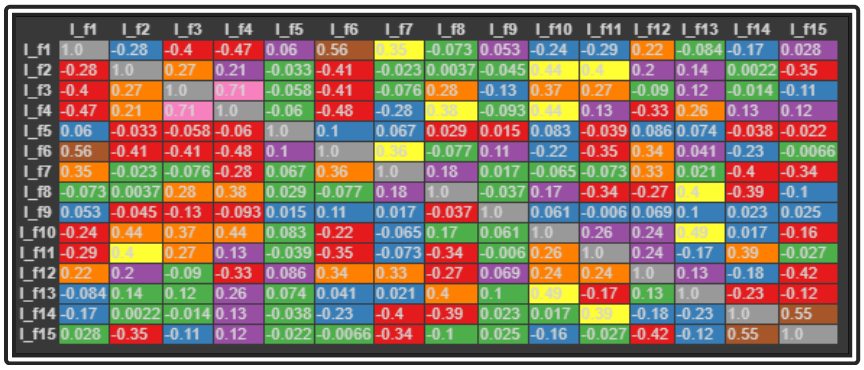
3.1 Task2: Sample Verification of PGM:

In this task, we are using the Bayesian networks to create a model and infer the value, which says weather the two images are similar or not.

Creating five Bayesian networks:

- We are creating a Bayesian network structure [f], and making a copy of the structure 'f' and naming the other structure as [g].
- The edges for a Bayesian network structure is selected by using the Correlation values.

  The correlation values for the 15 features are given below:



	l_f1	l_f2	l_f3	l_f4	l_f5	l_f6	l_f7	l_f8	l_f9	l_f10	l_f11	l_f12	l_f13	l_f14	l_f15
l_f1	1.0	-0.28	-0.4	-0.47	0.06	0.56	0.35	-0.073	0.053	-0.24	-0.29	0.22	-0.084	-0.17	0.028
l_f2	-0.28	1.0	0.27	0.21	-0.033	-0.41	-0.023	0.0037	-0.045	-0.44	0.4	0.2	0.14	0.0022	-0.35
l_f3	-0.4	0.27	1.0	0.71	-0.058	-0.41	-0.076	0.28	-0.13	0.37	0.27	-0.09	0.12	-0.014	-0.11
l_f4	-0.47	0.21	0.71	1.0	-0.06	-0.48	-0.28	0.38	-0.093	-0.44	0.13	-0.33	0.26	0.13	0.12
l_f5	0.06	-0.033	-0.058	-0.06	1.0	0.1	0.067	0.029	0.015	0.083	-0.039	0.086	0.074	-0.038	-0.022
l_f6	0.56	-0.41	-0.41	-0.48	0.1	1.0	0.36	-0.077	0.11	-0.22	-0.35	0.34	0.041	-0.23	-0.0066
l_f7	0.35	-0.023	-0.076	-0.28	0.067	0.36	1.0	0.18	0.017	-0.065	-0.073	0.33	0.021	-0.4	-0.34
l_f8	-0.073	0.0037	0.28	0.38	0.029	-0.077	0.18	1.0	-0.037	0.17	-0.34	-0.27	0.4	-0.39	-0.1
l_f9	0.053	-0.045	-0.13	-0.093	0.015	0.11	0.017	-0.037	1.0	0.061	-0.006	0.069	0.1	0.023	0.025
l_f10	-0.24	0.44	0.37	0.44	0.083	-0.22	-0.065	0.17	0.061	1.0	0.26	0.24	0.48	0.017	-0.16
l_f11	-0.29	0.4	0.27	0.13	-0.039	-0.35	-0.073	-0.34	-0.006	0.26	1.0	0.24	-0.17	0.39	-0.027
l_f12	0.22	0.2	-0.09	-0.33	0.086	0.34	0.33	-0.27	0.069	0.24	0.24	1.0	0.13	-0.18	-0.42
l_f13	-0.084	0.14	0.12	0.26	0.074	0.041	0.021	0.4	0.1	0.48	-0.17	0.13	1.0	-0.23	-0.12
l_f14	-0.17	0.0022	-0.014	0.13	-0.038	-0.23	-0.4	-0.39	0.023	0.017	0.39	-0.18	-0.23	1.0	0.55
l_f15	0.028	-0.35	-0.11	0.12	-0.022	-0.0066	-0.34	-0.1	0.025	-0.16	-0.027	-0.42	-0.12	0.55	1.0

Figure 1: correlation values for 15 features.

- There are no edge connections between the two structures f and g.
- Connecting some nodes of the Bayesian network structures to the verification node [which handles the label: whether two images are similar or not]

Train the networks on the data:

- We are using model.fit(data) function to train the models of the data.
- By using the K2 Score we are finding the best model out of five models created.

Inferring a Query:

- We are using the variable elimination algorithm to infer the query.
- We loop over all the variables as ordered by o and eliminate them in that ordering.

Algorithm:

- for each variable X_i (ordered according to o),
  Multiply all factors ϕ_i containing X_i .
  Marginalize out X_i to obtain a new factor τ .
  Replace the factors ϕ_i with τ.
- we use model.query() function by passing 16 features [15 features of left image and 1 feature of verification node] and infer the CPD tables for the right image. We will run the query by assigning the verification node value as both 1 and 0.
- As we all ready have the features of right image, we take those feature values as index and multiply all the respective values from CPD table.
- We will perform the log-likelihood ratio for these product values and compare that to a threshold value of 0.2.
- If the value is below the threshold value, we say that the images are not similar and vice-versa.

Evaluation:

- Compare the log-likelihood value with the original label to determine the accuracy and loss of the model.

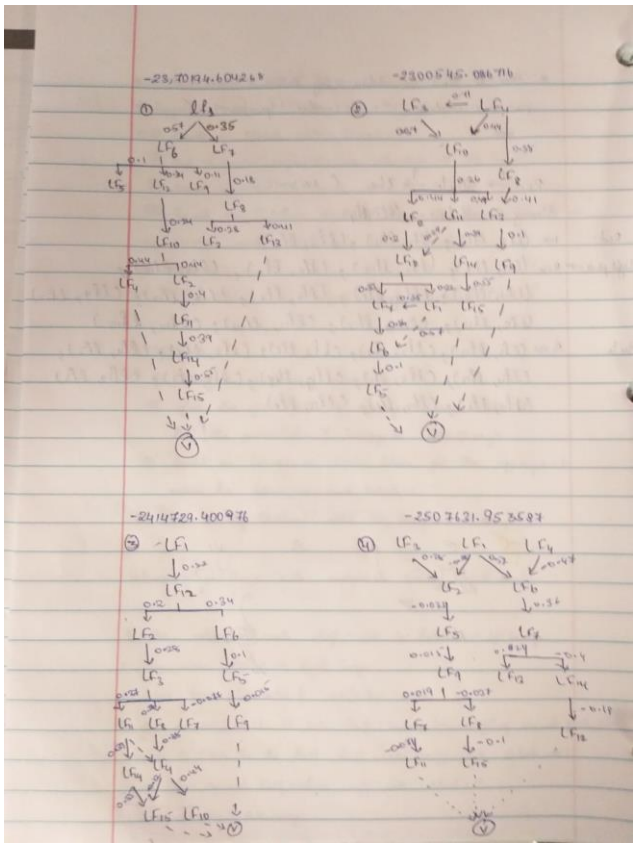


Figure 2 : Bayesian models

Table 2: K2 Scores of models

Model	**K2 Score**
Model 1	-2065660.94 [high probability model]
Model 2	-2370194.60
Model 3	-2300545.8
Model 4	-2414729.40
Model 5	-2507631.95 [Low probability model]

Table 3: Evaluation metrics for Seen Dataset

Accuracy	64.8
Number of edges	56
Time to train all models	2.8 minutes (approximately)
Time to train best model	152 seconds
Time to infer	7200 seconds

Table 4: Evaluation metrics for Unseen Dataset

Accuracy	62.9
Number of edges	56

Time to train all models	2.8 minutes (approximately)
Time to train best model	152 seconds
Time to infer	5400 seconds

Table 5: Evaluation metrics for Shuffled Dataset

Accuracy	59.8
Number of edges	56
Time to train all models	2.8 minutes (approximately)
Time to train best model	152 seconds
Time to infer	5608 seconds

3.2 Task3: Deep learning Inference

3.2.1 Siamese network

Create a Convolution 2D Siamese network:

- A convolution 2D Siamese network is created using keras.
- Inputs to the networks will be the pair of images. [left image to left Siamese network and right image to right Siamese network.]
- The network will have many layers which will perform convolution, max pooling operations with 'relu' and sigmoid as activation function, optimizer used is adadelta and loss function as binary_ cross entropy.

Issue with Siamese network:

- While working with Siamese network the training and validation accuracy is ranging from 0.5 to 0.65 (approximately). The problem might be that the model is not training properly.

3.2.2 Auto-encoder

Creating the Auto-encoder

- An auto-encoder is created using convolution 2d layers, where the model performs the convolution 2D and max pooling operations with relu and sigmoid as activation function.
- The encoder section of the model learns the knowledge representation of the input.
- The output of the encoder will be a vector of 1 x512.
- The decoder section of the model will take these 512 features as input and tries to reconstruct the image.

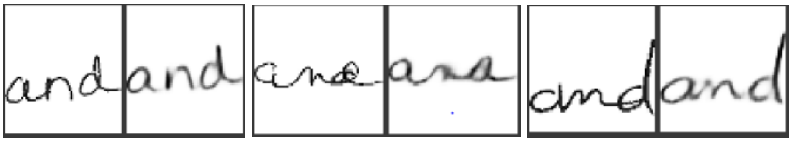


Figure 2: Reconstructed images from decoder

- Advantage of using the auto-encoder is that as we are reconstructing the images, we can keep a check whether the model is properly trained or not.
- The model is trained using the 'fit_generator()' method.

Evaluation

- The model is evaluated using the cosine similarity formula.
- First, we are predicting the images using predict function.
- we are extracting the latent features of predicted images.

Table 6: output of Latent features.

Sno	Index	Identity	Imagename	latent
1	6	0001	0001a_num1.png	[0.0, 0.25730813, 0.0, 0.656732, 0.0, 0.0, 1.4...

- The cosine similarity for latent features obtained by using cosine similarity function.

Evaluation metrics for Auto-encoder

Table 6: Evaluation metrics for Seen Dataset

Accuracy	0.95
Validation Accuracy	0.91
Validation loss	0.09

Table 7: Evaluation metrics for Unseen Dataset

Accuracy	0.92
Validation Accuracy	0.91
Validation loss	0.09

Table 8: Evaluation metrics for ShuffledDataset

Accuracy	0.94
Validation Accuracy	152 seconds
Validation loss	0.09

3.2.3. Explainable AI

Creating the auto encoder model

Creating the auto encoder model same as above process.
Extracting the 512 latent features from the model.
Passing the feature values to neural network and obtaining the one hot vector of the feature.

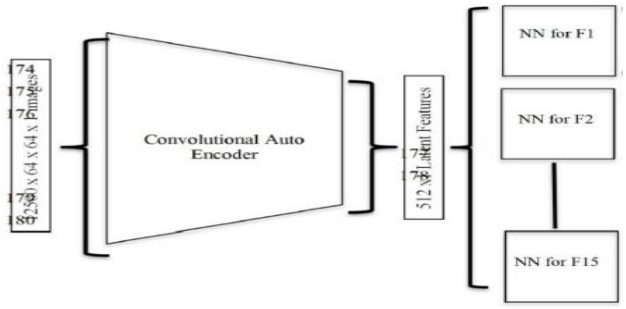


Figure 4: Block-diagram of auto encoder.

Feature	Real image	Predicted image
Pen_pressure	2	1
Letter_spacing	2	2
Size	1	1
Dimensions	1	1
Is_lowercase	2	2
Is_continuous	2	1
Slatness	2	2
Tilt	2	1
Entry stroke_'a'	2	2
Staff of 'a'	2	2
Formation of 'n'	2	2
Staff of 'd'	2	2
Exit stroked d	3	2
Word formation	2	2
constancy	1	2

From the above table we can see that for the given image the model predicted some features correctly which are colored in green and some features are wrongly predicted which are colored red

Evaluation:

Table 9: Evaluation metrics for Seen Dataset

Accuracy	0.82
Validation Accuracy	7.70

Table 10: Evaluation metrics for Unseen Dataset

Accuracy	0.61
Validation Accuracy	7.70

Table 11: Evaluation metrics for ShuffledDataset

Accuracy	0.62
Validation Accuracy	7.70