PGN-LM Model and Forcing-Seq2Seq Model: Multiple automatic models of title generation for natural text using Deep Learning

Nhan To Thanh
University of Technology - VNUHCM
Ho Chi Minh city, Vietnam
1970022@hcmut.edu.vn

Thuan Nguyen Thi Hiep
University of Technology - VNUHCM
Ho Chi Minh city, Vietnam
1870365@hcmut.edu.vn

Tho Quan Thanh
University of Technology - VNUHCM
Ho Chi Minh city, Vietnam
qtttho@hcmut.edu.vn

***Abstract*—In the current era, the amount of information from the Internet in general and the electronic press in particular has increased rapidly and has extremely useful information value in all aspects of life, many popular users have posted several high-quality writings as casual blogs, notes or reviews. Some of them are even selected by editors to be published in professional venues. However, the original posts often come without titles, which are needed to be manually added by the editing teams. This task would be done automatically, with the recent advancement of AI techniques, especially deep learning. Even though *auto-title* can be considered as a specific case of text summarization, this job poses some major different requirements. Basically, a title is generally short but it needs to capture major content while still maintaining the writing style of the original document. To fulfill those constraints, we introduce PGN-LM Model, an architecture evolved from the Pointer Generator Network, with the ability to solve Out-of-Vocabulary problems that traditional Seq2Seq models cannot handle, and at the same time combined with language modeling techniques. In addition, we also introduce a model called Forcing-Seq2Seq Model, an enhanced Seq2Seq architecture, in which the classical TF-IDF scores are incorporated with Named Entity Recognition method to identify the major keywords of the original texts. To enforce the appearance of those keywords in the generated titles, the specific Teacher Forcing mechanism combined with the language model technique are employed. We have tested our approaches with real datasets and obtained promising initial results, on both metrics of machine and human perspectives.**

***Index Terms*—Sequence to sequence, Attention Mechanism, Named Entity Recognition, TF-IDF, Language Model, Teacher Forcing, Pointer Generator Network.**

I. Introduction

Nowadays, the world has been witnessing the rapid development of social media, where people find a convenient channel to express their ideas, opinions and feelings. There are several blogs, posts and reviews made by popular users that are very interesting and attract much attention from the audience. Those writings are many times selected by editors to be published as high quality articles. However, as non-professional writers, users do not often make titles for their writing and instead the editors will handle this job manually, which costs relatively much time and energy. With the remarkable emerging of *Artificial Intelligence* (AI) techniques, those titles can be considered to be generated automatically in short time and should be sufficiently reasonable.

At present, there is no specific research project exactly addressing the issue of *auto-title* of a textual document though the demand is real. Generally, the task of *auto-title* can be considered as a specific case of *text summarization*. According to [1], a summary is defined as a document created from one or more other documents, which conveys important information in the original text(s) and is no longer than half of the original text(s), and usually, less meaningful and text summarization is the task of creating a concise and fluent summary, while still being able to retain the main content and overall meaning of the original text. In general, there are two methods to solve the text summary problem, including *extraction* and *abstraction* summary methods. These two methods of summarization have been explored and studied extensively over the past several decades. [1] [2].

The extraction summary methods identify important parts (sentences, paragraphs, etc.) of the text and produce them verbatim. With this method, the importance of each part of a sentence is decided based on the statistical and linguistic properties of the sentence [3]. In contrast, abstract summary methods will try to understand the context of the full text, and then generate a new summary based on the style and content of the original text with completely new words. This is much more difficult as it involves manually rewriting new sentences. It also requires natural language generation techniques. Obviously, this approach is more similar to the way that human works.

For the purpose of *auto-title*, we focus on researching abstractive summary method. This method shows the strength of creating a new concise title, which shows the main ideas of the original text with flexible words. In [4], the authors have introduced a new model called Document-context *Sequence-to-Sequence* (Seq2Seq), for both abstraction and extraction summary methods. Then, the authors demonstrate that constructing abstract models for content-based text using Seq2Seq model with RNN algorithm has achieved good results for both short and long text. In [5], the authors recommend a *deep-recurrent-generative-decoder* (DRGD) to improve abstraction

summary performance. This model is composed of an encode-decode framework, is oriented to the Seq2Seq architecture, and is equipped with a latent structure modeling component. The output of this model is generated based on both the latent variable and the defined state, achieving improvements over modern methods. In [6], the authors present *Relevance-Sensitivity-Attention for Query-Focus-Summary* (RSA-QFS), a method of combining relevance-relevance related to the Seq2Seq model with *attention-mechanism*, for the purpose of abstracting abstractions for QFS tasks. Besides, this method will be compared with modern extraction methods and the accuracy is greatly improved for the *Recall-Oriented Understudy for Gisting Assessment* (ROUGE) score. In [7], the authors introduce a new architecture to augment the standard Seq2Seq architecture with the attention model. First, they use a pointer-generator network that can copy words from the source text through pointers, supporting accurate information reconstruction, while retaining the ability to generate new words through generator. Second, they use a coverage mechanism to track the generated words in the summary text, thereby reducing repetition of words.

Perhaps the work that is most similar to ours is reported in [8], where the authors describe an application of an encoder-decoder recurrent neural network with *Long short-term memory* (LSTM) units, and combines with the attention mechanism to generate *summaries* from the full text of news papers. This model generates a concise summary, which is mostly valid and grammatically correct. However, as compared to text summarization, the process of title generation poses some visible different points as follows:

- Title is generally far shorter than a summarization.
- Due to its remarkably short length, a title must capture the most important keywords of the full text, generally is main *named entities* of the documents.
- Meanwhile, the title still needs to maintain the writing style of the original documents.

To tackle these issues, we adopt the abstraction-based summarization approach, which can help to capture the writing style from the original document. A common problem with text summarization problems is the Out-of-vocabulary (OOV) problem. OOV are words that are not present in the trainer, but do appear in the input data, most traditional Seq2Seq models cannot solve this problem. As a result, we introduce a novel architecture known as $PGN - LM$ Model. $PGN - LM$ is an architecture based on the idea of the $PointerGeneratorNetwork$ [7], allows copying of words in the source text, and the ability to create new words and phrases. This technique combines two methods of extraction and abstraction to process OOV words, and also improves the relevance of the original text for the summary text, use the *Language Model* technique [9] to smooth the output as well as fix some grammatical errors.

Besides, we also introduce a second new architecture called $Forcing - Seq2Seq$ Model. Basically, $Forcing - Seq2Seq$ Model is a Seq2Seq model whose *encoder* can produce proper representation from the input and *decoder* can generate a corresponding short summarization. To make the generated text adaptively meet the requirements for a title, as mentioned above, our model is trained to capture not only the semantics of the input but also the suitable sizes from the training titles. Moreover, to guide the title to capture the important keywords, we leverage the *Term Frequency-Inverse Document Frequency (TF-IDF)* [10] to identify crucial terms and particularly the *Teacher-Forcing* mechanism [11] to force the resulted titles commenced with those identified terms. Finally, we also use the *Language Model* technique to smooth the output as well as fix some grammatical errors to make the final output more coherent.

We experimented with our approaches with the real dataset of the Amazon Fine Food reviews and enjoyed very promising results, on both metrics of machine and human validations.

II. Preliminaries

A. The Seq2Seq Model and Attention Mechanism

Seq2Seq Model: The Seq2Seq model includes an LSTM encoder and decoder [12]. LSTM is widely adopted because it is very suitable for string data processing and capable of capturing the long-term dependency by overcoming the vanishing gradient problem. The LSTM encoder is composed of multiple layers of LSTM stacked on top of each other, which provides better sequence representation, reads input data with tokens, and generates a sequence of encoder hidden states h_i that encode or represent input. The LSTM decoder also consists of multiple layers of LSTM stacked on top of each other, generating each hidden state of the decoder s_t, each of which produces the output sequence as a summary.

Besides the advantages of this Seq2Seq model, there are still certain limitations that come with it. The encoder converts the entire input string into a fixed-length vector, and then the decoder predicts the output sequence based on this input sequence. This shows that the model only works well for short strings, because the decoder has to consider the entire input sequence to make a prediction. However, the encoder is unlikely to memorize long strings into a fixed-length vector. The *Attention* mechanism was introduced to overcome this problem, with the aim of predicting a word by considering only a few specific parts of the string, instead of the whole string as it is.

Attention Mechanism: In the attention mechanism, the attention distribution is calculated as a probability distribution based on the words in the source text, potentially helping the decoder to decide which word to focus on in the next word generation step [13]. The attention distribution a_t is calculated for each decoder timesteps t as:

$$e_i^t = \ v^T tanh(W_h h_i + W s_t + b_{attn}) \tag{1}$$

$$a^t = \ softmax(e^t) \tag{2}$$

In formula (1), v, W_h, W_s, b_{attn} are the parameters to learn. At each step of the decoder, the weighted attention a_i^t, which

is part of the distribution a_t for words in the source text is computed. Attention weight represents the amount of attention that needs to be given to a given source word, to produce an output word (decoder state) in the decoder. [14]. The attention distribution is used to calculate the weight sum of the encoder's hidden states, called the context vector h_t^*, which represents what was read from the source text at this step and has can be calculated as follows:

$$h_t^* = \sum_i a_i^t h_i \tag{3}$$

In this paper, we use the Seq2Seq model with attention mechanism to generate abstraction-oriented summary text [15], the details of the model are shown in Figure 1. This method will also be the baseline method for the next sections in our paper, denoted as $Base - Seq2Seq$ model.

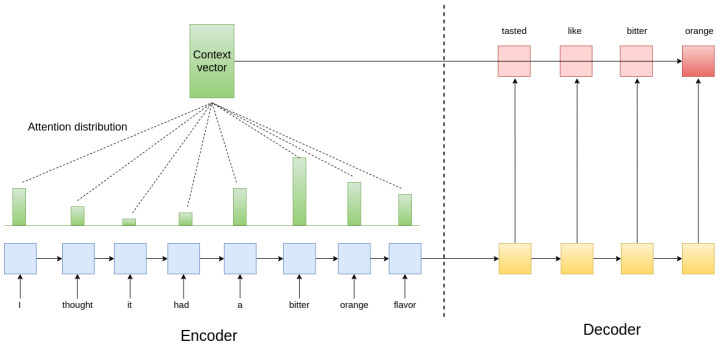

Fig. 1. Architecture of Seq2Seq model with attention mechanism

B. TF-IDF score

The TF-IDF score is one of the most commonly used terminology scoring theories in information retrieval problems. TD-IDF is a statistical metric that evaluates the relevance of a word to a document in a document collection, calculated by multiplying two metrics: the number of times a word occurs in the document, and inverse the document frequency of the word on a set of documents [16] [17]. In this paper, we use TF-IDF score with the purpose of identifying the most important terms in a full text and input this word as the first output word to decoder phase, to creating an automatic title sentence with more meaning, applied in the $Forcing - Seq2Seq$ model. The details on this step will be presented in the following sections.

C. Teacher-Forcing Mechanism

Teacher-Forcing is a strategy that aims to train recurrent neural networks, which uses ground truth as input, instead of model output from a prior time step as an input. "*Models that have recurrent connections from their outputs and then feed back can be trained with the Teacher Forcing mechanism.*" [11] Experimental results show that the training converge faster when using the *Teacher-Forcing* mechanism. At the beginning of the training process, the model's predictions are very bad. If we do not use *Teacher-Forcing*, the hidden states of the model will be updated by a series of false predictions, errors will accumulate and it is difficult for the model to learn from it [18]. However, in addition to the purpose of improving training efficiency, we have adapted the *Teacher-Forcing* mechanism for more effective automatic title creation at prediction phase. For doing so, we pass one ground truth word as the starting point for auto-tile generation. In our $Forcing - Seq2Seq$ model, the grounth truth we use for *Teacher-Forcing* is the word with the highest TF-IDF score in the full text.

D. Pointer Generator Network

Although the $Seq2Seq$ model has been able to summarize text in an abstractive way, that is, it is capable of creating automatic title sentences with many meanings and descriptions like how humans do. However, for the problem of input documents containing words that have never appeared in the vocabulary, the Seq2Seq model does not solve this problem (it will recognize UNK characters). So the $PointerGeneratorNetwork$ model will help solve these problems.

$PointerGeneratorNetwor$ can copy words from source text through pointers, aiding in accurate information reconstruction, while preserving the ability to generate new words through generator, model overview is shown as Figure 3. The calculation formula for the attention distribution a^t and the context vector h_t^* is shown in section 2.1. Besides, the *generates probability* p_{gen} in $[0, 1]$ for timestep t is calculated from context vector h_t^*, decoder state s_t and the input of the decoder x_t [7] are expressed by the following formula:

$$p_{gen} = \sigma(w_{h^*}^T h_t^* + w_s^T s_t + w_x^T x_t + b_{ptr}) \tag{4}$$

In the above formula, vector w_h, w_s, w_x and scalar b_{ptr} are the parameters to learn and σ is the sigmoid function. The coefficient P_{gen} is used as a gate to decide between creating a new vocabulary by taking probabilities from P_{vocab} or copying a word in the input text by take the probability from the attention distribution a^t. For each text, we get the following probability distribution over the extended vocabulary:

$$P(w) = p_{gen} P_{vocab}(w) + (1 - p_{gen}) \sum_{i:w_i=w} a_i^t \tag{5}$$

With the above formula, we can see that if w is a Out-of-vocabulary (OOV) word then $P_{vocab}(w)$ will be 0, similarly if w does not appear in source text then $\sum_{i:w_i=w} a_i^t$ is zero. This ability has helped create OOV words, one of the problems that most traditional models cannot handle [7].

In this paper, we apply the $PointerGeneratorNetwork$ Model to generate titles from full text, this is also the $PGN - LM$ model that we propose. Details of this section will be presented in Section 3.

III. The PGN-LM Model

Figure 2 presents the overall architecture of our $PGN-LM$ Model, which consists of the following components:

- **Text preprocessing**: This component has the function of preprocessing the input data to normalize the data,

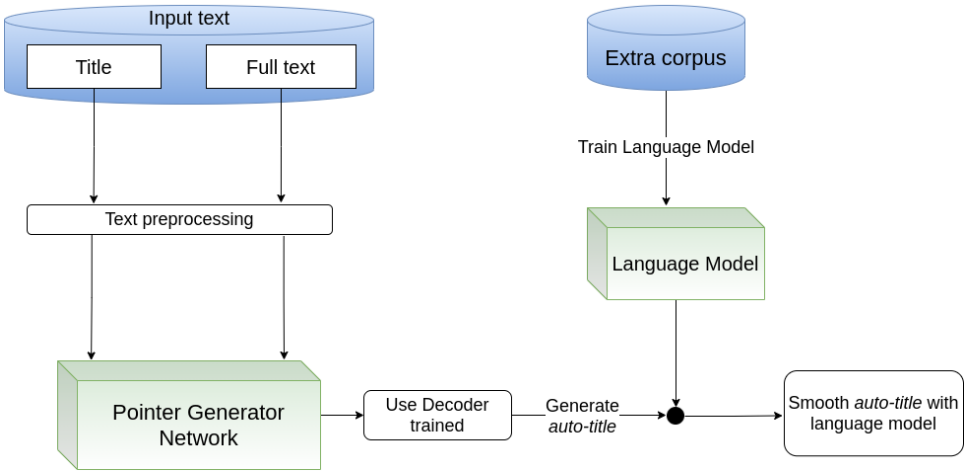

Fig. 2. Overview Architecture of PGN-LM Model

removing unnecessary information, in order to create the cleanest and most valuable input for the model.

- **Pointer Generator Network**: The model uses $PointerGeneratorNetwork$ architecture, details of this model have been mentioned in the Preliminaries section and shown in Figure 3, which is responsible for creating abstract titles, data is trained as input text, including full text and original title.
- **Language Model**: This component has the main function to fix some grammatical errors generated in the *auto-title*, make the final output more coherent. We use Word-Level *Language Model* to fix errors for the *auto-title*.

$PGN-LM$ Model uses 2 datasets to train, the first dataset contains input texts, include full texts that we need to generate title. The first goal of this dataset is to help the $PGN-LM$ Model learn how to generate titles from full text, the second goal is to learn the length from the title. In parallel with that, it also uses the second dataset, the *extra corpus* to train *Language Model*, which requires a very large and general corpus set.

A. Text preprocessing

First, the use of cluttered, uncleaned text data is a potentially disastrous problem. So we try to do basic preprocessing steps to get information from raw text. Text preprocessing includes the following steps:

- **Text cleaning**: We drop duplicates, remove missing value, unwanted symbols including punctuation, stop words, short words ,etc.
- **Text Normalization**: We convert everything to lower case and do contraction mapping.
- **Word embedding**: *Word2Vec* was created and published in 2013, by a team of researchers led by Tomas Mikolov at Google [19]. Besides, there have been other researchers who have helpfully analyzed and explained about this algorithm. Embedding vectors generated with the *word2Vec* algorithm has some major advantages over previous algorithms, such as the problem of latent semantic analysis of sentences [20]. So in this paper, we tokenizer and use *Word2Vec* to make a vector from each text. This will make the training more effective.

B. Pointer Generator Network

After the initial data preprocessing step for full text and title in order to create the cleanest and most valuable input to the model, those input will be train with $PointerGeneratorNetwork$ Model. Figure 3 shows the architecture of this model. As described in the preliminaries section, this model helps to solve two main problems. The first problem is to help create words in the *auto-title* that have closer meanings to the full text, and the second problem is to solve the problem of encountering strange words in the input text (also known as OOV). After the training is complete, we use a trained decoder to generate *auto-title* from the full text.

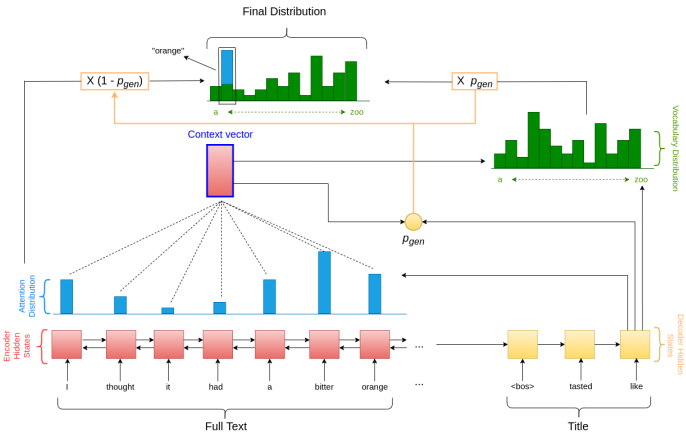

Fig. 3. Overview Architecture of Pointer Generator Network model

C. Language Model

In this component, to solve the problem of repeating words in automatically generated title in the

$PointerGeneratorNetwork$ model, we use the word-level *Language Model* to correct errors for predicted titles.

A *Language model* is considered a mathematical function, or a learning algorithm with the aim of capturing the salient features of the distribution of sequences of words in a natural language, allowing Probabilistic prediction for the next word based on previous words [9]. Among them, Ngram-based approach is the most popular technique for language modeling. This method is based on Markov's assumption that the probability of occurrence of a particular word in the string depends only on the occurrence of $n-1$ from the previous word [21] [22]. In this paper, we have trained the *Language Model* from the *extra corpus*, our *Language Model* uses the LSTM network for training, the training process is described in the Figure 4.

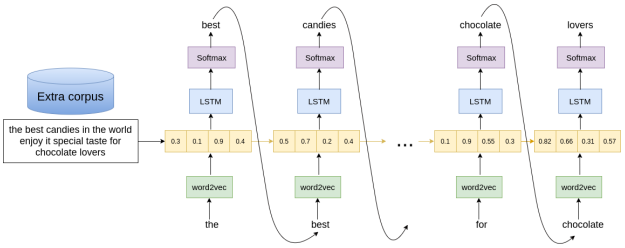

Fig. 4. Training *Language-Model*

After training, this *Language Model* is intended for error correction for generated *auto-title*. Error correction is shown in the diagram in Figure 5.

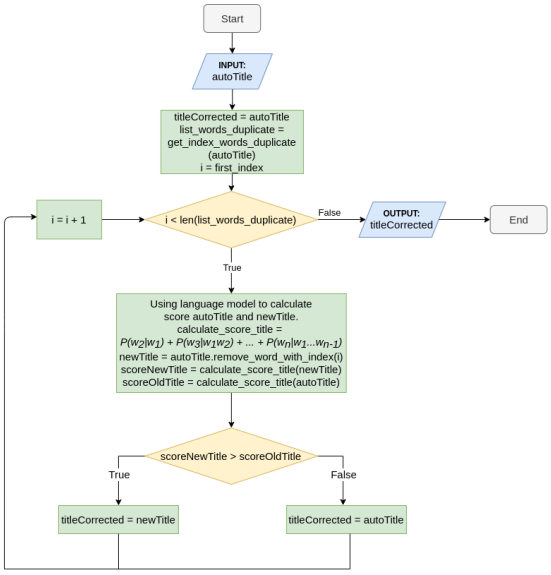

Fig. 5. Using *Language-Model* to correct errors for *auto-title*

We use the trained *Language Model* to evaluate the score for an generated title with the formula:

$$Score = P(w_2|w_1) + P(w_3|w_1w_2) + ... + P(w_n|w_1...w_{n-1}) \quad (6)$$

First, we will retrieve an indexed list of words that are repeated in the *auto-title* and remove each of these repeated words in turn, each step will evaluate the score for this *auto-title* before and after removing each word with formula (4), and finally retain the *auto-title* with higher score.

With this approach, we have virtually eliminated words that were repeated many times in the title.

D. Running Example

Figure 6 shows an example with our baseline model $Base-Seq2Seq$, as mentioned in Section Preliminaries. By the trained Seq2Seq with attention mechanism model, from full text: *"You just cannot beat bigelow products all of their teas are very very good"*, we have created the corresponding automatic title sentence: *"UNK nice tea"*. We can see that this approach is not very efficient, especially in the source text that has its own word list **bigelow**, this is a new word and is not in the vocabulary at all, so $Base-Seq2Seq$ will produce UNK character, and it is not much different from the original title.

Fig. 6. Automatic title creation with $Baseline$

Finally, Figure 7 shows the complete flow of the $PGN-LM$ Model that we propose. First, from the original full text sentence, after going through the preprocessing step, we will use the $PointerGeneratorNetwork$ to generate the corresponding title sentence. As we can see, the decoder trained will generate auto-title is *"I love bigelow tea are the best tasting earth organic"* from full text *"You just cannot beat bigelow products all of their teas are very very good"*. We can see that $PGN-LM$ is able to derive the proper noun **bigelow**, which $Base-Seq2Seq$ cannot handle. In addition, we also see that this title makes much more sense than the $Base-Seq2Seq$ model, even better than the original title.

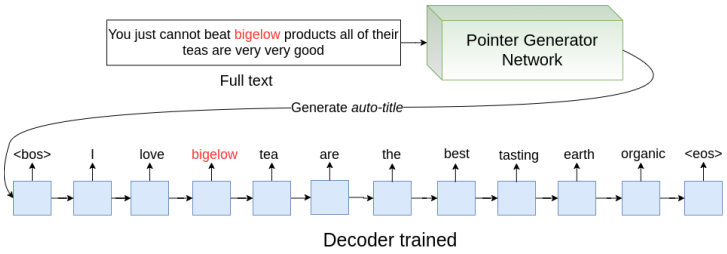

Fig. 7. Create *auto-title* with $PGN-LM$ Model

IV. The Forcing-Seq2Seq Model

Figure 8 presents the overall architecture of $Forcing-Seq2Seq$ Model, which consists of the following components:

- **Text preprocessing**: Like $PGN-LM$ Model, this component has the function of preprocessing the input data to normalize the data, removing unnecessary information, in

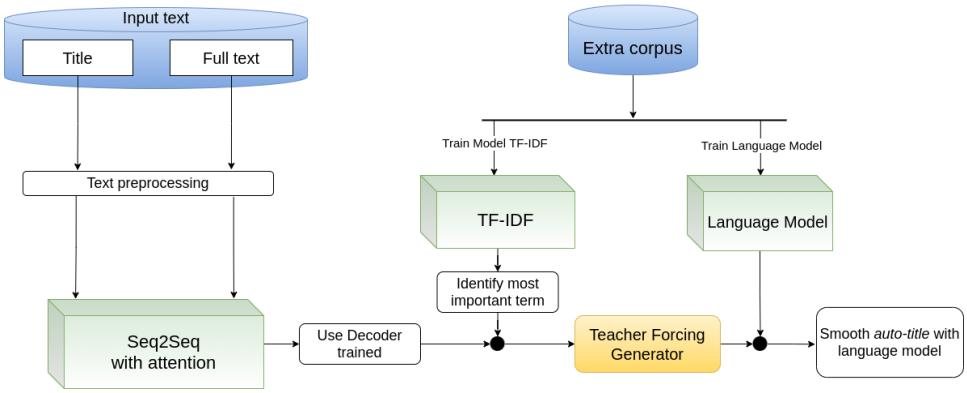

Fig. 8. Overview Architecture of Forcing-Seq2Seq model

order to create the cleanest and most valuable input for the model.

- **Model Seq2Seq with attention**: This is the $Base - Seq2Seq$ model which is responsible for creating abstract titles, training data is included full text and original title.
- **Model TF-IDF**: Used for the purpose of identifying the most important terms in a full text, use this term as the first output word for the decoding phase, in order to create an automatic title sentence with multiple more meaningful.
- **Teacher Forcing Generator**: The most important component of the $Forcing - Seq2Seq$ Model, which is responsible for creating more logical, meaningful automatic title sentences.
- **Language Model**: Like the last step of $PGN - LM$ Model, we combine the Language model to fix some grammatical errors.

Just like $PGN - LM$, $Forcing - Seq2Seq$ Model uses 2 datasets to train. The learning objectives will be to help the $Forcing - Seq2Seq$ Model learn how to create a auto title from full text, learn the length of this title word, and finally use the *Language Model* to correct grammatical errors.

The details of each components in the $Forcing - Seq2Seq$ model will be presented in the following sections.

A. Model Seq2Seq with attention

After put both full text and title through simple preprocessing steps in order to create the cleanest and most valuable input to the model, those input will be train with Seq2Seq Model with attention layer. In this stage, we consider this model as $Base - Seq2Seq$ model that we mentioned in the Preliminaries section.

B. Model TF-IDF

This model is used for the purpose of identifying the most important term in a complete document. The most important term is the word that has the highest TF-IDF score. Then $Forcing - Seq2Seq$ model use this term as the first input word for the decoding phase.

C. Teacher Forcing Generator

First, we still use the $Base - Seq2Seq$ during training as shown in Figure 1. However, in the automatic title creation step, we use TF-IDF model to increase efficiency for the title creation process during decoding.

After train TF-IDF Model, at the decoding step of the automatic title sentence creation, we will use this TF-IDF model to identify the most important term in the corresponding full text sentence, and force it to be the starting term to create an *auto-title*, rather than the character beginning of the sentence (for example the $<bos>$ character).

And finally, we still use the *Language Model* to correct errors in case the *auto-title* has the word repeated many times in a sentence with the mechanism as shown in Figure 5. With this approach, the automatically generated title sentences will have more meaning than the usual way, while avoiding repetition errors in a sentence using the *Language Model*. Our results will be presented in the experimental section.

D. Running Example

Finally, Figure 9 shows the complete flow of the $Forcing - Seq2Seq$ Model. First, from the original full text sentence, we will use the TF-IDF model to identify the most important term (in this example the term *"spectrum"*), then we will use the Seq2Seq model with attention mechanism to generate the corresponding title sentence using the *Teacher-Forcing* mechanism. As we can see, the decoder will receive the first character as the word *"spectrum"*, instead of the starting character (eg the $<bos>$ character) as usual. The automatically generated title sentence result will be: **"spectrum is the best coconut oil i have ever"**.

V. Evaluation Method

Automatic text summarization is a challenging task, because when humans want to summarize a full text, we need to read the whole text to understand it, and then rewrite it into a full text. Brief, concise summary of its content. So it is also a challenge to confirm the accuracy of the summary model, because we will have many summaries that can condense an original text. There is problem with no clear label. Therefore,

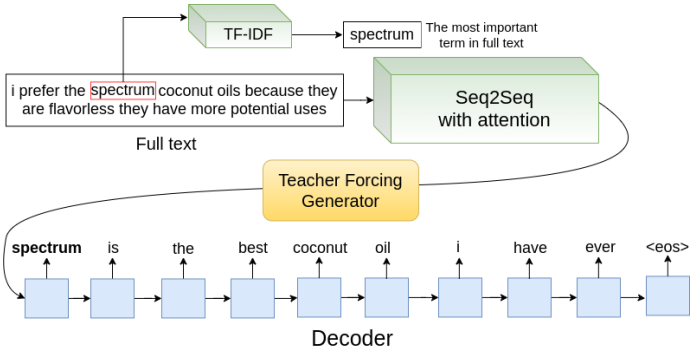

Fig. 9. Create *auto-title* with Teacher Forcing Generator

we decided to use 2 different methods to measure the accuracy and reasonableness of the model: automatic evaluation and human evaluation.

In the 2000s, there was a dataset to automatically evaluate the summary problem. The evaluation of text summarization methods is primarily based on metrics that measure the similarity of summaries, generated by the system, with a set of standard human-written summaries [23]. Some of the metrics used by many models are ROUGE, *Summary Assessment by Relevancy Analysis* (SERA), *Bilingual Evaluation Understudy* (BLEU), etc. This model uses one of those metrics to automatically evaluate the output: BLEU. BLEU is one of the algorithms used to evaluate the quality of text in machine translation problems, which can be a translation problem from one language to another or a text summary problem, etc. This measurement represents the correspondence between the reference and the machine output. The idea of BLEU is The closer the machine translation is to the human translation, the better [24]. BLEU is one of the first metrics that correlates with human judgments of quality, is one of the most popular and best low cost automated assessment indicators. [25] [26].

$$BLEU_{score} = \frac{Covered}{Total} \tag{7}$$

In addition to automatic evaluation, we also evaluated system output by eliciting human judgments.

VI. Experiment

A. Data set

$PGN - LM$ model and $Forcing - Seq2Seq$ use 2 datasets. The first dataset is the input texts that need to be generated title. We take input dataset for our model from Kaggle: *https://www.kaggle.com/snap/amazon-fine-food-reviews*. This dataset consists of reviews including full text, summary and title of fine foods from amazon. The data span a period of more than 10 years, including all 500,000 reviews up to October 2012.

The second dataset is an *extra corpus* to training TF-IDF model and *Language Model*. This dataset need large and general, so we decide to use Amazon reviews: *https://snap.stanford.edu/data/web-Amazon.html*. This corpus consists of reviews from amazon. The data span a period of 18 years, including 35 million reviews up to March 2013.

B. Experiment result

We show the input data and 3 auto tiles generated by 3 our model in Table 1. As introduced in the Preliminaries Section, $Base - Seq2Seq$ Title is the *auto-title* generated by Seq2Seq model and Attention Mechanism. $Forcing - Seq2Seq$ Title is the *auto-title* generated by the $Base - Seq2Seq$ Model, uses the TF-IDF model with the *Teacher-Forcing* mechanism and the *Language Model* to correct grammar errors. $PGN - LM$ Title is the *auto-title* generated by using the model we mention in Section 3, including the $PointerGeneratorNetwork$ Model and the *Language Model* to correct grammar errors.

Looking at Table 1, we can see that the automatic sentences generated by our 3 models all make more sense. So, we can use the titles generated from our model to replace the original title. The result also shows the $Forcing - Seq2Seq$ Title and $PGN - LM$ Title are better and more detail than $Base - Seq2Seq$ Title, even better than the original title. We use 2 evaluation methods to validate our 3 models.

1) Automatic Evaluation: We calculate the BLEU score between full text and *auto-title* to benchmark for our model. Our 3 models learn and try to generate a new title that represents the main idea of the original text with words not only dependent on the wording of the original text but of the entire corpus set. Therefore the BLEU score is usually low and does not indicate the accuracy of the model. That is also why we need human judgment. However, according to table 2, with BLEU Score, $Forcing - Seq2Seq$ and $PGN - LM$ still gave higher results than the $Base - Seq2Seq$ Model.

2) Human Evaluation: In addition to automated evaluation, human evaluation is used to measure the rationality and feasibility of models. We conducted a manual evaluation with the help of 11 graduated volunteers. We use the 2 criteria to evaluate auto-title. First, rationality is calculated by the proportion of the title that is well-formed and meaningful. Second, feasibility is calculated by the proportion of the title that is easier to read and makes more sense than the original title. To conduct the evaluation, we randomly selected 251 titles from the input dataset and asked the volunteers to evaluate subjectively. Each example includes a full text, original title and four generated auto titles, including *auto-title* from $Base - Seq2Seq$ Model, *auto-title* from $Forcing - Seq2Seq$ Model and *auto-title* from $PGN - LM$ Model. The volunteers selected the best *auto-title* for each full text according to the above criteria (many options possible). Scores for each model are calculated by the proportion of sentences selected by candidates divided by the number of given sentences.

Tables 2 shows the final results compiled from the reviews of the 11 volunteers. The results show that all 3 auto titles are rational and feasible, expressing the message of the original text. The $PGN - LM$ model outperforms compare with $Base - Seq2Seq$ and $Forcing - Seq2Seq$ model, with the rationality and feasibility of **99.49%**.

VII. Conclusions

This paper concentrates on improving the abstractive summarization method by Seq2Seq model with Atten-

TABLE I
SOME AUTO TITLES FROM THE DATASET

Full text	Original Title	Base-Seq2Seq Title	Forcing-Seq2Seq Title	PGN-LM Title
Delicious alternative pretzel for those who cannot eat wheat with just the right amount of crunch and salt	You do not have to give up pretzels	Pretzels	**Pretzel** with little salt and gluten free	Excellent snack food for those who can not eat wheat
These gummies were great not too sweet with a nice firm texture i will definitely be ordering more	Excellent	Great tasting	**Gummies** are flavor and texture and very tasty too	Excellent reminds me of all the cheese mint gummi bears
Awesome flavor awesome crunch it almost melts in your mouth could not ask for a better granola bar	Best tasting granola bar	Best tasting granola bar ever	**Awesome** best granola bar i have had	Sweet crunch and perfect for a better gluten free granola
These are great a mix of corn and sweet potato really addictive and gluten free a great product	Really good	Really good	**Addictive** and good just we like chips	Really good really sweet of a nice treat sweet potato

TABLE II
EVALUATION RESULT

Method	Metric	Base-Seq2Seq	Forcing-Seq2Seq	PGN-LM
Automatic validation	BLEU	35,23%	49,73%	42,88%
Human validation	Rationality	64,73%	82,10%	99,49%
Human validation	Feasibility	26,64%	69,57%	99,49%

tion Mechanism, introduces the combination of the extractive and abstractive approaches with idea of the $PointerGeneratorNetwork$, in order to better handle OOV words and factual inaccuracies, and *Language Model* to smooth output text. This method produces outperform results on the input dataset with high accuracy, concise, coherent, grammatically and human-validated. Besides, we also tested using *Teacher-Forcing* with Named Entity Recognition and *Language Model* to smooth output text.

However, $PNG - LM$ model and $Forcing - Seq2Seq$ is currently focusing on the field of food, but can be extended to apply in other fields.

REFERENCES

[1] Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assef, Saeid Safaei, Elizabeth D. Trippe, Juan B. Gutierrez and Krys Kochut, Text Summarization Techniques: A Brief Survey. arXiv:1707.02268v3, 2017.

[2] K. S. Jones, Automatic summarizing: the state of the art. Information Processing and Management, Elsevier, Vol. 43, No. 6, 2007.

[3] Vishal Gupta and Gurpreet Leha, A Survey of Text Summarization Extractive Techniques. Journal of Emerging Technologies in Web Intelligence 2, 2010.

[4] Chandra Khatri, Gyanit Singh and Nish Parikh, Abstractive and Extractive Text Summarization using Document Context Vector and Recurrent Neural Network. arXiv:1807.08000, 2018.

[5] P. Li, W. Lam, L. Bing, and Z. Wang, Deep recurrent generative decoder for abstractive text summarization. arXiv preprint arXiv:1708.00625, 2017.

[6] Tal Baumel, Matan Eyal, Michael Elhada, Query Focused Abstractive Summarization: Incorporating Query Relevance, Multi-Document Coverage, and Summary Length Constraints into seq2seq Models. arXiv:1801.07704, 2018.

[7] Abigail See, Peter J. Liu, Christopher D. Manning, Get To The Point: Summarization with Pointer-Generator Networks. arXiv:1704.04368, 2017.

[8] K. Lopyrev, Generating News Headlines with Recurrent Neural Networks. arXivpreprint arXiv:1512.01712, 2015.

[9] Thomas Cherian, Akshay Badola, Vineet Padmanabhan, Multi-cell LSTM Based Neural Language Model. arXiv:1811.06477, 2018.

[10] A. Aizawa, An information-theoretic perspective of tf–idf measures. Information Processing and Management, vol. 39, no. 1, pp. 381-397, 2003.

[11] Anirudh Goyal, Alex Lamb, Ying Zhang, Saizheng Zhang, Aaron Courville and Yoshua Bengio, Professor Forcing: A New Algorithm for Training Recurrent Networks. NeurIPS 2016.

[12] Ralf C. Staudemeyer, Eric Rothstein Morris, Understanding LSTM - a tutorial into Long Short-Term Memory Recurrent Neural Networks. arXiv:1909.09586, 2019.

[13] Andrea Galassi, Marco Lippi, Paolo Torroni, Attention in Natural Language Processing. arXiv:1902.02181, 2019.

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, Attention Is All You Need. arXiv:1706.03762, 2017.

[15] Ilya Sutskever, Oriol Vinyals, Quoc V. Le, Sequence to Sequence Learning with Neural Networks. arXiv:1409.3215, 2014.

[16] Shahzad Qaiser, Ramsha Ali, Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents. International Journal of Computer Applications (0975 – 8887), 2018.

[17] Amir Jalilifard, Vinicius F. Caridá, Alex F. Mansano, Rogers S. Cristo, Felipe Penhorate C. da Fonseca, Semantic Sensitive TF-IDF to Determine Word Relevance in Documents. arXiv:2001.09896, 2021.

[18] Qingyun Dou, Yiting Lu, Joshua Efiong, Mark J. F. Gale, Attention Forcing for Sequence-to-sequence Model Training. arXiv:1909.12289, 2019.

[19] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean, Efficient Estimation of Word Representations in Vector Space. arXiv:1301.3781, 2013.

[20] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean, Distributed Representations of Words and Phrases and their Compositionality. arXiv:1310.45, 2013.

[21] Fred Jelinek and Robert L. Mercer, Interpolated estimation of Markov source parameters from sparse data. In: Proceedings, Workshop on Pattern Recognition in Practice, pp. 381-397, 1980.

[22] Christopher D. Manning and Hinrich Schutze, Foundations of Statistical Natural Language Processing, 1999.

[23] Arman Cohan, Nazli Goharian, Revisiting Summarization Evaluation for Scientific Articles. arXiv:1604.00400, 2016.

[24] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu, BLEU: a method for automatic evaluation of machine translation. In: ACL-2002: 40th Annual meeting of the Association for Computational Linguistics, pp. 311-318, 2002.

[25] Kishore Papineni, Salim Roukos, Todd Ward and John Henderson, Corpus-based comprehensive and diagnostic MT evaluation: Initial Arabic, Chinese, French, and Spanish results. In: Proceedings of Human Language Technology, pp. 132-137, 2002.

[26] Chris Callison-Burch, Miles Osborne and Philipp Koehn, Re-evaluating the Role of BLEU in Machine Translation Research. In: 1th Conference of the European Chapter of the Association for Computational Linguistics: EACL, pp. 249-256, 2006.