Synthetic Data Generation and Multi-Task Learning for Extracting Temporal Information from Health-Related Narrative Text

Heereen Shim[1,2,3], Dietwig Lowet[3], Stijn Luca[4] and Bart Vanrumste[1,2]
[1]Campus Group T, e-Media Research Lab, KU Leuven, Leuven, Belgium
[2]Department of Electrical Engineering (ESAT), STADIUS, KU Leuven, Leuven, Belgium
[3]Philips Research, Eindhoven, the Netherlands
[4]Department of Data Analysis and Mathematical Modelling, Ghent University, Ghent, Belgium
{heereen.shim, bart.vanrumste}@kuleuven.be
{dietwig.lowet}@philips.com
{stijn.luca}@ugent.be

Abstract

Extracting temporal information is critical to process health-related text. Temporal information extraction is a challenging task for language models because it requires processing both texts and numbers. Moreover, the fundamental challenge is how to obtain a large-scale training dataset. To address this, we propose a synthetic data generation algorithm. Also, we propose a novel multi-task temporal information extraction model and investigate whether multi-task learning can contribute to performance improvement by exploiting additional training signals with the existing training data. For experiments, we collected a custom dataset containing unstructured texts with temporal information of sleep-related activities. Experimental results show that utilising synthetic data can improve the performance when the augmentation factor is 3. The results also show that when multi-task learning is used with an appropriate amount of synthetic data, the performance can significantly improve from 82. to 88.6 and from 83.9 to 91.9 regarding micro-and macro-average exact match scores of normalised time prediction, respectively.

1 Introduction

Extracting temporal information from text is important linguistic skill to process health-related text. Also, there are a lot of potential applications of temporal information extraction in the health-related domain, including forecasting treatment effect (Choi et al., 2016), early detecting diseases (Khanday et al., 2020), and tracking treatment progress (Demner-Fushman et al., 2021). With the recent trends of telehealth, an automated system that can extract temporal information from the health-related narrative text can provide benefits to not only healthcare professionals but also recipients enabling active engagement, such as self-monitoring.

I went to bed around 11 pm. Used the phone for around 15 minutes and after that switched the light off. It took around 30 minutes to fall asleep. My sleep was disturbed at 5:45 am. and I spend in the bed for other 45 minutes to get sleep. I got off the bed around 6:30 am. Overall the sleep was normal. I felt refreshed.

Event	Time
Bed time	23:00
Lights off time	23:15
Sleep time	23:45
Sleep disturbance	05:45
Duration of disturbance	00:45
Out of bed time	06:30

Figure 1: Example of free-text sleep diary (top) and the extracted temporal information (down).

In this paper, we consider the use-case of a sleep diary, which is a summary of sleep designed to gather information about daily sleep patterns (Carney et al., 2012). A typical sleep diary consists of a series of close-ended questions to record the time. By writing sleep diaries, people can keep track of sleep, monitor sleep habits, and document sleeping problems which can be shared with their sleep therapists. We focus on extracting temporal information from a free-text sleep diary, To achieve this, a system should extract temporal expressions from the unstructured user-generated text and normalise the extracted temporal expressions into a standard format, as illustrated in Figure 1.

Temporal information extraction from user-generated text is a challenging task. First of all, it requires processing not only text but also numbers (e.g., 11pm or 23:00). But recent pre-trained language models (Devlin et al., 2019; Yang et al., 2019) have difficulty in processing numbers (Saxton et al., 2018; Ravichander et al., 2019; Dua et al., 2019) because these language models are pre-trained with language modelling objectives. Even though there have been recent studies on training

language models to process numerical information (Andor et al., 2019; Geva et al., 2020), the remaining challenge is how to obtain a large amount of training data.

A second challenge is that there are various ways of describing the same normalised time. For example, the normalised time 23:00 can be expressed as 11, 11 pm, 23:00, eleven o'clock, etc. This issue is, even more, severe when dealing with user-generated text that is typically noisy: the user-generated text is prone to spelling errors and grammatical errors and contains a lot of abbreviations (Petz et al., 2013). To address this, a sufficient amount of training dataset containing pairs of various temporal expressions and normalised time values is required.

A third challenge is that there are different types of temporal expressions which of each is difficult to extract. For example, temporal expressions include not only standalone times (e.g., 23:00) but also relative times (e.g., 5 minutes after), counts (e.g., 3 times), duration (e.g., for an hour), and frequencies (e.g., once per hour). For relative time expressions, the challenge is how to annotate temporal expressions and model dependencies. For count time expressions, the challenge is to deal with ambiguous terms, such as '*several times*' and '*a few times*'.

The last challenge is how to collect large-scale data while developing a proof-of-concept model to validate the hypothesis. Especially for health-related data, the data collection requires rigorous process of considering privacy and ethical aspects, which might result in a slow process. Moreover, typical machine learning development process includes the multiple cycles of collecting a new dataset and updating a model to improve the performance of model. Therefore, the challenge is how to train a machine leanring model when only a low very low amount of training data is available.

Therefore, the main research question of this paper is how to extract temporal information from user-generated noisy text with the limited number of training data. To this end, we propose a synthetic data generation algorithm to augment the size of training data. We also propose a multi-task model and investigate whether the multi-task learning strategy is beneficial to the target task by exploiting additional training signals from the existing training data. The main contributions of this paper include the followings:

- A new custom dataset has been collected to demonstrate the success of the free-text sleep diary use-case (Section 3).
- The temporal information extraction and normalisation tasks are reformulated as a question and answering task (Section 4.1).
- A novel model that can extract temporal expressions from unstructured text and normalise them into the standard format is proposed (Section 4.2).
- Experimental results show that utilising synthetic data and multi-task learning can be beneficial to performance improvement (Section 5.5).
- We also provide further analysis on experimental results to reveal insights of the model behaviours (Section 6).

2 Related Work

There are two lines of approach in temporal information processing. One is rule-based and the other is machine learning-based. Generally, rule-based systems achieve high performances in a normalisation task (Chang and Manning, 2012). However, rule-based systems have difficulties in dealing with ambiguous phrases or relative expressions (Verhagen et al., 2010; Chang and Manning, 2012).

Another line of approach is machine learning-based approaches. Previous works have focused on detecting temporal links between entities and classify the temporal relations between them (Ning et al., 2017; Meng and Rumshisky, 2018) rather than predicting the exact time of events. Recently, Leeuwenberg and Moens (2020) propose a system that can directly extract start and end-points for events from the text. However, the remaining gap is that it is not entirely end-to-end: Leeuwenberg and Moens (2020) used the text with ground truth event spans and normalized temporal expressions as inputs. Moreover, even though machine learning models show promising results, the fundamental challenge is how to obtain data. Not only data acquisition can be difficult but also data labelling can be time-consuming and expensive.

3 Sleep Diary Analysis

This section describes the dataset collected for experiments. The following subsections explain the

details of use-case definition, data collection protocol, data labelling scheme, and an initial data analysis result.

3.1 Use-case definition

A sleep diary is a summary of sleep designed to gather information about daily sleep patterns. A typical sleep diary consists of a series of close-ended questions to record the time (i.e., the time people went to bed last night, woke up, etc), factors that may have influenced the way people slept, and how people felt when they woke up. In this study, we introduce free-text sleep diary use-case that allows people to describe their nights' of sleep in text. The goal of this study is to extract structured information from unstructured sleep diaries, as described in Figure 1.

3.2 Data collection protocol

We conducted an online survey via Amazon's Mechanical Turk (MTurk) to collect experimental data. At the beginning of the survey, the participants were given a questionnaire with a brief background of the study purpose. Then the participants were asked to provide information about their sleep of the previous night via an open-ended question (i.e., *"Please describe, in a few lines, your sleep last night."*). The details of data subject selection criteria and examples of responses are given in Appendix A. In total, 600 participant inputs are collected and used for the experiments.

3.3 Data labelling scheme

To annotate the collected data, several sleep-related event entities are defined based on sleep study (Carney et al., 2012) as summarised in Table 1. Each event entity text was annotated with its span (i.e., start and end positions in the text), entity label, expression type (i.e., standalone, relative[1], count, frequency[2]), and normalised time value. Expression types are used to assign a specific type value: None, +/-, *, *t* for standalone, relative, frequency, count, respectively. A normalised time value includes a type value and 4 digits indicating HH:MM, except for a count type: for an entity with a count type, a normalised time value is a cardinal number padded with leading zeros (e.g., 1 becomes 0001). Also,

[1]Relative type includes both relative time (e.g., after 5 minutes) and duration (e.g., for 5 minutes).

[2]Frequency type includes expressions of events occurring periodically (e.g., every 1 hour)

Event entity	Explanation
bed time	The time when the participant went to bed/bedroom.
lights off	The time when the participant switched off the lights and began trying to fall asleep.
sleep time	The time when the participant fell asleep
sleep latency	The amount of time it took for the participant to fall asleep after deciding to go to sleep.
sleep disturbance	The times when the participant's sleep was disturbed.
wake up	The time when the participant woke up from their sleep.
out of bed	The time when the participant finally got out of bed to start their day.
sleep duration	The total duration of time the user was asleep.

Table 1: The list of sleep-related event entities used in this study.

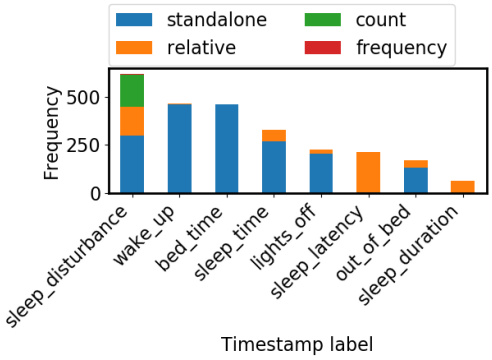

Figure 2: Distribution of the annotated data over the event entities.

we set rules for ambiguous expressions. For example, 'a lot of times', 'several times', and 'many times' are annotated as five times (t0005) and 'a few times', 'a couple times' are annotated as two times (t0002). The example of an annotated data point is illustrated in Appendix B.

Figure 2 shows the distribution of the collected data set over the sleep-related event entities. It is observed that `sleep_disturbance` entity appears more than other entities. This is because different types of sleep disturbance are often mentioned together (i.e., "*I woke up 1 time*

to use restroom at midnight"). Meanwhile, some entities (e.g., `lights_off`, `sleep_latency`, `out_of_bed`, `sleep_duration`) are often missing in sleep diary entries. In general, `bed_time` and `wake_up` entities appear once per each sleep diary entry.

4 Multi-task temporal information extraction model

4.1 Task formulation

We formulate a temporal information extraction and normalisation task similar to a question and answering task. Therefore, each data point is transformed into <entity, text, answer> where the entity is a sleep event entity label, the text is sleep diary text, and the answer is a normalised time with a type value and 4 digits. For example, a system is expected to predict a list of answers [None, 2, 2, 3, 0] given input <bed time, I went to bed at half past 10...>.

Formally, an entity $q_j \in Q$, where Q is the set of the sleep-related event entities decribed in Table 1, is tokenised[3] with m_j tokens $q_j = [q_j^1, ..., q_j^{m_j}]$ and sleep diary text is tokenised with n_i tokens $p_i = [p_i^1, ..., p_i^{n_i}]$. Then the task is to predict an answer $a_{ij} = [a_{ij}^{type}, a_{ij}^{t1}, a_{ij}^{t2}, a_{ij}^{t3}, a_{ij}^{t4}]$ given a sequence of tokens [`[CLS]` q_j `[SEP]` p_i, `[SEP]`], where `[CLS]` and `[SEP]` are special tokens for classification and separation, respectively. a_{ij}^{type} is the ground truth label for the type value of the normalised time and a_{ij}^{t1}, a_{ij}^{t2}, a_{ij}^{t3}, and a_{ij}^{t4} is the the ground truth labels for the each digit of the normalised time.

4.2 Model architecture

We propose a multi-task model that utilises a pre-trained language model with specific heads, motivated by recent works (Andor et al., 2019; Geva et al., 2020). The overview of the proposed model is illustrated in Figure 3.

Firstly, the model computes $e_{ij} = [e_{ij}^{cls}, .., e_{ij}^{l_{ij}}]$ which are contextualised representations for the $l_{ij} = m_j + n_i + 3$ input tokens ([CLS] q_j [SEP] p_i, [SEP]) by using a pre-trained language model BERT (Devlin et al., 2019). The contextualised embedding vector $\boldsymbol{e}_{ij}^{cls} \in \mathbb{R}^{d \times 1}$, corresponding to the classification token `[CLS]`, is fed to the type classification head (H_{type}) that uses a fully-connected

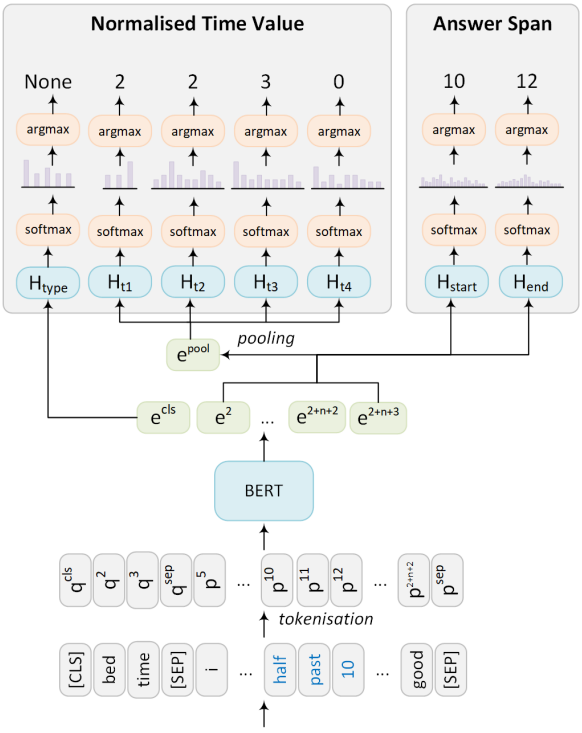

Figure 3: Illustration of the proposed model.

layer followed by a softmax to compute distributions over the type values {None, +, -, *, t}. Then the remaining sequence of contextualised embedding vectors $[e_{ij}^2, .., e_{ij}^{l_{ij}}]$ is used to create pooled embedding $e_{ij}^{pool} \in \mathbb{R}^{d \times 1}$ by using average pooling. Then the pooled embedding e_{ij}^{pool} is passed to normalised time value heads ($H_{t1}, H_{t2}, H_{t3}, H_{t4}$). H_{t1} head computes a distribution over the number {0, 1, 2}[4] by using a fully-connected layer followed by a softmax layer. Similarly, H_{t2}, H_{t3}, and H_{t4} heads compute distributions over the numbers {0, ..., 9}.

The contextualised embedding vectors $[e_{ij}^2, .., e_{ij}^{l_{ij}}]$ are fed to additional answer span heads, H_{start} and H_{end}, to compute a score for each token, corresponding to whether that token is the start or the end of the answer span, respectively. The start and end probability for each token is computed as follow:

$$p_{ij}^{start} = \text{softmax}(H_{start}(e_{ij}^2), ..., H_{start}(e_{ij}^{l_{ij}}))$$
$$p_{ij}^{end} = \text{softmax}(H_{end}(e_{ij}^2), ..., H_{end}(e_{ij}^{l_{ij}}))$$

[3] Underbars are replaced by whitespace characters during tokenisation.

[4] Since the answer is formulated as a standard time format (e.g., HH:MM), the first digit is limited to {0, 1, 2}.

Normalised time prediction loss For normalised time prediction, cross-entropy between the target answer and the model estimation is used to compute $\mathcal{L}_{t_1}, \mathcal{L}_{t_2}, \mathcal{L}_{t_3}, \mathcal{L}_{t_4}$, and $\mathcal{L}_{type}$ which is a loss function for each head, respectively. Then the time loss function ($\mathcal{L}_{\text{time}}$) is defined as the linear combination of the each loss function, i.e.

$$\mathcal{L}_{\text{time}} = \alpha\mathcal{L}_{\text{type}} + \beta(\mathcal{L}_{t_1} + \mathcal{L}_{t_2} + \mathcal{L}_{t_3} + \mathcal{L}_{t_4})$$

Answer span detection loss For answer span detection, we follow the previous work on a question and answering task by using a pre-trained language model (Devlin et al., 2019) to compute cross-entropy losses for the start $\mathcal{L}_{\text{start}}$ and end $\mathcal{L}_{\text{end}}$. Then the span loss function ($\mathcal{L}_{\text{span}}$) is defined as the sum of the start loss and the end loss:

$$\mathcal{L}_{\text{span}} = \mathcal{L}_{\text{start}} + \mathcal{L}_{\text{end}}$$

Multi-task loss The final multi-task loss function ($\mathcal{L}_{\text{multi}}$) is defined as the linear combination of the normalised time prediction and answer span detection loss functions:

$$\mathcal{L}_{\text{multi}} = \mathcal{L}_{\text{time}} + \gamma\mathcal{L}_{\text{span}}$$

4.3 Synthetic data generation

To augment the size of annotated data containing temporal information, we propose a simple yet effective rule-based synthetic data generation algorithm. Figure 4 illustrates the proposed algorithm. The first step is to create a template by masking out labelled event entities from the annotated data and replacing them with placeholders. The second step is to generate random entity types and corresponding normalised time values. Then the randomly generated normalised time values are translated into texts by using regular expressions. The last step is to replace placeholders with the translated texts. Details of regular expressions and examples of generated texts are given in the Appendix C.

5 Experiments

5.1 Dataset

The collected dataset from the Section 3 was used for the experiments. We randomly split the collected data ($n = 600$) into train, validation, and test sets with the ratio of 0.8, 0.1, and 0.1. After splitting data sets, we dropped the data points that do not contain any event entities. During pre-processing, we lowercased and tokenised data sets by using a WordPiece algorithm (Schuster and Nakajima, 2012). Numbers and punctuation symbols were not removed during pre-processing because they play important role in temporal expressions. Table 2 shows the statistics of each data set after pre-processing.

The proposed synthetic data generation algorithm is applied to the train set to augment the size by a factor of k. The validation set is used to check the training progress and perform early stopping. The test set is used to evaluate the performance of the trained model.

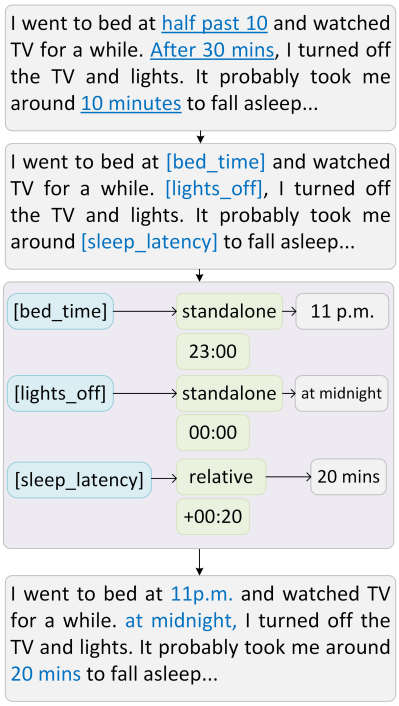

Figure 4: Illustration of the proposed synthetic data generation algorithm to augment the size of training data.

Statistics	Train	Valid	Test
# sleep diaries	467	56	57
Total # tokens	197,695	24,007	23,528
Unique # tokens	1,687	604	585
Avg. # tokens/diary	95.9	101.7	95.3
Total # entities	2061	236	247
Avg. # entities/diary	4.4	4.2	4.3

Table 2: Data set statistics across the different splits

5.2 Settings

We use a pre-trained BERT to implement the proposed model. All heads are implemented as fully-connected layers with dropout followed by softmax functions. Also, to investigate the effect of multi-task learning, we implement a baseline model (BASE) that uses only normalised time prediction loss ($\mathcal{L}_{\text{time}}$) and a multi-task model (MULTI) that uses multi-task loss ($\mathcal{L}_{\text{multi}}$), as defined as:

$$\mathcal{L}_{\text{time}} = \alpha\mathcal{L}_{\text{type}} + \beta(\mathcal{L}_{t_1} + \mathcal{L}_{t_2} + \mathcal{L}_{t_3} + \mathcal{L}_{t_4})$$
$$\mathcal{L}_{\text{span}} = \mathcal{L}_{\text{start}} + \mathcal{L}_{\text{end}}$$
$$\mathcal{L}_{\text{multi}} = \mathcal{L}_{\text{time}} + \gamma\mathcal{L}_{\text{span}}$$

α, β, and γ are set to 0.25, 1, and 0.25, respectively. See Appendix D for more system and training details.

5.3 Configuration

To configure the input and output of a model, each data point is expanded into multiple <entity, text, answer> triples. If a single data point contains the same entity more than once, the ordinal number is added to an entity label from the second occurrence. For example, `second_sleep_disturbance` will be used for the second `sleep_disturbance` event in a data point. For `sleep_disturbance` entity with a count type, the type is also added to an entity label, i.e., `count_sleep_disturbance` will be used. For output, a special character of a normalised time value (i.e., None, +, -, *, t) is used as a ground truth type value and each digit of a 4 digit normalised time is used as a ground truth normalised time value. For the multi-task model, the start and end position of entity text are used as ground truth value of start and end position, respectively.

5.4 Evaluation metric

Normalised time prediction We use Exact Match (EM) as an evaluation metric. EM considers only when a model predicts both a correct type value and correct 4 digits of normalised time values as a correct prediction. Since the experimental data set has an imbalance over event entities, both micro-and macro-averaged scores are used: the micro-EM score is computed by taking the average over inputs and the macro-EM score is computed by taking the average at the entity level.

Answer span detection Following Rajpurkar et al. (2016), we use Exact Match (EM) and F1 score for answer span detection: EM measures the percentage of predictions that match the ground truth answers exactly. F1 score measures the average overlap between the prediction and ground truth answer. We treat the prediction and ground truth as bags of tokens, and compute the F1 score per entity and average over all of the entities. Both metrics consider articles, numbers, and punctuation symbols.

5.5 Results and Analysis

Normalised time prediction Table 3 summarises the experimental results. As expected, models trained on the synthetic training data generally achieve higher performances. Results show that the benefit of using synthetic data for training shows a peak at $k = 3$ for both models and the improvements are statistically significant. However, the benefit of using synthetic data decreases afterwards. It can even harm the performance of the baseline model when $k = 10$.

To further investigate the effects of using synthetic data for training, we calculate the performance per event entity label as shown in Table 4 and the performance per expression type as shown in Table 5. The models trained on synthetic data with the factor of $k = 3$ are used for comparison. Table 4 shows that using synthetic data generally improves the performance of almost all event entities. From both models, the biggest improvements are observed at `sleep_disturbance` entity, which is the most frequent entity label in the training set. However, as shown in Table 5, the biggest improvements in terms of expression type are observed at the count type, which is one of the least frequent expression types in the training set. It is worth mentioning that the count type is only included in `sleep_disturbance` entity label, as illustrated in Figure 2. These results imply that using synthetic data can be the most beneficial to both models in terms of predicting normalised time values with the count type.

Answer span detection The answer span detection results of the multi-task model are also summarised in Table 3. Similar to the normalised time prediction results, the performances tend to increase till $k = 3$ and decrease afterwards. It is observed that the utilising synthetic data with the augmentation factor $k = 3$ can provide the signif-

	BASE		MULTI			
	Normalised time prediction		Normalised time prediction		Answer span detection	
k	micro-EM	macro-EM	micro-EM	macro-EM	EM	F1
-	82.0	83.9	66.9	62.9	64.9	82.9
$\times 2$	83.3	**87.2***	86.9**	88.0**	**66.9**	85.1
$\times 3$	**85.3***	85.1	**88.6****	**91.9****	66.1	**86.7***
$\times 5$	81.6	81.6	86.1**	86.8**	64.5	84.1
$\times 8$	82.0	83.2	78.4**	80.8**	60.8	79.2*
$\times 10$	72.2**	74.7**	68.6	71.2*	53.5**	75.3**

Table 3: The performances of the baseline model (BASE) and the multi-task model (MULTI). k refers the augmentation factor. * and ** indicate that this result is significantly different (approximate randomisation test (Dror et al., 2018)) from the result without the synthesised data (the first row in that column) with p-value < 0.05 and < 0.01, respectively. Best performances are boldfaced.

	BASE		MULTI	
	-	+SD	-	+SD
sleep disturb.	66.1	80.6	50.0	83.9
bed time	95.1	97.6	95.1	95.1
wake up	86.5	86.5	65.4	88.5
sleep time	75.0	69.4	72.2	75.0
lights off	100.	93.3	93.3	100.
sleep latency	95.2	100.	66.7	100.
out of bed	78.6	78.6	35.7	92.9
sleep dur.	75.0	75.0	25.0	100.
micro-EM	82.0**	85.3*	66.9**	**88.6**
macro-EM	83.9**	85.1*	62.9**	**91.9**

Table 4: Normalised time prediction results per entity label. +SD indicates that synthetic data are used. * and ** indicate that this result is significantly different from the best result in that row (bolded) with p-value < 0.05 and < 0.01, respectively.

	BASE		MULTI	
	-	+SD	-	+SD
Standalone	86.8	86.3	72.0	**90.1**
Relative	78.7	78.7	55.3	**83.0**
Count	37.5	**93.8**	43.8	87.5

Table 5: Normalised time prediction results per expression type. +SD indicates that synthetic data are used. Best performances are boldfaced.

icant improvement in terms of F1 measure. But the effect to the EM measure is not statistically significant (p>.05). It is also observed that using synthetic data with augmentation factor $k = 10$ can significantly harm the performances.

6 Discussion

Effects of using synthetic data The first row of Table 3 shows that the multi-task learning model achieves lower normalised time prediction performances than the baseline model when no synthetic data is used for training. However, when the multi-task model utilises an appropriate amount of synthetic data ($k = 3$), as it is shown in the last two rows in Table 4, the multi-task model significantly outperforms ($p < .01$) the baseline model without synthetic data. These results imply two things: 1) multi-task learning can be beneficial to improve the target performances of normalised time prediction. However, training the multi-task model may require a larger training set; and 2) the proposed synthetic data generation algorithm can mitigate this issue to a certain degree. Also, as shown in the last two rows in Table 4, when the same amount of synthetic data ($k = 3$) are used for both models, the multi-task model significantly outperforms ($p < .05$) the baseline model in terms of normalised time prediction. This result may not be so surprising since the multi-task model receives additional training signals during training.

Effects of using multi-task learning To get a further understanding of the effect of multi-task learning, we conduct a qualitative analysis. In Table 6, we highlight some examples of the predictions of the proposed models. In the first example, it is observed that both models can process the

Sleep Diary

I went to bed about 9 pm. we sleep with the lights on for my toddler who co-sleeps. I was asleep about 10. I woke up a lot of times in the night to blow my nose or to try and get comfortable. I got out of bed at 2:30 am. Sleep was terrible. I feel exhausted today.

	BASE	**MULTI**		**Ground Truth**	
	time	time	text	time	text
bed time	21:00	21:00	9 pm	21:00	9 pm
sleep time	22:00	22:00	about 10	22:00	about 10
cnt. sleep disturb.	t0005	t0002	a lot of times	t0005	a lot of times
out of bed	02:30	02:30	2:30 am	02:30	2:30 am

Sleep Diary

I turned the lights off at 9:30 layed down in bed at 10 pm. I fell alseep around 11pm I woke up at 1am to turn on my other side. I fell back to sleep until 4 am to use the rest room went back to sleep until 5:30 am.

	BASE	**MULTI**		**Ground Truth**	
	time	time	text	time	text
lights off	21:30	21:30	9:30	21:30	9:30
bed time	22:00	22:00	10 pm	22:00	10 pm
sleep time	23:00	23:00	11pm	23:00	11pm
sleep disturbance	01:00	01:00	1am	01:00	1am
second sleep disturb.	01:00	04:00	4 am	04:00	4 am
wake up	05:00	05:30	5:30 am	05:30	5:30 am

Sleep Diary

I went to bed around 10p. m. I read for about 30 minutes. Put my kindle away and was asleep by 10:30. I woke up at 1a. m. and it took me around 20 minutes to fall back asleep. I woke up at 2:45 and used the bathroom. I went back to sleep until 4 which is when I normally get up.

	BASE	**MULTI**		**Ground Truth**	
	time	time	text	time	text
bed time	22:00	22:00	10p. m.	22:00	10p. m.
sleep time	22:30	22:30	10:30	22:30	10:30
sleep disturbance	01:00	01:00	1a. m	01:00	1a. m
second sleep time	01:00	00:20	1a. m. and it took me around 20 minutes	+00:20	20 minutes
second sleep disturb.	01:00	01:45	1a. m.	02:45	2:45
wake up	02:45	04:45	-	04:00	until 4

Table 6: Qualitative examples showing the outputs of the proposed models. Underline indicates temporal expressions and red colour indicates wrong predictions. Due to limited space, we use the following abbreviations: count sleep disturbance (cnt. sleep disturb.) and second sleep disturbance (second sleep disturb.).

combination of number and text (*9 pm*) and an ambiguous expression (*10*), correctly predicting corresponding the normalised time values (21:00 and 22:00). It is also observed that the baseline model correctly predicts a normalised time value of a cardinal number (t0005) from the text-only temporal expression (*a lot of times*). The multi-task model fails at predicting the correct normalised time value but extracts the correct answer span (*a lot of times*). Based on this observation, it seems that answer span detection results can be useful to decide how to synthesise training data, such as generating pairs of <'a lot of times', t0005>.

In the second example, it is observed that the baseline model correctly predicts only the first occurrence of `sleep_disturbance` entity, predicting the identical timestamps for the second occurrences. Meanwhile, the multi-task model correctly predicts both occurrences with correct answer spans. We found that the multi-task model generally performs better on extracting normalised time values that occur multiple times in the text. However, in general, both models have difficulties in dealing with entities that occur several times in a single sleep diary. It is also observed that the normalised time value heads ($H_{type}, H_{t1}, H_{t2}, H_{t3}, H_{t4}$) and the answer span heads (H_{start}, H_{end}) of the multi-task model are not fully aligned: as shown in the third example, the multi-task model estimates the second sleep disturbance as 01:45 while extracting the answer span as '1a.m'. This error is challenging because the current model is a black box model so that we do not know where the error occurs and how the error propagates. To address this issue, one interesting area for future work may be in investigating shared information between the normalised time value heads and the answer span heads.

7 Limitations and Future Work

Even though we show the effectiveness of the proposed method by validating on the collected dataset, some points can be further studied. First of all, the proposed models estimate a normalised time value conditioned on an input which is a pair of sleep diary text and sleep-related event entity label. However, since most sleep diary texts do not contain all the event entities, an additional module is required to detect which entities are mentioned in the given text and how many times each entity is mentioned in the given text. Similar to the previous work by Liu et al. (2020), the answer span detection head of the proposed multi-task model can be used as a detection module.

Secondly, even though the proposed models can handle relative expressions by using specific type values (i.e., +, -), a linking algorithm is currently missing. A potential solution is to add a head that can predict a starting point, similarly to the previous work by Leeuwenberg and Moens (2018).

The third limitation is that the proposed models process only temporal information. To completely analyse sleep diary, extracting contextual and qualitative information is required (Ibáñez et al., 2018). In the future study, we will train a model to extract both temporal and other information from text data. To achieve this, we will collect more data that are longer and contain rich information about the context of the night and sleep.

8 Conclusions

In this paper, we propose a model that can extract temporal information from health-related narrative text. We conducted experiments to investigate how to utilise synthetic data and multi-task learning to improve the performance of normalised time prediction. Experimental results show that utilising synthetic data for training can contribute to performance improvement the most when the augmentation factor is set to 3. The results also show that when multi-task learning is used with synthetic data appropriately, the performance can be significantly improved. In the future study, we will extend the current work to extract not only temporal information but also contextual and qualitative information from text.

9 Ethical Considerations

Table 7 summarises ethical and privacy considerations of the data collection in this study.

Acknowledgements

We thank anonymous reviewers for providing valuable feedback on this work. This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 766139. This article reflects only the author's view and the REA is not responsible for any use that may be made of the information it contains.

Question	Answer
Are children under the age of 18 involved as test subjects in the study?	No
Are test subjects over the age of 65 involved in the study?	Yes
Do the test subjects belong to vulnerable groups?	No
Does the study induce harm or discomfort to the test subjects?	No
Is there any doubt on the test subjects' freedom in deciding on their participation?	No
Collection of any personal data	No
Collection of data by means of audio recording	No
Collection of data by means of video recording or photographs	No
Collection of data by means of observation of test subjects and logging in written format	No
Collection of data by means of filling-in questionnaires/surveys/interviews	Yes

Table 7: Ethical and privacy considerations for the data collection. Vulnerable groups include military veterans, terminally ill, educationally or socioeconomically disadvantaged, employees, students who could be unduly influenced, individuals with lack of or loss of autonomy due to immaturity or through mental disability that might suggest their consent is not of free will, etc.

References

Daniel Andor, Luheng He, Kenton Lee, and Emily Pitler. 2019. Giving bert a calculator: Finding operations and arguments with reading comprehension. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5947–5952.

Colleen E Carney, Daniel J Buysse, Sonia Ancoli-Israel, Jack D Edinger, Andrew D Krystal, Kenneth L Lichstein, and Charles M Morin. 2012. The consensus sleep diary: standardizing prospective sleep self-monitoring. *Sleep*, 35(2):287–302.

Angel Chang and Christopher D Manning. 2012. Sutime: A library for recognizing and normalizing time expressions. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 3735–3740.

Edward Choi, Mohammad Taha Bahadori, Joshua A Kulas, Andy Schuetz, Walter F Stewart, and Jimeng Sun. 2016. Retain: an interpretable predictive model for healthcare using reverse time attention mechanism. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 3512–3520.

Dina Demner-Fushman, Noémie Elhadad, and Carol Friedman. 2021. Natural language processing for health-related texts. In *Biomedical Informatics*, pages 241–272. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378.

Mor Geva, Ankit Gupta, and Jonathan Berant. 2020. Injecting numerical reasoning skills into language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 946–958.

Vanessa Ibáñez, Josep Silva, and Omar Cauli. 2018. A survey on sleep assessment methods. *PeerJ*, 6:e4849.

Akib Mohi Ud Din Khanday, Syed Tanzeel Rabani, Qamar Rayees Khan, Nusrat Rouf, and Masarat Mohi Ud Din. 2020. Machine learning based approaches for detecting covid-19 using clinical text data. *International Journal of Information Technology*, 12(3):731–739.

Artuur Leeuwenberg and Marie Francine Moens. 2018. Temporal information extraction by predicting relative time-lines. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1237–1246.

Artuur Leeuwenberg and Marie-Francine Moens. 2020. Towards extracting absolute event timelines from english clinical reports. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2710–2719.

Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. 2020. Event extraction as machine reading comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1641–1651.

Yuanliang Meng and Anna Rumshisky. 2018. Context-aware neural model for temporal information extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 527–536, Melbourne, Australia. Association for Computational Linguistics.

Qiang Ning, Zhili Feng, and Dan Roth. 2017. A structured learning approach to temporal relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1027–1037, Copenhagen, Denmark. Association for Computational Linguistics.

Gerald Petz, Michał Karpowicz, Harald Fürschuß, Andreas Auinger, Václav Stříteskỳ, and Andreas Holzinger. 2013. Opinion mining on the web 2.0–characteristics of user generated content and their impacts. In *International Workshop on Human-Computer Interaction and Knowledge Discovery in Complex, Unstructured, Big Data*, pages 35–46. Springer.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Abhilasha Ravichander, Aakanksha Naik, Carolyn Rose, and Eduard Hovy. 2019. Equate: A benchmark evaluation framework for quantitative reasoning in natural language inference. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 349–361.

David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. 2018. Analysing mathematical reasoning abilities of neural models. In *International Conference on Learning Representations*.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152. IEEE.

Marc Verhagen, Roser Sauri, Tommaso Caselli, and James Pustejovsky. 2010. Semeval-2010 task 13: Tempeval-2. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 57–62.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, pages arXiv–1910.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763.

A Details of data collection protocol

Participants were recruited through Amazon's MTurk-service. We selected data subjects who meet the following criteria:

- People who are 18 years or older
- People who are USA residents

When participants selected the study, they received a link to the web page hosting the survey. During the survey, participants were asked to answer an open-ended question (i.e., *"Please describe, in a few lines, your sleep last night."*) with the guidance of the following sentences: "*While describing your answer, please include the following information: sleep-related events (e.g., the time you went to bed, the time you switched off to go to sleep, the time it took you to fall asleep, the number of times you woke up and the time at those moments, the time you woke up, the time you got out of bed) and the overall sleep evaluation or how you refreshed after you woke up.*"

At any moment, a participant was allowed to end her/his participation in the study. In this case, the test participant was not replaced. Furthermore, every participant was received informed consent, on the landing page that participants enter when following the link from MTurk. Only answers from the participants who gave their own consent to the

ID	Answers
#1	I went to bed at 11 pm. I switched off the lights and lay down around 15 minutes before falling asleep. It was a deep sleep and i wake only 1 time to witch off ceiling fan. I had couple of dreams that I remember partially not scary. I wake up at 6 am. lay on bed for 15 minutes more and got up.
#2	I turned off the lights around 9:45 PM. I closed my eyes and went to sleep around 10 PM. I did not wake up at all during the night. I slept straight through. I woke up at 6 AM. I lied in bed for a few minutes before actually getting up around 6:05 AM. I felt fairly well rested.
#3	I went to bed at 10:00 and immediately turned off the light. I fell asleep in just a few minutes. I slept without waking until 5:00. I immediately got out of bed when I woke up and felt great.

Table 8: Examples of responses to the open-ended question regarding the previous night's sleep.

study were used for experiments in this study. Table 8 shows examples of the collected data used for experiments.

B Example of annotated data

Figure 5 shows the exmample of an annotated data point.

C Examples of synthetic data

A synthetic data set was generated by the following steps: 1) Template generation; 2) Random timestamps generation; and 3) Rule-based timestamps-to-texts translation. Table 9 summarises a set of rules used for timestamp-to-text translation and the examples of generated texts.

D Experimental settings

The detailed specification of hardware and software is summarised in Table 10. For model deployment, PyTorch version of BERT with the pre-trained weights `bert-base-uncased` (Wolf et al., 2019) was used. Table 11 summarises hyperparameter values used for the experiments. All hyperparameters are obtained based on non-exhaustive experiments. During the inference phase, we followed the settings from the original

```
{'text':'I went to bed at 10'o clock
    and I switched off the light at 11
    'o clock. After that I fall asleep
     in 30 minutes as my guess. I woke
     up 3 times and I felt restless. I
     was awake up to 3 hours. I woke
    up at 7'o clock in the morning and
     got out of the bed at 7:45 a.m.
    My overall sleep is up to 5 hours
    and it was not a sound sleep. If I
     awake during night then it causes
     me heavy headache and drowsiness.
     But today I din't get any of the
    above symptoms even though I woke
    up in the night. I felt fresh in
    the morning.',
 'labels': [
 { 'text': '10'o clock',
  'span': (17, 27),
  'entity': 'bed_time',
  'type': 'standalone',
  'norm_time': 2200},
 { 'text': '11'o clock',
  'span': (60, 69),
  'entity': 'lights_off',
  'type': 'standalone',
  'norm_time': 2300},
 { 'text': 'in 30 minutes',
  'span': (97, 110),
  'entity': 'sleep_latency',
  'type': 'relative',
  'norm_time': +0030},
 { 'text': '3 times',
  'span': (134, 141),
  'entity': 'sleep_disturbance',
  'type': 'count',
  'norm_time': t0003},
  {'text': '3 hours',
  'span': (181, 188),
  'entity': 'sleep_disturbance',
  'type': 'relative',
  'norm_time': +0300},
 {'text': '7'o clock',
  'span': (203, 212),
  'entity': 'wake_up',
  'type': 'standalone',
  'norm_time': 0700},
 {'text': '7:45 a.m',
  'span': (254, 261),
  'entity': 'out_of_bed',
  'type': 'standalone',
  'norm_time': 0745},
 {'text': '5 hours',
  'span': (290, 297),
  'entity': 'sleep_duration',
  'type': 'relative',
  'norm_time': +0500}]}
```

Figure 5: Example of annotated data point containing free-text sleep diary and labels of event entities.

Regular Expression	Entity type	Format	Example	
			Timestamp	Text
?(around|at|until|by|) t_1t_2[:,.]t_3t_4 ?(((a.?m.?)|(A.?M.?))| (hrs|hours|hour|hr))	standalone	$t_1t_2t_3t_4$	1130	at 11:30 am
?(around|at|until|by|) (t_1t_2|t_1t_2-12)[:,.]t_3t_4 ?(((p.?m.?)|(P.?M.?))| (hrs|hours|hour|hr))	standalone	$t_1t_2t_3t_4$	2230	around 10:30 PM
?(around|at|until|by|) t_1t_2 ?((o['\]clock)|((a.?m.?)| (A.?M.?))|(hrs|hours|hour|hr)) ?(o('| |)clock|[:,.]00)	standalone	$t_1t_2$00	0600	until 6 o'clock
?(around|at|until|by|) (t_1t_2|t_1t_2-12) ?((o['\]clock)| ((p.?m.?)|(P.?M.?))|(hrs|hours|hour|hr)) ?(o('| |)clock|[:,.]00)	standalone	$t_1t_2$00	2200	22 o'clock
?(after|within|) t_3t_4 ?(min|mins|minute|minutes) ?(later)	relative	+00t_3t_4	+0010	10 minutes later
t_3t_4 ?(min|mins|minute|minutes) ?(before|prior)	relative	-00t_3t_4	+0010	5 minutes before
every t_3t_4 ?(min|mins|minute|minutes)	frequency	*00t_3t_4	*0010	every 10 mins
every (t_1t_2|t_1t_2+1) ?(hour|hours)	frequency	*$t_1t_2$00	*0100	every 1 hour
every (t_1t_2|t_1t_2+1) and ?(hour|hours) t_3t_4 ?(min|mins|minute|minutes)	frequency	*$t_1t_2t_3t_4$	*0115	every 1 hour and 15 minutes
((one|1) (times| time)|once)	count	t0001	t0001	once
((two|a couple of|a few|few|2) (times| time)|twice)	count	t0002	t0002	a couple of times
(several|many|a lot of|multiple|5) (times| time)	count	t0005	t0005	several times
(t_4|t_4 or t_4+1|t_4 to t_4+1|t_4-t_4+1) ?(time|times)	count	t000t_4	t0003	3 times

Table 9: Regular expression patterns used to translate timestamps into texts based on the given entity type and format.

Item	Specification
CPU	Intel Xeon W-2123 CPU(3.60 GHz)
GPU	NVIDIA GeForce GTX 1080 ti, 11 GB memory
Driver	NVIDIA graphic driver ver. 416.34
CUDA	Version 10.0
OS	Windows 10, 64-bit
Python	Version 3.6.6
Pytorch	Version 1.5.1

Table 10: Detailed implementation specification.

Hyperparameter	Assignment
α	0.25
β	1.
γ	0.25
max training epoch	9
batch size	32
learning rate	$4e-5$
dropout rate	0.1
optimaser	AdamW

Table 11: Hyperparameters for fine-tuning.

paper (Devlin et al., 2019) to compute the scores of a candidate span.