

**Major Mid\_Eval Report**

***Project Title: Short Text Classification***

***Submitted By-***

Palak Arora (16103046)

Anjali Sharma(16103015)

Aman Parmar(16103221)

***Batch:***

*B-15*

**Introduction**

In micro-blogging services such as Twitter, the users may get overwhelmed by the raw data. One solution to this problem is the classification of Twitter messages (tweets). As short texts like tweets do not provide sufficient word occurrences, classification methods that use traditional approaches such as “Bag-Of-Words” have limitations. To address this problem, we propose to use a small set of domain-specific features extracted from the author’s profile and text. The proposed approach effectively classifies the text to a predefined set of generic classes such as News, Events, Opinions, Deals, and Private Messages.

Existing works on classification of short text messages integrate every message with meta-information from external information sources such as Wikipedia and WordNet. Automatic text classification and hidden topic extraction approaches perform well when there is meta-information or the context of the short text is extended with knowledge extracted using large collections. But these approaches require online querying which is time consuming and unfit for real time applications. When external features from the world knowledge is used to enhance the feature set, complex algorithms are required to carefully prune overzealous features. These approaches eliminate problem of data sparseness but create a new problem of the curse of dimensionality . Hence efficient ways are required to improve the accuracy of classification by using minimal set of features to represent the short text. Many classiﬁcation models work poorly on short texts due to data sparsity. To address this issue, we propose topic memory networks for short text classiﬁcation with a novel topic memory mechanism to encode latent topic representations indicative of class labels. Different from most prior work that focuses on extending features with external knowledge or pre-trained topics, our model jointly explores topic inference and text classiﬁcation with memory networks in an end-to-end manner. Experimental results on four benchmark datasets show that our model outperforms state-of-the-art models on short text classiﬁcation, meanwhile generates coherent topics

Short texts have become an important form for individuals to voice opinions and share information on online platforms. A large body of daily generated contents, such as tweets, web search snippets, news feeds, and forum messages, have far outpaced the reading and understanding capacity of individuals. As a consequence, there is a pressing need for automatic language understanding techniques for processing and analyzing such texts (Zhang et al., 2018). Among those techniques, text classiﬁcation is a critical and fundamental one proven to be useful in various downstream applications, such as text summarization (Hu et al., 2015), recommendation (Zhang et al., 2012), and sentiment analysis (Chen et al., 2017). Although many classiﬁcation models like support vector machines (SVMs) (Wang and Manning,2012) and neural networks have demonstrated their success in processing formal and well-edited texts, such as news articles ,their performance is inevitably compromised when directly applied to short and informal online texts. This inferior performance is attributed to the severe data sparsity nature of short texts, which results in the limited features available for classiﬁers (Phan et al., 2008). To alleviate the data sparsity problem, some approaches exploit knowledge from external resources like Wikipedia (Jin et al., 2011) and knowledge bases (Lucia and Ferrari, 2014; Wang et al., 2017a). These approaches, however, rely on a large volume of high-quality external data, which may be unavailable to some speciﬁc domains or languages (Li et al., 2016a). To illustrate the difﬁculties in classifying short texts, we take the tweet classiﬁcation in Table1 as an example. In the test instance S, only given the 11 words it contains, it is difﬁcult to understand why its label is New. Music. Live. Without richer context, classiﬁers are likely to classify S into the same category as the training instance R1, which happens to share many words with S, in spite of the different categories they belong to,1 rather than R2, which only shares the word “wristbands” with S. Under this circumstance, how might we enrich the context of these short texts? If looking at R2, we can observe that the semantic meaning of “wristbands” can be extended from its co

We propose an intuitive approach to determine the class labels and the set of features with a focus on user intentions on Twitter such as daily chatter, conversations, sharing information/URLs. We classify incoming tweets into five generic categories – news, opinions, deals, events and private messages. We believe that these categories are diverse and cover most of the topics that people usually tweet about. Experimental results using our proposed technique outperform the baseline Bag-Of-Words model in terms of accuracy and speed.

Next, we allow users to add new categories based on their interest. Since the accuracy is bound to deteriorate with increase in number of classes, we also allow users to add new features corresponding to the new classes. Our system takes in sampled tweets from the user and diagnoses the feature. Experimental results show that with our small feature set and the user-defined features, the classification accuracy is better than the Bag-Of-Words model

***Dataset used:***

A large dataset, consisting of multiple JSON objects related to NEWS from around the world along with other related features like:

* Short description
* Timestamp
* Category/ class/ emotion

, was collected from kaggle. The dataset collected had about 2lkh plus JSON instances.

The final dataset used for different classification models was extracted, cleaned and improved, using certain methods and common wit on the basis of the requirements of the task.

The dataset was extracted and converted into the below mentioned format:

* For BOW model: Instance######category
* For TMN model: a csv with instance and category as separate columns

From each of the JSON object, 3 fields were extracted:

* Instance
* Short Description
* Category

Each of the JSON object was converted into 2 instances, thus the final dataset consisted of 4lkh plus instances.

**Algorithms used:**

* Bag of words
* n-grams
* topic memory networks
* 8F model

Brief about the models used:

***Bag of words:***

A bag-of-words model, or BoW for short, is a way of extracting features from text for use in modelling, such as with machine learning algorithms.

The approach is very simple and flexible, and can be used in a myriad of ways for extracting features from documents.

A bag-of-words is a representation of text that describes the occurrence of words within a document. It involves two things:

1. A vocabulary of known words.
2. A measure of the presence of known words.

It is called a “*bag*” of words, because any information about the order or structure of words in the document is discarded. The model is only concerned with whether known words occur in the document, not where in the document.

*A very common feature extraction procedures for sentences and documents is the bag-of-words approach (BOW). In this approach, we look at the histogram of the words within the text, i.e. considering each word count as a feature.*

— Page 69, [Neural Network Methods in Natural Language Processing](http://amzn.to/2wycQKA), 2017.

The intuition is that documents are similar if they have similar content. Further, that from the content alone we can learn something about the meaning of the document.

The bag-of-words can be as simple or complex as you like. The complexity comes both in deciding how to design the vocabulary of known words (or tokens) and how to score the presence of known words.

***N-grams:***

In the fields of computational linguistics and probability, an ***n*-gram** is a contiguous sequence of *n* items from a given sample of text or speech. The items can be phonemes, syllables, letters, words or base pairs according to the application. The *n*-grams typically are collected from a text or speech corpus. When the items are words, *n*-grams may also be called *shingles*

Using Latin numerical prefixes, an *n*-gram of size 1 is referred to as a "unigram"; size 2 is a "bigram" (or, less commonly, a "digram"); size 3 is a "trigram". English cardinal numbers are sometimes used, e.g., "four-gram", "five-gram", and so on. In computational biology, a polymer or oligomer of a known size is called a k-mer instead of an *n*-gram, with specific names using Greek numerical prefixes such as "monomer", "dimer", "trimer", "tetramer", "pentamer", etc., or English cardinal numbers, "one-mer", "two-mer", "three-mer", etc.

An ***n*-gram model** is a type of probabilistic language model for predicting the next item in such a sequence in the form of a (*n* − 1)–order Markov model. *n*-gram models are now widely used in probability, communication theory, computational linguistics (for instance, statistical natural language processing), computational biology (for instance, biological sequence analysis), and data compression. Two benefits of *n*-gram models (and algorithms that use them) are simplicity and scalability – with larger *n*, a model can store more context with a well-understood space–time tradeoff, enabling small experiments to scale up efficiently.

Topic memory networks:

There are three major components: (1) a neural topic model (NTM) to induce latent topics (described in Section 2.1), (2) a topic memory mechanism that maps the inferred latent topics to classiﬁcation features (described in Section 2.2), and (3) a text classiﬁer, which produces the ﬁnal classiﬁcation labels for instances. These three components can be updated simultaneously via a joint learning process, which is introduced in Section 2.3. In particular, for the classiﬁer, our TMN framework allows the combination of multiple options, e.g., CNN and RNN, which can be determined by the speciﬁc application scenario. Formally, given X = {x1,x2,...,xM} as the input with M short text instances, each instance x is processed into two representations: bag-ofwords (BoW) term vector xBoW ∈ RV and word index sequence vector xSeq ∈ RL, where V is the vocabulary size and L is the sequence length. xBoW is fed into the neural topic model to induce latenttopics. Suchtopicsarefurthermatchedwith the embedded xSeq to learn classiﬁcation features in the topic memory mechanism. Then, the classiﬁer concatenates the representations produced by the topic memory mechanism and the embedded xSeq to predict the classiﬁcation label y for x.

***Random forest***

**Random forests** or **random decision forests** are an [ensemble learning](https://en.wikipedia.org/wiki/Ensemble_learning) method for [classification](https://en.wikipedia.org/wiki/Statistical_classification), [regression](https://en.wikipedia.org/wiki/Regression_analysis) and other tasks that operate by constructing a multitude of [decision trees](https://en.wikipedia.org/wiki/Decision_tree_learning) at training time and outputting the class that is the [mode](https://en.wikipedia.org/wiki/Mode_(statistics)) of the classes (classification) or mean prediction (regression) of the individual trees.[[1]](https://en.wikipedia.org/wiki/Random_forest#cite_note-ho1995-1)[[2]](https://en.wikipedia.org/wiki/Random_forest#cite_note-ho1998-2) Random decision forests correct for decision trees' habit of [overfitting](https://en.wikipedia.org/wiki/Overfitting) to their [training set](https://en.wikipedia.org/wiki/Test_set).[[3]](https://en.wikipedia.org/wiki/Random_forest#cite_note-elemstatlearn-3):587–588

The first algorithm for random decision forests was created by [Tin Kam Ho](https://en.wikipedia.org/wiki/Tin_Kam_Ho)[[1]](https://en.wikipedia.org/wiki/Random_forest#cite_note-ho1995-1) using the [random subspace method](https://en.wikipedia.org/wiki/Random_subspace_method),[[2]](https://en.wikipedia.org/wiki/Random_forest#cite_note-ho1998-2) which, in Ho's formulation, is a way to implement the "stochastic discrimination" approach to classification proposed by Eugene Kleinberg.[[4]](https://en.wikipedia.org/wiki/Random_forest#cite_note-kleinberg1990-4)[[5]](https://en.wikipedia.org/wiki/Random_forest#cite_note-kleinberg1996-5)[[6]](https://en.wikipedia.org/wiki/Random_forest#cite_note-kleinberg2000-6)

An extension of the algorithm was developed by [Leo Breiman](https://en.wikipedia.org/wiki/Leo_Breiman)[[7]](https://en.wikipedia.org/wiki/Random_forest#cite_note-breiman2001-7) and [Adele Cutler](https://en.wikipedia.org/wiki/Adele_Cutler),[[8]](https://en.wikipedia.org/wiki/Random_forest#cite_note-rpackage-8) who registered[[9]](https://en.wikipedia.org/wiki/Random_forest#cite_note-9) "Random Forests" as a [trademark](https://en.wikipedia.org/wiki/Trademark) (as of 2019, owned by [Minitab, Inc.](https://en.wikipedia.org/wiki/Minitab)).[[10]](https://en.wikipedia.org/wiki/Random_forest#cite_note-10) The extension combines Breiman's "[bagging](https://en.wikipedia.org/wiki/Bootstrap_aggregating)" idea and random selection of features, introduced first by Ho[[1]](https://en.wikipedia.org/wiki/Random_forest" \l "cite_note-ho1995-1) and later independently by Amit and [Geman](https://en.wikipedia.org/wiki/Donald_Geman" \o "Donald Geman)[[11]](https://en.wikipedia.org/wiki/Random_forest#cite_note-amitgeman1997-11) in order to construct a collection of decision trees with controlled variance.

***8F model***

8F model refers to a custom model built over customised handling and identification of features which are further used for classification of the dataset so as to form the base of supervised learning.

To begin with, we identified seven generic categories (classes) that the users may be

interested in. We choose these categories to be as diverse as possible and ideally hope

that almost all the tweets can be classified into one of seven chosen categories. Therefore,

based on the user intentions on Twitter [1] such as daily chatter, conversations, sharing

information, URLs, and reporting news, we come up with the following seven classes:

* Neutral News
* Personal News
* Opinionated News
* Opinions
* Deals
* Events
* Private Messages

These eight features are defined as authorship information (Nominal) and the presence of:

* Shortening of words and slangs (Binary)
* Time-event information (Binary)
* Opinions (Binary)
* Emphasis on words (Binary)
* Currency, statistical information (Binary)
* Reference to another user at beginning of tweet (Binary)
* Reference to another user within tweet (Binary)

***Inference:***

On analysing the above models for the task of short text classification, a valuable insight has been extracted.

***8F model:***

* Thought of to be the first and foremost model to be used
* proved to be based on a completely generic approach
* which can only perform in some strict custom made conditions and not in general conditions.
* Used custom made rules to create classes and to create a labelled dataset.
* For the custom rules, performed outstandingly well.

***BoW***

* Most classic model for text classification
* Generally performs well on text classification, but not likely in the case of handling short text
* Development of a superbly sparse word vector, lead to misclassification of the instances

***N-grams***

* Improvement over the basic bag of words model.
* Implementation of 2-3-grams improve the accuracy of the model by a significant level
* Similar problem to the bag of words, sparse word matrix leads to curse of dimensionality and thus misclassification of the instances.

***Topic Memory Networks:***

* A much more intelligent solution to the problem of short text classification
* Use of deep learning models improve the learning rate of the model, thus leading to significantly better results.
* Latent topics are extracted from each of the instance to add additional level of information to the data.
* Use of latent topics for better classification reduces the sparsity of the word vector formed and thus leads to better results.

***Conclusion:***