SMAI - Project Mid-Report

Character Level CNNs for Text Classification

Members

Karthik Viswanathan

Abhishekh Sivakumar

Rohan Asokan

Satyakameshwara Rao

Objective or Aim of the Project

The aim of this project is to compare performance of LSTM and the use of character-level convolutional networks (ConvNets) for text classification and Comparisons are offered against traditional models such as bag of words, and LSTMs.

Baselines

Bag of Words

Bag of words is a method of simple feature formation for textual data. Text is something that cannot be easily parsed by machine learning models. To combat this bag of words was developed as a statistical formation for numerical vectors. Initially, a corpus based on the tokenization is formed. Post which a simple one hot vector can be assigned for each token or sentence accordingly. A simple extension to this would be addition of further statistical properties, by including document frequency of a term and global frequency of a term, also known as TF-IDF. Further additions like BM-25 also exist for vectorization of textual data. These vectors are then passed through simple feed forward networks for further processing and classification.

Word2Vec

Developed by Tomas Mikolov, It is a classical method that creates word embeddings in the field of Natural Language Processing (NLP). Word2vec takes in words from a large corpus of texts as input and learns to give out their vector representation.

As like CNNs do, the word2vec algorithm extracts features from the text for particular words. Using those features, word2vec creates vectors that represent a word in the vector space. These vectors are chosen using the cosine similarity function, which indicates the semantic similarity between words.

LSTM

LSTM stands for Long-Short Term Memory. LSTMs perform better comparing to traditional RNNs. Like NN, LSTM can have multiple hidden layers and as it passes through every layer, the relevant information is kept and all the irrelevant information gets discarded in every single cell.

We have implemented a standard single layer LSTM network.

Character-level Convolutional Networks

Important concepts from the paper are listed below. All other techniques used for training are quite common in machine learning research. The usage of temporal convolutions and character quantization makes this paper novel.

Temporal Convolutions

The temporal convolutional module simply computes a 1-D convolution. This is a prime part of the CNN paper.

The convolution $h(y) \in [1, b(l-k+1)/dc] \to R$ between $f(x)$ and $g(x)$ with stride d is defined as:

$$h(y) = \sum_{x=1}^{k} f(x) \cdot g(y \cdot d - x + c)$$

The key module to train deeper models is temporal max-pooling:

$$h(y) = (sigma(x=1,k))f(x) \cdot g(y \cdot d - x + c)$$

Given a discrete input function $g(x) \in [1, l] \to R$ $g(x) \in [1, l] \to R$, the max-pooling function

$h(y) \in [1, \frac{b(l-k+1)}{dc}] \to R$ of g(x) is defined as

$$h(y) = (max(x=1,k))g(y \cdot d - x + c)$$

Character Quantization

The developed models accept a sequence of encoded characters as input.

It is done by prescribing an alphabet of size m for the input language, and then quantize each character using 1-hot encoding. Then, the sequence of characters is transformed to a sequence of such m sized vectors with fixed length L0. Any character exceeding length L0 is ignored, and any characters that are not in the alphabet including blank characters are quantized as all-zero vectors.

The character quantization order is backward so that the latest reading on characters is always placed near the begin of the output, making it easy for fully connected layers to associate weights with the latest reading.

About the Dataset

To implement this, we have used the Yelp reviews dataset that is obtained from the Yelp Dataset Challenge in 2015. This dataset contains 1,569,264 samples that have review texts. Two classification tasks are constructed from this dataset – one predicting full number of stars the user has given, and the other predicting a polarity label by considering stars 1 and 2 negative, and 3 and 4 positive. The full dataset has 130,000 training samples and 10,000 testing samples in each star, and the polarity dataset has 280,000 training samples and 19,000 test samples in each polarity.

We have primarily used the polarity dataset, which is a two-class classification problem. Either a positive review or a negative review is the target parameter.

Yelp Review Polarity | Kaggle

Progress

1. Had gone through the following paper "Character-level Convolutional Networks for Text
   Classification" (below is the link)

   ```
   https://proceedings.neurips.cc/paper/2015/file/250cf8b51c773f3f8dc8b4be867a9a02-Paper.pdf
   ```

2. Internally Discussed and took suggestions from TA

3. Finalized the approach

4. Collected the Yelp Reviews Dataset

5. Implemented the char CNN module using pytorch
6. Implemented bag of words and LSTM for benchmarking using pytorch.

Possible Additions from now

1. Testing and optimization
2. Acquiring baseline values from the implemented models.
3. Creating a real-time inference system

Code

The code for implementations of the several models can be found here: abhishekh2001/SMAI-Project: Implementation of Character-level Convolutional Networks for Text Classification (github.com)

The dataset used can be found here: Yelp Review Polarity | Kaggle