A Method of Combining Hidden Markov Model and Convolutional Neural Network for the 5G RCS Message Filtering

Bibu Gao * and Wenqiang Zhang

Academy for Engineering and Technology, Fudan University, Shanghai 200433, China; wqzhang@fudan.edu.cn
* Correspondence: 18110860031@fudan.edu.cn

Abstract: As one of the 5G applications, rich communication suite (RCS), known as the next generation of Short Message Service (SMS), contains multimedia and interactive information for a better user experience. Meanwhile, the RCS industry worries that spammers may migrate their spamming misdeeds to RCS messages, the complexity of which challenges the filtering technology because each of them contains hundreds of fields with various types of data, such as texts, images and videos. Among the data, the hundreds of fields of text data contain the main content, which is adequate and more efficient for combating spam. This paper first discusses the text fields, which possibly contain spam information, then use the hidden Markov model (HMM) to weight the fields and finally use convolutional neural network (CNN) to classify the RCS messages. In the HMM step, the text fields are treated differently. The short texts of these fields are represented as feature weight sequences extracted by a feature extraction algorithm based on a probability density function. Then, the proposed HMM learns the weight sequence and produces a proper weight for each short text. Other text fields with fewer words are also weighted by the feature extraction algorithm. In the CNN step, all these feature weights first construct the RCS message matrix. The matrices of the training RCS messages are used as the CNN model inputs for learning and the matrices of testing messages are used as the trained CNN model inputs for RCS message property prediction. Four optimization technologies are introduced into the CNN classification process. Promising experiment results are achieved on the real industrial data.

Keywords: rich communication suite (RCS); 5G; anti-spam; hidden Markov model (HMM); convolutional neural network (CNN)

1. Introduction

Rich communication suite (RCS) messages, known as the evolution of Short Message Service (SMS) messages, are transmitted between mobile devices built with RCS features through 5G communication infrastructure. It is expected to expedite the 5G ecosystem by way of its RCS business messaging feature, known as A2P (Application to Person) messaging, to proper palm devices for a secure, multimedia and interactive user experience. Currently, according to the 5G Message White Paper [1], many telecommunications providers have released their 5G RCS messaging solutions and numerous RCS build-in applets are under development.

Fundamentally, the RCS technology is a comprehensive upgrade of Short Message Service in terms of data, functions, features, experiences and services. An RCS message is a rich media message that is not only full of various types of data, such as texts, audios, pictures and videos, but also full of comprehensive functions, including group chatting, map navigation, calendar, e-Payment systems and applets. The user experience is greatly improved by phone number entry, security and easy accessibility. The faster reading rate, feedback rate and action rate enhance user loyalty and stickiness, which benefit value-added services and e-commercial cooperation.
In fact, it is foreseeable that RCS messaging will create a new ecosystem of 5G technologies, and apparently a great deal of revenue is growing dramatically. A study released by the Global System for Mobile Communications Association (GSMA) in 2019 shows that 1.4 billion 5G devices will account for 15% of the total mobile industry globally by the year 2025 [2]. According to the white paper released by the China Academy for Information and Communications Technology (CAICT) [3], 5G devices will increase to 816 million in China by the year 2025, which will account for 48% of mobile devices nationally. The number will continue to increase to 1.5 billion and account for 85% by the year 2030. In addition, by the year 2025, the volume of RCS messages is predicted to reach 3501.2 billion and its revenue will increase to 53 billion dollars in China.

However, in every challenge lies a greater opportunity. Although spam RCS messages will most probably appear in the age of 5G, the companies providing 5G telecommunication services have established research projects to filter spam RCS messages. Although the comprehensive RCS message data led to the complex filtering method, hundreds of fields of the data are adequate and benefit the filtering accuracy. In this research, a combination method of a hidden Markov model and convolutional neural networks is proposed to filter RCS messages based on the text data of RCS messages. We first choose 210 text fields possibly containing spam information from RCS interface specifications, label them ‘ham’ (not spam) and ‘spam’ (illegal text) for short text data fields, then interpret this information with proper spam-distinguishable weights with the hidden Markov model (HMM) and finally identify spam RCS messages with the convolutional neural network (CNN) method.

The main contributions of this research are threefold:

• This paper is the first time a method for RCS message filtering has been proposed.
• The proposed method is a combination of HMM and CNN methods and produces the RCS message property based on 210 text fields where spam information may exist.
• The proposed combination method achieves promising results.

The rest of the paper is organized as follows: Section 2 discusses related research, such as spam filtering methods and machine learning models. Section 3 proposes the combination method of HMM and CNN models. In detail, Section 3 first points out the 210 text fields and then discusses the HMM responsible for weighting and labeling short text with ham or spam. In addition, it describes how the feature extraction algorithm is implemented to interpret words in text fields to proper weights for classification and how the RCS message matrix is constructed from the weights for the deep learning CNN method to classify RCS messages. Section 4 discusses the experiment and its results. Section 5 concludes the main contribution of this work and outlines our future work.

2. Related Work

To the best of our knowledge, this study is a forerunner of spam RCS message filtering research, which is related to classification, opinion mining [4] and sentiment analysis [5] based on short text data. Several machine learning methods and models have been applied in this field, such as Naïve Bayes [6], support vector machine (SVM) [7], vector space model (VSM) [8], decision tree [9] and k-nearest neighbor [10]. Recently, neural networks approaches [11], including deep learning models [12], have become popular in this field.

All of these models are based on the Bag-of-words (BoW) assumption, which assumes words are independent from each other and ignores word order. For example, VSM and SVM models first represent features from a training text with proper weights in their feature extraction step, then rearrange these weights in the word–document matrix and perform subsequent learning algorithms in their classification step. These models make a great contribution to long text classification. However, difficulties and challenges emerge when applying these models to short text classification with the same feature extraction and classification steps. These difficulties originate from the information shortage of short texts and the sparsity of the word–document matrix [13]. As summarized in the review in [10], the four attributes of short texts hinder the effectiveness of these classical models. Their shortness results in the deficiency of text representation. Their diversity of topic
and expression leads to the sparsity of the feature weight matrix. Their idiosyncratic language causes a great variety of synonyms and even obfuscated words. Furthermore, the instancy means that millions of short texts are produced daily and they require a high filtering speed.

In order to overcome these difficulties, methods and models have been put forward to tackle these short text challenges. Rao et al. [11] introduced lexical analysis, phrase structure and attention mechanisms for classifying the sentiment of Chinese short financial texts. Lexical analysis and phrase structure are overlooked in BoW assumption based models. To overcome the shortness issue, researchers rely on presenting a novel term weighting scheme [8,10]. To tackle the feature–sparsity issue, Zhang et al. [14] proposed a non-negative matrix factorization feature expansion (NMFFE) approach by expanding short text features. Pang et al. [15] enriched the number of features by generating term groups existing in the same context, and then proposed two basic supervised topic models for associating sentiments with topics. In addition, Xu et al. [16] considered that word expansion methods might bring general concepts and noise into text representation and proposed a neural network called dual embeddings convolutional neural network (DE-CNN), which deploys an attention layer to extract and represent context-relevant concepts, then uses a convolutional neural network for short text classification. Other sparsity research includes online biterm topic models (BTM) [17], bag of biterms modeling (BBM) [18], tax2vec [19] and mixtures of Dirichlet–multinomial distributions [20]. To identify the obfuscated words in short text, Enamoto et al. [21] put forward a generic framework for multilingual short text classification based on CNN. Researchers have also focused on character-level analysis to remove the side effects of informal words. Hao et al. [22] proposed a mutual-attention CNN to concatenate word and character-level features by multiplying them into a matrix, and then stack the matrix as 3-dimensional tensors. To address the noisy and unstructured short text issues, Torres et al. [23] presented a recommending system by proposing a sequence-to-sequence neural networks recommendation model, which represents user profiles and the conversational context. To increase short text classification speed, the supervised topic models presented by Pang et al. [15] also include an accelerated algorithm.

Recently, sequential models, such as hidden Markov models, have been introduced in short text classification [4,24,25]. These HMMs treat word features of short text as observation states and categories as hidden states. They process short text in a sequential way. The HMM methods focus on learning the transmission of hidden states and the relationship between hidden states and observations. The HMM learning algorithm updates the transmission matrix and the emission matrix, which represent the transmission probabilities and the relationship. These methods do not inherit the BoW assumption or form a feature matrix. Therefore, they artfully walk around the difficulties of dealing with a short text. Kang et al. [4] proposed an opinion mining method based on text-based hidden Markov models (TextHMMs) for text classification. Xia and Chen [24] proposed a hidden Markov model for SMS filtering and then innovated it [25]. The HMM achieves a high accuracy comparable with deep learning models and high training and filtering speeds suitable for the filtering industry.

The RCS message filtering issue is far more complex and complicated. The 210 text fields in RCS messages that are being treated in this research include card title, card content, card menu button text, and so forth. All this information is critical when judging RCS messages as ham or spam. In this research, the HMM is first implemented to weight short texts, including preprocessing, feature extraction, training and decoding steps. Then, the feature extraction method is also applied to transfer the text fields with fewer words into proper weights. Finally, the CNN performs supervised learning of the training RCS message matrices formed by the weights and then produces the property for each testing RCS message.
3. 5G RCS Message Filtering Models and Methods

A 5G RCS message has a complex data structure and some parts of the data are redundant. Such data cannot be used directly in any machine learning model for spam filtering. In this research, we first select 210 fields of RCS message data for spam filtering based on our experience and build a training set by labeling the message and its cards. Then, the HMM is trained to transfer the short text fields into weights. Finally, the CNN model is proposed to filter the message based on the weights generated from the 210 fields and make the final judgment.

3.1. The 210 RCS Message Text Fields

According to the RCS universal profile service definition document [26], an RCS has a maximum of 15 cards and each card may have buttons, menus and other multimedia contents mainly for decoration, such as images. These data are stored in hundreds of fields and some of them may contain spam information. More precisely, spam information very possibly lies in the title, content fields of cards and the four card menus, which have menu button text, menu link domain and menu link content. The following fields are chosen for spam detection as outlined in Table 1.

Table 1. 210 Fields chosen from 5G message specification.

| Chosen Fields       | Menu 1 | ... | Menu 4 | Card Label |
|---------------------|--------|-----|--------|------------|
| Card1 Title         | Content| BT  | LD     | LC         | BT         | LD | LC     | ham or spam |
| Card2 Title         | Content| BT  | LD     | LC         | BT         | LD | LC     | ham or spam |
| ...                 | ...    | ... | ...    | ...        | ...        | ... | ...    | ...         |
| Card15 Title        | Content| BT  | LD     | LC         | BT         | LD | LC     | ham or spam |
| Message Label       |        |     |        | ham or spam |

BT, LD and LC are abbreviations for button text, link domain and link content respectively.

Based on our spam filtering experience, spam information most likely appears explicitly in the title or content of message cards for direct exposure. Sometimes, spam information hides behind the buttons of card menus. Therefore, the information retrieved by card menu buttons is also considered. Because each message card has 14 chosen fields, an RCS message with 15 cards in total has 210 chosen fields. To build a training RCS message set, we manually label each RCS message and its every card.

3.2. The HMM for RCS Message Short Text Fields Weighting

The chosen fields are of different kinds. The card content and the menu link content are short texts. Card title and button texts are phrases and the menu link domain is a plain text string. Therefore, it is necessary to interpret these various types of data into proper weights with different methods. In addition, each text field is processed individually. The rest of this section mainly discusses the HMM method for short text fields weighting. The weighting method for the fields with fewer words is discussed in the last subsection.

3.2.1. Short Text Preprocessing

The preprocessing step includes tokenization and stop words removal. Formal Natural languages always have distinct text strings that have specific meanings. In most cases, these distinct text strings are words. Tokenization finds all these distinct words among all texts in a given corpus. In addition, stop words refer to the redundant tokens that often stand for positioning without much meaning, such as a, an, the, of, did, which, and so forth, in English. Stop words are always programmed to be ignored based on the stop words corpus.
In this paper, words are the minimal processing metadata. Short texts are first split into words as tokens. The split process is simpler for alphabetic languages because blanks separate words naturally. Since our RCS messages for training and testing are written in Chinese, the texts are first segmented based on proven technology. The RCS message data used in this research are derived from our cooperated company and are written in Chinese. Since Chinese words only have one form, a common preprocessing step, morphemization, is omitted. Then, meaningless tokens, which are the stop words and punctuation, are removed to reduce noise in the training data. After preprocessing, each short text is transformed into a meaningful words sequence with their original order.

3.2.2. Feature Extraction for Training Short Texts

It is time consuming to directly process these distinct tokens by comparing them as text strings. Therefore, it is better to extract features from these words and substitute them with their proper weights representing their ability to distinguish categories. In addition, each RCS message in the training set has a message label and each card has a card label. These labels indicate two categories: ham or spam. The feature extraction step processes each short text field in each card, that is, the content field of card 1. Words in each short text field are measured based on the difference of the word distribution across each category.

In detail, to weight the words in each short text field, we statistically calculated the frequency of these words in ham and spam cards separately and the probability density functions (PDF) of each word on each category, respectively. Then, the difference between a word PDF value in the ham or spam category is used as the weight of the word after normalization. In this way, a word with higher positive weights indicates that it will most likely appear in a ham short text. Correspondingly, a word with lower negative weights means that it possibly occurs in a spam short text. Apparently, the weight is able to distinguish between the two categories.

Finally, the words in sequences are all substituted by their weight and then the words sequences become the weight sequences.

3.2.3. HMM Training with the Observation Sequence and the Hidden State Sequence

The HMMs are built for each short text field. While training, the HMM learns from the observation and hidden state pairs in the observation sequence and the hidden state sequence. Therefore, the proper hidden states should be labeled to each observations. In this case, the observation states are the word weights calculated based on words in the training set and the hidden states are labeled according to the sign of word weights. Observations with positive word weights are labeled as the ham state and, accordingly, observations with negative weights are labeled as the spam state. While training, the pairs of observation and hidden state feed the Baum–Welch algorithm (BM algorithm) [27]. It starts with initial parameters and repeats the expectation step (E-step) and the maximization step (M-step) until convergence, to find the optimal HMM parameters, such as the emission probability matrix and the state transition probability matrix. For efficient batch training, the observation sequences and hidden state sequences are connected together to feed the BM algorithm one after another.

In terms of HMM, the HMM mentioned above $\lambda$ can be defined as a 3-tuple:

$$\lambda = (\pi, A, B),$$  \hspace{1cm} (1)

where $\pi$ is the initial state distribution, $A$ is the state transition probability matrix and $B$ is the emission probability matrix.

$\pi$ is initialized as $\pi_0 = [0.5, 0.5]$, which refers to the opportunity for starting from the spam state or the ham state.

$A$ indicates the transmission probability between the hidden states. In this case, because the ham and spam categories are used for short text field labeling, the HMM has two states, which are represented as $S = \{s_1, s_2\} = \{spam, ham\}$. Therefore, $A$ is
represented as $A = (a_{ij})_{2 \times 2}$, where $a_{ij} = P(q_{t+1} = s_j | q_t = s_i)$, for $i, j = 1, 2$ and $\sum_{j=1}^{2} a_{ij} = 1$, for $i = 1, 2$. Before training, $A$ is initialized as $A_0 = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix}$, supposing each state has the same opportunity to transfer to another including itself.

$B$ determines the probabilities of observations emitted by hidden states respectively. The matrix $B_0$ with initial values is determined by the PDF parameter values of the spam and ham observations obtained in Section 3.2.2.

While training, the HMM $\lambda$ starts with its initial parameters $(\pi_0, A_0, B_0)$. Then, it is optimized by the BM algorithm while being fed the training observation sequences and hidden state sequences. Finally, the optimal HMM parameters are derived. The HMMs for each short text field follow the same training process.

The process of HMM training is shown in Figure 1.

![Figure 1. HMM training.](image)

3.2.4. HMM Decoding with the Observation Sequences of Testing Set

When testing, the trained HMM of each short text field processes each observation sequence in the training set and predicts the optimal hidden states. In detail, as with the short text in the training set, each testing short text is first preprocessed, including segmentation, stop words and punctuation removal, and becomes a meaningful words sequence. Then, each word in the testing words sequence is substituted by the weight of the same word in the training set. If the word does not exist in the training set, the weight value 0 is applied. Then, these testing word weight sequences feed the trained HMM and the corresponding optimal hidden state sequence is decoded by the Viterbi algorithm [27].

The method in paper [25] labels the property of a testing short text based on majority rule. That is, a short text is labeled ham when the decoded hidden states have more ham than spam and vice versa. In this paper, a proper membership degree is calculated for each testing short text based on the difference of the proportion of ham or spam states in all decoded states. For example, if a trained HMM decodes six ham states and four spam states for a testing short message, its membership degree should be $-0.2$, indicating spam category membership.

The process of HMM decoding is shown in Figure 2.

![Figure 2. HMM decoding.](image)
3.2.5. Weighting for Other Texts with Fewer Words

Except for the short texts fields discussed above, the other text fields only contain texts with fewer words, such as card title, menu button text and menu link domains. Many of these fields even have only one word. Therefore, it is unnecessary to perform the HMM method on these simple texts. In this research, these text fields are also treated individually.

The texts in each text field of all RCS messages’ cards are first segmented for tokenization. Then, each distinct word is weighted separately according to the difference of the word PDF in the ham or spam categories in the same way as described in Section 3.2.2. Then, the weight of the field of a certain RCS message equals the average weight of the words in the field.

Finally, the weight values of the 210 chosen fields are mapped to $[0, 100]$ for CNN learning.

3.3. The CNN Model for RCS Message Filtering

In this part, the CNN model and its methodology is discussed, including hidden features generation and ham and spam categories classification.

3.3.1. Creation of Feature Matrix

A total of 210 fields are chosen from the RCS message fields and are weighted to represent the RCS message’s category likelihood. The weights of the $i^{th}$ card construct a card vector which can be represented as

$$\text{card}_i = (w_{i1}, w_{i2}, w_{i3}, \cdots, w_{id}),$$

where for $1 \leq i \leq m, 1 \leq j \leq d$, $d$ is the vector dimension ($d = 14$) and $m$ is the total number of cards ($m = 15$). The 15 card vectors of an RCS message are concatenated as

$$M = \text{card}_1 \cdot \text{card}_2 \cdot \cdots \cdot \text{card}_m.$$ 

In general, an RCS message can be represented as the message matrix $M \in \mathbb{R}^{m \times d}$:

$$M = \begin{bmatrix}
    w_{11} & w_{12} & w_{13} & \cdots & w_{1d} \\
    w_{21} & w_{22} & w_{23} & \cdots & w_{2d} \\
    \vdots & \vdots & \vdots & \ddots & \vdots \\
    w_{m1} & w_{m2} & w_{m3} & \cdots & w_{md}
\end{bmatrix}$$

3.3.2. 2D Convolution for Relevant Features Extraction

The convolution operation is applied to the RCS message matrix $M \in \mathbb{R}^{m \times d}$ and kernel $F, F \in \mathbb{R}^{d \times k}$, where $k$ is the kernel region. In this case, the CNN model has four convolutional layers, in which the first convolutional layer has 16 filters, the second one has 32 filters, the third one contains 64 filters and the fourth one contains 128 filters. For each convolutional layer, different square filter kernel sizes are chosen, that is, $2 \times 2, 3 \times 3, 4 \times 4$ and $5 \times 5$. For each CNN, filters are involved to yield a matrix of relevant features. In this case, since the RCS service has just started, the current RCS messages usually have a few cards filled with contents and other cards remain blank. To emphasize the features on the matrix edge, a padding value, represented by $p$, is applied to expand the matrix $M$ and the yielded feature matrix for each CNN. The stride value, represented by $s$, is also specified with 1.

Therefore, in the process of convolution operation, the kernel region first slides over the matrix $M$, strides one line for each step and produces the feature matrix with size $16 \times 15$, calculated by the following formula when $padding = 1$, $kernelsize = 2$ and $stride = 1$:

$$out = (((in + 2 \times p - k) / s) + 1). \quad (2)$$

Then, the features go through an activation function. Rectified linear unit (ReLU) is applied.

$$f_{\text{ReLU}}(x) = \max(0, x). \quad (3)$$
It returns the input value if it is positive and returns 0 for negative inputs.

3.3.3. MaxPooling to Avoid over Fitting

After the convolution operation process, each RCS message matrix $M$ is transferred into a relevant feature matrix, represented by $K$, which has only non-negative values. These relevant features, also known as hidden features, may not share the same importance and may contain redundant information. To avoid over-fitting and to reduce this high-level vector representation, a max pooling layer is present after the convolutional layer; a method called pooling is also applied to extract the most important one. Among the pooling functions, such as max-pooling, min-pooling and average pooling, max-pooling is found to yield the best results. The max-pooling function returns the maximum value of a given window’s size, represented by $k_{\text{maxpooling}}$. The window slides over the feature matrix to produce an output matrix, which consists of the maximum values of each window, and the weak activation information generated by the ReLU function is removed.

3.3.4. Optimizations for CNN

In order to increase filtering accuracy, four optimization methods are applied.

- **Batch normalization**—Batch normalization is used to standardize the extracted relevant features between CNNs. It not only speeds up learning but also decreases the internal covariate shift of a CNN.
- **Dropout**—Before the last fully connected layer, a dropout function runs on the output matrix of CNNs to reduce computational complexity and redundant information. RCS message filtering is a dropout-suitable scenario in which the card data of most messages are insufficient.
- **Optimizer**—Adam optimizer is applied to optimize parameters, in this case to increase accuracy.
- **Dense**—Dense, also known as the fully connected layer, runs an activation function to determine the property of an RCS message. Many activation functions, such as Sigmoid, Softmax and Linear, are tested. The Linear function is the present setting as it produces the best results.

The proposed CNN is shown in Figure 3.

Figure 3. Proposed CNN for RCS filtering.

4. Experiments and Discussion

The experiments and results are outlined and discussed in this section. In order to simplify the RCS filtering process and speed it up for industrial deployment, non-textual data, such as images, video and audio, are omitted in the proposed models and methods. This section uses real industrial data to find the filtering results based on textual data.
4.1. Data Analysis

The dataset was derived from the RCS message database of our cooperated company. Since the RCS service starts in China, most of the RCS messages do not contain much information, and look like traditional SMS messages. A total of 440 RCS messages with rich information were manually selected for this experiment. The contents of spam RCS messages include lottery, money or tax defrauding, and advertisements with illegal content. The legal RCS messages refer to government notifications, express delivery notifications, bank notifications and legal advertisements. The data of the mentioned 210 fields were extracted. Then, these messages were manually labeled, including each message and its cards. The statistics of these messages are outlined in Table 2.

|                 | Ham       | Spam      | Total |
|-----------------|-----------|-----------|-------|
| **Total number**| 381       | 59        | 440   |
| **Percentage**  | 86.6%     | 13.4%     | 100%  |

In addition, we randomly chose two-thirds of the messages to generate the training set in the experiment. The rest of the messages form the testing set as shown in Table 3.

|                          | Training Set | Testing Set | Total |
|--------------------------|--------------|-------------|-------|
| **Total number**         | 293          | 147         | 440   |
| **Percentage**           | 66.6%        | 33.4%       | 100%  |

4.2. Experiments and Results

The experiment code was programmed in Python 3.7. The import packages included pyhanlp for Chinese segmentation, pomegranate for HMM modeling, pytorch for CNN modeling, collections for statistics and matplotlib for figures. The program ran on a laptop with Intel Core i7-7820 CPU and 16 GB memory. The performance results were measured with well-known metrics, including precision (P), recall (R), F-measure (F1), accuracy (A) and area under the curve (AUC) [28].

In the experiment, the accuracy consistently increases during the first six epochs. Then, the accuracy tends to be stable. The average accuracy on these stable epochs turns out to be 96.5%. The confusion matrix of the experiment’s results is shown in Table 4. Table 5 outlines the evaluation metrics.

| Model                  | Actual | Predicted | Predicted % | AUC  |
|------------------------|--------|-----------|-------------|------|
| The proposed method    | Spam   | 3         | 20          | 13.0% 86.9% 0.965 |
|                        | Ham    | 122       | 2           | 98.4% 1.6%  |

Table 5. Evaluation results.

| Model                  | Class | Accuracy | Precision | Recall | F-Measure |
|------------------------|-------|----------|-----------|--------|-----------|
| The proposed method    | Ham   | 0.965    | 0.983     | 0.976  | 0.979     |
|                        | Spam  |          | 0.870     | 0.909  | 0.889     |
4.3. Discussions

This experiment was performed on an industrial RCS messages data and its results turn out to be promising. The average accuracy achieved was 96.5% when two thirds of the RCS messages were chosen for training. The proposed models and methods perform better on ham messages by correctly predicting 122 out of the total 124 testing ham messages. The precision for ham messages hits 98.4%. On the other hand, the experiment produces acceptable results for spam messages. This is due to the shortage of spam RCS messages. Since the RCS service is just at its initial stage, spam messages are not easy to collect due to their limited quantity. Insufficient training limits the accuracy of spam messages. With the development of the RCS service and its economy ecosystem, we believe the proposed models and methods will produce better and more reliable results.

5. Conclusions and Future Work

To filter spam RCS messages, this research proposed the method of combining the hidden Markov model and convolutional neural networks. This research first investigated the fields of RCS message data and chose 210 text fields that possibly contain spam information. Due to the complexity of RCS, the texts in these fields may be short texts or just one or two words. To weight all these fields properly, the HMM method was used for the short texts. They were first preprocessed to become word sequences, including the tokenization and the stop words removal steps. Then, the distinct words were weighted by the feature extraction algorithm based on the probability density function. The word sequences were transformed into the observation sequences by substituting their words with their weights. The hidden state sequences were generated according to the sign of each weight in the weight sequence. Such pairs of observation and the hidden states of the sequences fed the proposed HMM for training. While predicting, each testing short text was first preprocessed as well and was transformed into an observation sequence by substituting its words with the weights of the same words in the training set. Then, the trained HMM decoded the optimal hidden state sequence, which was used to weight the short text. The HMM method was performed on all short text fields. For other fields with fewer words, they were preprocessed and weighted by the feature extraction algorithm. When all the 210 fields are weighted, the RCS message matrix was formed. In the CNN step, the matrices of the training RCS messages fed the proposed CNN for deep learning. Then, the trained CNN produced the property of each testing RCS message based on its matrix. The RCS messages for the experiment were derived from the production environment of our cooperated company. The experiment showed promising results, such as 96.5% average accuracy, indicating prosperous future implementation of this study in the RCS industry.

In the near future, the RCS service is expected to spread across the world to provide a secure, convenient and interactive user experience. Sufficient RCS messages will be used for filtering research and development. We will focus on improving accuracy and filtering speed by introducing HMM variants, hybrid models and other machine learning methods into this research field. We will also take advantage of whole RCS data, including images, videos, and so forth, for filtering.

Author Contributions: B.G. and W.Z. conceived the idea and developed the algorithm; B.G. developed the experimental software; B.G. and W.Z. performed the data analysis. B.G. prepared the manuscript; B.G. and B.G. wrote and made revisions to the paper. Both authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Restrictions apply to the availability of these data. Data was obtained from Dahan Tricom and are available at www.dahantc.com (accessed on 8 July 2021) with the permission of Dahan Tricom.

Conflicts of Interest: The authors declare no conflict of interest.
References

1. ChinaMobile; ChinaTelecom; ChinaUnicom. 5G Messaging White Paper. 2020; pp. 1–12. Available online: https://www.gsma.com/futurenetworks/wp-content/uploads/2020/04/5G-Messaging-White-Paper-EN.pdf (accessed on 8 July 2021).

2. GSMA. The Mobile Economy. 2020; pp. 2–62. Available online: https://www.gsma.com/mobileeconomy/wp-content/uploads/2020/03/GSMA_MobileEconomy2020_Global.pdf (accessed on 8 July 2021).

3. CAICT. White Paper on China’s 5G Development and Its Economic and Social Impacts. China Acad. Inf. Commun. Technol. 2020, 12, 1–46.

4. Kang, M.; Ahn, J.; Lee, K. Opinion mining using ensemble text hidden Markov models for text classification. Expert Syst. Appl. 2018, 94, 218–227. [CrossRef]

5. Xie, J.; Chen, B.; Gu, X.; Liang, F.; Xu, X. Self-Attention-Based BiLSTM Model for Short Text Fine-Grained Sentiment Classification. IEEE Access 2019, 7, 180558–180570. [CrossRef]

6. Hadi, W.; Al-Radaideh, Q.A.; Alhawari, S. Integrating associative rule-based classification with Naïve Bayes for text classification. Appl. Soft Comput. J. 2018, 69, 344–356. [CrossRef]

7. Liu, Z.; Kan, H.; Zhang, T.; Li, Y. DUKMSVM: A framework of deep uniform kernel mapping support vector machine for short text classification. Appl. Sci. 2020, 10, 2348. [CrossRef]

8. Samant, S.S.; Bhanu Murthy, N.L.; Malapati, A. Improving Term Weighting Schemes for Short Text Classification in Vector Space Model. IEEE Access 2019, 7, 166578–166592. [CrossRef]

9. Gashti, M.Z. Detection of Spam Email by Combining Harmony Search Algorithm and Decision Tree. Eng. Technol. Appl. Sci. Res. 2017, 7, 1713–1718. [CrossRef]

10. Alsmadi, I.; Hoon, G.K. Term weighting scheme for short-text classification: Twitter corpuses. Neural Comput. Appl. 2019, 31, 3819–3831. [CrossRef]

11. Rao, D.; Huang, S.; Jiang, Z.; Deverajan, G.G.; Patan, R. A dual deep neural network with phrase structure and attention mechanism for sentiment analysis. Neural Comput. Appl. 2021, 6. [CrossRef]

12. Yu, S.; Liu, D.; Zhu, W.; Zhang, Y.; Zhao, S. Attention-based LSTM, GRU and CNN for short text classification. J. Intell. Fuzzy Syst. 2020, 39, 333–340. [CrossRef]

13. Barushka, A.; Hajek, P. Spam filtering using integrated distribution-based balancing approach and regularized deep neural networks. Appl. Intell. 2018, 48, 3538–3556. [CrossRef]

14. Zhang, L.; Jiang, W.; Zhao, Z. Short-text feature expansion and classification based on nonnegative matrix factorization. Int. J. Intell. Syst. 2020, 1–15. [CrossRef]

15. Pang, J.; Rao, Y.; Xie, H.; Wang, X.; Wang, F.L.; Wong, T.L.; Li, Q. Fast Supervised Topic Models for Short Text Emotion Detection. IEEE Trans. Cybern. 2021, 51, 815–828. [CrossRef] [PubMed]

16. Xu, J.; Cai, Y.; Wu, X.; Lei, X.; Huang, Q.; fung Leung, H.; Li, Q. Incorporating context-relevant concepts into convolutional neural networks for short text classification. Neurocomputing 2020, 386, 42–53. [CrossRef]

17. Hu, X.; Wang, H.; Li, P. Online Bitemr Topic Model based short text stream classification using short text expansion and concept drifting detection. Pattern Recognit. Lett. 2018, 116, 187–194. [CrossRef]

18. Tuan, A.P.; Tran, B.; Nguyen, T.H.; Van, L.N.; Than, K. Bag of biterms modeling for short texts. Knowl. Inf. Syst. 2020, 62, 4055–4090. [CrossRef]

19. Škrilj, B.; Martinc, M.; Kralj, J.; Lavrač, N.; Pollak, S. tax2vec: Constructing Interpretable Features from Taxonomies for Short Text Classification. Comput. Speech Lang. 2020, 65. [CrossRef]

20. Anderlucci, L.; Viroli, C. Mixtures of Dirichlet-Multinomial distributions for supervised and unsupervised classification of short text data. Adv. Data Anal. Classif. 2020, 14, 759–770. [CrossRef]

21. Enamoto, L.; Weigang, L.; Filho, G.P. Generic framework for multilingual short text categorization using convolutional neural network. Multimed. Tools Appl. 2021. [CrossRef]

22. Hao, M.; Xu, B.; Liang, J.Y.; Zhang, B.W.; Yin, X.C. Chinese short text classification with mutual-attention convolutional neural networks. ACM Trans. Asian Low-Resour. Lang. Inf. Process. 2020, 19, 1–13. [CrossRef]

23. Torres, J.; Vaca, C.; Terán, L.; Abad, C.L. Seq2Seq models for recommending short text conversations. Expert Syst. Appl. 2020, 150, 113270. [CrossRef]

24. Xia, T.; Chen, X. A discrete hidden Markov model for SMS spam detection. Appl. Sci. 2020, 10, 5011. [CrossRef]

25. Xia, T.; Chen, X. A Weighted Feature Enhanced Hidden Markov Model for Spam SMS Filtering. Neurocomputing 2021, 444, 48–58. [CrossRef]

26. GSMA. Official Document RCC.71—RCS Universal Profile Service Definition Document. 2020; pp. 1–260. Available online: https://www.gsma.com/futurenetworks/wp-content/uploads/2019/10/RCC.71-v2.4.pdf (accessed on 8 July 2021).

27. Rabiner, L.R.; Juang, B.H. An Introduction to Hidden Markov Models. IEEE ASSP Mag. 1986, 3, 4–16. [CrossRef]

28. Roy, P.K.; Singh, J.P.; Banerjee, S. Deep learning to filter SMS Spam. Future Gener. Comput. Syst. 2020, 102, 524–533. [CrossRef]