Set to Ordered Text: Generating Discharge Instructions from Medical Billing Codes

Litton J Kurisinkel, Nancy F. Chen
Institute for Infocomm Research, A*STAR
{Litton_Kurisinkel, nfychen}@i2r.a-star.edu.sg

Abstract

We present set to ordered text, a natural language generation task applied to automatically generating discharge instructions from admission ICD (International Classification of Diseases) codes. This task differs from other natural language generation tasks in the following ways: (1) The input is a set of identifiable entities (ICD codes) where the relations between individual entities are not explicitly specified. (2) The output text is not a narrative description (e.g. news articles) composed from the input. Rather, inferences are made from the input (ICD codes, which represent diagnoses and clinical procedures) to generate the output (instructions). (3) There is an optimal order in which each sentence (instruction) should appear in the output. Unlike most other tasks, neither the input (ICD codes) nor their corresponding text representations of diagnoses and clinical procedures appear in the output, so the ordering of the output instructions needs to be learned in an unsupervised fashion. We hypothesize that each instruction in the output is mapped to a subset of ICD codes specified in the input. We propose a neural architecture that jointly models (a) subset selection: choosing relevant subsets from a set of input entities; (b) content ordering: learning the order of instructions; (c) text generation: representing the instructions corresponding to the selected subsets in natural language. In addition, we penalize redundancy during beam search to improve tractability for long text generation. We formulate the problem setup and conducted experiments using the MIMIC-III dataset. Our model outperforms baseline models in both BLEU scores and human evaluations.

1 Introduction

1.1 Problem Statement

Many healthcare applications exhibit a strong mapping between numerical or categorical information and clinical instructions. To develop language generation capabilities for these settings, we define a task where discharge instructions are automatically generated using admission ICD (International Classification of Diseases) codes[1] to potentially streamline clinical workflow. We define a task where the input is a set of identifiable items (ICD codes) and the output consists of ordered text sequences (instructions), which are inferred from the input (see Figure 1).

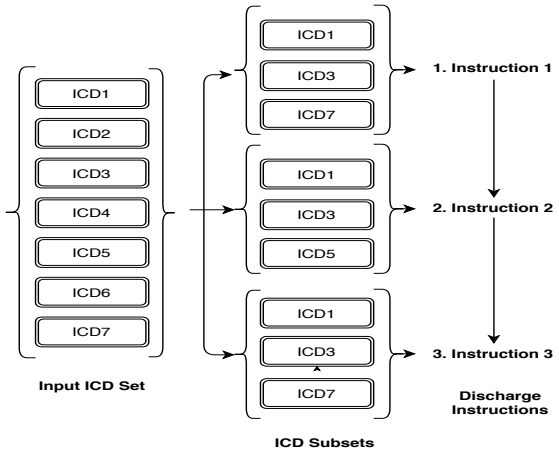

Figure 1: Set to Ordered Text: Problem Definition

1.2 Proposed Approach

We hypothesize that each discharge instruction in the output is mapped to a subset of ICD codes specified in the input. Our proposed approach thus models the correlations between individual entities in the input set to choose the most relevant subsets, and learn to generate their corresponding textual outputs in the appropriate order. We also incorporate explicit means for reducing redundancy during decoding. We empirically verify

[1]It is standard practice for healthcare providers to assign ICD codes to represent information regarding diagnosis and/or procedures for each hospital visit for medical billing purposes.

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 6165–6175,
Hong Kong, China, November 3–7, 2019.

the proposed approach by generating discharge instructions from ICD codes assigned during hospital admissions.

1.3 Relation to Other Work

For most natural language generation tasks, the relations between the input entities are specified in one way or another. For text-to-text generation (e.g. news articles), the relation between entities are semantically encoded in sequences of words (Paulus et al., 2017). For graph-to-text generation, relations between the nodes in a graph are characterized through labeled or unlabeled edges (Liu et al., 2019b). For text generation with database inputs, relations are usually specified through the attributes of each data entry (Lebret et al., 2016). In our problem setup, there is no explicit characterization of the relations between the input ICD codes assigned to a patient's visit.

Most natural language generation problems focus on generating descriptions, which often include the entities specified in the input. Examples include summarization or text generation from database records (Cheng and Lapata, 2016; Jhamtani et al., 2018). In our case, both the content and the ordering of the output need to be inferred from the input data.

Our work is closest to the line of research on text expansion, where the generated text is conditioned on a set of entities (Clark et al., 2018; Zhao et al., 2018; Kiddon et al., 2016). Although in this line of work the relations between the entity input set are not specified, as in our case, these input entities often appear in the output text, making it more straightforward to model the order of the input entities appearing in the output. In our case, neither the input entity set (ICD codes) nor their corresponding text representations (diagnoses and clinical procedures) occur in the generated output (instructions).

2 Approach

2.1 Task Definition

For input set S, the generation task can be specified as follows,

$$x \leftarrow F(S, X)$$
$$o \leftarrow Gen(x),$$

where F represents the mechanism to choose the next subset x given input S and the already chosen sequence of subsets X; Gen generates the output sentence o from the chosen subset x.

2.2 Neural Architecture

The proposed neural architecture is shown in Figure 2. The major components of the network are a lookup table for ICD codes, gates for content and subset selection, one RNN for content ordering and another one for decoding each discharge instruction. The network also posses two attention layers for finding the correlations between input ICD codes and for attending to the chosen content at each stage of instruction generation. The instruction ordering RNN is initialized with a zero vector. Below we elaborate on the major components of the proposed neural architecture in more detail. Bold symbols in equations represent parameter matrices.

2.2.1 Content and Subset Selection

The content selection gate selects the relevant content from each ICD code during each instruction generation phase. The gate takes the ICD code embedding, the previous state of the content ordering RNN H_{t-1} and the correlations among ICD codes into account for selecting the content. The content correlation vector C_j for each ICD code embedding icd_j in the input is computed as follows:

$$\alpha_{j,k} = \exp(icd_j^T \mathbf{W_a} icd_k)$$
$$C_j = \sum_{k \neq j} \alpha_{j,k} icd_k$$

The content gate value computation and the subsequent content selection is conducted as follows:

$$gc_j = \text{sigmoid}(\mathbf{W_{gc}}[icd_j; C_j; H_{t-1}])$$
$$icd_j^{'} = gc_j \odot icd_j$$

The selected content from each of the input ICD code passes through a subset selection gate. Each subset is selected as a probability distribution over the input set of ICD codes as follows:

$$gs_j = \text{sigmoid}(\mathbf{W_{gs}}[icd_j; C_j; H_{t-1}])$$
$$gs_j = \text{softmax}(gs_j)\ \forall j$$

$gs_j, \forall j$ represents the distribution of ICD codes in the subset chosen at the current time step of content ordering RNN. The subset selection gate updates the output from content selection gate as follows:

$$icd_j^{''} = gs_j icd_j^{'} \quad (1)$$

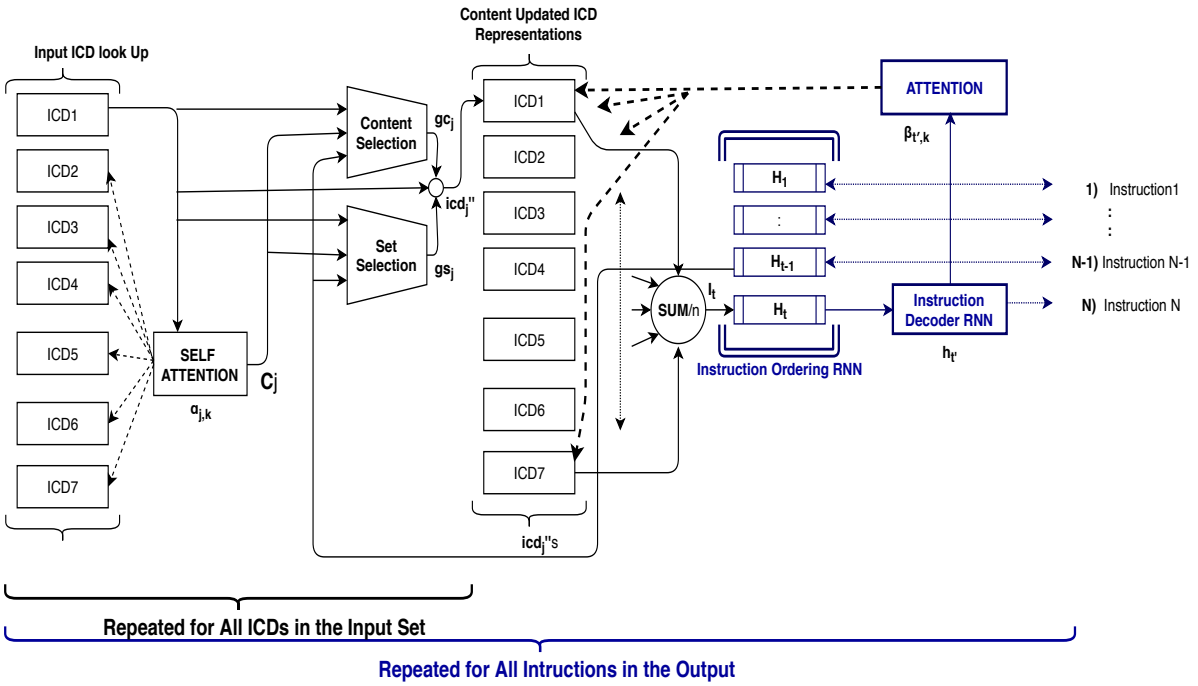

Figure 2: Proposed Neural Architecture

In the Figure 2, trapezoids represent the content and subset selection gates. During each stage of content and subset selection, the gates receive information regarding content and order of already selected subsets from the instruction ordering RNN. The figure also depicts the self attention layer which computes C_j.

2.2.2 Content Ordering and Instruction Generation

We use a GRU recurrent neural network (Chung et al., 2014) for content ordering. The RNN is initialized with a zero vector before the network activity begins and it takes the selected content of ICD codes as input during each time step. At time-step t,

$$H_t = \mathbf{GRU}(I_t, H_{t-1}), \tag{2}$$

where I_t is the mean vector of $icd_j^{"}$s computed using the Equation 1. The instruction decoder RNN (also a GRU) is initialized with H_t to generate the instruction at the current time step of content ordering RNN. During each time step of decoding, the decoder RNN attends over the current set of $icd_j^{"}$ to generate the sequence of words $w_{t'}$ in the instruction as formulated below.

$$\beta_{t',k} = \exp(h_{t'}^T \mathbf{W_c} icd_k^{"})$$
$$c_{t'} = \sum \beta_{t',k} icd_k^{"}$$
$$h_{t'}^{'} = \tanh(\mathbf{W_h}[h_{t'}, c_{t'}])$$

where $h_{t'}$ is the hidden state of the decoder at time-step $t^{'}$. The probability distribution over the output vocabulary for generating the word $w_{t'}$ at time step $t^{'}$ of the instruction generating decoder is computed as below,

$$P(w_{t'}|w_{<t'}, \{icd_i^{"}, .., icd_n^{"}\}) = \text{softmax}(\mathbf{W_o} h_{t'}^{'} + b_o) \tag{3}$$

The portion of Figure 2 marked in blue, represents the set of computations detailed in the current section.

2.2.3 Beam Search with Redundancy Penalization

In our approach, instructions are decoded one after another, corresponding to each hidden state of the content ordering RNN, as generating one single long text sequence could lead to intractability. In addition, we include a penalization factor for reducing redundancy in the cost function $\mathcal{C}$ of beam search:

$$\mathcal{C} = -\sum_{i=1}^{t} \log P_i + \lambda * J_{sim}(q_t, Q_I) \tag{4}$$

where q_t is the set of words currently chosen into the instruction under generation, Q_I represents the set of words in the previously generated instructions, J_{sim} computes Jaccard's Similarity(Hamers et al., 1989) and λ is a constant tuned to obtain a maximum BLEU score on development set.

Model	Accuracy
SVM	73 %
Logistic Regression	91 %

Table 1: Instruction vs. non-instruction classification results

3 Experiments

In this section, empirical evaluation is conducted to quantify the accuracy of our model regarding text generation, content ordering, correctness of grammar and informativeness.

3.1 Corpus: ICD Codes to Discharge Instructions[2]

Our dataset consisting of admission ICD codes and their corresponding discharge instructions is derived from MIMIC-III[3]. MIMIC-III is a database containing clinical information regarding patients, admission details, lab tests and medical notes (Ew et al., 2016). For each patient admitted to the hospital, there is a recorded set of ICD codes assigned for billing purposes to specify the diagnoses and clinical procedures related to the patient's admission. We assigned unique IDs for diagnoses and procedure codes and did not distinguish between them. Patients receive a list of discharge instructions written by clinical staff before they return home. These discharge instructions are embedded in larger documents called discharge reports.

We trained statistical models (SVM and logistic regression) to classify instruction sentences (e.g. commands) from non-instruction sentences in the discharge reports using TF-IDF vectors computed from sentences in the discharge reports as features. The dataset is split into the training set, developmental set and test set, each comprising 2,000, 500 and 500 sentences, respectively. The accuracy of this binary classification is in Table 1. The logistic regression classifier was used to construct the corpus that maps ICD codes to discharge instructions, resulting in 18,900 of input output pairs for training and 900 each for testing and development. Manual verification was performed to ensure data quality for testing and developmental sets. Following customary data post-processing protocols, named entities such as numerical values and person names were replaced by place holder tags such as [num] and [person name].

[2]https://github.com/littoncode/mimic_scripts3

[3]https://mimic.physionet.org/gettingstarted/overview/

3.2 Implementation Details

We chose different base models and different settings of our model to evaluate the efficiency of our method. In this section we describe each of the models in detail.

3.2.1 Seq2Seq

In this setting we use sequence to sequence model with the attention mechanism (Bahdanau et al., 2014). The set of input ICD codes are arranged as a single sequence in random order and the model generates the entire set of instructions as a single sequence. The learning algorithm minimizes negative log-likelihood. Beam search decoding with beam size of 9 is used during testing.

3.2.2 Set2SingleSeq

In this setup, the model is a variant of (Zhou et al., 2017), where input is a set and content selection is conducted by learning correlations between the input items. This is also a variant of our method, where the output is treated as a single sequence instead of a set of instructions. Any icd_j in the input set of ICD codes are updated to selected content $icd_j^{'}$ using the content selection gate. The content correlation vector C_j for each icd_j is computed below:

$$\alpha_{j,k} = \exp(icd_j^T \mathbf{W_a} icd_k)$$
$$C_j = \sum_{k \neq j} \alpha_{j,k} icd_k$$

The content gate value computation and subsequent content selection is conducted as follows:

$$gc_j = \text{sigmoid}(\mathbf{W_{gc}}[icd_j; C_j])$$
$$icd_j^{'} = gc_j \odot icd_j$$

The entire sequence of instructions is decoded using the computed set of $icd_j^{'}$. For this purpose, the decoder is initialized with the mean vector of $icd_j^{'}$ and the decoder attends over all the set of $icd_j^{'}$ during each time-step of decoding. The learning algorithm minimizes negative log likelihood for optimizing parameters. Beam search decoding with beam size of 9 is used during testing.

3.2.3 Set2MultipleSeq

This setting represents the method explained in the Section 2.2. In this setup, instructions are generated one after another in the learned order. The optimized size for ICD embedding, content ordering RNN, and decoder RNN is 600. Sizes of network parameter matrices $\mathbf{W_a}$, $\mathbf{W_{gc}}$, $\mathbf{W_{gs}}$, $\mathbf{W_c}$ and $\mathbf{W_o}$ are adjusted accordingly. The learning algorithm minimizes negative log likelihood for optimizing parameters. Error is averaged for all instruction decoder time steps for the set of instructions generated. Beam search with beam size of 9 is used during testing.

3.2.4 Set2MultipleSeq+Opt

This setup is an enhanced version of *Set2MultipleSeq* (Section 3.2.3), but during beam search decoding redundancy is penalized (see Section 2.2.3). The value of λ in Equation 4 is set to 2.7 to obtain maximum content coverage on the development set.

4 Evaluation Study

4.1 Evaluation I: Content Generation

Instruction generation is quantitatively estimated by measuring the N-gram match of the generated content with ground-truth in test set using the BLEU metric (Papineni et al., 2002). We did not set any stopping criteria for the number of instructions to be generated for simplicity sake. However, we generated as many number of instructions in the corresponding test record. The results are shown in Table 2. *Set2MultipleSeq* yields a better score than *Seq2Seq*, indicating the effectiveness of content selection and self attention for modeling the correlation between ICD codes. The proposed *Set2MultipleSeq* approach yields even more improvement, validating that the introduction of subset selection and content selection helped in defining the content and context of an instruction during generation. This resulted in more accurate generation of instructions one after the other. When penalizing redundancy during decoding (*Set2MultipleSeq+Opt*), it explicitly forces the instruction decoder to generate an instruction with novel content during each stage of the content ordering RNN, thereby reducing errors caused by repeated content.

Method	Content Generation	
	BLEU	**BLEU-2**
Seq2Seq	6.3	22.30
Set2SingleSeq	7.6	27.00
Set2MultipleSeq	9.6	33.30
Set2MultipleSeq+Opt	**12.6**	**36.30**

Table 2: Evaluation I: Content Generation.

Method	Content Ordering		
	P	**R**	**F**
Seq2Seq	0.06	0.07	0.07
Set2SingleSeq	0.07	0.09	0.08
Set2MultipleSeq	0.18	0.19	0.18
Set2MultipleSeq+Opt	**0.22**	**0.23**	**0.22**

Table 3: Evaluation II: Content Ordering: P is Precision, R is Recall and F is F- Measure. The values are approximated to two decimal points

4.2 Evaluation II: Content Ordering

For evaluating content ordering, we use a variant of the metric used by Gong et al. (2016). In this scheme, we compare the order of words in generated sequence of instructions with ground-truth: We take the set of all order bigrams S_o from the generated sequence of instructions where the first word in each bigram is from a preceding instruction and the second word is from any instruction succeeding it in the sequence. Precision and recall is computed as:

$$Precision = \frac{|S_o \bigcap S_0'|}{|S_0|}$$

$$Recall = \frac{|S_o \bigcap S_0'|}{|S_0'|}$$

S_0' is the order bigrams in the corresponding human written set of instructions in testset. Thus the metric scores the ordering better when the right content is generated in the right order. F-Measure is computed as the harmonic mean of precision and recall. The results in Table 3 show that there is a considerable improvement in the quality of ordering with after introducing content ordering mechanism in the neural architecture. Better ordering score for $Set2MultipleSeq + Opt$ is obvious as redundant content adversely affect instruction ordering.

Method	%
Set2SingleSeq	20
Set2MultipleSeq	77
Ambigous	3

Table 4: Evaluation III: Gramaticality: The values represent percentage of times instructions generated by the model is chosen by the human evaluator.

Method	%
Set2SingleSeq	30
Set2MultipleSeq	63
Ambigous	7

Table 5: Evaluation III: Informativeness: The values represent percentage of times instructions generated by the model is chosen by a human evaluator.

Method	%
Set2MultipleSeq	38
Set2MultipleSeq+opt	61
Ambigous	1

Table 6: Evaluation III: Informativeness: The values represent percentage of times instructions generated by the model is chosen

4.3 Evaluation III: Human Analysis

We conduct human evaluations to measure if the generated instructions are grammatically correct and how informative they are. Four human evaluators who are post graduate students in linguistics were recruited and each given 30 sets of instructions from the testset.

4.3.1 Grammaticality

For each of the 30 instructions chosen for human evaluation, the evaluators were given a number of choices, each generated from a different model, and asked to choose the option that was the most grammatically correct. For each question, the instructions from the different models were shown to the evaluators in a random order to avoid any kind of bias. Instructions generated by *Set2SingleSeq* and *Set2MultipleSeq*. *Set2MultipleSeq+Opt* is excluded as it is an optimization for avoiding redundancy without any direct influence on the grammatical quality of text generated by neural models.

Results aggregated across evaluators through majority voting are shown in Table 4. The results show that incorporating neural network components for content selection and ordering helps in defining the context of an instruction and generating the right content in the right form. *Set2MultipleSeq* generates one instruction at a time, while *Set2SingleSeq* generates the entire set of instructions as one sequence. Grammaticality is shown to improve *Set2MultipleSeq* when only one (shorter) instruction needs to be decoded at a time .

4.3.2 Informativeness

We also conducted human experiments to evaluate the informativeness of generated instructions. The evaluators were asked to read the reference set of instructions prior to examining the instructions generated by competing methods. They were then asked to choose the model that generated instructions retaining the most information from the reference instructions. If two models present instructions that are equally informative, then the model providing richer information density is chosen.

We conducted separate evaluations for two pairs of competing systems (*Set2SingleSeq*, *Set2MultipleSeq*) and (*Set2SingleSeq*, *Set2MultipleSeq+Opt*). The results are shown in Tables 5 and 6 respectively. Results shown in Table 5 explains that incorporating neural components for subset selection and content ordering helps in improving informative instruction generation. We observe that conducting content selection multiple times during each time step through content ordering RNN helps in generating a discrete set of instructions (*Set2MultipleSeq*). Table 6 shows that penalizing redundancy during beam search decoding reduces noise and helps in generating instructions with rich information density. Inter-evaluator agreement for the entire set of human evaluation is reasonably high: Cohen's kappa coefficient is 0.79.

5 Discussions

5.1 Qualitative Comparison Across Models

Table 7 shows an example of instructions generated by the different approaches we investigated. Though *Set2SingleSeq* generates relevant instructions, it repeats the same content, misses out on instructions, and is less grammatically correct.

Set2MultipleSeq shows qualitative improvement over *Set2SingleSeq*, where the repeated content is largely reduced and the grammaticality is improved. However, for *Set2MultipleSeq*, there is

Method	Generated Content
Set2SingleSeq	shower daily including washing incisions gently with mild soap , no baths or swimming until cleared by surgeon. shower daily and pat incisions dry no lotions , creams or powders on incisions no driving for [num]. shower daily , let water flow over wounds , pat dry with a towel towel , do no
Set2MultipleSeq	**1)** shower daily and pat incisions dry no lotions , creams or powders on incisions, no baths or swimming until cleared by surgeon. **2)** no lifting greater then [num] pounds for [num] weeks , do not drive or operate heavy machinery while taking any narcotic pain medication such as percocet **3)** call for any fever , redness or drainage from wounds or weight gain more than [num] pounds **4)** call your doctor for any fever , redness or drainage from wounds
Set2MultipleSeq+Opt	**1)** shower daily and pat incisions dry no lotions , creams or powders on incisions, no baths or swimming until cleared by surgeon. **2)** no lifting greater then [num] pounds for [num] weeks , do not drive or operate heavy machinery while taking any narcotic pain medication such as percocet **3)** call for any fever , redness or drainage from wounds or weight gain more than [num] pounds **4)** follow up with your primary care doctor , dr [person name] , in the next week as well
Nurse Written Instructions	**1)** shower, no baths, no lotions,creams or powders to incisions **2)** no lifting more than [num] pounds for [num] weeks from surgery **3)** do not drive or drink alcohol while taking narcotic pain medications **4)** call with fever, redness or drainage from incision or weight gain more than [num] pounds in one day or five in one week.

Table 7: Examples from Evaluation II: Content Ordering. Person names and numeric values are specified as [person name] and [num] respectively. Repetitive content is color coded in blue for *Set2SingleSeq* and in purple for *Set2MultipleSeq*.

still repeated content in the 3rd and 4th instructions.

Set2MultipleSeq+Opt brings more tractability in the neural model by penalizing redundancy and preventing noisy content generation. In the example shown, *Set2MultipleSeq+Opt* improves over *Set2MultipleSeq*, as the content in the 3rd instruction is no longer repeated in the 4th instruction.

5.2 Variability in Groundtruth References

Variability stemming from communication style differences across clinicians could potentially be one reason why the overall scores for content generation (BLEU scores) and content ordering (F-measure) are on the low end. We observed that for the same set of ICD codes, different clinicians have different writing styles, even when the underlying content of the instructions are the same. Consider the following examples:

(a) shower daily and pat incisions dry no lotions , creams or powders on incisions, no baths or swimming until cleared by surgeon

(b) shower, no baths, no lotions,creams or powders to incisions

In (a), a more detailed way of representing the in-

struction is presented, while in (b) the same information is represented in a more succinct manner.

If a clinician decides to be more detailed in writing the instructions, the clinician might also include specific information such as medication details: "Please be sure to take aspirin and plavix everyday as directed." Such inconsistency of how information such as medication details are specified in the instructions can potentially lead the models to generate noisy content.

6 Related Prior Work

6.1 Natural Language Generation

Here we define natural language generation as the task of generating text from textual input or non-linguistic input (e.g. graphs, data records). Previous work on text generation span across various input types and methods. Initial models used learned rules (Reiter et al., 2003, 2005) and manually engineered templates (Kukich, 1983; McRoy et al., 2000; McKeown, 1985) for constructing the output text. There is also work using automatic means for generating templates (Angeli et al., 2010; Howald et al., 2013).

Such template approaches could be inherently less efficient in modeling semantics when compared to neural networks, especially if there is ample training data. Initial neural models for text generation were first motivated by machine translation (Bahdanau et al., 2014; Cho et al., 2014; Srivastava et al., 2014). However, such approaches are less suitable in comprehending the semantics of structures that are more complex than short sequences and in generating longer sequences (Paulus et al., 2017; Wiseman et al., 2018). Recent advancement in text to text generation has primarily focused on news document summarization (Cheng and Lapata, 2016; Nallapati et al., 2017; See et al., 2017; Paulus et al., 2017).

A portion of related work focuses on generating text from an input graph (Koncel-Kedziorski et al., 2019; Song et al., 2018, 2017): Graphs with labeled edges (e.g., knowledge graphs or abstract meaning representation) are used to generate a direct description of the information characterized in the input graph.

Text generation methods from data records has been investigated for different datasets such as wikipedia infobox (Lebret et al., 2016; Liu et al., 2018; Sha et al., 2018; Perez-Beltrachini and Lapata, 2018), weather predictions (Mei et al., 2016) or sport game summaries (Wiseman et al., 2017). There is a subset of work on text generation from data records which relies on content planning for generating a single sentence (Liu et al., 2018), while some other researchers generate multi-sentence outputs from structured data inputs (Puduppully et al.; Jhamtani et al., 2018). Puduppully et al. generated basketball game summaries by explicitly learning the order in which information should be mentioned in the output by generating an intermediate content plan.

6.2 Text Generation in Healthcare

A lot of NLP work in the medical domain has focused on information extraction (see (Wang et al., 2018) for a review). Driven by healthcare application demands, recently there is emerging interest in areas such as automatic ICD code assignment (Zhang et al., 2017; Scheurwegs et al., 2017; Mullenbach et al., 2018), risk prediction (Ma et al., 2018), and dialogue comprehension (Liu et al., 2019a). One of the earliest work on text generation in the medical domain dates back to more than two decades ago, where interactive systems generate summaries of the patient status for physicians and tailored explanations of clinical information for individual patients (Buchanan et al., 1995). Recent advances of neural modeling has rekindled interest in text generation. Text generation has had more presence in the media and journalism domain, ranging from image captioning (Kinghorn et al., 2018) to news reports on weather (Reiter et al., 2005) and sports (Wiseman et al., 2017). For the medical domain, interest in neural text generation has been springing up, with a primary focus on document summarization (Moradi and Ghadiri, 2018). For interested readers, Pauws et al. (2019) provides a recent review on how data-to-text technology can be applied in healthcare settings.

7 Conclusion

We proposed a neural architecture that learns to generate content in a specific order (discharge instructions for patients) without explicit specifications of the relations between input entities (ICD codes representing diagnoses and procedures) and how the input entities relate to the output. Our approach yields encouragingly better results in comparison with strong baselines. We further improved performance by penalizing redundancy during decoding.

8 Acknowledgement

We thank Jiewen Wu for pointing us to the MIMIC dataset and sharing his experience with us. We thank Ai Ti Aw, Mien Ho, Pavitra Krishnaswamy, and Zhengyuan Liu for their insightful discussions. We are also grateful for the encouraging and constructive feedback from the anonymous reviewers.

Research efforts were supported by funding for Digital Health from the Science and Engineering Research Council (SERC Project No: A1818g0044), A*STAR, Singapore. In addition, this work was conducted using resources and infrastructure provided by the Human Language Technology unit at I2R.

References

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

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Bruce G Buchanan, Johanna D Moore, Diana E Forsythe, Giuseppe Carenini, Stellan Ohlsson, and Gordon Banks. 1995. An intelligent interactive system for delivering individualized information to patients. *Artificial intelligence in medicine*, 7(2):117–154.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494.

Kyunghyun Cho, B van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. In *Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation (SSST-8), 2014*.

Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014*.

Elizabeth Clark, Yangfeng Ji, and Noah A Smith. 2018. Neural text generation in stories using entity representations as context. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2250–2260.

Alistair Ew, Tom J Pollard, Lu Shen, H Lehman Li-wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3:160035.

Jingjing Gong, Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. 2016. End-to-end neural sentence ordering using pointer network. *arXiv preprint arXiv:1611.04953*.

Lieve Hamers et al. 1989. Similarity measures in scientometric research: The jaccard index versus salton's cosine formula. *Information Processing and Management*, 25(3):315–18.

Blake Howald, Ravikumar Kondadadi, and Frank Schilder. 2013. Domain adaptable semantic clustering in statistical nlg. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)–Long Papers*, pages 143–154.

Harsh Jhamtani, Varun Gangal, Eduard Hovy, Graham Neubig, and Taylor Berg-Kirkpatrick. 2018. Learning to generate move-by-move commentary for chess games from large-scale social forum data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1661–1671.

Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. 2016. Globally coherent text generation with neural checklist models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 329–339.

Philip Kinghorn, Li Zhang, and Ling Shao. 2018. A region-based image caption generator with refined descriptions. *Neurocomputing*, 272:416–424.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text generation from knowledge graphs with graph transformers. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2284–2293.

Karen Kukich. 1983. Design of a knowledge-based report generator. In *Proceedings of the 21st annual meeting on Association for Computational Linguistics*, pages 145–150. Association for Computational Linguistics.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213.

Tianyu Liu, Kexiang Wang, Lei Sha, Baobao Chang, and Zhifang Sui. 2018. Table-to-text generation by structure-aware seq2seq learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Zhengyuan Liu, Hazel Lim, Nur Farah Ain Suhaimi, Shao Chuen Tong, Sharon Ong, Angela Ng, Sheldon Lee, Michael R Macdonald, Savitha Ramasamy, Pavitra Krishnaswamy, et al. 2019a. Fast prototyping a dialogue comprehension system for nurse-patient conversations on symptom monitoring. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers)*, pages 24–31.

Zhibin Liu, Zheng-Yu Niu, Hua Wu, and Haifeng Wang. 2019b. Knowledge aware conversation generation with reasoning on augmented graph. *arXiv preprint arXiv:1903.10245*.

Fenglong Ma, Jing Gao, Qiuling Suo, Quanzeng You, Jing Zhou, and Aidong Zhang. 2018. Risk prediction on electronic health records with prior medical knowledge. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1910–1919. ACM.

Kathleen R McKeown. 1985. Discourse strategies for generating natural-language text. *Artificial Intelligence*, 27(1):1–41.

Susan W McRoy, Songsak Channarukul, and Syed S Ali. 2000. Yag: A template-based generator for real-time systems. In *Proceedings of the first international conference on Natural language generation-Volume 14*, pages 264–267. Association for Computational Linguistics.

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

Milad Moradi and Nasser Ghadiri. 2018. Different approaches for identifying important concepts in probabilistic biomedical text summarization. *Artificial intelligence in medicine*, 84:101–116.

James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable prediction of medical codes from clinical text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1101–1111.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Steffen Pauws, Albert Gatt, Emiel Krahmer, and Ehud Reiter. 2019. Making effective use of healthcare data using data-to-text technology. In *Data Science for Healthcare*, pages 119–145. Springer.

Laura Perez-Beltrachini and Mirella Lapata. 2018. Bootstrapping generators from noisy data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1516–1527.

Ratish Puduppully, Li Dong, and Mirella Lapata. Data-to-text generation with content selection and planning.

Ehud Reiter, Roma Robertson, and Liesl M Osman. 2003. Lessons from a failure: Generating tailored smoking cessation letters. *Artificial Intelligence*, 144(1-2):41–58.

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

Elyne Scheurwegs, Boris Cule, Kim Luyckx, Léon Luyten, and Walter Daelemans. 2017. Selecting relevant features from the electronic health record for clinical code prediction. *Journal of biomedical informatics*, 74:92–103.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083.

Lei Sha, Lili Mou, Tianyu Liu, Pascal Poupart, Sujian Li, Baobao Chang, and Zhifang Sui. 2018. Order-planning neural text generation from structured data. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Linfeng Song, Xiaochang Peng, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2017. Amr-to-text generation with synchronous node replacement grammar. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 7–13.

Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2018. A graph-to-sequence model for amr-to-text generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), pages 1616–1626.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Yanshan Wang, Liwei Wang, Majid Rastegar-Mojarad, Sungrim Moon, Feichen Shen, Naveed Afzal, Sijia Liu, Yuqun Zeng, Saeed Mehrabi, Sunghwan Sohn, et al. 2018. Clinical information extraction applications: a literature review. *Journal of biomedical informatics*, 77:34–49.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2018. Learning neural templates for text generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3174–3187.

Danchen Zhang, Daqing He, Sanqiang Zhao, and Lei Li. 2017. Enhancing automatic icd-9-cm code assignment for medical texts with pubmed. In *BioNLP 2017*, pages 263–271.

He Zhao, Chong Feng, Zhunchen Luo, and Changhai Tian. 2018. Entity set expansion from twitter. In *Proceedings of the 2018 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 155–162. ACM.

Qingyu Zhou, Nan Yang, Furu Wei, and Ming Zhou. 2017. Selective encoding for abstractive sentence summarization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1095–1104.