

Thesis for the Degree
of Doctor

Contextualized representation learning
for named entity recognition and event
scheduling

by

Donghyeon Kim

Department of Computer and Radio
Communications Engineering

Graduate School

Korea University

February 2020

姜在雨 教授指導

博 士 學 位 論 文

Contextualized representation learning
for named entity recognition and event
scheduling

이 論文을 컴퓨터·電波通信工學
博士學位 論文으로 提出함

2019年 12月 6日





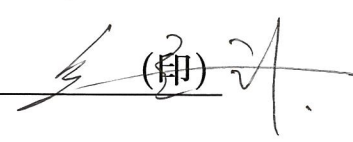
高麗大學校大學院

컴퓨터·電波通信工學科

金東玄 (印)

金東玄의 컴퓨터·電波通信工學 博士學位論文
審査를 完了함

2019年 12月 6日

委員長	강재우	(印) 
委 員	임희석	(印) 
委 員	주재걸	(印) 
委 員	김현우	(印) 
委 員	석준희	(印) 

Abstract

In a classification task, considering a context around a target helps to improve inference by providing additional clues to a model. For example, it is possible to improve the word-level classification accuracy by obtaining a contextualized word representation considering words before and after each word in a sentence. Bidirectional models such as Bidirectional Long Short-Term Memory (BiLSTM) networks, Embeddings from Language Models (ELMo), and Bidirectional Encoder Representations from Transformers (BERT) have shown their effectiveness in obtaining contextualized representations for words, and recently, transfer learning using a model pre-trained on a large dataset has become the mainstream.

In this paper, we use and propose models which learn contextualized representations for named entity recognition (NER) and event scheduling. For NER, we propose a neural biomedical NER and multi-type normalization tool called Biomedical Entity Recognition and Normalization (BERN) which uses BioBERT pre-trained on 19M PubMed abstracts and 2M PubMed Central full-texts. Using deep contextualized representations, BioBERT NER models outperform existing NER models in F1 and even discover new entities that are not in entity dictionaries, using their language model and WordPiece embeddings. Also, we developed probability-based decision rules to identify the entity types of overlapping entities in multi-type NER results. The BERN provides a Web service for tagging entities in PubMed articles or raw text. For event scheduling, we propose Neural Event Scheduling Assistant (NESA) which learns user preferences and understands calendar contexts, directly from raw online calendars for fully automated and highly effective event scheduling. We leverage over 593K calendar events for NESA to learn scheduling personal events, and we further utilize NESA for multi-attendee event scheduling. NESA successfully incorporates deep neural networks for event scheduling based on natural languages, users, durations, and pre-registered events. The experimental results show that NESA significantly outperforms previous baseline models on both personal and multi-attendee event scheduling tasks.

Contents

Abstract

Contents i

List of Figures iii

List of Tables iv

1 Introduction 1

1.1 Named entity recognition 2

1.2 Event scheduling 5

2 Contextualized word representations for named entity recognition 9

2.1 Background 9

2.1.1 Named entity recognition for biomedical text mining 9

2.1.2 Resolving overlapping entities 10

2.1.3 Named entity normalization models for biomedical text mining . . 10

2.2 Methods 11

2.2.1 BioBERT for named entity recognition 12

2.2.2 Decision rules for overlapping entities 14

2.2.3 The multi-type normalization model 17

2.3 Implementation 19

2.3.1 Demonstrations 21

2.3.2	Application programming interfaces	22
2.4	Discussion	23
2.4.1	Use cases	23
2.4.2	Advantages and limitations of having a separate NER model for each entity type	25
2.4.3	Additional dependencies of BERN	25
2.4.4	Overcoming the network distance between a server and a client . .	26
3	Contextualized event representation learning for scheduling	27
3.1	Background	27
3.1.1	Preference learning for event scheduling	27
3.1.2	Multi-attendee event scheduling	28
3.1.3	Representation learning using deep neural networks	29
3.2	Problem formulation	29
3.2.1	Attributes of calendar data	29
3.2.2	Personal event scheduling	30
3.2.3	Multi-attendee event scheduling	31
3.3	Methodology	31
3.3.1	Title layer	32
3.3.2	Intention layer	34
3.3.3	Context layer	35
3.3.4	Output layer	36
3.4	Experiment	37
3.4.1	Dataset	37
3.4.2	Experimental settings	39
3.4.3	Quantitative analysis	42
3.4.4	Qualitative analysis	46
4	Conclusion	52
	Bibliography	54

List of Figures

1.1	Overview of the RESTful Web service of BERN	3
1.2	Example of calendar event scheduling. Mary requests NESAs to schedule a meeting with John. NESAs considers each user’s preference and calendar context, and the purpose of the event.	6
2.1	The percentages of partial overlapping entities of each entity pair	14
2.2	Decision rules for NER results	16
2.3	BERN demonstration of PMID:30429607 (best viewed in color mode) . .	21
3.1	NESA overview. Given the title, duration, user attributes, and pre-registered events, NESAs suggests suitable time slots for events.	32
3.2	Performance changes with different numbers of pre-registered events in NESAs.	44
3.3	Output probabilities of NESAs given different titles.	45
3.4	Output probabilities of NESAs in multi-attendee meeting scheduling. . . .	48
3.5	Output probabilities of NESAs in multi-attendee event scheduling given lunch and dinner events.	49
3.6	Output probabilities of NESAs in multi-attendee event scheduling given misspelled and non-English events.	50

List of Tables

1.1	Performance comparison of NER models at the entity mention level (E: ezTag, T: tmTool, P: PubTerm. The second highest scores are underlined.)	4
2.1	COMPLETE and PARTIAL overlap percentages of entity pairs. The numbers in each cell indicate the percentages of overlaps of the entity type pair in all COMPLETE (or all PARTIAL) overlaps. The numbers in parentheses are PARTIAL overlap percentages.	13
2.2	Examples of applying the decision rules of BERN to test sets	17
2.3	The multi-type normalization model and dictionaries of BERN	18
2.4	The performance of the multi-type normalization model (i.e., the combined normalization models) of BERN. The authors of tmChem did not report the normalization performance of tmChem independently. (P: Precision, R: Recall, F: F1-score, Acc.: Accuracy)	19
2.5	Normalization results of recognized entities in 19.4 million PubMed articles	19
2.6	Runtime statistics for 10K PubMed articles (seconds per article, STD: standard deviation)	20
2.7	BERN APIs and URL examples (PMID: PubMed ID)	22
2.8	Discovered new entity examples of each entity type (discovered new entities (bold) and known entities (underlined))	24
3.1	Event Scheduling Dataset Statistics	38
3.2	Hyperparameters of MLP and NESAs	41

3.3	Personal Event Scheduling Results	42
3.4	Multiple Attendee Event Scheduling Results	43
3.5	NESA Model Ablation (Diff. %: average performance difference % of 4 metrics)	44
3.6	Baseline Model Ablation	46
3.7	Nearest Neighbors (NNs) of Title Representations Given the Title Family lunch	47
3.8	Nearest Neighbors of Title/Intention Representations Given the Title App project work (duration 120 min.)	47

Chapter 1

Introduction

Unlike word2vec [1] and GloVe [2] which represent a word in a context-independent method, a contextual language representation model such as BERT [3] represents a word depending on a context of the word. And, inferring an answer taking into account a context around a target has been successful on various natural language processing (NLP) tasks. In recent years, contextualized language representation models [4, 3, 5] pre-trained on a large corpus have shown their effectiveness on downstream NLP tasks such as sentence classification [6], named entity recognition (NER) [7], and question answering [8, 9] even if a dataset of a task is small. To take advantage of a pre-trained language model’s effect on biomedical domain, we use BioBERT NER models of Lee et al. [10], and as a result we obtain high-quality contextualized **word** representations for biomedical NER. Also, we propose Neural Event Scheduling Assistant (NESA) which is a model which learns contextualized **event** representations for calendar event scheduling. The model considers events registered before an event in the same week as a context. Our qualitative analysis demonstrates that context representations have a significant impact on the overall event scheduling performance. As expected, BioBERT NER models and NESA outperform most of existing biomedical NER models and baseline event scheduling models.

1.1 Named entity recognition

There are over 30 million articles in PubMed as of November 2019, and the amount of biomedical literature has been growing rapidly in recent years. Fast and precise text mining tools can reduce the amount of effort and time it takes researchers to find and extract useful information from the vast amount of biomedical literature. Researchers have used named entity recognition (NER) and named entity normalization (NEN) models to develop effective biomedical text mining tools for information retrieval [11], question answering [12], relation extraction [13], and so on.

Existing Web-based text mining tools such as tmTool [14], ezTag [15], and PubTerm [16] have obtained excellent NER performance on various types of biomedical entities at the time of their publication. However, they have a few limitations. First, the Web-based text mining tools use older NER models which obtain lower performance than recent NER models. Moreover, pre-trained NER models such as tmChem [17] and DNorm [18] used by Web-based text mining tools cannot effectively discover new entities.

Second, the text mining tools do not consider the different types of entities that frequently overlap in NER results [19].¹ For instance, in “The androgen is synthesized from ...,” NER models can tag “androgen” as both a gene/protein and a drug/chemical because an androgen is a natural or synthetic steroid hormone. The correct entity type should be determined based the context of the sentence; in other words, if “androgen” refers to the synthetic hormone, NER models should tag “androgen” only as a drug/chemical. However, existing text mining tools have no pre-defined rules for such overlapping entities.

After NER, text mining tools need to normalize recognized entities since an entity can have multiple names (i.e., synonyms) and a name can be associated with multiple entities (i.e., polysemy). However, as normalization models vary greatly depending on the type of entity, it is difficult to build a text mining tool that performs normalization for multiple entity types. Generally, there are NER models for various types of biomedical entities in biomedical texts. However, it takes a considerable amount of time and effort to obtain

¹In the GENIA corpus, the percentage of nested named entities among all entities is 18.1%.

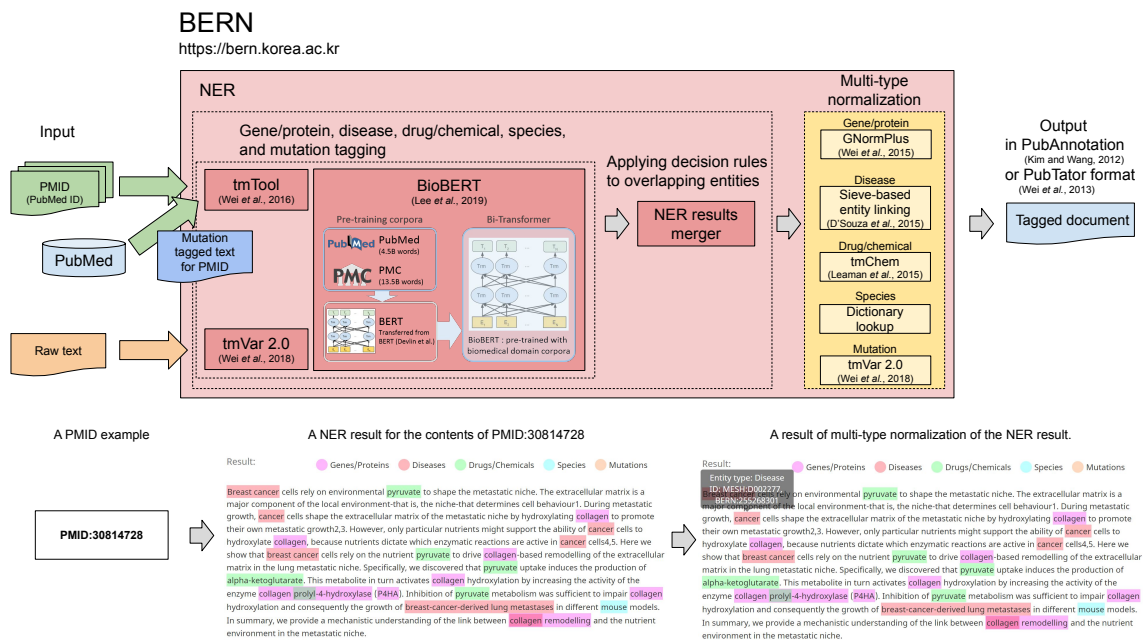


Figure 1.1: Overview of the RESTful Web service of BERN

computing resources and set up models for tagging entities.

We propose a neural biomedical named entity recognition and multi-type normalization (BERN) tool that recognizes known entities and discovers new entities, and identifies the types of overlapping entities. The overview of the BERN Web service is shown in Figure 1.1. When a PMID is inputted into BERN, it uses tmTool APIs to fetch texts annotated with mutations for the PMID. When raw text is inputted into BERN, it tags mutations in the text using tmVar 2.0 [20]. Next, BERN uses the BioBERT NER models to tag genes/proteins, diseases, drugs/chemicals, and species. After the multi-type NER, probability-based decision rules are applied to identify overlapping entities. Finally, normalization is performed for each entity type and the result is returned.

The BioBERT NER models obtained the highest F1-score in recognizing genes/proteins, diseases, and drugs/chemicals as shown in Table 1.1. Due to the lack of a high-quality public training set of mutations, BERN uses tmVar 2.0 as a pre-trained mutation NER model. Furthermore, BERN uses probability-based decision rules to determine whether

Table 1.1: Performance comparison of NER models at the entity mention level (E: ezTag, T: tmTool, P: PubTerm. The second highest scores are underlined.)

Text mining tools	Entity type	Pre-trained NER models	Test sets	Precision	Recall	F1-score
BERN	Gene/Protein	BioBERT [10]	BC2GM [21]	0.8516	0.8365	0.8440
–		Sachan et al. [22]		0.8181	<u>0.8157</u>	<u>0.8169</u>
–		MTM-CW [23]		<u>0.8210</u>	0.7942	0.8074
–		CollaboNet [24]		0.8049	0.7899	0.7973
E, T, P		GNormPlus [25]		0.7840	0.7920	0.7880
–		Giorgi and Bader [26]		0.7862	0.7871	0.7866
–		LSTM-CRF (iii) [27]		0.7750	0.7813	0.7782
BERN	Disease	BioBERT [10]	NCBI disease [28]	<u>0.8904</u>	0.8969	0.8936
–		Sachan et al. [22]		0.8641	<u>0.8831</u>	<u>0.8734</u>
–		MTM-CW [23]		0.8586	0.8642	0.8614
–		CollaboNet [24]		0.8548	0.8727	0.8636
–		Giorgi and Bader [26]		0.8262	0.8695	0.8472
–		LSTM-CRF (iii) [27]		0.8531	0.8358	0.8444
–		D3NER [29]		0.8503	0.8380	0.8441
–		Lou et al. [30]		0.9072	0.7489	0.8205
E		^a TaggerOne [31]		0.8510	0.8080	0.8290
E		^b TaggerOne [31]		0.8350	0.7960	0.8150
T, P		DNorm [18]		0.8030	0.7630	0.7820
BERN	Drug/Chemical	BioBERT [10]	BC4CHEMD [32]	<u>0.9223</u>	0.9061	0.9141
–		MTM-CW [23]		0.9130	0.8753	0.8937
–		CollaboNet [24]		0.9078	0.8701	0.8885
–		Giorgi and Bader [26]		0.8343	0.8883	0.8605
–		LSTM-CRF (iii) [27]		0.8783	0.8545	0.8662
–		Att-BiLSTM-CRF [33]		0.9229	<u>0.9001</u>	<u>0.9114</u>
T, P		tmChem (Model 2) [17]		0.8909	0.8575	0.8739
BERN	Species	^c BioBERT [10]	LINNAEUS [34]	<u>0.9384</u>	0.8611	0.8981
–		Giorgi and Bader [26]		0.9280	<u>0.9429</u>	<u>0.9354</u>
–		LSTM-CRF (iii) [27]		0.9357	0.9324	0.9340
–		LINNAEUS [34]		0.9710	0.9430	0.9570
E, T, P	Mutation	SR4GN [35]	tmVar2 [20]	0.8582	0.8528	0.8555
BERN, E		tmVar 2.0 [20]		0.9725	0.9040	0.9370
T, P		tmVar [36]	MutationFinder [37]	0.9880	0.8962	0.9398

^aJoint model of TaggerOne. ^bNER-only model of TaggerOne.

^cFor the LINNAEUS dataset, it is not clear whether training/dev/test sets of NER models are split in the same way. For species, the BioBERT NER model uses the LINNAEUS dataset split of Pyysalo (<https://github.com/spyysalo/linnaeus-corpus>).

to include all the overlapping entities or only the overlapping entities that are most likely to be entities predicted by the BioBERT NER models or tmVar 2.0.

We also combined multiple NER models into one multi-type normalization model and integrated it into BERN to assign IDs to recognized entities. The multi-type normalization model uses a high performance normalization model for each entity type, and uses a dictionary lookup such as SR4GN [35] for species. As a result, researchers can use the RESTful Web service of BERN to obtain NER and normalization results on PubMed articles or their raw text.

To the best of our knowledge, BERN is the first Web-based biomedical text mining tool that leverages neural network based NER models to recognize known entities and discover new entities. Our main contributions for biomedical NER and normalization are as follows:

- BERN is a biomedical text mining tool that uses neural network based high performance BioBERT NER models for recognizing known entities and discovering new entities.
- We developed the probability-based decision rules for identifying the types of overlapping entities after conducting case studies.
- BERN uses the multi-type normalization model to assign a specific ID to each recognized entity.
- BERN provides a Web service for tagging and normalizing entities in PubMed articles or raw text. The RESTful Web service of BERN and annotated PubMed articles are freely available at <https://bern.korea.ac.kr>.
- We made the code of BERN publicly available at <https://github.com/dmis-lab/bern>.

1.2 Event scheduling

Calendar data has become an important context source of user information due to the popularity of online calendar services such as Google Calendar and Outlook Calendar. According to a research study conducted by Promotional Products Association International in 2011, about 40% of people referred to calendars on their computers, and about

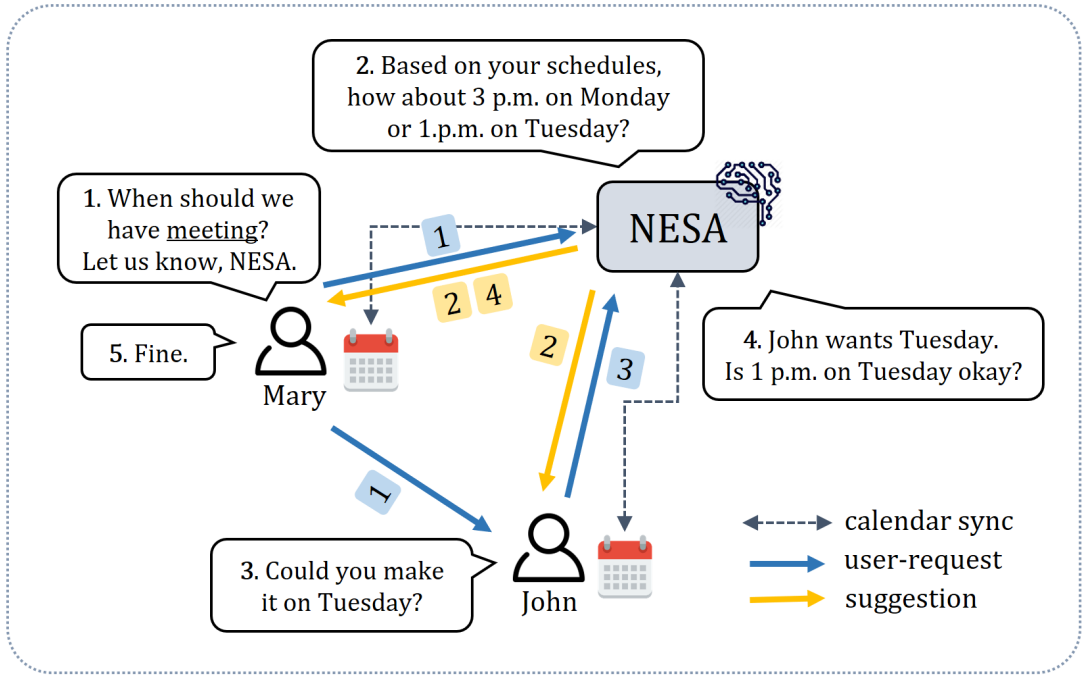


Figure 1.2: Example of calendar event scheduling. Mary requests NESA to schedule a meeting with John. NESA considers each user’s preference and calendar context, and the purpose of the event.

22% of people used their mobile calendars every day [38]. As more people use online calendar services, more detailed user information is becoming available [39].

Event scheduling is one of the most common applications that uses calendar data [40, 41]. Similar to Gervasio et al. [42] and Berry et al. [43], we define event scheduling as suggesting suitable time slots for calendar events given user preferences and calendar contexts. However, even with the development of communication technology, event scheduling is still time-consuming. According to Konolabs, Inc., the average number of emails sent between people to set the time for a meeting is 5.7.² At the same time, the market for digital assistants is growing fast. Gartner, Inc. stated that by 2019, at least 25% of households will use digital assistants on mobiles or other devices as primary interfaces of connected home services [44]. Thus, it is important for digital assistants to effectively

²Statistics obtained by Konolabs, Inc. (<https://kono.ai>) in 2017.

schedule users’ events [45].

An example of scheduling an event using NESAs is illustrated in Figure 1.2. When a user (**Mary**) requests NESAs to arrange an appointment with the other user (**John**), NESAs suggests candidate time slots considering the purpose of the event (e.g., **meeting**), preferences of each user (e.g., **Mary** usually has meetings in the afternoon, and **John** likes to have meetings early in the week), and each user’s calendar context. As a result, NESAs reduces the communication cost between the users by assisting with event scheduling.

Despite its importance, automated event scheduling [46, 41, 40] has had limited success due to several reasons. First, previous studies heavily relied on hand-crafted event features such as predefined event types, fixed office/lunch hours, and so on. In addition to the cost of defining the hand-crafted event features, they could not accurately understand calendar contexts based on natural language. For instance, if a user requests to schedule a **late lunch** with other users, traditional scheduling systems do not suggest late lunch hours unless the keyword **late** is registered in the systems. Furthermore, raw online calendars frequently contain abbreviations (e.g., **Mtg** stands for **meeting**) and misspellings. To deal with natural language, recent studies have combined human labor with scheduling systems [47]. Second, most previous studies have developed their own scheduling systems to learn user preferences, which makes it difficult to apply their methodologies to other scheduling systems. Despite the wide use of the Internet standard format iCalendar [48], developing scheduling assistants based on iCalendar gained much less attention among researchers [49].

In this paper, we propose Neural Event Scheduling Assistant (NESAs) which is a deep neural model that learns to schedule calendar events using raw user calendar data. NESAs is a fully automated event scheduling assistant which learns user preferences and understands raw calendar contexts that include natural language. To understand various types of information in calendars, NESAs leverages several deep neural networks such as Bidirectional Long Short-Term Memory (Bi-LSTM) [50, 51] and Convolutional Neural Network (CNN) [52]. The following four layers in NESAs are jointly trained to schedule personal calendar events: 1) Title layer, 2) Intention layer, 3) Context layer, and 4) Out-

put layer. After training, NESA is utilized for scheduling personal events (e.g., homework) and multi-attendee events (e.g., meetings). We compare NESA with previous preference learning models for event scheduling [53, 46, 42], and find that NESA achieves the best performance in terms of various evaluation metrics.

The contributions for event scheduling are four-fold.

- We introduce NESA, a fully automated event scheduling model, which learns user preferences and understands calendar contexts, directly from their raw calendar data.
- NESA successfully incorporates deep neural networks for event scheduling tasks.
- We train NESA on 593,207 real online calendar events in Internet standard format which is applicable to any calendar systems.
- NESA achieves the best performance on both personal and multi-attendee event scheduling tasks compared with other preference learning models.

Chapter 2

Contextualized word representations for named entity recognition

2.1 Background

2.1.1 Named entity recognition for biomedical text mining

For biomedical text, the word embeddings of Pyysalo et al. [54], which were trained on PubMed, PubMed Central Open Access (PMC OA) Subset, and English Wikipedia articles using word2vec [1], were widely used [27, 23, 29, 24, 26]. Existing biomedical NER models [35, 36, 17, 25, 20] often use conditional random fields (CRFs) [55] or semi-Markov linear classifiers [31] with dictionaries to find entities in the biomedical literature. A CRF is flexible in terms of feature selection; however, it is computationally expensive.

In recent years, with the success of deep neural networks, a bidirectional Long Short-Term Memory (BiLSTM) with CRF [27, 29, 23, 24, 26, 22] has been widely used to extract word sequence features. However, BiLSTM is limited to parallelization. The re-

cently proposed Transformer [56], which is more parallelizable, obtained high performance in natural language processing tasks without using recurrent networks or convolutional networks. The Transformer connects an encoder and a decoder through self-attention to be more parallelizable and to reduce its training cost. Also, BERT (Bidirectional Encoder Representations from Transformers) [3], which can be used to understand deep contextual bidirectional language representations, was proposed. BERT pre-trains its weights on English Wikipedia and BooksCorpus, and then fine-tunes the pre-trained weights for each task.

More recently, novel methods were proposed to obtain better contextualized language representations than BERT. The first is to replace static token masking of BERT with dynamic span masking of SpanBERT [57], and the second is to reduce sentence-order prediction (SOP) loss of ALBERT [5] instead of next-sentence prediction loss of BERT.

2.1.2 Resolving overlapping entities

Since entity mentions in biomedical text can overlap, it is necessary to decide which entities to select during or after NER. In previous studies, NER models were used to recognize entities in biomedical text even when the entity mentions overlap. Zhou [58] found patterns of nested entity names in the GENIA corpus, and proposed a pattern-based rule generation method to resolve the nested entity names. In recent years, Wang and Lu [59], and Katiyar and Cardie [60] proposed models for learning time-efficient hypergraph representations of overlapping entity mentions. Greenberg et al. [61] proposed a model which consists of BiLSTM and an expectation-maximization (EM) marginal CRF, and recognizes disjoint or partially overlapping sets of entity types. However, their model does not have rules for determining the types of an entity mention if the mention belongs to different entity type spans.

2.1.3 Named entity normalization models for biomedical text mining

As mentioned in Section 1, there are various types of entities which are referred to as different names in biomedical text. Thus, individual normalization models for each

entity type have been proposed rather than an integrated normalization model. Lower-case conversion and abbreviation resolution are the most commonly used for normalizing biomedical entities. tmTool, PubTerm, ezTag, and BERN commonly use GNormPlus [25] for gene normalization, SR4GN [35] for species normalization (only dictionary lookup for BERN), and tmVar [36, 20] for mutation normalization. GNormPlus uses exact match and bag-of-words match to pair recognized names with concepts in Entrez Gene [62]. Also, GNormPlus applies Ab3P [63] to extract abbreviation pairs. SR4GN normalizes recognized species to the most specific concept if possible. Also, tmVar detects pairs of mutations and dbSNP RSIDs [64] using pattern matching and dictionary lookup.

On the other hand, text mining tools use different models for disease and chemical normalization. ezTag uses TaggerOne [31] to normalize diseases and chemicals, and TaggerOne jointly performs NER and normalization using semi-Markov models. tmTool and PubTerm use DNorm [18], which is based on pairwise learning to rank (pLTR), to normalize diseases. BERN uses the sieve-based entity linking approach of D’Souza et al. [65] to normalize diseases. Among the sieve-based approaches, exact match, abbreviation expansion, and partial match were especially effective. tmChem [17], which is used by tmTool, PubTerm and BERN, converts recognized chemicals and chemical names in the lexicon of tmChem to lowercase letters, and removes whitespace and punctuation. The lexicon is collected from MeSH [66] and ChEBI [67]. A chemical name in short form that can be recognized by Ab3P is assigned the same ID as the chemical name in long form.

2.2 Methods

First, we describe the BioBERT NER models used for recognizing named entities in biomedical text, review cases of overlapping entities, and explain the decision rules developed for determining which entities to choose when they overlap. Next, we discuss the multi-type normalization model which normalizes the remaining entities.

2.2.1 BioBERT for named entity recognition

BioBERT NER models used by BERN recognize known entities and discover new entities using WordPiece [68] embeddings. The word embeddings of Pyysalo et al. [54] suffer from the out-of-vocabulary problem. If a word in a text is not in the vocabulary of the embeddings, the embeddings cannot provide a rich representation for the word. On the other hand, the WordPiece embeddings are a way of dividing a word into several units (i.e., sub-word units) and expressing each unit. As a result, the WordPiece embeddings can be used to extract features of rare or unknown words, which is very helpful in discovering new entities.

BioBERT is initialized with the case-sensitive version of BERT-Base. BioBERT additionally pre-trains its weights on PubMed articles and PMC OA Subset articles, and fine-tunes the pre-trained weights for downstream tasks. The BioBERT NER models are fine-tuned as follows:

$$p(y_i = k|T_i) = \text{softmax}(T_i W^\top + b)_k, k = 0, 1, \dots, 6 \quad (2.1)$$

where p denotes the label probability, and $T_i \in \mathbb{R}^H$ denotes the final hidden representation for each token i . H is the hidden size, $W \in \mathbb{R}^{K \times H}$ is a classification layer, b is a bias, and K is 7. The classification loss L is calculated as follows:

$$L(\Theta) = -\frac{1}{N} \sum_{i=1}^N \log(p(y_i|T_i; \Theta)) \quad (2.2)$$

where Θ denotes trainable parameters, and N denotes sequence length.

BioBERT NER models compute the probabilities of the following seven tags: IOB2 tags (“I”inside, “O”outside, “B”egin) [69], “X” (a sub-token of WordPiece), “[CLS]” (the first token of every sequence for classification), “[SEP]” (a delimiter between sequences), and “PAD” (padding) of each word in a sentence. Note that the BioBERT NER models make predictions for “I,” “O,” and “B” tags but not for the “X,” “[CLS],” “[SEP],” or “PAD” tags. Words in a sentence are obtained using a tokenizer on a dataset with

Table 2.1: COMPLETE and PARTIAL overlap percentages of entity pairs. The numbers in each cell indicate the percentages of overlaps of the entity type pair in all COMPLETE (or all PARTIAL) overlaps. The numbers in parentheses are PARTIAL overlap percentages.

Entity type	Gene/ Protein	Disease	Drug/ Chemical	Species	Mutation
Gene/ Protein		1.76 (9.51)	12.55 (32.59)	0.17 (<u>12.83</u>)	0.43 (1.49)
Disease	1.76 (9.51)		1.47 (5.57)	<u>11.65</u> (8.93)	0.003 (0.012)
Drug/ Chemical	12.55 (32.59)	1.47 (5.57)		0.03 (0.06)	0.06 (0.87)
Species	0.17 (<u>12.83</u>)	<u>11.65</u> (8.93)	0.03 (0.06)		0.00001 (0.00163)
Mutation	0.43 (1.49)	0.003 (0.012)	0.06 (0.87)	0.00001 (0.00163)	

labels in CoNLL format [70] and then the sub-words of each word are obtained using the WordPiece tokenizer.

As a result, BERN can discover new entities using BioBERT NER models. As shown in Table 1.1, the BioBERT NER models used by BERN obtain the highest F1-scores on the test sets for all types except species. The BioBERT NER models of BERN outperform the NER models of tmTool, PubTerm, and ezTag on test sets of genes/proteins (BC2GM 5.6%), diseases (NCBI disease 6.46%), drugs/chemicals (BC4CHEMD 4.02%), and species (LINNAEUS 4.26%) in terms of F1-score. Also, the BioBERT NER models outperform the BiLSTM-CRF based NER models [27, 23, 24, 29, 33], the multi-task NER model [23], and the transfer learning NER models [26, 22] in recent years, on all the test sets except for the test set of species (LINNAEUS). Note that BioBERT NER models can recognize all types of entities if there is training data.

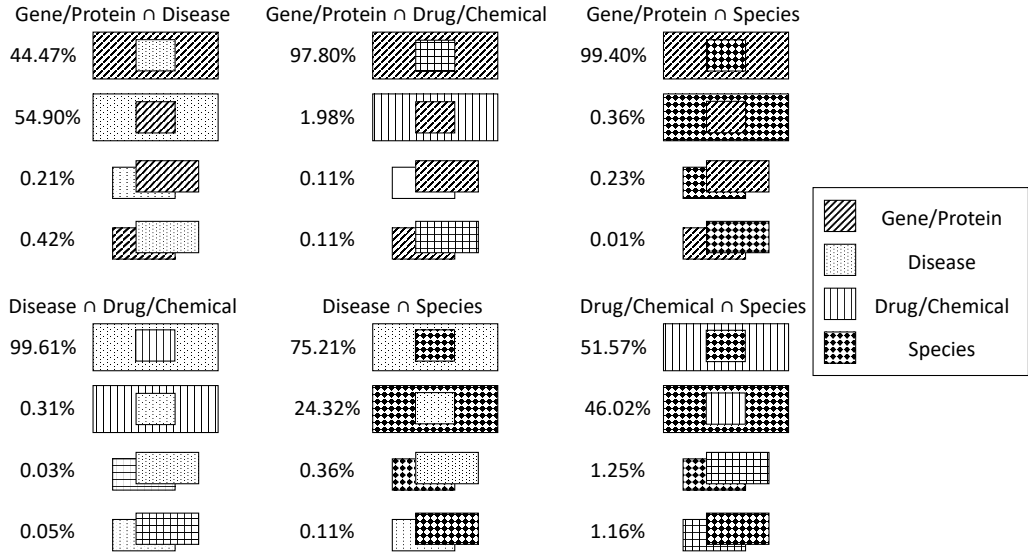


Figure 2.1: The percentages of partial overlapping entities of each entity pair

2.2.2 Decision rules for overlapping entities

Case studies

We performed comprehensive case studies on overlapping entities. First, we found that 26.2% of entities in 18.6 million PubMed articles¹ overlap. Among the entities in the articles, we also found 3.8 million cases where two or more entities overlap completely (COMPLETE overlap), and 9.6 million cases where they partially overlap (PARTIAL overlap). Table 2.1 shows the COMPLETE and PARTIAL overlap percentages of entities. Genes/proteins, diseases, and drugs/chemicals usually overlap. Genes/proteins and drugs/chemicals completely overlap the most (12.55%) of all entity types. Also, genes/proteins and drugs/chemicals have the largest number of PARTIAL overlaps (32.59%). Species usually overlap with genes/proteins (PARTIAL 12.83%) or diseases (COMPLETE 11.65%, PARTIAL 8.93%), and mutations generally overlap with genes/proteins (COMPLETE 0.43%, PARTIAL 1.49%). The proportion of species overlapping partially with

¹We excluded PubMed articles that have titles but no abstracts.

genes/proteins (12.83%) is much greater than that of species overlapping completely with genes/proteins (0.17%) because there are many cases where genes/proteins are associated with *Homo sapiens*.

For a more detailed analysis, we calculate the PARTIAL overlap percentages of each entity pair, which are shown in Figure 2.1. In our dataset, when genes/proteins and diseases partially overlap, a gene/protein mention is a part of a disease mention in more than half of cases. And for each pair, a drug/chemical or a species mention is a part of a gene/protein or a disease mention. The sum of percentages of overlaps between the end of an entity and the beginning of another entity is quite low ($< 1\%$ except for $\langle \text{drugs/chemicals and species} \rangle$).

Decision rules

After conducting the case studies above, we developed decision rules for overlapping entities in multi-type NER results. Unlike the model of Greenberg et al. [61], BERN uses decision rules for determining which entities to choose if overlapping entities are found. Figure 2.2 shows the decision rules that BERN uses for the multi-type NER results. First, if the entities do not overlap completely (71.3% in our dataset), BERN tags all the entities. If entities overlap completely and there is a mutation among the entities (0.5% ($28.7\% \times 1.9\%$)), BERN tags the mutation and entities that the mentions are most likely to be entities predicted by BioBERT NER models. Next, if entities completely overlap and all the entities are non-mutations (28.2% ($28.7\% \times 98.1\%$)), only the entity with the highest probability of being an actual entity is tagged.

Because mutations in a text are typically in distinct formats, which makes it easier to more accurately recognize them, mutation NER models such as tmVar 2.0 used by BERN generally achieve much higher precision (over 97%) on mutations than on other entity types. In our evaluation, in most cases, if there is a mutation among the overlapping entities, all the remaining overlapping entities are wrong and the mutation is the correct answer.

We applied the decision rules to each test set, and the results are shown in Table 2.2.

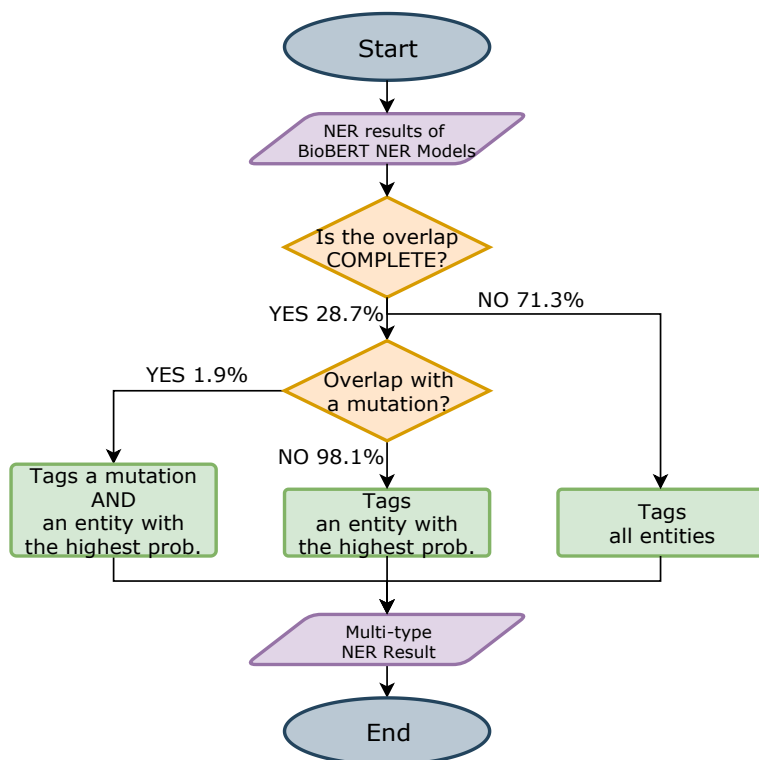


Figure 2.2: Decision rules for NER results

Since each test set in Table 2.2 has labels only for a particular entity type, there is no performance change if the NER model for the entity type labeled in the test set predicts that mentions are most likely to be entities of the entity type (e.g., In NCBI disease corpus, a NER model for diseases predicts that a mention is a disease.). If the NER models for the other entity types make stronger predictions on which mentions are most likely to be entities on wrong mentions than predictions of the NER model for the entity type (e.g., In BC2GM, a NER model for drugs/chemicals predicts that a non-gene/protein mention is a drug/chemical.), they can avoid giving the wrong answer due to the decision rules, which helps reduce the false positive rate. Conversely, if the NER models for other entity types make stronger predictions on correct mentions than predictions of the NER model for the entity type (e.g., In BC4CHEMD, a NER model for diseases predicts that a drug/chemical mention is a disease.), the true positive rate is reduced. Therefore, the

Table 2.2: Examples of applying the decision rules of BERN to test sets

Labeled entity type	Test set	Case	Prediction	Example
Gene/Protein	BC2GM	Preventing wrong answers	Drug/chemical	Results of these studies indicate that binding of biotin to the protein results in protection of regions of the central domain in the vicinity of the active site and the C-terminal domain from chemical cleavage.
Gene/Protein	BC2GM	Preventing correct answers	Drug/chemical	Similarly, TAM-67 reverted the morphology of the AdoMetDC-antisense expressors.
Disease	NCBI disease	Preventing wrong answers	Gene/protein	A century after its recognition as a syndrome by Vaughan Pendred, the disease gene (PDS) was mapped to chromosome 7q22-q31.
Disease	NCBI disease	Preventing correct answers	Gene/protein	The APC gene was analysed in 190 unrelated FAP and 15 non-FAP colorectal cancer patients using denaturing gradient gel electrophoresis.
Drug/chemical	BC4CHEMD	Preventing wrong answers	Species	Understanding the interactions between metabolites isolated from Achyrocline satureioides in relation to its antibacterial activity.
Drug/chemical	BC4CHEMD	Preventing correct answers	Gene/protein	Pinosylvin Induces Cell Survival, Migration and Anti-Adhesiveness of Endothelial Cells via Nitric Oxide Production.

decision rules improve the precision and lower the recall.

2.2.3 The multi-type normalization model

BERN uses the multi-type normalization model to more clearly distinguish entities. Table 2.3 shows the statistics of normalization models used by BERN. We added the disease names in the PolySearch2 dictionary (76,001 names of 27,658 diseases) to the sieve-based entity linking dictionary (76,237 names of 11,915 diseases) to increase the number of normalizable entities. We also added the drug names in DrugBank [71] and US FDA to the tmChem dictionary. Due to the lack of normalization models for species, BERN normalizes species by dictionary lookup, as mentioned above. Using tmVar 2.0, we made a dictionary of mutations with normalized mutation names; a mutation with several names was given one normalized name or ID.

According to the statistics, drugs/chemicals have the highest number of unique IDs (40% of the total), and species have the most names per entity. If the normalization model fails to normalize a recognized entity, the model returns a Concept Unique Identifier-less

Table 2.3: The multi-type normalization model and dictionaries of BERN

Entity type	Normalization model	Dictionaries	# of IDs	# of names	Avg. # of names per ID
Gene/Protein	GNormPlus	Entrez Gene [62]	139,375	248,581	1.8
Disease	Sieve-based entity linking [65]	MeSH [66], OMIM [72], SNOMED-CT [73], PolySearch2 [74]	32,954	172,650	5.2
Drug/Chemical	tmChem without Ab3P	MeSH [66], ChEBI [67], DrugBank [71], *FDA approved drugs	518,223	2,571,570	5.0
Species	Dictionary lookup	NCBI Taxonomy [75]	398,037	3,119,005	7.8
Mutation	tmVar 2.0	dbSNP [64], ClinVar [76]	208,474	302,498	1.5
Total			1,297,063	6,414,304	4.9

*<https://www.fda.gov>

(CUI-less) annotation for the entity.

Table 2.4 shows the performance of the multi-type normalization model (i.e., integrated normalization models) of BERN. For genes/proteins, there are 75 kinds of species in the BC3 Gene Normalization (BC3GN) test set, but GNormPlus focuses on only 7 kinds of species. As a result, GNormPlus obtains a much lower F1-score of 36.6% on the multispecies test set (BC3GN) than F1-score on the human species test set (BC2GN). For mutations, tmVar 2.0 achieved high F1-scores close to 90% on two corpora.

Also, we tested the multi-type normalization model on 19.4 million PubMed articles. The results are shown in Table 2.5. Because we constructed a gene dictionary of mostly *Homo sapiens* (i.e., human species), the percentage of normalized genes is low (53.4%). The result obtained by the sieve-based entity linking model had a high percentage (90.2%) of normalized diseases. Also, the percentage of normalized species was the highest (94.8%) and species were mentioned in most of the sample articles (70%). Mutations were mentioned the least in the articles (1.6%), but the percentage of normalized mutations was 100% since tmVar 2.0 for mutations always provides the normalized names of recognized entities. Although drugs/chemicals have the highest number of IDs and the second largest

Table 2.4: The performance of the multi-type normalization model (i.e., the combined normalization models) of BERN. The authors of tmChem did not report the normalization performance of tmChem independently. (P: Precision, R: Recall, F: F1-score, Acc.: Accuracy)

Entity type	Normalization model	Test sets	P %	R %	F %	Acc. %
Gene/Protein	GNormPlus	BC2GN, human [21]	87.1	86.4	86.7	–
		BC3GN, multispecies [77]	–	–	50.1	–
Disease	Sieve-based entity linking	ShARE/CLEF eHealth [78]	–	–	–	90.75
		NCBI disease	–	–	–	84.65
Mutation	tmVar 2.0	OSIRISv1.2 [79]	97.20	80.62	88.14	–
		Thomas [80]	89.94	88.24	89.08	–

Table 2.5: Normalization results of recognized entities in 19.4 million PubMed articles

Entity type	# of recognized entities	# of normalized entities	# of articles with each entity type
Gene/Protein	73,655,197	39,299,648 (53.4%)	7,844,921
Disease	91,204,877	82,242,319 (90.2%)	12,461,907
Drug/Chemical	76,367,837	63,409,437 (83.0%)	9,568,465
Species	60,389,187	57,275,053 (94.8%)	13,580,610
Mutation	1,279,525	1,279,525 (100%)	310,210

number of names in the dictionary as shown in Table 2.3, the percentage of normalized drugs/chemicals is 83.0%, which may be because Ab3P (abbreviation resolution), which is used by tmChem, was not applied.

2.3 Implementation

The RESTful Web service of BERN was implemented using Python and Node.js. BERN run BioBERT NER models which are pre-trained with TensorFlow², on our server to recognize incoming biomedical text such as PubMed articles and raw texts. The server

²<https://www.tensorflow.org>

Table 2.6: Runtime statistics for 10K PubMed articles (seconds per article, STD: standard deviation)

Models	Average \pm STD
Getting a mutation tagged PubMed article using tmTool APIs	1.254 ± 0.103
Gene/protein, disease, drug/chemical, and species NER	0.411 ± 0.325
Multi-type normalization	0.022 ± 0.043
Total	1.688 ± 0.355

specifications are as follows:

- Operating system: Ubuntu 18.04.2 LTS
- CPU: Intel Xeon E5-2687W v3
- RAM size: 128 GB
- GPU: NVIDIA Titan X (Pascal) with 12 GB of memory
- HDD size: 2 TB

Four BioBERT NER models for genes/proteins, diseases, drugs/chemicals, and species, use 2.4 GB (4 \times 0.6 GB) of GPU memory. We use 8 NVIDIA V100 GPUs for pre-training BioBERT, and we use a NVIDIA Titan X GPU for making predictions. And, we fine-tune BioBERT NER models on the following training datasets: BC2GM for genes/proteins, NCBI disease for diseases, BC4CHEMD for drugs/chemicals, and LINNAEUS for species. GNormPlus uses 8 to 16 GB, and tmVar 2.0 uses 4 to 8 GB of memory. And, the load time of the GNormPlus gene dictionary is about 5 seconds and the load time of the tmVar 2.0 part-of-speech tagger is about 1 second. To reduce their load time, we run GNormPlus and tmVar 2.0 processes in the background on the server.

Table 2.6 shows the runtime statistics of BERN. The statistics show that tmTool API calls and the NER models used by BERN have the longest time (98.6%) in each run.

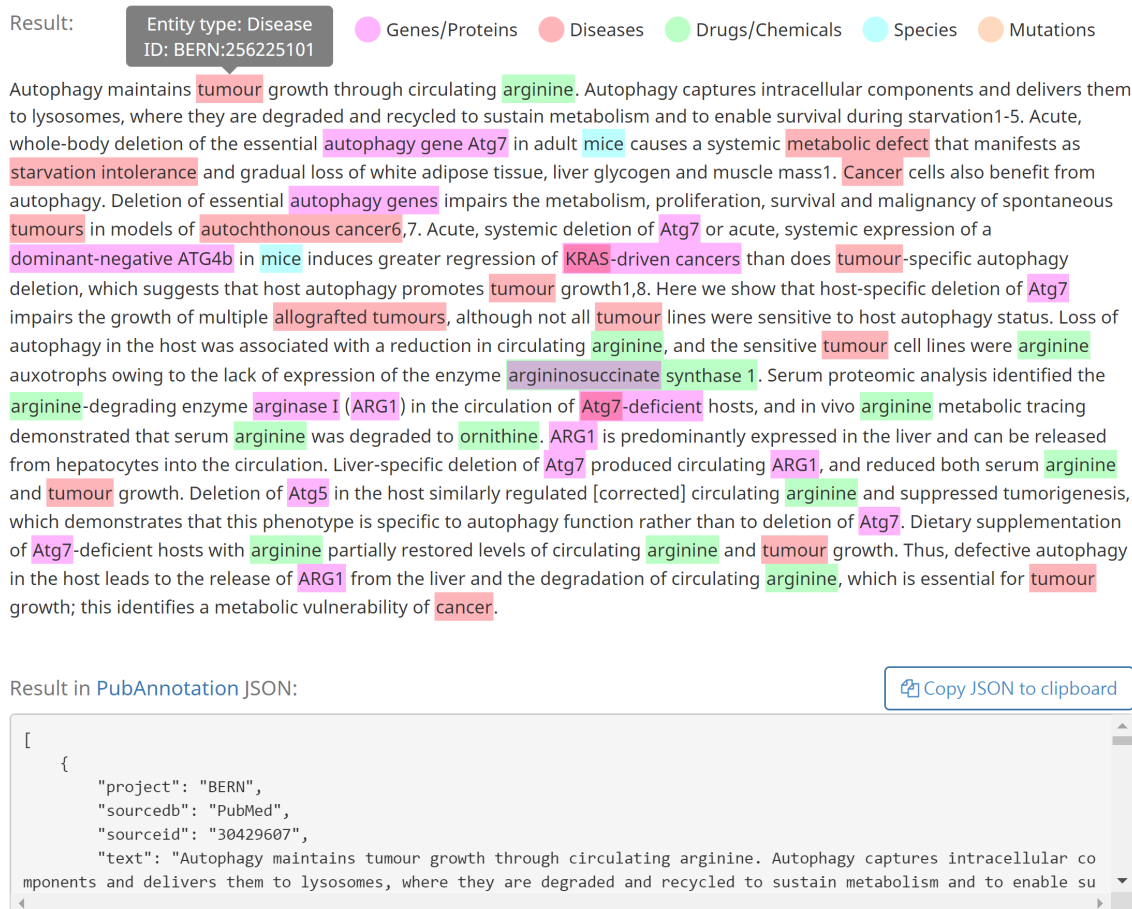


Figure 2.3: BERN demonstration of PMID:30429607 (best viewed in color mode)

If there is no recognized entity, the multi-type normalization model is not used. In this experiment, only one article was assigned to each batch.

2.3.1 Demonstrations

In the “Text” tab of BERN Web service, researchers can obtain NER+NEN results of submitted raw text in PubAnnotation JSON format, and see the visualized results under the text window. Also, as the BERN demonstration shows, entities are highlighted in their entity type color. When the mouse cursor is placed on an entity name, its entity type and entity ID are displayed in a tooltip. Figure 2.3 shows a BERN demonstration

Table 2.7: BERN APIs and URL examples (PMID: PubMed ID)

API	URL example
Single PMID	https://bern.korea.ac.kr/pubmed/29446767
PMIDs	https://bern.korea.ac.kr/pubmed/29446767,25681199
In PubTator format	https://bern.korea.ac.kr/pubmed/29446767/pubtator
Raw texts	https://bern.korea.ac.kr/plain/

which uses the title and abstract of an article (PMID:30429607). Also, in the “PMID” tab of BERN Web service, researchers can enter one or more PMIDs to obtain results in PubAnnotation JSON or PubTator format.

2.3.2 Application programming interfaces

BERN APIs return NER+NEN results for PMIDs and raw texts. For PMIDs, the Uniform Resource Locator (URL) form used by BERN APIs is,

[https://bern.korea.ac.kr/pubmed/<PMID\(s\)>\[/pubtator\]](https://bern.korea.ac.kr/pubmed/<PMID(s)>[/pubtator]).

In this URL form, the PMID parameter is required but the format parameter, “/pubtator”, is optional. For the convenience of researchers, we made it possible to obtain NER+NEN results by simply including one or more PMIDs in the URL. For the “Single PMID” URL of Table 2.7, BERN returns an NER+NEN result for a PubMed article with the following PMID:29446767. In addition, researchers can enter multiple comma-separated PMIDs to obtain NER+NEN results for multiple PubMed articles at the same time.³ For example, Table 2.7 shows the “PMIDs” URL for two PMIDs.

PubAnnotation JSON is the default format of the result of the APIs, which is also the format of the result of the demonstrations. Also, researchers can obtain results in PubTator format by adding “/pubtator” to the end of a URL. The “In PubTator format”

³Note that BERN allows up to 10 PMIDs at a time.

URL of Table 2.7 is an example of the result in PubTator format. In the PubTator format result, the title is included in the first line, the abstract is included in the second line, and then the NER+NEN result $\langle \text{PMID}, \text{start offset}, \text{end offset}, \text{entity name}, \text{entity type}, \text{entity ID} \rangle$ of BERN is in each line. Furthermore, BERN uses a HTTPS POST method where researchers must include their raw texts and the “Raw texts” URL in Table 2.7 in their code. For ease of use, we provide a sample Python code for API calls at <https://bern.korea.ac.kr>.

2.4 Discussion

2.4.1 Use cases

There are many use cases where BERN can be used.

- **Discovery of new named entities:** As mentioned earlier, BioBERT NER models can be used to discover new entities from the latest biomedical literature. Table 2.8 shows new entity examples of each type discovered by the BioBERT NER models. Although the new entities are not in the dictionaries of BERN, the BioBERT NER models can discover the new entities. For instance, the chemical compound “pentandricine” is not included in the dictionaries of BERN, but the BioBERT NER models accurately recognize the entity as a new chemical compound. Note that BioBERT NER models usually discovers new entities when sufficient contextual information (i.e., a complete sentence) is given (e.g., the chemical compound “osimertinib” without enough context may not be recognized by a BioBERT NER model).
- **Information retrieval:** BERN can serve as a fundamental NER+NEN model for various text mining tools. In the field of information retrieval, entity-based search engines such as LitVar [11] and BEST [81] can use BERN to find entities in queries and documents. BERN can greatly improve the performance of entity search engines in finding co-occurring entities in queries and documents.

Table 2.8: Discovered new entity examples of each entity type (discovered new entities (bold) and known entities (underlined))

Entity type	Example sentences
Gene/Protein	... To decipher the synaptic substrate of hyperexcitability, we examined pan-neuronal <u>Tsc1</u> knockout <u>mouse</u> and found a reduction in surface expression of a GABA receptor (<u>GABAR</u>) subunit but not AMPA receptor (AMPAR) subunit. ... (PMID:30683131)
Gene/Protein	... This metabolite in turn activates <u>collagen</u> hydroxylation by increasing the activity of the enzyme collagen prolyl-4-hydroxylase (<u>P4HA</u>). ... (PMID:30814728)
Disease	... A recent observational study in a large cohort of <u>critically ill</u> patients confirms the association between hyperlactatemia and mortality. ... (PMID:19691816)
Disease	... In contrast, high-performance liquid chromatography tandem mass spectrometry showed hypercholanaemia and high concentrations of biliverdin IX α in serum, urine, bile and milk. Hyperbiliverdinaemia disappeared after surgical correction of the <u>cholestasis</u> (PMID:21278388)
Drug/Chemical	... MEK inhibition with <u>trametinib</u> synergized with osimertinib to block growth. Alternately, a pan-RAF inhibitor as a single agent blocked growth of all cell lines with mutant EGFR and BRAF fusion (PMID:30831205)
Drug/Chemical	... A new <u>limonoid</u> , pentandricine (1), along with three known <u>limonoids</u> , ceramicine B (2), 6-de(acetyloxy)-23-oxochisocheton (3), 6-de(acetyloxy)-23-oxo-7-O-deacetylchisocheton (4), have been isolated from the stembark of <u>Chisocheton pentandrus</u> (PMID:29368952)
Species	... Specific evidence of coastal contamination of the marine ecosystem with the zoonotic protozoan parasite, <u>Toxoplasma gondii</u> , and extensive infection of southern sea otters (Enhydra lutris nereis) along the California coast was documented by this study. ... (PMID:12076629)
Species	The complete mitochondrial genome sequence of the Chinese Serow, Capricornis milneedwardsii (Cetartiodactyla: Caprinae). ... (PMID:24438312)

- Question answering: BERN can recognize biomedical named entities in questions and passages in question answering tasks such as BioASQ Task B [82, 83], and help improve performance, especially on “what” and “which” questions by classifying whether a span in a passage is an entity or not.

- Relation extraction: BERN can generate rich datasets for downstream biomedical text mining tasks such as relation extraction [84]. For instance, BERN can easily extract sentences with two or more recognized named entities from a biomedical corpus. Such sentences can be annotated, and the relationship of the recognized entities can be extracted from an existing database to generate a training dataset.
- A useful text mining tool: Using APIs, researchers can obtain NER+NEN results for texts from highly accessible Web services. Researchers can use commonly used entity IDs (e.g., HGNC IDs for genes, and MeSH IDs for diseases) [85] in the results of BERN more effectively for their text mining tasks.

2.4.2 Advantages and limitations of having a separate NER model for each entity type

Using a separate NER model for each entity type has the following advantages. First, the best performing model can be used for each entity type. In BERN, we can substitute the BioBERT NER models with new state-of-the-art models. Second, adding a new NER model for a different entity type to existing NER models is relatively easy. On the other hand, a single NER model that recognizes multiple entity types may need to be trained again on the dataset due to the changes in the architecture of the model.

However, having a separate NER model for each entity type can lower the efficiency. Multithreading can be performed to reduce the processing time, but it requires a larger number of computing resources. Also, it is possible to improve NER performance by multi-task learning which train a NER model to multiple tasks. A multi-task NER model can show higher performance than single-task models for various tasks with relatively few computing resources.

2.4.3 Additional dependencies of BERN

As mentioned in the Section 1, since BioBERT does not have a mutation NER model, BERN uses tmVar 2.0 and tmTool APIs for mutations. If we can obtain a high-quality

training set and build a mutation NER model that achieves higher performance than tmVar 2.0, BERN would not have to use tmVar 2.0 and tmTool APIs.

2.4.4 Overcoming the network distance between a server and a client

The network delay tends to increase with the distance between a server and a client. In such cases, cloud computing can be a solution. Researchers can run a cloud machine in the region closest to the server to reduce the distance between the server and the client. For example, a researcher can launch Elastic Compute Cloud instances or Lambda functions of Amazon Web Services in the same region or near the region of the server.

Chapter 3

Contextualized event representation learning for scheduling

3.1 Background

3.1.1 Preference learning for event scheduling

Since the development of online calendars, researchers have focused on learning user preferences for scheduling calendar events. Mitchell et al. proposed Calendar Apprentice (CAP) which is a decision tree based calendar manager that can learn user scheduling preferences from experience [46]. Blum et al. introduced the Winnow and weighted-majority algorithms that outperformed the CAP on predicting various attributes of calendar events [41]. Mynatt et al. also utilized the context of a user’s calendar to infer the user’s event attendance [86]. Berry et al. proposed an assistant called Personalized Calendar Assistant (PCalM), which is based on Naive Bayesian, for ranking candidate schedules [40]. Refanidis et al. have developed an intelligent calendar assistant which

uses a hierarchical preference model [87].

However, most event scheduling models were based on specific calendar systems using hand-crafted event features such as predefined event types and system dependent features. Previous scheduling methodologies are rarely used for modern event scheduling systems due to the high cost of designing hand-crafted features. Also, it is difficult for existing models to understand user calendars that often include user written texts such as event titles. In this chapter, we propose NESAs which learn to schedule calendar events, directly using raw calendar data that contains natural language texts. As NESAs are trained on the Internet standard format, it is generally applicable to other calendar systems.

3.1.2 Multi-attendee event scheduling

Event scheduling has also been studied in the context of multi-attendee event scheduling. Researches on event scheduling focus on solving constraint satisfaction problems (CSPs), and such researches often assume that user preferences are already given. Garrido et al. used heuristic functions for finding the priority value of each available time interval [88]. Wainer et al. proposed a model to find optimal time intervals based on user preferences and dealt with privacy issues of shared calendars [49]. Zunino et al. developed Chronos, a multi-attendee meeting scheduling system that employs a Bayesian network to learn user preferences [89].

However, most multi-attendee event scheduling models still depend on their own scheduling systems. Furthermore, due to the small amount of existing calendar event data (e.g., 2K events of 2 users [46, 41, 89]), some of the previous studies [41, 88] use complicated heuristic functions based on system dependent features to find proper time intervals, making their methodologies difficult to adopt. In contrast, NESAs leverage 593K standard formatted events and learn event scheduling directly from raw calendar data. While the recent work of Cranshaw et al. relied on human labor for more effective event scheduling [47], our event scheduling assistant is fully automated. We also demonstrate the effectiveness of NESAs on multi-attendee event scheduling.

3.1.3 Representation learning using deep neural networks

Many classification tasks such as image classification [52], sentiment analysis [90], and named-entity recognition [91] have benefited from the recent rise of neural networks. Deep neural networks learn how to represent raw inputs such as image pixels for any targeted task. Given a raw user calendar, NESA learns how to represent user preferences and calendar contexts for event scheduling. While the preliminary work of Mitchell et al. showed that decision tree based models with hand-crafted features are better than artificial neural network (ANN) based models with hand-crafted features [46], our work is the first to show that deep neural networks are effective for event scheduling tasks with raw calendar data.

Among various kinds of neural networks, Recurrent Neural Networks (RNNs) have achieved remarkable performance on natural language processing (NLP) tasks such as language modeling [92], machine translation [93], and so on. Inspired by a character-level language model [94] and a bidirectional attention flow (BiDAF) model for question answering [95], NESA handles various semantics coming from raw calendar events based on natural language. We use RNN and CNN to effectively represent user written event titles, and use Highway Network [96] to find nonlinear and linear relationships among various calendar attributes.

3.2 Problem formulation

3.2.1 Attributes of calendar data

A user’s calendar data consists of sequences of events which are sorted by their registered time. Each calendar event has at least five attributes: (1) *title* (what to do), (2) *start time*, (3) *duration*, (4) *registered time*, and (5) *user identifier* of an event. Although many other attributes (e.g., *location*, *description*) exist, we focus on the most common attributes of events. Note that the title of each event in iCalendar format does not have a label that indicates the event type, whereas previous scheduling systems rely

on a predefined set of event types.

To simplify the problem, we group all the events of each user by the week in which their events start. For example, user A’s events that start within the 46th week of 2019 will be grouped in A_2019_46. In each group, events are sorted by their registered time. For each user, all the K events in a specific week can be expressed as follows: $E = e_1, \dots, e_K$, and $e_i = (x_i, t_i, d_i, u_i)$ for $i = 1$ to K where x_i indicates the start time, t_i is the title, d_i is the duration, and u_i is the user identifier of event e_i . We assume that u_i represents the *preference* of a user, t_i and d_i represent the *purpose* of i -th event, and e_1, \dots, e_{i-1} represent the *context* of i -th event. Note that the context can be extended to multiple weeks.

3.2.2 Personal event scheduling

Event scheduling involves considering users’ preferences and calendar contexts to provide suitable time slots to users. We define personal event scheduling as scheduling events that have a single attendee (e.g., work, personal matters, and so on). We later describe how to extend personal event scheduling to multi-attendee event scheduling.

Personal event scheduling should consider the pre-registered events of the week (i.e., context) in which an event will be registered and the preferences of a user. Thus, an event scheduling model predicts the start time y_i of the i -th event e_i given the pre-registered events (e_1, \dots, e_{i-1}) which constitute the context of the week, and given the title t_i , duration d_i , and user u_i attributes of the i -th event e_i . Note that each pre-registered event also contains title, duration, user, and start time (x_i) attributes, making it difficult for any models to leverage all the given contexts.

Given the probability of target time slot y_i of event e_i , the optimal model parameters Θ^* are as follows:

$$\Theta^* = \underset{\Theta}{\operatorname{argmax}} p(y_i | e_1, \dots, e_{i-1}, t_i, d_i, u_i; \Theta) \quad (3.1)$$

where Θ denotes the trainable parameters of a model. Note that there exist K event scheduling problems in a week including weeks with no pre-registered events. We treat

each event scheduling problem as an independent problem to measure the ability of each model to understand calendar contexts and user preferences.

3.2.3 Multi-attendee event scheduling

Multi-attendee event scheduling further considers the preferences and calendar contexts of multiple users attending an event. Given U users attending a specific event e_μ with the optimal model parameter Θ^* , the most suitable time slot y_μ^* among candidate time slots \hat{y}_μ is computed as follows:

$$y_\mu^* = \underset{\hat{y}_\mu}{\operatorname{argmax}} \sum_{j=1}^U p(\hat{y}_\mu | E_{1:\mu-1}^j, t_\mu, d_\mu, u_j; \Theta^*) \quad (3.2)$$

where $E_{1:\mu-1}^j$ denotes a group of j -th user’s pre-registered events before the event e_μ (i.e., calendar context). In this way, we choose a time slot that maximizes the satisfaction of multiple users. Note that the number of pre-registered events may differ between users. Also, while we have assumed all users have the same influence in multi-attendee event scheduling, more sophisticated aggregation such as multiplying a weighting factor for each user is possible. However, we use the simplest form of aggregation to test the effectiveness of each model trained on personal event scheduling data.

3.3 Methodology

To deal with various types of raw calendar attributes, we propose NESAs which consist of four different layers: 1) Title layer, 2) Intention layer, 3) Context layer, and 4) Output layer. The Title layer aims to represent the meaning of user written event titles using both the words and characters of the titles. In the Intention layer, our model utilizes title, duration, and user representations to learn user preferences and understand the purpose of events. The Context layer consists of multiple convolutional layers for understanding raw calendar contexts. Finally, the Output layer computes the probability of each time slot based on the representations from each layer. The architecture of NESAs is illustrated

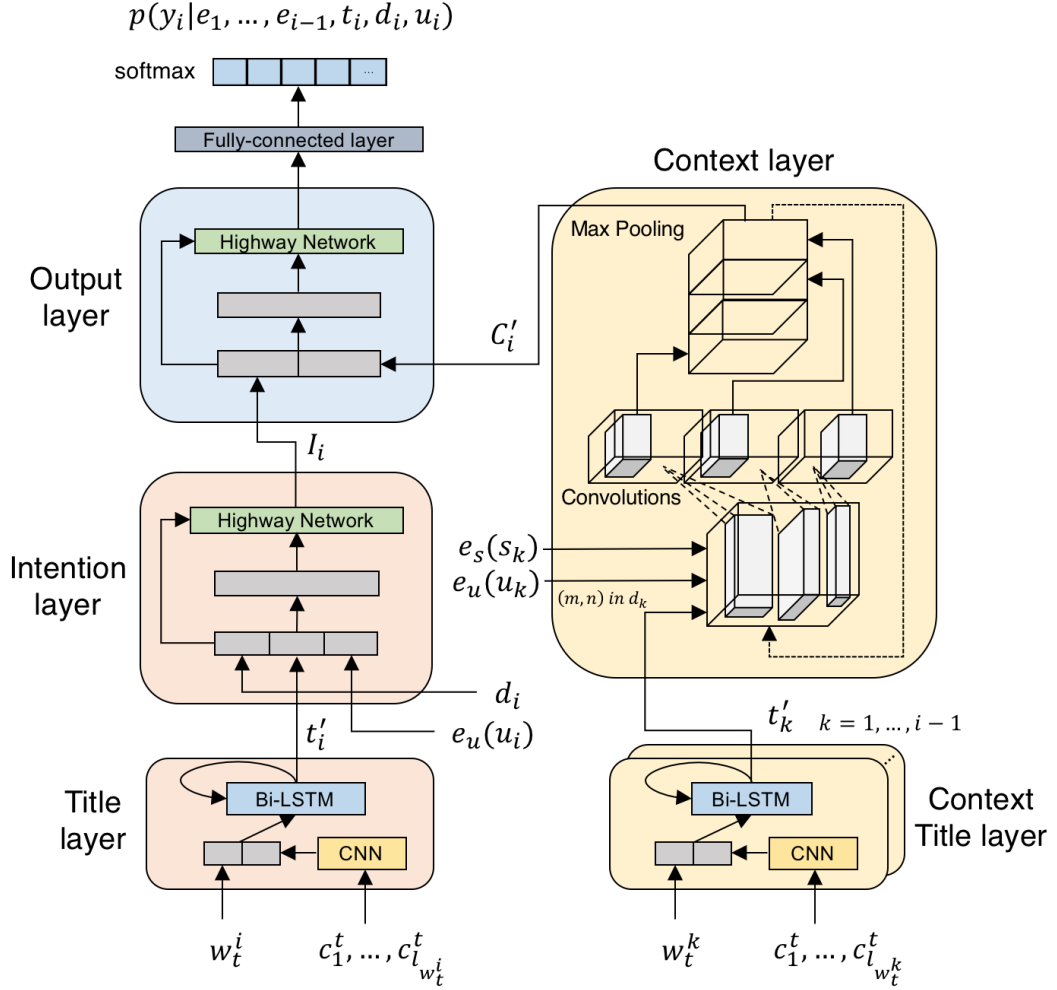


Figure 3.1: NESA overview. Given the title, duration, user attributes, and pre-registered events, NESA suggests suitable time slots for events.

in Figure 3.1.

3.3.1 Title layer

RNNs have become one of the most common approaches for representing the meaning of written text [92]. Among the various RNN models, BiDAF for question answering [95] and named entity recognition [91] often use not only word-level representations but also

character-level representations as inputs. While word-level representations effectively convey semantic/syntactic relationships between words [1], character-level representations are widely used to represent unknown or infrequent words [94]. In event scheduling tasks, it is essential to use character-level representations for understanding personal calendars that have numerous pronouns or abbreviations.

Following previous works on question answering, we represent each title t_i using BiLSTM [50, 51] with pre-trained word embeddings such as GloVe [2]. Given a title t_i comprised of T_i words, we map the words into a set of word embeddings $w_1^i, \dots, w_{T_i}^i$. The Title layer computes hidden state h_{T_i} of the LSTM as follows:

$$h_t = \text{LSTM}(w_t^i, h_{t-1}) \quad (3.3)$$

where h_t is the t -th hidden state of the LSTM which is calculated as follows:

$$\begin{aligned} i_t &= \sigma(W_{i1}w_t + W_{i2}h_{t-1} + b_i) \\ f_t &= \sigma(W_{f1}w_t + W_{f2}h_{t-1} + b_f) \\ g_t &= \tanh(W_{g1}w_t + W_{g2}h_{t-1} + b_g) \\ o_t &= \sigma(W_{o1}w_t + W_{o2}h_{t-1} + b_o) \\ c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\ h_t &= o_t \odot \tanh(c_t) \end{aligned} \quad (3.4)$$

where we have omitted i from w_t^i for clarity and \odot denotes element-wise multiplication. W_* and b_* are trainable parameters of the LSTM. The LSTM is effective in representing long-term dependencies between distant inputs using input gate i and forget gate f .

The Title layer uses BiLSTM for title representations. With the forward LSTM giving the final hidden state $h_{T_i}^f$, we build the backward LSTM which computes its hidden states with reversed inputs. The backward LSTM's last hidden state denoted as h_1^b is concatenated with $h_{T_i}^f$ to form the title representation. The title representation will be denoted as $t'_i = [h_{T_i}^f, h_1^b] \in \mathbb{R}^T$.

On the other hand, the characters of each word with length l_{w_t} can be represented as a set of character embeddings $c_1^t, \dots, c_{l_{w_t}}^t$. A common way to combine character embeddings into a word character representation is to use convolutions as follows:

$$f_k^c = \tanh(< C_{k:k+m-1}^t, F > + b) \quad (3.5)$$

where f_k^c is k -th element of a feature map f^c , $C_{k:k+m-1}^t$ is a concatenation of character embeddings from c_k^t to c_{k+m-1}^t , m is a convolution width, F is a filter matrix, and $< \cdot, \cdot >$ denotes the Frobenius inner product. Using max-over-time pooling [97], the single scalar feature is extracted as $f^c = \max_k f_k^c$. Given N types of filters and each of them having a different number of filters, resulting word character representations are obtained as $w_t^{c,i} = [f^{c,1}, \dots, f^{c,N}]$ where $[\cdot, \cdot]$ denotes a vector concatenation, and $f^{c,n}$ is a concatenation of the outputs of n -th filters. We concatenate word representation w_t^i with word character representation $w_t^{c,i}$, which is inputted into the LSTM in Equation 3.3.

3.3.2 Intention layer

Users have different intentions when registering a specific event. For instance, event titles that contain only personal names connote *meetings* to someone, but could mean *appointments* to others. To capture the intention of each user, we incorporate the title t_i , duration d_i , and user u_i attributes in the Intention layer. In this way, the Intention layer takes into account user preferences and purposes of events. In particular, we use the Highway Network that has some skip-connections between layers [96].¹ Given a title representation t'_i from a Title layer, duration d_i , and user u_i , the output of the Highway Network I_i is as follows:

$$x = [t'_i, d_i, e_u(u_i)] \quad (3.6)$$

$$q = \sigma(W_q x + b_q) \quad (3.7)$$

$$I_i = q \odot f(W_h x + b_h) + (1 - q) \odot x \quad (3.8)$$

¹While we could use Multi-Layer Perceptron (MLP) instead, the Highway Network achieved better performance in our preliminary experiments.

where $e_u(\cdot) \in \mathbb{R}^U$ is an embedding mapping for each user. $W_q, W_h \in \mathbb{R}^{(T+U+1) \times (T+U+1)}$ are trainable parameters and f is a nonlinearity. Due to the skip-connection from x to I_i in addition to the nonlinearity, the Intention layer easily learns both linear and nonlinear relationships between calendar attributes.

3.3.3 Context layer

We define a calendar context as a set of events that are pre-registered before the event e_i . We denote each pre-registered event as e_k where k is from 1 to $i - 1$. Note that each user's week has a varying number of events from 0 to more than 50. Also, each pre-registered event e_k is comprised of different kinds of attributes such as start time, title, and duration. In the Context layer, we represent the calendar context by reflecting the status of the current week and scheduling preferences of users. Then, we use CNNs to capture the local and global features of the calendar context to understand the calendar context representation.

Context title representation

For each title t_k in a pre-registered event e_k , we build a Context Title layer that processes only the titles of pre-registered events. Using BiLSTM and character-level CNN, each context title representation is obtained as t'_k . Note that multiple context title representations are obtained simultaneously in a mini-batch manner.

Calendar context representation

Given the context title representations $t'_k \in \mathbb{R}^T$, we construct a calendar context $C_i \in \mathbb{R}^{(M \times N) \times (T+U+S)}$ where U and S are dimensions of user and slot embeddings, respectively. M represents the number of days in a week, and N represents the number of hours in a day. Each depth is denoted as $C_i^{m,n} \in \mathbb{R}^{T+U+S}$ which is from m -th row (day) and n -th column (hour) of C_i . Each $C_i^{m,n}$ is constructed as follows:

$$C_i^{m,n} = [t'_{(m,n)}, e_u(u_i), e_s(s_{(m,n)})] \quad (3.9)$$

$$t'_{(m,n)} = \left\{ \begin{array}{ll} t'_k & \text{if } (m,n) \text{ lies in } e_k \text{'s duration } d_k \\ \mathbf{0} \in \mathbb{R}^T & \text{otherwise} \end{array} \right\} \quad (3.10)$$

where $e_u(\cdot) \in \mathbb{R}^U$ and $e_s(\cdot) \in \mathbb{R}^S$ are user and slot embedding functions, respectively, and $s_{(m,n)}$ is a slot representation on m -th day at n -th hour.

Similar to the work of Szegedy et al. [98], given the calendar context C_i , the first convolution layer convolves C_i with 100 (1×1), 200 (3×3), 300 (5×5) filters, followed by batch normalization [99] and element-wise rectifier nonlinearity. We pad the calendar context to obtain same size outputs for each filter, and concatenate each output depth-wise. The second convolution layer consists of 50 (1×1), 100 (3×3), 150 (5×5) filters, followed by batch normalization and max-over-time pooling. As a result, we obtain the final calendar context representation $C'_i \in \mathbb{R}^{300}$.

3.3.4 Output layer

Given a calendar context representation C'_i and an intention representation I_i , the Output layer computes the probability of each time slot in $M \times N$. We again adopt a Highway Network to incorporate the calendar context representation and the intention representation. Similar to Equation 3.6-3.8, given the input $x_o = [C'_i, I_i]$, the probability distribution of time slots is as follows:

$$z = q_o \odot f(W_h x_o + b_h) + (1 - q_o) \odot x_o \quad (3.11)$$

$$p_j = \text{softmax}(W_o z + b_o)_j \quad (3.12)$$

$$\text{softmax}(\alpha)_j = \frac{\exp(\alpha_j)}{\sum_{j'} \exp(\alpha_{j'})} \quad (3.13)$$

where q_o is obtained in the same way as Equation 3.7 and j is from 1 to $M \times N$. We have used a single fully-connected layer for predicting the start time slot y_i of the event

e_i . Given the outputs, the cross-entropy loss $CE(\Theta)$ of NESAs is computed as follows:

$$CE(\Theta) = -\frac{1}{K} \sum_{i=1}^K \log p(y_i | e_1, \dots, e_{i-1}, t_i, d_i, u_i; \Theta) \quad (3.14)$$

where K denotes the number of events in a week. The model is optimized on the weeks in the training set. We use the Adam optimizer [100] to minimize the loss $CE(\Theta)$.

3.4 Experiment

3.4.1 Dataset

Preprocessing

We used Google Calendar² data collected between April 2015 and March 2018 by Konolabs, Inc. The format of the data is based on iCalendar, which is the most commonly used online calendar format. We detected and removed noisy events from the raw calendar data to reflect only real online calendar events. Events that we considered as noise are as follows:

- Events automatically generated by other applications (e.g., phone call logs, weather information, and body weight).
- Having an event title that has no meaning (e.g., empty string).
- All-day events, i.e., the events that will be held all day long.

Although some of the all-day events are legitimate events such as vacations or long-term projects, most of them are regular events whose start times have been simply omitted by users. We represented time slots as integers ranging from 0 to 167 where each time slot was considered as one hour in a week (i.e., 7 days \times 24 hours). The duration of each event is scaled to a scalar value from 0 to 1.

²<https://www.google.com/calendar>

Table 3.1: Event Scheduling Dataset Statistics

Statistics	Personal	Multi-Attendee
# of users	859	260
# of unseen users ³	–	217
# of events	593,207	1,354
# of weeks	109,843	1,045
Avg. # of pre-registered events	6.9	22.2
Avg. # of attendees	–	2.1

In Table 3.1, the second column shows the statistics of the personal event scheduling dataset after filtering. Though we carefully filtered calendar events, the dataset still had a considerable number of unrecognizable events (e.g., personal abbreviations). However, to test the ability of our proposed model, we did not perform any further filtering. The third column shows the statistics of the multi-attendee event scheduling dataset. Each event in the multi-attendee event scheduling dataset has at least two attendees, and attendees in each event are in the same time zone.⁴ Due to the small number of multi-attendee events, we use them only as a test set for multi-attendee event scheduling. Also, we ensure that no events in the multi-attendee event scheduling dataset appear in the personal event scheduling dataset. As the multi-attendee event scheduling dataset has multiple attending users per event, it has more pre-registered events than the personal event scheduling dataset. Note that both the personal and multi-attendee event scheduling datasets have a much larger number of users than the CAP dataset⁵ which has events of only 2 users [46, 89]. We split the dataset into training (80%), validation (10%), and test (10%) sets, respectively.

³The number of users not seen in the personal event scheduling dataset.

⁴This can be easily extended to different time zone situations by shifting one of the time offsets.

⁵The CAP dataset contains system logs of Calendar Apprentice, which are difficult to convert to the iCalendar format.

Evaluation metrics

We used various metrics to evaluate the performance of each model in event scheduling. Recall@N is the metric that determines if the correct time slot is in the top n predictions. Recall@1 and Recall@5 were mainly used. We also used Mean Reciprocal Rank (MRR) which is the mean of the inverse of the correct answer’s ranks of results. Also, motivated by the fact that suggesting time slots close to the correct answers counts as proper event scheduling, we used Inverse Euclidean distance (IEuc) [101] which calculates the inverse distance between predicted slots \hat{y}_i and answer slots y_i in two-dimensional space in terms of days (m) and hours (n) as follows:

$$Euc(\hat{y}_i, y_i) = \sqrt{(\hat{y}_{i_m} - y_{i_m})^2 + (\hat{y}_{i_n} - y_{i_n})^2} \quad (3.15)$$

$$IEuc(\hat{y}_i, y_i) = \frac{1}{Euc(\hat{y}_i, y_i) + 1}. \quad (3.16)$$

3.4.2 Experimental settings

Baseline models

While recent automatic scheduling systems have proven to be effective on small sized datasets [49, 89, 47], it is difficult to directly apply their methodologies to our tasks for the following reasons: 1) some of them assume that user preferences are already given [49], 2) some use learning mechanisms based on systematic interactions with users [89], or 3) require human labor [47]. As a result, we use baseline models that are easily reproducible but still effective in our tasks.

In our study, the baselines are as follows: 1) a variant of CAP [46] using Random Forest (RF), 2) Support Vector Machine (SVM) [42, 53], 3) Logistic Regression (LogReg), and 4) Multi-Layer Perceptron (MLP). While RF and SVM are representative of previously suggested scheduling models, we further evaluate LogReg and MLP which are frequently adopted as classification baseline models.

As previous studies have focused on building interactive scheduling software, their learning algorithms rely largely on system dependent features such as event types, position

of attendees, names of college classes, and so on [46]. As the iCalendar format does not contain most of these system dependent features, we used the attributes in Section 3.2.1 as inputs to the four baseline models. Besides categorical or real-valued features, event titles are represented as the average of pre-trained word embeddings, and calendar contexts are given as binary vectors in which filled time slots are indicated as 1. For user representations, we used the normalized event start time statistics of each user (i.e., 168 dimensional vector whose elements sum to 1) to reflect the scheduling preferences of each user. The representation of an unseen user is obtained using the average start time statistics of all the users in the training set.⁶ The biggest difference between the baseline models and NESAs is that the baseline models use a fixed set of hand-crafted features, whereas NESAs learn to represent user preferences and calendar contexts for effective event scheduling.

Model settings

While CAP uses a single decision tree for event scheduling, we constructed RF using thousand decision trees to build a more effective baseline model. The SVM model uses squared hinge losses and the one-vs-rest strategy for training. For LogReg, we used the SAGA optimizer [102]. Rectified linear unit (ReLU) [103] was used for MLP’s activation function. Also for MLP, early stopping was applied based on the loss on the validation set, and we used the Adam optimizer for MLP. Both LogReg and MLP used L_2 regularizations to avoid overfitting.

The hyperparameters of MLP and NESAs were chosen based on the MRR scores on the validation sets and the results are shown in Table 3.2. We used the same hyperparameters from [94] for character-level convolutions. A dropout of 0.5 was applied to the non-recurrent part of the RNNs of NESAs to prevent overfitting [104]. We also clipped gradients when their norm exceeded 5 to avoid exploding gradients. Besides the character embedding, there are three additional embeddings in NESAs: 1) word, 2) user, and

⁶Each baseline feature representation was selected among various hand-crafted features based on our in-house experiments. For instance, statistics based user representation was better than one-hot user representation in terms of both event scheduling performance and generalization.

Table 3.2: Hyperparameters of MLP and NESA

Model	Parameter	Value
MLP	Hidden layer size	500
	# of hidden layers	2
	Learning rate	0.0005
NESA	LSTM cell hidden size	100
	# of LSTM layers	2
	LSTM dropout	0.5
	Day M , hour N	7, 24
	T, C, S, U	200, 30, 30, 30
	Learning rate	0.001

3) slot. We used pre-trained GloVe⁷ for word embeddings, and randomly initialized embeddings for character, user, and slot embeddings. Word embeddings were fixed during optimization while other embeddings were optimized during training.

For training NESA, we used PyTorch⁸ with a CUDA enabled NVIDIA TITAN Xp GPU. The baseline models were trained using Scikit-learn⁹. It took 8 hours of training for NESA to converge, which is quite short given the size of our training set and the complexity of NESA. NESA performs event scheduling as fast as baseline models by using mini-batches. We also experimented with increased number of layers and hidden dimensions in the MLP model so that it would have the same number of parameters as NESA (8.5M). However, the performance of the MLP model was lower than that of the MLP model trained on the best hyperparameters (7.0% in terms of MRR).

⁷For both NESA and baseline features, we used glove.840B.300d word embeddings.

⁸<https://pytorch.org>

⁹<https://scikit-learn.org>

Table 3.3: Personal Event Scheduling Results

Model	Recall@1	Recall@5	MRR	IEuc
RF [46]	0.0348	0.1483	0.0988	0.2520
SVM [42, 53]	0.0445	0.1762	0.1271	0.2619
LogReg	0.0442	0.1749	0.1279	0.2678
MLP	0.0442	0.1803	0.1277	0.2725
NESA	0.0604	0.2156	0.1542	0.2881

3.4.3 Quantitative analysis

Personal event scheduling

The scores of personal event scheduling are presented in Table 3.3. The reported scores are average test set scores after ten separate trainings. The best scores are in bold. We first see that the performance ranking of the IEuc scores is similar to that of other metric scores such as the Recall@5 scores. This shows that the more a model accurately predicts an answer, the more it suggests nearby time slots around the correct answer. Among the baseline models, MLP performed the best on average, and RF achieved the lowest overall scores. However, despite MLP’s deeper structure, performance improvements of MLP over LogReg were marginal, which shows the limitation of feature based models. NESA achieved higher scores than the baseline models in all metrics by learning to schedule directly using raw calendar data. NESA outperformed the baseline models by 29.6% on average in terms of MRR. More specifically, NESA outperformed MLP, which is the best baseline model, by 36.5%, 19.6%, 20.7%, and 5.7% in terms of Recall@1, Recall@5, MRR, and IEuc, respectively.

Multi-attendee event scheduling

The performance results of the models on multi-attendee event scheduling are presented in Table 3.4. The scores of each model are obtained by Equation 3.2. Compared to the performances on personal event scheduling, Recall@1 and Recall@5 of RF have been

Table 3.4: Multiple Attendee Event Scheduling Results

Model	Recall@1	Recall@5	MRR	IEuc
RF [46]	0.0635	0.2585	0.0742	0.2389
SVM [42, 53]	0.0030	0.0340	0.0234	0.2530
LogReg	0.0037	0.0332	0.0260	0.2608
MLP	0.0406	0.1928	0.0773	0.2507
NESA	0.0960	0.2740	0.1744	0.2950

greatly improved, but MRR and IEuc of RF have been degraded. This verifies the limited effectiveness of decision tree based models as reported in the work of Mitchell et al. [46]. RF fails to provide precise probability distribution over time intervals, that reflects user preferences and calendar contexts, as MRR and IEuc are more sensitive to suggestion quality over the whole week. Other baseline models such as SVM, LogReg, and MLP have failed to produce meaningful results on multi-attendee event scheduling. We found that the huge performance degradation of these models comes from generalization failure on unseen users as most users (217 out of 260 as shown in Table 3.1) in the multi-attendee event scheduling dataset are unseen during training on the personal event scheduling dataset. The performance of SVM, LogReg, and MLP on multi-attendee event scheduling was higher (but still insufficient compared to RF and NESA) when all the attendees were comprised of seen users during training.

NESA does not suffer from the unseen user problem by understanding raw online calendars to infer user preferences and understand calendar contexts. While preferences of known users can be encoded in user embeddings in NESA, preferences of unseen users can be inferred from their raw calendars. As with the personal event scheduling task, NESA outperforms the other baseline models by large margins on the multi-attendee event scheduling task. Specifically, NESA outperforms the best baseline model RF by 51.2%, 6.0%, 135.0%, and 23.5% in terms of Recall@1, Recall@5, MRR, and IEuc, respectively. This shows that using raw calendar data for understanding user preferences and calendar contexts is very important in event scheduling tasks.

Table 3.5: NESAs Model Ablation
(Diff. %: average performance difference % of 4 metrics)

Model	Recall@1	Recall@5	MRR	IEuc	Diff. %
NESA	0.0642	0.2241	0.1593	0.2899	–
- Title L. & Context Title L.	<u>0.0296</u>	<u>0.1239</u>	<u>0.0985</u>	<u>0.2520</u>	<u>-37.5</u>
- Intention L.	0.0433	0.1613	0.1218	0.2571	-23.9
- Context L.	0.0460	0.1749	0.1292	0.2806	-18.1
- Character-CNN	0.0499	0.1831	0.1345	0.2688	-15.9
- Word E.	0.0576	0.2022	0.1465	0.2767	-8.2
- Slot E.	0.0548	0.2052	0.1468	0.2847	-8.2
- Duration F.	0.0583	0.2099	0.1502	0.2818	-6.0
- User E.	0.0595	0.2121	0.1522	0.2831	-4.9

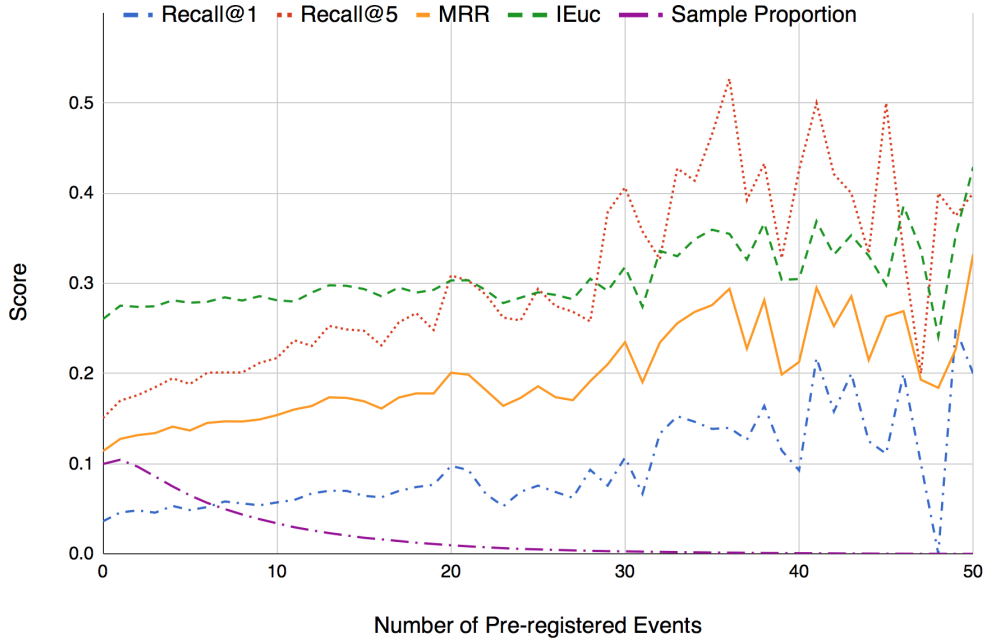


Figure 3.2: Performance changes with different numbers of pre-registered events in NESAs.

NESA model ablation and analysis

To analyze the architecture of NESAs, we removed each layer or component of NESAs. The results are shown in Table 3.5. When the Context layer is removed, the Output layer

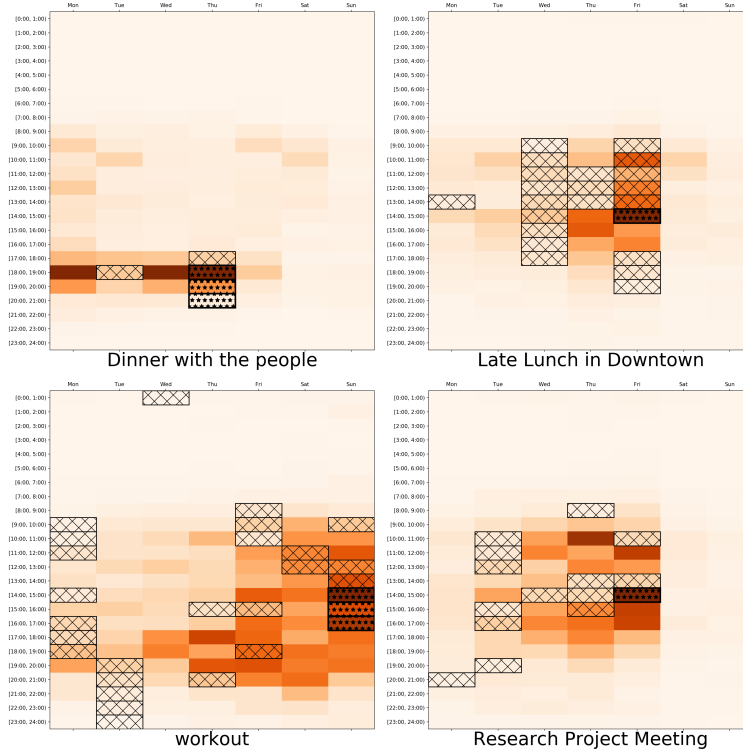


Figure 3.3: Output probabilities of NESA given different titles.

receives only the intention representation. We feed the title representation instead of the intention representation to the Output layer when the Intention layer is removed. The Title layer & Context Title layer have the most significant impact on the overall performance. The Intention layer also shows that incorporating user and duration attributes with title attributes is crucial for event scheduling. Following the Intention layer and the Context layer, the character-CNN also has substantial effects on the performance.

To demonstrate the Context layer’s impact, we illustrate the changes in performance of NESA based on different numbers of pre-registered events in Figure 3.2. As the number of pre-registered events grows, overall performance improves. Note that the sampling proportion decreases as the number of pre-registered events increases, which causes a high variance in performance.

Table 3.6: Baseline Model Ablation

Model	Recall@1	Recall@5	MRR	IEuc	Diff. %
MLP	0.0445	0.1805	0.1283	0.2719	–
- Context F.	<u>0.0384</u>	<u>0.1624</u>	<u>0.1026</u>	<u>0.2582</u>	<u>-12.2</u>
- Character F.	0.0433	0.1788	0.1271	0.2724	-1.1
- Word F.	0.0425	0.1710	0.1245	0.2661	-3.7
- Duration F.	0.0433	0.1760	0.1256	0.2704	-2.0
- User F.	0.0440	0.1790	0.1269	0.2722	-0.7

Baseline model ablation

Although the performance of the baseline models is lower than that of NESAs, models such as MLP still achieve reasonable performance. We present the ablated MLP model in Table 3.6 and compare all its features to determine which feature contributes the most to the performance. We removed each feature one by one, and retrained the MLP model. We found that the MLP model, like NESAs, largely depends on the context feature. It seems that MLP tends to choose empty slots based on the context features.

3.4.4 Qualitative analysis

Effect of the Title Layer

Given different titles, NESAs assign different probabilities to each slot. In Figure 3.3, we visualized the output probabilities of NESAs given four different input titles. The rows of each heatmap denote the hours in a day, and the columns denote the days in a week. The filled time slots are marked with lattices and the answers are marked with stars. For the title “Dinner with the people,” NESAs suggest mostly night times. Also, for the title “Late Lunch in Downtown,” NESAs not only suggest late lunch hours, but it also chooses days that the user may be in downtown. *workout* and *Meeting* are more ambiguous keywords than *Lunch* or *Dinner*, but NESAs suggest again suitable time slots based on each title. Figure 3.3 shows *workout* is done on weekends or at evening-time

Table 3.7: Nearest Neighbors (NNs) of Title Representations Given the Title **Family lunch**

MLP NNs	NESA NNs
Family Dinner out	Birthday <u>lunch</u>
Family Dinner	Themed <u>Lunch</u>
<u>Lunch</u> with Family Friends	UNK / BDP <u>lunch</u>
family dinner	Hope <u>lunch</u>

Table 3.8: Nearest Neighbors of Title/Intention Representations Given the Title **App project work** (duration 120 min.)

Title layer	Intention layer		
	User A, 120 min.	User B, 120 min.	User A, 240 min.
App project work (120 min.)	Make V1 of <u>app</u> (120 min.)	Create paperwork for meetings (60 min.)	Meet Databases Team (<u>240 min.</u>)
App work (540 min.)	Do <u>Databases</u> project (120 min.)	<u>Try</u> Fontana again (60 min.)	App work (<u>540 min.</u>)
App Description to Richard (60 min.)	<u>Databases</u> (120 min.)	<u>Try</u> Peter @ UNK again (60 min.)	Watch databases, do algorithmics (<u>240 min.</u>)
App w Goodman (60 min.)	UNK and spot market (120 min.)	<u>Try</u> pepper Jaden Mark (60 min.)	Databases Final Meeting (<u>180 min.</u>)

while *Meeting* is held during office hours.

In Table 3.7, we show the 4 nearest neighbors of title representations of MLP and NESA. The distances between each representation were calculated using the cosine similarity. MLP’s title representation is the element-wise average of word embeddings, and NESA uses the Title layer for title representations. With the title “Family lunch,” we observe that MLP’s title representations do not differentiate each keyword in event scheduling. Although the keyword *lunch* should have more effect on event scheduling, most nearest neighbors of MLP’s title representation are biased towards the keyword *Family*, while nearest neighbors of NESA’s title representation are mostly related to *lunch*.

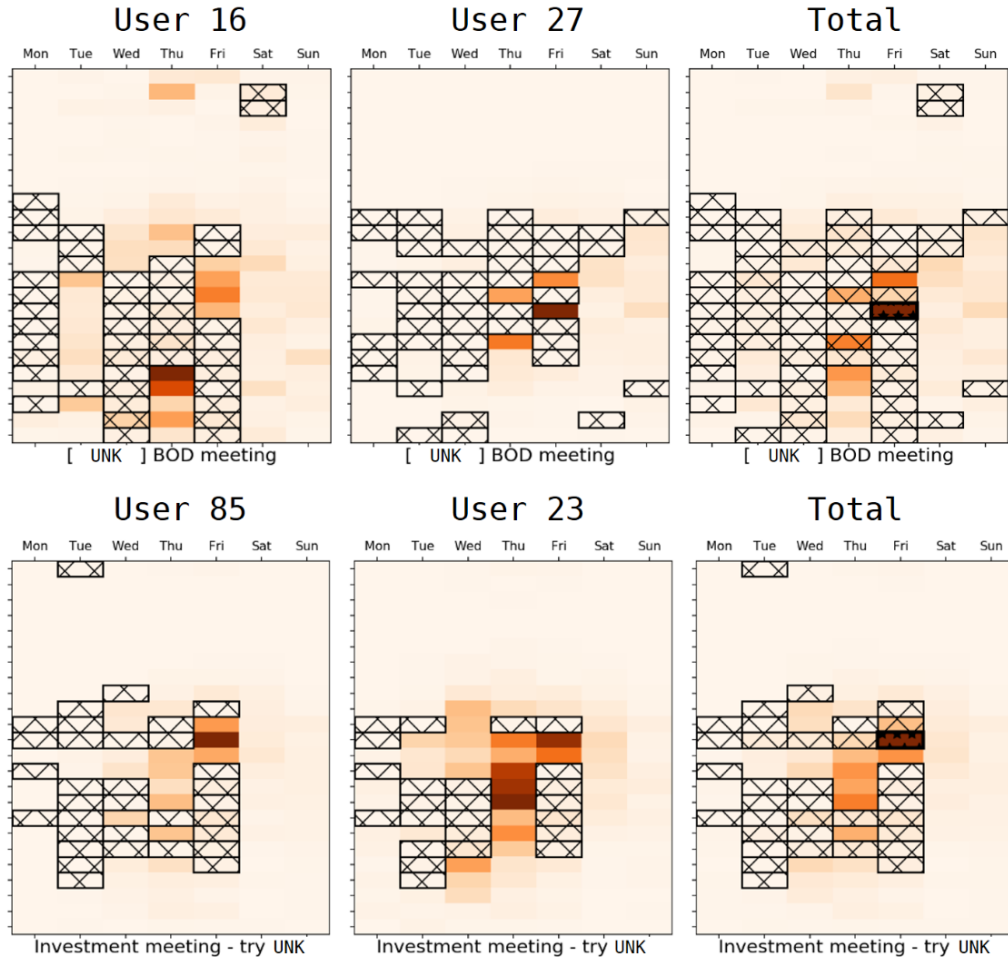


Figure 3.4: Output probabilities of NESAs in multi-attendee meeting scheduling.

Effect of the Intention layer

The Intention layer in NESAs combines different types of attributes from calendar data. In Table 3.8, we present the 4 nearest neighbors of the title and intention representations based on the cosine similarities. Given the title “App project work,” the Title layer simply captures semantic representations of the title. Titles with similar meanings such as “App work” are its nearest neighbors (1st column). On the other hand, the nearest neighbors of the intention representation are related to not only the keyword *app* but also the keyword *database*, which is one of user A’s frequent terms (2nd column). We observe that the

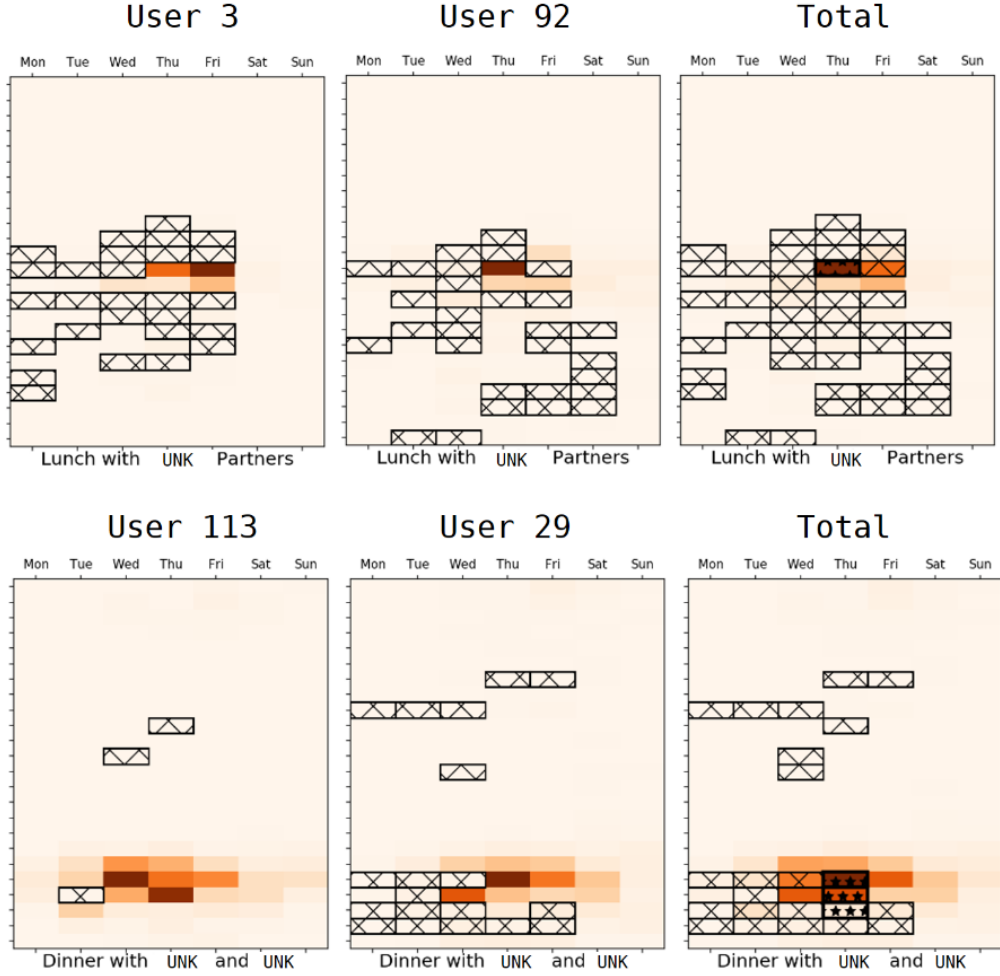


Figure 3.5: Output probabilities of NESA in multi-attendee event scheduling given lunch and dinner events.

intention representation changes by replacing user A with user B who frequently uses the term *Try* (3rd column). The duration attribute is also well represented as events with longer durations are closer to user A’s 240 minute long event (4th column).

Multi-attendee event scheduling analysis

In Figure 3.4–3.6, we present examples of multi-attendee event scheduling. Using NESA, we obtain each user’s preferred time slots, and the suggested time slots for multi-

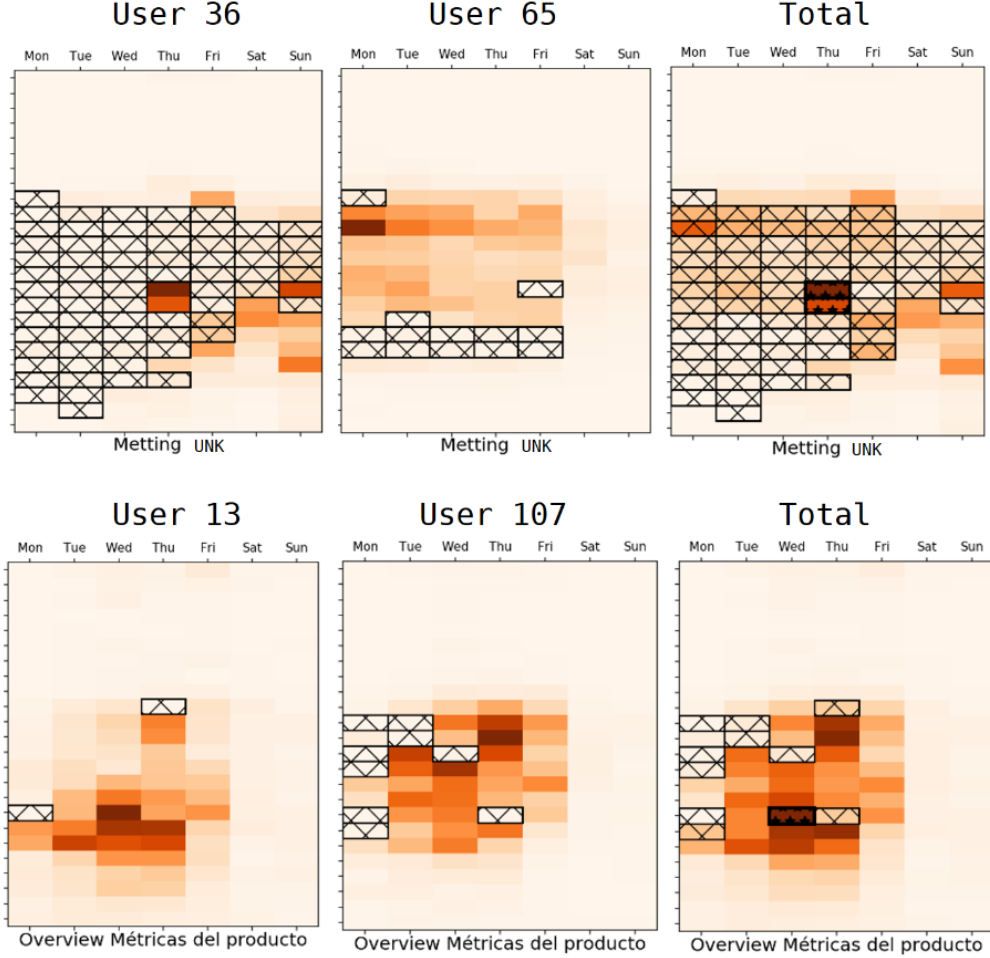


Figure 3.6: Output probabilities of NESAs in multi-attendee event scheduling given misspelled and non-English events.

attendee events are calculated by Equation 3.2. Again, the filled time slots are marked with lattices and the answers are marked with stars. We show a multi-attendee event in each row, and each row contains the preferences of two different users and their summed preference (total). We anonymized any pronouns as *UNK* tokens for privacy issues.

Figure 3.4 shows examples of event scheduling for meetings. The two examples clearly show that NESAs understands each user’s calendar context, and suggests time intervals mostly during office hours. Figure 3.5 shows appointments such as lunch and dinner rather

than meetings. While each example accurately represents the purpose of each event, note that NESA does not suggest weekends for “Lunch with UNK Partners.” We think that NESA understands the keyword *Partner*, which is frequently related to formal meetings. In Figure 3.6, we show how misspellings (*Metting* for *meeting*) and non-English (“Métricas del producto” means “product’s metric” in Spanish) are understood by NESA. As NESA has the Title layer that leverages the characters of infrequent words, NESA successfully suggests suitable office hours for each event.

Chapter 4

Conclusion

In this paper, we showed that contextualized representation learning is effective for both NER and event scheduling. For biomedical NER, our proposed tool BERN recognizes known entities and discovers new entities using BioBERT NER models. The BioBERT NER models outperform NER models of existing Web-based text mining tools such as ezTag and tmTool in terms of F1 on genes/proteins, diseases, drugs/chemicals, and species. After reviewing a vast number of cases of overlapping entities, we developed and used the decision rules on identifying the entity types of overlapping entities which occur frequently in multi-type NER results. For assigning a specific ID to each recognized entity, multiple normalization models are combined and integrated into BERN. The RESTful Web service of BERN is freely available and can be used for various types of input. Researchers can use BERN for text mining tasks such as NER including new named entity discovery, information retrieval, question answering, and relation extraction. We plan to use a multi-task NER model for higher NER performance. Also, we will develop a novel entity type decision model that uses transfer learning to consider not only the entity types and probabilities of overlapping entities but also the deeper contextual meaning of a text. Furthermore, we can improve the performance of NER models used by BERN by applying the methods of ALBERT to pre-training objectives of their language model.

For calendar event scheduling, we proposed a novel way to fully make use of raw online

calendar data. Our proposed model NESAs learns how to perform event scheduling directly from raw calendar data, and to consider user preferences and calendar contexts. We also showed that deep neural networks are highly effective in scheduling events. Unlike previous works, we leveraged a large-scale online calendar dataset in the Internet standard format, which makes our approach more applicable to other systems. NESAs achieves the best performance among the existing baseline models on both personal and multi-attendee event scheduling tasks. We plan to study the relationships between users for multi-attendee event scheduling after obtaining a sufficient number of multi-attendee calendar events. And, if a duration of an event is several time slots long, we can broaden the correct answer. Also, we can use BERT with SentencePiece embeddings instead of GloVe to handle rare or out-of-vocabulary words and to obtain contextualized subword unit representations of a given vocabulary size.

Bibliography

- [1] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *Advances in Neural Information Processing Systems*, pp. 3111–3119, 2013.
- [2] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” in *EMNLP*, vol. 14, pp. 1532–43, 2014.
- [3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “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)*, (Minneapolis, Minnesota), pp. 4171–4186, Association for Computational Linguistics, June 2019.
- [4] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, “Deep contextualized word representations,” in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, (New Orleans, Louisiana), pp. 2227–2237, Association for Computational Linguistics, June 2018.
- [5] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, “Albert: A lite bert for self-supervised learning of language representations,” *arXiv preprint arXiv:1909.11942*, 2019.

- [6] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, “GLUE: A multi-task benchmark and analysis platform for natural language understanding,” in *International Conference on Learning Representations*, 2019.
- [7] E. F. Tjong Kim Sang and F. De Meulder, “Introduction to the conll-2003 shared task: Language-independent named entity recognition,” in *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL ’03, (Stroudsburg, PA, USA), pp. 142–147, Association for Computational Linguistics, 2003.
- [8] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, “SQuAD: 100,000+ questions for machine comprehension of text,” in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, (Austin, Texas), pp. 2383–2392, Association for Computational Linguistics, Nov. 2016.
- [9] P. Rajpurkar, R. Jia, and P. Liang, “Know what you don’t know: Unanswerable questions for SQuAD,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, (Melbourne, Australia), pp. 784–789, Association for Computational Linguistics, July 2018.
- [10] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang, “BioBERT: a pre-trained biomedical language representation model for biomedical text mining,” *Bioinformatics*, 09 2019. btz682.
- [11] A. Allot, Y. Peng, C.-H. Wei, K. Lee, L. Phan, and Z. Lu, “Litvar: a semantic search engine for linking genomic variant data in pubmed and pmc,” *Nucleic acids research*, vol. 46, no. W1, pp. W530–W536, 2018.
- [12] V. Sharma, N. Kulkarni, S. Pranavi, G. Bayomi, E. Nyberg, and T. Mitamura, “Bioama: Towards an end to end biomedical question answering system,” in *Proceedings of the BioNLP 2018 workshop*, pp. 109–117, 2018.
- [13] B. Percha and R. B. Altman, “A global network of biomedical relationships derived from text,” *Bioinformatics*, vol. 34, no. 15, pp. 2614–2624, 2018.

- [14] C.-H. Wei, R. Leaman, and Z. Lu, “Beyond accuracy: creating interoperable and scalable text-mining web services,” *Bioinformatics*, vol. 32, no. 12, pp. 1907–1910, 2016.
- [15] D. Kwon, S. Kim, C.-H. Wei, R. Leaman, and Z. Lu, “eztag: tagging biomedical concepts via interactive learning,” *Nucleic acids research*, vol. 46, no. W1, pp. W523–W529, 2018.
- [16] J. Garcia-Pelaez, D. Rodriguez, R. Medina-Molina, G. Garcia-Rivas, C. Jerjes-Sánchez, and V. Trevino, “Pubterm: a web tool for organizing, annotating and curating genes, diseases, molecules and other concepts from pubmed records,” *Database*, vol. 2019, 2019.
- [17] R. Leaman, C.-H. Wei, and Z. Lu, “tmchem: a high performance approach for chemical named entity recognition and normalization,” *Journal of cheminformatics*, vol. 7, no. 1, p. S3, 2015.
- [18] R. Leaman, R. Islamaj Doğan, and Z. Lu, “Dnorm: disease name normalization with pairwise learning to rank,” *Bioinformatics*, vol. 29, no. 22, pp. 2909–2917, 2013.
- [19] M. G. Sohrab and M. Miwa, “Deep exhaustive model for nested named entity recognition,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2843–2849, 2018.
- [20] C.-H. Wei, L. Phan, J. Feltz, R. Maiti, T. Hefferon, and Z. Lu, “tmvar 2.0: integrating genomic variant information from literature with dbsnp and clinvar for precision medicine,” *Bioinformatics*, vol. 34, no. 1, pp. 80–87, 2017.
- [21] A. A. Morgan, Z. Lu, X. Wang, A. M. Cohen, J. Fluck, P. Ruch, A. Divoli, K. Fundel, R. Leaman, J. Hakenberg, *et al.*, “Overview of biocreative ii gene normalization,” *Genome biology*, vol. 9, no. 2, p. S3, 2008.

- [22] D. S. Sachan, P. Xie, M. Sachan, and E. P. Xing, “Effective use of bidirectional language modeling for transfer learning in biomedical named entity recognition,” in *Proceedings of Machine Learning Research*, vol. 85, pp. 383–402, 2018.
- [23] X. Wang, Y. Zhang, X. Ren, Y. Zhang, M. Zitnik, J. Shang, C. Langlotz, and J. Han, “Cross-type biomedical named entity recognition with deep multi-task learning,” *Bioinformatics*, vol. 35, no. 10, pp. 1745–1752, 2018.
- [24] W. Yoon, C. H. So, J. Lee, and J. Kang, “Collabonet: collaboration of deep neural networks for biomedical named entity recognition,” *BMC bioinformatics*, vol. 20, no. 10, p. 249, 2019.
- [25] C.-H. Wei, H.-Y. Kao, and Z. Lu, “Gnormplus: an integrative approach for tagging genes, gene families, and protein domains,” *BioMed research international*, vol. 2015, 2015.
- [26] J. M. Giorgi and G. D. Bader, “Transfer learning for biomedical named entity recognition with neural networks,” *Bioinformatics*, vol. 34, no. 23, pp. 4087–4094, 2018.
- [27] M. Habibi, L. Weber, M. Neves, D. L. Wiegandt, and U. Leser, “Deep learning with word embeddings improves biomedical named entity recognition,” *Bioinformatics*, vol. 33, no. 14, pp. i37–i48, 2017.
- [28] R. I. Doğan, R. Leaman, and Z. Lu, “Ncbi disease corpus: a resource for disease name recognition and concept normalization,” *Journal of biomedical informatics*, vol. 47, pp. 1–10, 2014.
- [29] T. H. Dang, H.-Q. Le, T. M. Nguyen, and S. T. Vu, “D3ner: biomedical named entity recognition using crf-bilstm improved with fine-tuned embeddings of various linguistic information,” *Bioinformatics*, vol. 34, no. 20, pp. 3539–3546, 2018.
- [30] Y. Lou, Y. Zhang, T. Qian, F. Li, S. Xiong, and D. Ji, “A transition-based joint

- model for disease named entity recognition and normalization,” *Bioinformatics*, vol. 33, no. 15, pp. 2363–2371, 2017.
- [31] R. Leaman and Z. Lu, “Taggerone: joint named entity recognition and normalization with semi-markov models,” *Bioinformatics*, vol. 32, no. 18, pp. 2839–2846, 2016.
 - [32] M. Krallinger, O. Rabal, F. Leitner, M. Vazquez, D. Salgado, Z. Lu, R. Leaman, Y. Lu, D. Ji, D. M. Lowe, *et al.*, “The chemdner corpus of chemicals and drugs and its annotation principles,” *Journal of cheminformatics*, vol. 7, no. 1, p. S2, 2015.
 - [33] L. Luo, Z. Yang, P. Yang, Y. Zhang, L. Wang, H. Lin, and J. Wang, “An attention-based bilstm-crf approach to document-level chemical named entity recognition,” *Bioinformatics*, vol. 34, no. 8, pp. 1381–1388, 2017.
 - [34] M. Gerner, G. Nenadic, and C. M. Bergman, “Linnaeus: a species name identification system for biomedical literature,” *BMC bioinformatics*, vol. 11, no. 1, p. 85, 2010.
 - [35] C.-H. Wei, H.-Y. Kao, and Z. Lu, “Sr4gn: a species recognition software tool for gene normalization,” *PLoS one*, vol. 7, no. 6, p. e38460, 2012.
 - [36] C.-H. Wei, B. R. Harris, H.-Y. Kao, and Z. Lu, “tmvar: a text mining approach for extracting sequence variants in biomedical literature,” *Bioinformatics*, vol. 29, no. 11, pp. 1433–1439, 2013.
 - [37] J. G. Caporaso, W. A. Baumgartner Jr, D. A. Randolph, K. B. Cohen, and L. Hunter, “Mutationfinder: a high-performance system for extracting point mutation mentions from text,” *Bioinformatics*, vol. 23, no. 14, pp. 1862–1865, 2007.
 - [38] S. Kuruvilla, “An in-depth look at the usage of calendars in the u.s. workplace, particularly the use of advertising calendars,” 2011.
 - [39] D. Montoya, T. Pellissier Tanon, S. Abiteboul, and F. M. Suchanek, “Thymeflow, a personal knowledge base with spatio-temporal data,” in *Proceedings of the 25th*

ACM International on Conference on Information and Knowledge Management, pp. 2477–2480, ACM, 2016.

- [40] P. Berry, M. Gervasio, T. Uribe, K. Myers, and K. Nitz, “A personalized calendar assistant,” in *Working notes of the AAAI Spring Symposium Series*, vol. 76, 2004.
- [41] A. Blum, “Empirical support for winnow and weighted-majority algorithms: Results on a calendar scheduling domain,” *Machine Learning*, vol. 26, no. 1, pp. 5–23, 1997.
- [42] M. T. Gervasio, M. D. Moffitt, M. E. Pollack, J. M. Taylor, and T. E. Uribe, “Active preference learning for personalized calendar scheduling assistance,” in *Proceedings of the 10th international conference on Intelligent user interfaces*, pp. 90–97, ACM, 2005.
- [43] P. Berry, M. Gervasio, B. Peintner, and N. Yorke-Smith, “Balancing the needs of personalization and reasoning in a user-centric scheduling assistant,” tech. rep., Artificial Intelligence Center, SRI International, 2007.
- [44] Y. Sun, N. J. Yuan, Y. Wang, X. Xie, K. McDonald, and R. Zhang, “Contextual intent tracking for personal assistants,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 273–282, ACM, 2016.
- [45] P. Bjellerup, K. J. Cama, M. Desikan, Y. Guo, A. G. Kale, J. C. Lai, N. Lethif, J. Lu, M. Topkara, and S. H. Wissel, “Falcon: Seamless access to meeting data from the inbox and calendar,” in *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, pp. 1951–1952, ACM, 2010.
- [46] T. M. Mitchell, R. Caruana, D. Freitag, J. McDermott, D. Zabowski, *et al.*, “Experience with a learning personal assistant,” *Communications of the ACM*, vol. 37, no. 7, pp. 80–91, 1994.
- [47] J. Cranshaw, E. Elwany, T. Newman, R. Kocielnik, B. Yu, S. Soni, J. Teevan, and A. Monroy-Hernández, “Calendar. help: Designing a workflow-based scheduling

- agent with humans in the loop,” in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pp. 2382–2393, ACM, 2017.
- [48] B. Desruisseaux, “Internet calendaring and scheduling core object specification (icalendar),” *IETF*, 2009.
 - [49] J. Wainer, P. R. Ferreira Jr, and E. R. Constantino, “Scheduling meetings through multi-agent negotiations,” *Decision Support Systems*, vol. 44, no. 1, pp. 285–297, 2007.
 - [50] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
 - [51] M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks,” *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.
 - [52] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.
 - [53] P. Berry, M. Gervasio, B. Peintner, and N. Yorke-Smith, “Ptime: Personalized assistance for calendaring,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 4, p. 40, 2011.
 - [54] S. Pyysalo, F. Ginter, H. Moen, T. Salakoski, and S. Ananiadou, “Distributional semantics resources for biomedical text processing,” in *Proceedings of the 5th International Symposium on Languages in Biology and Medicine*, pp. 39–44, 2013.
 - [55] J. Lafferty, A. McCallum, F. Pereira, *et al.*, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data,” in *Proceedings of the 18th International Conference on Machine Learning*, vol. 1, pp. 282–289, 2001.
 - [56] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

- [57] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy, “Spanbert: Improving pre-training by representing and predicting spans,” *arXiv preprint arXiv:1907.10529*, 2019.
- [58] G. Zhou, “Recognizing names in biomedical texts using mutual information independence model and svm plus sigmoid,” *International Journal of Medical Informatics*, vol. 75, no. 6, pp. 456–467, 2006.
- [59] B. Wang and W. Lu, “Neural segmental hypergraphs for overlapping mention recognition,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, (Brussels, Belgium), pp. 204–214, Association for Computational Linguistics, Oct.-Nov. 2018.
- [60] A. Katiyar and C. Cardie, “Nested named entity recognition revisited,” in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 861–871, 2018.
- [61] N. Greenberg, T. Bansal, P. Verga, and A. McCallum, “Marginal likelihood training of bilstm-crf for biomedical named entity recognition from disjoint label sets,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2824–2829, 2018.
- [62] D. Maglott, J. Ostell, K. D. Pruitt, and T. Tatusova, “Entrez gene: gene-centered information at ncbi,” *Nucleic acids research*, vol. 33, no. suppl_1, pp. D54–D58, 2005.
- [63] S. Sohn, D. C. Comeau, W. Kim, and W. J. Wilbur, “Abbreviation definition identification based on automatic precision estimates,” *BMC bioinformatics*, vol. 9, no. 1, p. 402, 2008.
- [64] S. T. Sherry, M.-H. Ward, M. Kholodov, J. Baker, L. Phan, E. M. Smigielski, and K. Sirotkin, “dbSNP: the NCBI database of genetic variation,” *Nucleic acids research*, vol. 29, no. 1, pp. 308–311, 2001.

- [65] J. D’Souza and V. Ng, “Sieve-based entity linking for the biomedical domain,” in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pp. 297–302, 2015.
- [66] C. E. Lipscomb, “Medical subject headings (mesh),” *Bulletin of the Medical Library Association*, vol. 88, no. 3, p. 265, 2000.
- [67] K. Degtyarenko, P. De Matos, M. Ennis, J. Hastings, M. Zbinden, A. McNaught, R. Alcántara, M. Darsow, M. Guedj, and M. Ashburner, “Chebi: a database and ontology for chemical entities of biological interest,” *Nucleic acids research*, vol. 36, no. suppl_1, pp. D344–D350, 2007.
- [68] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, *et al.*, “Google’s neural machine translation system: Bridging the gap between human and machine translation,” *arXiv preprint arXiv:1609.08144*, 2016.
- [69] E. F. Sang and J. Veenstra, “Representing text chunks,” in *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, pp. 173–179, Association for Computational Linguistics, 1999.
- [70] S. Buchholz and E. Marsi, “Conll-x shared task on multilingual dependency parsing,” in *Proceedings of the tenth conference on computational natural language learning*, pp. 149–164, Association for Computational Linguistics, 2006.
- [71] V. Law, C. Knox, Y. Djoumbou, T. Jewison, A. C. Guo, Y. Liu, A. Maciejewski, D. Arndt, M. Wilson, V. Neveu, *et al.*, “Drugbank 4.0: shedding new light on drug metabolism,” *Nucleic acids research*, vol. 42, no. D1, pp. D1091–D1097, 2013.
- [72] A. Hamosh, A. F. Scott, J. S. Amberger, C. A. Bocchini, and V. A. McKusick, “Online mendelian inheritance in man (omim), a knowledgebase of human genes and genetic disorders,” *Nucleic acids research*, vol. 33, no. suppl_1, pp. D514–D517, 2005.

- [73] K. Donnelly, “Snomed-ct: The advanced terminology and coding system for ehealth,” *Studies in health technology and informatics*, vol. 121, p. 279, 2006.
- [74] Y. Liu, Y. Liang, and D. Wishart, “Polysearch2: a significantly improved text-mining system for discovering associations between human diseases, genes, drugs, metabolites, toxins and more,” *Nucleic acids research*, vol. 43, no. W1, pp. W535–W542, 2015.
- [75] S. Federhen, “The ncbi taxonomy database,” *Nucleic acids research*, vol. 40, no. D1, pp. D136–D143, 2011.
- [76] M. J. Landrum, J. M. Lee, M. Benson, G. Brown, C. Chao, S. Chitipiralla, B. Gu, J. Hart, D. Hoffman, J. Hoover, *et al.*, “Clinvar: public archive of interpretations of clinically relevant variants,” *Nucleic acids research*, vol. 44, no. D1, pp. D862–D868, 2015.
- [77] Z. Lu, H.-Y. Kao, C.-H. Wei, M. Huang, J. Liu, C.-J. Kuo, C.-N. Hsu, R. T.-H. Tsai, H.-J. Dai, N. Okazaki, *et al.*, “The gene normalization task in biocreative iii,” *BMC bioinformatics*, vol. 12, no. 8, p. S2, 2011.
- [78] S. Pradhan, N. Elhadad, B. R. South, D. Martinez, L. M. Christensen, A. Vogel, H. Suominen, W. W. Chapman, and G. K. Savova, “Task 1: Share/clef ehealth evaluation lab 2013.,” in *CLEF (Working Notes)*, pp. 212–231, 2013.
- [79] L. I. Furlong, H. Dach, M. Hofmann-Apitius, and F. Sanz, “Osirisv1. 2: a named entity recognition system for sequence variants of genes in biomedical literature,” *BMC bioinformatics*, vol. 9, no. 1, p. 84, 2008.
- [80] P. E. Thomas, R. Klinger, L. I. Furlong, M. Hofmann-Apitius, and C. M. Friedrich, “Challenges in the association of human single nucleotide polymorphism mentions with unique database identifiers,” *BMC bioinformatics*, vol. 12, no. 4, p. S4, 2011.
- [81] S. Lee, D. Kim, K. Lee, J. Choi, S. Kim, M. Jeon, S. Lim, D. Choi, S. Kim, A.-

- C. Tan, *et al.*, “Best: next-generation biomedical entity search tool for knowledge discovery from biomedical literature,” *PLoS one*, vol. 11, no. 10, p. e0164680, 2016.
- [82] G. Tsatsaronis, M. Schroeder, G. Paliouras, Y. Almirantis, I. Androutsopoulos, E. Gaussier, P. Gallinari, T. Artieres, M. R. Alvers, M. Zschunke, *et al.*, “Bioasq: A challenge on large-scale biomedical semantic indexing and question answering,” in *2012 AAAI Fall Symposium Series*, pp. 92–98, 2012.
- [83] M. Wasim, W. Mahmood, M. N. Asim, and M. U. Khan, “Multi-label question classification for factoid and list type questions in biomedical question answering,” *IEEE Access*, vol. 7, pp. 3882–3896, 2018.
- [84] B. Xu, X. Shi, Z. Zhao, and W. Zheng, “Leveraging biomedical resources in bi-lstm for drug-drug interaction extraction,” *IEEE Access*, vol. 6, pp. 33432–33439, 2018.
- [85] G. Wu, Y. He, and X. Hu, “Entity linking: an issue to extract corresponding entity with knowledge base,” *IEEE Access*, vol. 6, pp. 6220–6231, 2018.
- [86] E. Mynatt and J. Tullio, “Inferring calendar event attendance,” in *Proceedings of the 6th international conference on Intelligent user interfaces*, pp. 121–128, ACM, 2001.
- [87] I. Refanidis and N. Yorke-Smith, “On scheduling events and tasks by an intelligent calendar assistant,” in *Proceedings of the ICAPS Workshop on Constraint Satisfaction Techniques for Planning and Scheduling Problems*, pp. 43–52, 2009.
- [88] L. Garrido and K. Sycara, “Multi-agent meeting scheduling: Preliminary experimental results,” in *Proceedings of the Second International Conference on Multiagent Systems*, pp. 95–102, 1996.
- [89] A. Zunino and M. Campo, “Chronos: A multi-agent system for distributed automatic meeting scheduling,” *Expert Systems with Applications*, vol. 36, no. 3, pp. 7011–7018, 2009.

- [90] C. N. dos Santos and M. Gatti, “Deep convolutional neural networks for sentiment analysis of short texts,” in *COLING*, pp. 69–78, 2014.
- [91] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, “Neural architectures for named entity recognition,” *CoRR*, vol. abs/1603.01360, 2016.
- [92] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur, “Recurrent neural network based language model,” in *Interspeech*, vol. 2, p. 3, 2010.
- [93] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *CoRR*, vol. abs/1409.0473, 2014.
- [94] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush, “Character-aware neural language models,” in *AAAI*, pp. 2741–2749, 2016.
- [95] M. J. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, “Bidirectional attention flow for machine comprehension,” in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, OpenReview.net, 2017.
- [96] R. K. Srivastava, K. Greff, and J. Schmidhuber, “Training very deep networks,” in *Advances in Neural Information Processing Systems (C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, eds.)*, pp. 2377–2385, Curran Associates, Inc., 2015.
- [97] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, “Natural language processing (almost) from scratch,” *Journal of Machine Learning Research*, vol. 12, no. Aug, pp. 2493–2537, 2011.
- [98] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, “Inception-v4, inception-resnet and the impact of residual connections on learning,” in *Thirty-First AAAI Conference on Artificial Intelligence*, pp. 4278–4284, 2017.
- [99] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *Proceedings of the 32nd International Con-*

- ference on International Conference on Machine Learning*, ICML'15, pp. 448–456, JMLR.org, 2015.
- [100] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2015.
 - [101] S. Toby, *Programming Collective Intelligence*. O'Reilly Media, 2007.
 - [102] A. Defazio, F. Bach, and S. Lacoste-Julien, “Saga: A fast incremental gradient method with support for non-strongly convex composite objectives,” in *Advances in Neural Information Processing Systems*, pp. 1646–1654, 2014.
 - [103] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in *Proceedings of the 27th International Conference on Machine Learning*, pp. 807–814, 2010.
 - [104] W. Zaremba, I. Sutskever, and O. Vinyals, “Recurrent neural network regularization,” *arXiv preprint arXiv:1409.2329*, 2014.