Learning to Generate Market Comments from Stock Prices

Soichiro Murakami[†,*] **Akihiko Watanabe** [†,*] **Akira Miyazawa** [‡,¶,*] **Keiichi Goshima** [†,*]
Toshihiko Yanase [§] **Hiroya Takamura** [†,*] **Yusuke Miyao** [‡,¶,*]
[†] Tokyo Institute of Technology [‡] The Graduate University for Advanced Studies
[§] Hitachi, Ltd. [¶] National Institute of Informatics
[*] National Institute of Advanced Industrial Science and Technology
{murakami,watanabe}@lr.pi.titech.ac.jp,
goshima.k.aa@trn.dis.titech.ac.jp, takamura@pi.titech.ac.jp
toshihiko.yanase.gm@hitachi.com, {miyazawa-a,yusuke}@nii.ac.jp

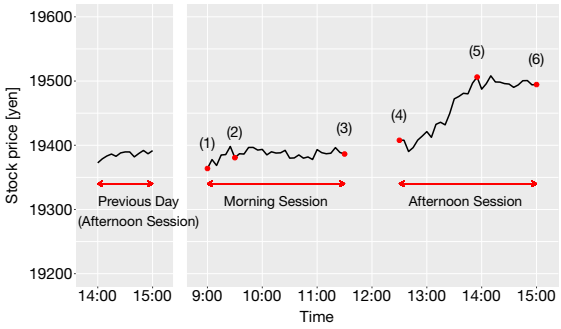
Abstract

This paper presents a novel encoder-decoder model for automatically generating market comments from stock prices. The model first encodes both short- and long-term series of stock prices so that it can mention short- and long-term changes in stock prices. In the decoding phase, our model can also generate a numerical value by selecting an appropriate arithmetic operation such as subtraction or rounding, and applying it to the input stock prices. Empirical experiments show that our best model generates market comments at the fluency and the informativeness approaching human-generated reference texts.

1 Introduction

Various industries such as finance, pharmaceuticals, and telecommunications have been increasingly providing opportunities to treat various types of large-scale numerical time-series data. Such data are hard for non-specialists to interpret in detail and time-consuming even for specialists to construe. As a result, there has been a growing interest in automatically generating concise descriptions of such data, i.e., data summarization. This interest in data summarization is encouraged by the recent development of neural network-based text generation methods. Given an appropriate architecture, a neural network can generate a sentence that is mostly grammatical and semantically reasonable.

In this study, we focus on the task of generating market comments from a time-series of stock prices. We adopt an encoder-decoder model (Sutskever et al., 2014) and exploit its capability to learn to capture the behavior of the input and generate a description of it. Although encoder-decoder models can learn to do this, they need to be provided with an appropriate network-architecture and necessary information. We use Figure 1 to illustrate the characteristic problems of comment generation for time-series of stock prices. The figure shows the Nikkei Stock Average (*Nikkei 225*, or simply *Nikkei*), which is a stock market index calculated from 225 selected issues, on some consecutive trading days accompanied by the market comments made at some specific time points in the span. The first problem is that market comments do not merely describe the increase and decrease of the price. They also often describe how the price changes compared with the previous period, such as "*continues to fall*" in (3) of Figure 1, "*turns to rise*" in (2), and "*rebound*" in (6). Market comments sometimes describe the change in price compared with the prices in the previous week. The second problem is that market comments also



	Time	Comment
(1)	09:00	Nikkei opens with a continual fall.
(2)	09:29	Nikkei turns to rise.
(3)	11:30	Nikkei continues to fall. The closing price of the morning session decreases by 5 yen to 19,386 yen.
(4)	12:30	Nikkei rises at the beginning of the afternoon session.
(5)	13:54	Nikkei gains more than 100 yen.
(6)	15:00	Nikkei rebounds and closes up 102 yen to 19,494 yen.

Figure 1: Nikkei 225 and market comments.

Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, pages 1374–1384
Vancouver, Canada, July 30 - August 4, 2017.
https://doi.org/10.18653/v1/P17-1126

contain expressions that depend on their delivery time: e.g., "*opens with*" in (1), "*closing price of the morning session*" in (3), and "*beginning of the afternoon session*" in (4). The third problem is that market comments typically contain numerical values, which often cannot be copied from the input prices. Such numerical values probably cannot be generated as other words are generated by the standard decoder. This difficulty can be easily understood as analogous with the difficulty of generating named entities by encoder-decoder models. To derive such values, the model needs arithmetic operations such as subtraction as in examples (3) and (6) mentioning the difference in price and rounding as in example (5).

To address these problems, we present a novel encoder-decoder model to automatically generate market comments from stock prices. To address the first problem of capturing various types of change in different time scales, the model first encodes data consisting of both short- and long-term time-series, where a multi-layer perceptron, a recurrent neural network, or a convolutional network is adopted as a basic encoder. In the decoding phase, we feed our model with the delivery time of the market comment to generate the expressions depending on time of day to address the second problem. To address the third problem regarding with numerical values mentioned in the generated text, we allow our model to choose an arithmetic operation such as subtraction or rounding instead of generating a word.

The proposed methods are evaluated on the task of generating Japanese market comments on the Nikkei Stock Average. Automatic evaluation with BLEU score (Papineni et al., 2002) and F-score of time-dependent expressions reveals that our model outperforms a baseline encoder-decoder model significantly. Furthermore, human assessment and error analysis prove that our best model generates characteristic expressions discussed above almost perfectly, approaching the fluency and the informativeness of human-generated market comments.

2 Related Work

The task of generating descriptions from time-series or structured data has been tackled in various domains such as weather forecasts (Belz, 2007; Angeli et al., 2010), healthcare (Portet et al., 2009; Banaee et al., 2013b), and sports (Liang et al., 2009). Traditionally, many studies used hand-crafted rules (Goldberg et al., 1994; Dale et al., 2003; Reiter et al., 2005). On the other hand, interest has recently been growing in automatically learning a correspondence relationship from data to text and generating a description of this relationship since large-scale data in diversified formats have become easy to acquire. In fact, a data-driven approach has been extensively studied nowadays for various tasks such as image caption generation (Vinyals et al., 2015) and weather forecast generation (Mei et al., 2016b).

The task, called data-to-text or concept-to-text, is generally divided into two subtasks: content selection and surface realization. Whereas previous studies tackled the subtasks separately (Barzilay and Lapata, 2005; Wong and Mooney, 2007; Lu et al., 2009), recent work has focused on solving them jointly using a single framework (Chen and Mooney, 2008; Kim and Mooney, 2010; Angeli et al., 2010; Konstas and Lapata, 2012, 2013).

More recently, there has been some work on an encoder-decoder model (Sutskever et al., 2014) for generating a description from time-series or structured data to solve the subtasks jointly in a single framework, and this model has been proven to be useful (Mei et al., 2016b; Lebret et al., 2016). However, the task of generating a description from numerical time-series data presents difficulties such as the second and third problems mentioned in Section 1. For the second problem, the model needs to be fed with information on delivery time. Also, the model needs arithmetic operations such as subtraction for the third problem because even if we simply apply a copy mechanism (Gu et al., 2016; Gulcehre et al., 2016) to the model, it cannot derive a calculated value such as (3), (5), or (6) in Figure 1 from input. Thus, in this work, we tackle these problems and develop a model on the basis of the encoder-decoder model that can mention a specific numerical value by referring to the input data or producing a processed value with mathematical calculation and mention time-dependent expressions by incorporating the information on delivery time into its decoder.

There has also been some work on generating market comments. Kukich (1983) developed a system consisting of rule-based components for generating stock reports from a database of daily stock quotes. Although she used several components individually and had to define a number of rules for the generation, our encoder-decoder model can

perform it with fewer and simpler rules for the calculation. Aoki and Kobayashi (2016) developed a method on the basis of a weighted bi-gram language model for automatically describing trends of time-series data such as the Nikkei Stock Average. However, they did not attempt to refer to specific numerical values such as closing prices and amounts of rises in price although such descriptions are often used in market comments as shown in Figure 1 (3), (5), and (6). In contrast, we present a novel approach to generate natural language descriptions of time-series data that can not only able to describe trends of the data but also mention specific numerical values by referring to the time-series data.

3 Generating Market Comments

To generate market comments on stock prices, we introduce an encoder-decoder model. Encoder-decoder models have been widely used and proven useful in various tasks of natural language generation such as machine translation (Cho et al., 2014) and text summarization (Rush et al., 2015). Our task is similar to these tasks in that the system takes sequential data and generates text. Therefore, it is natural to use an encoder-decoder model in modeling stock prices.

Figure 2 illustrates our model. In describing time-series data, the model is expected to capture various types of change and important values in the given sequence, such as absolute or relative changes and maximum or minimum value, in different time-scales. Moreover, it is necessary to generate time-dependent comments and numerical values that require arithmetic operations for derivation, such as "The closing price of *the morning session* decreases by *5 yen*...". To achieve these, we present three strategies that alter the standard encoder-decoder model.

First (Section 3.1), we use several encoding methods for time-series data, as in (1) of Figure 2, to capture the changes and important values. Second (Section 3.2), we incorporate delivery-time information into the decoder, as in (2) of Figure 2, to generate time-dependent comments. For the decoder, we use a recurrent neural network language model (RNNLM) (Mikolov et al., 2010), which is widely used in language generation tasks. Finally (Section 3.3), we extend the decoder to estimate arithmetic operations, as in (3) of Figure 2, to generate numerical values in market comments.

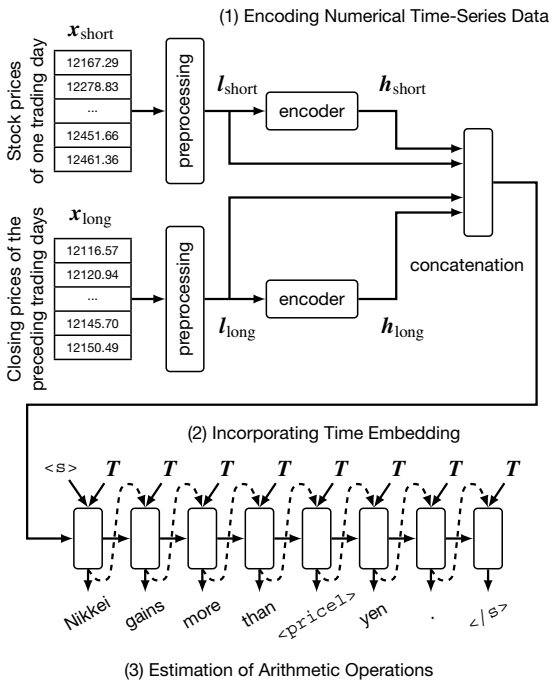


Figure 2: Overview of our model. Here $\boldsymbol{l}_{\text{short}}$ and $\boldsymbol{l}_{\text{long}}$ represent two vectors of preprocessed values, and $\boldsymbol{h}_{\text{short}}$ and $\boldsymbol{h}_{\text{long}}$ indicate hidden states of the encoder. $\boldsymbol{T}$ represents a time embedding vector.

3.1 Encoding Numerical Time-Series Data

We prepare short- and long-term data, using the five-minute chart of Nikkei 225. A vector for short-term data consists of the prices of one trading day and has N elements. We denote it as $\boldsymbol{x}_{\text{short}} = \left(x_{\text{short},\,i}\right)_{i=0}^{N-1}$. On the other hand, a vector for long-term data consists of the closing prices of the M preceding trading days. It is denoted as $\boldsymbol{x}_{\text{long}} = \left(x_{\text{long},\,i}\right)_{i=0}^{M-1}$.

Data are commonly preprocessed to remove noise and enhance generalizability of a model (Zhang and Qi, 2005; Banaee et al., 2013a). We use two preprocessing methods: *standardization* and *moving reference*. Standardization substitutes each element x_i of input $\boldsymbol{x}$ by

$$x_i^{\text{std}} = \frac{x_i - \mu}{\sigma}, \tag{1}$$

where μ and σ are the mean and standard deviation of the values in the training data, respectively. Standardized values are less affected by scale. The second method, moving reference (Freitas et al., 2009), substitutes each element x_i of input $\boldsymbol{x}$ by

$$x_i^{\text{move}} = x_i - r_i, \tag{2}$$

where r_i is the closing price of the previous trading day of $\boldsymbol{x}$. This is introduced to capture price fluctuations from the previous day.

By applying one of the preprocessing methods to $\boldsymbol{x}_{\text{short}}$ and $\boldsymbol{x}_{\text{long}}$, we obtain two vectors of preprocessed values $\boldsymbol{l}_{\text{short}}$ and $\boldsymbol{l}_{\text{long}}$. Given these, each encoder emits the corresponding hidden states $\boldsymbol{h}_{\text{short}}$ and $\boldsymbol{h}_{\text{long}}$. After obtaining the hidden states, we concatenate the two vectors of the preprocessed values and the outputs of the encoders as a *multi-level representation* of the input time-series data. The multi-level representation is an approach developed by Mei et al. (2016a) that enable the decoder to take into account both the high-level representation, e.g., $\boldsymbol{h}_{\text{short}}$, $\boldsymbol{h}_{\text{long}}$, and the low-level representation, e.g., $\boldsymbol{l}_{\text{short}}$, $\boldsymbol{l}_{\text{long}}$, at the same time. They have shown that it improves performance in terms of selecting salient objects in input data. We thus set the initial hidden state $\boldsymbol{s}_0$ of the decoder as

$$\boldsymbol{s}_0 = \boldsymbol{l}_{\text{short}} \oplus \boldsymbol{l}_{\text{long}} \oplus \boldsymbol{h}_{\text{short}} \oplus \boldsymbol{h}_{\text{long}}, \quad (3)$$

where $\oplus$ is the concatenation operator.

When we use both preprocessing methods, we have four preprocessed input vectors: $\boldsymbol{l}^{\text{move}}_{\text{short}}$, $\boldsymbol{l}^{\text{std}}_{\text{short}}$, $\boldsymbol{l}^{\text{move}}_{\text{long}}$, and $\boldsymbol{l}^{\text{std}}_{\text{long}}$. In this case, we introduce four encoders, and set the initial hidden state $\boldsymbol{s}_0$ of the decoder as

$$\begin{aligned}\boldsymbol{s}_0 = {} & \boldsymbol{l}^{\text{move}}_{\text{short}} \oplus \boldsymbol{l}^{\text{std}}_{\text{short}} \oplus \boldsymbol{l}^{\text{move}}_{\text{long}} \oplus \boldsymbol{l}^{\text{std}}_{\text{long}} \\ & \oplus \boldsymbol{h}^{\text{move}}_{\text{short}} \oplus \boldsymbol{h}^{\text{std}}_{\text{short}} \oplus \boldsymbol{h}^{\text{move}}_{\text{long}} \oplus \boldsymbol{h}^{\text{std}}_{\text{long}}. \quad (4)\end{aligned}$$

Since several encoding methods can be used for the time-series data, we use any one of the three conventional neural networks: Multi-Layer Perceptron (MLP), Convolutional Neural Network (CNN), or Recurrent Neural Network (RNN) with Long Short-Term Memory cells (Hochreiter and Schmidhuber, 1997). In the experiments, we empirically evaluate and compare the encoding methods.

3.2 Incorporating Time Embedding

Even if identical sequences of values are observed, comments usually vary in accordance with price history or the time they are observed. For instance, when the market opens, comments usually mention how much the stock price has increased or decreased compared with the closing price of the previous trading day, as in (1) and (3) in Figure 1.

Our model creates vectors called *time embedding* vectors $\boldsymbol{T}$ on the basis of the time when the comment is delivered (e.g., 9:00 a.m. or 3:00 p.m.). Then a time embedding vector is added to each hidden state $\boldsymbol{s}_j$ in decoding so that words are generated depending on time. This mechanism is inspired by *speaker embedding* introduced by Li et al. (2016). They use an encoder-decoder model for a conversational agent that inherits the characteristics of a speaker, such as his/her manner of speaking. They encode speaker-specific information (e.g., dialect, age, and gender) into speaker embedding vectors and used them in decoding.

3.3 Estimation of Arithmetic Operations

Text generation systems based on language models such as RNNLM often generate erroneous words for named entities; that is, they often mention a similar but incorrect entity, e.g., Nissan for Toyota. To overcome this problem, Gulcehre et al. (2016) developed a text generation method called *copy mechanism*. The method copies rare words missing from the vocabulary from a given sequence of words using an attention mechanism, and emits the copied words.

Market comments often mention numerical values that appear in the input data, but they also mention values obtained through arithmetic operations, such as differences in prices as in (3) and (6) in Figure 1, or rounded values as in (5). Thus, another problem arises: what *type of operation* is suitable for text to be generated? In this work, we solve this problem by extending the idea of copy mechanism.

To enable our model to generate text with values calculated from input values, we add *generalization tags* to the vocabulary used in the model. Each generalization tag represents a type of arithmetic operation. When a generalization tag is emitted, the model performs the operations on the designated values in accordance with the tag, replaces the tag with the calculated value, and finally outputs text containing numerical values. For preprocessing, we replace each numerical value appearing in the market comments in the training data with generalization tags such as `<price1>`. The tag for a numerical value depends on what the value stands for in the text. Table 1 displays all the tags and the corresponding types of calculation. To illustrate, suppose a market comment says

(a) *Nikkei rebounds. The closing price of the morning session is 16,610 yen, which is 227 yen higher.*

Since this comment omits the phrase "than the

Tag	Arithmetic operation
<price1>	Return Δ
<price2>	Round down Δ to the nearest 10
<price3>	Round down Δ to the nearest 100
<price4>	Round up Δ to the nearest 10
<price5>	Round up Δ to the nearest 100
<price6>	Return z as it is
<price7>	Round down z to the nearest 100
<price8>	Round down z to the nearest 1,000
<price9>	Round down z to the nearest 10,000
<price10>	Round up z to the nearest 100
<price11>	Round up z to the nearest 1,000
<price12>	Round up z to the nearest 10,000

Table 1: Generalization tags and corresponding arithmetic operations. Here z and Δ stand for latest price and difference between z and closing price of previous trading day.

closing price of the previous day", 227 in this example indicates the difference between the closing price of the previous trading day $x_{\text{long},\,M-1}$ and the latest price $x_{\text{short},\,N-1}$ denoted by z in Table 1. Therefore, we replace 227 with the tag <price1>. Likewise, we replace 16,610 with <price6> because it represents the latest price z. To find the optimal tag for each value, we try all the types of operations listed in Table 1 using the values appearing in the text, i.e., 227 and 16,610 in this case. Then, we select the tag that has the operation that yields the value closest to the original one.

In prediction, the model first generates a tentative comment, which includes tags as well as words. Suppose that the input vectors are $\boldsymbol{x}_{\text{short}}$ and $\boldsymbol{x}_{\text{long}}$, with $x_{\text{short},\,N-1} = 14508$ and $x_{\text{long},\,M-1} = 14612$, and that the model generates the comment below:

(b) *Nikkei opens turning down. The loss exceeds* <price2> *yen, and it falls to the* <price7> *yen level.*

Since the tag <price2> represents "the difference between $x_{\text{short},\,N-1}$ and $x_{\text{long},\,M-1}$ rounded down to the nearest 10", we replace the tag with 100. Similarly, we replace <price7>, which is "the last price $x_{\text{short},\,N-1}$ rounded down to the nearest 100", with 14,500. Finally, we have a market comment containing the numbers as below:

(c) *Nikkei opens turning down. The loss exceeds 100 yen, and it falls to the 14,500 yen level.*

4 Experiments

4.1 Experimental Settings

We used the five-minute chart of Nikkei 225 from March 2013 to October 2016 as numerical time-series data, which were collected from IBI-Square Stocks[1], and 7,351 descriptions as market comments, which are written in Japanese and provided by Nikkei QUICK News. We divided the dataset into three parts: 5,880 for training, 730 for validation, and 741 for testing. For a human evaluation, we randomly selected 100 comments and their time-series data included in the test set.

We set $N = 62$, which is the number of time steps for stock prices for one trading day, and $M = 7$, which is the number of the time steps for closing prices of the preceding trading days. We used Adam (Kingma and Ba, 2015) for optimization with a learning rate of 0.001 and a mini-batch size of 100. The dimensions of word embeddings, time embeddings, and hidden states for both the encoder and decoder are set to 128, 64, and 256, respectively. For CNN, we used a single convolutional layer and set the filter size to 3.

In the experiments, we conducted three types of evaluation: two for automatic evaluation, and one for human evaluation. For one automatic evaluation, we used BLEU (Papineni et al., 2002) to measure the matching degree between the market comments written by humans as references and output comments generated by our model. We applied paired bootstrap resampling (Koehn, 2004) for a significance test. For the other automatic evaluation metric, we calculate F-measures for time-dependent expressions, using market comments written by humans as references, to investigate whether our model can correctly output time-dependent expressions such as "*open with*" and describe how the price changes compared with the previous period referring to the series of preceding prices such as "*continual fall*". Specifically, we calculate F-measures for 13 expressions shown in Figure 3.

For the human evaluation, we recruited a specialist in financial engineering as a judge to evaluate the quality of generated market comments. To evaluate the difference in the quality of generated comments between our models and human, we showed both system-generated and human-generated market comments together with their

[1] http://www.ibi-square.jp/index.htm

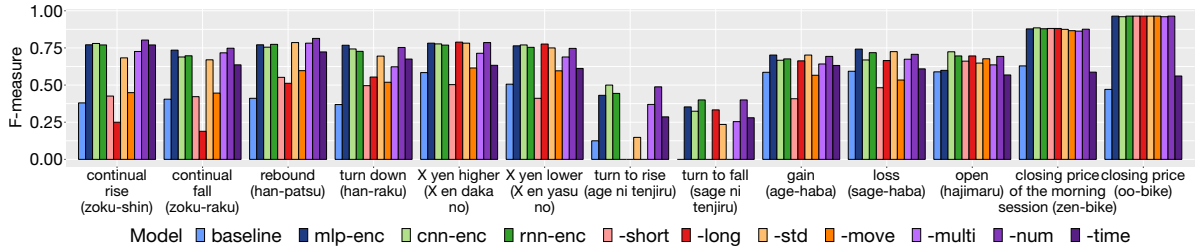


Figure 3: F-measure values for the expressions on the test set. Each expression is accompanied by its original Japanese expression transliterated into English alphabet in parenthesis. Out of the 13 expressions, 10 on the left are expressions that describe how the price changes compared with the previous period, and 3 on the right are time-dependent expressions.

Model		baseline	mlp-enc	cnn-enc	rnn-enc	-short	-long	-std	-move	-multi	-num	-time
Encoder		MLP	MLP	CNN	RNN	MLP	MLP	MLP	MLP	MLP	MLP	MLP
Input data	$\boldsymbol{x}_{\text{short}}$	✓	✓	✓	✓	–	✓	✓	✓	✓	✓	✓
	$\boldsymbol{x}_{\text{long}}$	–	✓	✓	✓	✓	–	✓	✓	✓	✓	✓
Preprocessing	Standardization	✓	✓	✓	✓	✓	✓	–	✓	✓	✓	✓
	Moving reference	✓	✓	✓	✓	✓	✓	✓	–	✓	✓	✓
	Multi-level	–	✓	✓	✓	✓	✓	✓	✓	–	✓	✓
	Arithmetic operation	–	✓	✓	✓	✓	✓	✓	✓	✓	–	✓
	Time-embedding	–	✓	✓	✓	✓	✓	✓	✓	✓	✓	–

Table 2: Overview of the models we used in the experiments.

time-series data consisting of $\boldsymbol{x}_{\text{short}}$ and $\boldsymbol{x}_{\text{long}}$, without letting the judge know which comment is generated by which method. We asked the judge to give each market comment two scores: one for informativeness and one for fluency. Both scores have two levels, 0 or 1, where 1 indicates high informativeness or fluency. For informativeness, the judge used both generated comments and their input stock prices to rate the comments. Specifically, if the judge deem that a generated comment describes an important price movement or an outline of the movement properly, such comments are considered to be informative. For fluency, the judge read only the generated comments and rate them in terms of readability, regardless of their content of the comment.

In addition, since some of the market comments written by humans sometimes include external information such as "*Nikkei opens with a continual fall as yen pressures exporters*", we also asked the judge to ignore the correctness of external information mentioned in comments, for the sake of fairness in comparison, because external information cannot be retrieved from the time-series data.

To assess the effectiveness of the techniques we introduced, we conducted experiments with 11 models. Table 2 shows an overview of the models we compared. We compared three types of models: a baseline, full models (e.g., *mlp-enc*), and ablated models (e.g., *-short*). For example, *-short* is a model that does not use the short-term time series.

Model	baseline	mlp-enc	cnn-enc	rnn-enc	-short	-long
BLEU	0.243	**0.464**	0.449	0.454	0.380	0.433

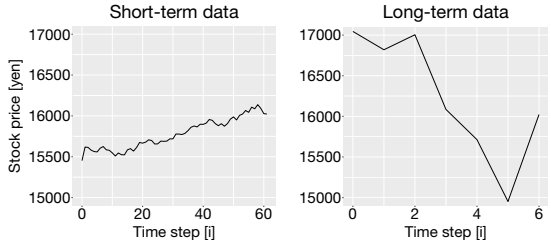
Model	-std	-move	-multi	-num	-time
BLEU	0.455	0.393	0.435	0.318	0.395

Table 3: BLEU scores on the test set. Differences between the best model, *mlp-enc*, and other models are statistically significant at $p < 0.05$.

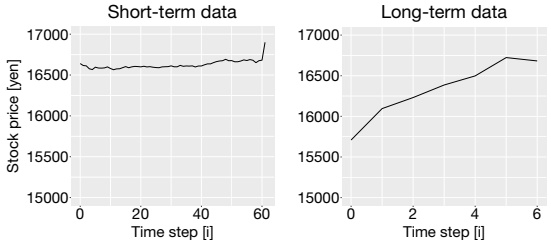
4.2 Results

Table 3 shows the BLEU scores on the test set. Figure 3 presents the F-measure of the models for each phrase. We also present output examples with human-generated market comments (Human) for reference in Figure 4.

In the results for the automatic evaluation in BLEU, the model using both MLP as encoders and all the techniques we developed, *mlp-enc*, outperformed *baseline* and the other models. The BLEU scores and F-measure values revealed differences among the models using MLP, CNN, or RNN (*mlp-*



(a) Price movements of Nikkei on Feb. 15, 2016.



(b) Price movements of Nikkei on July 21, 2016.

Model	F	I	Generated comment
baseline	1	0	Nikkei heikin, han-patsu zen-bike wa 81 en daka no <unk> en Nikkei average, rebound-pop first_half-closing TOP 81 yen higher GEN <unk> yen *Nikkei rebounds. The closing price of the morning session is <unk> yen, which is 81 yen higher.*
mlp-enc	1	1	Nikkei heikin, oo-haba han-patsu oo-bike wa 1,069 en daka no 16,022 en Nikkei average, big-range rebound-pop big-closing TOP 1,069 yen higher GEN 16,022 yen *Nikkei significantly rebounds. The closing price is 16,022 yen, which is 1,069 yen higher.*
human	1	1	Nikkei heikin, oo-haba han-patsu oo-bike wa 1,069 en daka no 16,022 en Nikkei average, big-range rebound-pop big-closing TOP 1,069 yen higher GEN 16,022 yen *Nikkei significantly rebounds. The closing price is 16,022 yen, which is 1,069 yen higher.*

(c) Comments on price at 3:00 p.m. on February 15, 2016.

Model	F	I	Generated comment
baseline	1	0	Nikkei heikin, zoku-shin de hajimaru age-haba 100 en koeru Nikkei average, continual-advance INSTR open-IMPERF raise-range 100 yen exceed-IMPERF *Nikkei opens with a continual rise. The gain exceeds 100 yen.*
mlp-enc	1	1	Nikkei heikin, age-haba 200 en koeru Nikkei average, raise-range 200 yen exceed-IMPERF *Nikkei gains more than 200 yen.*
human	1	1	Nikkei heikin, age-haba 200 en kosu Nikkei average, raise-range 200 yen exceed-IMPERF *Nikkei gains more than 200 yen.*

(d) Comments on price at 9:00 a.m. on July 21, 2016.

Figure 4: Examples of short- and long-term movements of Nikkei, and comments models made on them, where <unk> represents an unknown word. Columns F and I show scores on fluency and informativeness in human evaluation. Each example is accompanied by original Japanese comment transliterated into English alphabet, its literal translation, and the corresponding English sentence. Abbreviations used here are as follows. TOP: topic case, GEN: genitive case, INSTR: instrumental case, and IMPERF: imperfect form of a verb.

enc, *cnn-enc*, *rnn-enc*). In the comparison between the models that took two types of the time-series data $\boldsymbol{x}_{\text{short}}$, $\boldsymbol{x}_{\text{long}}$ as input (e.g., *mlp-enc* or *rnn-enc*) and the models that only used one of them (*-short*, *-long*), the models using both types of data such as *mlp-enc* and *rnn-enc* gained higher BLEU scores than *-short* and *-long*. Also, the models that encoded the two types of time-series data to capture their short- and long-term changes correctly output more expressions that described the changes such as "*turn to rise*", "*continue to fall*", and "*rebound*" than *-short* and *-long* as shown in Figure 3.

According to the comparison between preprocessing methods, *mlp-enc*, which used both standardization and moving reference as preprocessing methods, obtained a higher BLEU score than the models that used neither (*-std*, *-move*). In terms of the F-measure values, *mlp-enc* output phrases mentioning changes more appropriately and therefore achieved the higher values than the other two models as in "turn to rise" or "turn to fall" in Figure 3. Furthermore, we found that the BLEU score of *-multi*, which did not use the multi-level representation of the data, was inferior. In other words, incorporating the multi-level representation along with an output of an encoder into a decoder seems

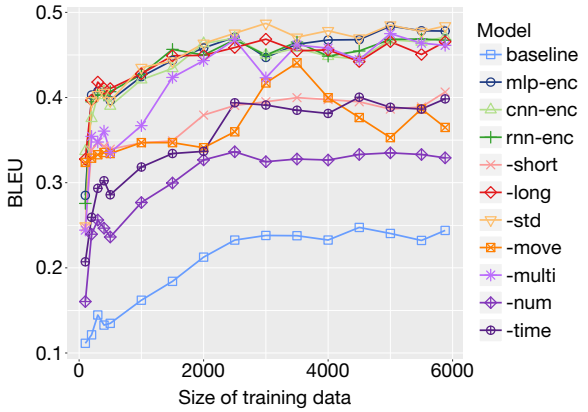


Figure 5: BLEU scores of market comments generated by models for each size of training data on the validation set.

to contribute to improving the automatic evaluation and producing a better representation of the input data.

baseline and *-num* output numerical values as "words" from the vocabulary for RNNLM because these models do not use any arithmetic operation. Therefore, there were many cases including <unk> that should be output as a numerical value as shown in Figure 4 (a). We found that *-num* had a lower BLEU score than the models such as *mlp-enc* and *-std* that used arithmetic operations. Furthermore, we observed that the models with arithmetic operations correctly generated stock prices in most cases.

By comparing *-time*, which did not incorporate time-embeddings into a decoder, and other models such as *mlp-enc* with respect to the F-measure of expressions depending on delivery time (e.g., "*open with*" or "*closing session*"), we found that the models that took time information into account, such as *mlp-enc*, generated those phrases more accurately than *-time*.

Moreover, we analyzed the effect of different sizes of training data. Figure 5 shows BLEU scores of market comments generated by our models for each size of training data on the validation set. According to the results, we found that the BLEU scores for the models saturated when we used 3000 training data. In addition, there was not much difference in convergence speed among the models.

The human evaluation results in Table 4 indicate that market comments generated by our model (*mlp-enc*) achieved a quality comparable even to that of market comments written by humans. Moreover, we found that *mlp-enc* significantly outperformed *baseline* in terms of informativeness but was outperformed by *baseline* in terms of fluency. The reason was that *mlp-enc* occasionally generated a market comment such as "*Nikkei gains more than 0 yen*" because of an error in the prediction of the operation, and such comments were not considered not to be fluent or informative by the judge, although most of comments generated by *mlp-enc* were as fluent as those of *baseline*. Note that *baseline* does not generate expressions like "*0 yen*" because they are not normally used in market comments and so not included in the vocabulary. Therefore, the judge considered all the comments generated by *baseline* to be fluent.

Model	Informativeness	Fluency	External
Human	**95**	95	25
mlp-enc	85	93	1
baseline	28	**100**	6

Table 4: Results of human evaluation. Each score indicates number of market comments judged to be level-1. External shows number of market comments including external information.

For another possibility to enhance our model, we have to consider that the model should mention a difference or gain for a duration from when to when. For example, our current model sometimes generated a market comment such as "*Nikkei gains more than 200 yen*", although Nikkei actually gained more than 300 yen. Such a comment is not incorrect but is imprecise. Therefore, we consider that a mechanism is needed to select the period to be mentioned when the model generates a comment to this problem and increase the generalizability of our model for generating a description from various time-series data.

5 Conclusion and Future Work

In this study, we presented a novel encoder-decoder model to automatically generate market comments from numerical time-series data of stock prices, using the Nikkei Stock Average as an example. Descriptions of numerical time-series data written by humans such as market comments have several writing style characteristics. For example, (1) content to be mentioned in the market comments varies depending on short- or long-term changes of the time-series data, (2) expressions depending on delivery time at which text is written are used, and (3) numerical values obtained through arith-

metic operations applied to the input data are often described. We developed approaches for generating comments that have these characteristics and showed the effectiveness of the proposed model.

In future work, we plan to apply our model to descriptions of time-series data in various domains such as weather forecasts and sports, which share the above writing-style characteristics. We also plan to use multiple time-series as input such as multiple brands of stock.

Acknowledgements

This paper is based on results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

References

Gabor Angeli, Percy Liang, and Dan Klein. 2010. A simple domain-independent probabilistic approach to generation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 502–512. http://aclweb.org/anthology/D10-1049.

Kasumi Aoki and Ichiro Kobayashi. 2016. Linguistic summarization using a weighted n-gram language model based on the similarity of time-series data. In *Proceedings of IEEE International Conference on Fuzzy Systems*. pages 595–601. https://doi.org/10.1109/FUZZ-IEEE.2016.7737741.

Hadi Banaee, Mobyen Uddin Ahmed, and Amy Loutfi. 2013a. A framework for automatic text generation of trends in physiological time series data. In *Processing of IEEE International Conference on Systems, Man, and Cybernetics*. pages 3876–3881. https://doi.org/10.1109/SMC.2013.661.

Hadi Banaee, Mobyen Uddin Ahmed, and Amy Loutfi. 2013b. Towards NLG for physiological data monitoring with body area networks. In *Proceedings of the 14th European Workshop on Natural Language Generation*. Association for Computational Linguistics, pages 193–197. http://aclweb.org/anthology/W13-2127.

Regina Barzilay and Mirella Lapata. 2005. Collective content selection for concept-to-text generation. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*. pages 331–338. http://aclweb.org/anthology/H05-1042.

Anja Belz. 2007. Probabilistic generation of weather forecast texts. In *Proceedings of the 2007 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pages 164–171. http://aclweb.org/anthology/N07-1021.

David L. Chen and Raymond J. Mooney. 2008. Learning to sportscast: A test of grounded language acquisition. In *Proceedings of the 25th international conference on Machine learning*. pages 128–135. https://doi.org/10.1145/1390156.1390173.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 1724–1734. https://doi.org/10.3115/v1/D14-1179.

Robert Dale, Sabine Geldof, and Jean-Philippe Prost. 2003. CORAL: Using natural language generation for navigational assistance. In *Proceedings of the 26th Australasian Computer Science Conference*. pages 35–44. http://dl.acm.org/citation.cfm?id=783106.783111.

Fabio D. Freitas, Alberto F. De Souza, and Ailson R. de Almeida. 2009. Prediction-based portfolio optimization model using neural networks. *Neurocomputing* 72(10):2155–2170. https://doi.org/10.1016/j.neucom.2008.08.019.

Eli Goldberg, Norbert Driedger, and Richard I. Kittredge. 1994. Using natural-language processing to produce weather forecasts. *IEEE Expert* 9(2):45–53. https://doi.org/10.1109/64.294135.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 1631–1640. https://doi.org/10.18653/v1/P16-1154.

Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the unknown words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 140–149. https://doi.org/10.18653/v1/P16-1014.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Joohyun Kim and Raymond J. Mooney. 2010. Generative alignment and semantic parsing for learning from ambiguous supervision. In *Proceedings of the 23rd International Conference on Computational Linguistics*. pages 543–551. http://aclweb.org/anthology/C10-2062.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*. https://arxiv.org/abs/1412.6980.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 388–395. http://aclweb.org/anthology/W04-3250.

Ioannis Konstas and Mirella Lapata. 2012. Unsupervised concept-to-text generation with hypergraphs. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. pages 752–761. http://aclweb.org/anthology/N12-1093.

Ioannis Konstas and Mirella Lapata. 2013. Inducing document plans for concept-to-text generation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 1503–1514. http://aclweb.org/anthology/D13-1157.

Karen Kukich. 1983. Design of a knowledge-based report generator. In *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 145–150. http://aclweb.org/anthology/P83-1022.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 1203–1213. https://doi.org/10.18653/v1/D16-1128.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016. A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 994–1003. https://doi.org/10.18653/v1/P16-1094.

Percy Liang, Michael Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *Proceedings of Association for Computational Linguistics and International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, pages 91–99. http://aclweb.org/anthology/P09-1011.

Wei Lu, Hwee Tou Ng, and Wee Sun Lee. 2009. Natural language generation with tree conditional random fields. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 400–409. http://aclweb.org/anthology/D09-1042.

Hongyuan Mei, Mohit Bansal, and Matthew R. Walter. 2016a. Listen, attend, and walk: Neural mapping of navigational instructions to action sequences. In *Proceedings of Association for the Advancement of Artificial Intelligence*. https://arxiv.org/abs/1506.04089.

Hongyuan Mei, Mohit Bansal, and Matthew R. Walter. 2016b. What to talk about and how? selective generation using lstms with coarse-to-fine alignment. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pages 720–730. https://doi.org/10.18653/v1/N16-1086.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proceedings of the 11th Annual Conference of the International Speech Communication Association*. International Speech Communication Association, 9, pages 1045–1048. http://www.isca-speech.org/archive/interspeech_2010/i10_1045.html.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 311–318. http://aclweb.org/anthology/P02-1040.

François Portet, Ehud Reiter, Albert Gatt, Jim Hunter, Somayajulu Sripada, Yvonne Freer, and Cindy Sykes. 2009. Automatic generation of textual summaries from neonatal intensive care data. *Artificial Intelligence* 173(7-8):789–816. https://doi.org/10.1016/j.artint.2008.12.002.

Ehud Reiter, Somayajulu Sripada, Jim Hunter, Jin Yu, and Ian Davy. 2005. Choosing words in computer-generated weather forecasts. *Artificial Intelligence* 167(1-2):137–169. https://doi.org/10.1016/j.artint.2005.06.006.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 379–389. https://doi.org/10.18653/v1/D15-1044.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*. pages 3104–3112. https://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural

image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pages 3156–3164. https://arxiv.org/abs/1411.4555.

Yuk Wah Wong and Raymond Mooney. 2007. Generation by inverting a semantic parser that uses statistical machine translation. In *Proceedings of the 2007 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pages 172–179. http://aclweb.org/anthology/N07-1022.

G. Peter Zhang and Min Qi. 2005. Neural network forecasting for seasonal and trend time series. *European journal of operational research* 160(2):501–514. https://doi.org/10.1016/j.ejor.2003.08.037.