Assignment No. 2

1 TITLE

Analysis of algorithms and mathematical modelling.

2 AIM

Improving Human Computer Interaction with Machine Emotional Intelligence using NAO robot.

3 OBJECTIVE

3.1 To learn analysis of algorithm

3.2 To learn algorithmic strategies

3.3 To draw P and NP for project

3.4 To learn how to draw mathematical model for different models of the project

4 NEED FOR ANALYSIS TO BE DONE

Informally an algorithm is any well-defined computational procedure that takes some value or a set of values as input and produces some value, or set of values as output. An algorithm is thus a sequence of computational steps that transform the input into output. To analyse an algorithm is to determine the amount of resources (such as time and storage) necessary to execute it. In order to choose the best algorithm for a particular task, we need to be able to judge how long a particular solution will take to run; or how long two solutions will take to run and choose the better of the two. We don't need to know how many minutes and seconds they will take, but we need some way to compare algorithms against one another. This is why we need to analyse an algorithm.

5 TIME COMPLEXITY

P, NP, NP-complete, NP-hard

5.1 P:

If the running time is some polynomial function of the size of input, for instance if the algorithm runs in linear time or quadratic time or cubic time, then we say that algorithm runs in polynomial time and the problem solves in class P.

5.2 NP:

There are a lot of programs that don't run in polynomial time on regular computer, but do run in polynomial time on nondeterministic Turing machine. These programs solve the problems in NP, which stands for nondeterministic polynomial time. NP problems are solvable in nondeterministic polynomial time, but verifiable in deterministic polynomial time.

5.3 NP-complete:

NP is a complexity class which represents the set of all problems X for which it is possible to reduce any other NP problem Y to X in polynomial time. Intuitively this means that we can solve Y quickly if we know how to solve X quickly. Precisely, Y is reducible to X, if there is a polynomial time algorithm f to transform instances y of Y to instances x = f(y) of Xin polynomial time, with the property that the answer to y is yes, if and only if the answer to f(y) is yes.
NP-complete problems are atleast as hard as all the other NP problems.
Levin-Cook's theorem gives that CNF-satisfiability problem is NP complete.

5.4 NP-hard:

These problems are the problems that are even harder than the NP-complete problems. NP-hard problems do not have to be in NP, and they do not have to be decision problems. The precise definition here is that a problem X is NP-hard, if there is an NP-complete problem Y, such that Y is reducible to X in polynomial time. But since any NP-complete problem can be reduced to any other NP-complete problem in polynomial time, all NP-complete problems can be reduced to any NP-hard problem in polynomial time. Then, if there is a solution to one NP-hard problem in polynomial time, there is a solution to all NP problems in polynomial time.

6 MATHEMATICAL MODEL

6.1 Tone analysis

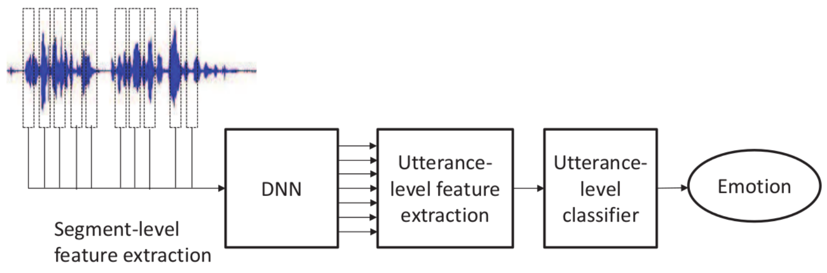
DNN
Utterance-level feature extraction
Utterance-level classifier
Emotion
Segment-level feature extraction

Figure 1: Tone analysis : Mathematical Model - Deep Neural Network(DNN) + Extreme Learning Machine(ELM)

- Input to DNN
  - Frames composed into segments according a sliding window, certain top percentage of segments qualify as input with respect to their energy.
  - MFCC featues, pitch based features are extracted per frame to give feature vector per frame z(m)
  - 2m+1 frames are stacked to generate a segment level feature vector x(m)

$$x(m) = [z(m-w), .., z(m), .., z(m+w)] \tag{1}$$

- Output of DNN
  - A sequence of probability distribution t over all emotion states for each segment

$$t = [P(E_1), ..., P(E_K)]^T \tag{2}$$

- Input to ELM (Utterance level feature extraction)
  Statistical features per each probability distribution.
  f_1, f_2, f_3 which correspond to maximal, minimal and mean of segment-level probabilityof k_{th} level emotion over utterance. f_4 is percentage of the segments which have high probability of emotion k.
- Output of ELM(Utterance level classifier)
  K-dimensional vector corresponding to scores of each emotion state. (k emotions considered).
- Objective function for DNN
  Gradient descent - mini-batch
- Objective function for DNN
  Cross entropy
- ELM is trained with least squared error.

6.2 Speech analysis

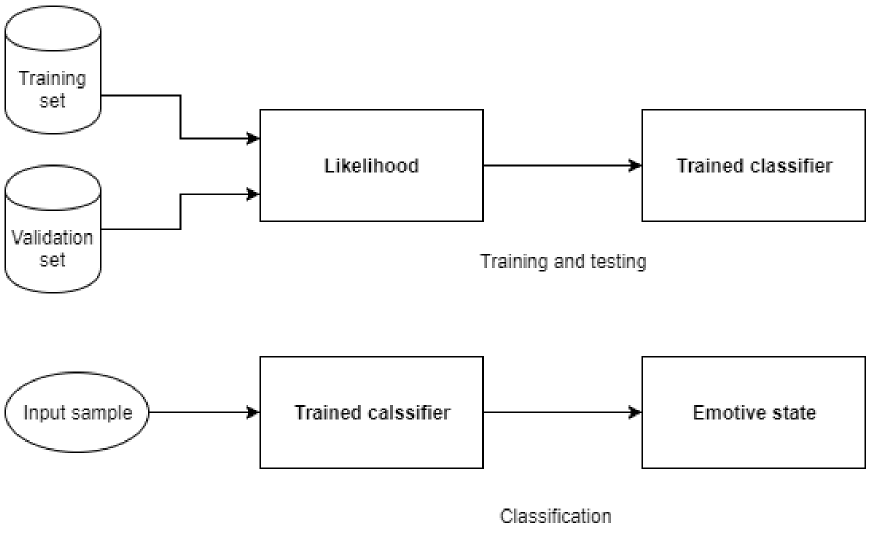

Figure 2: Speech analysis : Mathematical Model - Multinomial Naive Bayesian Classifier

- Training stage

- Input
  Dataset split into training set and testing(validation) set with k-fold cross-validation for assessing accuracy.
- Output
  Trained classifier with Likelihood
- Likelihood(evidence) calculation
  Conditional probabilities of attributes for class labels are calculated from the dataset, termed as evidence Z.

$$Z = p(x) = \sum_k p(C_k) p(x|C_k) \tag{3}$$

- Classification stage
  This stage uses the likelihood or evidence calculated in training to classify a novel input.

$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c) \tag{4}$$

In text classification, the goal is to find the best class for the input text. The best class in Multinomial NB is the most likely or maximum posteriori (MAP) class c_{map}

$$c_{map} = argmax_{c \in C} P(c|d) = argmax_{c \in C} P(c) \prod_{1 \leq k \leq n_d} P(tk|c) \tag{5}$$

In the above equation, many conditional probabilites are multiplied, and with a large enough vocabulary, raw multiplication will almost definitely result in an underflow. It is therefore better to perform the computation by adding logarithms of probabilites instead of multiplying probabilites. The class with highest log probability score is still the most probable; log(xy) = log(x) + log(y) and the logarithmic function is monotonic. Hence, the maximization that is actually done in our implementation of the Multinomial NB classifier is as follows:

$$c_{map} = argmax_{c \in C}[logP(c) + \sum_{1 \leq k \leq n_d} logP(t_k|c)] \tag{6}$$

6.3 Facial feature analysis

- Input
  Image (frame) from the video

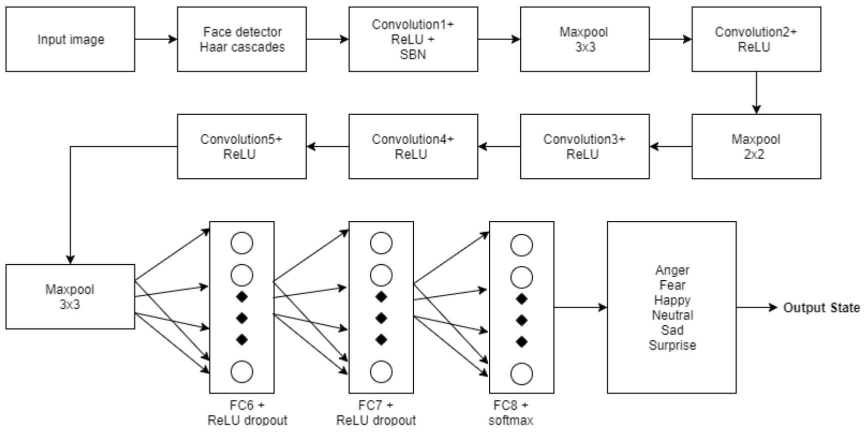

Figure 3: Facial feature analysis : Mathematical Model - Convolutional Neural Network(CNN)

- Output
  Prediction based on the output softmax layer
- Process
  - Image processing is narrowed to the regions of the image containing faces, by performing facial recognition in the image using a classifier (Haars cascades).
  - Convolutional Layer
    It computes the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and a small region they are connected to in the input volume.
  - ReLU layer
    It applies element-wise activation function which is rectifier function = *max(0,x)* thresholding at zero.
  - Pool layer
    It performs downsampling operation along spatial dimensions.
  - Fully-Connected (FC) layer
    It computes the class scores among the final categories. Each neuron in this layer is connected to all the numbers in the previous one.
  - Dropout randomly ignores nodes to prevent interdependencies emerging between nodes.

6.4 Response generation

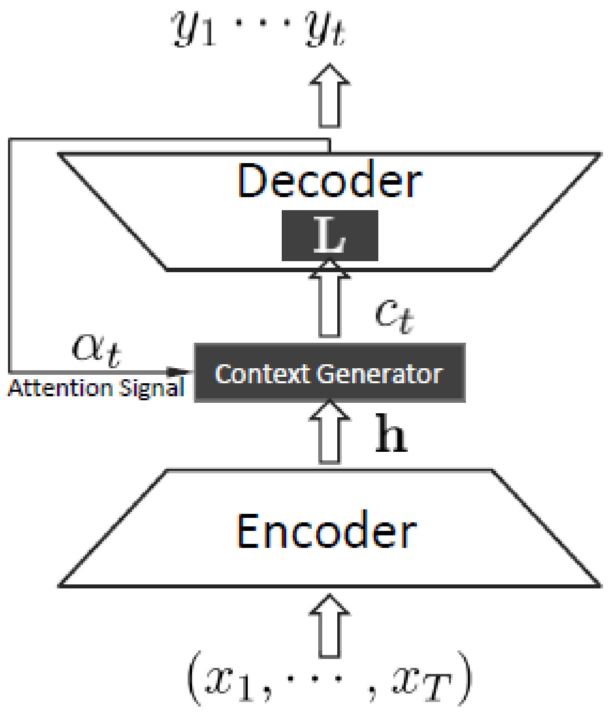

Figure 4: Response generation : Mathematical Model - Neural Responding Machine(NRM)

- Basic idea of NRM is to build a hidden representation of a statement, then generate a response based on it.
- Encoder converts input sequence $(x_1,\ldots,x_T)$ into a set of high-dimensional hidden representations h = $(h_1,\ldots,h_T)$
- h and attention signal (previous decoded response) α_t are fed into context generator to build context input to decoder c_t.
- c_t is linearly transformed by matrix L (as a part of the decoder) into a stimulus of generating RNN to produce the t-th word of a response y_t.

7 ALGORITHM TYPE

- Backpropogation is a statistical method for function approximation using multilayer neural networks. Back propogation is performed with gradient descent function.
- Backpropogation algorithm has the disadvantage that it becomes very slow in flat regions of error function. In that case the algorithm should use a larger iteration step. However, this is precluded by the length of the gradient, which is too small in these problematic regions. Gradient descent can be slowed arbitrarily in these cases.
- It is proved that finding the appropriate weights for a learning problem consisting of just one input-output pair is computationally hard, by mapping it to an 3SAT NP complete problem. (Hilbert's problem and Kolmogorov's theorem)
- Thus, backpropogation algorithm is NP complete, no polynomial time algorithm is known.

8 FEASIBILITY STUDY

8.1 Technical Feasibility

- Required hardware resources for training classifiers are available.
- Licensed softwares and libraries are available.

8.2 Economic Feasibility

- Datasets and libraries used are provided under licenses which allow free use for non-commercial purposes.

8.3 Time Feasibility

- The training of deep neural networks largely constitues the time requirements for the projects involving deep learning.

8.4 Privacy Feasibility

- Datasets and libraries are provided with licenses which allow use for non-commercial purposes.

- GNU GPL license, MIT license and Apache license allow use of datasets and libraries for non-commercial purposes. GNU GPL allows free use and distribution of software under its license, as long as it's derivatives follow the same licensing model.