Character-level Convolutional Networks for Text Classification

Github Link: https://github.com/TanishqGoel/Text-Classifer-CNN

Report by team Dh_Dulla

Members -

Vaibhav Kashera - 2019111003
Tanishq Goel - 2019114015
Pranav Kannan - 2019111033
Kshitij Mishra - 2019111014

Paper Referred :

Zhang, X., Zhao, J. and LeCun, Y., 2015. Character-level convolutional networks for text classification. Advances in neural information processing systems, 28, pp.649-657.

Abstract

This work looks into the use of convolutional networks on character level instead of traditional methods for text classification. Large datasets are used to show that results are as competitive as other methods. Comparison is made with models such as a bag of words, n-grams, and deep learning models such as word-based ConvNets and RNNs.

Introduction

Text categorization is a well-known issue in natural language processing, in which preset categories are assigned to free-text texts. Text classification research covers everything from developing the best features to selecting the best machine learning classifiers. Almost all text categorization algorithms to date are based on words, with basic statistics of some ordered word combinations doing the best.

Convolutional networks, on the other hand, have been shown to be beneficial in extracting information from raw signals by various researchers, spanning from computer vision applications to voice recognition and others. Time-delay networks, for example, are basically convolutional networks that handle sequential input and were popular in the early days of deep learning research.

This work treats text as a raw signal at the character level, and then we apply temporal convolutional networks. We have only used a classification task in order to exemplify ConvNets' ability to comprehend text. We then use datasets to compare with traditional models.

Convolutional networks to natural language processing at large can be directly applied to distributed or discrete words without any emphasis on the syntactic or semantic structures of a language. These approaches, as compared to traditional models, have been proven to be competitive.

We have applied ConvNets only on characters. This ease could be useful for a single system that can work for different languages, for example, Latin languages like English, French, Espanol, etc. Using this, we can also address the issues of misspelling, emphasis words (like hmm and hmmm) and emoticons like :) and ;)

Datasets

We have used a large-scale dataset because the model takes smaller entities like characters. Statistics of our large-scale datasets where epoch size is the number of mini-batches in one epoch

Dataset	Classes	Train Samples	Test Samples	Epoch Size
AG’s News	4	1,20,000	7,600	935
DBPedia Ontology	14	5,60,000	70,000	4300

AG’s news corpus contains 496,835 categorized news articles from more than 2000 news sources. The four largest classes from this corpus to construct our dataset have been chosen, using only the title and description fields. The number of training samples for each class is 30,000, and testing 1900.

DBPedia ontology dataset is a crowd-sourced community effort to extract structured information from Wikipedia. The DBpedia ontology dataset is constructed by picking 14 unique classes from DBpedia 2014. The training and testing samples are 56000 and 70000, respectively, spread out equally across 14 classes.

Methodology

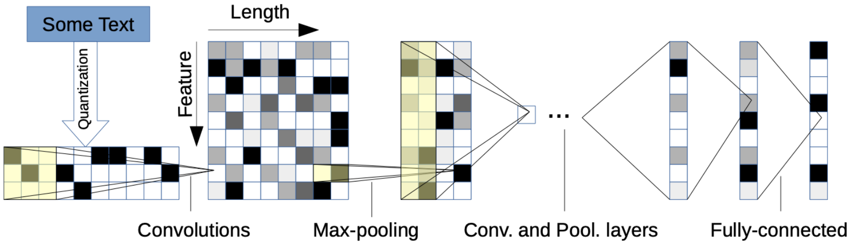
In this section, modular design with gradients obtained by back-propagation to perform optimization has been introduced for the design of character-level ConvNets for text classification.

Preprocessing We used standard 70 characters, including 26 English letters, ten digits, 33 other characters and the new line character. The non-space characters are:

abcdefghijklmnopqrstuvwxyz0123456789
-,;.!?:’ ’ ’/|_@#$%^&*~‘+-=<>()[]{}

The sequence of characters is transformed to a sequence of m sized vectors with fixed length ’l’. Any character exceeding length ’l’ is ignored, and any characters that are not in the alphabet, including blank characters, are quantized as all-zero vectors.

Design 2 ConvNets have been designed with nine layers, including six convolutional layers and three fully-connected layers.



Convolutional layers used in our experiments-

Layer	Large Feature	Small Feature	Kernel	Pool
1	1024	256	7	3
2	1024	256	7	3
3	1024	256	3	N/A
4	1024	256	3	N/A
5	1024	256	3	N/A
6	1024	256	3	3

The mean and standard deviation used for initializing the large model is (0, 0.02) and the small model (0, 0.05). Fully-connected layers used in our experiments-

Layer	Output Units Large	Output Units Small
7	2048	1024
8	2048	1024
9	Depends On the problem	Depends On the problem

For different problems, the input lengths may be different (for example, in our case, $l_0 = 1014$), and so are the frame lengths.

Observations

We trained our CNN model on two different datasets for 6-8 epochs. The results we got had high variance and were highly dependent on the dataset. Results of the model are shown in the following slides in terms of Accuracy/ Precision/ Recall & F1 score.

DBPedia

Type	Precision	Recall	F1 score
Company	0.922482	0.955929	0.938908
EducationalInstitution	0.979555	0.969157	0.974328
Artist	0.95495	0.968149	0.961504
Athlete	0.996745	0.982161	0.989399
OfficeHolder	0.978353	0.969138	0.973724
MeanOfTransportation	0.965639	0.985769	0.9756
Building	0.969841	0.939752	0.954559
NaturalPlace	0.978118	0.984976	0.981535
Village	0.99281	0.995394	0.994101
Animal	0.98259	0.983771	0.98318
Plant	0.982393	0.983574	0.982983
Album	0.983984	0.984773	0.984378
Film	0.983593	0.984776	0.984184
WrittenWork	0.983262	0.964364	0.973721

AG NEWS

Type	World	Sports	Business	Sci/Tech
Precision	0.810827	0.888694	0.674049	0.678486
Recall	0.722694	0.801373	0.742312	0.750397
F1_score	0.764228	0.842778	0.706535	0.712632

Comparison with Other Models

After implementing the required code, we decided to implement the same task using different models to analyze our results by comparison. We implemented this task using the following simple transformers: Multilingual cased BERT (mBERT) which was pre-trained on 104 languages. ALBERT, which was pre-trained on the English language using a masked language modelling (MLM) objective; Roberta base, which is a model pre-trained on a large corpus of English data in a self-supervised fashion.
The results are as follows:

Model	F1 Score	Accuracy	Precision
CNN	0.93	0.93	0.9
BERT-Cased	0.9	0.91	0.9
Bert-Uncased	0.87	0.9	0.84
XLnet	0.84	0.83	0.82
ALBERT	0.83	0.84	0.81

Model	F1 Score	Accuracy	Precision
RoBerta	0.83	0.81	0.81

Conceptual Discussion and Analysis

Why does CNN excel?

- **Effective method-** the most interesting conclusion of our work is that character level ConvNets work for text classification without the need for words. This indicates that language can be interpreted as a signal of characters.
- **Effective on user-generated data-** ConvNets work well on user-generated data. The degree of word curation varies in human-generated data significantly. Our results show that ConvNets works better in real-world scenarios, like identifying exotic character combinations, misspellings, emoticons and exclamatory words (like hmm and oh!).
- **No significance of semantics-** semantics, emotions, and sentiments don't contribute as a factor in deciding which method works better.
- **No homogeneity-** results verify that there is no single machine learning model that can produce the same results for all varieties of datasets. A lot of factors play important roles in deciding which method is best for which kind of application.

Conclusion and Outlook

The research paper given uses ConvNets at the character level for text classification. The implementation of the algorithm and the approach used in this paper was very interesting and challenging. As the authors have mentioned, this research can be used to apply character-level ConvNets for a broader range of language processing tasks as future work.

Acknowledgement

We gratefully acknowledge the support of Prof. Namboodiri and TA Aditya Jain in Statistical Methods in Artificial Intelligence course on Monsoon '21 semester.

References

1. Zhang, X., Zhao, J. and LeCun, Y., 2015. Character-level convolutional networks for text classification. Advances in neural information processing systems, 28, pp.649-657.