IJCNN 2019. International Joint Conference on Neural Networks. Budapest, Hungary. 14-19 July 2019

Seq2Seq Deep Learning Models for Microtext Normalization

Ranjan Satapathy, Yang Li, Sandro Cavallari, Erik Cambria
School of Computer Science and Engineering
Nanyang Technological University
Singapore
{satapathy.ranjan, li.yang, sandro001, cambria}@ntu.edu.sg

***Abstract*—Microtext analysis is a crucial task for gauging social media opinion. In this paper, we compare four different deep learning encoder-decoder frameworks to handle microtext normalization problem. The frameworks have been evaluated on four different datasets in three different domains. To understand the impact of microtext normalization, we further integrate the framework into a sentiment classification task. This paper is the first of its kind to incorporate deep learning into a microtext normalization module and improve the sentiment analysis task. We show our models as a sequence to sequence character to word encoder-decoder model. We compare four deep learning models for microtext normalization task which further improve the accuracy of the sentiment analysis. Results show that the attentive LSTM and GRU cell both increase the sentiment analysis accuracy in the range of 4%–7% whereas LSTM and CNN with LSTM improve the accuracy in the range of 2%–4%.**

I. Introduction

In recent years, the rise and expansion of social media enabled users to share their views and interests in an impromptu manner and in real time. Since Web 2.0 is meant for human consumption, web users tend to abbreviate English terms by relying on the phonetic form of both numbers and letters. For example, they write terms or sentences such as "c u 2morrow" (see you tomorrow), "tgif" (thank God it's Friday) and "abt" (about) which may not be found in standard English but are widely seen in SMS, tweets, Facebook posts, blogs, discussion forums and chat logs. In this way, computer-mediated communication has generated a slang often referred to as "microtext" which differs from well-written text [59].

Microtext became one of the most widespread communication forms among users due to its casual writing style and colloquial tone [40]. According to CTIA [37], Americans sent 196.9 billion text messages in 2011 compared to 12.5 billion in 2006. Another statistic showed that until May 2016 there were nearly 500 million tweets sent each day, i.e., 6,000 tweets every second[1].

Given that most data today is mined from the Web, microtext analysis is vital for many natural language processing (NLP) and data mining tasks, as most text classifiers are trained on plain English. In the context of sentiment analysis, microtext normalization is a necessary step for pre-processing text before polarity detection is performed [13].

[1]http://brandwatch.com/blog/44-twitter-stats-2016

Since it is heavily based on phonetics, microtext is very much language-dependent: the same set of characters could have completely different meaning in different languages, e.g., '555' is negative in Chinese language (because the number '5' is pronounced as 'wu' and 'wuwuwu' resembles a crying sound) but positive in Thai (as the number 5 is pronounced as 'ha' and three consecutive 5s correspond to the expression 'hahaha').

The two main features of microtext are relaxed spelling and reliance on emoticons and out-of-vocabulary (OOV) words involving phonetic [60] (e.g., 'b4' for 'before'), emotional emphasis (e.g., 'goooooood' for 'good') and popular acronyms (e.g., 'otw' for 'on the way') [57], [59], [67]. The challenge arises when trying to automatically rectify and replace them with the correct in-vocabulary (IV) words [41], [52]. The structure of our framework follows [17].

This paper is the first of its kind to incorporate deep learning into a microtext normalization module and shows how to improve the sentiment analysis by a significant margin. For polarity detection, we have used Sentic API[2]. We compare four deep learning models which improved the accuracy of the sentiment analysis namely: long short-term memory (LSTM), Attentive LSTM, convolutional neural network (CNN) with LSTM and gated recurrent units (GRU). Results show that the attentive LSTM and GRU both increase the sentiment analysis accuracy in the range of 4%–7% though LSTM and CNN with LSTM improves the accuracy in the range of 2%–4%.

The rest of the paper is organized as follows: Section II presents related work in the field of microtext normalization; Section III describes the models used for the microtext normalization task; Section IV describes the datasets used to train and test the models; Section V proposes experimental results; finally, Section VI provides concluding remarks.

II. Related Work

Since the beginning of human history, people have been considered by nature as social animals who are highly susceptible to opinions as practically all undertakings and behaviors are influenced by them. Accordingly, when choices are to be taken, individuals and organizations frequently look for others' opinions.

[2]http://sentic.net/api

Real Time Tweets	Transformed In-Vocabulary Tweets
D show jst gettin started .	The show is just getting started .
idc who had u first , ima have u last	I Don't Care who had you first , i am going to have you last
I've had my Xbox 360 since $8th$ grade . It's so old lol	I have had my Xbox 360 since $8th$ grade . It is so old laughing_out_loud
lol dey r asum	laughing_out_loud they are awesome
r u thr ?	Are you there ?

TABLE I
SAMPLE REAL TIME TWEETS AND ITS IN-VOCABULARY FORM

Opinions and their associated concepts such as sentiments, emotions, attitudes, and evaluations are the focuses of the study of sentiment analysis. This section is further subdivided into sentiment analysis and microtext analysis.

A. Sentiment Analysis

Sentiment analysis [12] is a branch of affective computing research [53] that aims to classify text (but sometimes also audio and video [56]) into either positive or negative (but sometimes also neutral [16]). While most works approach it as a simple categorization problem, sentiment analysis is actually a suitcase research problem [13] that requires tackling many NLP sub-tasks, including aspect extraction [54], named entity recognition [45], word polarity disambiguation [66], temporal tagging [69], personality recognition [46], and sarcasm detection [55].

[1], [25] state that sentiment analysis has numerous applications with many purposes such as the detection of the mood of the market based on specialists' opinions [35], [49], the analysis of customers' reviews about products or services [23], [28], the analysis of touristic sites through tourists' comments [7], and the analysis of politicians [4] or topics connected to politics [24]. By itself, sentiment analysis systems are often characterized into statistics-based and knowledge-based systems [11]. On the one hand, statistical approaches have proven to be generally semantically feeble [15]. This is because statistical text classifiers only work with adequate precision when given a satisfactorily large text input [14]. Although statistical approaches can effectively classify users' text on the page or section level, they do not work correctly on smaller text parts such as sentences.

On the other hand, concept-level sentiment analysis is a task which relies on large semantic knowledge bases which have recently growing interest within the scientific community as well as the business world. It uses semantic text analysis through the use of web ontologies or semantic networks which enable aggregation of the conceptual and affective information associated with natural language opinions [2], [8], [26], [58]. The analysis at concept level enables to infer the semantic and affective information associated with natural language opinions and, well ahead, to enable a comparative fine-grained feature-based sentiment analysis. Henceforth, the approach proposed relies on concept-level sentiment analysis because it leaves behind the sightless use of keywords and word co-occurrence counts somewhat depending on the implied features linked with natural language concepts.

B. Microtext Analysis

Microtext has become ubiquitous in today's communication. This is partly a consequence of Zipf's law, or principle of least effort (for which people tend to minimize energy cost at both individual and collective levels when communicating with one another), and it poses new challenges for NLP tools which are usually designed for well-written text [29]. Normalization is the task of transforming unconventional words/sentences to their respective standard counterpart.

In [39], the authors present a novel unsupervised method to translate Chinese abbreviations. It automatically extracts the relation between a full-form phrase and its abbreviation from monolingual corpora and induces translation entries for the acronym by using its full-form as a bridge. [27] uses a classifier to detect OOV words, and generates correction candidates based on morphophonemic similarity. The types and features of microtext are reliant on the nature of the technological support that makes them possible. This means that microtext will vary as new communication technologies emerge. In our related work, we categorized normalization into three well-known NLP tasks, namely: spelling correction, statistical machine translation (SMT), and automatic speech recognition (ASR).

1) Spelling Correction: Correction is executed on a word-per-word basis seen as a spelling checking task. This model gained extensive attention in the past and diversity of correction practices have been endorsed by [21], [9], [38], [50], [62]. Instead, [61] and [22] proposed a categorization of abbreviation, stylistic variation, prefix-clipping, which was then used to estimate their probability of occurrence. Thus far, the spelling corrector became widely popular in the context of SMS, where [18] advanced the hidden Markov model whose topology takes into account both "graphemic" variants (e.g., typos, omissions of repeated letters, etc.) and "phonemic" variants (e.g., spellings that resemble the word's pronunciation).

2) Statistical Machine Translation: SMT outlooks microtext as a foreign language that has to be translated to plain English, meaning that normalization is done through an SMT task. When compared to the previous task, this method appears to be rather straightforward and better since it has the possibility to model (context-dependent) one-to-many relationships which were out-of-reach previously [34]. Some examples of works include [3], [30], [51]. However, the SMT still overlooks some features of the task, particularly the fact that lexical creativity verified in social media messages is barely captured in a stationary sentence board.

3) Automatic Speech Recognition: ASR considers that microtext tends to be a closer approximation of the word's phonemic representation rather than its standard spelling. As follows, the key of microtext normalization becomes very similar to speech recognition which consists of decoding a word sequence in a (weighted) phonetic framework. [34] proposed to handle normalization based on the observation that text messages present a lot of phonetic spellings, while more recently [31] proposed an algorithm to determine the probable pronunciation of English words based on their spelling. Although the computation of a phonemic representation of the message is extremely valuable, it does not solve entirely all the microtext normalization challenges (e.g., acronyms and misspellings do not resemble their respective IV words' phonemic representation). Authors in [6] have merged the advantages of SMT and the spelling corrector model.

Some previous work on text normalization that has made use of neural techniques includes [19], [48]. The latter work, for example, achieved second place in the constrained track of the ACL 2015 W-NUT Normalization of Noisy Text [5], achieving an F1 score of 81.75%. In the work, we report below on microtext normalization in English, we achieve accuracies that are comparable or better than that result (to the extent that it makes sense to compare across such quite different tasks).

III. Deep Learning Models

We developed a seq2seq model with variations to discover which deep learning seq2seq model works best for most datasets in different domains. The proposed framework for normalization consists of two parts: an encoder for reading the input sequence and encoding it into a fixed-length vector, and a decoder for decoding the fixed-length vector and outputting the predicted sequence. The task of translating OOV words can be understood from the perspective of machine learning as learning the conditional distribution $p(\mathbf{w} \mid \mathbf{s})$ of a target sentence $\mathbf{w}$ given a source sentence $\mathbf{s}$. Once the conditional distribution is learned by a model, one can use the model to directly sample a target sentence given a source sentence either by actual sampling or by using a search algorithm to find the maximum of the distribution.

The model takes an input characters, and outputs an IV word. An OOV word is unconventional in structure whereas, an IV word is written in its standard form. We structured our normalization task by using a character-based encoder-decoder model with variations which will be discussed in III-A in detail.

$$p(\mathbf{w} \mid \mathbf{s}, \theta) = \prod_{j=1}^{m} p(w_j \mid w_{<j}, \mathbf{s}), \tag{1}$$

where θ represents a set of all model parameters in the encoder-decoder model, which are determined by the parameter-estimation process of a standard softmax cross-entropy loss minimization using training data.

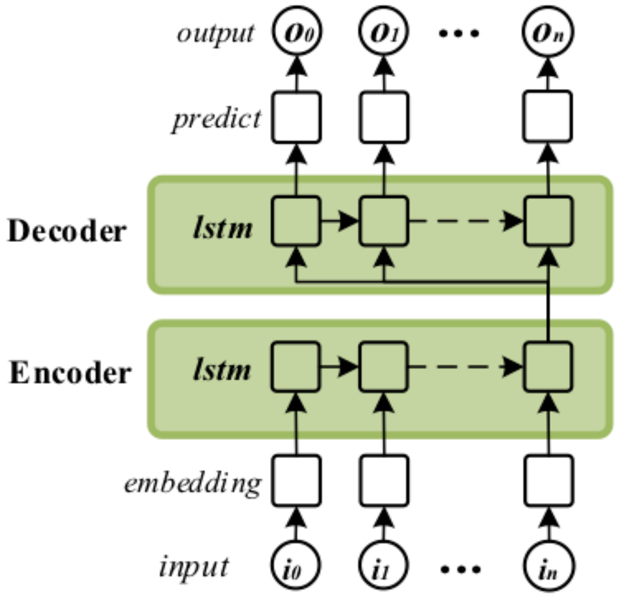

Fig. 1. Encoder-Decoder framework.

Therefore, given θ and $\mathbf{s}$, our normalization task is defined as finding $\mathbf{w}$ with maximum probability given by Equation 2

$$\hat{\mathbf{w}} = \arg\max_{\mathbf{w}} p(\mathbf{w} \mid \mathbf{s}, \theta), \tag{2}$$

where $\hat{\mathbf{w}}$ represents the model output.

Each character in the input can be encoded using one-hot vector $x_i \in \{0,1\}^v$. Hence, the inputs are represented as a binary matrix $x_1, \cdots, x_n \in (0,1)$ where v is the vocabulary and n is the number of input texts, with padding wherever necessary. Then LSTM was applied to get the feature of the input, in the LSTM cell the hidden units were varied from 128, 256, and 512 and was fixed at 256 as this gave the best result. Based on the learned feature, another LSTM was applied to get the reconstructed characters. In this paper, the learning rate is fixed at 10^{-4}. Batch size is set to 64.

The prediction of word at each time step is given by the Equation 3:

$$p(w_t|\cdot) = \delta(\phi(x_t), h_{t-1}) \tag{3}$$

where $\delta(\cdot)$ denotes the softmax function, w_t refers to the word at time-step t, $\phi(x_t)$ represents the encoding value from the character at time-step t, and h_{t-1} denotes the previous hidden value of LSTM in the decoding process.

To learn the model parameters, there are two cases in the building. The cross-entropy loss in this framework is depicted in Equation 4. Because all of the predicted output are set to the fixed length, whereas the output word length varies. The empty places which has no character is filled with a placeholder, and it will be counted into the final loss in case of "no mask" explained in section V-A.

Cases	Datasets	LSTM	Attentive LSTM	GRU	CNN with LSTM
With Mask	Lexicon	77.36%	**80.63%**	78.40%	77.82%
	Tweets	64.71%	64.04%	69.85%	**69.90%**
	SMS	76.15%	76.84%	**77.24%**	77.20%
	CMUdict	82.78%	82.17%	**85.21%**	84.73%
Without Mask	Lexicon	78.58%	**80.33%**	77.22%	78.55%
	Tweets	65.71%	66.93%	65.85%	**68.55%**
	SMS	76.06%	**77.09%**	76.60%	76.59%
	CMUdict	81.16%	81.1%	85.75%	**86.87%**

TABLE II
ACCURACY OF ENCODER-DECODER MODELS ON MULTIPLE DATASET WITH MASK AND WITHOUT MASK

The cross-entropy losses are trained by the Adam Optimization algorithm [33].

$$L = -\frac{1}{m}\sum_{i}^{m} w_i \log p(w_i, \theta) \tag{4}$$

$$L_M = -\frac{1}{m}\sum_{i}^{m} w_i \log p(w_i, \theta) M \tag{5}$$

where m in the denominator is the output length, θ represents a set of all model parameters in the encoder-decoder model. $M \in \{1, 0\}^m$ is from the true length of the output, the place has character is set to 1, otherwise is 0.

A. Model Variations

We used a 2-layer reader that reads input characters, an embedding layer, and a 2-layer decoder that produces word sequences. We used L2 loss for encoder and ReLU loss for decoder. For training we used AdamOptimizer and learning rate was set to $1e-4$. Batch size is set to 64 and latent dimension = 256. The data were partitioned on train-test set as 80-20.

We have considered four different encoder-decoder architectures namely, LSTM, LSTM with Attention [43], CNN with LSTM and GRU encoder-decoder. We compare LSTM, attentive LSTM, GRU and LSTM with CNN in the task of sequence modeling for text normalization. Sequence modeling aims at learning a probability distribution over sequences, as described in Equation 1, given a set of training sequences. Here are detailed descriptions:

- **LSTM:** LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods is practically their default behavior, not something they struggle to learn. Mathematically LSTM is given by 6

$$\begin{aligned} f_t &= \sigma(W_f.[h_{t-1}, x_t] + b_f) \\ i_t &= \sigma(W_i.[h_{t-1}, x_t] + b_i) \\ C_t^+ &= \tanh(W_C[h_{t-1}, x_t] + b_C) \\ C_t &= f_t * C_{t-1} + i_t * C_t^+ \\ o_t &= \sigma(W_o[h_{t-1}, x_t] + b_o) \\ h_t &= o_t * \tanh(C_t) \end{aligned} \tag{6}$$

- **Attentive LSTM:** We added attention mechanism [44] in the decoder module. Then, the attention scores were calculated and that outputs the context vector which will be concatenated with hidden state of the decoder, thereby the final word is predicted. Attention weight-vector α is given by Equation 7:

$$\begin{aligned} h &= LSTM(\mathbf{s}) \\ \alpha &= \delta(W_h h) \\ c &= \alpha h \end{aligned} \tag{7}$$

  W_t, W_h are the parameters, $\delta(\cdot)$ is the softmax function, $\mathbf{s}$ is input characters, h is the encoding of the input vector (hidden value) and c denotes the output.
- **GRU:** GRUs are a gating mechanism in recurrent neural networks, introduced by [20]. Apart from LSTM, GRU is another popular model for sequence data. The mathematical representations are given as:

$$\begin{aligned} z_t &= \sigma(W_z.[h_{t-1}, x_t]) \\ r_t &= \sigma(W_r.[h_{t-1}, x_t]) \\ h_t^+ &= \tanh(W.[r_t * h_{t-1}, x_t]) \\ h_t &= (1 - z_t) * h_{t-1} + z_t * h_t^+) \end{aligned} \tag{8}$$

  where h_t, z_t, r_t denote the hidden value, the update gate vector, and the reset gate vector, respectively. W, U, b are the parameter matrices and vector. σ and $\tanh$ are the sigmoid function and hyperbolic tangent respectively. h_t^+ is the current memory content and h_t is the final memory at the current time step
- **LSTM with CNN as a feature extractor:** We also used CNN as a feature extractor and compared it with the rest of the models. CNNs are often applied both on image data [36] and the text data [32]. In our work, the structure of CNN from [32] is used to extract the features from the input data. The mathematical representations are given as:

$$h_i = \tanh(w x_{i:i+j-1} + b) \tag{9}$$

  This filter is applied to each possible window of characters in the source sentence $x_{1:j}, x_{2:j+1}, ..., x_{n-j+1:n}$ to produce a feature map given by:

$$h = [h_1, h_2, ..., h_{n-j+1}] \tag{10}$$

  where h is the hidden value.

One-hot embedding has been used to convert input texts (in all the datasets) to vectors. As microtext comprises OOV words, word embeddings [47] pretrained on English words will not be useful as they will not be able to determine the correct embeddings.

	Features	Microtext Lexicon	Norm Tweets	NUS SMS	CMUdict
Input	Char set	97	165	159	85
	Char MaxLen	15	146	223	33
Output	Wordset	1565	6328	5397	77
	Word MaxLen	13	46	59	32

TABLE III
STATISTICS OF THE DATASETS

Instead, each input (word or sentence) is treated as a composition of characters, which contains no more than 165 characters in total in our corpus. In our model, after encoding all of the characters with one-hot embedding, a single layer of the dense embedding is added.

IV. Datasets Used

The purpose of our work is to normalize any kind of data irrespective of their structure and property. Hence, we evaluate our model on four datasets from three different domains. In the tables below, we renamed Microtext Lexicon as Lexicon, Normalized Tweets as Tweets, NUS Social Media Text Normalization and Translation Corpus as SMS and the phonetic dictionary as the Carnegie Mellon Pronouncing Dictionary (CMUdict). The statistics of these datasets are shown in Table III. Our model is a character-level input to word-level output encoder-decoder model. The *Input char set* is the total number of input characters in the respective datasets. *Word set* is the number of output words in the respective corpus corresponding to the given input. *Input MaxLen* is the max length of the input, and the *Word MaxLen* which is the max length of the corresponding words. The details of the datasets used are explained in the following subsections.

A. Microtext Lexicon

Authors in [60], developed a lexicon for microtext normalization. This acts as a look-up table for performing normalization. We used their dataset to learn the intrinsic character-level behavior of the different classes of microtext as mentioned in their work.

B. Carnegie Mellon Pronouncing Dictionary

Carnegie Mellon Pronouncing Dictionary (CMUdict) is a free pronouncing dictionary of English, suitable for uses in speech technology CMU dictionary[3]. It consists of phonetic representation for each word. We use this dictionary to learn the phonetics of the words in the lexicon. Our results show that the encoder-decoder model works best with the CMU dictionary. As it is a speech dictionary, preprocessing was required before inserting it to our model. Preprocessing includes removing terms like '!EXCLAMATION-POINT' which is written as 'EH2 K S K L AH0 M EY1 SH AH0 N P OY2 N T' from the dictionary.

[3]http://speech.cs.cmu.edu/cgi-bin/cmudict

C. NUS SMS Corpus

This corpus has been created from the NUS English SMS corpus[4], the authors [64] randomly selected 2,000 messages. The messages were first normalized into standard English and then translated into standard Chinese. For our training and testing purposes, we only used the actual messages and their normalized English version. It also contains non-English terms specifically the Singaporean colloquial language called Singlish, which LSTM had no problem in learning. Singlish is an English-based creole that is lexically and syntactically influenced by Hokkien, Cantonese, Mandarin, Malay and Tamil [10]. It is primarily a spoken variety in Singapore, to emerge as a means of online communication for Singaporeans [65].

D. Normalized Tweets

Authors in [60], built a lexicon which consisted of normalized OOV words from tweets and their IV counterparts. We demonstrate our results on this dataset utilizing the context within the sentences by including unconventional written sentences together with the transformed standard ones.

V. Results Discussion

In this paper, we show different seq2seq models to normalize informal English to plain English. Initially, we observed no change in accuracy with a consistent decrease in model loss. With a change in variation in the batch size, we found that 32 is the best batch size for all these datasets. In this section, we will introduce the result discussions with and without mask and discussion on experimental results.

A. Effect of Masking

The padding process has shown that not only the model can be efficiently processed but also that the output is affected as perceived in our results. On the one hand, the addition of the mask is performed to the position where there are no characters in the output. On the other hand, without the mask it pads the output vector to make it have a fixed length. The results have been different since without mask the position without a word is also counted as the right prediction.

The results of each model on different datasets have been depicted in Table II. In this paper, we show different seq2seq models to normalize the unconventional texts to standard English texts.

B. Sentiment Analysis

Sentiment analysis is evaluated on the NUS SMS data and normalized tweets, and not on the other lexicons as most of the words are abbreviated and do not contain any sentiment associated with the words. For polarity detection, we relied on the Sentic API[5]. In this section, we compare the OOV and models' IV texts for polarity detection. Both the OOV and model's output IV texts are fed to the polarity detection module.

[4]http://github.com/kite1988/nus-sms-corpus
[5]http://sentic.net/api

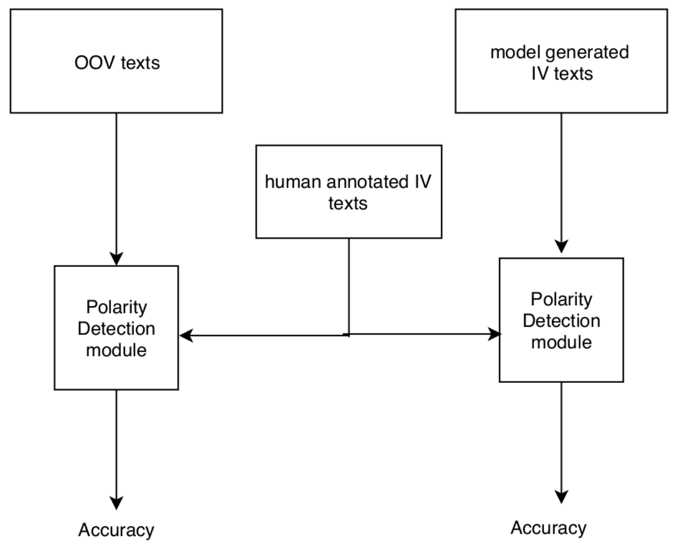

Fig. 2. Architecture for polarity detection comparison.

Both of them are compared against the human annotated IV texts' output, as shown in figure 2. Table IV depicts the output for polarity detection module for each of the microtext normalization model discussed in section III. It can be seen that attentive LSTM and GRU have improved the accuracy by 6% and 6.3% respectively for the normalized tweets and 4.4% and 5.3% respectively for the NUS SMS data. Other models also increase the accuracy but only in the range of 2%–4% for both the dataset. The improvement in accuracy depicts that microtext is indeed an important preprocessing module when dealing with social media texts.

C. Discussion

From Table II, it can be observed that the best results from the different models across a variety of datasets are not the same, e.g., in the Microtext Lexicon, it was observed that the input and output comprised no more than "15 characters" and "13 words", respectively. This means that attentive LSTM works well for short texts.

The GRU model works better for both longer and shorter sentences, as well as for words with the mask, yet CNN with LSTM works better without the mask. Therefore, GRU and CNN with LSTM both demonstrate the capability to be used in text normalization in all the three domains.

Dataset	OOV Output	Models			
		LSTM	Attentive LSTM	GRU Cell	CNN with LSTM
nus_sms_data	78%	80.4%	82.4%	82.8%	81.3%
norm_tweets	77.47%	79.8%	83.5%	83.8%	81.5%

TABLE IV
COMPARISON OF POLARITY ACCURACY FOR DIFFERENT MODELS

VI. Conclusion

It could be thought that microtext normalization is as simple as performing find-and-replace pre-processing [31]. However, the wide-ranging diversity of spellings makes this solution impractical (e.g., the spelling of the word "tomorrow" includes "tom, 2moro, 2mr, tmr" among others). Social media language is considerably different from other written text. Many of the efforts to illustrate and overcome this discrepancy have focused on normalization. In this paper, we show different seq2seq models to normalize unconventional sentences to standard English. Initially, we observed no change in accuracy with a consistent decrease in model loss. However, with the change in batch sizes, we found 32 to be the best batch size over all these datasets. The best performing dataset is CMUdict followed by Lexicon. CMUdict gives above 80% accuracy in both with and without mask model.

Our method depicts the application of encoder-decoder model, with variations on multiple datasets. In our experiments, we observed that the results of "PHON" classes in microtext lexicon [60] were promising. However, due to very little training data in "PHON" class (roughly 500 words), we had to train with the whole dataset (more than 7000 words). We believe that an artificial intelligence needs to understand how to map an unusual word like "lol" to "laughing out loud" rather converting "12" to "twelve" [68] to detect sentiments with more precision from an input text.

The results show that the proposed encoder-decoder GRU model works overall well for all the normalization dataset. In the future, the obtained results of this model can be potentially used for detecting multilingual patterns [42] along with contributing to the normalization of multilingual frameworks [63]. The increase in accuracy for polarity detection when compared to OOV texts clearly shows the need for microtext normalization when performing NLP tasks over social media text. In the future, we plan to expand it to other tasks in NLP such as subjective detection, aspect extraction and others.

References

[1] André Luiz Firmino Alves, Cláudio de Souza Baptista, Anderson Almeida Firmino, Maxwell Guimarães de Oliveira, and Anselmo Cardoso de Paiva. A Spatial and Temporal Sentiment Analysis Approach Applied to Twitter Microtexts. *Journal of Information and Data Management*, 6(2):118–129, 2016.

[2] Matheus Araújo, Pollyanna Gonçalves, Meeyoung Cha, and Fabrício Benevenuto. iFeel: a system that compares and combines sentiment analysis methods. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 75–78. ACM, 2014.

[3] AiTi Aw, Min Zhang, Juan Xiao, and Jian Su. A phrase-based statistical model for sms text normalization. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 33–40. Association for Computational Linguistics, 2006.

[4] Rawia Awadallah, Maya Ramanath, and Gerhard Weikum. PolariCQ: Polarity classification of political quotations. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 1945–1949. ACM, 2012.

[5] Timothy Baldwin, Marie-Catherine de Marneffe, Bo Han, Young-Bum Kim, Alan Ritter, and Wei Xu. Shared tasks of the 2015 workshop on noisy user-generated text: Twitter lexical normalization and named entity recognition. In *Proceedings of the Workshop on Noisy User-generated Text*, pages 126–135, 2015.

[6] Richard Beaufort, Sophie Roekhaut, Lousie-Amé lie Cougnon, and Cé drick Fairon. A hybrid rule/model-based finite-state framework for normalizing SMS messages. In *ACL*, pages 770–779. Association for Computational Linguistics, 2010.

[7] Eivind Bjørkelund, Thomas H. Burnett, and Kjetil Nørvåg. A study of opinion mining and visualization of hotel reviews. *Proceedings of the 14th International Conference on Information Integration and Web-based Applications & Services*, pages 229–238, 2012.

[8] Felipe Bravo-Marquez, Marcelo Mendoza, and Barbara Poblete. Meta-level sentiment models for big social data analysis. *Knowledge-Based Systems*, 69:86–99, 2014.

[9] Eric Brill and Robert C. Moore. An improved error model for noisy channel spelling correction. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 286–293, 2000.

[10] Adam Brown. *Singapore English in a nutshell: An alphabetical description of its features*. Federal Publications, 1999.

[11] Erik Cambria. Affective computing and sentiment analysis. *IEEE Intelligent Systems*, 31(2):102–107, 2016.

[12] Erik Cambria, Dipankar Das, Sivaji Bandyopadhyay, and Antonio Feraco. *A Practical Guide to Sentiment Analysis*. Springer, Cham, Switzerland, 2017.

[13] Erik Cambria, Soujanya Poria, Alexander Gelbukh, and Mike Thelwall. Sentiment Analysis is a Big Suitcase. *IEEE Intelligent Systems*, 32(6):74–80, 2017.

[14] Erik Cambria, Soujanya Poria, Devamanyu Hazarika, and Kenneth Kwok. SenticNet 5: Discovering conceptual primitives for sentiment analysis by means of context embeddings. In *AAAI*, pages 1795–1802, 2018.

[15] Erik Cambria and Bebo White. Jumping NLP curves: a review of natural language processing research. *IEEE Computational Intelligence*, 9(2):48–57, 2014.

[16] Iti Chaturvedi, Edoardo Ragusa, Paolo Gastaldo, Rodolfo Zunino, and Erik Cambria. Bayesian network based extreme learning machine for subjectivity detection. *Journal of The Franklin Institute*, 355(4):1780–1797, 2018.

[17] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[18] Monojit Choudhury, Rahul Saraf, Vijit Jain, Sudeshna Sarkar, and Anupam Basu. Investigation and modeling of the structure of texting language. *International Journal of Document Analysis and Recognition*, 10(3-4):157–174, 2007.

[19] Grzegorz Chrupała. Normalizing tweets with edit scripts and recurrent neural embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 2, pages 680–686. Association for Computational Linguistics Baltimore, Maryland, 2014.

[20] Junyoung Chung, Çalar Gülçehre, Kyunghyun Cho, and Yoshua Bengio. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv e-prints*, abs/1412.3555, 2014. Presented at the Deep Learning workshop at NIPS2014.

[21] Kenneth W. Church and William A. Gale. Probability scoring for spelling correction. *Statistics and Computing*, 1(2):93–103, 1991.

[22] Paul Cook and Suzanne Stevenson. An unsupervised model for text message normalization. In *Proceedings of the workshop on computational approaches to linguistic creativity*, pages 71–78, 2009.

[23] Magdalini Eirinaki, Shamita Pisal, and Japinder Singh. Feature-based opinion mining and ranking. *Journal of Computer and System Sciences*, 78(4):1175–1184, 2012.

[24] Yi Fang, Luo Si, Naveen Somasundaram, and Zhengtao Yu. Mining contrastive opinions on political texts using cross-perspective topic model. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 63–72. ACM, 2012.

[25] Ronen Feldman. Techniques and applications for sentiment analysis. *Communications of the ACM*, 56(4):82–89, 2013.

[26] Gizem Gezici, Rahim Dehkharghani, Berrin Yanikoglu, Dilek Tapucu, and Yucel Saygin. Su-sentilab: a classification system for sentiment analysis in Twitter. In *Proceedings of the International Workshop on Semantic Evaluation, Atlanta*, pages 471–477, 2013.

[27] Bo Han and Timothy Baldwin. Lexical normalisation of short text messages: Makn sens a# twitter. In *ACL*, pages 368–378, 2011.

[28] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177, 2004.

[29] Clayton J. Hutto and Eric Gilbert. VADER: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth International AAAI Conference on Weblogs and Social Media*, pages 216–225, 2014.

[30] Max Kaufmann and Jugal Kalita. Syntactic normalization of Twitter messages. *natural language processing, Kharagpur, India*, 2010.

[31] Richard Khoury. Microtext Normalization using Probably-Phonetically-Similar Word Discovery. In *Wireless and Mobile Computing, Networking and Communications (WiMob), 2015 IEEE 11th International Conference on.*, pages 392–399, 2015.

[32] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751. Empirical Methods in Natural Language Processing (EMNLP), 2014.

[33] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *3rd International Conference for Learning Representations (ICLR)*, 2015.

[34] Catherine Kobus, François Yvon, and G éraldine Damnati. Normalizing SMS: are two metaphors better than one? In *Proceedings of the 22nd International Conference on Computational Linguistics*, volume 1, pages 441–448. Association for Computational Linguistics, 2008.

[35] Moshe Koppel and Itai Shtrimberg. Good news or bad news? Let the market decide. In *Computing attitude and affect in text: Theory and applications*, pages 297–301. Springer Netherlands, 2006.

[36] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[37] Chen Li and Yang Liu. Normalization of Text Messages Using Character-and Phone-based Machine Translation Approaches. *INTERSPEECH*, pages 2330–2333, 2012.

[38] Mu Li, Yang Zhang, Muhua Zhu, and Ming Zhou. Exploring distributional similarity based models for query spelling correction. In *ACL*, pages 1025–1032, 2006.

[39] Zhifei Li and David Yarowsky. Unsupervised translation induction for chinese abbreviations using monolingual corpora. *Proceedings of ACL-08: HLT*, pages 425–433, 2008.

[40] Fei Liu, Fuliang Weng, and Xiao Jiang. A broad-coverage normalization system for social media language. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 1035–1044. Association for Computational Linguistics, 2012.

[41] Fei Liu, Fuliang Weng, Bingqing Wang, and Yang Liu. Insertion, deletion, or substitution? Normalizing text messages without pre-categorization nor supervision. *ACL-HLT 2011 - Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2:71–76, 2011.

[42] Siaw Ling Lo, Erik Cambria, Raymond Chiong, and David Cornforth. A multilingual semi-supervised approach in deriving singlish sentic patterns for polarity detection. *Knowledge-Based Systems*, 105:236–247, 2016.

[43] Minh-Thang Luong, Eugene Brevdo, and Rui Zhao. Neural Machine Translation (seq2seq) Tutorial. *https://github.com/tensorflow/nmt*, 2017.

[44] Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. Effective Approaches to Attention-based Neural Machine Translation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1412–1421, Lisbon, Portugal, 2015. Association for Computational Linguistics.

[45] Yukun Ma, Erik Cambria, and Sa Gao. Label embedding for zero-shot fine-grained named entity typing. In *COLING*, pages 171–180, 2016.

[46] Navonil Majumder, Soujanya Poria, Alexander Gelbukh, and Erik Cambria. Deep learning-based document modeling for personality detection from text. *IEEE Intelligent Systems*, 32(2):74–79, 2017.

[47] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[48] Wookhee Min Bradford W Mott. Ncsu_sas_wookhee: A deep contextual long-short term memory model for text normalization. *ACL-IJCNLP 2015*, page 111, 2015.

[49] Neil O'Hare, Michael Davy, Adam Bermingham, Paul Ferguson, Páraic Sheridan, Cathal Gurrin, and Alan F Smeaton. Topic-dependent sentiment analysis of financial blogs. In *Proceedings of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion*, pages 9–16. ACM, 2009.

[50] Deana L. Pennell and Yang Liu. A Character-Level Machine Translation Approach for Normalization of SMS Abbreviations. In *IJCNLP*, pages 974–982, 2011.

[51] Deana L. Pennell and Yang Liu. Normalization of informal text. *Computer Speech & Language*, 28(1):256–277, 2014.

[52] Saŝa Petrović, Miles Osborne, and Victor Lavrenko. The Edinburgh Twitter corpus. In *Proceedings of the NAACL HLT Workshop on Computational Linguistics in a World of Social Media*, pages 25–26, 2010.

[53] Soujanya Poria, Erik Cambria, Rajiv Bajpai, and Amir Hussain. A review of affective computing: From unimodal analysis to multimodal fusion. *Information Fusion*, 37:98–125, 2017.

[54] Soujanya Poria, Erik Cambria, and Alexander Gelbukh. Aspect Extraction for Opinion Mining with a Deep Convolutional Neural Network. *Knowledge-Based Systems*, 108:42–49, 2016.

[55] Soujanya Poria, Erik Cambria, Devamanyu Hazarika, and Prateek Vij. A Deeper Look into Sarcastic Tweets Using Deep Convolutional Neural Networks. In *COLING*, pages 1601–1612, 2016.

[56] Soujanya Poria, Iti Chaturvedi, Erik Cambria, and Amir Hussain. Convolutional MKL based multimodal emotion recognition and sentiment analysis. In *ICDM*, pages 439–448, 2016.

[57] Jonathon Read. Using emoticons to reduce dependency in machine learning techniques for sentiment classification. In *Proceedings of the ACL student research workshop*, pages 43–48. Association for Computational Linguistics, 2005.

[58] Diego Reforgiato Recupero, Valentina Presutti, Sergio Consoli, and Andrea Nuzzolese Gangemi. Sentilo: frame-based sentiment analysis. *Cognitive*, 7(2):211–225, 2015.

[59] Kevin Dela Rosa and Jeffrey Ellen. Text classification methodologies applied to micro-text in military chat. In *Proc. Eight International Conference on Machine Learning and Applications*, pages 710–714, Miami, 2009.

[60] Ranjan Satapathy, Claudia Guerreiro, Iti Chaturvedi, and Erik Cambria. Phonetic-Based Microtext Normalization for Twitter Sentiment Analysis. In *ICDM*, pages 407–413, 2017.

[61] Richard Sproat, Alan W. Black, Stanley Chen, Shankar Kumar, Mari Ostendorf, and Christopher Richards. Normalization of non-standard words. *Computer speech & language*, 15(3):287–333, 2001.

[62] Kristiana Toutanova and Robert C. Moore. Pronunciation modeling for improved spelling correction. In *ACL*, pages 144–151, 2002.

[63] David Vilares, Haiyun Peng, Ranjan Satapathy, and Erik Cambria. Babelsenticnet: A commonsense reasoning framework for multilingual sentiment analysis. In *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1292–1298. IEEE, 2018.

[64] Pidong Wang and Hwee Tou Ng. A Beam-Search Decoder for Normalization of Social Media Text with Application to Machine Translation. In *HLT-NAACL*, pages 471–481, 2013.

[65] Mark Warschauer. The internet and linguistic pluralism. *Silicon literacies: Communication, innovation and education in the electronic age*, pages 62–74, 2002.

[66] Yunqing Xia, Erik Cambria, Amir Hussain, and Huan Zhao. Word polarity disambiguation using bayesian model and opinion-level features. *Cognitive Computation*, 7(3):369–380, 2015.

[67] Zhenzhen Xue, Dawei Yin, and Brian D. Davison. Normalizing Microtext. *Analyzing Microtext*, pages 74–79, 2011.

[68] Wojciech Zaremba and Ilya Sutskever. Learning to execute. *arXiv preprint arXiv:1410.4615*, 2014.

[69] Xiaoshi Zhong, Aixin Sun, and Erik Cambria. Time expression analysis and recognition using syntactic token types and general heuristic rules. In *ACL*, pages 420–429, 2017.