AIM: To find suitable models and algorithms for analyzing unstructured data, ex: consumer complaints

An LSTM based Tool for Consumer Complaint Classification (Thomas N T)

Thousands of queries and complaints are posted on digital platforms on a daily basis regarding different issues. It's a tedious process for a human to go through a database of such queries and manually tag each and every query to its correct class or category. The consumer service representative who is handling that department can start working on the issue only after the tagging process. The manual intervention in this scenario is very high. A Long Short Term Memory (LSTM) based tool is proposed to automatically tag consumer queries to its corresponding class. Also, it identifies outliers or queries which are not related to the company issues or matters. The minimal manual intervention will reduce the response time.

The dataset used for this work is US Consumer Finance Complaints. It is about issues people experienced in the marketplace. The 'issue' and 'sub_issue' columns in the data shows the problems faced by consumers. The product column shows products like mortgages, student loans, payday loans, debt collection, credit reports, and other financial products and services. Each record or sequence is a combination of 'issue' and 'sub_issue' column. The **product column corresponding to each record is taken as the class label**.

STEP 1: Dataset preprocessing

- Tokenization and stopword removal are common pre-processing steps.
- Next, the dataset must be filtered. The queries not related to company services/products (outliers) must be removed. For this purpose, an outer router is used based on one class SVM. The 'issue' and 'sub_issue' columns in the data are considered as good data. The good data is trained with **One-class SVM**.
- The filtered data is sent to the inner router where actual classification happens.

STEP 2: Data Classification

- The LSTM sequence classification is used here.
- Steps followed are
  - **Sequence creation:** create a sequence of words from the tokens of the preprocessed data. Make sure that all the sequences are of the same defined length. If less, then padding is done, if more than discard.
  - **Label encoding:** encode the class labels. (in this case company products)

- ➢ **Word embedding:** vectorize the words using word2vec. **The representation of words is given as an input to the embedding layer**.
- ➢ **LSTM network:** Input the number of memory neurons and the activation function. LSTM network learns what to select and forget from the features. The model is then complied by defining the optimization algorithm and loss function. **Adam optimizer algorithm** is used in the network. It is then fitted to the model. The model is evaluated with the validation data. The dropout layers will assign zeros to a percentage of data for each epoch. The loss functions are used by the optimization algorithm in every epoch to update the weights in every epoch.

Attribution-based Explanations of TextCNN (Wenting Xiong et al.)

Attribution-based explanations are the explanations that quantify the impact (or attribution) of each feature on the prediction obtained. For instance, on a lending model, the explanation may point out that a "reject" prediction was due to income being low and the number of past delinquencies being high.

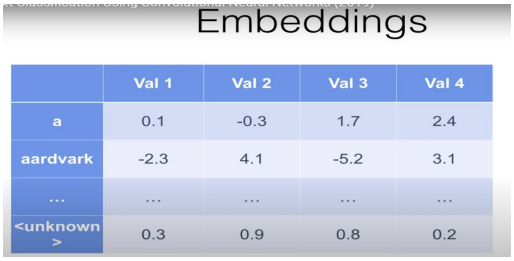
Layer-wise Relevance Propagation (LRP) and saliency maps are two such attribution based explanation techniques based on **backpropagation methods** that have been recently used to explain the predictions of Deep Learning models, specifically in the domain of text classification where these can be used to highlight/obtain relevant words for a predicted class label (i.e. finding suitable context words for a focus word) both for binary and multiclass classifications.

In this paper, a **feature-based** evaluation framework (instead of the already present word removal perturbation which not only eliminates the contribution of the particular words, but could also affect the contribution of other words within the same context window (n-grams), sentence, or document) has been proposed to compare the two attribution techniques on customer reviews (public dataset) and Customer Due Diligence (CDD) extracted reports (corporate data set). **Perturbations based on embedded features removal from intermediate layers of Convolutional Neural Networks has been done**.

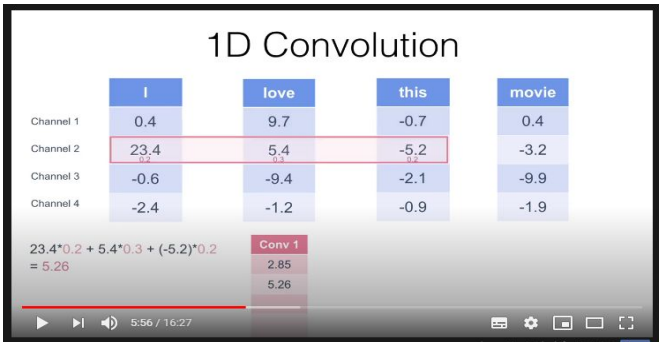
STEP 1: TEXT CNNs

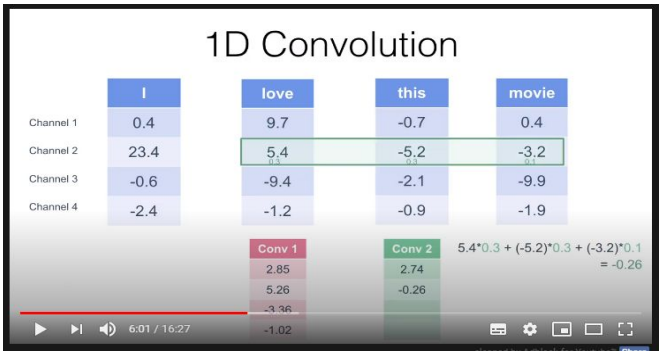
- ❖ Using different text classification techniques like BOW or TF-IDF loses the order in which the texts occur
- ❖ Using LSTM with one-hot encoding can lead to massive and raw data space and cause the networks a lot to learn

- We need something in between and that's where word encodings/ word embeddings come in



- **GloVe (Global vectors for word representation) + Word2vec** can be used for such embeddings. Using these encoding techniques a certain level of semantics encoding can also be achieved.
- The filters used in the convolution layer are the 3-gram,4-gram,5-gram 1D kernels (matrices)





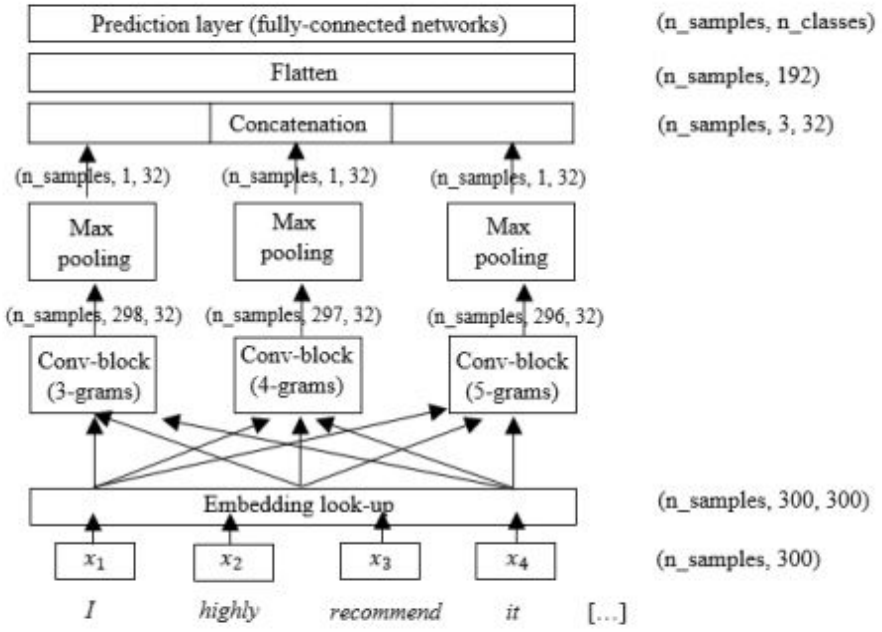


Figure 1: Architecture of the TextCNN

- Word padding can be done to make the data space of the same length (in the above case of length 300). Passing these sentences to word2vec or GloVe will convert them into embeddings.
- Now we can use the CNN model for classification.

STEP 2: Assigning Attribution scores

- Attribution scores can be assigned to each feature (each dimension of the embedded words) by using either LRP or saliency map.
- We need to compare these two attribution scores schemes.

STEP 3: Evaluation Framework

- Given a three-dimensional output of the embedding layer (n-samples (i), n-sequence of words (j), the dimension of word embedding (k)), the attribution score is assigned for each k dimension of this matrix.
- To create a document/sentence representation (document embedding), the attribution score of each word is used as a weighting factor.
- The attribution score of a word is simply the addition of the attribution scores of all the k dimensions (features)
- The non-weighted document representation for document-i is the average of representation of words in that document. If the word 'j' is represented by $e(w_j) = v(0), v(1), \cdots, v(k)$ then non weighted document representation is simply:

$$\frac{1}{j}\sum_{j} e(w_j)$$

- Whereas the weighted document representation is given as, where Rj is the sum of the attribution scores of all its k dimensions and used as the weight value for the word ‘j’

$$\frac{1}{j}\sum_{j} R_j e(w_j)$$

- Now for evaluation purposes ‘embedded document perturbations’ are applied. Intuitively, different fragments of a document (e.g. between sentences) may tell different sentiment polarity weights. A review could be started by mentioning a negative criticism about a small aspect of a product, but the final conclusion may give a positive recommendation.
- Assuming these different aspects of polarities are embedded as features of the learned document embedding, we utilize “feature” or each dimension of the embedded document to evaluate the importance of scores assigned by attribution methods in the corresponding prediction task.
- **The feature removal was done by setting all values in the corresponding columns to be 0**. 3 different settings were employed and the difference in the accuracy scores was analyzed.

    - Removing features with the largest attribution scores. (accuracy expected to be lower)
    - Removing features with the smallest attribution scores (accuracy shouldn’t be affected much which checks that the features removed were truly unimportant)
    - Removing features that contribute differently for different classes (suppose the true class is c while the predicted class is c’, then the attribution score differences are taken between the class c and c’ for all k columns. When the columns with the largest attribution differences are removed, the predicted probability for class c should decrease while the probability for class c’ should increase. This setting was only applied to classification tasks with multiple classes. (ex: US customer financial complaints)

- An evaluation of the document representations to investigate whether words relevant to the output are identified or not is done.
- The US customer financial complaints database was used for evaluation and different CNN techniques were employed like KNN, SVM, Decision tree, Random Forest, etc. from the results obtained it was shown that LRP gives better results.

- Finally using PCA the documents which are correctly classified can be clustered in different groups where it was shown that using weighted documents is beneficial as clear clusters were seen there.