Master of Science

Bridge Damage Factor Recognition from Inspection Reports Using Active Recurrent Neural Network

August 2018

Department of Civil and Environmental Engineering
The Graduate School
Seoul National University

Chung, Sehwan

Abstract

Bridge Damage Factor Recognition from Inspection Reports Using Active Recurrent Neural Network

Sehwan Chung
Department of Civil and Environmental Engineering
The Graduate School
Seoul National University

Extracting information from bridge inspection reports has recently attracted research interests, since the reports contain valuable information for predictive maintenance of bridges. However, a considerable amount of the reports has limited manually collecting such information from the inspection reports. The research objective was to propose a methodology for automatically recognizing information on bridge damages and the causal factors from the reports with less amount of training data. This study applied recurrent neural network to develop the recognition model, and active learning to train the model. In the active learning scheme, a human annotator was asked to label text data which the model had difficulty in recognizing

damages and causal factors from. Experimental results showed that the developed model performs well using only 140 training data to get f-1 score of 0.778, 90.5% of the maximum performance of the model. The proposed methodology will be applied to develop a model for extracting valuable information from bridge inspection reports, and eventually enable efficient preventive maintenance of bridges.

Keywords: Bridge Inspection Reports, Named Entity Recognition, Recurrent Neural Network, Active Learning

Student Number: 2016-21272

Contents

List of Tables

List of Figures

Chapter 1. Introduction

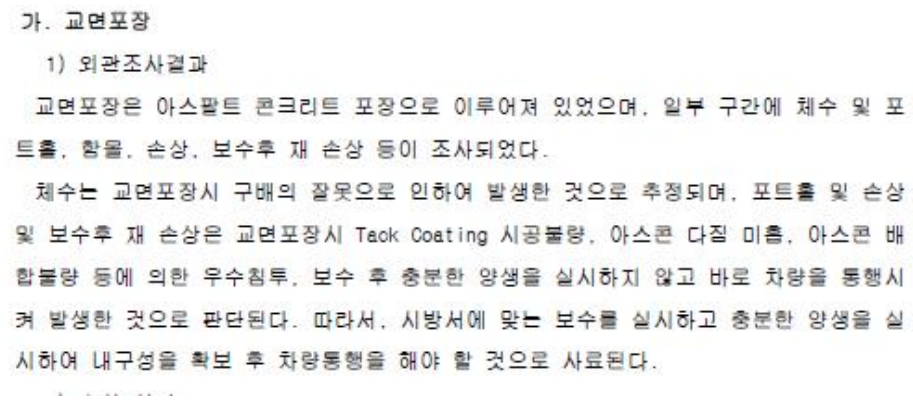
1.1 Research Background

Bridges play a key role in keeping traffic flow and contributing an economic development of a society, thus maintaining the infrastructure safe is essential for the public safety and the national economy. It has been known that common causes of bridge damages are in early phases of projects, such as planning, design, or construction. (Jeon et al., 2017) analyzed causes of damages in Korean expressway bridges and showed that 56.7% of the damages resulted from defects in planning, design, or construction phases (Jeon et al., 2017). These statistics imply that preventive maintenance, taking proactive actions from the beginning of bridge construction projects, might be much effective to prevent damages of bridges.

Predictive maintenance for bridges needs to be based on historical record on previous causal analyses of bridge damages, to correctly predict damages in the future and rationally diagnose the causes of the damages before the operation of the bridges.

Bridge inspection reports have the information, because the reports generally describe damages identified during the inspection and analyze causes of the damages (KISTEC, 2017). A project manager of a bridge construction project can utilize such information in the reports to predict possible damages of the bridge and even plan mitigations for the expected damages in the early phases of the project. Previous studies on causal analysis

of bridge damages have paid attention to the potential of buried information in the bridge inspection reports, thus these studies have utilized bridge inspection reports as information sources (Jeon et al., 2017; Lokuge et al., 2016; Peris-Sayol et al., 2017).


가. 교면포장

1) 외관조사결과

교면포장은 아스팔트 콘크리트 포장으로 이루어져 있었으며, 일부 구간에 체수 및 포트홀, 함몰, 손상, 보수후 재 손상 등이 조사되었다.

체수는 교면포장시 구배의 잘못으로 인하여 발생한 것으로 추정되며, 포트홀 및 손상 및 보수후 재 손상은 교면포장시 Tack Coating 시공불량, 아스콘 다짐 미흡, 아스콘 배합불량 등에 의한 우수침투, 보수 후 충분한 양생을 실시하지 않고 바로 차량을 통행시켜 발생한 것으로 판단된다. 따라서, 시방서에 맞는 보수를 실시하고 충분한 양생을 실시하여 내구성을 확보 후 차량통행을 해야 할 것으로 사료된다.

Figure 1.1 Example of Bridge Inspection Report

However, a considerable number of bridge inspection reports make it difficult to collect and utilize such valuable information in the reports (Liu and El-Gohary, 2017; Ryu and Shin, 2014).

Text mining with the recent advancement of machine learning enables automatic extraction of information from numerous text data, so that it can reduce human effort to manually collect it. Named entity recognition (NER), one of such text mining techniques which aims to recognize ‘named entity’ such as persons, organizations, or places from text data, has been extensively applied to recognize and extract information from text data in various fields

(Spasic et al., 2005; Tanabe et al., 2005; Zhu et al., 2013). Liu and El-Gohary (2017) proposed a NER model for recognizing and extracting the information about damages, the causes, and suggestions for maintenance from the text data in bridge inspection reports.

However, few studies have addressed the limitation of applying NER to the text data in bridge inspection reports that a much amount of text data must be tagged with labels by human annotators in advance to train NER models, and manually labeling text data is so expensive that such labeled text data is not available in the construction research domain.

1.2 Problem Statement

Although the information in the inspection reports enables project managers to predict possible bridge damages and plan mitigations for the expected damages in the early phases of the project, a considerable number of bridge inspection reports make it difficult to collect and utilize such valuable information in the reports.

While previous studies have applied NER to text data in various domain for automatic and efficient information extraction, most of the studies have not addressed the limitation that NER models require labeled text data as training data which demand tremendous human effort to manually label all the text data. Although a NER model proposed by Liu and El-Gohary aimed to solve the limitation and used a self-training scheme to train their model, self-training method which they applied has two deficiencies that, first, a self-training method assumes that prediction results with high prediction confidence score would be correctly predicted, and a self-training model does not provide a user with an opportunity to correct its prediction errors when the model is deployed in the practice.

1.3 Research Objective

This paper proposes a bridge damage factor recognition model which aims to automatically recognize and extract bridge damages and the factors from inspection reports, as NER models recognize named entities in text data. To build and train the model, this paper applies two methods: recurrent neural network (RNN) to build the model, and active learning to train the model.

RNN has shown empirical success in various kind of text analytics, such as natural language processing including NER, language modeling, and text generation. Because text data is often represented as a sequential data of individual words in the text, sequence models such as RNN were used in such text analytic tasks.

Active learning is selected because an active learning scheme has shown its potential to reduce the human effort to construct a training data for a machine learning model. Active learning thus is expected to probably solve the limitation of previous studies.

1.4 Research Scope and Process

The research scope is confined to bridges in general national highways in Korea, considering their significant portion to the total highway bridges in Korea, and data availability. The remaining paper is organized as follows. First, in Chapter 2, previous studies related to this study is examined and background knowledge is presented. Chapter 3, then, explains a methodology to develop the proposed bridge damage factor recognition model. Experimental results will be presented and discussed in Chapter 4. Finally, Chapter 5 wraps up the whole paper, clarifies the contribution of this paper, and suggests a direction for further studies to address the limitation of this study.

Chapter 2. Background Information and Related Works

2.1 Causal Analysis on Bridge Damages

Previous studies on analysis of bridge damages and their causes were found to be classified into two categories. Qualitative studies selected a specific bridge accident or a specific natural disaster, examined causes of the accident or influences of the disaster on the bridges, and benefitted from in-depth inspections, mechanical and structural knowledge, and experiences of experts in this domain. Studies in the other category were quantitative rather than qualitative, collecting and analyzing historical data on damages of a relatively lot of bridges.

Quantitative studies could be even classified further into two types by the type of data which the studies utilized. While researchers have analyzed structured data on bridge inspection, such as Bridge Management System (BMS) or National Bridge Inventory (NBI), unstructured text data such as inspection reports have recently attracted research interests. For example, (Jeon et al., 2017; Lokuge et al., 2016; Peris-Sayol et al., 2017) have utilized bridge inspection reports to analyze causes of damages in various kind of bridges in different regions. Although methods for the analyses were different in the studies, they commonly utilized bridge inspection reports as the main source for data collection.

However, most of previous studies on bridge damages and causal analysis focused on the analyses itself and did not present the method of data collection. (Jeon et al., 2017) analyzed the causes of damages in Korean expressway bridges using the information in 915 bridge inspection reports. Since the information in the inspection reports is unstructured and no specific method of data collection was presented, the data must have been manually collected by the authors, which is not a generally applicable solution to collect data from thousands of bridge inspection reports.

2.2 Bridge Inspection Reports in Korea

Safety control for bridges and other facilities in Korea is regulated by Special Act on the Safety Control of Public Structures. According to the Act, bridges in Korea are periodically inspected, and results of the inspections are documented as bridge inspection reports. The bridge inspections generally consist of several subtasks, including visual inspections, material test results, and analyses on structural safety (KISTEC, 2017).

Typical bridge inspection reports consist of several chapters as presented in Table 2.1. Among the chapters, the chapter on the visual inspection usually contains the information on causal analysis, and thus can be a data source for this study.

Table 2.1 Contents of Bridge Inspection Reports

No.	Content
1	Introduction to Inspection
2	Visual Inspection Results, Causal Analysis, and Material Test
3	Structural Safety Analyses
4	Suggestions for Maintenance
5	Conclusion

Although the visual inspection chapters contain the information of the bridge damages and their causal factors, the other parts of the reports were

also used in this study for constructing the list of vocabularies, tokenizing, and embedding a word into a word vector. Because the results of such procedures are often ensured by the large quantity of text data to be analyzed, this study will use whole parts of the inspection reports for the above procedures, to fully benefit from the amount of text data in the inspection reports to be collected.

2.3 Text Analytics in Construction

Since a construction project is unique and full of unknown uncertainties, information from the project is often recorded in a textual data such as documents or reports. While the other industries have benefitted from the advance of big data analytics, artificial intelligence, and machine learning techniques, such industries have collected structured data from surveys, sensors, or other databases, which are not available in the construction industry.

Unstructured text data in construction field have recently attracted research interests for knowledge management in entire cycle of construction project and lessons-learned feedback from historical practices. Such researches have been actively developed in countries where people use Latin-based language including English, since a large number of corpora, text analyzing tools such as tokenizers and stemmer were already available for that languages.

Text analytics for data from construction practices in Korean, however, have only been conducted in the level of frequency analysis of terms in the text document. Researches on text analytics for construction documents in Korean usually have focused on identifying main keywords and analyzing relationships between those keywords using time-series analysis or association rule mining (Jeong and Kim, 2012; Lee et al., 2016).

This study preliminarily conducted a word frequency-based analysis on the bridge inspection reports data, examined a feasibility of such frequency-

based methodology, and showed limitations of this approach.

2.4 Named Entity Recognition and Applications

NER is a method in natural language processing (NLP) and aims to recognize 'named entities' from natural language text, such as a name of a person, an organization, or a place (Fig 2.1). With a recent advance of machine learning techniques, many NLP models have been developed by following a machine learning framework. Besides of recognizing such general named entities, recent studies have widely applied NER to various text data to extract domain-specific terms. For example, researchers in biomedical domain have developed numerous NER models to extract biomedical terms such as names of genes, proteins, or diseases (Tanabe et al., 2005; Zhu et al., 2013).

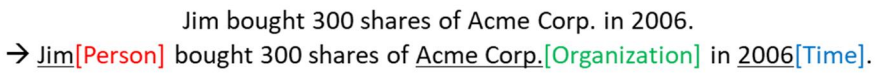


Figure 2.1 Example of NER

While application of NER is not as activated in construction research as in other researches, Liu and El-Gohary (2017) applied NER to bridge inspection reports and suggested a self-training model for automatically extracting information from the reports. Their model aimed to automatically recognize information in 11 categories, including bridge element, deficiency, deficiency cause, and maintenance action. Although their model showed an acceptable recognition performance, their model applied a self-training

method to train their model, which showed a limitation for correctly predicting labels of unlabeled words. Since self-training scheme assumes that high confidence of model prediction ensures that the prediction is correct, it might badly affect the model performance and reliability that a self-training model incorrectly labels a word even if the confidence of the prediction is high.

A NER model, with text data in natural language as inputs, aim to predict classes of all words in the text as outputs. In Fig 2.1, for example, the input data is the sentence of "Jim bought 300 shares of Acme Corp. in 2006.", and the output is categories of each words in the sentence. In other words, the input of the NER model is a sequence of words, and the output is a sequence of corresponding classes of the words. In this perspective, a NLP task is considered as a typical sequence labeling task, and thus a NER model based on machine learning framework is required to process sequence data as inputs of itself.

RNN thus is a proper method for being applied to NLP because a RNN model can process sequential input and output data. While RNN have diverse form by the structure and connection between layers and nodes, the simplest form is a RNN with self-feedback of its hidden layer (Fig 2.2). Such recurrent feedback structure enables a RNN model to learn sequential patterns in input data when input data of the model is a sequential data and the model is fed with each item in the sequences in a row.

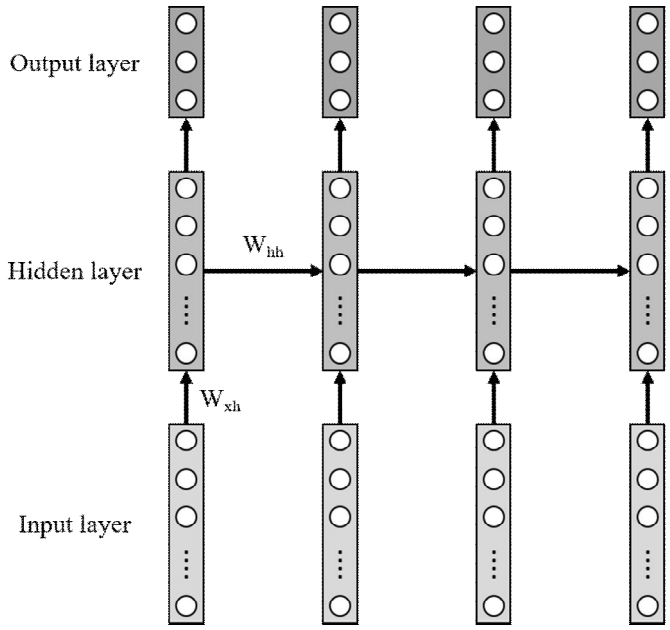


Figure 2.2 Basic RNN Architecture

However, a conventional RNN model was found to often fail to learn long-term dependency in input sequences, and long short-term memory (LSTM) was proposed to overcome the limitation (Hochreiter and Schmidhuber, 1997). LSTM uses several activation functions in one node, rather than one activation function such as sigmoid or hyperbolic tangent. Basically, an output of a LSTM cell consists of two parts: a hidden state and a cell state. A hidden state is propagated to the output nodes and hidden nodes in the next time step, as a hidden node in typical RNN architecture is. A cell state, a unique characteristic of LSTM, is propagated only to the next hidden nodes, but affects the calculation of final hidden states.

Another limitation of a regular RNN is that the propagation of node

weights is forward-directional only, which limits a RNN model to learn only forward-directional sequential patterns. (Schuster and Paliwal, 1997) suggested bidirectional RNN and adopted two-way hidden nodes in one hidden layer in their architecture to enable a bidirectional RNN model to learn forward-directional sequential patterns and backward-directional patterns both (Fig. 2.3). This characteristic has been considered to be especially useful for text analytics because text data in written language is often inputted to a model all at once unlike other sequential data such as time-series data which is inputted to a model in a row.

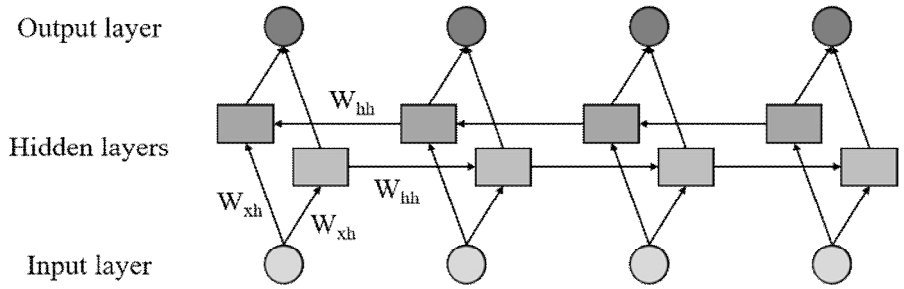


Figure 2.3 Bidirectional RNN Architecture

Combining these features into one architecture for application of NER, a numerous bidirectional LSTM models were suggested and applied to NER as well as other sequence labeling tasks such as speech recognition and part-of-speech tagging (Chiu and Nichols, 2016; Huang et al., 2015; Lample et al., 2016; Lee, 2015; Lee et al., 2017).

2.5 Active Learning

While recent studies have benefitted from the advance of neural network architectures in developing NER models in various fields, concerns for availability of training data for NER models have remained. To develop a machine learning model for NER tasks, the model first should be trained with enough training data. In NER tasks, training data is a text data whose words are correctly labeled in advance to the model development, but manually labeling all the words is time and labor-consuming.

Active learning aims to reduce the effort to prepare the training data for model development, under the assumption that using only some data is enough for a model to reach a certain performance if we properly select more informative data in our data pool. Contrary to a self-training method, where an initial training set is constructed and then a model automatically adds the most confident prediction to the training data, active learning method requires a human to annotate data points which are selected by the model as they are the least confident, or the most informative. It is analogous to a student who selects questions which he or she feels the most difficult to solve for the best efficiency of his or her study.

Settles and Crave (2008) analyzed general strategies for active learning in sequence labeling tasks, compared general methods for query selection and measurement of the ‘confidence’ of model prediction, and tested several methods by applying those methods to conditional random fields. The comparison results revealed that performances of the methods depend on the

corpus which the methods were applied to and no 'best strategy' can be determined, but also showed that active learning method nevertheless a better performance than a random selection method in terms of the reduction of the number of labeling to get the same model accuracy.

Key features of active learning are how to select the most 'informative' samples from unlabeled data to query to a human annotator. Settles (2009) surveyed and analyzed strategies for query selection, including uncertainty sampling which is the simplest and the most commonly used in active learning applications. Prediction by a model can be considered less confident if uncertainty of the prediction is high. Uncertainty of prediction is generally measured by entropy as presented in Eq. 1, where y_i ranges over all possible classes.

$$\text{entropy(x)} = -\sum_i P(y_i|x) \log P(y_i|x) \qquad \textbf{Eq. 1}$$

Tomanek and Hahn (2009) proposed an semi-supervised active learning process for sequence labeling tasks, which only focused on the specifically highly uncertain tokens in a sequence and assumed that the other remaining tokens with low uncertainty would be correctly labeled by a model. Their results showed that their active learning method can reduce human effort to manually label words in text data by 60%, in terms of the number of labeled words out of the total words.

Chapter 3. Frequency-Based Damage Pattern Identification

3.1 Damage Pattern Identification by Term Frequency Analysis

As preliminary studies, the author examined applicability of term frequency-based analyses for identifying patterns between bridge damages and their factors from the inspection reports. In those analyses, the main assumption was that 1) a word which frequently appears in the inspection reports is important and thus should be considered as a key component in the bridge damage patterns to be identified from the reports, and 2) two or more words which are used together in the same context have a correlation between them. The context where the words are used can be an individual report, an individual sentence, or a fixed-size window consisting of about four or five words.

To identify correlations between frequent words in the inspection reports, text data in the collected inspection reports were separated into individual words first. Tokenizing Korean language often involves with morphologic analysis due to the unique characteristic of Korean language. Therefore, many existing Korean tokenizers usually analyze text morphologically first and then separate each morpheme. Splitting the text data into words were preliminary conducted by an existing Korean language analytic tool, a Twitter tokenizer in

KoNLPy (Korean Natural Language Processing in Python) (Park and Cho, 2014).

While frequent words can be considered as important and meaningful words in the text data, some words are so frequent that every document has the word and the word cannot take any role in patterns between words in the text. Such too frequent words, called stopwords, were manually identified in the preprocessing steps and removed from the tokenized text so that analysis results can give more realistic and meaningful correlation between bridge damages and their causal factors. In the preliminary study, words whose part-of-speech (POS) were neither noun, verb, nor adjective were categorized as stopwords and therefore removed.

Correlations between two factors were calculating basically in co-occurrence of the two words representing the two concepts. If two factors are closely related to each other, the frequency that the two words are used in the same document would be high. In this concept, a frequency matrix was constructed, whose columns represent preliminary set context words, words for main damage categories in this study, and rows represent all the other words, and each cell counts for the frequency that the word in the row were used in the document where the context word in the column were also used.

For example, <TABLE> shows the results of frequencies counting that frequencies of some remarkable words which were used in the documents where the word ‘crack’ were also used. From the results, a damage pattern that ‘crack in concrete element due to drying shrinkage during construction phase’ can be deducted.

Table 3.1 Selected Words Related to the Keyword 'Crack'

No.	Word	POS	Count
1	콘크리트 (concrete)	Noun	11877
2	수축 (shrinkage)	Noun	6875
3	시공 (construction)	Noun	6488
4	건조 (drying)	Noun	6185

However, some words turned out to be still too frequent to distinguish one document from others. For example, the words 'concrete' and 'construction' are so frequent that the correlation between those words can be biased due to the natural high frequency of the words. To revise the effect of the natural high frequency of such words, the frequency of each word were adjusted by positive pointwise mutual information (PPMI). If one word is the most frequent word in a document, but appears in all the other documents, that word gives is considered to be little informative. Since PMI tends to be biased toward rare terms, Laplace smoothing, adding a small constant k to the word counts or a constant α to the probability of word, was applied.

$$\mathrm{PMI}(\mathrm{word_i}, \mathrm{word_j}) = \log_2 \frac{P(word_i, word_j)}{P(word_i) \times \{P(word_j) + \alpha\}} \quad \textbf{Eq. 2}$$

$$\mathrm{PPMI}(\mathrm{word_i}, word_j) = \max(PMI(word_i, word_j), 0) \quad \textbf{Eq. 3}$$

Table 3.2 shows the results of PPMI weighting for the context word

‘leakage’, which showed a prominent improvement of identifying damage patterns comparing with the results of simple word counting. The regularization factor α was empirically set to 0.0001. Top 10 terms sorted by their PPMI for the word “leakage” included “sub(structure)”, “replacement”, “rainwater”, “expansion”, “joint”, drainage”, and “contamination”, which indicate the mechanism that ‘a leakage from an expansion joint may lead rainwater flow to the substructure and cause contamination of the substructure’.

Table 3.2 Top 10 Words with the Highest PPMI with the Keyword ‘Leakage’

No.	Word	POS	PPMI
1	하부 (substructure)	Noun	1.62
2	교체 (replacement)	Noun	1.61
3	빗물 (rainwater)	Noun	1.56
4	이음 (joint)	Noun	1.44
5	신축 (expansion)	Noun	1.40
6	부속 (appurtenance)	Noun	1.31
7	배수 (drainage)	Noun	1.25
8	흔적 (sign)	Noun	1.20
9	유도 (induction)	Noun	1.20
10	오염 (contamination)	Noun	1.13

3.2 Limitations of Frequency-Based Approach

Although the tokenization tool showed somewhat acceptable results, it was shown that technical terms or abbreviations in construction and infrastructure inspection domains were often recognized as non-words. For example, "아스콘 (ascon)", an abbreviation for "asphalt concrete", was wrongly separated as the interjection "아 (a)" and an unknown word "스콘 (scon)" (Fig 3.1).

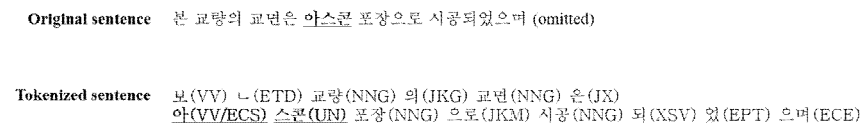


Figure 3.1 Tokenization Error Case

This is considered to be because the vocabulary list which was used for morphological analysis and tokenization did not contain such domain-specific terms in construction field and bridge inspection reports. For the ease of data collection, vocabulary lists are generally constructed based on the news articles, text in books or famous novels which are easily available for the researchers who develop the morphologic analyzer and tokenizer. Therefore, it turned out that technical and domain-specific terms, especially words for bridge elements and damages, were not included in the existing tokenizers.

Another limitation of the term frequency-based methodology for identifying damage patterns from the text in the inspection reports was that

such frequency-based methods were only able to extract general and well-known damage patterns from the inspection reports. The exemplified damage patterns, ‘dry shrinkage crack in concrete’ and ‘substructure contamination due to leakage of expansion joint’, are already well-known and established mechanisms of bridge damages. From the frequency-based damage pattern identification, one can capture such patterns from the text data by quantitative and objective manner, but is hard to find a novel or unusual pattern from the text data.

The most prominent limitation is, however, that general and overall damage pattern results based on the frequency-based methodology cannot specify in which bridge the damage occurred and what factor specifically caused the damage. Term frequency-based methods are appropriate for discovering overall and dominating patterns from a large amount of data, but not suitable for identifying damages and the actual causes in individual bridges and their elements.

Chapter 4. Development of Bridge Damage Factor Recognition Model

4.1 Bridge Damage Factor Recognition Model Overview

The aim of this study is to propose a bridge damage factor recognition model which automatically recognizes damages and their factors from text in bridge inspection reports. Since relationships between bridge damages and the factors are closely related to the element where the bridge has occurred, the model consequently determines which class each word in the input text is classified as, among four possible classes: 'element', 'damage', 'factor', and 'others' for a word which is neither a bridge element, a damage, nor a causal factor. Therefore, the input of the model is a sequence of words in the text, and the output is a corresponding sequence of classes which the words in the input belong to (Fig 3.1).

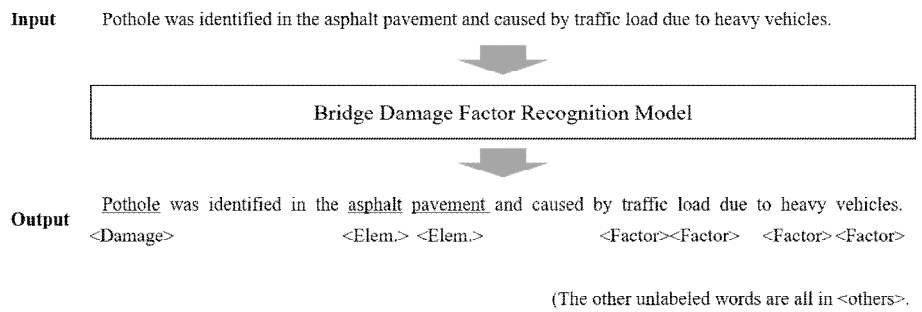


Figure 4.1 Bridge Damage Factor Recognition Model

4.2 Input Features of the Model

4.2.1 Preprocessing Text Data in Bridge Inspection Reports

Bridge inspection reports in Korea are often stored in formats of Portable Document Format (PDF) or Hangeul (HWP) word processor files. Since common data analytic tools cannot use data in such formats, raw text should be extracted from the reports files as plain text format.

Languages in bridge inspection reports were found to be formal, and spellings and syntaxes such as spacing or punctuation were almost correct. From this characteristic, this study applied rule-based sentence segmentation since all sentences should end with periods, not a question mark, an exclamation mark, or any other punctuations.

4.2.2 Tokenization

Tokenization is the process of segmenting text data into individual words. The proposed bridge damage factor recognition model aims to classify each tokenized word as bridge element, damage, factor, or others. One eojeol (어절), an unit of Korean language separated by blanks, often consists of a word which contains meaning of itself, and a functional unit such as a postposition and an ending. Therefore, a tokenizer for Korean text has to identify a word in an eojeol and separate the word from the other part in the eojeol. Identifying a complete word in Korean text is generally based on a list of all vocabularies used in the text, which is known to perform a crucial role

in tokenization because any word not included in the vocabulary list is not identified as a word.

This study constructs a vocabulary list from words used in bridge inspection reports, and tokenize the text data in the reports based on the vocabulary list, by applying a corpus-based tokenization process suggested by Kim. All candidates to be complete words are identified from the text to be tokenized, and likelihoods, or word score, that these word candidates are actually complete words are measured. Among all possible strings of characters in each eojeol, the string whose word score is the largest is identified as a word, and the other characters of the eojeol are separated from the word.

Measuring a word score is based on a cohesion probability (Kim, 2013) and a branching entropy (Jin and Tanaka-Ishii, 2006). A cohesion probability of a text string represents the probability that the last character of the string is actually used when the other characters in the string were observed. A branching entropy, when given a text string, represents the uncertainty about the next character after the string.

A cohesion probability for a text string is calculated as shown in Eq. 2, on the assumption that a string of characters frequently used together with each other is more likely to be a complete word. For example, the word '콘크리트' (concrete) has a high cohesion probability because '트' is maybe the only possible characters which can appear after the string '콘크리', and it is therefore likely to be considered as a complete word. On the other hand, the word '콘크리트의' havs much lower cohesion probability than '콘크리트'

because many possible options exist for the next character after '콘크리트', considering usages such as '콘크리트의', '콘크리트에', or '콘크리트는'.

$$\text{cohesion}(c_1, c_2, \ldots, c_n) = \sqrt[n-1]{\prod_{i=1}^{n-1} P(c_1, \ldots, c_{i+1} | c_1, \ldots, c_i)} \quad \textbf{Eq. 4}$$

A branching entropy measures an uncertainty for the next character after given successive characters, as shown in Eq. 3. For example, the string '콘크리' has a low branching entropy, because it is obvious that the character '트' will be used after '콘크리'. On the other hand, '콘크리트' has much higher branching entropy because it is not certain to guess which character will be used after the string, and therefore the string '콘크리트' is more likely to be a word.

$$H(X|X_n) = \sum_{x \in X} P(x|x_n) \log P(x|x_n) \quad \textbf{Eq. 5}$$

Combining the cohesion probability and the branching entropy, the likelihood that a given string is a complete word is measured as in (Eq. 4). In this process, word scores of all possible word candidates from the text data are measured, and list of all possible vocabularies is constructed with their corresponding word scores.

$$\text{score}(\text{word}) = \text{cohesion}(\text{word}) \times e^{H(\text{word})} \quad \textbf{Eq. 6}$$

To tokenize one eojeol, assuming that spacing is correct, the tokenizer calculates word scores of all possible strings in the eojeol, and extracts the string whose word score is the highest among all the strings. To tokenize the eojeol '콘크리트에서', for example, the tokenizer looks up all words scores for all possible candidate words: '콘', '콘크', '콘크리', '콘크리트', '콘크리트에', and '콘크리트에서'. If the word score of '콘크리트' is the highest, then '콘크리트' would be extracted as a complete word, and the other part, '에서', will be separated. Consequently, the exampled '콘크리트에서' will be tokenized as '콘크리트' and '에서'. All eojeols in the text data are applied to the process, and tokenized finally.

4.2.3 Word Embedding

Since a machine learning model cannot process a raw text data, the text data should be converted into numerical values so that the model can be fed with the numerically represented text data. While the simplest method is considering the words in the text as categorical values and applying one-hot encoding, the one-hot encoded words are too big and sparse, and lose their semantic and syntactic relationship between the words in the original text data.

To solve the issue, researchers have intensively developed methods for representing a word as a 'dense' word vector, or embedding a word into a vector space in other words, by using artificial neural network methods with

the advance of such techniques. Mikolov et al. (2013) proposed ‘word2vec’ algorithm for word embedding, and their method has been widely applied to language modeling tasks in various fields.

Word2vec algorithm is based on distributional hypothesis in linguistics that two words having similar context in their neighbors would have similar meaning. For example, the words ‘cat’ and ‘dog’ have similar context when these words are used in two sentences: “A cat sits on the table” and “A dog sits on the board.” Even if a person does not know the exact meaning of two words, ‘cat’ and ‘dog’, it could be inferred that two words have similar meaning from the sentences presented because the words were used in the same context (Fig 3.2).

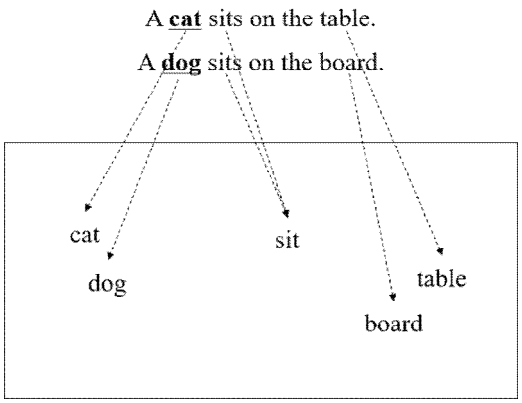


Figure 4.2 Example of Word Embedding

Word2vec algorithm aims to embed words in text data into a vector space so that similar words sharing similar contexts have similar vector values, in

accordance with the distributional hypothesis. For this purpose, word2vec algorithm uses a simple two-layer neural network for embedding a word into a numerical vector. The neural network is trained to predict a specific word when other neighboring words to the target word is fed into the input layer of the network. To be specific, input layers are given one-hot vectors of context words of the target word. The values in the input layer is then feed forward to the hidden layer, which has the size specified by a user. The values in the hidden layer is finally feed forward to the output layer, and the weights of the neural network are adjusted so that the network successfully predict target words when the network is fed with context words of the target words. After training the network, a specific word is mapped to the values in the hidden layer, which is finally defined as a word vector of the specific word.

4.3 Internal Model Architecture

In this study, RNN is applied to the bridge damage factor recognition model, since a RNN is a useful for processing sequential input data with varying lengths. To be specific, bidirectional LSTM is used in this study because determining a label of a specific word must be based on the usage of all the other words in the sentence in which the target word is used, before and after the target word. The overall architecture of the model is described in Fig 3.3.

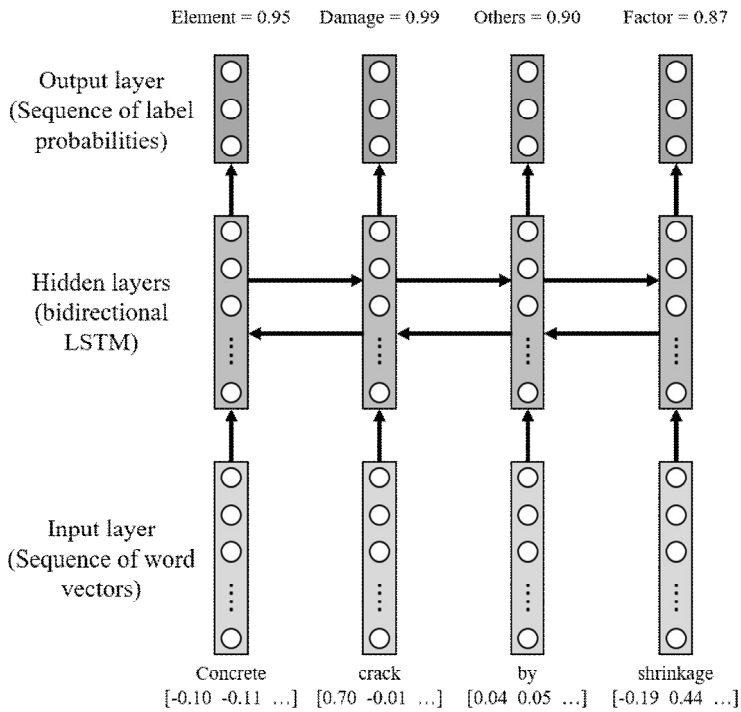


Figure 4.3 Model Architecture

The model input is a sentence, represented as a sequence of word vectors embedded by the word2vec algorithm. Two hidden layers, a forward-directional layer and the other backward-directional layer, have the same number of LSTM cells with the length of input sequence, or the number of words in the input sentence. Each cell in one hidden layer is connected in one-way, which makes the model able to learn linguistic patterns for expressing cause-effect relationships. The output from the hidden layers is a sequence of vectors with the size of four; each element in a vector corresponds to each class which a word is classified as by the model. Interpreting the vector in the

output layer as a logit of predictions for each word, the model classifies each word in the input sentence as the class whose logit is largest among four classes. For the loss function of the model, softmax cross entropy function, normalized by the length of an input sentence, is used.

4.4 Model Training Process Using Active Learning

To overcome the limitation that building a training data for the model is time-consuming and labor-intensive, this study applies an active learning method to the process of model training, in order to select the most informative samples from unlabeled text data and efficiently train the model with less training data. Right after being collected, text data in the inspection reports is not unlabeled and therefore not able to be used as a training data for model development. While the text data needs to be labeled before being used to train the model, a part of the text data is enough to gain equivalent model performance, which is an active learning aims to achieve. The overall process of model training in this study is described in Fig 3.4.

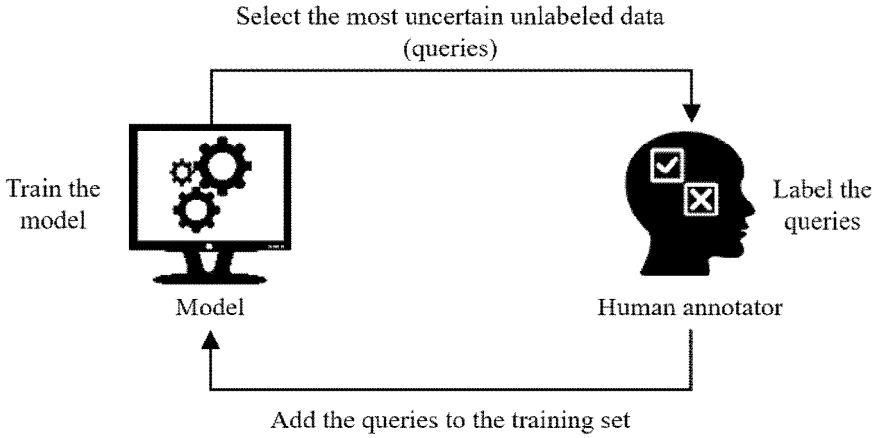


Figure 4.4 Model Training Process

1) Initial training data is selected from the collected but unlabeled text

data in the reports, manually labeled, and then added to the training set which is used later to train the model.

2) The model is trained using the training set prepared in the previous step.

3) Sequence entropies (Eq. 5) of the other remaining unlabeled data are calculated, based on the prediction results by the trained model. Using the sequence entropies, k unlabeled data whose sequence entropies are the largest among all the unlabeled data, where k is the number of sampling pool, and queried to the human annotator.

4) All words in the selected queries are labeled by the human annotator, and added to the training set.

5) Procedures from 2) to 4) are repeated, until all the data are labeled and used for the model training, or a performance measure of the being trained model meets the requirement set by a model developer or a user.

$$\text{seq\_entropy}(x_1, x_2, \ldots, x_n) = \frac{-\text{entropy}(x_n)}{n} \qquad \textbf{Eq. 7}$$

Chapter 5. Implementation and Validation

5.1 Data Collection and Preparation

This study collected 1,188 inspection reports on bridges in general national highways in Korea. For constructing input features of the model, tokenization and word embedding in other words, all the collected reports were used in order to fully benefit from the abundance of text data collected. However, it was severely time-consuming and labor-intensive to manually label all words in the text data. Therefore 350 inspection reports among the total were used for training and testing the model.

5.2 Experiments Process

According to the proposed methodology, raw text was extracted from all the collected inspection reports. Since the reports were saved as Portable Document Format (PDF) or Hangeul (HWP) files, the files were converted into plain text (TXT) files by open-sourced text extraction tools <ref. pdftotext, pyhwp>.

Sentence segmentation was conducted based on a rule-based algorithm. Assuming that every sentence in the reports end with a period, segmentation rules were established as follows.

1) Every text string which ends with a period was considered as a sentence candidate.

2) Every string in which the number of characters just before the period was less than two were excluded from the candidates, in order to rule out captions of tables.

3) Every string whose length, or the number of characters in the string, was less than ten were excluded, in order to rule out errors from raw text extraction and captions not excluded by the second rule.

Total 724,288 sentences were identified by the rule-based algorithm from the raw text, and then tokenized by “soynlp” package in Python which implemented the tokenization process described in the methodology section (Kim, 2018). The tokenized words were embedded into a vector space with the size of 50 by word2vec algorithm, implemented by “gensim" package in Python (Rehurek et al., 2010).

The bidirectional LSTM model for bridge damage recognition were implemented with “Tensorflow” package, which offers a framework for developing deep learning models, having two hidden bidirectional LSTM layers with the size of 100 (Abadi et al., 2016).

Training and testing data for the model development and validation were prepared by manually labeling all words in the training and testing data. This study used only the visual inspection chapter in 350 reports as training and testing data. While the training and testing data were segmented into 6,134 sentences, due to limited time to label all the words in the sentences, 1,650 sentences were labeled up to the date and used for training and testing the model, which might be still enough to validate recognizing performance of the model and effectiveness of the active learning method.

The first experiment was conducted to validate recognizing performance of the bidirectional LSTM model. Labeled 1,650 sentences were divided into training data of 1,300 sentences and testing data of 350 sentences, and the model was trained by the training data. In this experiment, the model was fed with the training data at once, not in a sequential manner.

In the second experiment, the same training and testing data were used, but the model was fed with the training data in a sequential manner, following the active learning method. Ten sentences were randomly selected as the initial training data. The model learned from the selected ten sentences and predicted labels of all words in the testing data, and f-1 scores for all classes were calculated to trace the increase of recognizing performance of the trained model. Another ten sentences were selected from the testing data whose

sequence entropies were the highest. This process was repeated until all sentences in the training data were used for model training.

5.3 Tokenization and Word Embedding

Since no golden standards for tokenization and word embedding were available, results of tokenization and word embedding were qualitatively examined for several representative words.

For tokenization, the author examined whether the words such as "콘크리트 (concrete)", "균열 (crack)", "망상균열 (alligator crack)", and "배수관 (drainage pipe)" were properly tokenized. It was found that "콘크리트", "균열", or "망상균열" was correctly tokenized probably because these words were quite frequently used in the inspection reports. However, "배수관" tended to be tokenized as "배수" and "관", and other words related to drainage, such as "배수시설" or "배수구" showed similar tokenization results. Such incorrect segmentation implies that the tokenizing algorithm interpreted "배수" as a separate word and consequently divided other drainage-related words into two parts even though the other words are also complete compound words, and suggests the necessity for postprocessing the results of tokenization to get more reliable and acceptable results.

Results of word embedding were mainly examined whether similar words had similar word vectors in the embedded space, by using cosine similarity to measure the similarity between two word vectors (Eq. 6).

$$\text{similarity}(\overrightarrow{x_1}, \overrightarrow{x_2}) = \frac{\overrightarrow{x_1} \cdot \overrightarrow{x_2}}{|\overrightarrow{x_1}| \times |\overrightarrow{x_2}|} \qquad \textbf{Eq. 8}$$

Table 5.1 shows the ten most similar words with “주형 (girder)”, “균열 (crack)”, and “습기 (humidity)”, respectively, and suggests that word categories such as bridge element, damage, and factors can be clearly determined by a classification model with the embedded word vectors.

Table 5.1 Examples of Word Embedding Results (Translated in English)

Words	Similarity
Scaling	0.580
Spalling	0.573
Destroy	0.525
Damage	0.500
Delamination	0.492
Efflorescence	0.491
Pothole	0.481
Efflorescence	0.464
Vertical	0.449
Micro	0.446

(a) Crack

Word	Similarity
Girder	0.724
Support	0.671
Inner	0.508
Stringer	0.499
Span	0.482
Cantilever	0.479
Stringer	0.472
Center	0.472
Deck	0.466
Box	0.463

(b) Girder

Word	Similarity
Water	0.708
Rainwater	0.623
Surface water	0.622
Salt water	0.581
Salt	0.576
Foreign substance	0.565
Surface water	0.557
Dust	0.548
Substance	0.542
Ice-melting agent	0.528

(c) Humidity

5.4 Recognizing Performance of the RNN Model

To validate the recognizing performance of the proposed bridge damage factor recognition model, the RNN model was trained with the training data of 1,350 labeled sentences, and tested with the other testing data. For every word in the testing data, the predicted class of the word by the model and the actual class were compared. Examples of the model prediction results are presented in Fig 5.1, with the actual classes of the words in the sentences.

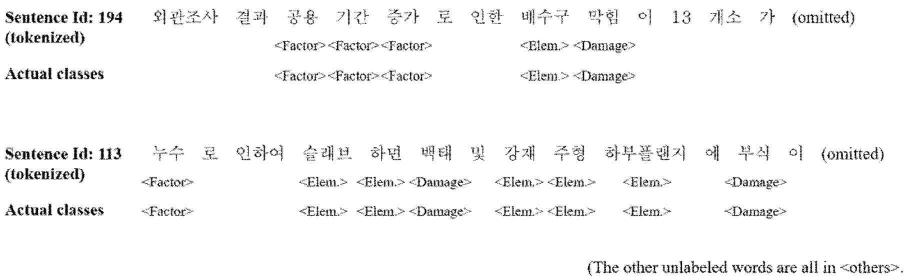


Figure 5.1 Examples of Model Prediction Results

As presented in Fig 5.1, the trained RNN model could recognize bridge elements, damages, and their factors from the actual text in bridge inspection reports. For example, in the 194th sentence, it was recognized that a drainage hole was clogged due to the extended operation time by the model. In the 113th sentence, it was recognized that a leakage caused the efflorescence in the bottom of the slab, and the corrosion in the bottom flange of the steel girder.

Table 5.2 shows a confusion matrix of prediction results for the testing

data after the model was trained with the training data. While the accuracy of prediction was as high as 0.927, it might be misleading to evaluate the model using accuracy because most of the words neither belonged to bridge element, damage, nor factor.

Table 5.2 Confusion Matrix

		Predicted class			
		Element	Damage	Factor	Others
Actual class	Element	979	10	13	197
	Damage	16	376	16	86
	Factor	10	18	697	75
	Others	186	57	67	7,417

To mitigate the effect of overbalanced words in 'others' class to the model accuracy, F-1 score for each class and averaged F-1 score for all four classes were presented in Table 5.3.

Table 5.3 F-1 Scores

Class	Element	Damage	Factor	Others	averaged
F-1 score	0.819	0.787	0.876	0.957	0.860

Compared with the results by Liu and El-Gohary (2017), the averaged f-1 score of 0.860 in this study is competitive to their results, while the f-1 score

for ‘damage’ class was slightly lower by 0.787 than f-1 scores for other classes. Such lower score for recognizing ‘damage’ class may have resulted from a mechanism of bridge damages that one damage in an element can cause another damage in another nearby element, which make it difficult for the model to classify a word for a bridge damage as ‘damage’ class or ‘factor’ class. For example, in predicting the 279th sentence, the model misclassified the class of the word “열화 (deterioration)” as ‘Damage’, although the word should be classified as ‘Factor’, because the sentence is reporting that surface exfoliation was identified, as a result from freezing effect, calcium chloride, extended operation time, and the deterioration of the concrete curb Fig 5.2.

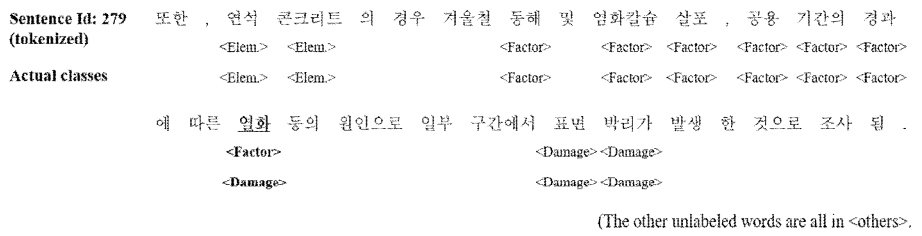


Figure 5.2 Model Prediction Error Case

Nevertheless, the results of recognizing damages and factors may be still effective when the relationship between the damages and the factors is straightforward, and practically useful because inspection reports generally describe the analyzed mechanisms of bridge damages as clearly as possible.

5.5 Validation for the Active Learning Method

While the effectiveness of the active learning process in this study was validated with manually pre-labeled training and testing data, the author implemented an annotation interface for manual labeling. The annotation interface, revised from the annotation tool by Roth and Mayhew (2018), was developed in this study to enable for a model developer to easily construct a training and testing data, by clicking a word to be labeled and select the appropriate class in GUI (Graphical User Interface) environment (Fig. 5.3).

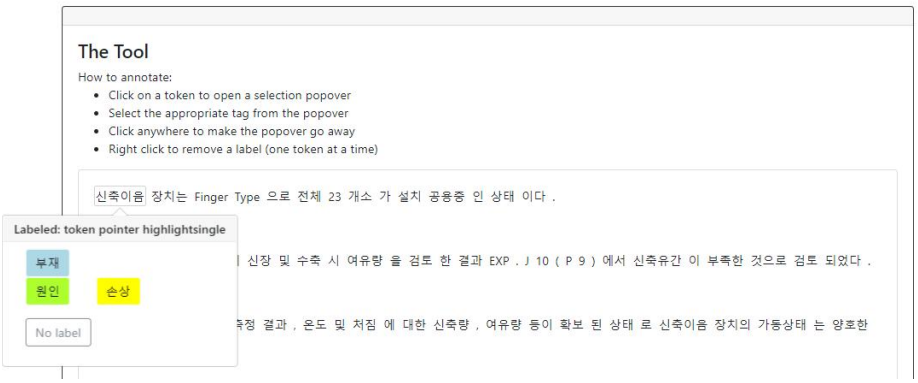


Figure 5.3 Interface for Data Labeling and Active Learning

Along with the active learning process described in the previous section, the interface in the beginning presents the initial query, which a human annotator labels all words as corresponding classes except for the class ‘others’, for the labeling efficiency. The human annotator can click a word to open a popup which requires the annotator to select the appropriate class for

the word. After labeling the target word, the labeled word is highlighted to clarify which class the word is labeled as (Fig. 5.4).

Figure 5.4 Labeling Demonstration

The manual labeling results are provided with the server end, and the RNN model running on the server are trained with the labeling results. After the training, the model predicts the classes of all the other words in the remaining unlabeled training data, and evaluates the sequence entropies of the predicted unlabeled sentences. The model selects the k query sentences with the highest sequence entropies to be labeled by the human annotator, and provides the query selection results with the interface. The interface then updates the sentences to be labeled in the screen with new queries from the server (Fig. 5.5).

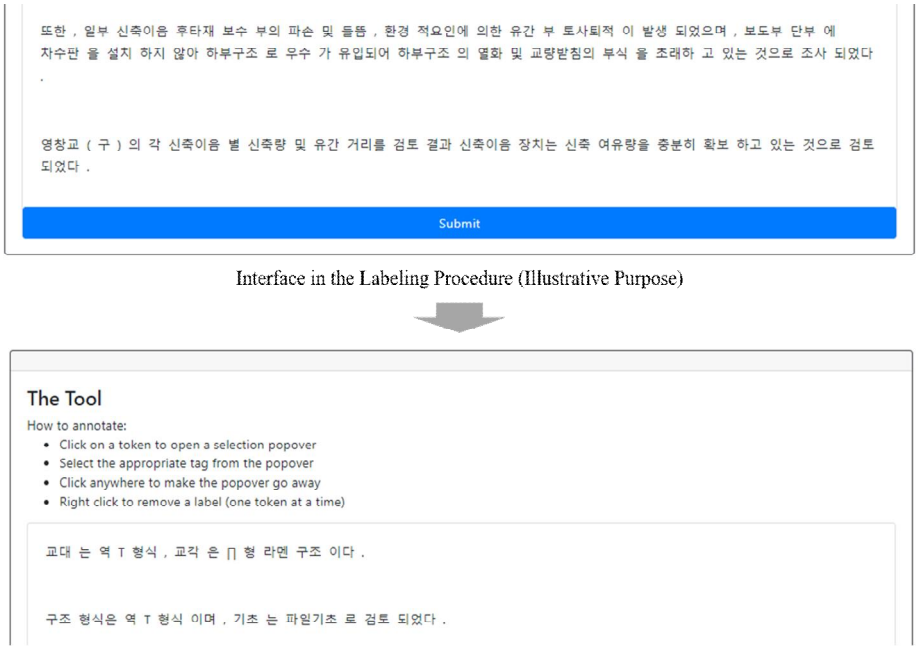


Figure 5.5 Working Process of the Labeling Interface

Fig. 5.6 shows the relationship between the number of input training data and averaged f-1 score of the trained model for the testing data. At the first iteration of the active learning, the averaged f-1 score was as low as 0.459, but it then rapidly rose up to 0.765 at 10th iteration with 100 input sentences, reached as high as 0.791 at 20th iteration, and eventually ends up with 0.816 at the end of the iteration.

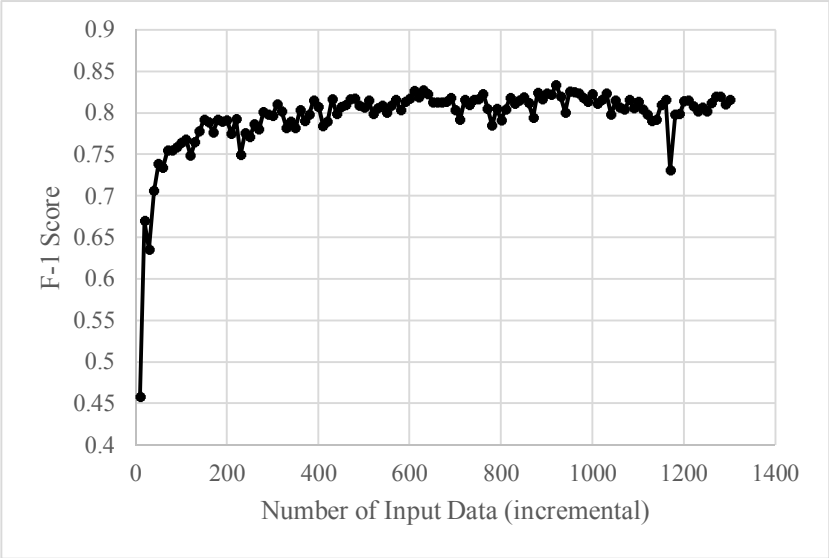


Figure 5.6 Active Learning Results

These results strongly suggest that the active learning method can effectively reduce the effort for labeling text data required to train the model. For example, using the active learning method, only 140 labeled sentences are enough to ensure the averaged f-1 score of 0.778, 90.5% of the possible maximum performance of the model, the averaged f-1 score of 0.860, when the model was trained in supervised learning as in the first experiment.

On the other hand, it was observed that increasing the amount of training data not always results in the improvement of the recognizing performance, but sometimes worsen the performance especially when a sentence with incorrectly tokenized words or typos were selected as the query. While the performance drops were not quite critical, impact of errors in input features on the model performance should be more thoroughly investigated in the

following works.

Chapter 5. Conclusion

5.1 Summary and Contributions

Information on bridge damages and causal factors in inspection reports can support predictive maintenance for bridges in early phases of projects, but tremendous amount of the reports and limited applicability of previous studies have limited the information to be extracted and analyzed for the purpose. This study therefore proposed a methodology for developing the bridge damage factor model, by applying RNN with bidirectional LSTM and active learning method. The experimental results showed the ability of the model to successfully recognize bridge elements, damages, and their factors from inspection reports, and also validated the effectiveness of active learning method to reduce the human effort to manually label all the words in the inspection reports text.

While previous studies proposed NER models for extracting information from bridge inspection reports, this study is the first one which applies a deep learning method to information extraction in construction researches as well as bridge inspection reports up to the author's knowledge. Furthermore, the active learning method used in this study would enhance the applicability of the methodology for automatic information extraction in practice, because user feedback during the model learning process is expected to give the user reliability about the recognition results by the model.

Subsequently, the methodology proposed in this study would make it

possible to analyze accumulated but unused text data in bridge inspection reports. Results of analysis on historical records of bridge damages will eventually shed light on the effects of early phases of infrastructure projects, planning, design, and construction, to the maintenance of the facility in operation and maintenance phases.

5.2 Limitations and Further Study

Three points still remain as limitations of this study. First, text extraction and preprocessing are not entirely automated, and the demand for manual processing for text data might hinder the use of the methodology and the model in practice. Second, tokenization in the proposed methodology was automated, but post-processing for the results of tokenization were not included to the methodology. Finally, since only raw text were extracted from the reports and the text were separated into individual sentences, structures of documents such as chapters, paragraphs, or tables were not considered.

Further studies should therefore include automation of the preprocessing steps to enhance the applicability of the methodology to the practice. Also, results of tokenization should be validated because it was found that wrongly tokenized words might reduce the recognition performance of the model. Meanwhile, tools or methods for utilizing information from the structure of the document or information in tables need to be further examined and applied.

Bibliography

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mane, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viegas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2016). "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems." *arXiv*:1603.04467v2

Carrillo, P., Harding, J., and Choudhary, A. (2011). "Knowledge Discovery from Post-project Review." *Construction Management and Economics*, 29(7), 713-723.

Chiu, J. P. C. and Nichols, E. (2016). "Named Entity Recognition with Bidirectional LSTM-CNNs." *Trainsitions of the Association for Computational Linguistics*, 4, 357-370.

Church, K. W. and Hanks, P. (1990). "Word Association Norms, Mutual Information, and Lexicography." *Computational Linguistics*, 16(1), 22-29.

Dekker, R. (1996). "Applications of Maintenance Optimization Models: A

Review and Analysis." *Reliability Engineering & System Safety*, 51(3), 229-240.

Gegick, M., Rotella, P., and Xie, T. (2010). "Identifying Security Bug Reports via Text Mining: An Industrial Case Study." *2010 7th IEEE Working Conference on Mining Software Repositories (MSR 2010)*, Cape Town, South Africa, 11-20.

Hochreiter, S. and Schmidhuber, J. (1997). "Long Short-Term Memory." *Neural Computation*, 9(8), 1735–1780.

Huang, Z., Xu, W., and Yu, K. (2015). "Bidirectional LSTM-CRF Models for Sequence Tagging." *arXiv*:1508.01991v1 [cs.CL]

Jeon, J. C., Lee, I. K., Park, C. H., and Hyun, L. H. (2017). "A Study on Improvement of Inspection Activity Based upon Condition Analysis of Expressway Bridges." *Journal of the Korean Society of Civil Engineers*, 37(1), 19–28.

Jeong, C-W., and Kim, J-J. (2012). "Analysis of trend in construction using textmining method." *Journal of the Korean Digital Architecture Interior Association*, 12(2), 53–60.

Jin, Z., and Tanaka-Ishii, K. (2006). "Unsupervised Segmentation of Chinese Text by Use of Branching Entropy." *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, 428–435.

Jo, H. B. (2014) "Improvement of Bridge Inspection Efficiency by the

Damage Analysis.” Master`s thesis, Korea University.

Kim, H. (2013) “Cleansing Noisy Text Using Corpus Extraction and String Match.” Master's Thesis, Seoul National University.

Kim, H. (2018). “soynlp.” *Github repository*. Retreived from https://github.com/lovit/soynlp

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016). “Neural Architectures for Named Entity Recognition.” *arXiv*:1603.01360v3 [cs.CL].

Lee, C. (2015). “Named Entity Recognition Using Long Short-Term Memory Based Recurrent Neural Network." *Korea Computer Congress 2015*, 645-647.

Lee, D., Yu, W., and Lim, H. (2017). “Bi-directional LSTM-CNN-CRF for Korean Named Entity Recognition System with Feature Augmentation.” *Journal of the Korea Convergence Society*, 8(12), 55–62.

Lee, I. K., Moon, M. K, Park, H. S., Jeon, J. C., and Lee, H. H. (2014). “Statistical Analysis of Damages in Expressway Bridges.” *Magazine of the Korea Institute for Structural Maintenance and Inspection*, 18(2), 2–9.

Lee, J-H., Yi, J-S., and Son, J. (2016). “Unstructured construction data analytics using R programming - focused on overseas construction adjudication cases.” *Journal of the Architectural Institute of Korea*

Structure & Construction, 32(5), 37−44.

Lee, S., Kim, B., Huh, M., Park, J., Kang, S., Cho, S., Lee, D., and Lee, D. (2014). "Knowledge Discovery in Inspection Reports of Marine Structures." *Expert Systems with Applications*, 41(4), 1153-1167.

Liu, K. and El-Gohary, N. (2017). "Ontology-based semi-supervised conditional random fields for automated information extraction from bridge inspection reports." *Automation in Construction*, 81, 313–327.

Lokuge, W., Gamage, N., and Setunge, S. (2016). "Fault tree analysis method for deterioration of timber bridges using an Australian case study." *Built Environment Project and Asset Management*, 6(3), 332–344.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013). "Distributed Representations of Words and Phrases and their Compositionality." *arXiv*:1310.4546v1 [cs.CL].

Ministry of Land, Infrastructure and Transportation (2016). "Special Act on The Safety Control of Public Structures."

Ministry of Land, Infrastructure and Transport (2016). "Yearbook of Road Bridges and Tunnel Statistics."

Park, E. L. and Cho, S. (2014). "KoNLPy: Korean Natural Language Processing in Python." *Proceedings of the 26th Annual Conference on Human & Cognitive Language Technology*, Chuncheon, Korea.

Peris-Sayol, G., Paya-Zaforteza, I., Balasch-Parisi, S., and Alós-Moya, J.

(2017). “Detailed Analysis of the Causes of Bridge Fires and Their Associated Damage Levels.” *Journal of Performance of Constructed Facilities*, 31(3), 04016108.

Rehurek, R., Rehurek, R., and Sojka, P. (2010). “Software Framework for Topic Modelling with Large Corpora.” *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, 45-50.

Ryu, J. M. and Shin, E. C. (2014). “Database Construction Plan of Infrastructure Safety Inspection and In-depth Inspection Results.” *Journal of Korean Geosynthetics Society*, 13(4), 133–141.

Schuster, M., and Paliwal, K. K. (1997). “Bidirectional recurrent neural networks.” *IEEE Transactions on Signal Processing*, 45(11), 2673–2681.

Settles, B. (2009). “Active learning literature survey.” *Computer Sciences Technical Report 1648*, University of Wisconsin-Madison.

Spasic, I., Ananiadou, S., McNaught, J., and Kumar, A. (2005). “Text mining and ontologies in biomedicine: Making sense of raw text.” *Briefings in Bioinformatics*, 6(3), 239–251.

Swanson, L. (2001). “Linking Maintenance Strategies to Performance.” *International Journal of Production Economics*, 70(3), 237-244.

Korea Infrastructure Safety and Technology Corporation (2017). "Specific Guidelines for Safety Inspection and Precise Safety Diagnosis."

Tanabe, L., Xie, N., Thom, L. H., Matten, W., and Wilbur, W. J. (2005).

"GENETAG: a tagged corpus for gene/protein named entity recognition." *BMC Bioinformatics*, 6(Suppl 1), 1–7.

The American Society of Civil Engineers (ASCE) (2013). "2013 Report Card for America`s Infrastructure. 2013.

Tomanek, K., and Hahn, U. (2009). "Semi-supervised active learning for sequence labeling." *Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP*, (August), 1039–1047.

Zhu, F., Patumcharoenpol, P., Zhang, C., Yang, Y., Chan, J., Meechai, A., Vongsangnak, W., and Shen, B. (2013). "Biomedical text mining and its applications in cancer research." *Journal of Biomedical Informatics*, 46(2), 200–211.

초 록

본 연구는 딥러닝을 활용하여 교량 점검보고서에서 손상 및 손상 인자를 자동으로 식별하는 방법을 제안한다. 교량 점검보고서에는 점검 결과 발견된 손상 및 원인 분석 결과가 기록되어 있다. 그러나 점검보고서의 양이 방대하여 인력으로 보고서로부터 정보를 수집하는 데 한계가 있다. 점검보고서에서 자동으로 원하는 정보를 수집하기 위해 선행연구들에서 개체명 인식 방법을 적용한 모델을 제안하였으나, 모델 학습에 필요한 훈련 데이터를 구축하는 데 인력, 시간, 비용이 상당하기 때문에 기존의 모델 학습 방법을 적용하는 데 한계점이 존재한다. 따라서 본 연구에서는 순환신경망(Recurrent neural network) 및 능동학습(Active learning) 방법을 활용하여 교량 점검보고서 텍스트로부터 손상 및 손상 인자에 해당하는 단어들을 식별할 수 있는 모델을 제안한다. 실험 결과 제안된 모델은 1)훈련 데이터에 포함된 손상 및 손상 인자 단어들을 잘 식별할 수 있을 뿐만 아니라, 전체 학습 데이터의 10% 남짓한 140개 문장만으로도 전체 데이터를 활용했을 때에 비해 90.5%에 달하는 성능(F-1 score 0.778)을 얻을 수 있는 것으로 확인되었다. 제안된 방법론은 교량 점검보고서에서 자동으로 원하는 정보를 손쉽게 얻는 데 활용될 수 있을 것이며, 궁극적으로 교량시설물에 대한 예측적 유지관리를 가능하게 할 것으로 기대된다.

주요어: 교량 점검보고서, 개체명 인식, 순환신경망, 능동학습
학 번: 2016-21272