Team 10 :
Character Level Models for Text Classification

Code

Gayatri Madduri, 2019102043

Kawshik Manikantan Sundar, 2019111004

Sai Akarsh C, 2019111017

Srividya Srigiri, 2019102041

Abstract

This report covers all our studies regarding the classification of text using character level models and also how they compare with other baseline models. This includes description about the methodologies tried, end-to-end system, results obtained on evaluation and analysis.

1. Introduction

Text classification is a common problem in natural language processing. It involves classifying a free-text document into a predefined set of categories. The performance of text classification is improved by designing the best features and choosing the best classifiers for those particular features. Since it is text that we are classifying, the most common unit used for feature extraction is words, where we design features using some ordered word combinations to model the meaning in the sentence.

We have found out through recent advancements that we can still produce a very accurate model without modifying the input into a more reformed manner, assuming that the network used is complex enough to capture information from that raw data. Rather than converting words into n-grams to capture relational information, we directly use the characters for the model.

Our work involves exploring various character level classification methodologies and building a real-time application based on the best of the lot. After the implementation of multiple word and character-level baselines, we have used CANINE [1].CANINE is a Transformer-based language model which does not need any explicit tokenization step. In CANINE, we directly convert our characters into its ASCII equivalent and use that as input rather than more traditional methods of word embedding or sentence embedding which capture the relation between the words and meaning of the sentences. The model is a combination of down-sampling which reduces the length of the input sequence and a deep transformer stack that encodes the context of the text. We have shown that CANINE has a better performance over other baseline models.

We implemented multiple baselines such as Bag of Words,Bag of Words with TF-IDF, vanilla LSTM to compare with the selected model. All these models, convert the words or sentences into a vector using some embedding to capture the meaning which will lead to better classification.

There are other works which deal with characters rather than words and sentences. The Character Level Convolution Network [4] baseline used is one such approach that uses a CNN network to classify text on the hypothesis that when trained on large scale datasets, we do not require explicit input of words or the meaning of the sentences.

Without having any prior knowledge of the semantic or syntactic structure of a language, we show that it is possible to capture the information in the sentence and classify it. This makes it easier to adapt for different languages and also effectively removes the hurdle of spelling errors and emoticons present in word based models.

2. Dataset

We worked with three datasets:

- Yelp Review Full which contains reviews from Yelp. There are a total of 5 classes which represent the num-

ber of review stars.There are 650,000 training samples and 50,000 testing samples.

- dbpedia_14 is constructed from 14 ontology classes in DBPEDIA 2014. The training dataset consists of 560000 data points and the test set consists of 70000 data points.

- Amazon polarity contains reviews on Amazon. This dataset is classified into two classes:1(positive) and 0(negative). The data spans over a period of 18 years. The data fields include title and body of the review and there are 1800000 training samples and 200000 test samples

3. Related Work

There are multiple works that deal with the task of text classification. Here we discuss works that are either character level approaches or have achieved state of the art results on multiple benchmarks:

1. In [4], a detailed model of character-level ConvNets for text classification is discussed. CNNs are deep learning networks that contains of kernels for the convolution. These kernels help identify patterns in signals. Computational complexity and network overfitting is prevented in CNNs with the help of max-pooling,a technique of selecting the maximum element from the region of the signal covered by a filter. Hence, text here is analysed as a signal.

2. [3] discusses the Attention based Transformer architecture. Attention mechanisms allows modeling of dependencies without regard to their distance in the input or output sequences. Though attention mechanism were employed in various neural network based approaches, the transformer architecture proposed a method to do away with recurrences and rely solely on attention mechanisms. Transformer models are very popular in solving various NLP tasks like language understanding, question answering and text classification and have achieved state of the art results.

3. [2] applies bidirectional training of transformer for language modelling. Transformer architecture consists of two mechanisms: encoder and decoder. Encoder read the text input and forms the basis for BERT. The key feature of the BERT model is that it takes into account the surroundings of a word on both left and right sides to derive the context of the word. It is, thus, aptly named Bidirectional Encoder representation from Transformers). BERT is a very popular model and reports good accuracies. It has been implemented, cited and analysed by several people. It reports an accuracy of 97.3% for amazon_polarity dataset, 99.3% for dbpedia dataset and 70.7% for yelp dataset.

4. [1],much similar to BERT involves pre-training an efficient tokenization-free transformer-based encoder for Language Representation, albeit at a uni-code character level. Language based systems commonly use models that require an explicit tokenization step which are not efficient which require several preprocessing steps and not suited to all languages. This paper discusses the CANINE model in detail as it is the front-runner in the class of character level models. The lack of literature due to its recent publication encouraged us to dig deeper.

4. Baseline Methodologies

This section describes few existing models for text classification. For all the below architecture implementations, after loading the data set, pre-processing was performed, i.e, data was cleaned appropriately so as to remove unwanted characters, very short words etc. It was initially shuffled, and split for training, validation and testing respectively. Different metrics like accuracy, precision, recall and f1-score were logged for analysis of the model's performance under different scenarios with respect to data sampling.

- **Bag of Words**: Bag of Words is a methodology to extract features from a given text. This method requires a vocabulary list,which may be constructed from the text data to be trained or fed externally. The length of this vocabulary list determines the number of features. Each word in the vocabulary list corresponds to a particular feature. The value of this feature is obtained from the frequency of occurrence of that word in the text data.
  After this process of feature extraction, conventional methods of classification such as neural networks, logistic regression etc. can be applied. We have implemented logistic regression in this project. The results obtained on the data sets mentioned above can be found in a subsequent section.

- **TFIDF**: TFIDF expands on the Bag of Words methodology by incorporating the occurrence of a word in other related documents. These method is based on three values: term frequency,document frequency and inverse document frequency
  Term frequency is the normalised frequency of occurrence of a word in a document. Document frequency gives us the number of documents in which a particular word occurs. Inverse document frequency is the ratio of the total number of documents in the

corpus to the document frequency. Inverse document frequency ascertains the importance of a word in a document. For example, a word like ”is” will have very low inverse document frequency for any document because it is not very relevant in understanding the text.
Thus, the TFIDF value of a feature corresponding to a word is obtained by multiplying the values of term frequency and inverse-document frequency of that word. The total number of features equals the number of words in the entire document corpus.

- **LSTM**: The traditional neural network models work with a fixed length input and a fixed length output. But when dealing with text sentences, the word count is not fixed. Hence we need to use a model that can handle variable length input. Given that in a sentence a word is dependant on the words around it, we can say that those words are related. To handle variable lengths and temporal dependencies, we use LSTM to take in the sentence and give a meaningful vector that captures its meaning.
Firstly, we use a pre-trained word embedder like glove and convert each sentence into a list of vectors. The output is , then, passed through a classifier. This classifier is a simple linear unit which maps the LSTM output to the class labels and then applies softmax to the output. This represents the probability for the sentence belonging to each one the classes. We use a huge corpus to train this model to obtain satisfactory results. LSTM is good at encapsulating the long term and short term dependencies, which makes it a good classifier.

- **CNN**:CNN is a neural network model which was introduced to extract higher representations for the image content with the help of convolution and max-pooling. CNNs can learn patterns and features of the image that is helpful for the task. For our purpose, we follow [4], where a character sequence is created and with the help of 1-D convolutions and 1-D max-pooling patterns that help in classification are learnt. The model accepts a sequence of encoded characters as input and then quantizes that character sequence using one-hot encoding. The quantization order is backward which makes it easy in the case of a fully connected network. The model consists of 6 convolutional networks and 3 fully connected layers. This effort is based on the hypothesis that language can also be thought of as a signal. This method gives great results in few scenarios but lacks in others probably due to its non-temporal base.

5. Architecture

We are using a transformer model (CANINE) for text classification in the final architecture. Transformers and transfer learning have achieved SOTA results in multiple datasets. Transformers combine positional encoding and the non-sequential single processing of the input to achieve better long range dependencies than LSTMs and RNNs. Attention mechanism is targeted towards such sequential data and hence is more preferred than the classical approaches, which perform poorly for sequential data.

CANINE is a neural encoder that directly opearates on the character sequence without an explicit tokenization step or vocabulary. CANINE solves the problem of longer sequence length using an efficient downsampling strategy and a deep transformer stack is used to encode the context of the sentence. CANINE model is a composition of a downsampling function, a primary encoder and an upsampling function. The input to the model is a sequence of unicode code points, which are embedded using multiple hash functions. Embedded characters are passed into a single-layer block-wise local attention transformer which encodes the characters. Strided convolution is used to reduce the number of sequence positions. In the next stage, the context of the text is encoded using a deep transformer stack.

The dataset is cleaned by a very basic procedure that removes strays and urls and lower cases the text. The cleaned dataset is then passed to the CANINETokeniser, which converts the sequence of characters into characters and then returns their code-point integer values. The length of each input is fixed to be 512 tokens. This involves adding special tokens like [CLS], [SEP] and [PAD] at the beginning of every sentence, at the ending and for padding sentences less than 2048 in length.

Only the first embedding (corresponding to the [CLS] token) produced by the CANINE on input is used for classification. This encodes all the information required for the classification of the input content. This embedding is then passed to a linear layer that produces logits that are used for predicting the labels of the input posts. In this model, AdamW optimiser is used with an initial learning rate of 2e-5 and a linear warm-up schedule. Backward propagation of the loss is done and the weights of the linear layer and the model are updated. This process of fine-tuning helps the model to learn the specific domain problem. The model begins to overfit after 4 epochs and the training is stopped here.

We have also developed a simple application, which has a text box for taking a piece of text as input. In the

background, the model is loaded once at the beginning of executing the application, and whenever user enters text on the User-Interface and clicks on the submit button, instant prediction of the rating pertaining to the text is displayed on the screen (the prediction takes place with the help of the saved transformer model, in the background). The application was developed with the help of Streamlit library.

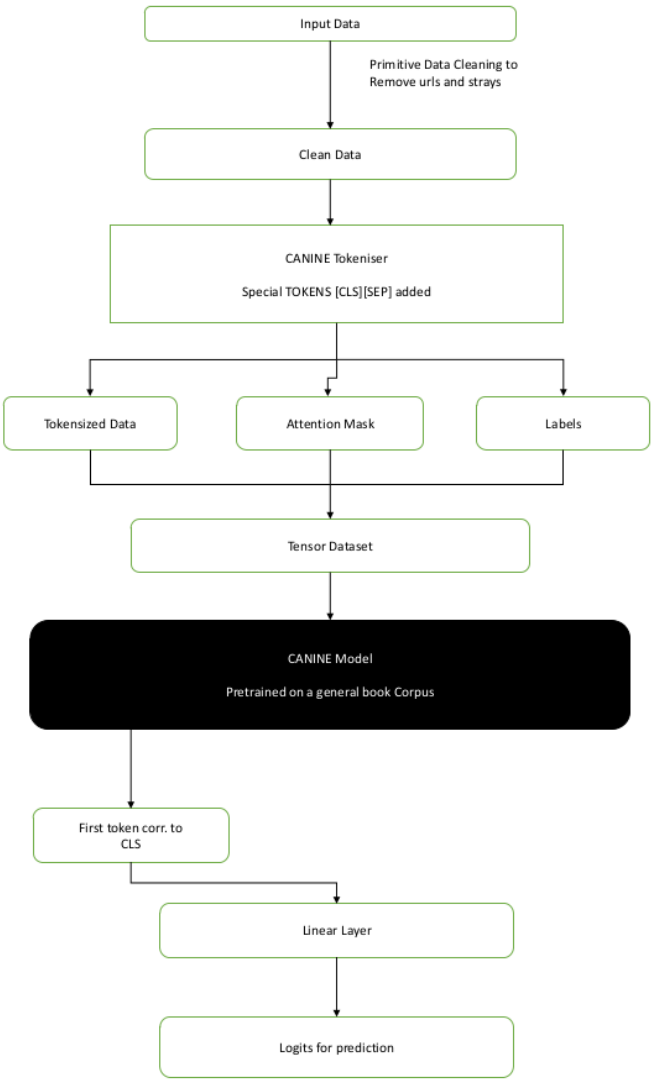

Figure 1. Architecture of the model

6. Evaluation Mechanism and Results

Accuracy, precision, recall and F1-score are used as metrics for evaluating performances of various baseline models, the cnn model presented in the paper and the canine method.

Table 1. DBPEDIA

	Accuracy	Precision	Recall	F1-score
Bag of Words	0.90	0.91	0.90	0.90
BoW with TFIDF	0.82	0.82	0.82	0.82
LSTM	0.93	0.94	0.93	0.93
CNN	0.98	0.98	0.98	0.98
Canine	0.99	0.99	0.99	0.99

Table 2. Yelp full review

	Accuracy	Precision	Recall	F1-score
Bag of Words	0.47	0.47	0.47	0.45
BoW with TFIDF	0.51	0.50	0.51	0.50
LSTM	0.45	0.45	0.45	0.44
CNN	0.60	0.60	0.60	0.60
Canine	0.63	0.64	0.63	0.63

Table 3. Amazon polarity

	Accuracy	Precision	Recall	F1-score
Bag of Words	0.81	0.81	0.81	0.81
BoW with TFIDF	0.81	0.81	0.81	0.81
LSTM	0.80	0.80	0.80	0.80
CNN	0.912	0.912	0.912	0.912
Canine	0.934	0.93	0.93	0.93

7. Analysis

The results obtained for each model using various data sets tell us that some models work good on some data sets but not so great on others. We can also infer that our proposed CANINE based classifier consistently outperforms the other models. We will try to analyse and understand why some baseline models are performing better than others and how our model is able to beat these baseline models.

Bag of words is a method that ignores the semantics and only looks at the frequency. This works for most cases but will not where the structure of the sentence makes a huge difference. Hence we notice the least accuracy for this model.

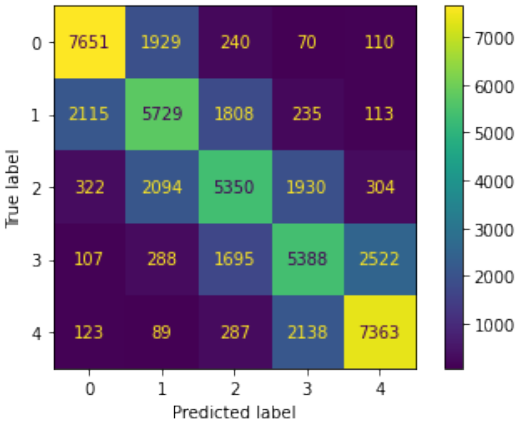

Figure 2. Confusion Matrix of CANINE on Yelp

To tackle this we use LSTM, which can model semantic data and can also fine tune the time dependency. Because of the various parameters the LSTM offers, it has high flexibility. LSTM has both a short term memory and a long term memory that model the relationships. Hence it gives a better result than Bag of Words.

The main advantage of CNN is that it can automatically detect important features from the input signal. By treating characters as a signal, this model is able to find various patterns in the text. This advantage helps to better model the semantics and patterns in the text and hence gives better results when compared to LSTM.

CANINE achieves a model that learns memorisation and composition as and when required by working at the character level and using transformer networks.Large frequency-derived vocabularies are also capable of generating such models by memorizing more, as in the bag of words model,but language inherently requires aspects of both memorization and composition.CANINE also outperforms LSTM because it processes a sentence as a whole and do not stand a risk to forget previous information.

CANINE achieves extremely good results on the DBPedia dataset and the Amazon polarity dataset. This shows its power to learn languages despite no frequency or meaning information input. The Yelp-Full-Review dataset provides good insights on the model. We can note that even though CANINE provides an accuracy of 63%, the confusion matrix shows that the mis-classifications are generally nearer to the label. Though CANINE is good at determining a close approximate to the user's rating, it still lacks the ability to accurately understand the sentiment behind the rating and the review. Also to be noted is that the problem is in itself quite complex and significantly difficult for humans as well.

Though our CANINE is a diluted version of the main model and we are in no position to compare the result on a benchmark basis, we can see that it is still slightly behind BERT in its results. This might be due to the high computational stress of characters and lesser input due to the same or due to the efforts required by CANINE to additionaly learn tokenisation. The non-tokenisation approach of CANINE, whose advantages are many, compensates the slight decrease in accuracy.

8. Challenges Faced

The different approaches used such as Bag of words, LSTM, CREPE and the proposed Canine classifier model had different challenges to deal with, when designing and getting good results

1. **LSTM**

   (a) When designing a classifier using LSTM, the goal was to break down the sentence into tokens based on the words. The challenge here was to use a suitable embedder that is able to model the relationships of the words properly.

   (b) The main challenge of having variable length input is that we need to somehow increase the length of each of these inputs to the same length for training. The standard way of doing this is by defining a STOP token and a PAD token. Rather than doing that, we used the pack sequence module which packs all these variable length inputs into an object and enables us to train. Another issue is the rate of convergence.

   (c) A lot of fine tuning was required to get a fast convergence like choosing the number of stacked LSTM required, learning rate, the initial state values, etc. But once we find the right values, the convergence is achieved quickly with good accuracy.

   (d) The maximal length of tokenisation was limited to 100 tokens due to low computational capabilities.

2. **Bag of Words**

   (a) A major challenge was the variable length input. This was tackled by obtaining the length of the longest sentence and padding all the other sentences with a 50,000 to make all sentences in the text data of same length. In the feature extraction step, 50,000 was not considered a word.

(b) In order to resolve computational limitations, run-time encoding was performed. Use of data loader for run time encoding required us to convert the text into numeric data. Automatic feature extraction would not make sense as the numbers do not represent the relationship between different words. Hence, collate function is used to convert the numeric data into words and extract features.

(c) The model is very slow. Methods tried to make it faster include accelerating on GPU and limiting feature vector size. They did not show a significant improvement, however.

3. **CNN**

(a) Though many sentences of the dataset exceeded 1014 characters, the first 1014 characters were taken to reduce the otherwise large computation that would go waste for the majorly short sentenced dataset.

(b) Initial attempts were made to create the one-hot representation of the entire dataset at once to preserve run-time. However, this attempt failed as this caused memory errors so, the dataset was encoded in the categorical form and was converted to one-hot encoding at run-time with the help of the collate function of DataLoader.

(c) The model learns and converges quite fast and hence we need to modify the learning rate carefully so as to achieve the best results. Otherwise, the model overfits or suffers from a bias.

4. **CANINE**

(a) The pre-trained CANINE model used has a maximum limit of 2-048 characters of input. However due to our computational constraints, only the 512 character input version fits the GPU.

(b) The Amazon polarity dataset being very huge could not be used completely for learning. The complex CANINE model, even with the downsampling takes considerable amount of time to train and hence only 650000 data points were considered.

9. Conclusion

We observe that CANINE outperforms Bag of Words and LSTM due to its capability to process a sentence as a whole,compose words and memorise words. It employs attention mechanisms to overcome the problem of recurrent networks, wherein older states might be forgotten.
Also,CANINE doesn't involve any sub-word tokenisation. Sub-word tokenisation is a major drawback in contexts such as informal texts,spelling errors,emojis,languages without an impoverished morphology etc. CANINE offers a huge advantage in such scenarios.
The performance of other papers varies with the morphology of the language chosen. CANINE is more generic in nature since it operates at the character level. It can be readily used in different languages.

References

[1] Jonathan H. Clark, Dan Garrette, Iulia Turc, and John Wieting. Canine: Pre-training an efficient tokenization-free encoder for language representation, 2021.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[4] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification, 2016.