I. DATASET

In this work, we use the IEMOCAP released in 2008 by researchers at the University of Southern California (USC). It contains five recorded sessions of conversations from ten speakers and amounts to nearly 12 hours of audio-visual information along with transcriptions. It is annotated with eight categorical emotion labels, namely, anger, happiness, sadness, neutral, surprise, fear, frustration and excitement. It also contains dimensional labels such as values of the activation and valence from 1 to 5; however, they are not used in this work.

The dataset contains dialogue files and is splitted according to different sessions. We are extracting the utterances and labels (for all the sessions) storing them into one csv file. Later using this data for further preprocessing.

II. METHODOLOGY

A. Data Pre-processing

A preliminary frequency analysis revealed that the dataset is not balanced. The emotions “fear” and “surprise” were under-represented and use upsampling techniques to alleviate the issue. We then merged examples from “happy” and “excited” classes as “happy” was under-represented and the two emotions closely resemble each other. In addition to that, we discard examples classified as “others”; they corresponded to examples that were labeled ambiguous even for a human.

B. Deep Learning Models

TextCNN

TextCNN, the convolutional neural network for text, is a useful deep learning algorithm for sentence classification tasks such as sentiment analysis and question classification. However, neural networks have long been known as black boxes because interpreting them is a challenging task. Researchers have developed several tools to understand a CNN for image classification by deep visualization, but research about deep TextCNNs is still insufficient. In this paper, we are trying to understand what a TextCNN learns on two classical NLP datasets. Our work focuses on functions of different convolutional kernels and correlations between convolutional kernels.

III. Implementation

- We use pandas, a Python library, to process the various transcriptions and extract features from them.
- We use pickle modules to generate the pickle files to be later used as training and test sets.
- We use torch, tensor and ignite for nn model creation, training, getting metrics etc.
- we have use GLOVE (`A module to download and use pretrained GloVe embedings.`) and vocab (To generate the vocabulary)

We randomly split our dataset into a train (80%) and test (20%) set.

Dataset format: [(labels, utterance)]

IV. Model architecture